DISEÑO E IMPLEMENTACIÓN DE LA BASE DE DATOS DE CONTROL ENERGÉTICO.

Trabajo de Fin de Carrera

Estudiante: EDUARD MONZONIS HIERRO.

emonzonis@uoc.edu

Titulación: Ingeniería Técnica Informática de Gestión.

Consultor: Ismael Pérez Laguna

Fecha: 18 de marzo de 2012

Dedicatoria y agradecimientos.

Este camino lo inicie hace algunos años con mucha ilusión, no ha estado nada fácil su singladura, valorar y gestionar mi tiempo ha estado un hecho constante durante esta etapa de mi vida, la autoexigencia, el auditarme día a día para adquirir nuevos conocimientos, han sin lugar a dudas quienes han marcado un antes y un después en mi persona.

Quiero expresar mis agradecimientos algunos grandes amigos que la Universitat Oberta de Catalunya me ha brindado la oportunidad de conocer. Así mismo su soporte, entusiasmo y constancia trasmitidos han sido pilares importantes para poder conseguir este objetivo común a todos. La inquietud para adquirir conocimientos ha estado presente y un punto esencial en esta trayectoria. Han sido muchas las horas dedicadas y muchos festivos sacrificados amigo Albert y Joan, pero ha valido la pena el conjugar todo, amistad y objetivos, donde los lazos de amistad y en muchos casos sean materializado en el ámbito profesional, siempre estarán patentes entre nosotros.

Agradezco a mi Consultor Ismael Pérez López por su labor, la cual, me ha guiado en la última etapa de mi carrera y sin lugar a dudas expresar el gran honor que he tenido de haber podido interactuar con un excepcional equipo de Consultores, destacando mi Tutor Quím Trullas por su permanente feedback, durante mis estudios en esta Universidad.

Un especial agradecimiento a mi familia, dedico este trabajo a mi esposa y a mis dos hijos, que en muchos casos ha implicado sacrificar eventos importantes o posponer compromisos, sin vosotros no hubiera llegado hasta aquí, aunque algunas veces poníais en tela de juicio tal esfuerzo exigido. A veces un cambio de hábitos se propaga, los hijos siguen y hacen lo que ven de sus progenitores, Marc y Núria seguid así os falta poco para sentir esta sensación de haber culminado un merecido objetivo.

Resumen.

La Comunidad Económica Europea, dentro de la partida presupuestaria ha destinado en adquirir una aplicación informática para controlar el uso de la energía.

Dentro del marco de colaboración con la Universitat Oberta de Catalunya, le ha sido encargado el diseño de la Base de Datos Relacional, como primera fase del desarrollo de la aplicación de gestión y de que en una segunda fase se desarrollara el software que reunirá las especificaciones requeridas para el uso de dicha aplicación en el plan de sistemas de la información de la Comunidad Económica Europea.

INDICE.

ÍNDICE ILUSTRACIONES.

1 Introducción.

1.1.1 Justificación del TFC y contexto de desarrollo.

La Comunidad Económica Europea, dentro de los presupuestos destinados a controlar el uso de la energía, a decidido abrir un concurso público para recibir propuestas sobre el diseño de la Base de Datos que servirá de almacén en un futuro de toda la información crítica con el fin de ser explotada por una aplicación la cual permitirá la generación de datos estadísticos sobre la energía.

Dentro del marco de colaboración de dicha institución europea con la Universitat Oberta de Catalunya (UOC). La facultad de Ingeniería Técnica Informática de Gestión de dicha universidad se centrará en el diseño y desarrollo de la arquitectura de la Base de Datos Relacional como primera fase de la aplicación de gestión que se desarrollará en su segunda fase del plan de Sistemas de Información de la Comunidad Económica Europea.

1.1.2 Planteamiento inicial Requerimientos Funcionales del Sistema.

A nivel general la Base de datos su finalidad esencial es de guardar de forma eficiente y racionalizada aquella información crítica susceptible de ser tratada para llevar a término los análisis estadísticos correspondientes, con el fin de poder mejorar el consumo energético de la euro zona, que en consecuencia el Staff asignado a estos estudios pueda diseñar los planes estratégicos en materia de energía y adoptar decisiones coherentes, racionalizando de forma optimizada los recursos energéticos de la Comunidad Económica Europea.

El sistema diseñado debe de permitir almacenar los datos de los Clientes con las características definidas para su identificación y el tipo de usuario que es (persona física o empresa), los cuales están asociados un contador de consumo de energía, dicho contador físicamente depende de una o más de una central de distribución, la cual verifica constantemente el flujo energético y en caso de caída del sistema de flujo procedente del nodo superior, realice sistemáticamente la continuidad del flujo energético. Dichas subestaciones y estaciones dentro de la red, tienen unas características definidas de identificación nivel máximo soportado de energía a distribuir.

El modelo del sistema de base de datos, tiene que ser capaz de almacenar los datos referentes a los nodos primarios o origen, es decir los datos referentes a la producción generada por las mismas, su identificación geográfica. De estas centrales deberá de almacenarse un histórico de la energía producida. Así mismo el ratio máximo de producción suministrada a la red, junto con la fecha de las revisiones técnicas efectuadas durante su visa útil.

Debido a la necesidad de controlar el origen del consumo energético, con el fin de tomar las decisiones coherentes y racionalizadas. Es necesario el controlar la tipología de energía que dichas centrales generadoras producen y distribuyen. En este sentido debe de controlarse en el sistema de Base de Datos Relacional, el nivel de contaminantes que emiten con el fin de poder tomar las medidas oportunas o equilibradas en los sistemas de producción que se hallan inmersas.

La red de distribución esta configurada como una malla arbolada, la cual en la misma existe una configuración donde el nodo primario es la central de producción y los diferentes nodos dependientes son los subestaciones de distribución, cada una de ellas identificadas de forma unívoca dentro del sistema de suministro .

Los contadores que los clientes tienen en el sistema de base de datos almacenará las lecturas correspondientes y el tipo de medio en el que sea ha efectuado, telemáticamente o presencial.

La gestión del sistema de datos almacenados tiene tres estados alta, baja y modificación que afectan a los contadores de distribución, la líneas de intercomunicación, las centrales de distribución, junto con las diferentes características asociadas como el consumo, la potencia generada y soportada, la procedencia tipológica de energía suministrada junto con el nivel de emisiones contaminantes.

Por tanto debe de cumplir a nivel de usuario el poder gestionar datos críticos y auxiliares para el almacenamiento de los mismos:

- ✓ Gestión de Clientes.
- ✓ Gestión de Poblaciones.
- ✓ Gestión Poblaciones a Provincias.
- ✓ Gestión de Provincias a Naciones.
- ✓ Gestión de Naciones.
- ✓ Gestión Vías Públicas.
- ✓ Gestión Tipología de Energías.
- ✓ Gestión de Centrales de Distribución.
- ✓ Gestión de Subcentrales o Subestaciones.
- ✓ Gestión de Contadores.
- ✓ Gestión de Fabricantes.
- ✓ Gestión de Modelos.
- ✓ Gestión de Líneas de Comunicación.
- ✓ Gestión de Asignación de Centrales a la Línea de Comunicación.
- ✓ Gestión de Asignación de Subcentrales a la línea de Comunicación.
- ✓ Gestión de Asignación de Contadores a la Línea de Comunicación.
- ✓ Gestión de Lecturas de Consumos de los Clientes.
- ✓ Gestión Estadísticas.

La gestión anterior debe ser extensible para el control de datos de los clientes, que tienen definidos sus perfiles por el contrato de suministro energético.

1.1.3 Planteamiento final esperado y Requerimientos de Implementación del sistema.

El sistema de almacenamiento de datos generado, debe de ser capaz de dar respuesta a las peticiones de información establecidas por los usuarios de la misma.

Deberá de poderse consultar la ubicación geográfica en una fecha o en un rango de fechas, de los contadores en que el consumo mensual haya superado el 80% del consumo medio de todos los contadores de dicha ubicación geográfica, siempre dentro la temporalidad expresada en el rango parametrizada. Dicha información deberá de seguir un orden de mayor a menor consumo energético realizado.

El consultar las 10 centrales de distribución con mayor producción de energía, con los datos identificativos de las mismas y características. Siguiendo una ordenación en dicha extracción de datos de menor producción a mayor producción.

El sistema debe de permitir extraerse la información referente a las 10 líneas con más sobrecarga con relación a su máxima capacidad admitida. Proyectando su identificación, la potencia soportada dentro de la capacidad máxima adscrita por sus especificaciones técnicas, junto con la capacidad de flujo energético que puede ampliarse teniendo presente la dependencia del nodo superior o central de producción direccionada. Debiendo de mostrar este conjunto de datos de forma coherente y ordenada según su valor absoluto de forma descendente.

En el conjunto de datos almacenados, el sistema deberá de permitir poder hacerse la extracción de todos los clientes que dispongan de un contador en estado de alta y que dicho estado sea extensible a la central en el cual se halla conectado, la línea asignada y la ruta que le corresponda de conectividad de central de producción. Mostrando los datos identificativos únicos que definen al cliente como elemento unívoco dentro de los clientes.

Se quiere conocer el consumo efectuado por los contadores dependientes de la central de producción junto con el ratio de producción generado por dicha central, esta información será accesible en un rango de tiempo o intervalo parametrizado.

Para llevar a termino un control de lecturas, el sistema a de ser capaz de proyectar por que sistema de lectura sea obtenido el consumo durante un rango te tiempo.

Debido a la posibilidad de renovar los contadores a los clientes por motivos de eficiencia tecnológica, se precisa poder extraer un listado de los contadores con ciertos años de antigüedad.

Debido a que la finalidad funcional del sistema es obtener las estadísticas necesarias para tomar las decisiones necesarias para que el usuario final pueda optar en estrategias o tomar decisiones concretas en la optimización y eficiencia energética.

Se precisa de las estadísticas, por lo cual deberá de dar respuesta a las siguientes peticiones:

De una central de producción deberá de conocerse el consumo de los contadores que dependan de la misma.

El valor medio de consumo durante un año que ha soportado una línea de comunicación a una central adscrita. Con las restricciones de que dicho consumo depende del contador que

alimentan dicha red y de que un contador puede estar conectada a más de una central con el fin de garantizar el suministro y poder observar si la línea esta bien dimensionada.

Poder controlar que línea ha estado afectada por una desmesurada sobrecarga.

Conocer el conjunto de líneas el porcentaje de las mismas que hayan superado el 50% de energía consumida.

La existencia o posibilidad de que hayan centrales de producción que producen menos de un 30% de energía, es una de las informaciones criticas que deberá de poderse extraer.

Del histórico deberá de poderse extraer los 10 contadores qua hayan realizado el mayor consumo energético.

De los clientes deberá poderse extraer el consumo medio realizado en un rango o intervalos de tiempo.

1.1.4 Objetivos del Proyecto.

1.1.4.1 Generales.

El objetivo principal de este proyecto ha sido consolidar y ampliar los conocimientos adquiridos a lo largo de la carrera de Ingeniería Técnica Informática de Gestión en el área de las bases de datos relacionales, dando énfasis las asignaturas de Bases de Datos I, Bases de Datos II, Sistema de Gestión de Bases de Datos y Ingeniería de Programas. Adicionalmente las asignaturas de Organización y Administración de Empresas junto con asignatura Técnicas del Desarrollo del Software e Informática Aplicada a la Gestión los conocimientos adquiridos en estas asignaturas complementan en la planificación del TFC.

En la planificación del proyecto sea utilizado como herramienta de temporalidad el Microsoft Project, para el diseño de la estructura del diagrama de GANTT.

Durante las fases del Ciclo de Vida de Software, el análisis previo y de requerimientos se ha utilizado el Microsoft Visio 2007, el MagicDraw UML12.0. Como herramientas del diseño gráfico del modelaje Entidad Relación.

Des de la perspectiva fundamental anteriormente expresada como objetivo principal convive la utilización del lenguaje declarativo propio de las bases de datos relacionales PL/SQL y SQL Dinámic, utilizando como herramienta comercial de base de datos ORACLE versión 10. Esto ha permitido efectuar ensayos y construir la arquitectura de la base de datos mediante un SGB de gran impacto en el mercado por su robustez, portabilidad, versatilidad y eficiencia.

1.1.4.2 Concretos.

Los objetivos concretos o específicos del TFC, es tratar todo el proceso de implementación del sistema de Base de Datos Relacional, siguiendo estrictamente las diferentes fases en su desarrollo.

- ✓ Análisis.
- ✓ Diseño.
- ✓ Implementación.
- ✓ Pruebas.

Por lo tanto los objetivos concretos serán ampliar los conocimientos siguientes:

- ✓ Análisis de un Sistema de Base de Datos Relacional.
- ✓ Diseño de una Base de Datos Relacional según los requerimientos exigidos.
- ✓ Utilización de lenguaje SQL, propio de las Base de Datos Relacionales.
- ✓ Control de errores.
- ✓ Tratamientos de las excepciones.
- ✓ Gestión de ficheros.
- ✓ Creación del banco de pruebas.

Durante el transcurso del período del desarrollo del TFC, concretamente del semestre, se irán haciendo entregas parciales del trabajo, que progresivamente irá madurando hasta alcanzar el objetivo final del TFC, el producto finalizado.

Las entregas están definidas en tres PACS que culminará en el objetivo final de una memoria, el producto resultante y una presentación mediante la aplicación PowerPoint del proyecto realizado.

La primera entrega PAC1 se entregara el esqueleto de la memoria que contendrá el Plan de Trabajo que describe el conjunto de tareas a realizar para el desarrollo del proyecto y la Temporalidad o Planificación de las mismas.

La segunda entrega PAC2 se hará una entrega del conjunto de fases que componen el Ciclo de Vida de Software, es decir análisis y diseño.

La tercera entrega PAC3 la entrega consistirá en las fases Implementación y el Banco de Pruebas del producto final.

De la lectura detallada del proyecto a desarrollar se ha realizado una aproximación estimada y no definitiva de la estructura que deberá tener la memoria. Esta estructuración por motivos de evolución progresiva en el desarrollo del TFC no queda excluida la posibilidad de ser modificada, es decir ser ampliada.

HITO	FECHA PLANIFICADA	FECHA DE ENTREGA
Inicio del TFC	**29/02/2012**	**10/06/2012**
Reunión Trobada presencial Académica	**04/03/2012**	**04/03/2012**
Proceso desarrollo PAC 1 y entrega	**05/03/2012**	**18/03/2012**
Proceso desarrollo PAC 2 y entrega	**19/03/2012**	**15/04/2012**
Proceso desarrollo PAC 3 y entrega	**16/04/2012**	**20/05/2012**
Entrega de la Memoria Final TFC	**21/05/2012**	**10/06/2012**

Ilustración 1 Hitos

1.1.4.3 Enfoque y Metodología.

El primer paso para desarrollar este proyecto ha sido entre etapas analizando de forma real los requisitos que el enunciado ha propuesto y siguiendo una temporalidad ajustada par alcanzar los hitos establecidos fluyendo de los mismos los principios de calidad, estandarización, reutilización y portabilidad.

Para poder alcanzar el objetivo midiendo la carga de trabajo, se ha hecho una estimación del volumen de trabajo a realizar y una planificación ajustada a la realidad. Como punto fundamental se ha decidido llevar a término el Ciclo de Vida de Software clásico en cascada sin despreciar el sistema cíclico en espiral de refinamiento en cada una de sus fases.

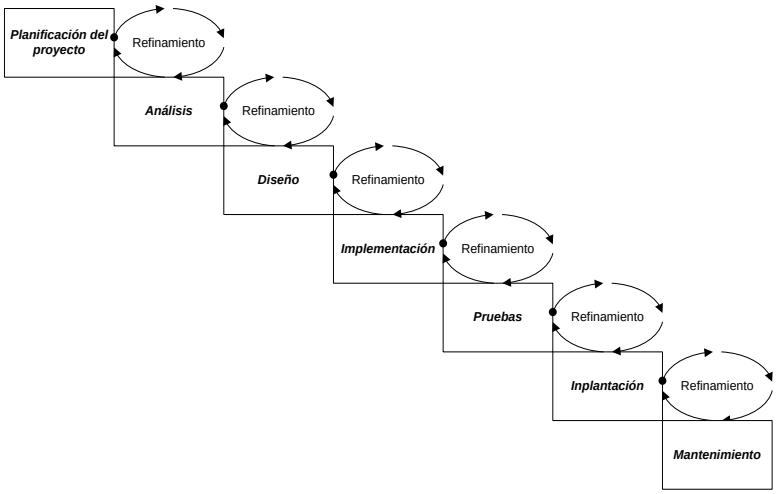


Ilustración 2 Ciclos de Vida software

La Planificación sea basada en los métodos adquiridos en la asignatura Técnicas del Desarrollo del Software e Informática Aplicada a la Gestión.

1.2 Planificación.

HITO	FECHA PLANIFICADA	FECHA DE ENTREGA
Inicio del TFC	29/02/2012	10/06/2012
Reunión Trobada presencial Académica	04/03/2012	04/03/2012
Proceso desarrollo PAC 1 y entrega	05/03/2012	18/03/2012
Proceso desarrollo PAC 2 y entrega	19/03/2012	15/04/2012
Proceso desarrollo PAC 3 y entrega	16/04/2012	20/05/2012
Entrega de la Memoria Final TFC	21/05/2012	10/06/2012

1.2.1 Tareas identificadas.

1. Descripción del TFC.

Descripción detallada del proyecto a desarrollar.

2. Objetivos del Proyecto.

Definir los objetivos a nivel General y Concretos del objeto del proyecto.

3. Tareas Identificadas.

Conjunto de tareas que componen el Proyecto, las fases que se llevarán a termino.

4. Planificación de la Temporalidad.

Planteamiento basado en la temporalidad de cada una de las tareas o fases del desarrollo del Proyecto siguiendo estrictamente el Ciclo de Vida de Software.

5. Ciclo de Vida del software de la Base de Datos.

Conjunto de fases para alcanzar el objetivo final, generar un proyecto estable y robusto. El proceso puede ser cíclico por el principio de refinamiento.

5.1. Análisis de Requerimientos.

Define de forma detallada las diferentes funcionalidades que deberá el proyector dar respuesta a las necesidades del cliente final.

5.1.1. Estudio del Área de Negocio.

Identificación de las principales áreas de negocio en que la aplicación debe de dar cobertura junto con el análisis de los usuarios y grupos que deberán de utilizar el sistema durante su explotación.

5.1.2. Estudio y análisis de la documentación inicial.

Estudio centrado al usuario y repercusión en la aplicación de gestión que se desarrollará en el futuro en la segunda fase.

5.1.3. Analítica del entorno actual, nivel operativo.

Captura coherente de la información sobre el uso que se piensa dar a la base de datos.

5.1.4. Estudio nivel de uso y tratamiento de datos.

Análisis de la usabilidad de la información, valoración de las frecuencias de peticiones de datos y flujos de información que interactúan en el sistema a nivel del entorno actual. Análisis de la consistencia y redundancias previsibles.

5.1.5. Resultados y valoraciones del análisis de requerimientos.

Descripción en lenguaje natural, primeras aproximaciones algorítmicas de las peticiones y resultados esperados.

5.1.6. Relación de requerimientos detectados.

Especificación de forma jerarquizada de la información a tratar analizando dependencias y relaciones existentes del mundo real.

5.1.7. Diagrama de flujo de datos.

Detección y protocolización de los datos a tratar y deberá almacenar, semántica y ortografía del usuario. Problemática de la diversificación de nomenclaturas de datos a introducir y consultar.

5.1.7.1. Casos de Uso.

Análisis de actores y escenarios acorde con la funcionalidad exigida por el cliente.

5.1.8. Requerimientos Funcionales.

Informe final de las necesidades de los usuarios.

5.1.9. Requerimientos no Funcionales.

Requerimientos del entorno del software.

5.1.10. Valoración de costes presupuesto previo.

Estudio de los costes del desarrollo del proyecto y presupuesto orientativo no cerrado.

5.2. Diseño del Sistema.

Diseño estructural de la Base de Datos, arquitectura de la Base de Datos.

5.2.1. Diseño del modelo Conceptual.

Descripción esquematizada de la Base de Datos, utilizando para dicho fin un modelo de datos conceptual independiente al esquema conceptual de la BD i del SGB. Sin tener en cuenta los aspectos tecnológicos.

5.2.1.1. Modelo Entidad/Relación.

Representación estructurada de la Base Datos.

5.2.1.2. Descripción de las Entidades y los Atributos.

Identificación de las entidades de su rol con la descripción detallada del conjunto de atributos las caracteriza.

5.3. Diseño Lógico.

Transformación del modelo Entidad/Relación a un modelo relacional.

5.3.1. Localización e identificación del las Claves y su rol.

Del modelo Entidad/Relación se identificará el conjunto de claves primarias y foranas que definen la arquitectura, siguiendo el diseño de la fase anterior.

5.3.2. Refinamiento y depuración del Modelo Entidad/Relación.

Bajo el principio de cascada se procede a una revisión exhaustiva de las posibles optimizaciones posibles siguiendo escrupulosamente la normalización de las Bases de Datos Relacionales.

5.4. Diseño Físico.

Materialización del modelo lógico teniendo presente la implementación que deberá de llevarse acabo en la siguiente fase. Conjunto de decisiones referente a las tablas, restricciones de los atributos que las componen, índices, roles y permisibilidades de los Grupos de Usuarios y Usuarios.

5.4.1. Restricciones de Integridad Referencial.

Definición de las reglas de negocio que debe de cumplir la Base de Datos respetando escrupulosamente las establecidas en el modelo Entidad/Relación como los Roles de los Grupos de Usuarios y Usuarios definidos e la fase del Análisis de Requerimientos (actores y escenarios) .

5.5. Implementación.

Generación del código SQL para la creación de la Base de Datos, definición de usuarios y roles, relaciones, secuencias, disparadores, carga externa de datos, proyecciones de las peticiones exigidas en la explotación del proyecto.

5.6. Plan de Contingencias.

Estudio detallado del nivel de riesgos de recuperación en caso de perdida de datos críticos del sistema d base de datos.

Control de excepciones en datos no congruentes derivados de peticiones realizadas por los usuarios que explotan los datos críticos del sistema.

Redacción del protocolo nemotécnico a que deberá de seguir los usuarios en la introducción y manipulación de la información de la Base de Datos.

5.7. Banco de Pruebas.

Banco de pruebas originado por la carga de datos externos al sistema. Comprobaciones de incoherencias y tiempo de latencia entre petición y respuesta realizando métricas evaluativas del comportamiento temporal entre petición y respuesta obtenida.

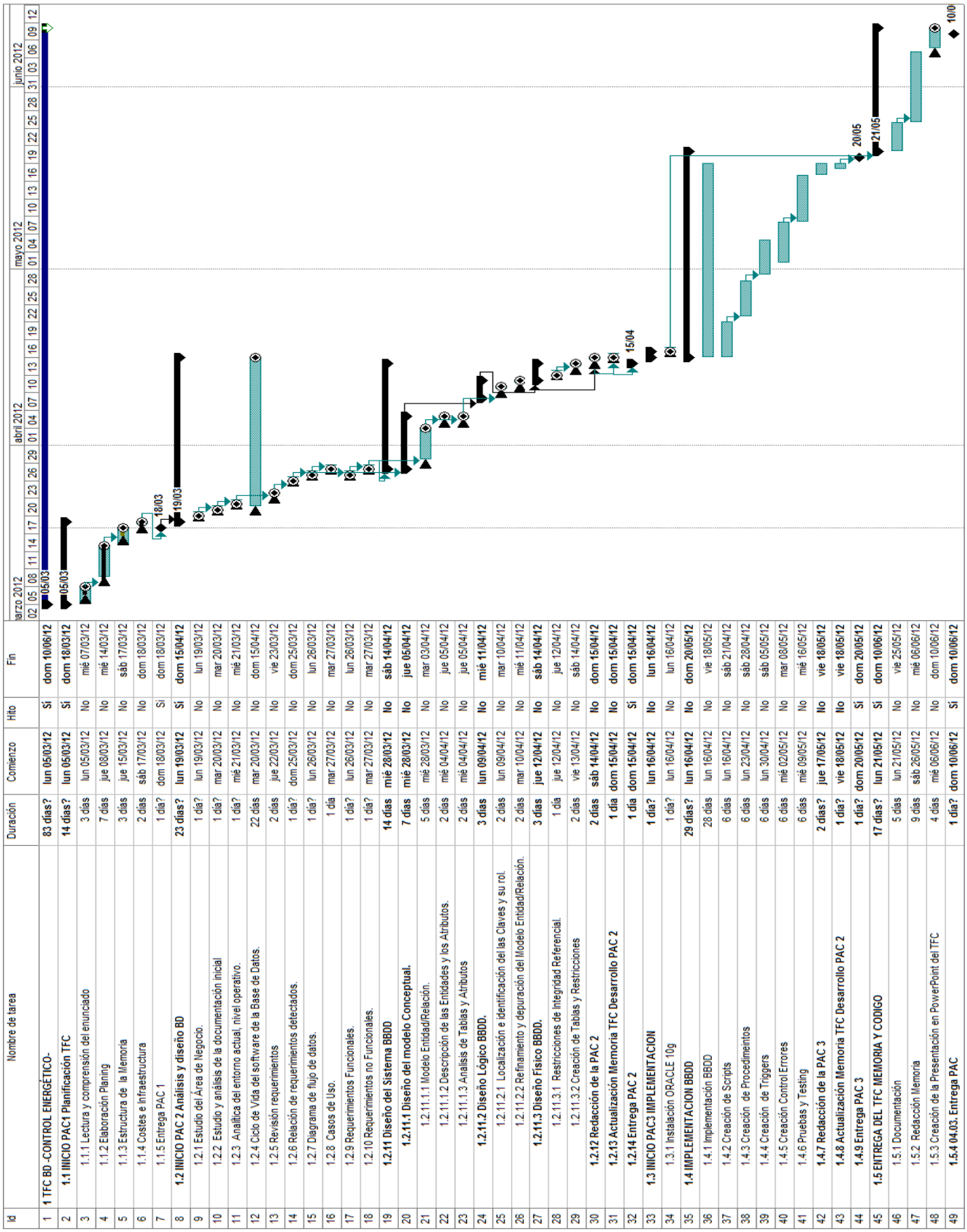
6. Redacción de la memoria.

Redacción de la memoria del proyecto de forma detallada de los hitos obtenidos durante el proceso de desarrollo y la entrega del producto final

7. Presentación Virtual.

Presentación del producto en diapositivas de PowerPoint del producto terminado con las fases que intervienen en el mismo. Su enfoque esta destinado al cliente final para su evaluación.

1.3 DIAGRAMA DE GANTT



Id	Nombre de tarea	Duración	Comienzo	Hito	Fin
1	**1 TFC BD -CONTROL ENERGÉTICO-**	**83 días?**	**lun 05/03/12**	**Sí**	**dom 10/06/12**
2	**1.1 INICIO PAC1 Planificación TFC**	**14 días?**	**lun 05/03/12**	**Sí**	**dom 18/03/12**
3	1.1.1 Lectura y comprensión del enunciado	3 días	lun 05/03/12	No	mié 07/03/12
4	1.1.2 Elaboración Planing	7 días	jue 08/03/12	No	mié 14/03/12
5	1.1.3 Estructura de la Memoria	3 días	jue 15/03/12	No	sáb 17/03/12
6	1.1.4 Costes e Infraestructura	2 días	sáb 17/03/12	No	dom 18/03/12
7	1.1.5 Entrega PAC 1	1 día?	dom 18/03/12	Sí	dom 18/03/12
8	**1.2 INICIO PAC 2 Análisis y diseño BD**	**23 días?**	**lun 19/03/12**	**Sí**	**dom 15/04/12**
9	1.2.1 Estudio del Área de Negocio.	1 día?	lun 19/03/12	No	lun 19/03/12
10	1.2.2 Estudio y análisis de la documentación inicial	1 día?	mar 20/03/12	No	mar 20/03/12
11	1.2.3 Analítica del entorno actual, nivel operativo.	1 día?	mié 21/03/12	No	mié 21/03/12
12	1.2.4 Ciclo de Vida del software de la Base de Datos.	22 días	mar 20/03/12	No	dom 15/04/12
13	1.2.5 Revisión requerimientos	2 días	jue 22/03/12	No	vie 23/03/12
14	1.2.6 Relación de requerimientos detectados.	1 día?	dom 25/03/12	No	dom 25/03/12
15	1.2.7 Diagrama de flujo de datos.	1 día?	lun 26/03/12	No	lun 26/03/12
16	1.2.8 Casos de Uso.	1 día	mar 27/03/12	No	mar 27/03/12
17	1.2.9 Requerimientos Funcionales.	1 día?	lun 26/03/12	No	lun 26/03/12
18	1.2.10 Requerimientos no Funcionales.	1 día?	mar 27/03/12	No	mar 27/03/12
19	**1.2.11 Diseño del Sistema BBDD**	**14 días**	**mié 28/03/12**	**No**	**sáb 14/04/12**
20	**1.2.11.1 Diseño del modelo Conceptual.**	**7 días**	**mié 28/03/12**	**No**	**jue 05/04/12**
21	1.2.11.1.1 Modelo Entidad/Relación.	5 días	mié 28/03/12	No	mar 03/04/12
22	1.2.11.1.2 Descripción de las Entidades y los Atributos.	2 días	mié 04/04/12	No	jue 05/04/12
23	1.2.11.1.3 Analisis de Tablas y Atributos	2 días	mié 04/04/12	No	jue 05/04/12
24	**1.2.11.2 Diseño Lógico BBDD.**	**3 días**	**lun 09/04/12**	**No**	**mié 11/04/12**
25	1.2.11.2.1 Localización e identificación del las Claves y su rol.	2 días	lun 09/04/12	No	mar 10/04/12
26	1.2.11.2.2 Refinamiento y depuración del Modelo Entidad/Relación.	2 días	mar 10/04/12	No	mié 11/04/12
27	**1.2.11.3 Diseño Físico BBDD.**	**3 días**	**jue 12/04/12**	**No**	**sáb 14/04/12**
28	1.2.11.3.1 Restricciones de Integridad Referencial.	1 día	jue 12/04/12	No	jue 12/04/12
29	1.2.11.3.2 Creación de Tablas y Restricciones	2 días	vie 13/04/12	No	sáb 14/04/12
30	**1.2.12 Redacción de la PAC 2**	**2 días**	**sáb 14/04/12**	**No**	**dom 15/04/12**
31	**1.2.13 Actualización Memoria TFC Desarrollo PAC 2**	**1 día**	**dom 15/04/12**	**No**	**dom 15/04/12**
32	**1.2.14 Entrega PAC 2**	**1 día**	**dom 15/04/12**	**Sí**	**dom 15/04/12**
33	**1.3 INICIO PAC3 IMPLEMENTACION**	**1 día?**	**lun 16/04/12**	**No**	**lun 16/04/12**
34	1.3.1 Instalación ORACLE 10g	1 día?	lun 16/04/12	No	lun 16/04/12
35	**1.4 IMPLEMENTACION BBDD**	**29 días?**	**lun 16/04/12**	**No**	**dom 20/05/12**
36	1.4.1 Implementación BBDD	28 días	lun 16/04/12	No	vie 18/05/12
37	1.4.2 Creación de Scripts	6 días	lun 16/04/12	No	sáb 21/04/12
38	1.4.3 Creación de Procedimeintos	6 días	lun 23/04/12	No	sáb 28/04/12
39	1.4.4 Creación de Triggers	6 días	lun 30/04/12	No	sáb 05/05/12
40	1.4.5 Creación Control Errores	6 días	mié 02/05/12	No	mar 08/05/12
41	1.4.6 Pruebas y Testing	6 días	mié 09/05/12	No	mié 16/05/12
42	**1.4.7 Redacción de la PAC 3**	**2 días?**	**jue 17/05/12**	**No**	**vie 18/05/12**
43	**1.4.8 Actualización Memoria TFC Desarrollo PAC 2**	**1 día?**	**vie 18/05/12**	**No**	**vie 18/05/12**
44	**1.4.9 Entrega PAC 3**	**1 día?**	**dom 20/05/12**	**Sí**	**dom 20/05/12**
45	**1.5 ENTREGA DEL TFC MEMORIA Y CODIGO**	**17 días?**	**lun 21/05/12**	**Sí**	**dom 10/06/12**
46	1.5.1 Documentación	5 días	lun 21/05/12	No	vie 25/05/12
47	1.5.2 Redacción Memoria	9 días	sáb 26/05/12	No	mié 06/06/12
48	1.5.3 Creación de la Presentación en PowerPoint del TFC	4 días	mié 06/06/12	No	dom 10/06/12
49	**1.5.4 04.03. Entrega PAC**	**1 día?**	**dom 10/06/12**	**Sí**	**dom 10/06/12**

Ilustración 3 Diagrama de Gantt

1.4 Recursos e Infraestructura, Costes y Presupuesto Inicial.

En este apartado se detallan los recursos necesarios para llevar a término el proyecto, los clasifico en tres puntos esenciales y un estudio de costes para generar el presupuesto orientativo.

1.4.1.1 Hardware.

Para el desarrollo de este proyecto se utilizaran dos ordenadores de un PC de sobremesa y portátil, los dos compatibles.

- Equipo de sobremesa HP 7700 Intel Pentium IV de 3,4 GHz 3 Gb de memoria RAM disco duro de 1 Tb, Sistema Operativo principal Windows XP Professional Service Pack 3 y secundario LINUX Ubuntu. Tiene instalada un Máquina Virtual VirtualVPC2007 para hace pruebas con oreos sistemas Operativos y Base de Datos ProstgrSQL, Informix y MySql.

- Ordenador Portátil BENQ AMD Turion(tm) 64 Mobile Tecnology MT-32 de 3,4 GHz, 3 Gb de memoria RAM disco duro de 500 Gb. Con el Sistema Operativo Windows Professional Service Pack 3.

1.4.1.2 Software.

Para el desarrollo del proyecto de la Base de Datos Relacional:

- El sistema Gestor de Base de Datos ORACLE 10g Express Edition.

- Para los Diagramas del Modelo Entidad/Relación los aplicativos VISIO 2007 y el Magicdraw Uml Enterprise v12.

- Lenguaje de desarrollo SQL y PL/SQL.

Para la elaboración de las diferentes entregas de este proyecto:

- Microsoft Office 2007 (WORD, EXCEL, ACCESS y POWERPOINT).

Para la planificación y seguimiento de tareas.

- El Diagrama de GANTT y las temporalidades del proyecto se utiliza la aplicación Microsoft Project 2007.

1.4.2 Recursos Humanos.

En el desarrollo del proyecto intervienen dos expertos:

- Desarrollo del Proyecto.
  - Jefe de Proyectos, Ingeniero Técnico Informática de Gestión.
- Fase de análisis y desarrollo de la Base de Datos Relacional.
  - Analista Funcional.
- Fase de creación de los diferentes procedimientos de almacenamiento y test de pruebas.
  - Analista Programador Experto en Administración de Base de Datos Relacionales en el producto comercial ORACLE.

1.5 Costes.

El estudio de los costes, teniendo presente que el TFC es de 7,5 créditos y que cada crédito son aproximadamente 25 horas de trabajo más un 10% de carga, he considerado la semana laboral de de 5 días, por lo cual la carga de trabajo es de 84 días con un total de 3 horas y 32 minutos por día, es decir un total de 275 horas 56 minutos. El personal interviene según la planificación Jefe de Proyecto con una carga de 46 horas, un Analista Funcional en 100 horas y un Analista Experto en Base de Datos ORACLE con 133 horas. Se ha tenido en cuenta 8 horas de trabajo del administrativo del departamento de desarrollo.

1.5.1.1 Nivel Hardware.

El Hardware, se ha considerado que es una licencia amortizable durante los 365 días y distribuidos los costes del mismo en las siguientes partidas.

HARD	TIPO	PRECIO	UNIDADES	IMPORTE	RATIO
PCS	Hp Pentium	1.000,00 €	1	1.000,00 €	40%
PORTATIL	Benq	1.100,00 €	1	1.100,00 €	44%
CONNECTIVIDAD	RED OPTICA INS uso ADSL	200,00 €	2	400,00 €	16%
TOTAL				2.500,00 €	100%

Ilustración 4 Costes de Hardware

1.5.1.2 Nivel Software utilizado.

El uso del Software para el desarrollo del proyecto sea considerado en 276 horas, para cada una de las aplicaciones que han sido utilizadas durante el desarrollo.

SOFT USADO	HORAS	DIAS	PRECIO	IMPORTE	RATIO	IMPORTE LICENCIAS
OFFICE 2007	276 h.	84	0,02 €/h.	4,53 €	6,24%	144,00 €
USO ORACLE	276 h.	84	0,17 €/h.	47,18 €	65,05%	1.500,00 €
USO GANTT MICROSOFT PROJECT	276 h.	84	0,03 €/h.	8,40 €	11,58%	267,00 €
VISIO2007	276 h.	84	0,02 €/h.	4,72 €	6,50%	150,00 €
Magicdraw Uml Enterprise v12	276 h.	84	0,01 €/h.	2,99 €	4,12%	94,95 €
SISTEMAS OPERATIVOS	276 h.	84	0,02 €/h.	4,72 €	6,50%	150,00 €
TOTAL			0,04 €/h.	72,54 €	100,00%	2.305,95 €

Ilustración 5 Nivel de costes utilizados en el diseño

1.5.1.3 Nivel Gastos de Personal.

El personal que interviene en el desarrollo sea tenido presente que en el departamento de desarrollo hay un administrativo de soporte durante el período del desarrollo. El departamento de desarrollo interviene de forma directa un Analista Funcional que lleva el peso del inicio del proyecto hasta la fase del Diseño Físico, un Analista Programador experto en Bases de Datos Relacionales plataforma ORACLE. Dado que los costes administrativos de toda empresa constituye un factor a tener en cuenta a la hora de centrar el margen comercial aplicable. Sea tenido en cuenta la repercusión que tiene el departamento de administración y servicios que tiene la corporación.

CARGO	PERSONAL	PRECIO HORA	HORAS EFECTIVAS	TOTAL	RATIO	DIAS	DIA PERSONA	AÑO
JEFE DE PROYECTO	1 personas	80,00 €/h.	46 h.	3.680,00 €	84,21%	83 dias	44,34 €/dies	16.183,13 €
ANALISTA FUNCIONAL	1 personas	50,00 €/h.	100 h.	5.000,00 €	52,63%	83 dias	60,24 €/dies	21.987,95 €
ANALISTA PROGRAMADOR ADMINISTRADOR BASE DE DATOS ORACLE	1 personas	30,00 €/h.	133 h.	3.990,00 €	31,58%	83 dias	48,07 €/dies	17.546,39 €
ADMINISTRACION I SERVICIOS	1 personas	15,00 €/h.	8 h.	120,00 €	15,79%	75 dias	1,60 €/dies	584,00 €
TOTAL	4 personas	95,00 €/h.	72 h.	12.790,00 €	100,00%	75 dias	170,53 €/dies	62.244,67 €

Ilustración 6 Estudio de Costes de Personal.

1.5.1.4 Nivel Licencias.

Al cliente final se le exige disponer de la Base de Datos multiplataforma ORACLE, por lo cual la instalación y la licencia son contabilizados en los costes.

LICENCIAS	DESCRIPCION	IMPORTE
1296651	ORACLE MULTIPLATAFORMA	7.000,00 €
TOTAL		**7.000,00 €**

Ilustración 7 Nivel coste Licencias.

1.5.1.5 Nivel Costes Estructurales, Margen Comercial y Beneficio Neto previsible.

Para poder dar una fiabilidad en el precio final del producto sea tenido que analizar la estructura de la Corporación a nivel de cetro de costes.

Amortizaciones es el conjunto del inmovilizado material e inmaterial necesario para el funcionamiento de la actividad empresarial, por lo cual para dar un coste del producto debe de ser repercutido.

Los costes estructurales, son el conjunto de costes necesarios fijos que la corporación debe de amortizar cada mes, alquiler, consumo de electricidad, mantenimiento, etc.

Los costes de ámbito general son el conjunto de costes variables que se generan durante un mes en la actividad empresarial, diferentes gastos que abarcan des de publicidad, asesorías, etc.

Los gastos de personal son los generados en la producción de los productos finales que contemplan el ciclo de vida del software.

Financieros, conjunto de gastos generados por la relación contractual existente con las entidades financieras con las que la corporación contempla como acreedores.

En base a todos estos capítulos en su análisis global y anual, se obtiene una previsión de flujo de caja, un margen estimado y el beneficio previsible. El margen estimado se ha considerado el doble del benefició previsto.

GASTOS	IMPORTE	DIAS	IMPORT	COSTE ANUAL	MES	HORAS AÑO	COSTES/HORA	RATIO ESCANDALLO
AMORTIZACIONES	20,00 €	365	0,05 €	7.300,00 €	608,33 €	8.760,00 h.	0,83 €/h.	6,88%
ESTRUCTURALES	40,00 €	365	0,11 €	14.600,00 €	1.216,67 €	8.760,00 h.	1,67 €/h.	13,77%
GENERALES	50,00 €	365	0,14 €	18.250,00 €	1.520,83 €	8.760,00 h.	2,08 €/h.	17,21%
FINANCIEROS	10,00 €	365	0,03 €	3.650,00 €	304,17 €	8.760,00 h.	0,42 €/h.	3,44%
SUBTOTAL				43.800,00 €	TOTAL COSTE GENERAL		5,00 €/h.	
PERSONAL PRODUCCION	170,53 €	365	0,47 €	62.244,67 €	5.187,06 €	8.760,00 h.	7,11 €/h.	58,70%
TOTAL COSTES				106.044,67 €	8.837,06 €	TOTAL	12,11 €/h.	100,00%
						COSTE		20,00%
FACTURACIO PREVISTA ANUAL				530.223,33 €				

COSTES	20%
BENEFICIO	20%
MARGEN	40%

Ilustración 8 Nivel Costes Estructurales.

1.5.1.6 Nivel Coste final aplicado.

Conociéndose el margen, el beneficio bruto (sin impuestos) podemos obtener el margen comercial aplicable a las partidas que intervienen en el presupuesto final y dar un valor final d tarifación para el precio hora durante el desarrollo del proyecto conocidos el valor del coste inicial de los elementos que intervienen durante el desarrollo del proyecto. Obteniendo el valor real del producto antes de aplicarle el impuesto del valor añadido IVA.

DESCRIPCION		IMPORTE NETO	BENEFICIO BRUTO	BENEFICIO TEORICO	MARGEN	MARGEN COMERCIAL	PRECIO
INFRAESTRUCUTRA		2.500,00 €	20%	500,00 €	10,56%	45%	3.625,00 €
DESARROLLO	275,56 h.	12.790,00 €	20%	2.558,00 €	54,04%	45%	18.545,50 €
LICENCIA MULTIPLATAFORMA ORACLE		7.000,00 €	20%	1.400,00 €	29,58%	45%	10.150,00 €
COSTES GENERALES	275,56 h.	1.377,80 €	20%	275,56 €	5,82%	45%	1.997,81 €
TOTALES				4.733,56 €	25,00%	180,00%	34.318,31 €

Ilustración 9 Nivel Coste final aplicado.

1.5.1.7 Presupuesto Inicial Orientativo.

De los análisis de los puntos anteriores podemos presentar un presupuesto aproximado del producto final a desarrollar y una propuesta de presupuesto orientativa y que se acerca a la realidad. La aceptación implica al cliente abonar un 60% al inicio del proyecto y el resto deberá de abonarse a la entrega final del producto.

PRESUPUESTO ORIENTATIVO			
ORACLE MULTIPLATAFORMA			10.150,00 €
DESARROLLO			20.543,31 €
INFRAESTRUCUTRA			3.625,00 €
		BASE IMPONIBLE	**34.318,31 €**
	IVA	18%	**6.177,30 €**
TOTAL			**40.495,61 €**

Ilustración 10 Presupuesto Inicial Orientativo.

1.6 Análisis de Riesgos.

Como en toda actividad, pueden surgir imprevistos, que pueden alterar seriamente el desarrollo de la misma. Hemos de tomar las medias oportunas que minimizar estas situaciones ya que no hay forma de evitarlas. Al menos reduciremos su impacto.

- Nivel técnico. Se tomarán todas las medidas posibles para no perder información; copia de seguridad por cada sesión de trabajo en unidades USB y en disco virtual –dropbox-.
    - La elaboración del documento elaborando versiones correlativas, minimizando el riesgo de pérdida en caso de desastre imprevisto.
    - La política de copia seguida es la de una por cada día de la semana que contendrá las sesiones correspondientes.
- Nivel personal. Salvo enfermedad o accidente, no consideramos especialmente ningún factor añadido, sólo las cargas de trabajo extras que en la actividad diaria, y que puedan tener incidencia.

2 Ciclo de Vida del software de la Base de Datos.

Para poder llevar a término el proyecto de forma satisfactoria y dentro de los estándares recomendados en todo desarrollo de una aplicación, se han realizado las siguientes subdivisiones para extraer el máximo rendimiento de las diferentes fases.

De las fases del ciclo al ser en cascada, solo puede cumplimentarse una fase sin antes haber terminado la anterior.

2.1 Análisis de Requerimientos.

El proyecto como punto esencial es la modelización e implementación de un sistema informático para gestionar el control de energía que se consume en la Comunidad Económica Europea con la finalidad de obtener estadísticas del uso eficiente de los recursos disponibles y utilizados por los usuarios en el ámbito geográfico de la Unión Europea.

El equipo de desarrollo, Analista Funcional y Jefe de Proyecto en esta fase ha estado crucial para tener una visión clara e inequívoca de las necesidades que el Cliente precisa y la arquitectura que debe de desarrollarse.

Esta fase no finalizará hasta que los sucesivos refinamientos y depuraciones den como resultado final la solución a las necesidades que el Cliente ha solicitado. Por tanto no podrá llevarse a termino la fase de implantación hasta que esta fase este totalmente optimizada y reúna las metodologías propuestas de resolución que el proyecto exige. Este hecho se justifica al coexistir muchos aspectos de la solución propuesta que no se pueden determinar de forma explícita en cualquier fase del desarrollo por depender del camino o de las decisiones.

Para lograr estos objetivos como punto final es el diseño de una base de datos relacional que mediante su implementación sea capaz de dar respuesta a las peticiones estadísticas exigidas, poder actualizar e introducir datos que deberán de ser relacionados algebraicamente para obtener la información estadística.

El sistema comercial ORACLE de base de datos relacional, se diseñara el conjunto de procedimientos que permitirán insertar, modificar, actualizar y consultar, junto con el modulo estadístico que mantendrá actualizada la información crítica.

2.1.1 Estudio del Área de Negocio.

Del análisis del que debe diseñarse y para que debe ser usado extraemos el área de negocio que justifica el proyecto. Respuestas que son la base para tener la visión claramente definida del como construir el sistema de Base de Datos Relacional.

Como área de negoció es gestionar datos críticos de los usuarios relacionados con el consumo generado, detectando la tipología de energía consumida localizando los puntos de flujo energético que recibe los contadores asignados a cada uno de los clientes conectados a la red de subestaciones o centrales de distribución conectadas a una central generadora de energía.

2.1.2 Estudio y análisis de la documentación inicial.

El usuario del sistema su accesibilidad debe de dar respuesta a la funcionalidad a las peticiones basadas en insertar datos críticos, modificar datos críticos y extraer datos relacionados o combinados de los mismos.

Por tanto en tiempo de respuesta y la simplicidad en obtener la información es el punto fundamental para que la aplicación que en la segunda fase se construya, disponga de la información ordenada, siempre bajo el prisma de fiabilidad en los datos manipulados y obtenidos.

Por lo cual se detecta como puntos base los Clientes, Contadores, Contratos, Líneas de Conexión, Centrales de Distribución, Centrales de Producción, Tipos Energías, Control de Inspecciones, Lecturas de Consumos, Localización física (geográfica) de cada una de las entidades, Histórico de datos y Estadísticas. Siendo estas últimas diversificadas según las necesidades establecidas por el enunciado inicial del proyecto.

Las Estadísticas extraídas están supeditadas a la conectividad existente entre los contadores y las líneas de comunicación donde el flujo del suministro se halla en situación de capacidad soportada. Las líneas de conexión tienen un máximo de nivel de capacidad, por lo cual el exceder a dicha capacidad implica una sobrecarga de red generándose la caída de red entre la central de distribución y un contador.

La cuantificación de energía producida, aunque el enunciado no lo refleja, debería de ser dada por la central de producción. El hecho de que a priori no es conocida, damos como valor de producción al consumo generado por los contadores que afectan al nivel de flujo que la central

de distribución suministra y absorbe de la central de producción. Teniendo como restricción el rango máximo de tolerancia que puede soportar de capacidad de energía.

Por tanto el conocer el consumo de energía será un factor decisivo para controlar no solo las posibles caídas de red sino que las asignaciones de los contadores a las líneas de comunicación con las centrales de distribución los cuales deberán de poder tener acceso a esta central de distribución de forma binaria, es decir poder conectarse a dos centrales, por motivos de persistencia de suministro energético anulando la posibilidad de caída de flujo eléctrico.

2.1.3 Analítica del entorno actual, nivel operativo.

La Consultaría con la que el Cliente ha estado dialogándose eferente a las especificaciones de funcionalidad y accesibilidad, sea podido obtener una visión clara del uso de la información y la finalidad de la misma.

Como punto esencial es el conocer el consumo máximo de energía por contador, por central de distribución, por central de producción y obtener los 10 contadores con más consumo en una período de tiempo.

Para llevar acabo el objetivo, se precisa de que se efectúen una lecturas sean on-line o emitidas de forma presencial por el cliente.

Sena detectado restricciones a nivel de control de asignación de contadores debido a las tolerancias de flujo.

Como condición de asignación de contador, debe de existir un contrato que religa contractualmente al cliente. Todo contrato debe de tener una modalidad de potencia contratada que definirá que tipo de modelo de contador suministrado por el fabricante deberá de ser conectado en la línea de comunicación que como restricción será la se no sobre pasar el nivel de tolerancia máxima de capacidad soportada por la misma línea y la central de distribución.

2.1.4 Estudio nivel de uso y tratamiento de datos.

Durante el tratamiento de datos, uno de los elementos fundamentales que pueden dar errores o extracción de información incoherente. Es la introducción de datos de los usuarios.

El diseño del protocolo de datos a modificar y a introducir debe de pactarse antes de la utilización del sistema.

Por tanto, el tratamiento de ubicación geográfica de los componentes que actúan en el proceso, clientes, contadores, centrales de producción y distribución. Debe de homogeneizarse, las entidades como elementos que reflejan hechos reales de la vida real obligan a los usuarios el adoptar una misma tipología semántica y ortográfica en su uso.

El incumplimiento de dicho protocolo puede generar inconsistencias, redundancias, duplicidades e incluso omisiones en los datos obtenidos durante las extracciones.

Como puntos críticos que deben de ser protocolizados son los nombres de las localidades, países y direcciones, donde las abreviaturas de los datos introducidos pueden dar resultados imprevisibles y erróneos.

En sistema llevará un control depurativo de excepciones que pueden generarse durante la manipulación de los datos o en la introducción de los mismos. Aún así, el margen de errores siguen coexistiendo, estos no son causados por la optimización generada durante el tratamiento automatizado del sistema, tales ocurrencias posibles que puedan surgir serán fruto de no seguir por parte del usuario el protocolo de trabajo que deberá de seguir de forma escrupulosa por motivos de eficiencia de usabilidad.

2.1.5 Resultados y valoraciones del análisis de requerimientos.

El esquema que modeliza a conceptuar el mundo real de forma analítica según las necesidades requeridas de usabilidad son deberán contener relaciones basadas en acciones específicas. Estas contendrán los atributos observados en la vida real susceptibles de ser manipulados por el usuario.

Por tanto se detectan las acciones siguientes:

- ✓ Insertar.

  En este proceso deberá de restringirse las duplicidades de información
  - Poblaciones.
  - Provincias
  - Países.
  - Vías
  - Ubicaciones

- Clientes.
- Fabricantes.
- Modelos.
- Contadores.
- Contratos.
- Centrales.
- Inspecciones.
- Tipos de Centrales.
- Lecturas.
- Tipo de Lecturas.

✓ Modificar.

Conjunto de datos susceptibles de ser modificados, las entidades afectadas de fecha de alta en estas funcionalidades quedarán restringidas.

- Poblaciones.
- Provincias
- Países.
- Ubicaciones.
- Clientes.
- Fabricantes.
- Modelos.
- Contadores.
- Contratos.
- Centrales.
- Inspecciones.
- Tipos de Centrales.
- Lecturas.
- Tipo de Lecturas.

✓ Consultar.

- Clientes.
- Clientes según tipo de Cliente.
- Ubicación de un Cliente.
- Estado de un Cliente.
- Contratos de Clientes.
- Estado de un Contrato.
- Contadores.
- Contadores conectados a Líneas de Comunicación.
- Contadores disponibles y año de fabricación.
- Modelos de Contadores, capacidad tolerable y año de fabricación.
- Contadores conectados a las Líneas de Comunicación.

- Líneas Comunicación.
- Líneas disponibles de comunicación según sobrecarga de potencia afectadas por los Contadores conectados.
- Disponibilidad de Línea de Comunicación para conectar un contador según proximidad de Ubicación.
- Fabricantes.
- Fabricantes según un Modelo de Contador y año de fabricación.
- Centrales
- Centrales según capacidad y Tipo de Central.
- Centrales según estado y fecha inspección.
- Central de Distribución conectadas a Centrales de Producción, estado y Línea de Comunicación.
- Lecturas de Contadores según período o rango de fecha.
- Consumo

✓ Actualizar.

- Poblaciones.
- Provincias
- Países.
- Ubicaciones.
- Clientes.
- Fabricantes.
- Modelos.
- Contadores.
- Contratos.
- Centrales.
- Inspecciones.
- Tipos de Centrales.
- Lecturas.
- Tipo de Lecturas.
- Histórico de Lecturas.

Descripción en lenguaje natural, primeras aproximaciones algorítmicas de las peticiones y resultados esperados.

2.1.6 Relación de requerimientos detectados.

Especificación de forma jerarquizada de la información a tratar analizando dependencias y relaciones existentes del mundo real.

2.1.7 Diagrama de flujo de datos.

Detección y protocolización de los datos a tratar y deberá almacenar, semántica y ortografía del usuario. Problemática de la diversificación de nomenclaturas de datos a introducir y consultar.

2.1.7.1 Casos de Uso.

Análisis de actores y escenarios acorde con la funcionalidad exigida por el cliente. El rol existente de los Usuarios viene acotado por los permisos subyacentes que el Administrador de la Base de Datos da al Usuario.

NIVELES ROL	FUNCIONES
Administrador.	✓ CREATE DATABASE ✓ CREATE SCHEMA ✓ CREATE TABLE ✓ CRETAE VIEW ✓ CREATE TRIGGERS ✓ CRETA PROCEDURE ✓ CREATE USUARIO ✓ SELECT ✓ INSERT ✓ UPDATE ✓ ALTER ✓ DELETE ✓ DROP DATABASE ✓ DROP TABLE ✓ DROP TRIGGERS ✓ DROP SCHEMA ✓ DROP PROCEDURE
Usuarios	PERMISOS ✓ Gestión Países. ✓ Gestión Provincias. ✓ Gestión Localidades. ✓ Gestión Vías Públicas. ✓ Gestión Clientes ✓ Gestión Fabricantes ✓ Gestión Modelos ✓ Gestión Contadores. ✓ Gestión Central. ✓ Gestión Ubicación ✓ Gestión Línea Comunicación ✓ Gestión Lecturas ✓ Gestión Modulo Estadísticas RESTRICCIONES. ✓ CREATE DATABASE ✓ CREATE SCHEMA ✓ CREATE TABLE ✓ CRETAE VIEW ✓ CREATE TRIGGERS ✓ CRETA PROCEDURE ✓ CREATE USUARIO ✓ SELECT ✓ INSERT ✓ UPDATE ✓ ALTER ✓ DELETE ✓ DROP DATABASE ✓ DROP TABLE ✓ DROP TRIGGERS ✓ DROP SCHEMA ✓ DROP PROCEDURE

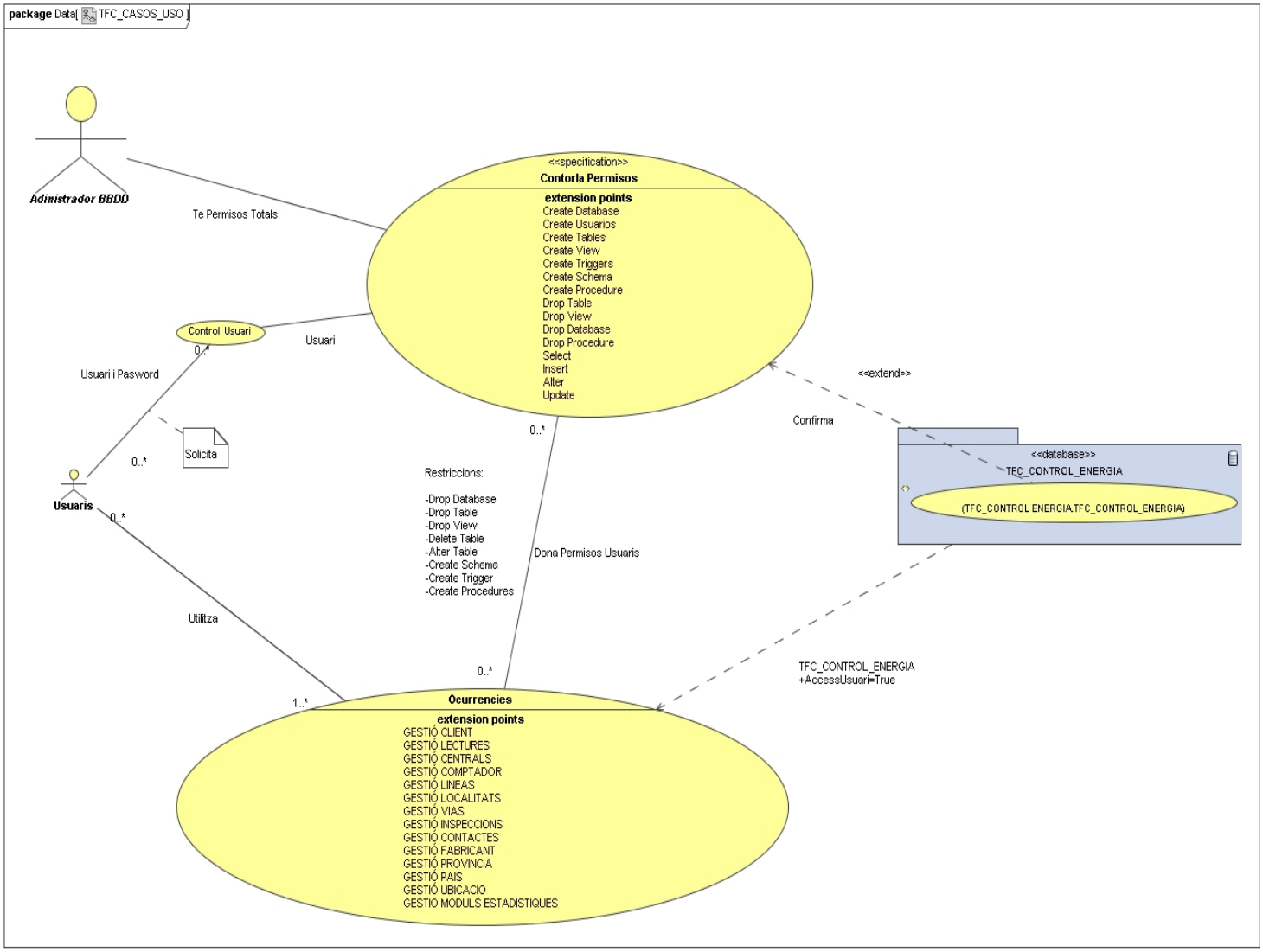


Ilustración 11 Casos de Uso

2.1.8 Requerimientos Funcionales.

La funcionalidad del sistema tiene que ser capaz de gestionar el conjunto de datos críticos del sistema i propagar los mismos a las entidades interrelacionadas bajo la integridad de referencial interna y las restricciones que cada una de ellas afectará en el uso de dichas funcionalidades. De esto se desprende que la integridad referencial deberá ser en cascada y dado que los datos afectan a históricos no se parametrizada en ningún caso la integridad referencial en cascada de eliminación, lo motivos son la propagación de dicha integridad que puede afectar a las actualizaciones de datos e incluso obtener información inconsistente e incoherente.

- ✓ Gestión de Clientes.

  Deberá de tener las siguientes funcionalidades. Contendrá dependencias directas con *Ubicación*, *Contratos*. De estas dependencias derivará las asignaciones del *Contador* que pertenece a un Contrato.

  - Alta de clientes.
  - Baja de clientes.
  - Modificación de datos de cliente.
  - Introducción de toda la información capaz de identificar de forma física a un cliente como individuo existente en la vida real y como elemento indivisible o atómico.
  - Definir su rol en el mundo real, Empresa o Particular.
  - Consultar un cliente, tipos de cliente y ubicaciones (localidades).

- ✓ Gestión de Poblaciones.

  Deberá de tener las siguientes funcionalidades. Contendrá dependencias directas con *Provincias.*

  - Alta de poblaciones o localidades.
  - Mantenimiento localidades.
  - Asignar una población a un Provincia.
  - Consultar localidades.

- ✓ Gestión Provincias.

  Deberá de tener las siguientes funcionalidades. Contendrá dependencias directas con *País.*

  - Alta de provincias.
  - Mantenimiento Provincias.
  - Asignación de País.
  - Consultar Provincias.

- ✓ Gestión de País.

  Deberá de tener las siguientes funcionalidades.

- Alta de País.
- Mantenimiento País.
- Consultar País.

✓ Gestión Ubicaciones.

Deberá de tener las siguientes funcionalidades que identificarán de forma geográfica la dirección concreta dónde se localiza físicamente un elemento susceptible de ser identificado.

Tendrá dependencia de esta gestión los *Contadores*, *las Centrales Producción, las Centrales de Distribución*, los *Clientes* y los *Fabricantes.*

- Alta de ubicación.
- Mantenimiento ubicación.
- Consulta de ubicación.

✓ Gestión Vías Públicas.

Deberá de contener las abreviaciones de las vía públicas.

Tendrá dependencias directas con *Ubicación* ya que en la gestión de ubicaciones deberá de insertarse el tipo de vía pública que tiene la dirección física concreta que pertenece a una *Población*.

- Alta de nombre nemotécnico.
- Mantenimiento del nombre nemotécnico.
- Consultar nombre nemotécnico.

✓ Gestión Tipología de Energías.

Deberá de facilitar el tipo de energía que interviene en el suministro y producción de la misma. Por lo cual deberá de definirse los componentes de cada tipo de energía y los factores cuantitativos que las genera.

Por motivos de reutilización es esencial el poder tener la usabilidad siguiente.

- Alta del tipo de energía.
- Mantenimiento del tipo de energía.

- Consultar tipo de energía.

✓ Gestión de Centrales.

Deberá de identificar de forma unívoca el tipo de central físicamente y el rol que tiene dentro del sistema de suministro de la misma.

Contendrá dependencias directas con *Ubicación*, la *Tipología de Energía* e *Inspecciones*.

- Alta de central.
- Mantenimiento del tipo de central.
- Consultar Centrales.

Como puntos fundamentales que deberá de tener dicha gestión para expandir jerárquicamente el rol que debe de tener dentro de la estructura del modelo basado en el mundo real, será de asignar de forma recursiva el tipo de rol excluyente de ser de Distribución o de Producción, identificar la capacidad máxima de suministrar o producir energía la dicha central, el estado de disponibilidad (alta o baja) que se encuentre, la fecha de la Inspección que tiene dependencias directas con *Inspecciones* y las fechas de la alta de dicha central y si procede la fecha de la ultima modificación.

✓ Gestión de Fabricantes.

Los fabricantes, su rol en el sistema es de identificar que tipo de modelo de contador son productores y suministradores en el sistema de control energético. Es un ente físico al igual que lo es Clientes. Por tanto sus dependencias directas están supeditadas a *Ubicación* con las herencias que esta conlleva, como derivadas por el rol que ocupa dentro del sistema las tiene dependencias con *Modelo* que *Contador* tiene como componente.

Por tanto deberá de facilitar las siguientes funcionalidades.

- Alta de Fabricante.
- Mantenimiento de Fabricantes.
- Consultar Fabricantes.

Como puntos esenciales a tener en cuenta serán el estado de alta o baja, el día de la modificación y el conjunto de datos que identifican de forma atómica a dicho fabricante basándose en la observancia del mundo real.

- ✓ Gestión de Modelos.

Esta funcionalidad identifica el conjunto de modelos suministrados por los *Fabricantes* y que configuran el parque de *Contadores* disponibles susceptibles de ser conectados a las *Líneas de Comunicación* de la red eléctrica.

Por lo tanto deberá esta funcionalidad ser capaz de insertarse el modelo de contador teniendo dependencias directas con *Fabricante*, reflejando el año de fabricación, su estado (alta o baja) entendiéndose como tales estados si están en mantenimiento por el Fabricante o no.

Por tanto deberá de facilitar las siguientes funcionalidades.

- Alta de Modelo.
- Mantenimiento de Modelo.
- Consultar Modelo.

- ✓ Gestión de Inspecciones.

Contiene la funcionalidad de registrar de forma ordenada el conjunto de inspecciones efectuadas a las *Centrales* en un día determinado por un inspector. Deberá contener la identificación de la central inspeccionada, por tanto tiene una dependencia directa con la *Central* y las observaciones detectadas junto con la validación de estado.

Por tanto deberá de facilitar las siguientes funcionalidades.

- Alta de Inspecciones.
- Mantenimiento de Inspecciones.
- Consultar Inspecciones.

- ✓ Gestión de Contratos.

Contiene la funcionalidad de insertar, actualizar y modificar los contratos de los Clientes, utilizando las relaciones existentes y con las entidades ***Cliente***, ***Contador***, y

Tipo de Cliente. En dicha gestión el numero de contrato que debe de generarse, será automático y no podrá existir duplicidades. Un contrato es de un cliente o de varios clientes por lo cual podrán existir más de un cliente con uno y solo un contrato.

Como restricciones tenemos que los contratos pueden sufrir estados, es decir estar de alta o bien de baja, así mismo es necesario controlar el día que se produce el alta y si coexisten modificaciones controlar el día que se produjo dicha modificación.

Por tanto deberá de facilitar las siguientes funcionalidades.

- Alta de Contratos.
- Mantenimiento de Contratos.
- Consultar Contratos

✓ Gestión de Contadores.

Los contadores tienen dependencias de un modelo y del fabricante, como restricciones tenemos que un contador puede estar en estado de alta o de baja.

La gestión de contadores, por definición de los requisitos del proyecto tiene dos elementos que definen su conectividad y el flujo energético que sustentan, por lo cual disponen de una conexión primaria y una secundaria a la ***Línea de Comunicación***

Por tanto deberá de facilitar las siguientes funcionalidades.

- Alta de Contadores.
- Mantenimiento de Contadores.
- Consultar Contadores

✓ Gestión de Líneas de Comunicación.

Las líneas de comunicación, son una de las columnas vertebrales de la arquitectura, contienen mucha información específica esencial para conocer que centrales de distribución están conectadas y que centrales de distribución reciben suministro se ellas.

Se controla en esta sus estados, alta o baja, por lo cual la disponibilidad efectiva de las mismas en la red energética.

Por tanto deberá de facilitar las siguientes funcionalidades.

- Alta de Líneas.
- Mantenimiento de Líneas.
- Consultar Líneas.

✓ Gestión de Tipos Lectura.

Las lecturas, dependiendo del modelo de contador pueden ser automatizadas por su tipología de dualidad.

Pueden tener la modalidad Presencial o bien la modalidad Telemática. Esta gestión sus entidades que concurren depende su existencia generalizada del modelo de contador y del contador.

Como funcionalidad fundamental en base a sus relaciones de dependencia, deberán de ser capaz este gestor, de indicar en las proyecciones que líneas son de lectura presencial o telemática y que cliente dispone de dicha funcionalidad en el contador asignado.

✓ Gestión de Lecturas.

Es el conjunto de procesos que durante las transacciones de información debe de capturar las lecturas de los contadores conectados a la líneas de comunicación.

Debe de capturar del sistema la fecha en que se produce la lectura y los datos heredados de la entidad Líneas de Comunicación.

Por tanto la funcionalidad que debe de tener este gestos es de poder insertar lecturas en una fecha, es decir hay una periodicidad cronológica en inserciones de la información referente a los consumos realizados por una línea en la dependen contadores y centrales.

Esta entidad podemos calificarla de punto nodal del sistema, ya que de esta podemos hacer el recorrido ascendente y descendente del árbol B+ que toda Base de Datos Relacional se le debe atribuir.

✓ Gestión de Asignación de Centrales de Producción a la Línea de Comunicación.

Las centrales están afectadas de una generalización dual, por un lado el tipo y por otro por la clase de central según sus funcionalidades, si son de Producción.

Sea detectado una recursividad que en cierto modo optimizad la arquitectura de la Base de Datos Relaciona.

De este hecho ya comentado en la gestión de centrales se desprende, el poder conectar una tipología concreta como es la central de producción que solo soportará conectividad con una o dos centrales y teniendo presente de que la sobrecarga que genere puede afectar a que un contador o más puedan estar conectados.

Por tanto el factor consumo y el de sobrecarga en la red o línea son elementos anidados y heredados, capaces de alertar la viabilidad de conexión de dicha central al sistema. Esto conlleva es disponer de una entidad Auxiliar de Excepciones tales que alertan al usuario de la confirmación de tal

Se establece en su diseño la economía del mismo y mostrando así una estructuración eficiente y optimizada.

- ✓ Gestión de Asignación de Centrales Distribución a la línea de Comunicación.

Este gestor tiene como funcionalidad el conectarse a una o dos centrales de producción. La restricción es de que no puede tener asignada más de una central de producción.

Del mismo modo las características analizadas en asignación de una central a una línea de comunicación, esta también tiene las mismas funcionalidades que el anterior gestor pero afectadas de las restricciones comentadas.

- ✓ Gestión de Asignación de Contadores a un Contrato.

La gestión de asignación de contadores a un contrato de un cliente, depende del estado de contador, es decir su situación en el ámbito temporal y de disponibilidad para ser incorporado a la infraestructura de la red energética.

Por tanto como restricción condicionante es que para todo contrato ha de haber un contador es situación de alta y un cliente en situación de alta para ser asignado el contrato con el contador disponible.

Contrato es un objeto del mundo real que su existencia nace por unas entidades fuertes que los configuran como tal, es decir es una entidad débil que sin el concurso de las entidades cliente, contador y modelo no podría generarse como tal.

- ✓ Gestión de Asignación de Contratos a Clientes.

Es el proceso por el cual la relación se establece la relación existente entre Contrato y Cliente, teniendo presente que la integridad establecida por configuración en cascada de flujo de datos conlleva la restricción de que no puede asignarse un contrato a un cliente si este contrato no lleva adosado un contador en el mismo.

Por tanto es condición que todo contrato esta a signado a un cliente y solo uno y un contador a un contrato y solo uno, definiéndose la cardinalidad de las tres entidades integrantes de este gestor.

- ✓ Gestión de Asignación de Contadores a Líneas de Comunicación.

La gestión de contadores a las líneas de comunicación tiene una dualidad de conectividad, es decir conexión primaria y secundaria. Estos dos puntos de conectividad son justificados en base a un latente estado de posible sobrecarga en la red que esta funcionalidad física impone la continuidad del suministro energético.

Los contadores disponen de dos atributos por los cuales se asigna la línea en función del nivel de potencia que tiene cada contador, se propone como mejora el control de proximidad mediante la lectura y discriminación de los dígitos que contiene el campo código postal del contador que hereda de la entidad UBICACIÓN con el que hereda LINEA de COMUNICACIÓN, los 3 últimos dígitos de la derecha (######) informan de la localización en la zona mas próxima, si existe sobrecarga se va explorando y comparando estos tres últimos dígitos hasta encontrar la validez de conexión, en caso contrario se leerán los 4 dígitos y se continuará con el mismo procedimiento, si el recorrido da como no valida la conexión luego entonces se leerá los 5 dígitos hasta encontrar su conectividad, en caso negativo (falso) queda el contador como pendiente y no conectado al la espera de producirse una baja de un contador en la línea o bien la entrada en la red de una nueva línea nueva.

- ✓ Gestión de Lecturas de Consumos de los Clientes.

Este proceso se obtiene de forma calculada la información del consumo de uno o varios clientes o todos.

- ✓ Gestión Actualización de Consumos al Histórico de Consumos.

Este proceso actualiza la información generada por LECTURAS a esta entidad, proviene de un procedimiento formado por una vista o proyección tal que implementa los datos resumidos.

Su finalidad en primera instancia es poseer a nivel de seguridad la información crítica del sistema y de forma resumida en años y meses del consumo generado.

✓ Gestión Modulo Estadísticas.

Contiene el conjunto de procedimientos estadísticos generados mediante procedimientos dónde las entidades que concursan que están interrelacionadas mediante la integridad referencial ofrecen las extracciones y peticiones establecidas por el planteamiento inicial del proyecto.

Para ello se construyen los diferentes procesos, utilizando en la fase de implantación los disparadores y vistas necesarias para las extracciones.

- *Modulo Estadístico Consumos Central de Producción Contador*
- *Modulo Estadístico Promedio Consumos Año Contador Línea.*
- *Modulo Estadístico Máximo Consumo Línea cargada.*
- *Modelo Estadístico Líneas Superiores al 50% Consumido Año.*
- *Modelo Estadístico Líneas Inferior al 30% Consumido Año.*
- *Modelo Estadístico de los 10 Contadores Máximo Consumo.*
- Modulo Estadístico Consumo Medio de Clientes.

2.1.9 Requerimientos no Funcionales.

Requerimientos del entorno del software. Se precisa un entorno lo suficientemente estable para poder llevar a término el fin perseguido, suministrar al usuario la funcionalidad de acceso a los contenidos primando la consistencia y la versatilidad del soporte. Por tanto independientemente que el sistema comercial utilizado sea en si mismo una estructura válida de portabilidad fácil en las futuras migraciones que pueda sufrir el proyecto.

2.2 Diseño del Sistema.

Diseño estructural de la Base de Datos, arquitectura de la Base de Datos.

2.2.1 Diseño del modelo Conceptual.

Descripción esquematizada de la Base de Datos, utilizando para dicho fin un modelo de datos conceptual independiente al esquema conceptual de la BD i del SGB. Sin tener en cuenta los aspectos tecnológicos. Por tanto la estructuración del paquete a desarrollar sean detectado entidades fuertes y débiles que se irán desarrollando en el análisis y de que de este emanan las relaciones existentes que en el desarrollo se plasman en el esquema conceptual.

2.2.1.1 Modelo Entidad/Relación.

Como punto de partida del análisis de requerimientos obtenemos el siguiente refinamiento para la obtención del Modelo Conceptual de entidad Relación. Para esto el análisis de Requerimientos nos muestra los siguientes objetivos perseguidos para conseguir el Modelo Conceptual.

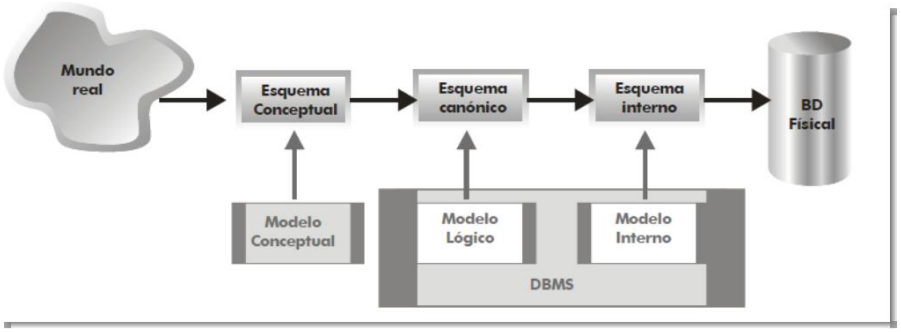


Ilustración 12 Modelo conceptual.

- **Mundo real.**

Contiene la información tal cual la percibimos como seres humanos y será la que el usuario debe de obtener con las sucesivas relaciones algebraicas que diseñemos.

- **Esquema conceptual.**

Representa el modelo de datos de forma independiente del Sistema Gestor de la Base de Datos (DBMS) que se utilizará en el proyecto.

- **Esquema canónico (o de base de datos).**

Representa los datos en un formato más próximo al que el Sistema físicamente tendrá el hardware instalado. Por tanto el diseño analizado debe de acercarse al máximo a las especificaciones técnicas que requieren y se deberán de disponer para su correcto funcionamiento.

- **Esquema interno.**

Representa los datos según el modelo concreto de un sistema gestor de Base de Datos que en este caso es ORACLE 10g, no obstante la arquitectura diseñada debe de cumplir el principio de reutilización, portabilidad requerida por los estándares de todo sistema de base de datos normalizado.

- **Base de datos física.**

Los datos tal cual son almacenados en disco interno del Host o bien en un Sistema de Base de Datos Distribuidas en diferentes Host interconectados.

2.2.1.2 Identificación de las entidades.

Las entidades identificadas según los requisitos son las siguientes:

ENTIDAD	DESCRIPCION
PAIS	**Contiene todos los datos pertenecientes a los países de la Comunidad Económica Europea.**
PROVINCIA	**Contiene todos los datos de la Provincias o bien regiones que pertenecen las Localidades de la Comunidad Económica Europea.**
LOCALIDAD	**Contiene el conjunto de ciudades pertenecientes a la Comunidad Económica Europea.**
VIA	**Contiene el conjunto de abreviaturas que identifican las vías públicas.**
UBICACION	**Contiene el conjunto de datos para poder ubicar geográficamente a las entidades Clientes, Fabricantes, Contadores y Centrales.** **Los datos son: Direcciones, Códigos Postales y el código de las Localidades que pertenecen dichas direcciones.**
PERSONA	**Contiene el tipo de Clientes, si son Particulares o Empresas.**
CLIENTE	**Contiene el conjunto de Personas que son Clientes. Los datos que tiene son el código de Ubicación que expresa los datos de la dirección el código postal y la localidad a que pertenece el Cliente.** **En dicha entidad también como atributo de ordenación disyuntiva hay el tipo de rol (Empresa o Particular).** **En el conjunto de información hay el día de alta y la situación de alta o baja.**
CONTRATOS	**Conjunto de contratos asignados a los Clientes con un contador determinado.**
FABRICANTE	**Conjunto de Fabricantes que realizan inspecciones a las Centrales.**
MODEL	**Información referente a los Modelos de Contadores que fabrican los Fabricantes.**
COMPTADOR	**Conjunto de dispositivos susceptibles de conectarse a una o más líneas de comunicación.**
CENTRAL	**Contiene el conjunto de Centrales con clases de Centrales de Producción y Distribución, con la información de las características de Tipos de Central de energía y funcionalidades junto con la fecha de la Inspección.**
TIPOS_CENTRAL	**Hay reflejada en esta entidad las diferentes funcionalidades de tipos de Central.**
INSPECCIONES	**Conjunto de Inspecciones realizadas en las Centrales. En una fecha determinada por un inspector y el resultado de la misma.**
LINEAS_COMUNICACION	**Información referente a las líneas de comunicación donde se conectan las Centrales y los Contadores.**
LECTURAS	**Las lecturas efectuadas en un período de los contadores que actualizan el Histórico de Consumos.**
TIPUS_LECTURAS	**Tipos de lecturas, Presenciales o Telemáticas, según contador.**
HISTORIC_CONSUMS	**Conjunto de consumos de Lecturas en un mes y año de los contadores.**

2.2.1.3 UML Modelo Entidad/Relación.

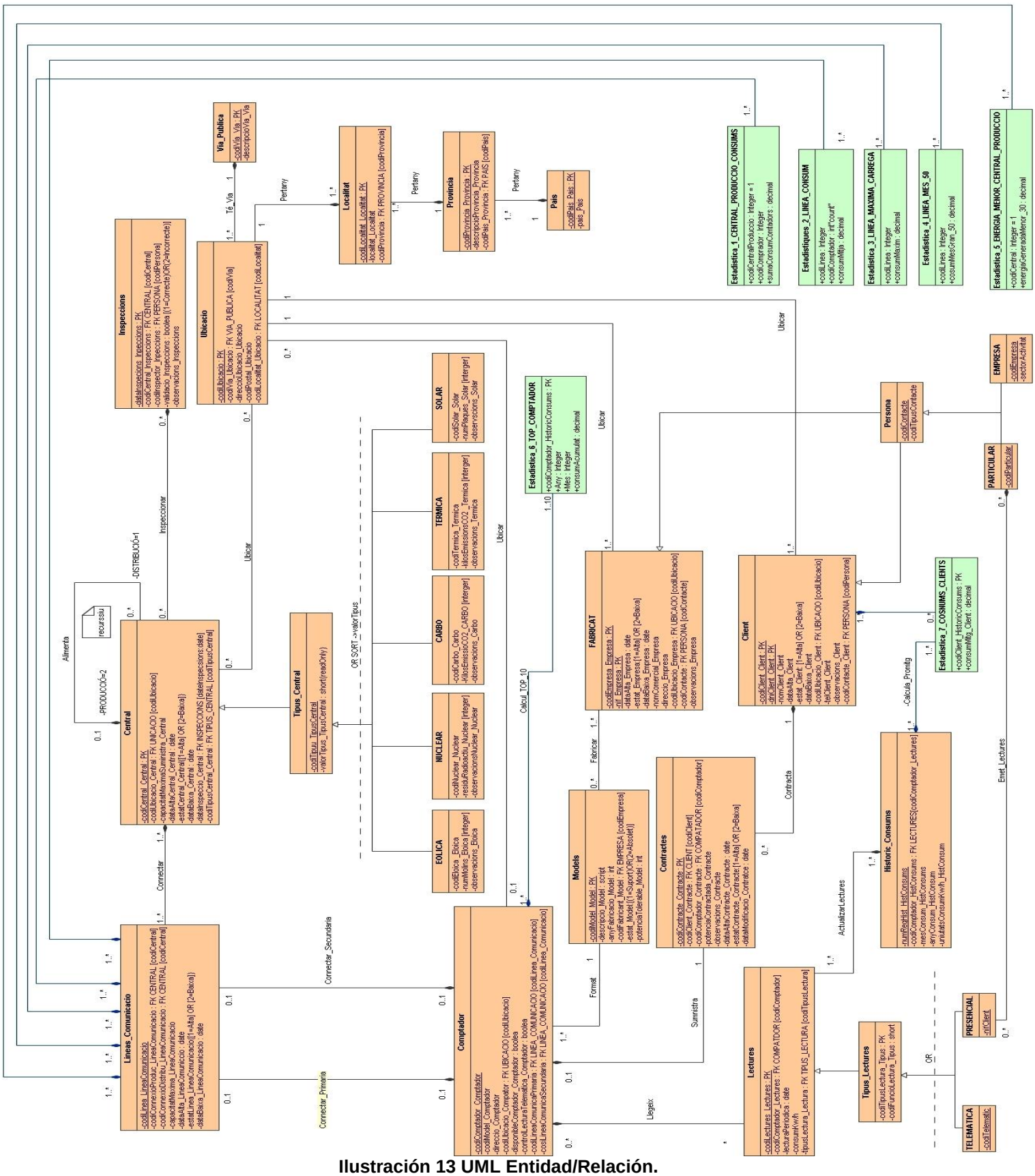


Ilustración 13 UML Entidad/Relación.

2.2.1.4 Descripción de las Entidades y los Atributos.

Identificación de las entidades de su rol con la descripción detallada del conjunto de atributos las caracteriza.

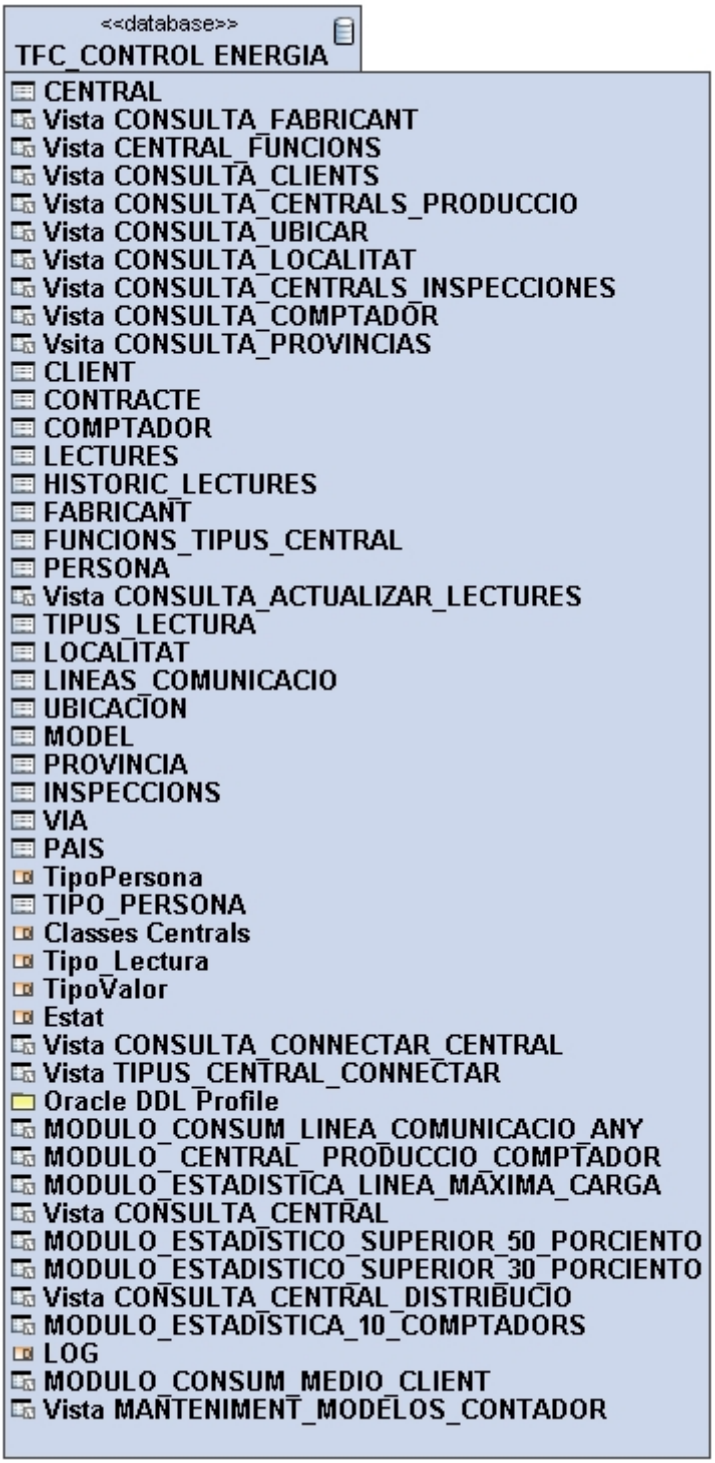


Ilustración 14 Package TFC CONTROL ENERGIA.

2.2.1.4.1 País.

Contiene los países pertenecientes a la Unión Europea.

Atributos:

2.2.1.4.1.1 codiPais_Pais

Es la Clave Primaria PK de tipo numérico entero, como restricción es que no puede contener un valor nulo y al ser PK no admite duplicidades.

2.2.1.4.1.2 descripcioPais_Pais

Tiene formato de texto con una longitud máxima de 70 caracteres, no admite valores nulos y ni duplicidades.

PAIS			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiPais_Pais	number	PrimaryKey	Not Null
descripcioPaís_Pais	VARCHAR2(70 CHAR)		Not Null

Ilustración 15 Estructura atributos entidad PAIS.

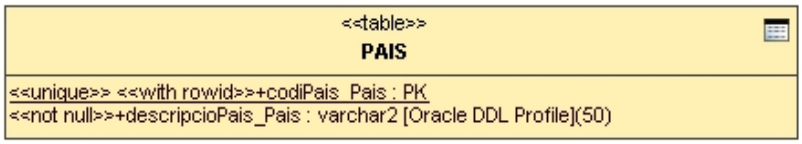


Ilustración 16 Entidad PAIS

2.2.1.4.2 Provincia.

Contiene las provincias que pertenece a País.

Atributos:

2.2.1.4.2.1 codiProvincia_Provincia

Es de tipo numérico entero, es una Clave Primaria PK, no admite valore Null, identifica de forma unívoca la provincia.

2.2.1.4.2.2 descripcioProvincia_Provincia

Es de tipo texto de longitud 70 posiciones, no admite valores nulos ni duplicados.

2.2.1.4.2.3 *codiPais_Provincia*

La relación establecida de la entidad PROVINCIA con PAÍS implica tener el atributo *codiPais_Provincia* que es forana de *PAÍS* con el *codiPais_Pais*. No admite valores nulos.

PROVINCIA			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiProvincia_Provincia	number	PrimaryKey Unike	Not Null
descripcioProvincia_Provincia	VARCHAR2(70 CHAR)		Not Null
codiPais_Provincia	number	FK PAIS[codiPais]	Not Null

Ilustración 17 Estructura atributos entidad PROVINCIA.

Ilustración 18 Entidad Provincia.

2.2.1.4.3 Localidad.

Esta entidad tiene como atributo que identifica de forma unívoca cada una de las tuplas el atributo

Atributos:

2.2.1.4.3.1 **codiLocalitat_Localitat**

Es de tipo numérico entero, es una Clave Primaria PK y no admite valores nulos.

2.2.1.4.3.2 descripcioLocalitat_Localitat

Es de formato de texto de 70 posiciones no admite valores nulos, pero si duplicados ya que puede existir el nombre de una localidad pero en distinta provincia.

2.2.1.4.3.3 *codiProvincia_Localitat*

Es de tipo numérico entero y es forana de la entidad *PROVINCIA* con *codiProvincia_Provincia*, debido a la relación existente entre las dos entidades.

LOCALITAT			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiLocalitat_Localitat	number	PrimaryKey Unike	Not Null
descripcioLocalitat_Localitat	VARCHAR2(70 CHAR)		Not Null
codiProvincia_Localitat	number	FK PROVINCIA[*codiProvincia]*	Not Null

Ilustración 19 Estructura atributos entidad LOCALITAT.

Ilustración 20 Entidad Localidad.

2.2.1.4.4 Vía.

Esta entidad contiene los tipos de vías públicas posibles, su finalidad es minimizar errores posibles en las extracciones de datos.

Atributos:

2.2.1.4.4.1 codiVia_Via

Como Clave Primaria PK es de tipo numérico entero no admite valores nulos.

2.2.1.4.4.2 descripcioVia_Via

Es de tipo texto de 15 posiciones no admite valores nulos ni duplicados.

VIA			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiVia_Via	number	PrimaryKey Unike	Not Null
descripcio_Via	VARCHAR2(15 CHAR)		Not Null

Ilustración 21 Estructura atributos entidad VÍA

Ilustración 22 Entidad Vía Pública.

2.2.1.4.5 Ubicación.

Esta entidad contiene las direcciones y códigos postales de una localidad, por tanto contiene:

Atributos:

2.2.1.4.5.1 codiUbicacio_Ubicacio

Como Clave Primaria PK de tipo numérico entero no admite valores nulos ni duplicados.

2.2.1.4.5.2 *codiVia_Ubicacio*

Es de tipo numérico entero y es forana de *VIA* [codiVia] no admite valores nulos pero si duplicados.

2.2.1.4.5.3 direccioUbicacio_Ubicacio

Es de tipo texto de 75 posiciones no admite valores nulos pero si duplicidades.

2.2.1.4.5.4 codiPostal_Ubicacio

Es de tipo texto de de 9 posiciones no admite valores nulos pero si duplicidades.

2.2.1.4.5.5 *codiLocalitat_Ubicacio*

Es numérico entero forana de *LOCALITAT* [codiLocalitat] no admite valores nulos pero si duplicidades.

UBICACION			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiUbicacio_Ubicacio	number	PrimaryKey Unike	Not Null
codiVia_Ubicacio	number	FK VIA [codiVia]	Not Null
direccioUbicacio_Ubicacio	VARCHAR2(75 CHAR)		Not Null
codiPostal_Ubicacio	VARCHAR2(9 CHAR)		Not Null
codiLocalitat_Ubicacio	number	FK LOCALITAT [codiLocalitat]	Not Null

Ilustración 23 Estructura atributos entidad UBICACIÓN.

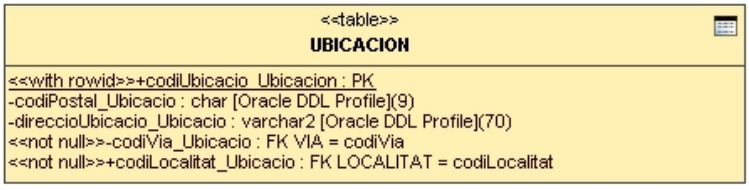


Ilustración 24 Entidad Ubicación.

2.2.1.4.6 Cliente.

Entidad que identifica del mundo real todas las características que debe tener un cliente y es contenedora de todos los clientes que consumen energía o bien la han consumido. Es una entidad fuerte y crítica del sistema.

Atributos:

2.2.1.4.6.1 codiCliente_Client

Como Clave Primaria PK, su funcionalidad incrementable cuya finalidad es identificar de forma unívoca la tupla que especifica los datos de un cliente. No puede contener valores nulos ni tan poco admite duplicidades.

2.2.1.4.6.2 dniClient_Client

Su rol es de Clave Primaria PK, es única e identifica unívocamente a las tulpas de forma unívoca. No admite duplicados ni tan poco duplicidades, su formato es de texto de longitud de 10 posiciones.

2.2.1.4.6.3 dataAlta_Client

Su formato es de fecha, se genera al dar de alta a un cliente y tiene como predeterminada el día mes y año de la alta, no admite nulos pero si duplicidades dentro de la entidad Cliente.

2.2.1.4.6.4 estatClient

Este atributo depende de las restricciones disjuntas de la situación en que esta el cliente. Su formato es numérico entero donde los valores 1= Alta y 2=Baja. La condición de los valores son excluyentes, es decir o un valor o el otro pero nunca los dos. No admite nulos y a nivel de entidad admite duplicados, ya que de las tuplas generadas o registros pueden existir varios clientes en situación 1 o 2.

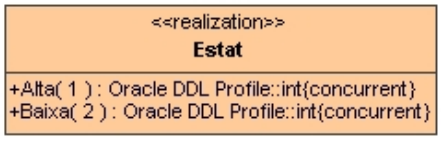


Ilustración 25 Auxiliar condición Estados Cliente.

2.2.1.4.6.5 dataModificacio_Cliente

Este atributo tiene formato de fecha día mes y año, durante el proceso de gestión de altas este atributo no es modificable, solo tiene premisos durante el procedo de modificaciones de datos o información crítica.

2.2.1.4.6.6 nomClient_Client

Es de formato de texto, se le da una longitud de 70 posiciones (caracteres) por tener que contener el nombre y los dos apellidos.

2.2.1.4.6.7 *codiUbicacio_Cliente*

Es de tipo numérico entero, contiene el código de ubicación de donde se encuentra físicamente en el mundo real el cliente, de este atributo se podrá obtener todos los datos referentes a la dirección código postal y localidad donde reside. Por tanto es una Clave Forana FK de la tabla UBICACIO. No admite nulos y puede ser duplicada dentro de los registros o tuplas de la tabla CLIENTE.

2.2.1.4.6.8 TelClient_Client

Su formato es de texto de 12 posiciones (caracteres), es de texto por motivos de no tener que efectuar ninguna operación de cálculo. Admite valores nulos.

2.2.1.4.6.9 observacionsCliente_Cliente

Su formato de texto de 255 posiciones libre para anotar observaciones referentes al cliente. Admite valores nulos.

2.2.1.4.6.10 *codiContacte*

Es numérico entero, contiene el código de tipo de cliente, es una opción disyuntiva no admite valores nulos y es una Clave Forana FK de PERSONA la cual contiene la tipología de dicho cliente des de el punto de vista jurídico o fiscal.

CLIENTE			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiClient_Client	number	PrimaryKey Unike	Not Null
dniCleint_Client	VARCHAR(10 CHAR)	PrimaryKey Unike	Not Null
dataAltaClient_Client	date		Not Null
estatClient_Client	number	Value: 1=Alta. 2=Baixa (condition OR)	Not Null
dataModoficacio_Client	date		Null
nomClient_Client	VARCHAR2(70 CHAR)		Not Null
codiUbicacio_Client	number	FK UBICACIO [codiUbicacio]	Not Null
observacionClient_Client	VARCHAR2(255 CHAR)		Null
codiContacteClient_Client	number	FK PERSONA [codiPersona]	Not Null

Ilustración 26 Estructura atributos entidad CLIENTE.

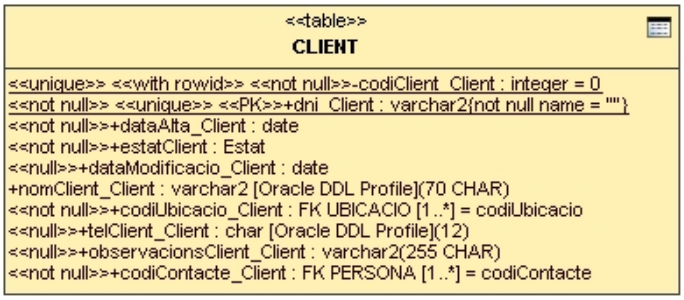


Ilustración 27 Entidad Cliente

La entidad Cliente tiene unas dependencias de composición tales que discriminan el estado del cliente y la tipología del cliente según el rol fiscal o jurídico en el mundo real. El tipo de persona de acuerdo con las restricciones establecidas en el enunciado solo pueden ser Particular o bien Empresa, no obstante se ha decidido diseñar una tabla de Persona por el principio de reutilización, los motivos son una mejora válida si del cliente se desea en un futuro conocer la actividad a la cual dedica su ámbito de negocio en caso de ser una empresa o en caso de un particular su profesión. En este ultimo supuesto de mejora solo debería de crearse un atributo más que sería el de activitatPersona_Persona que podría ser una Clave Forana FK de una entidad ACTIVITAT.

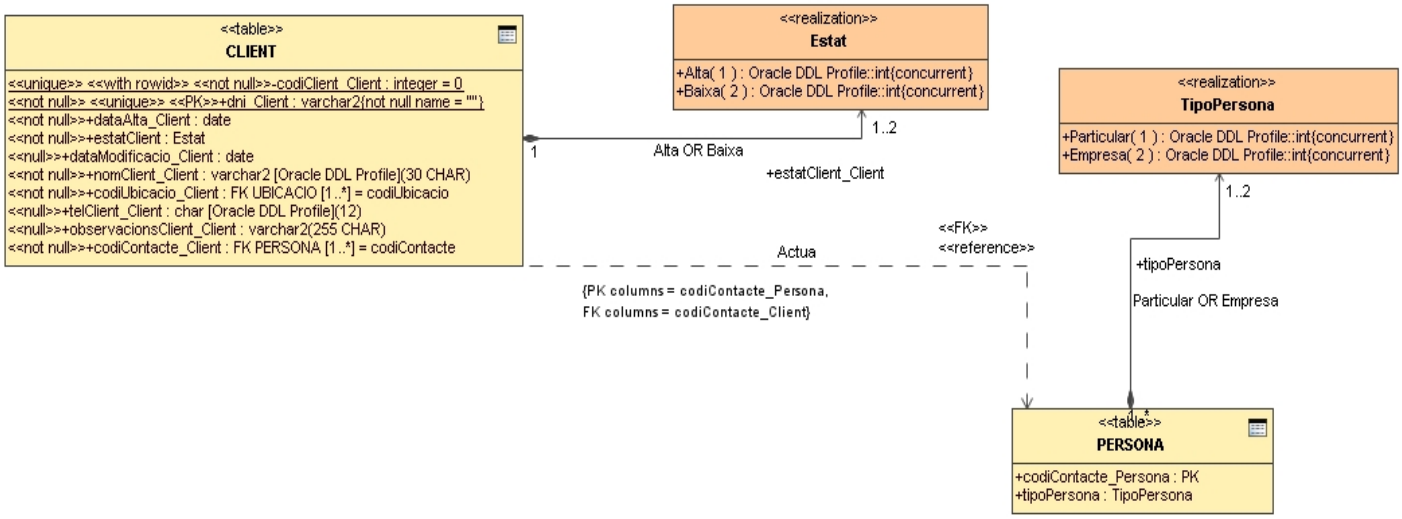


Ilustración 28 Dependencias Tabla Cliente.

2.2.1.4.7 Persona.

Esta entidad contiene el tipo de restricciones que un Cliente por sus características o rol tiene, esta entidad complementa los datos que la tabla CLIENTE tiene definidos.

Atributos:

2.2.1.4.7.1 codiPersona_Persona

Su formato es numérico entero incrementable, es una Clave Primaria PK que identifica de forma unívoca i atomizada el tipo de persona, no admite valores nulos ni duplicidades dentro del conjunto de tuplas de esta entidad.

2.2.1.4.7.2 tipoPersona_Persona

Contiene las restricciones referentes al rol del cliente, es una disyunción no admite valores nulos es excluyente. Su formato es numérico entero que identifica dicha restricción.

1= Particular
2= Empresa

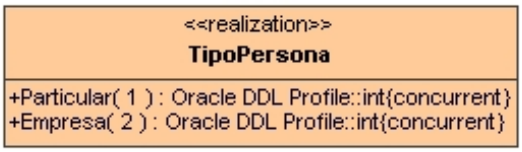


Ilustración 29 Tipo de Persona.

PERSONA			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiPersona_Persona	number	PrimaryKey Unike	Not Null
tipoPersona_Persona	number	Value: 1=Particular. 2=Empresa. (condition OR)	Not Null

Ilustración 30 Estructura atributos entidad PERSONA.

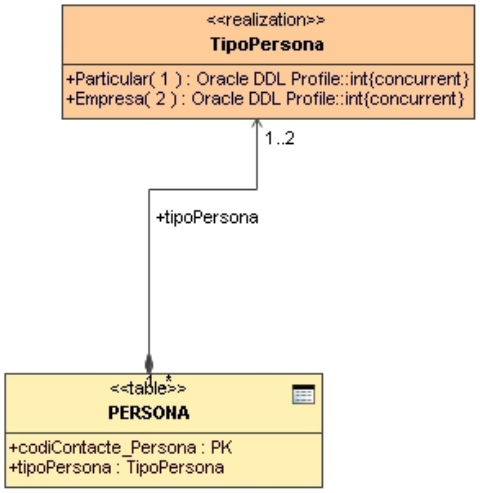


Ilustración 31 Dependencia Entidad Persona.

2.2.1.4.8 Contratos.

Esta entidad contiene los tipos de contratos que un Cliente tiene por recibir suministro energético, por tanto deberá de identificara al cliente, un contador asignado la potencia contratada, la fecha en que se produce el alta de contrato en el proceso de dar de alta del contrato tendrá como predeterminado el valor del sistema operativo, su estado si es un contrato en situación de Alta o de Baja y la fecha de modificación si se produce alguna.

Atributos:

2.2.1.4.8.1 codiContracte_Contracte

Es de formato numérico entero incrementable, es una Clave Primaria, no admite valores nulos ni duplicidades.

2.2.1.4.8.2 *codiComptador_Contracte*

Su formato es numérico entero, es una Clave Forana FK de *CONTADOR* que identifica el contador asignado, no puede tener valores y si duplicados, los motivos primero porque un cliente aunque este de baja en su momento siempre tuvo asignado un contador y el hecho de que todo el sistema esta configurado en cascada no admite por definición el estar en situación Null así mismo admite duplicidades por el hecho de que un Cliente puede tener asignado más de un contador aunque este contador este ubicado geográficamente en otro lugar.

2.2.1.4.8.3 potenciaContracte_Kw/h

Su formato es decimal de dos posiciones decimales, el motivo por ser un atributo del cual deberá de llevarse acabo operaciones.

2.2.1.4.8.4 dataAltaContracte_Contracte

Tiene el formato de fecha de día, mes y año. Contiene la fecha en que se produce la alta del contrato al Cliente y se asigna un contador al Cliente. Tiene como predeterminado la fecha del sistema, no admite valores nulos pero si duplicidades dentro del conjunto de tuplas de la entidad Contratos.

2.2.1.4.8.5 *estatContracte_Contracte*

Este atributo depende de las restricciones disjuntas de la situación en que esta el Contrato. Su formato es numérico entero donde los valores 1= Alta y 2=Baja. La condición de los valores son excluyentes, es decir o un valor o el otro pero nunca los dos. No admite nulos y a nivel de entidad admite duplicados, ya que de las tuplas generadas o registros pueden existir varios clientes en situación 1 o 2.

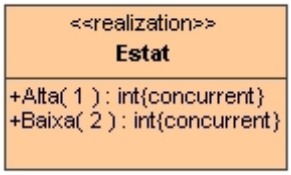


Ilustración 32 Estado situación Contrato

2.2.1.4.8.6 dataModificacio_Contracte

Este atributo tiene formato de fecha día mes y año, durante el proceso de gestión de altas este atributo no es modificable, solo tiene premisos durante el procedo de modificaciones de datos o información crítica.

2.2.1.4.8.7 *dniClient_Contracte*

Identifica el Cliente en el cual se ha formulado el contrato, es Clave Forana FR de *CLIENTE* [dniClient], su tipo es numérico entero, no admite valores nulos pero si duplicidades.

CONTRATO			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiContracte_Contracte	number	PrimaryKey Unike	Not Null
codiComptador_Contracte	number	FK UBICACIO [codiUbicacio]	Not Null
dataAltaContracte_Contracte	date		Not Null
estat_Contracte	number	Value: 1=Alta. 2=Baixa. (condition OR)	Not Null
dataModificacioContracte_Contracte	date		Null
dniClient_Contracte	number	FK CLIENT [codiClient]	

Ilustración 33 Estructura atributos entidad CONTRATO.

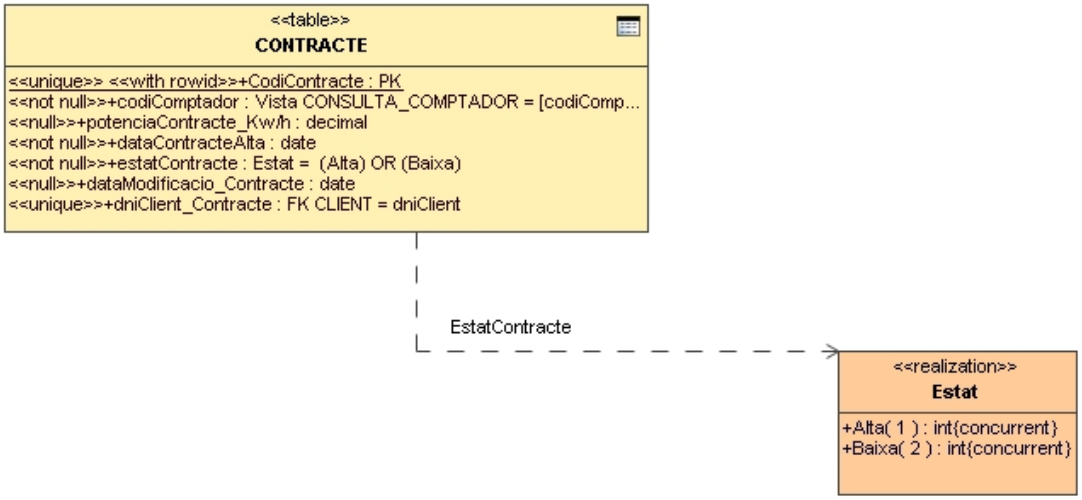


Ilustración 34 Entidad Contratos

2.2.1.4.9 Fabricante.

Esta entidad es contenedora de la información crítica del los Fabricantes de los Modelos de Contadores que requieren tener asignados los Contratos de los Clientes.

Por tanto es una entidad fuerte la cual le es dependiente Modelo, el hecho que todo Fabricante físicamente en el mundo real se halla geográficamente ubicado en un lugar concreto (Localidad) y con los diferentes datos que lo ubican como el código postal y su dirección. Implica que esta entidad dependa de la entidad *UBICACIÓN*, por otro lado todo Fabricante puede estar en una situación de estado (Alta o Baja). El sistema por el motivo de estar configurado en cascada, debe de permitir cambiar el estado de este, esto se basa en que un Fabricante en un momento dado puede dejar de suministrar o de fabricar un Modelo de Contador. Como condición de consistencia de datos en la estructura de la Base de Datos, debe de continuar sus datos en dicha entidad, si no fuera así implicaría la extracción de información errónea e imprevisible en el comportamiento de acceso a los datos relacionales de la arquitectura diseñada. También se a tomado la decisión de que tenga una dependencia de composición con Persona, basándonos en la vida real, un Fabricante puede ser una Empresa o un Particular Autónomo.

Atributos:

2.2.1.4.9.1 codiEmpresa_Fabricant

Tiene el formato numérico entero incrementable que identifica internamente cada uno de las tuplas insertadas. Es una Clave Primaria PK, no admite valores nulos ni duplicados.

2.2.1.4.9.2 nifEmpresa_Fabricant

Este atributo contiene el NIF de identificación fiscal del Fabricante, este atributo tiene el formato de texto de 10, es Clave Primaria PK que identifica de forma unívoca al Fabricante no admite valores nulos ni duplicidades.

2.2.1.4.9.3 dataAlta_Fabricant

Tiene el formato de fecha de día, mes y año. Contiene la fecha en que se produce la alta del Fabricante. Tiene como predeterminado la fecha del sistema, no admite valores nulos pero si duplicidades dentro del conjunto de tuplas de la entidad Fabricante.

2.2.1.4.9.4 *estat_Fabricant*

Este atributo depende de las restricciones disjuntas de la situación en que esta el Fabricante. Su formato es numérico entero donde los valores 1= Alta y 2=Baja. La condición de los valores son excluyentes, es decir o un valor o el otro pero nunca los dos. No admite nulos y a nivel de entidad admite duplicados, ya que de las tuplas generadas o registros pueden existir varios clientes en situación 1 o 2.

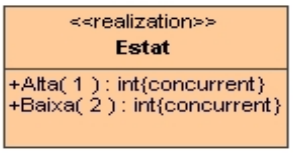


Ilustración 35 Estado Situación Fabricante.

2.2.1.4.9.5 dataModificacio_Fabricant

Este atributo tiene formato de fecha día mes y año, durante el proceso de gestión de altas este atributo no es modificable, solo tiene premisos durante el procedo de modificaciones de datos o información crítica.

2.2.1.4.9.6 nomComercial_Fabricant

Contiene el nombre Comercial y Fiscal del Fabricante, es de formato de texto de 70 posiciones, no admite valores nulos.

2.2.1.4.9.7 *codiUbicacio_Fabricant*

Los valores que adopta este atributo corresponden a los del código de Ubicaciones por tanto es una Clave Forana FK de la entidad *UBICACIÓN* [codiUbicacio], no admite valores nulos pero si duplicidades.

2.2.1.4.9.8 *codiPersona_Fabricant*

Identifica el código del rol que tiene dicho fabricante, en una Clave Forana FK de *PERSONA*, no admite valores nulos pero si duplicidades.

FABRICANTE			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiFabricant_Fabricant	number	PrimaryKey Unike	Not Null
nifEmpresa_Fabricant	VARCHAR(10 CHAR)	PrimaryKey Unike	Not Null
dataAltaFabricant_Fabri	date		Not Null
Estat_Fabricant	number	Value: 1=Alta. 2=Baixa. (condition OR)	Not Null
dataModificacio_Fabrica	date		Null
nomComercial_Fabrican t	VARCHAR2(70 CHAR)		Not Null
codiUbicacio_Fabricant	number	FK UBICACIO [codiUbicacio]	Not Null
codiPersona_Fabricant	number	FK PERSONA	Not Null

Ilustración 36 Estructura atributos entidad FABRICANTE.

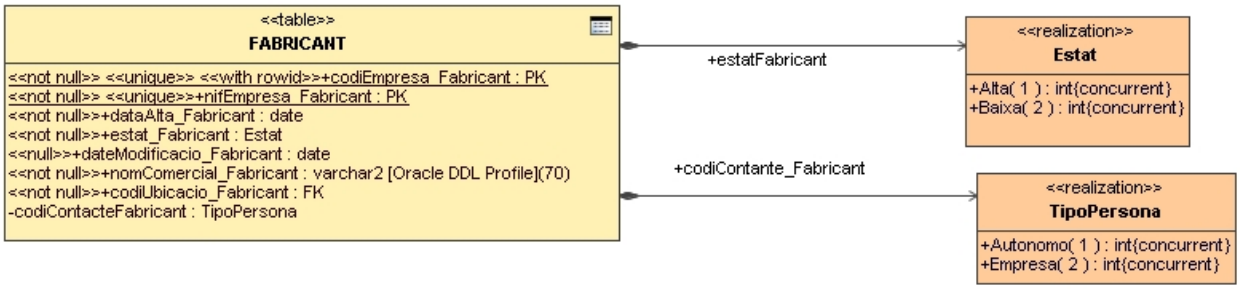


Ilustración 37 Entidad Fabricante.

2.2.1.4.10 Modelo.

Esta entidad contiene el conjunto de datos correspondientes al tipo de modelo que un Contador tiene. Por tanto tiene una dependencia de la entidad Fabricante.

Atributos:

2.2.1.4.10.1 codiModel_Model

Identifica de forma unívoca y atomizada el tipo de Modelo del Contador. Tiene formato numérico entero incrementable y es Clave Primaria PK, no admite valores nulos ni duplicidades.

2.2.1.4.10.2 descripcioModel_Model

Es la descripción del Modelo del Contador, su formato es de texto de 30 posiciones, no admite valores nulos, si duplicidades.

2.2.1.4.10.3 anyFabricacio_Model

Contiene el año de fabricación del Modelo de Contador, su formato es de texto 4 posiciones, no admite valores nulos y si duplicidades.

2.2.1.4.10.4 *codiFabricant_Model*

El valor que adopta es el valor del código del fabricante del modelo de contador. Es una Clave Forana FK de la entidad *FABRICANTE* [codiFabricant], no admite valores nulos pero si duplicidades.

2.2.1.4.10.5 connexioDual_Model

El valor es de tipo booleano, es decir admite 1=Verdadero o 2=Falso, su finalidad es informar si el Modelo de Contador puede o no conectarse y recibir flujo energético des de dos Líneas de Comunicación.

2.2.1.4.10.6 potenciaTolerable_Model

Es la potencia máxima tolerable que el Modelo de Contador admite, tiene formato numérico decimal de dos posiciones, no admite valores nulos pero si duplicidades.

MODELOS			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiModel_Model	number	PrimaryKey Unike	Not Null
descripcioModel_Model	VARCHARD2(30 CHAR)		Not Null
anyFabricacio_Model	CHAR(4)		Not Null
codiFabricant_Model	number	FK	Not Null
connexioDual_Model	boolean	1= True 2=False (condition OR)	Not Null
potenciaTolerabel_Model	decimal		Not Null

Ilustración 38 Estructura atributos entidad MODELOS.

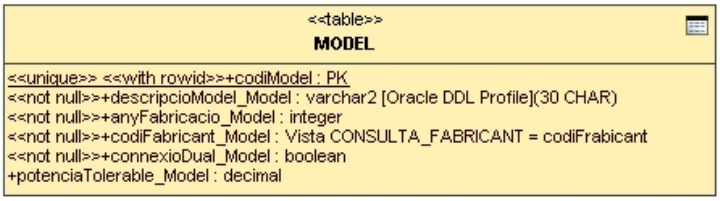


Ilustración 39 Entidad Modelos.

2.2.1.4.11 Inspecciones.

Contiene el conjunto de inspecciones realizadas a las Centrales tanto de distribución como las de Producción. Tiene una dependencia con Centrales por el motivo de tener que asignar a una Central la Inspección efectuada.

Atributos:

2.2.1.4.11.1 dateInspeccions

La fecha que se produce físicamente la Inspección, tiene la funcionalidad de Calve Primaria PK, no admite valores nulos ni duplicidades. El formato es de tipo DateTime, es decir toma el valor de día, mes, año, hora y minutos.

2.2.1.4.11.2 codiCentral_Inspeccions

Toma el valor seleccionado de la Central inspeccionada independientemente de que sea de Producción o de Distribución. Es una valor numérico entero siendo Calve Forana FK de la entidad *CENTRAL* [codiCentral], no admite valores nulos pero si duplicidades.

2.2.1.4.11.3 codiInspector_Inspeccions

El valor se le a dado de tipo de texto de 9 posiciones, no admite valores Nulos pero si duplicidades.

2.2.1.4.11.4 estadoInspeccion_Inspeccios

Su funcionalidad es detectar si se procedido o no a la inspección de una de las Centrales independientemente que sea esta de Producción o Distribución. Su formato es de tipo booleano por lo cual su valor puede ser 1=Verdadero OR 2=Falso.

INSPECCIONES			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
dateInspeccions	dateTime	PrimaryKey Unike	Not Null
codiCentral_Inspeccions	number	FK CENTRAL[codiCentral	Not Null
codiInspector_Inspeccions	VARCHAR2(9		Not Null
estatInspeccio_Inspeccions	boolean	1= True 2=False (condition OR)	Not Null

Ilustración 40 Estructura atributos entidad INSPECCIONES.

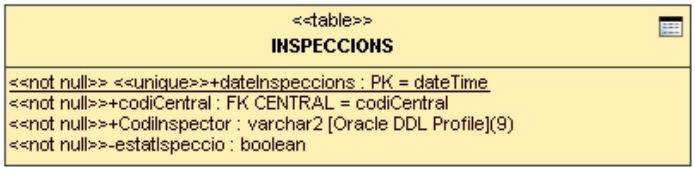


Ilustración 41 Entidad Inspecciones.

2.2.1.4.12 Contador.

Es la entidad que recibe el flujo energético de la Central de Distribución a través de la Línea de Comunicación. El contador esta asignado a un Cliente por medio del Contrato creado al Cliente.

La entidad precisa de una ubicación física geográfica en donde se encuentra, independientemente de la situación geográfica del Cliente. La disponibilidad del contador va en función de si esta o no conectado a la red energética.

El contador tiene una propiedad restrictiva del tipo de Lectura que contiene la disyunción de 1=Verdadero OR 2=Falso. Esto significa que la lectura es automatizada o es de forma presencial por exclusión.

Atributos:

2.2.1.4.12.1 codiComptador

Identifica de forma unívoca el contador en el sistema de flujo energético, es Clave Primaria PK, no admite valores nulos ni duplicidades, su formato es numérico entero incrementable y no admite duplicidades.

2.2.1.4.12.2 *codiUbicacio_Comptador*

Su funcionalidad es ubicar geográficamente el contador, es Clave Forana FK *UBICACIÓN* [codiUbicacio]. No admite valores Nulos ni duplicidades.

Como mejora en base a una Consulta de *LÍNEAS_COMUNICACION* y CONTADOR, tanto una entidad como otra dependen de UBICACIÓN por tanto puede encontrarse de forma eficiente la Línea de Comunicación más próxima mediante la selección y comparación de codiPostal_Vista Consulta LINEAS_COMUNICACION y el codiPostal Vista Consulta CONTADOR.

Si efectuamos la lectura de izquierda a derecha 3 posiciones encontramos la zona geográfica dentro de una Localidad, si la Línea de Comunicación no existe o es nula, entonces sumamos 1 al valor de las 3 posiciones de forma recursiva hasta alcanzar la línea más próxima, en caso de continuar siendo nula, se leerá de derecha a izquierda 2 posiciones para encontrar cual es la línea de conexión más cercana en Provincia y se volverá a leer las 3 ultimas posiciones con valor mínimo para encontrar la proximidad.

Esto implica verificar las restricciones del atributo disponibilidad de la entidad CONTADOR y si la conexión de dicho contador excede a la máxima potencia conectada que en estos instantes tiene la Línea de Comunicación, para evitar la sobrecarga y la caída de línea.

Este proceso puede ser generado mediante un procedimiento o ser codificado en la aplicación de la segunda fase del proyecto, es decir la aplicación que usaría este sistema de Base de Datos Relacional.

2.2.1.4.12.3 disponiblidad_Comptador

La disponibilidad es de tipo booleano, recibe 1=Verdadero o 2=Falso si el contador tiene en connexioPrimaria_Comptador=0 OR connexioSecundaria_Comptador=0. Si el resultado de la comparación es falso, entonces tenemos las conexiones libres del contador para ser asignado a una Línea.

Si es diferente debe de comprobarse connexioPrimaria_Comptador>0 si es Verdadero entonces connexioPrimaria_Comptador esta libre y pude conectarse a una Línea de Comunicación luego disponibilidadContador=1, debe de verificarse de forma recursiva connexioSecundaria_Comptador>0 si es así connexioSecundaria_Comptador esta libre y puede conectarse a una Línea de Comunicación luego disponibilidadContador=1 en caso contrario disponibilidadComptador=2, informando de que dicho contador no esta libre.

2.2.1.4.12.4 controlLecturaTelematica_Comptador

Informa de si tiene el modelo el dispositivo de lectura on-line telemática, es de tipo booleano, debe de tener un valor no nulo, los valores que puede contener son disyuntivos 1=Verdadero OR 0=Falso.

2.2.1.4.12.5 *codiModel_Comptador*

Contiene el código del Modelo del Contador, es una Clave Forana FK de *MODELO* [codiModel], es de tipo numérico entero, no admite valores nulos y si duplicados.

2.2.1.4.12.6 *connexioPrimaria_Comptador*

Depende de la *Gestión Líneas Comunicación Contadores.* Tiene el valor del código de la Central de Distribución a la cual se conecta el contador des de la Línea de Comunicación. Es de tipo numérico entero y es Clave Forana FK de *LINEA_COMUNICACION* [codiLineaComunicacion_LineaComunicacion].

Con la restricción de que {([codiCentralDistribucion_LineaComunicacion] <>0) AND ([connexioSecundaria_Comptador] <> [connexioPrimaria_Comptador]}

2.2.1.4.12.7 *connexioSecundaria_Comptador*

Depende de la *Gestión Líneas Comunicación Contadores*. Tiene el valor del código de la Central de Distribución a la cual se conecta el contador des de la Línea de Comunicación. Es de tipo numérico entero y es Clave Forana FK de *LINEA_COMUNICACION* [codiLineaComunicacion_LineaComunicacion].

Con la restricción de que {([codiCentralDistribucion_LineaComunicacion] <>0) AND ([connexioPrimaria_Comptador] <> [connexioSecundaria_Comptador])}.

CONTADOR			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiComptador	number	PrimaryKey Unike	Not Null
codiUbicacio_Comptador	number	FK UBICACIO [codiUbicacio]	Not Null
disponiblitat_Comptador	boolean	1= True 2=False (condition OR)	Not Null
codiModel_Comptador	number	FK MODEL[codiModel]	Not Null
connexioPrimaria_Comptador	number	FK LINEAS_COMUNICACIO [codiCentralDistribucio]	Null
connexioSecundaria_Comptador	number	FK LINEAS_COMUNICACIO [codiCentralDistribucio]	Null
controlLectura_Comptador	boolean	1= True 2=False (condition OR)	Not Null

Ilustración 42 Estructura atributos entidad CONTADOR.

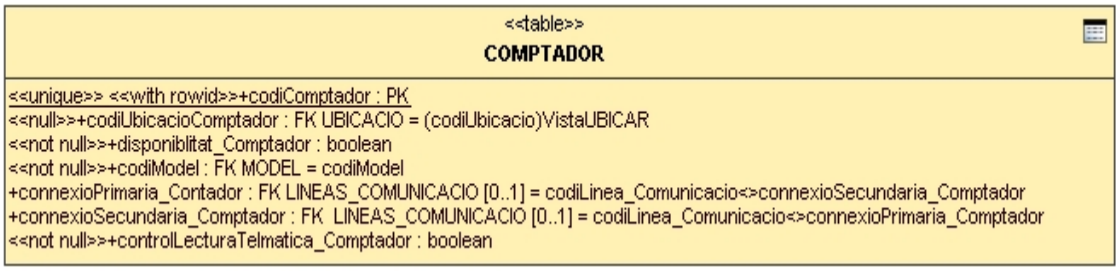


Ilustración 43 Entidad Contador.

2.2.1.4.13 Central.

Entidad contenedora de las Centrales, en ella se identifica la Clase de Central que según su rol puede ser 1=Producción y 2=Distribución.

La entidad tiene una recursividad la cual selecciona el tipo de Central mediante el código de Central.

Las Centrales están supeditadas a una capacidad máxima y a un estado, dicho estado es modificable y es disyuntiva, Alta o Baja. Este estado puede debe de tener obligatoriamente una fecha, sea por alta o por modificación de cualquier de los elementos que constituyen una Central. El tratamiento es el mismo que debe de hacerse con Fabricante, Contador, Contrato y Líneas de Comunicación.

Atributos:

2.2.1.4.13.1 codiCentral

Es el atributo que identifica de forma unívoca una Central, es de tipo numérico entero incrementable, no admite valores nulos ni duplicados, es una Clave Primaria PK.

2.2.1.4.13.2 classesCentral_Central

Contiene las clases de Central, su relación es disyuntiva, 1=Producción OR 2=Distribución, identifica el rol que ocupa dentro del conjunto de Centrales. El tipo de atributo es un numérico entero, no admite nulos y si admite duplicados.

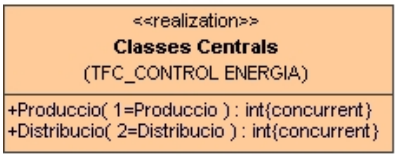


Ilustración 44 Clases de Centrales.

2.2.1.4.13.3 *codiFuncionesCentral_Central*

El valor que adopta es del tipo de funcionalidades que pude tener una Central en función de la energía que genera. Es una Clave Forana FK de *TIPO CENTRAL*, las Centrales de Producción contiene este código de funcionalidades y solo ellas. Las Centrales de Distribución este valor es nulo. El tipo de formato es de numérico entero, admite valores nulos y duplicidades.

2.2.1.4.13.4 *codiTipusCentral_Central*

Recibe el código de Central, es una recursividad de la misma tabla que identifica la asignación del rol de la misma. El valor numérico entero y es actualizado mediante Gestión Selección Líneas Conectar Centrales. Como restricción en este caso es de que la seleccionar el código de la Central a conectar es de que no puede conectarse una Central de Producción reflexivamente, que una Central de Distribución solo puede conectarse a una de Producción y solo una, que más de dos Centrales de Distribución no pueden conectarse a una Central de Producción y que las Centrales indistintamente debe de estar en situación de Alta. Puede contener valores nulos y duplicidades. Actúa de Clave Forana FK de CENTRAL

2.2.1.4.13.5 *codiUbicacioCentral_Central*

Depende de la entidad *UBICACIÓN* es una Clave Forana FK [*codiUbicacio*], es de tipo numérico entero, no admite valores nulos pero si duplicados.

2.2.1.4.13.6 capacitatMáximaSuministra_Central

Contiene la capacidad máxima de potencia capaz de suministrar, es de tipo decimal de dos posiciones. Puede contener valores nulos y duplicados.

2.2.1.4.13.7 dataAlta_Central

Es la fecha de alta de la Central, como predeterminada tiene la fecha del Sistema Operativo donde se realiza el proceso de *Gestión de Centrales*. Es de tipo fecha no admite valores nulos pero si duplicidades.

2.2.1.4.13.8 *estatCentral_Central*

Las Centrales pueden encontrase por definición de enunciado en 1= Alta OR 2=Baja, son estados disjuntos, no admite valores nulos pero si duplicidades. Es de tipo numérico entero.

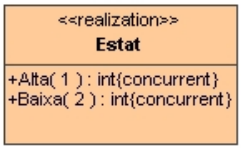


Ilustración 45 Estados de la Central.

2.2.1.4.13.9 dataModificacio_Central

Las modificaciones de estado o cualquier modificación como de conectividad a Líneas de Comunicación por asignar o modificar datos en codiTipusCentral o bien por cambio de ubicación. Dicho atributo debe de contener la fecha de esta modificación efectuada. Es de tipo fecha, admite valores nulos y duplicidades.

2.2.1.4.13.10 *dataInspeccion_Central*

Recibe la fecha de la inspección realizada, es una Clave Forana FK de la entidad *INSPECCIONES,* es de tipo fecha, no admite valores nulos pero si duplicidades.

CENTRAL			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiCentral	number	Primary Key Unike	Not Null
classesCentral_Central	int	1=Producció 2=Distribució	Not Null
codiFuncionsCentral_Central	number	FK TIPO_CENTRAL [codiTipusFuncio]	Not Null
codiTipoCentral_Central	number	FK Vista CONSULTA_CONNECTAR_CENTRAL [codiTipusCentral]	Not Null
codiUbicacioCentral_Central	number	FK UBICACIO [codiUbicacio]	Not Null
dataAlta_Central	date		Not Null
estatCentral_Central	int	1= Alta 2=Baixa (condition OR)	Not Null
dataModificacio_Central	date		Null
dateInspeccio_Central	date	FK INSPECCIONS[dateInspeccions]	Not Null
capacitatMaxmiaSuministra_Central	decimal		Null

Ilustración 46 Estructura y atributos entidad CENTRAL.

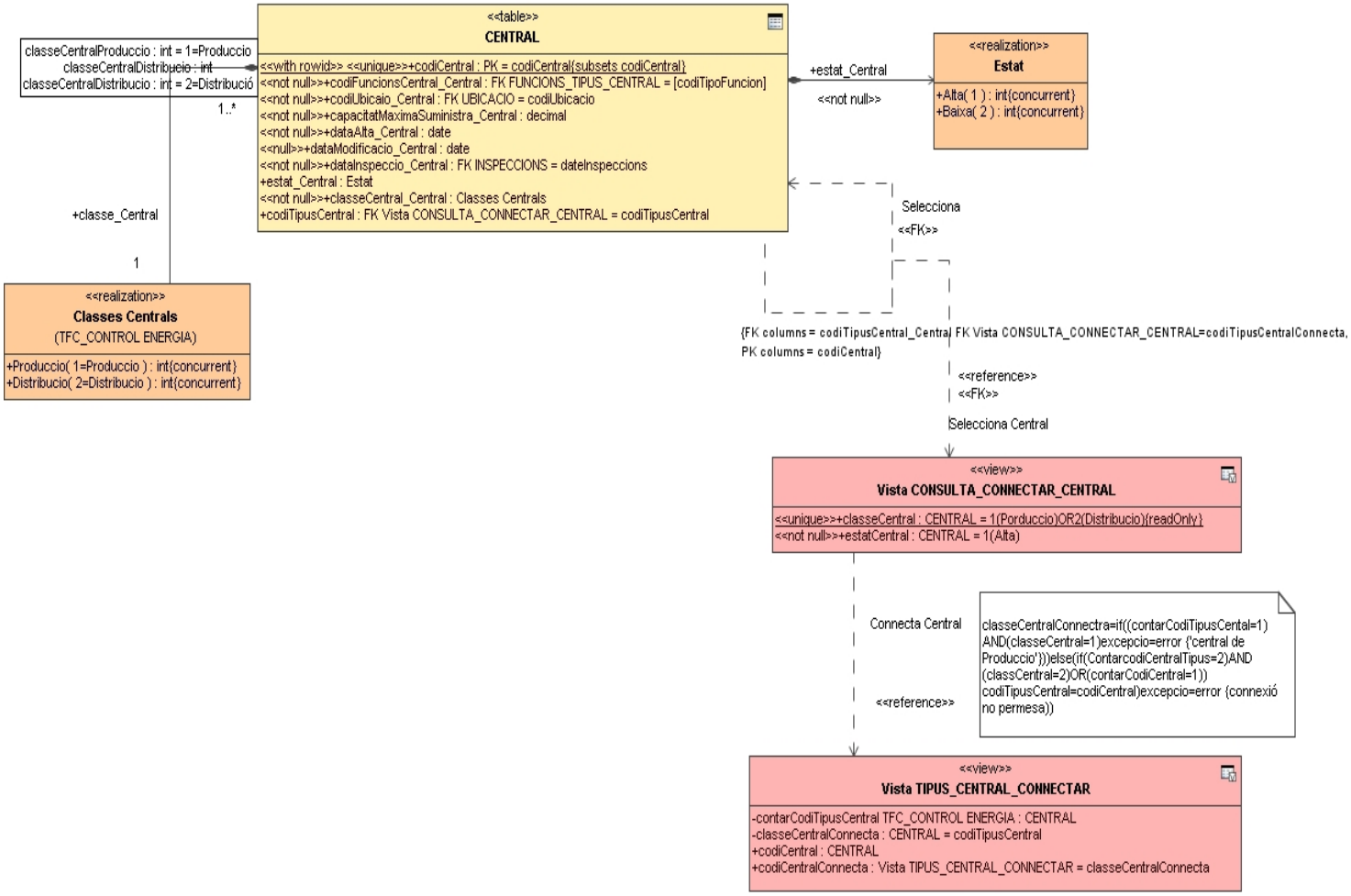


Ilustración 47 Entidad Central.

2.2.1.4.14 Tipo Central.

Esta entidad tiene los datos críticos que definen de forma física las funcionalidades y las características de las mismas. Sea tomado esta decisión de crear una entidad propia de tipo de funcionalidades por el principio de reutilización, la evolución en el campo de la energía puede incrementar la tipología de recursos energéticos susceptibles de suministrar energía una Central de Producción.

Atributos:

2.2.1.4.14.1 codiTipusFuncions

Identifica de forma unívoca el tipo de funcionalidad que tiene una *Central*, es una Clave Primaria PK, de tipo numérico entero, no admite valores nulos ni duplicidades.

2.2.1.4.14.2 *tipusFuncions*

Depende de la tipología de funcionalidades que pueda adoptarse. Su tipo es numérico entero donde los valores equivalen a los elementos que constituyen la Central Energética de Producción.

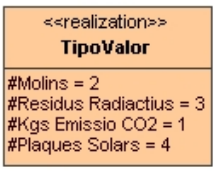


Ilustración 48 Componentes de Funcionalidad Centrales.

2.2.1.4.14.3 quantitat

Es el valor cuantitativo del número de elementos que constituyen la funcionalidad de la Central de Producción. Es de tipo decimal de dos posiciones, admite valores nulos y duplicidades.

TIPO_CENTRAL			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiTipusFuncions	number	Primary Key Unike	Not Null
tipusFuncions	int	1= Kgrs.Emissio CO2. 2= Molins. 3= Residus Radiactius. 4= Plaques Solars. (condition OR)	Not Null
quantitat	decimal		Null

Ilustración 49 Estructura y atributos entidad TIPO CENTRAL.

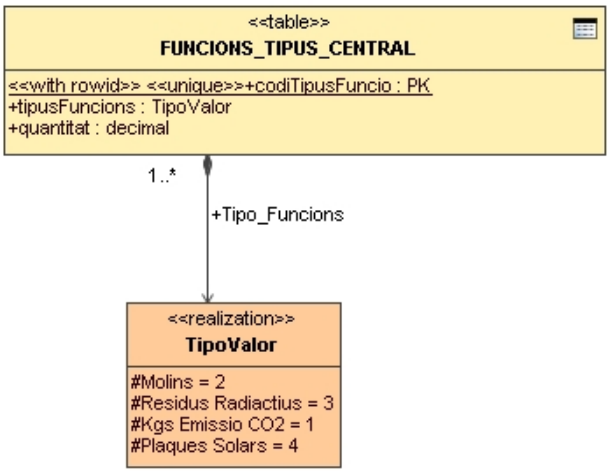


Ilustración 50 Entidad Tipo Central.

Las funcionalidades vienen determinadas por un proceso que se obtiene a partir de una consulta de funcionalidades de la central llamada Vista CENTRAL_FUNCIONES que tiene la entidad *CENTRAL*, dicho proceso pertenece a la *GESTIÓN FUNCIONES CENTRAL* tal como se describe en la siguiente ilustración.

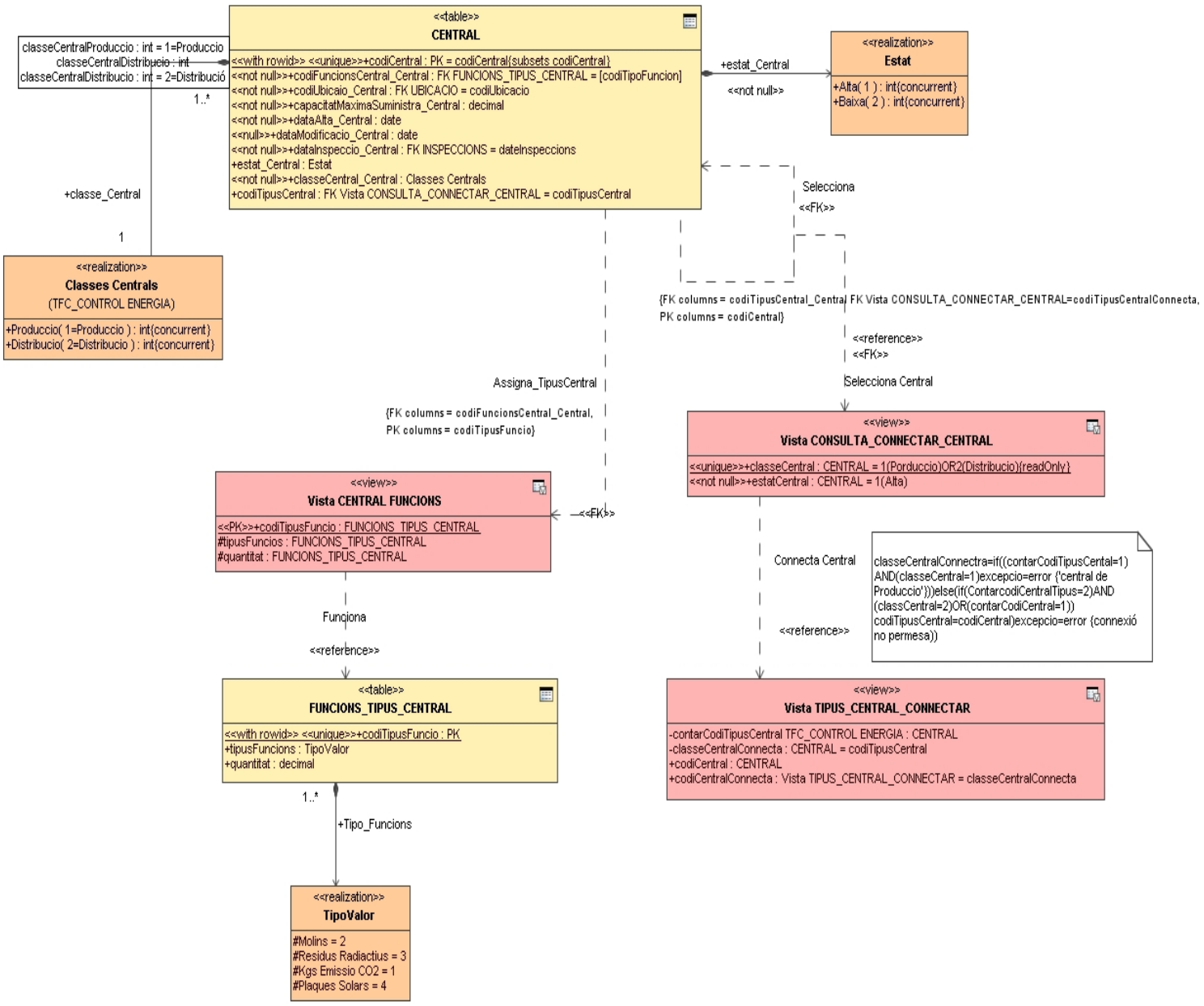


Ilustración 51 Dependencias Central y Funcionalidades

2.2.1.4.15 Líneas Comunicación.

La línea de comunicación es la entidad débil conectora capaz bajo restricciones de no exceder el máximo de potencia suministrada por la Central de Distribución y el conjunto de Contadores, por lo cual la entidad fuerte *CENTRAL* depende de esta y la entidad *CONTADOR* precisa de su existencia para establecer las conexiones.

La líneas por sus características sea por motivos de sobrecarga o bien por estar en sustitución provisional modificada, puede tener dos estados disjuntos de alta o baja.

Atributos:

2.2.1.4.15.1 codiLinea

Identifica unívocamente de forma atomizada el número de línea de comunicación. Es una Clave Primaria PK de tipo numérico entero incrementable, no admite valores nulos ni duplicidades.

2.2.1.4.15.2 codiConnexioProduccio_Linea_Comunicacio

Identifica el código de Central de Producción, es Clave Forana FK de una consulta mediante la Vista CONSULTA_CENTRAL_PRODUCCIO

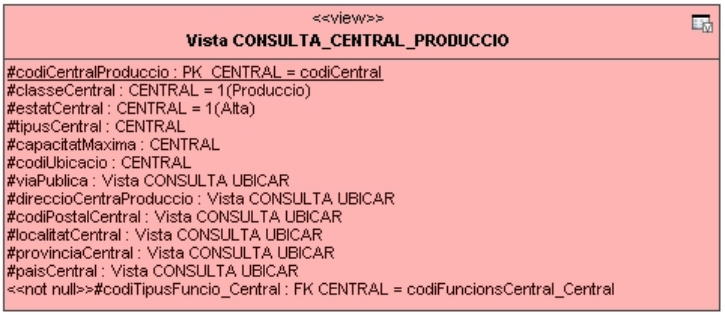


Ilustración 52 Consulta Central de Producción.

Su tipo de formato es numérico entero y no admite valores nulos. Como restricciones debe de estar su estado de Alta y el código que identifica las funcionalidades no puede tener valores nulos ya que estamos tratando una central de Producción y como tal tiene funciones particulares de generación de energía.

2.2.1.4.15.3 codiConnexioDistribucio_Linea_Comunicacio

Identifica el código de Central de Distribución, es Clave Forana FK de una consulta mediante la *Vista CONSULTA_CENTRAL_DISTRIBUCIO.*

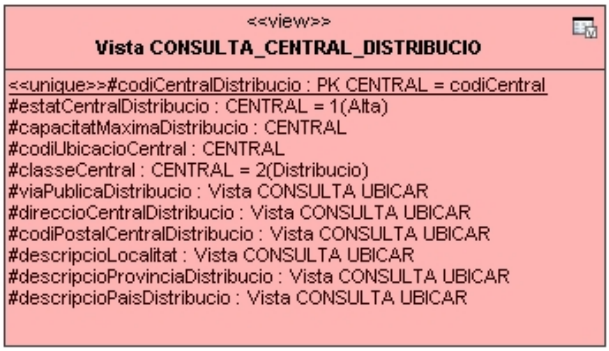


Ilustración 53 Consulta Central de Distribución.

Su tipo de formato es numérico entero y no admite valores nulos. Como restricciones debe de estar su estado de Alta y el código que identifica las funcionalidades debe de tener obligatoriamente valores nulos ya que estamos tratando una central de Distribución y como tal no tiene funciones particulares de generación de energía.

2.2.1.4.15.4 dataAlta_Linea_Comunicacio

Contiene el día en que se produce el alta de la línea, como predeterminado tiene la fecha del sistema operativo en el momento de dar el alta.de una nueva línea. Su formato es de fecha y admite duplicados.

2.2.1.4.15.5 *estat_Linea_Comunicacio*

Las Líneas pueden encontrase por definición de enunciado en 1= Alta OR 2=Baja, son estados disjuntos, no admite valores nulos pero si duplicidades. Es de tipo numérico entero.

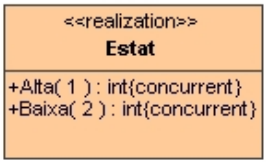


Ilustración 54 Estado Líneas Comunicación.

2.2.1.4.15.6 dataModificacio_Linea_Comunicacio

Toma los valores de fecha cuando se realiza alguna modificación en algún contenido de algún atributo de la entidad Línea de Comunicación.

LINEAS_COMUNICACIO			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiLinea	number	Primary Key Unike	Not Null
codiConnexioProduccio_Linea_Comunicacio	number	FK CONSULTA_CENTRAL_PRODUCCIO ([codiCentral].CENTRAL)	Not Null
codiConnexioDistribucio_Linea_Comunicacio	number	FK CONSULTA_CENTRAL_DISTRIBUCIO ([codiCentral].CENTRAL)	Not Null
dateAlta_Linea_Comunicacio	date		Not Null
estat_Linea_Comunicacio	int	1= Alta 2=Baixa (condition OR)	Not Null
dataModificacio_Linea_Comunicacio	date		Null

Ilustración 55 Estructura y atributos entidad LÍNEAS COMUNICACIO.

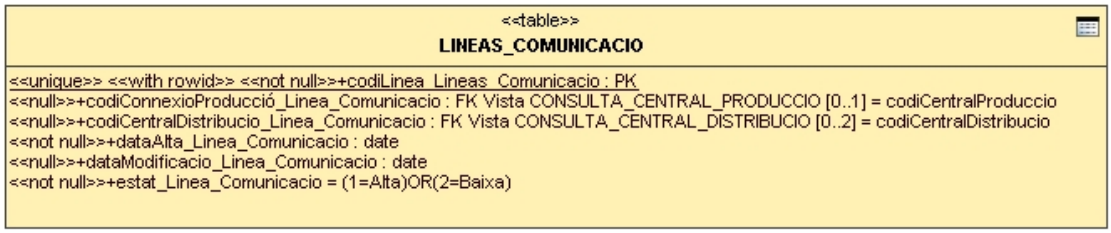


Ilustración 56 Entidad Líneas Comunicación.

2.2.1.4.16 Lecturas.

Esta entidad donde se materializa las lecturas del sistema, de esta entidad se extraen y actualizan las lecturas en el Histórico Consumos del Modulo Estadísticas.

Atributos:

2.2.1.4.16.1 codiLectures

Identifica de forma unívoca la lectura realizada, es una Clave Primaria PK, su tipo es numérico entero, no admite valores nulos ni duplicidades.

2.2.1.4.16.2 *codiComptador_Lectures*

Adopta el valor del código de contador del cual se efectúa la lectura, es Clave Forana FK *CONTADOR*[codiComptador], es de tipo numérico entero, no admite valores nulos pero si duplicidades en la entidad.

De este atributo podemos conocer casi toda la información del sistema de control de energía, ya que su diseño en cascada, las interrelaciones existentes no pueden perfectamente conocer des de un Contrato de un Cliente a partir de la lectura hecha de un Contador, el tráfico de suministro energético generada por una Línea de Comunicación a la que esta conectado un Contador

2.2.1.4.16.3 *codiLineaComunicacio_Lectures*

Contiene la línea la cual se efectúa la lectura es Clave Forana FK de *LÍNEAS_COMUNICACION* [codiLinea], es de tipo numérico entero no admite nulos, pero si duplicidades.

2.2.1.4.16.4 dateLectura_Lectures

Es la fecha en que se produce la lectura, es de tipo fecha, admite nulos y duplicidades.

2.2.1.4.16.5 *tipusLectura_Lectures*

Tiene el tipo de lectura realizada, esta lectura puede ser de un tipo determinado, sea considerado que sea heredera de una tabla de *TIPOS_LECTURAS*, el motivo es por el principio de reutilización, diseño en cascada y por normalización de la Base de Datos.

Es una Clave Forana FK de *TIPOS_LECTURAS* [codiTipusLectura], no admite valores nulos y si duplicidades en la entidad LECTURES.

LECTURAS			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiLectura	number	Primary Key Unike	Not Null
codiComptador_Lectures	number	FK CONTADOR[codiComptador]	Not Null
codiLineaComunicacio_Lectures	number	FK LÍNEAS_COMUNICACION [codiLinea]	Not Null
dateLectura_Lectures	date		Not Null
tipusLectura_Lectures	number	FK TIPOS_LECTURAS [codiTipusLectura]	Not Null

Ilustración 57 Estructura y atributos de la entidad LECTURAS.

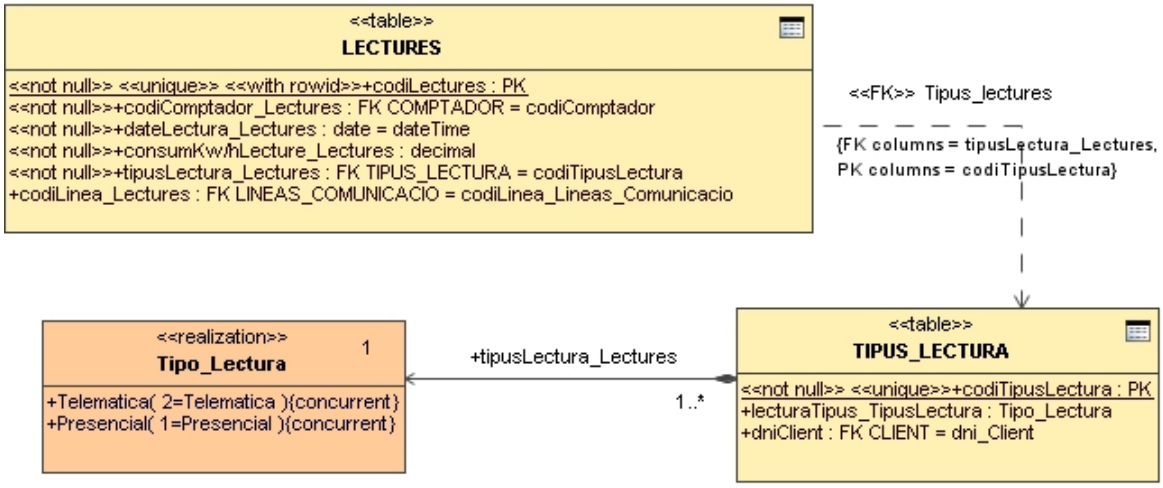


Ilustración 58 Entidad Lecturas

2.2.1.4.17 Tipos Lecturas.

Las lecturas que se producen dependiendo del tipo de contador y del modelo, admite las lecturas que sean Presenciales o Telemáticas. Sea diseñado una entidad para este proceso con el fin de poder ser reutilizada a posteriori. Sea aplicado así el principio de reutilización y refinamiento de la misma, estandarizándose dicho proceso.

Atributos:

2.2.1.4.17.1 codiTipusLectura_Tipus_Lectura

Identifica el tipo de lectura, es una Clave Primaria PK, es del tipo numérico entero, no admite valores nulos ni duplicidades.

2.2.1.4.17.2 lecturaTipus_Tipus_Lectura

Son el tipo de lecturas permisibles, son de tipo numérico entero, no admite valores nulos ni duplicidades, es disyuntiva solo puede tener un valor 1 OR 2.

1=Presencial.

2=Telemática.

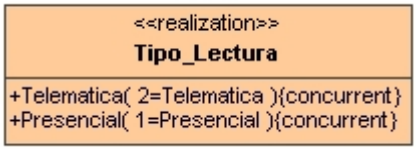


Ilustración 59 Tipo Lecturas.

2.2.1.4.17.3 *dniClient_Tipus_Lectura*

Sea diseñado este atributo para identificar que clientes tienen esta funcionalidad de lectura, es una Clave Forana FK de CLIENT [dniClient], es del tipo numérico entero, no admite valores nulos pero si duplicidades.

LECTURAS			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiTipoLectura	number	Primary Key Unike	Not Null
lecturaTipus_Tipus_Lectura	number	1=Presencial. 2=Telemática. (condition OR)	Not Null
dniClient_Tipus_Lectura	number	FK CLIENT [dniClient]	Not Null

Ilustración 60 Estructura y atributos de la entidad TIPO LECTURA.

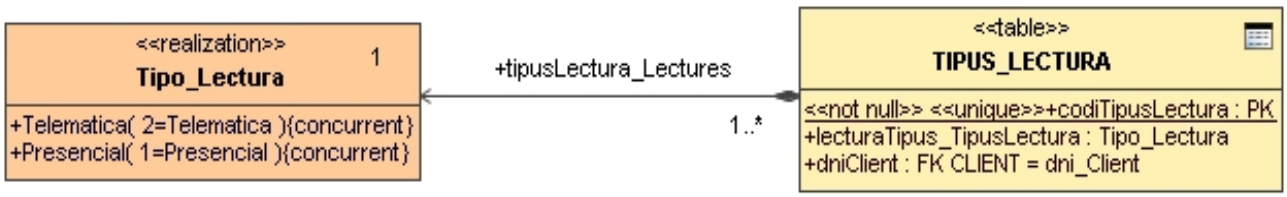


Ilustración 61 Entidad Tipo Lectura.

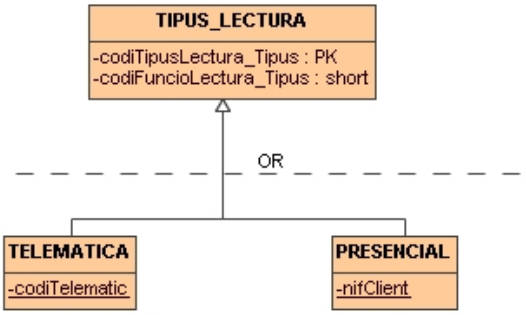


Ilustración 62 Tipo Lectura Generalización.

2.2.1.4.18 Histórico Lecturas.

Es la entidad colectora del conjunto de lecturas por meses y años de las diferentes líneas de suministro, es una entidad débil que solo puede existir si lecturas contiene información susceptible de ser tratada y actualizada en esta entidad histórico de lecturas.

De esta entidad pueden extraerse información referente al Modulo Estadístico que el proceso de tratamientos de datos críticos el sistema tiene que dar a las peticiones del usuario.

Atributos:

2.2.1.4.18.1 codiHistoric

Identifica de forma unívoca el registro o tupla que sea actualizado en la entidad, es Clave Primaria PK, es de tipo numérico entero incrementable, no admite valores nulos ni duplicados.

2.2.1.4.18.2 *codiLectures_Historic*

Contiene el codigo de las lecturas realizadas en la entidad LECTURAS, es una Clave Forana FK *LECTURAS* [codiLecturas], es de tipo numérico entero, admite no admite valores nulos, pero si duplicados.

2.2.1.4.18.3 mes

Es de formato de texto, contiene el mes en que sea realizado la lectura, es actualizado por medio de la *Gestión Actualizar Histórico*.

2.2.1.4.18.4 any

Es de formato de texto, contiene el año en que sea realizado la lectura, es actualizado por medio de la *Gestión Actualizar Histórico*.

2.2.1.4.18.5 consums

Contiene los consumos realizados y obtenidos a partir de Gestión Actualizar Histórico donde suma los consumos realizados en las líneas. Es de tipo numérico con dos decimales, admite valores nulos y duplicados.

2.2.1.4.18.6 numComptadors

Este atributo contiene el número de contadores que han consumido energía en la línea de comunicación a la que están conectados. Es de tipo numérico entero, admite valores nulos y duplicados.

HISTORICO LECTURES			
ATRIBUTO	**TIPOS DE DATOS**	**TIPO DE ROL**	**RESTRICCIONES**
codiHistoric	number	Primary Key Unike	Not Null
codiLectures_Historic	number	FK LECTURAS [codiLecturas]	Not Null
Mes	char	FK CLIENT [dniClient]	Null
any	char		Null
consums	decimal		Null
numComptadors	number		Null

Ilustración 63 Estructura y atributos entidad HISTORICO LECTURAS.

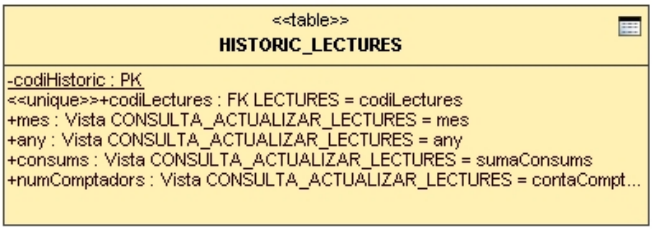


Ilustración 64 Entidad Histórico Lecturas.

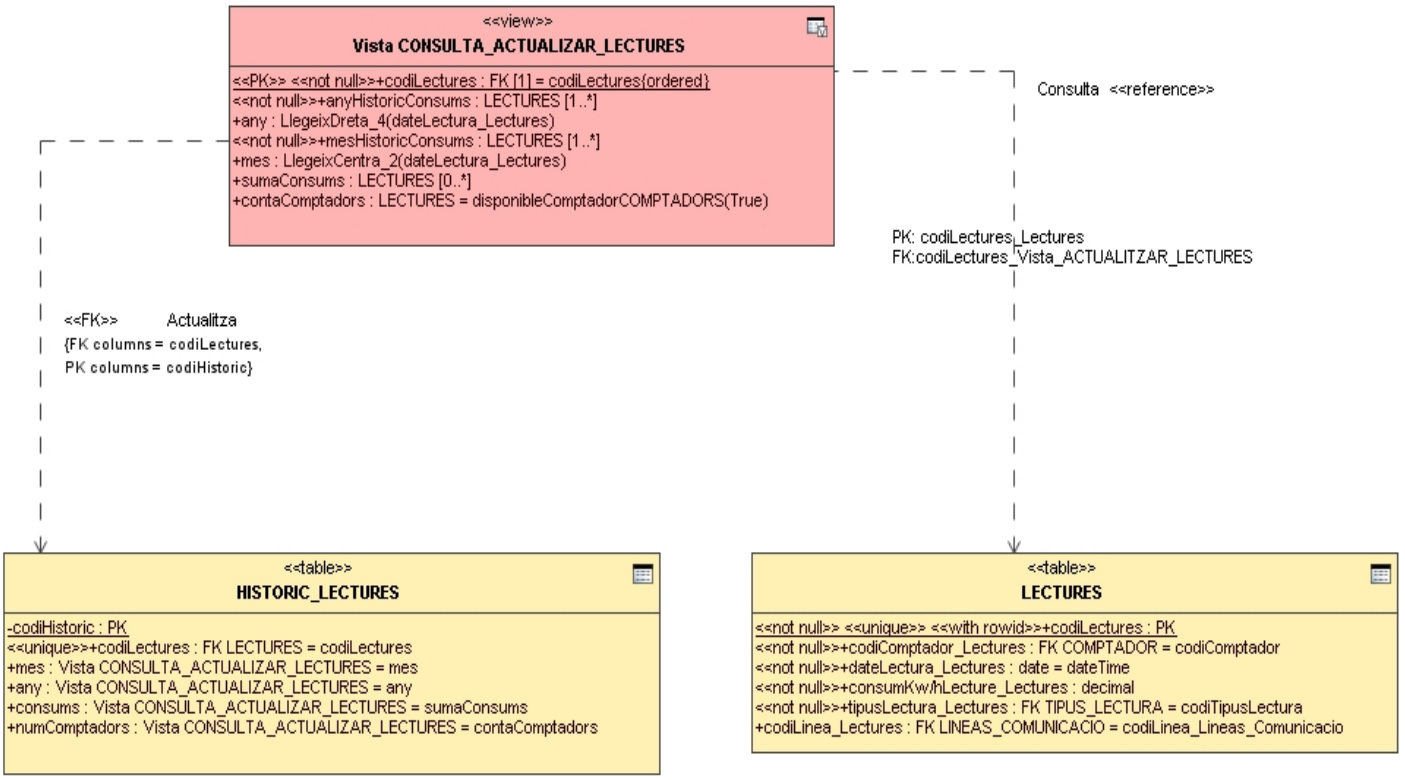


Ilustración 65 Dependencias Entidad HISTORICO CONSUMOS.

2.2.1.4.19 Modulo Estadístico.

Este modulo constituye el análisis del conjunto de la información que contiene el sistema de Base de Datos Relacional del Proyecto.

A nivel de implementación se generará el conjunto de Triggers (o disparadores) necesarios en la Base de Datos, que son los procedimientos se ejecutaran cuando se cumplan una condición establecida al realizar una operación o varias. Por tanto la estructura del Sistema Gestor de Base de Datos contendrá el conjunto de procedimientos que la ejecutarse darán respuesta a las peticiones que efectuará el usuario. De estos procedimientos también se diseñarán los referentes a gestión, manipulación y validación de datos. Esto permitirá no sobrecargar la información que se este tratando.

2.2.1.4.19.1 Modulo Estadístico Consumos Central de Producción Contador.

Este modulo es para obtener el consumo de los generado en la Central de Producción. Para ello se crea una proyección mediante una vista donde concursan las entidades *CONTADOR*, *LINEAS_COMUNICACION* y *LECTURAS.* Es el consumo de los contadores que dependen de una Central de Producción determinada.

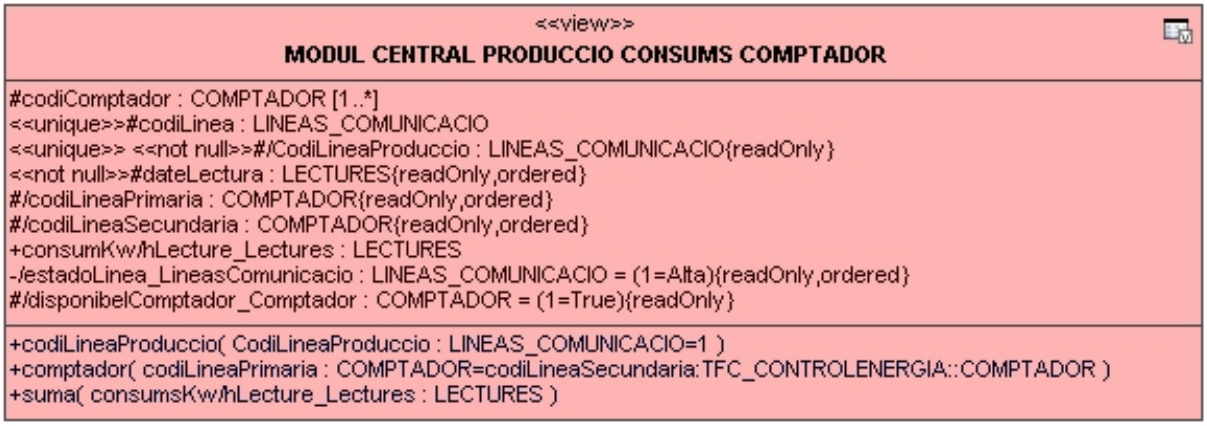


Ilustración 66 Extracción CONSUMOS CENTRAL PRODUCCIO POR CONTADOR.

Este modulo estadístico será ejecutado mediante procedimiento preestablecido que se generará en el proceso de implementación de código en el lenguaje SQL.

El diseño del cual emana el tratamiento de los datos en este proceso es el que puede verse en la ilustración siguiente.

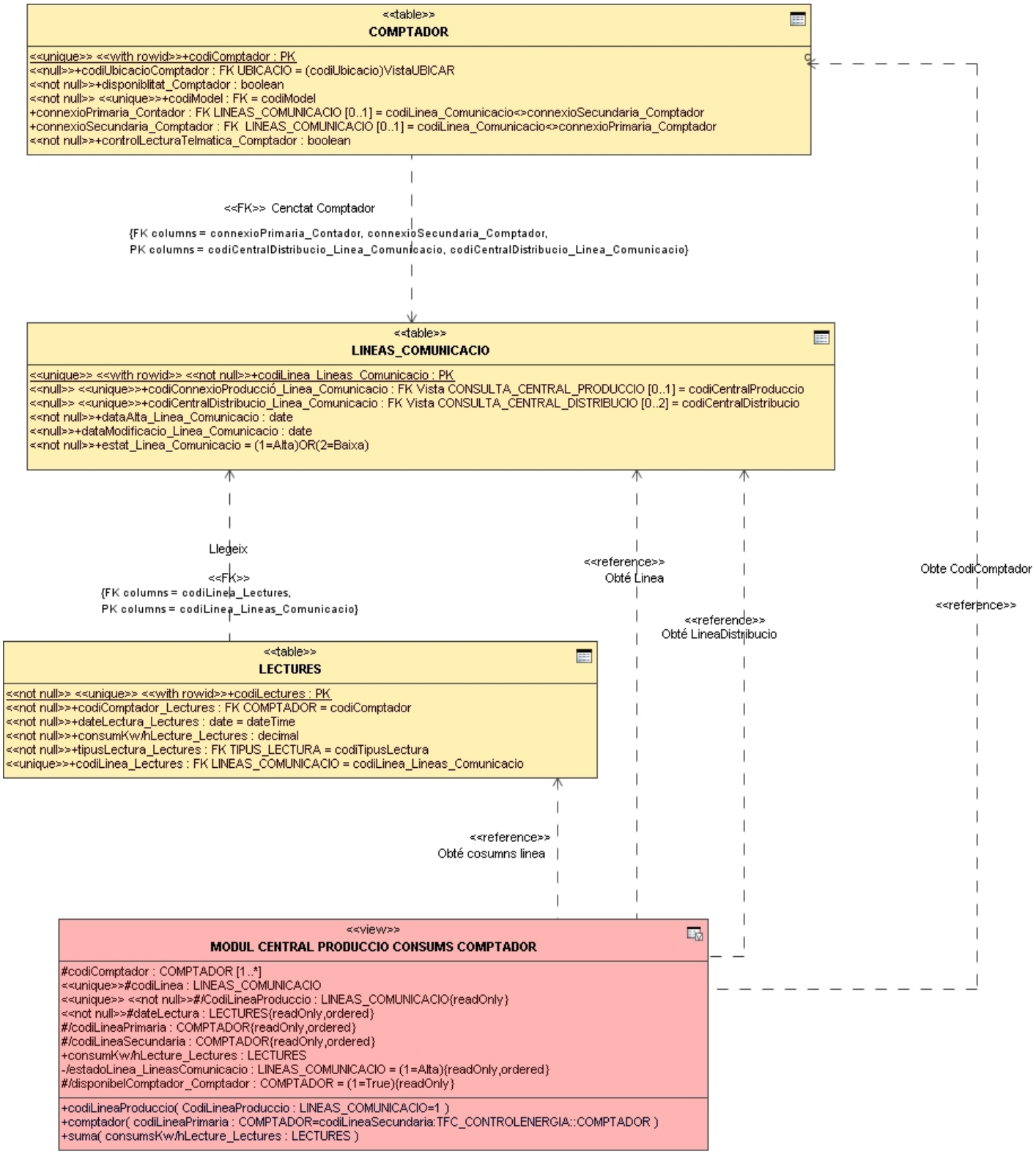


Ilustración 67 Extracción Consumos CENTRAL PRODUCCIÓN.

2.2.1.4.19.2 Modulo Estadístico Promedio Consumos Año Contador Línea.

Corresponde a la extracción en un año pasado por parámetro, la obtención del promedio de la energía consumida por una línea donde la suma de las potencias dividido por dos conexiones que puede tener el contador.

La proyección de basa en la relación existente entre las entidades *CONTADOR, LINEAS_COMUNICACIÓN*, *MODELO* y *LECTURAS, HISTORICO LECTURAS.*

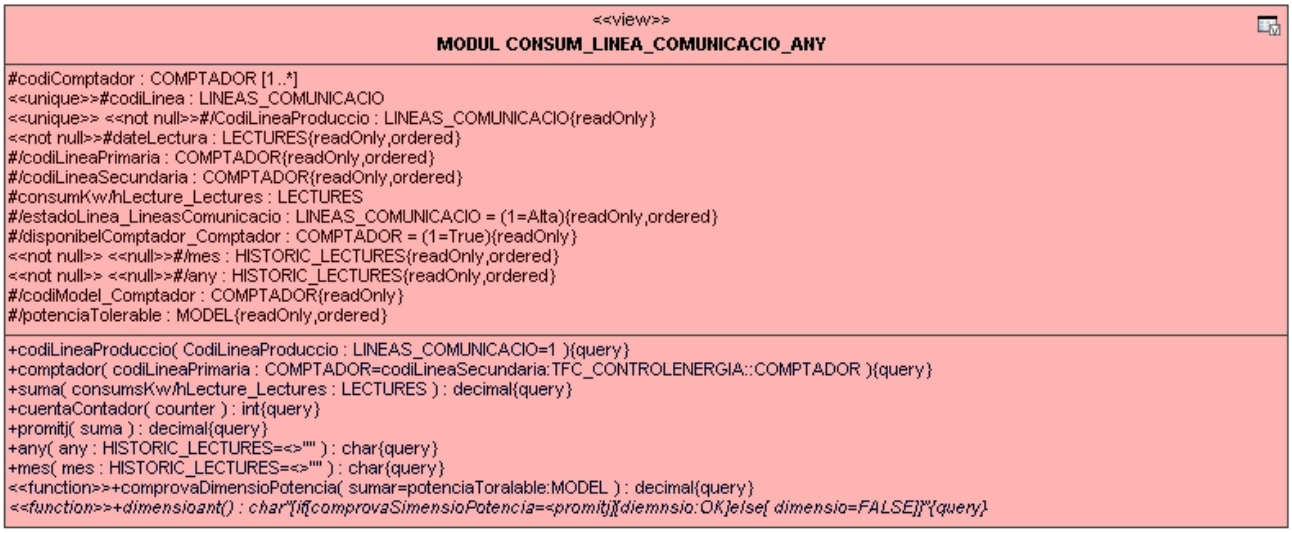


Ilustración 68 Modulo Promedio Consumos Año Contador Línea.

La estructura del flujo de información se origina en la interrelación existente entre las entidades que heredan la información contenida debido a las claves foranas que hay entre las mismas.

La vista que se genera efectúa la suma de los consumos que hay en la entidad *LECTURAS* con la restricción de que en *codiLineaProducció* de la entidad *LINEAS_COMUNICACIÓN* sea igual a 1 (identifica Central de Produción) y de que el *codiComptadorPrimaria* y *codiComptadorSecundaria* de la entidad *CONTADOR* sean iguales, de esta forma nos aseguramos de que poder sumar la *potenciaTolerable* de la entidad *MODELO* donde hay dicha información. El resultado de la suma esta debe de compararse con dicha potencia.

El tratamiento como debe de ser entre un rango, *mes* y *año*, entonces obtenemos la fecha de la entidad *HISTORICO LECTURAS*, previamente actualizada y desconcatenada de *LECTURAS*.

La ilustración siguiente muestra el proceso de dicho tratamiento y las interrelaciones de las entidades que entran en concurso.

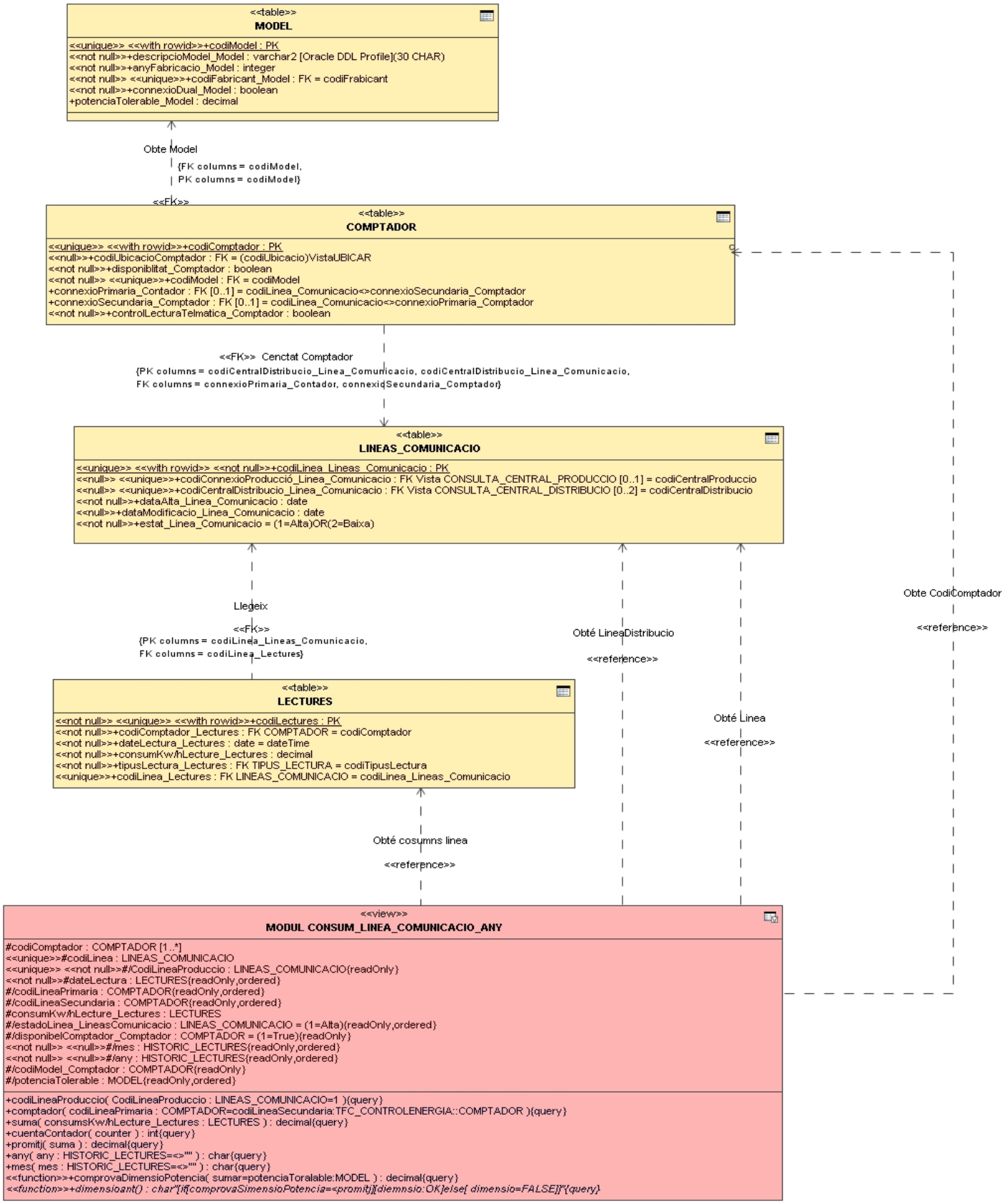


Ilustración 69 Extracción Modulo Promedio Consumos Año Contador Línea.

2.2.1.4.19.3 Modulo Estadístico Máximo Consumo Línea cargada.

Este modulo estadístico su finalidad es encontrar el nivel máximo de consumo generado en una línea, con el fin de saber la máxima carga real existente.

Para ello se utiliza la entidad *LECTURAS* y teniendo en cuenta el *codigoLinea_Lecturas*, se construye una proyección donde la suma de los consumos *consumKwhLecture_Lectures* se la máxima de las mismas.

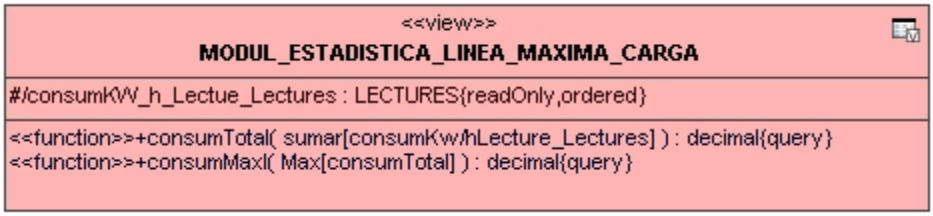


Ilustración 70 Modulo Máximo Consumo Línea.

Los datos que se obtienen están ordenados y la estructura es la se muestra en la siguiente ilustración.

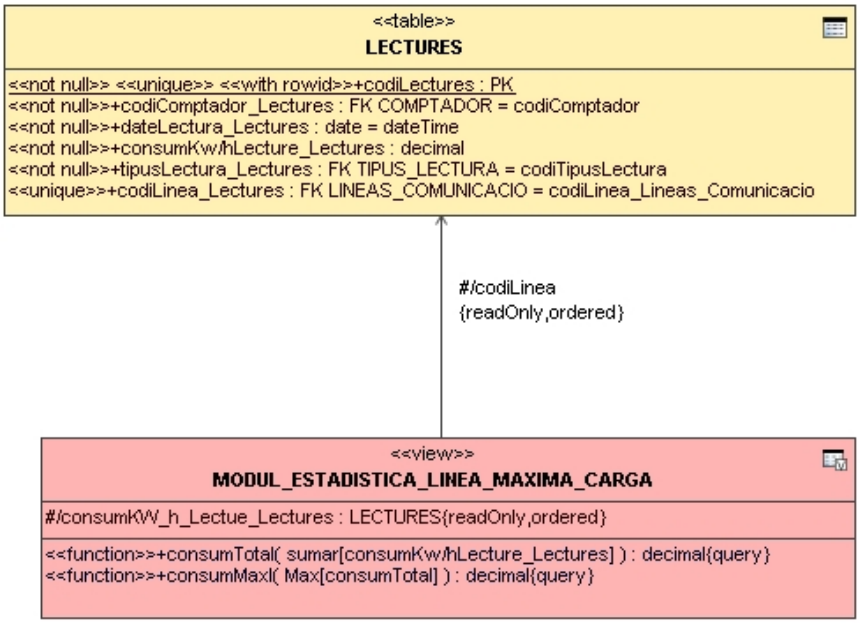


Ilustración 71 Extracción Máximo Consumo, carga de Línea.

2.2.1.4.19.4 Modelo Estadístico Líneas Superiores al 50% Consumido Año.

Esta estadística hace un cálculo para obtener en un año determinado el porcentaje de líneas sea igual o superior al 50% consumido.

Sea diseñado una proyección en la que interviene la entidad *LECTURES*, calculando al numero de líneas que hay, la fecha la cual mediante una función se lee de izquierda a derecha 4 posiciones convertido en texto para obtener el año en concreto que mediante un procedimiento se efectuará la consulta del año deseado, también de crea el atributo mes que mediante una función se lee las 2 posiciones centrales de la fecha y convertidas en texto, se extrae en la misma proyección el consumo de la línea la cual mediante una función suma obtenemos el campo *sumaTotal.*

Del valor obtenido de *sumaTotal* se multiplica por 100 y divide por el número de líneas, obteniendo el porcentaje en formato decimal de dos posiciones, llamándose el nuevo campo *percent.* Este campo último se compara si es mayor o igual *sumaTotal* si es verdadero, el campo supera recibe un bit=1 (booleano) que informa de que es cierto, por lo cual tenemos todas la líneas que superan el 50 % que el campo *numlineaspercent* mediante la función contar se obtiene el numero de líneas afectadas por la restricción que se desea en la extracción.

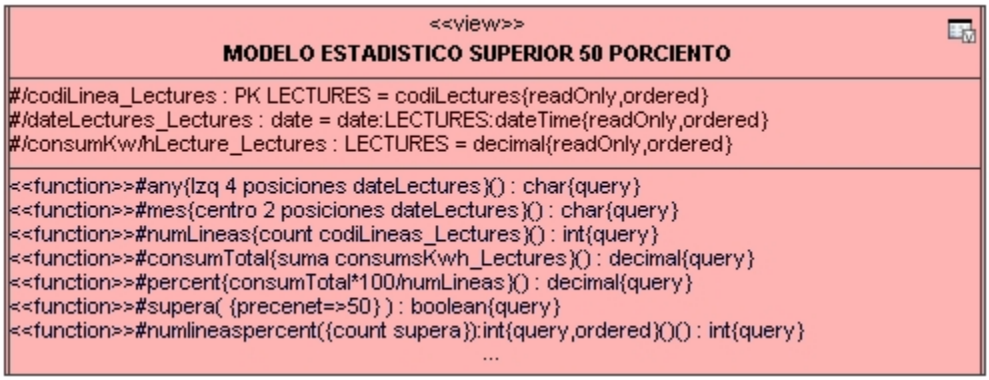


Ilustración 72 Modulo Porcentaje de Líneas en un año Superior al 50% Consumido.

Las entidades que se utilizan en este proceso de extracción es *LECTURAS* la cual contiene a tiempo real las lecturas realizadas. La ilustración muestra el proceso de la vista y la entidad utilizada.

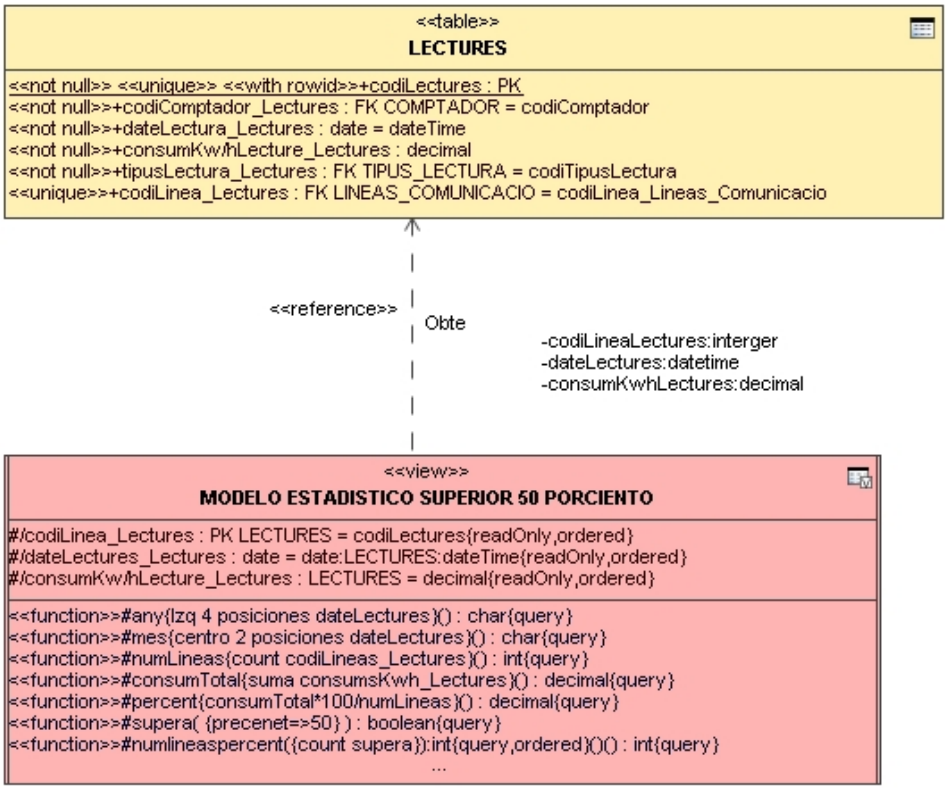


Ilustración 73 Extracción de Líneas en un año Superior al 50% Consumido.

2.2.1.4.19.5 Modelo Estadístico Líneas Inferior al 30% Consumido Año.

El proceso de extracción es similar al anterior, la variación es de que en el procedimiento a utilizar la comparación se realiza por la constante del atributo calculado supera si es menor de la constante 30, si es verdadero luego proyecta los datos correspondientes a la restricción de este valor parametrizado.

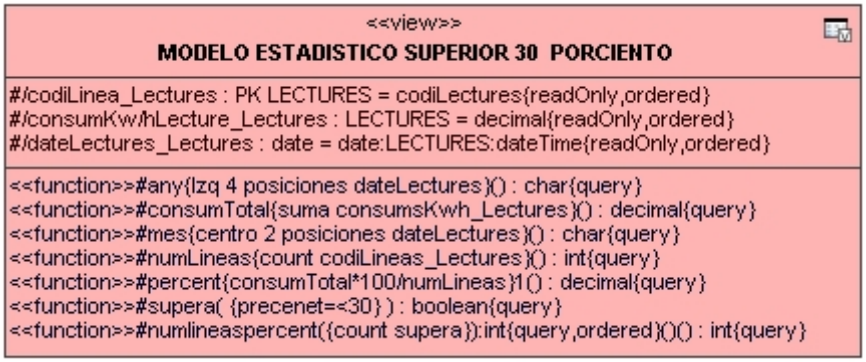


Ilustración 74 Modulo Estadístico Líneas Inferior a 30% Consumido Año.

La ilustración siguiente muestra la entidad *LECTURAS* que es donde hay en estos momentos los valores correspondientes a tiempo real.

Como atributos necesarios tenemos el *codiLectura_Lectures* de la entidad *LECTURAS,* el *consumKwhLecture_Lectures* que contiene las lecturas efectuadas en la línea y *dateLectures_Lectures* que contiene la fecha en que se produce dicha lectura.

La tener que obtener el año *any*, se deberá de desconcatenar el atributo *dateLectures_Lectures* para obtener los 4 *ultimos digitos* de la fecha que es donde hay el año, así mismo deberá de realizarse el mismo proceso para discriminar el *mes* leyendo de izquierda a derecha 4 posiciones para volver a leer de derecha a izquierda 2 posiciones y convertirlo al tipo texto, obteniendo así el mes en el formato comentado.

Para obtener el numero de líneas que deben, es decir contar la líneas, se efectuara el procedimiento de función contar en código SQL, para obtener el valor entero del numero de líneas, que en las restricciones que comentare seguidamente discriminaran el valor que se desea obtener.

El calculo del tanto porciento es la operación matemática que la función debe de realizar y adoptar su valor en el atributo *percent*, donde intervienen los valores heredados de *consumTota*l, este último es la suma de todos los consumos para las restricciones establecidas, anteriores y las que comentare a continuación.

Como restricción *percent* debe de ser igual o menor que 30, si es así retorna un bit=1 que indica el booleano de verdadero, siendo el atributo *supera* quien recibe este retorno.

Contando el atributo supera par obtener el número de líneas que cumplen con los requisitos que sean ido desarrollando para esta extracción.

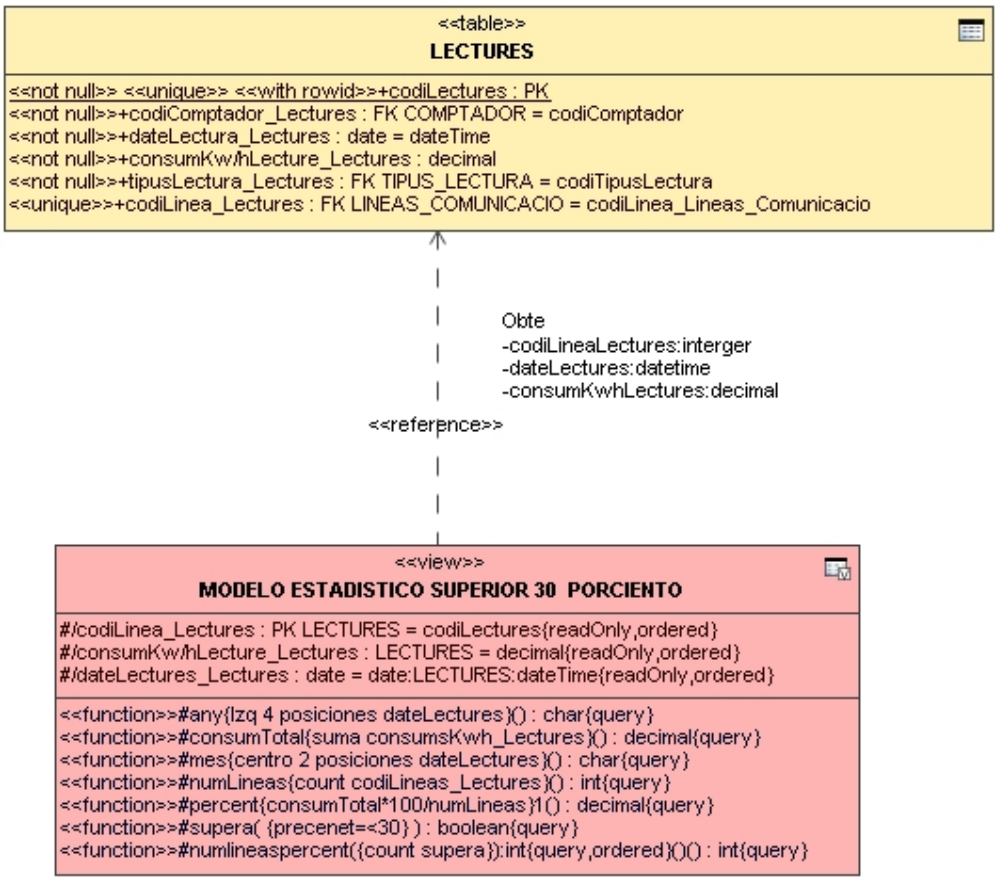


Ilustración 75 Extracción de Líneas en un año Inferior al 30% Consumido.

2.2.1.4.19.6 Modelo Estadístico de los 10 Contadores Máximo Consumo.

Esta extracción se al igual que las demás, deberán de construirse con procedimientos que llamarán la proyecciones correspondientes.

Para su diseño se reutiliza la *vista CONSULTA_ACTUALIZAR_LECTURES* de *ACTUALIZAR LECTURAS* junto con la entidad *LECTURAS* que es donde se puede extraer el *codiLectures* y el *codiComptador_Lectures*, que estos dos atributos son actualizados a tiempo real.

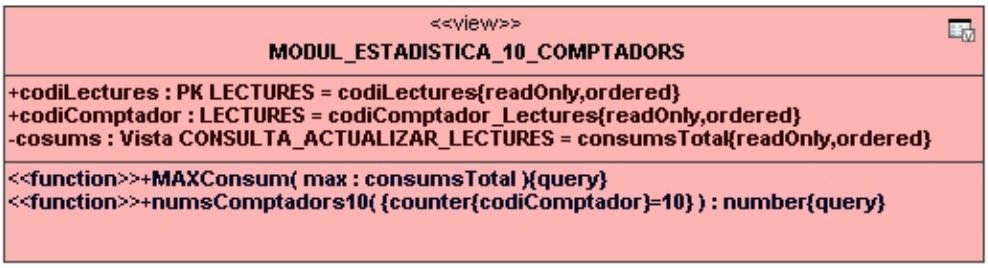


Ilustración 76 Modelo Estadístico de los 10 Contadores Máximo Consumo.

El procedimiento a seguir es de que en la *vista CONSULTA_ACTUALIZAR_LECTURES* tenemos ya de entrada el total de consumos realizados y actualizados a tiempo real hasta el instante en que proyectamos dicha extracción.

Mediante el procedimiento establecido de MaxConsum donde se obtiene para este atributo de tipo decimal, el valor máximo de consumo existente para esta línea y contador que tenemos heredado de LECTURAS.

Como valor calculado necesario para la finalidad que este procedimiento exige, en atributo numsComptador10 recibe mediante la función counter el numero de contadores contados igual a 10. Como restricción que se impone para esta extracción si no coincide con la constante retorna un valor nulo.

La siguiente ilustración muestra el proceso y las entidades relacionadas que intervienen y la proyección de la que se hereda la información a tratar.

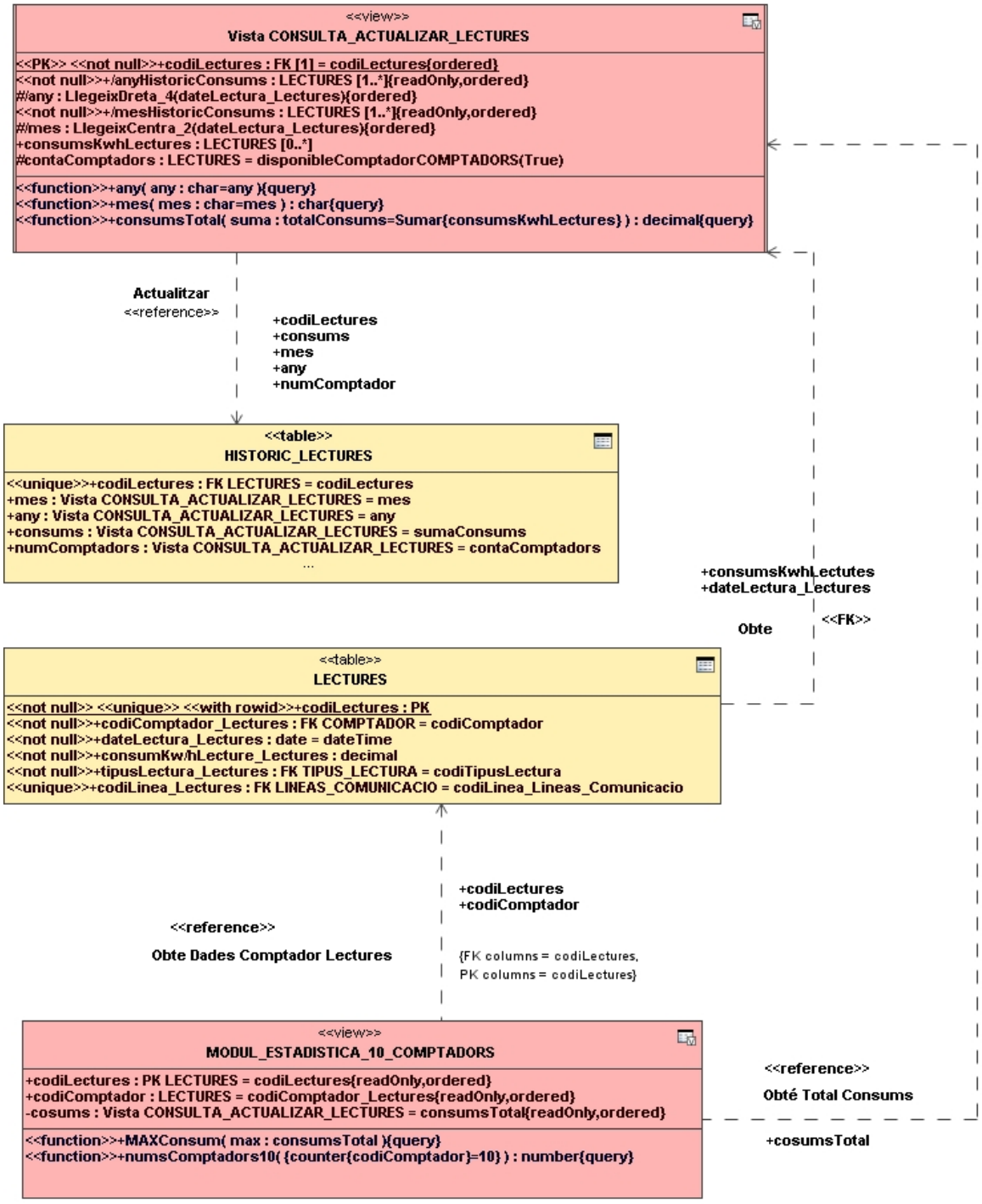


Ilustración 77 Extracción Estadístico de los 10 Contadores Máximo Consumo.

2.2.1.4.19.7 Modulo Estadístico Consumo Medio de Clientes.

Este modulo estadístico calcula el consumo medio de los clientes. Pa ello se genera una vista que podrá ser disparada con un procedimiento implementado.

En los atributos *numsClientes* contiene el numero de clientes en que en *comprovaComptador* el valor booleano es verdadero, en este último se compara el numero de contador que hay en los contratos y el numero de contador que hay en las lecturas, este código de contador que hay en *LECTURAS* esta extraído por la relación existente de *LINEAS COMUNICACIÓN*, ya que el procedimiento de implementación de *ACTUALIZAR LECTURAS* obtiene dicho valor y lo actualiza en esta entidad *LECTURAS*.

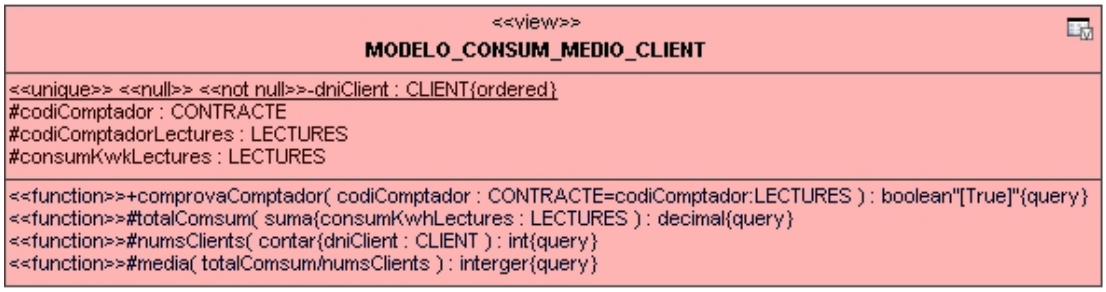


Ilustración 78 Modulo Estadístico Consumo Medio de Clientes.

La ilustración siguiente muestra las entidades y la vista que intervienen en el proceso. De LECTURAS se obtiene el contador y el consumo realizado del contador. De la entidad CONTRACTO se obtiene el código del contador que el cliente tiene asignado y el DNI de cliente que hereda de CLIENTE.

Se realiza la verificación de la restricción en que el código de contador sea el contador de LECTURAS y con la función sumar se extrae el total consumo realizado, que al contar el numero de clientes proyecta la suma de los clientes totalizado.

Al haber obtenido el total consumido y tener el total de lecturas efectuada por cliente sea hace la división de el total por el valor contar, obteniendo el consumo medio de los clientes a nivel global.

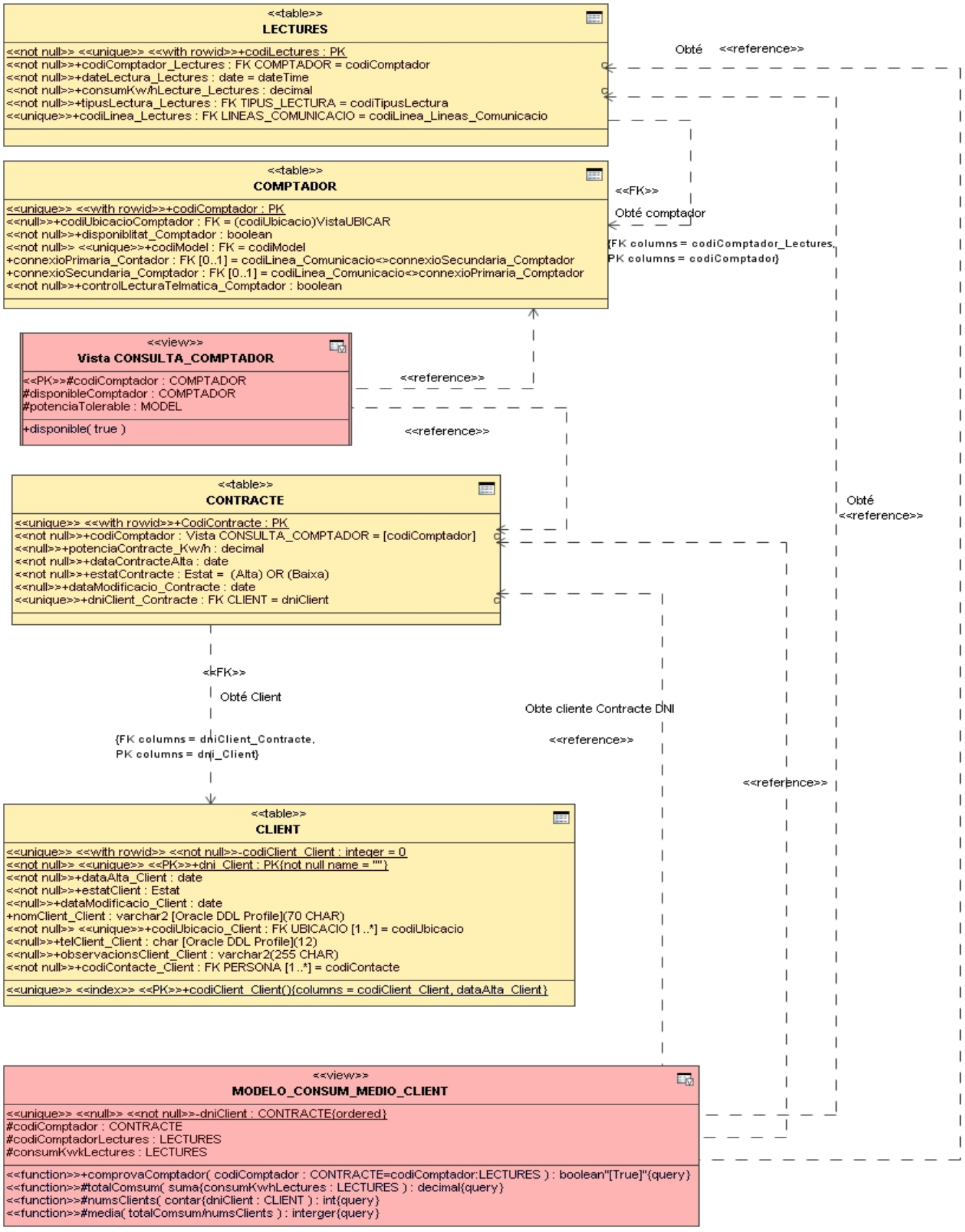


Ilustración 79 Extracción Consumo Medio de Clientes

2.2.1.4.20 Procedimientos LOG.

Contiene el registro del conjunto de llamas a procedimientos existentes dentro de SGBD. Podría decirse en lenguaje formal que es el cuaderno de bitácora del Sistema Gestor de la Base de Datos Relacional.

Cuando se procede la ejecución de cualquier proceso en el SGBD se implementan la siguiente información:

2.2.1.4.20.1 idLog

Numero identificativo, es un número entero incrementable no duplicado que identifica internamente el proceso ejecutado.

2.2.1.4.20.2 procesLog

Nombre del procedimiento ejecutado, es de tipo texto, no admite valores nulos.

2.2.1.4.20.3 parametreEntrada

Parámetro de entrada de la ejecución, no admite valores nulos..

2.2.1.4.20.4 parametreSortida

Parámetro de salida, no admite valores nulos.

2.2.1.4.20.5 RSPLog

Mensaje de tipo booleano, el bit=1 verdadero OR BIT=Falso del cual dependiendo de su valor equivale a la muestra:

1= OK

0=“ERROR”+” tipo de error”

2.2.1.4.20.6 dateLog

Fecha y hora de ejecución, no admite valores nulos, es de tipo dateTime.

Entendiéndose como RSP el definido por el enunciado, el cual especifica de forma clara un indicador de resultado del comportamiento de la ejecución, correcto o incorrecto. Dicho comportamiento se traduce en la visualización por el sistema de un mensaje de tipo String que

mostrará “OK” si la respuesta es correcta o bien la de “ERROR”+” tipo de error” en caso de que el comportamiento sea incorrecto o coexista algún error en la ejecución del tratamiento de datos del SGBD.

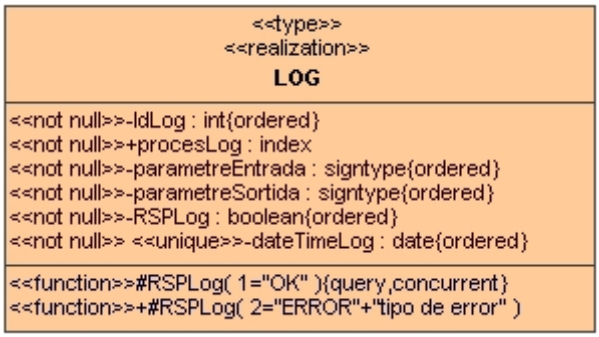


Ilustración 80 Procedimientos LOG.

2.2.2 Relaciones.

Son conjunto de enlaces fruto de las claves primarias y foranas, de esta relación de integridad referencial y de tipo en cascada. Da como funcionalidad el poder hacer las extracciones parametrizada mediante las procedures o bien por acceso directos a las SELECT o consultas que proyectan las vistas requeridas.

De este proyecto sea podido detectar las siguientes relaciones con las diferentes tablas que intervienen en las mismas.

Todas estas relaciones de gestión de datos, tienen en muchos casos la finalidad de insertar, actualizar y mantener información dentro el sistema gestor de base de datos. Por tanto cada entidad deberá de tener un proceso específico para poder ejecutar estas funciones comentadas. Algunos de los procesos se hallan inmersos en dependencias de otros procedimientos de llamadas a segundos procedimientos o a ejecutar vistas interdependientes entre ellas.

Siempre en estos tipos de gestiones definidos se sigue el rigor en las reglas de normalización de base de datos, teniendo presente que la divisibilidad de las tablas al máximo nivel es una optimización exigible, aunque esto puede implicar que se ralentice las transacciones que se hagan, es decir el tiempo de latencia de respuesta puede ser más elevado, pero el nivel de seguridad y de consistencia de datos es muy elevado.

2.2.2.1 Gestión Localidades.

Esta gestión actualiza o modifica datos de las Localidades.

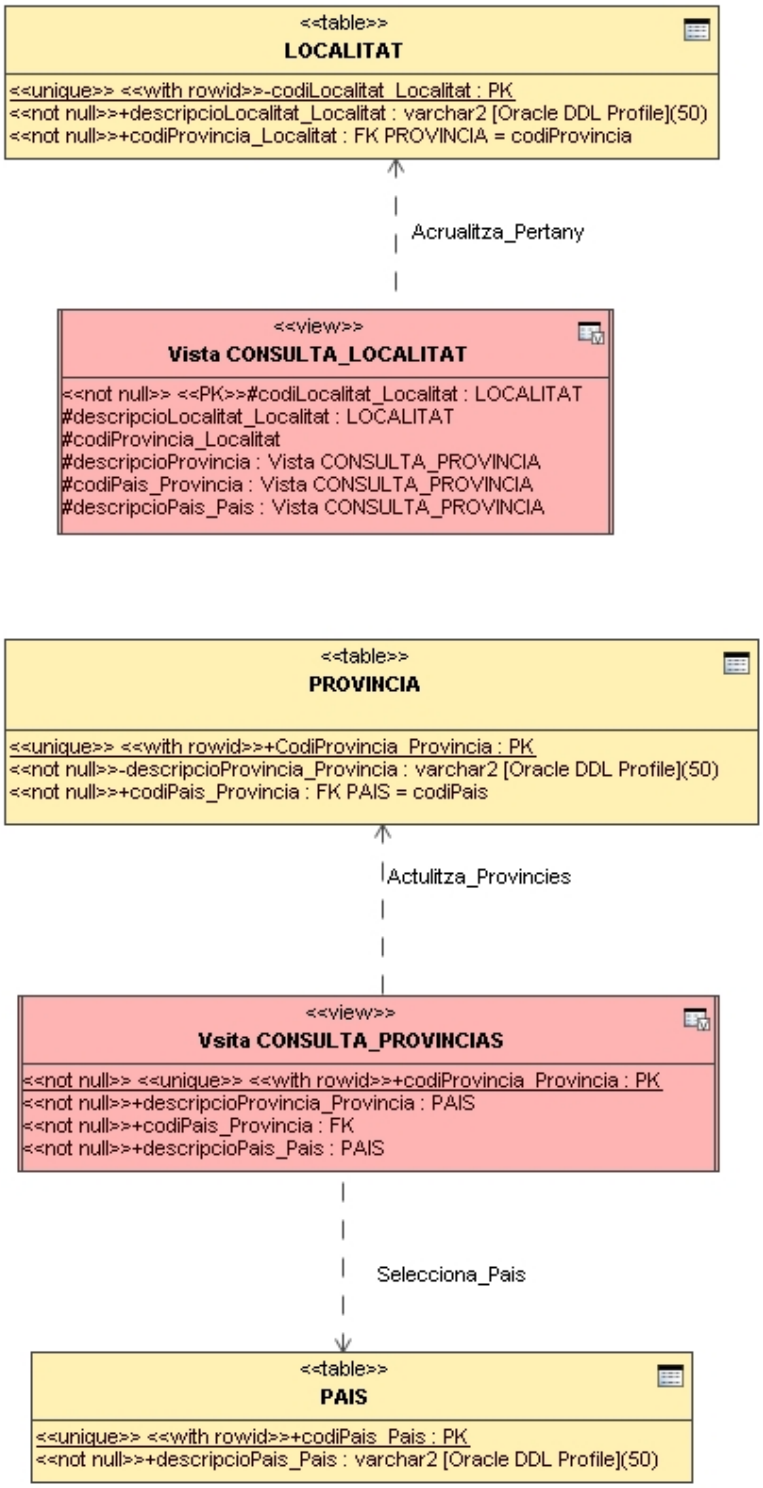


Ilustración 81 Gestión Localidades

2.2.2.2 Gestión Ubicaciones.

Gestiona las altas y las modificaciones de las ubicaciones, este proceso dependerá el poder ubicar las entidades susceptibles de contener la población física donde estén emplazadas.

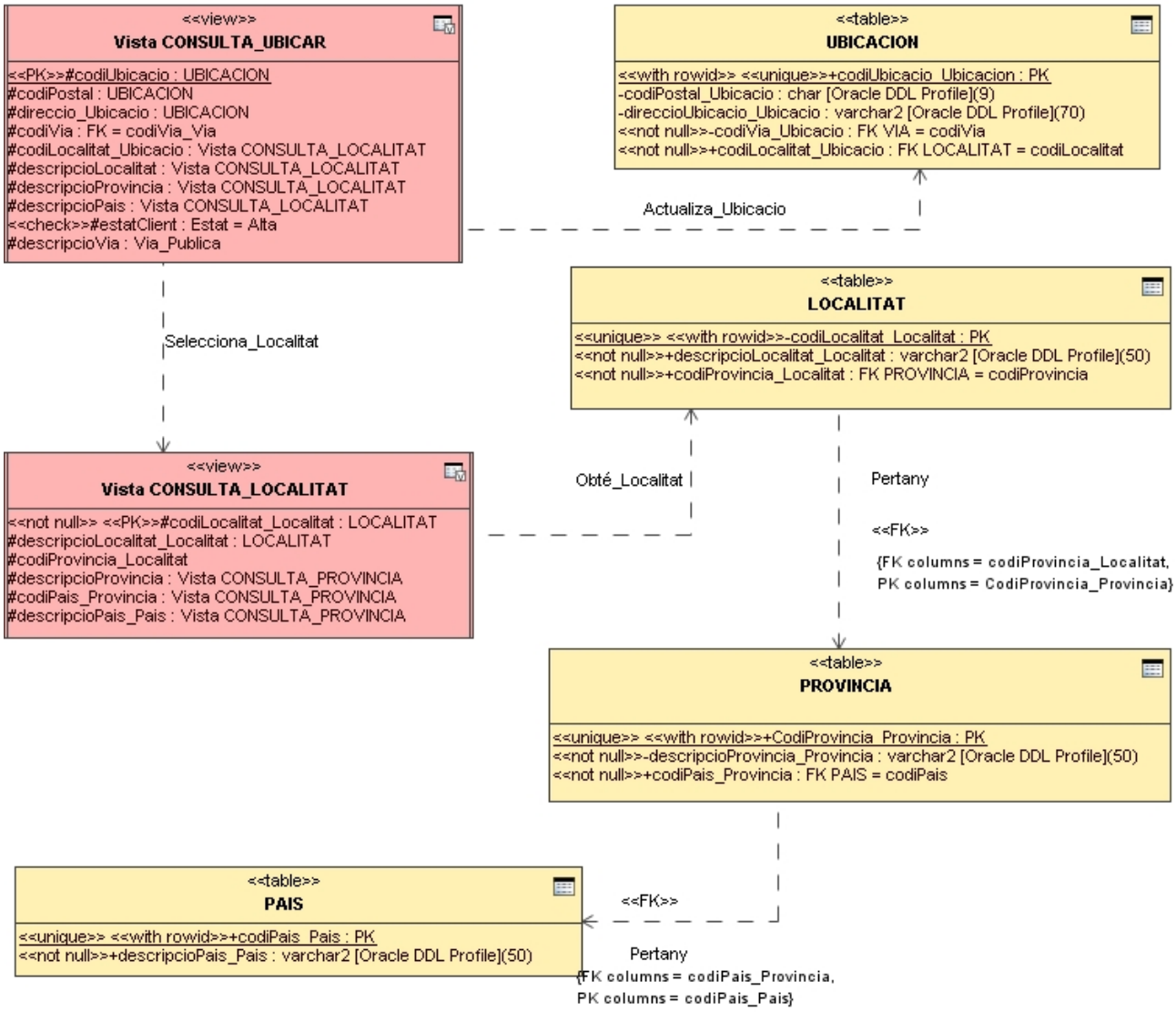


Ilustración 82 Gestión Ubicación.

2.2.2.3 Gestión Cliente.

Es la entidad que identifica las características que en el mudo real describen a un cliente. Este tipo de gestión debe de garantizar el dar altas, modificaciones e insertar clientes en la entidad clientes mediante las tablas dependientes que precisa para su existencia como cliente.

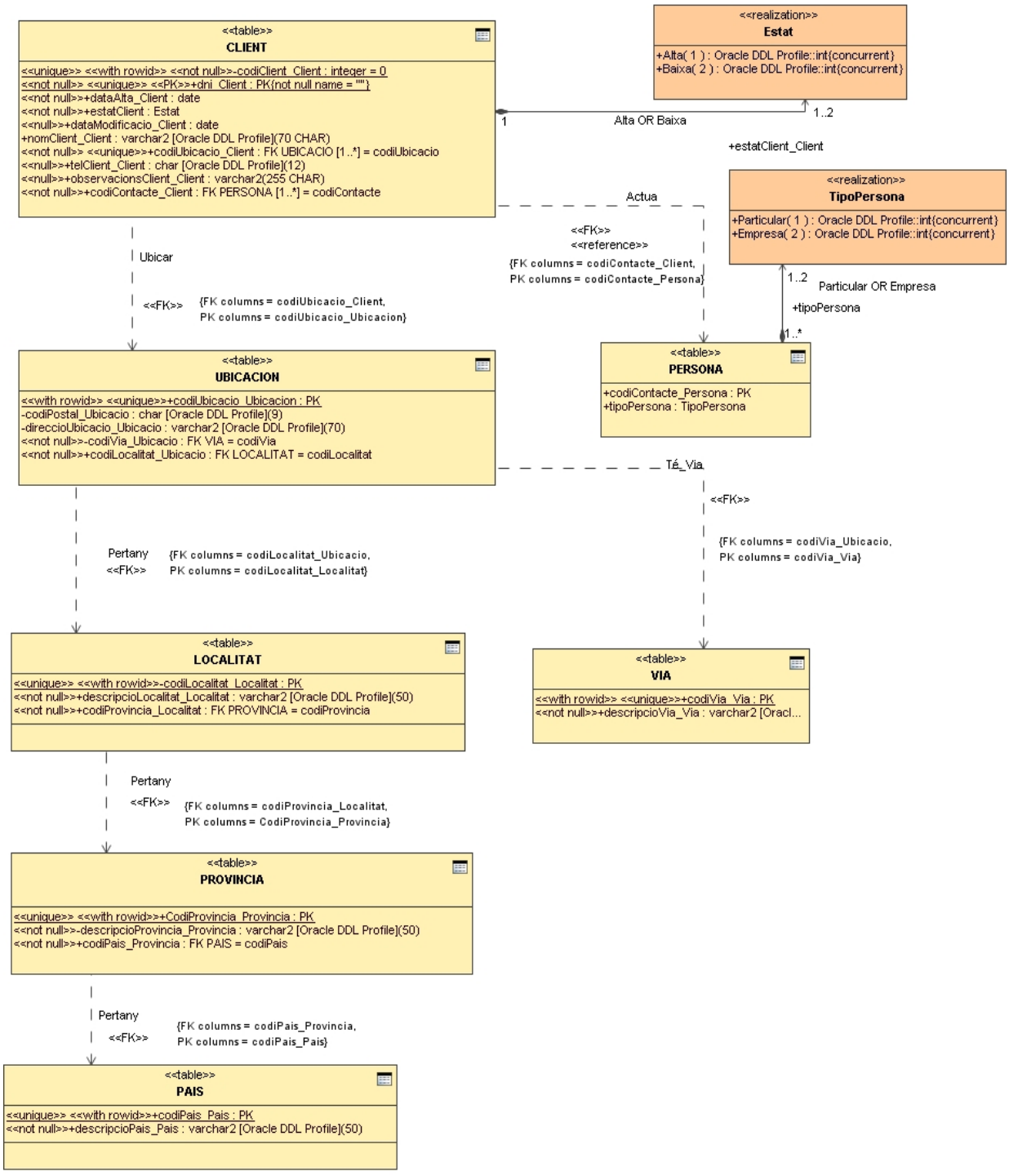


Ilustración 83 Gestión Cliente.

2.2.2.4 Gestión Fabricante.

Es la entidad que identifica las características que en el mudo real describen a un fabricante. Este tipo de gestión debe de garantizar el dar altas, modificaciones e insertar fabricantes en la entidad fabricantes mediante las tablas dependientes que precisa para su existencia como fabricante.

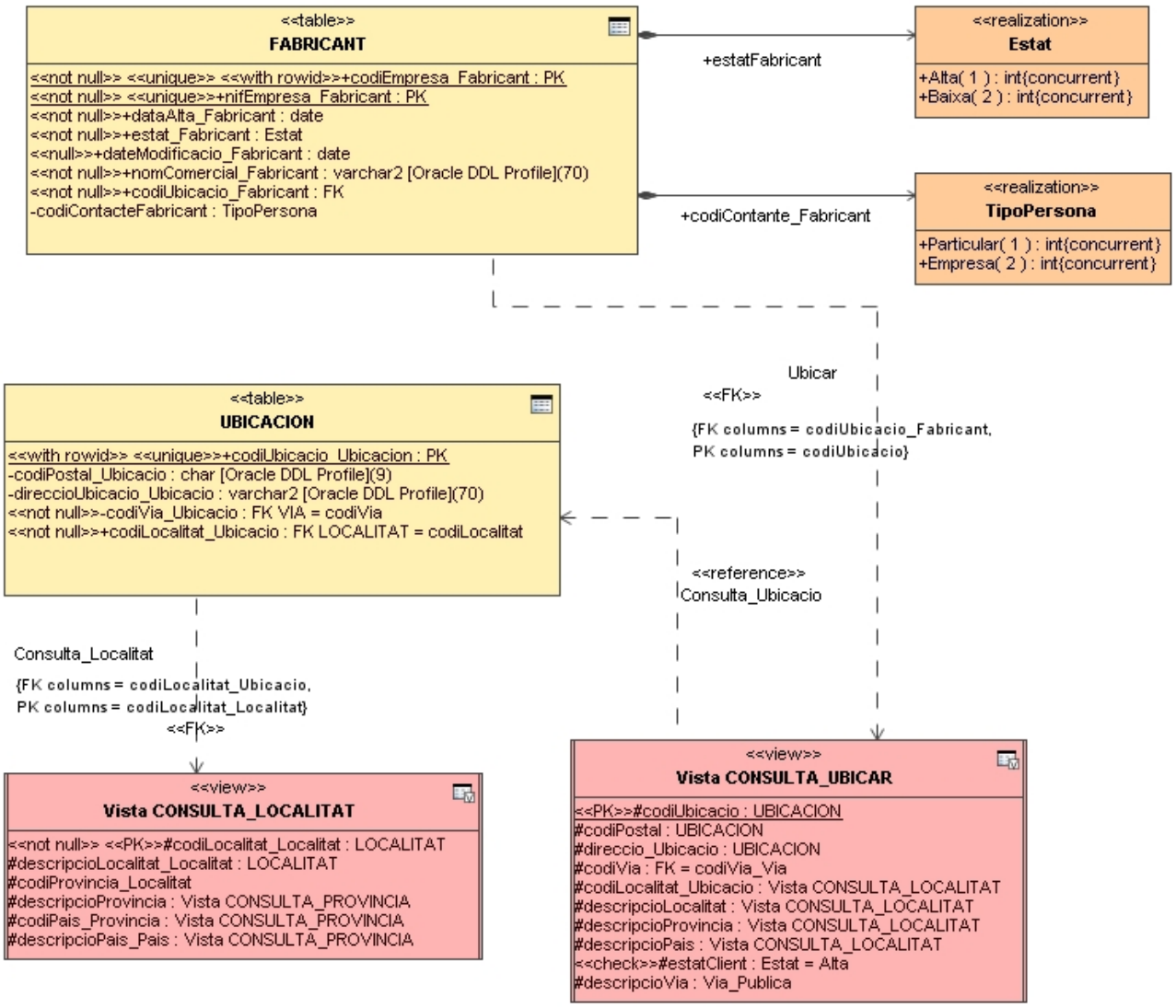


Ilustración 84 Gestión Fabricante

2.2.2.5 Gestión Inspecciones.

Gestiona las inspecciones realizadas, debe de permitir dar altas, modificaciones, de ella depende una vista capaz de relacionar las tablas precisas dependientes de este proceso.

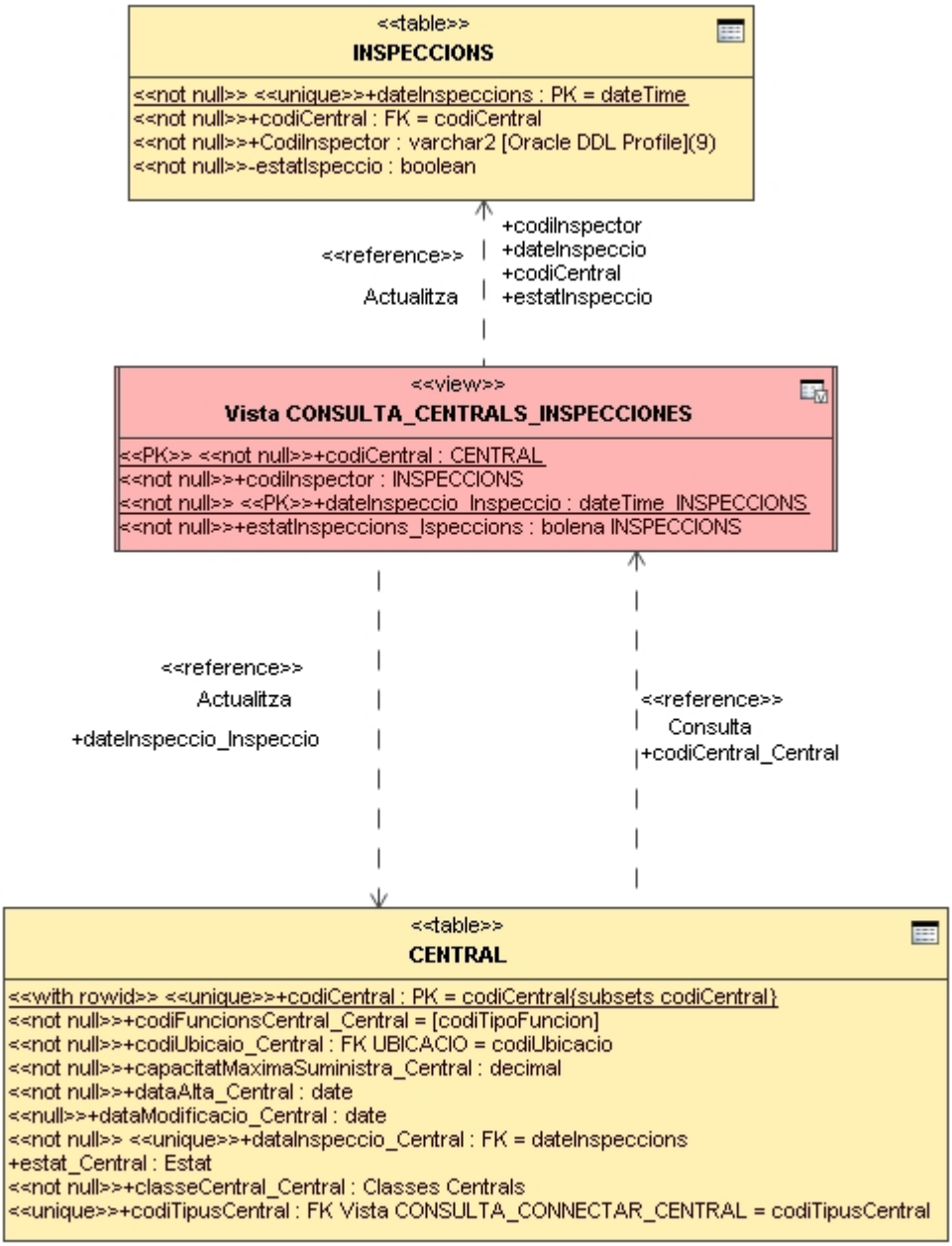


Ilustración 85 Gestión Inspecciones.

2.2.2.6 Gestión Contadores.

Gestiona los contratos del sistema, es una gestión compleja que precisa de varias tabla que su información es heredada por medio de una vista y la dependencia de la tabla que define el tipo de contador que físicamente en el mundo real es instalado en la red energética. Su característica fundamental es de que contiene el acceso mediante dos atributos a la conexión primaria y a la secundaria de la tabla línea siendo sus valores diferentes entre si. Su finalidad es de asegurar su conectividad.

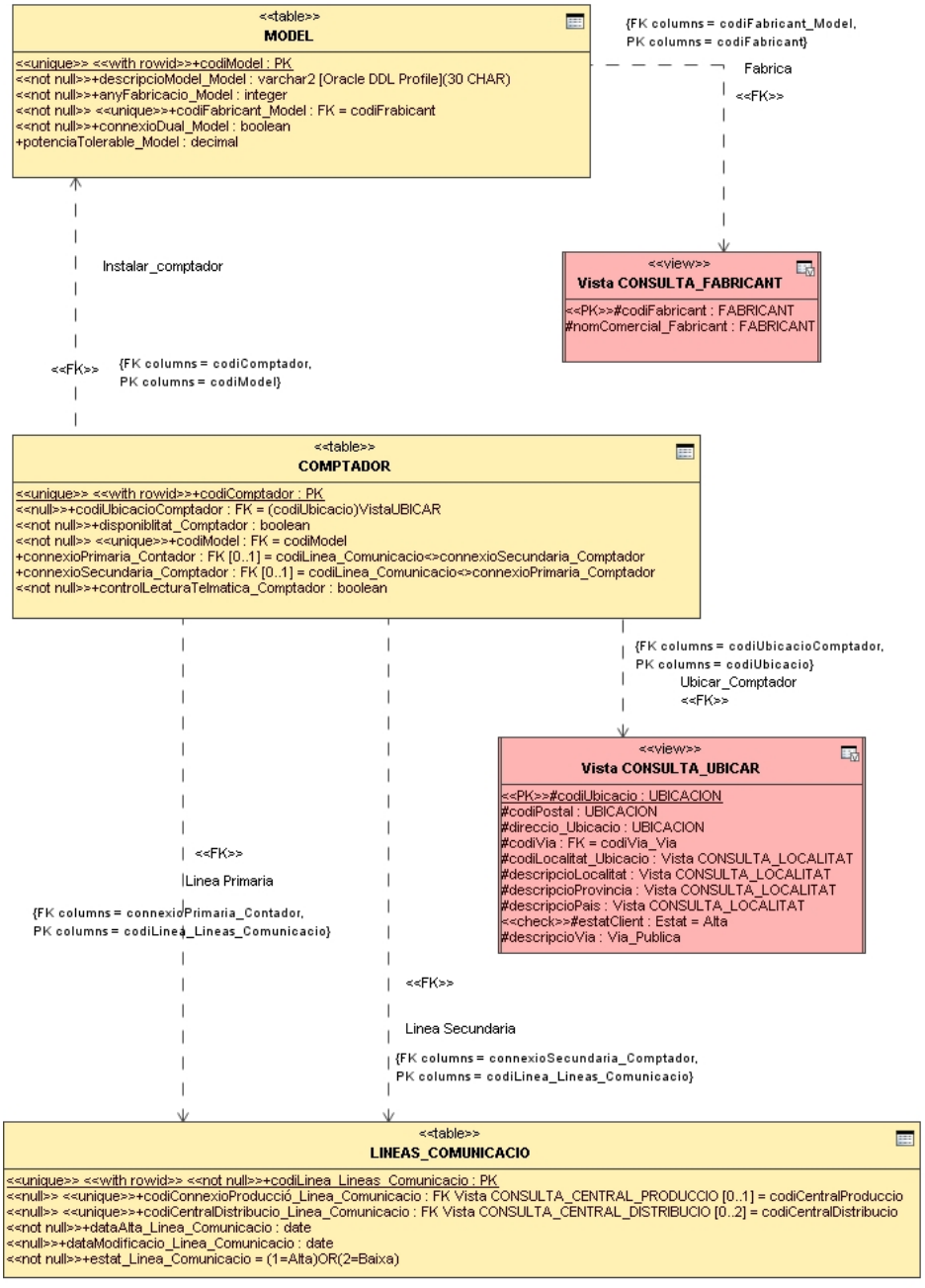


Ilustración 86 Gestión Contadores

2.2.2.7 Gestión Modelos.

Mediante este gestor damos de alta los modelos de contadores, en los modelos hay la potencia tolerable que cada contador dispone y el fabricante que lo suministra.

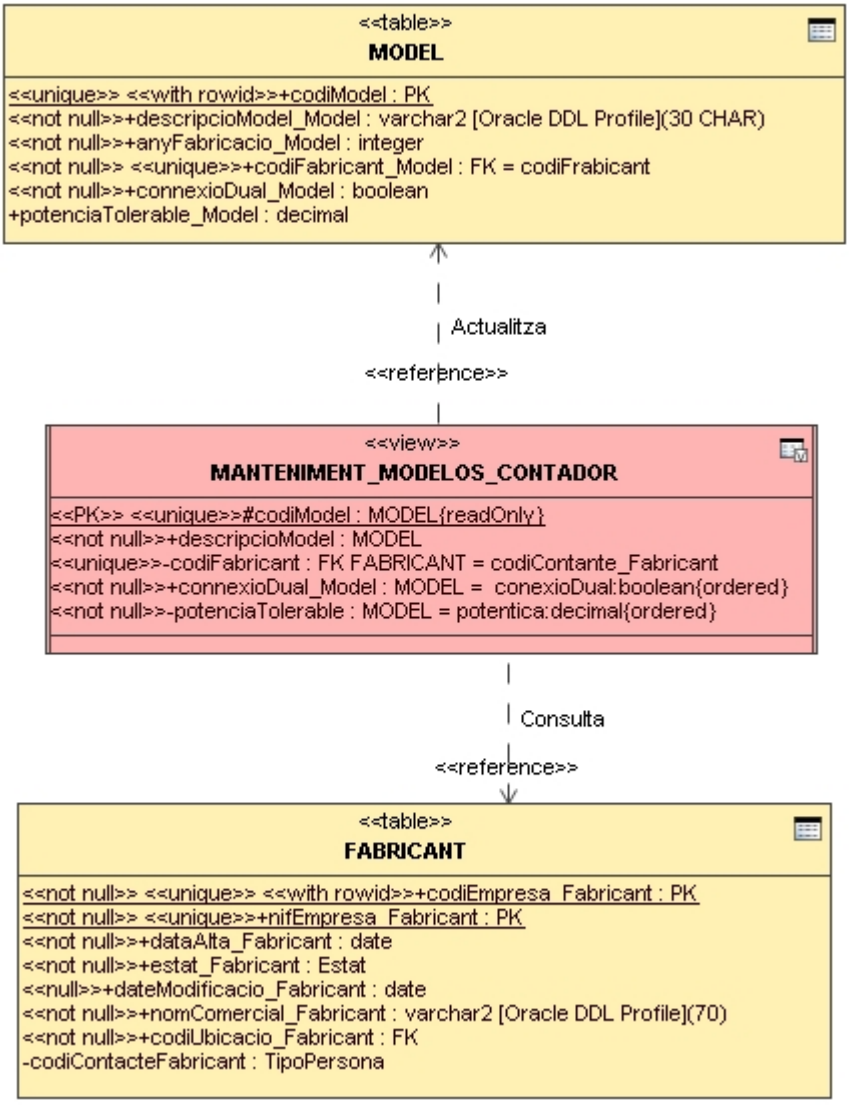


Ilustración 87 Gestión Modelos Contadores

2.2.2.8 Gestión Contratos.

Este tipo de proceso debe ser capaz de asignar un contrato a un cliente, teniendo presente que un cliente puede tener más de un contrato asignado diferente y de que cada contrato tiene un solo contador y solo uno.

Este proceso se nutre de las vistas correspondientes que heredan de las tablas relacionadas por medio de las claves primarias y foranas que intervienen.

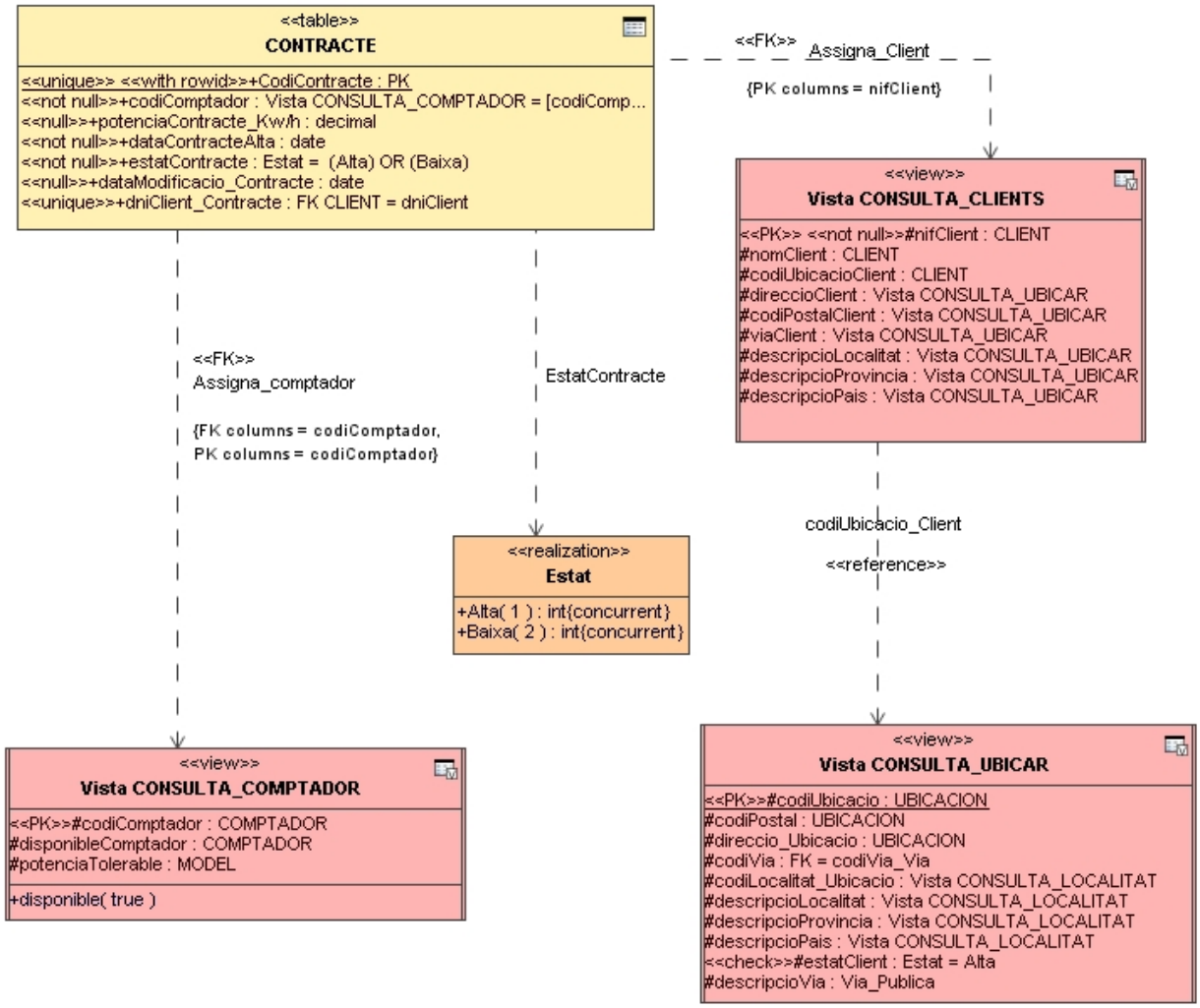


Ilustración 88 Gestión Contratos.

2.2.2.9 Gestión Selección Conectar Centrales.

Este gestor conecta las centrales de producción con las de distribución, la restricción es de que una central de producción no soporta a más de una central de distribución.

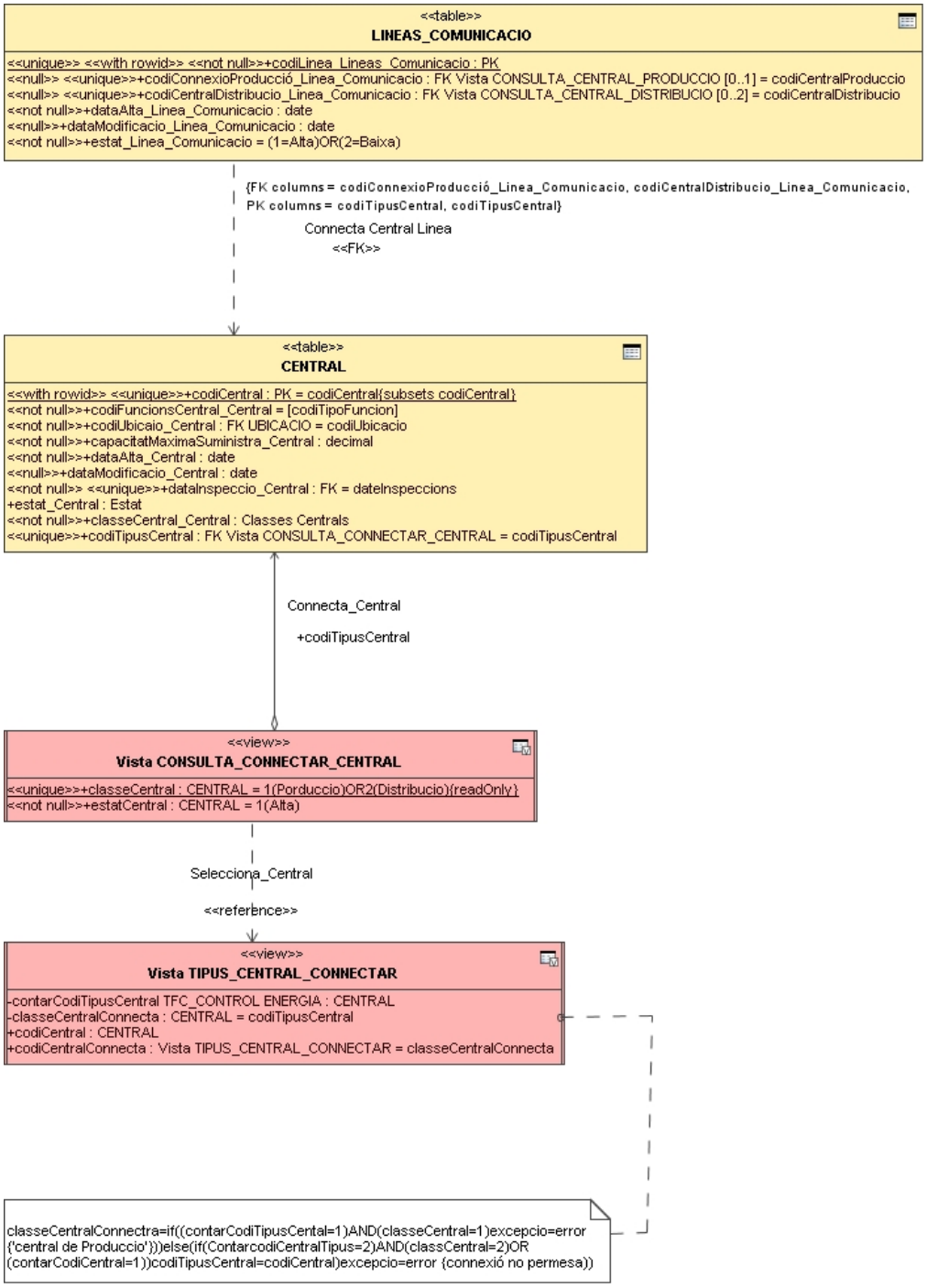


Ilustración 89 Gestión Selección Conectar Centrales

2.2.2.10 Gestión Central.

En este gestor, se procede a realizar las altas de las Centrales. La entidad Central tiene una recursividad y depende de una generalización que le atribuye las funcionalidades de la central dependiendo del tipo de energía que produce. También controla los estados de alta o modificaciones que se haya producido.

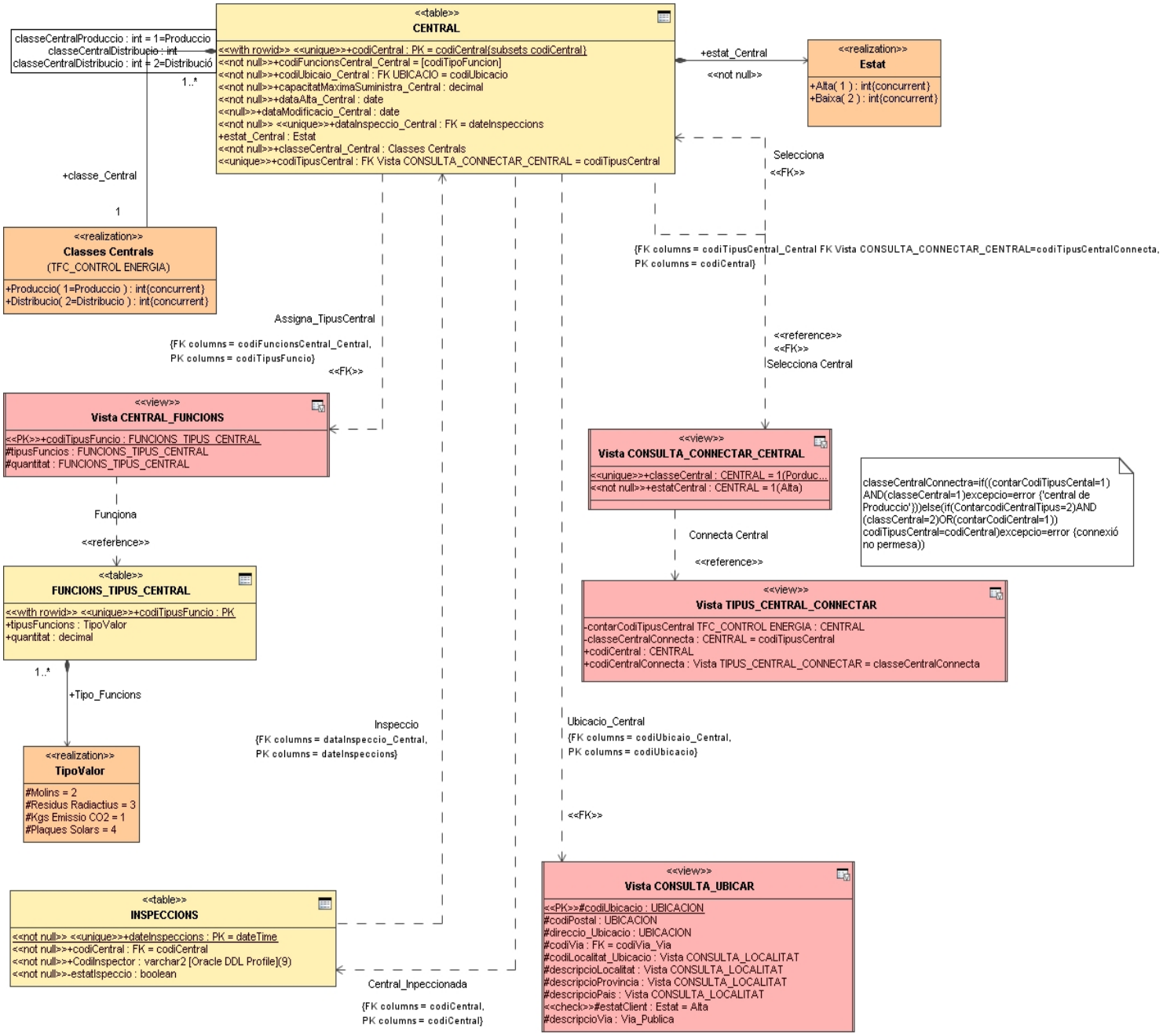


Ilustración 90 Gestión Centrales

2.2.2.11 Gestión Funciones Central.

Este gestor hace el mantenimiento de la tipología de centrales a nivel de suministro energético mediante la entidad Funcione Tipo Central. Para ello se utiliza una vista de proyección de tipos de central para asignar esta funcionalidad a la misma.

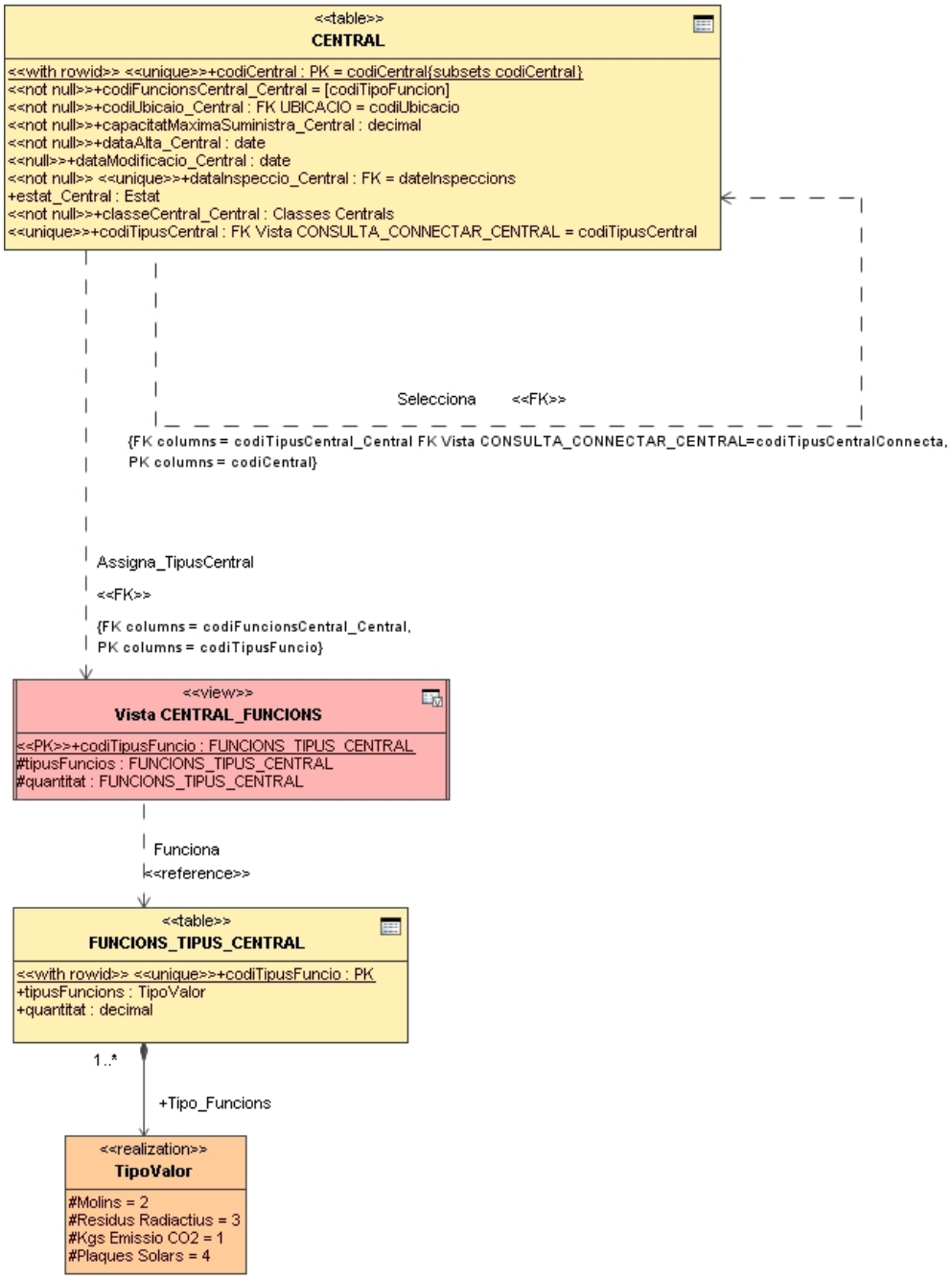


Ilustración 91 Gestión Funciones Central.

2.2.2.12 Gestión Líneas Comunicación Centrales.

Es el gestor que controla la conectividad entre Centrales de Producción y Distribución con las restricciones preestablecidas.

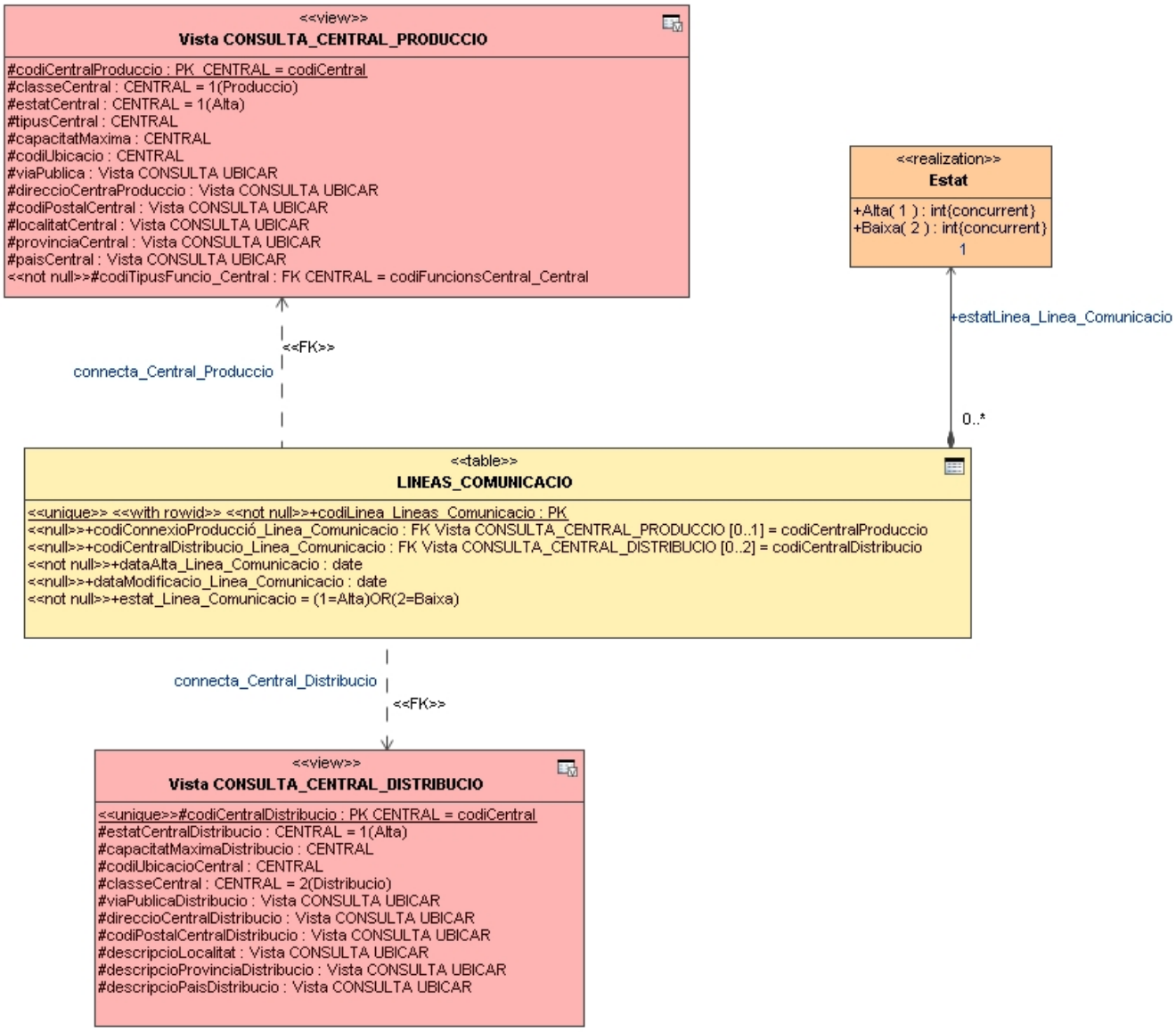


Ilustración 92 Gestión Líneas Comunicación Centrales.

2.2.2.13 Gestión Líneas Comunicación Contadores.

Efectúa las altas y controles de conectividad de los contadores con las línea de comunicación en estado disponible y de carga disponible, la conectividad de un contador tiene la asignación de dos líneas de las que recibe suministro energético.

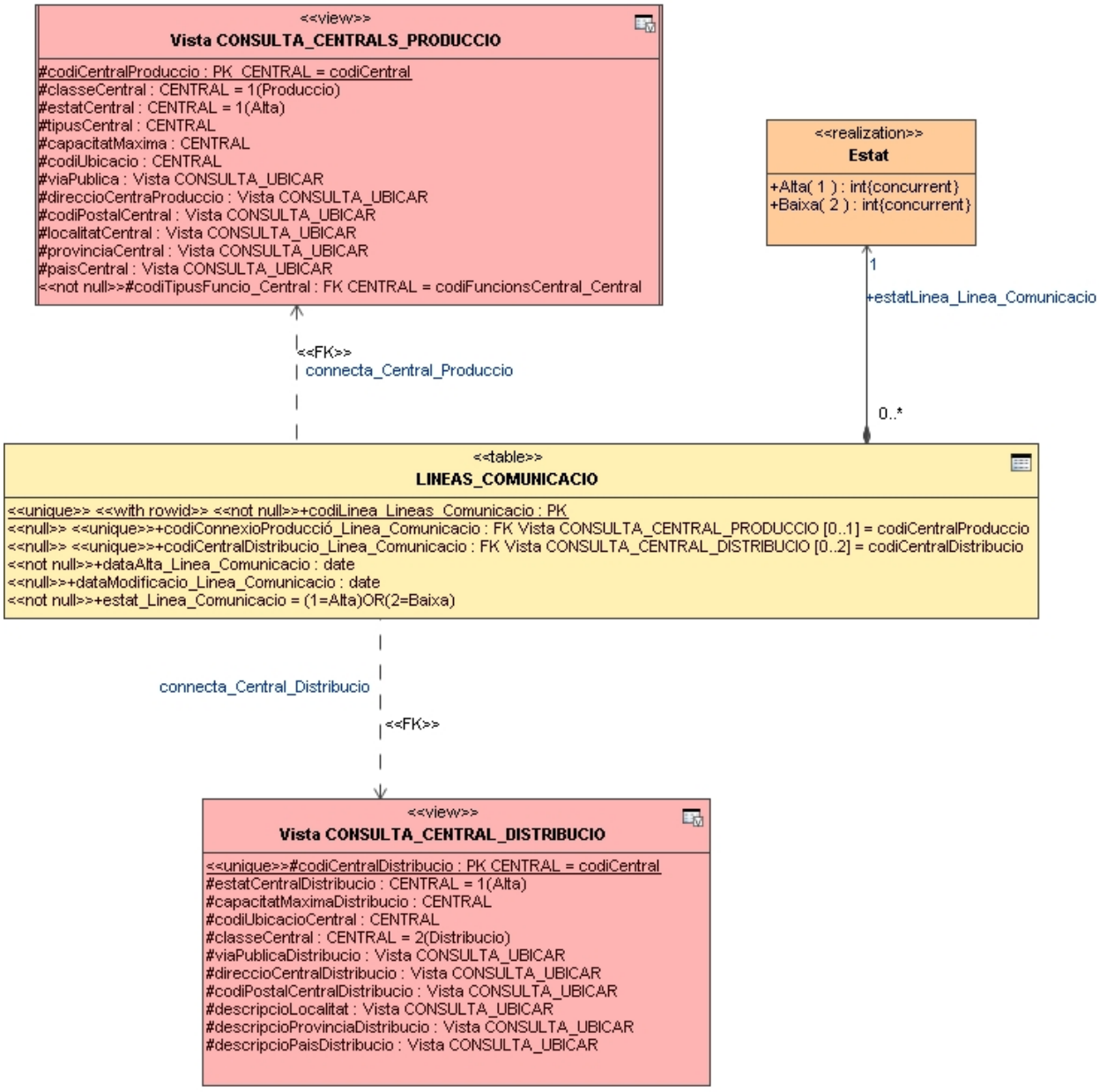


Ilustración 93 Gestión Líneas de Comunicación Contadores.

2.2.2.14 Gestión Lecturas.

Este gestor obtiene las lecturas de los contadores, de el se obtiene el consumo por línea y por consiguiente se puede extraer por herencia el consumo de una Central de Producción y de Distribución. Al disponer de fechas de las extracciones que se van actualizando a tiempo real, se puede conocer mediante proyecciones definidas el consumo a nivel de clientes, modelos de contador, contadores, etc., incluso el tipo de lectura efectuada.

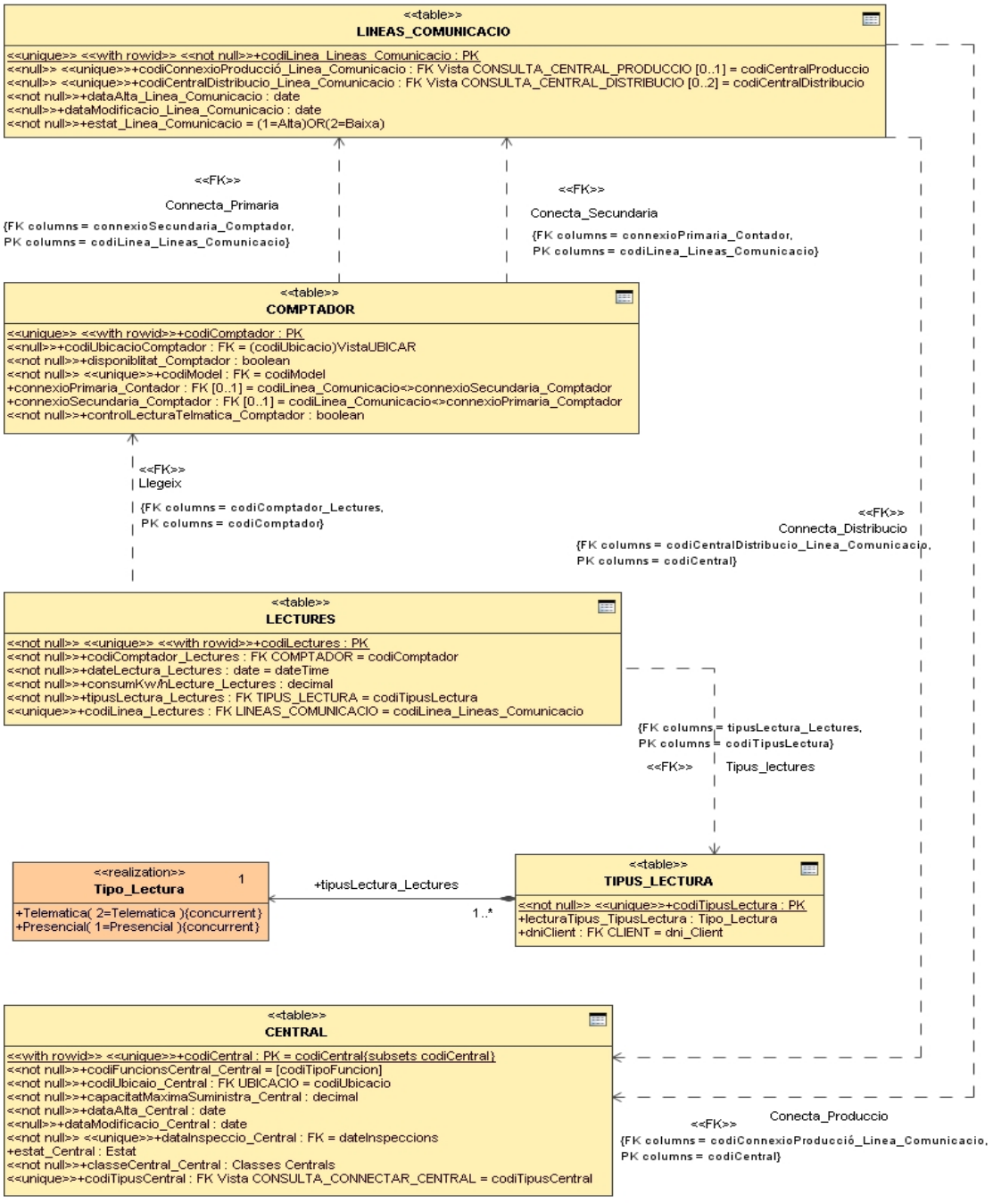


Ilustración 94 Gestión Lecturas.

2.2.2.15 Actualizar Lecturas HISTORICO LECTURAS.

Este gestor actualiza las lecturas al histórico de consumos. La finalizad es de poseer la información más estructurada y susceptible de ser tratada según las peticiones que los usuarios necesiten.

De esta entidad, se pueden extraer información de los módulos estadísticos previa ejecución de esta actualización de lecturas. Una de las eficacias es de que implica un nivel de seguridad en el almacenamiento de datos críticos y la opción de poder extraen información optimizando el tiempo de proceso según el tipo de extracción que el usuario precisa en el ámbito de la temporalidad de la información contenida.

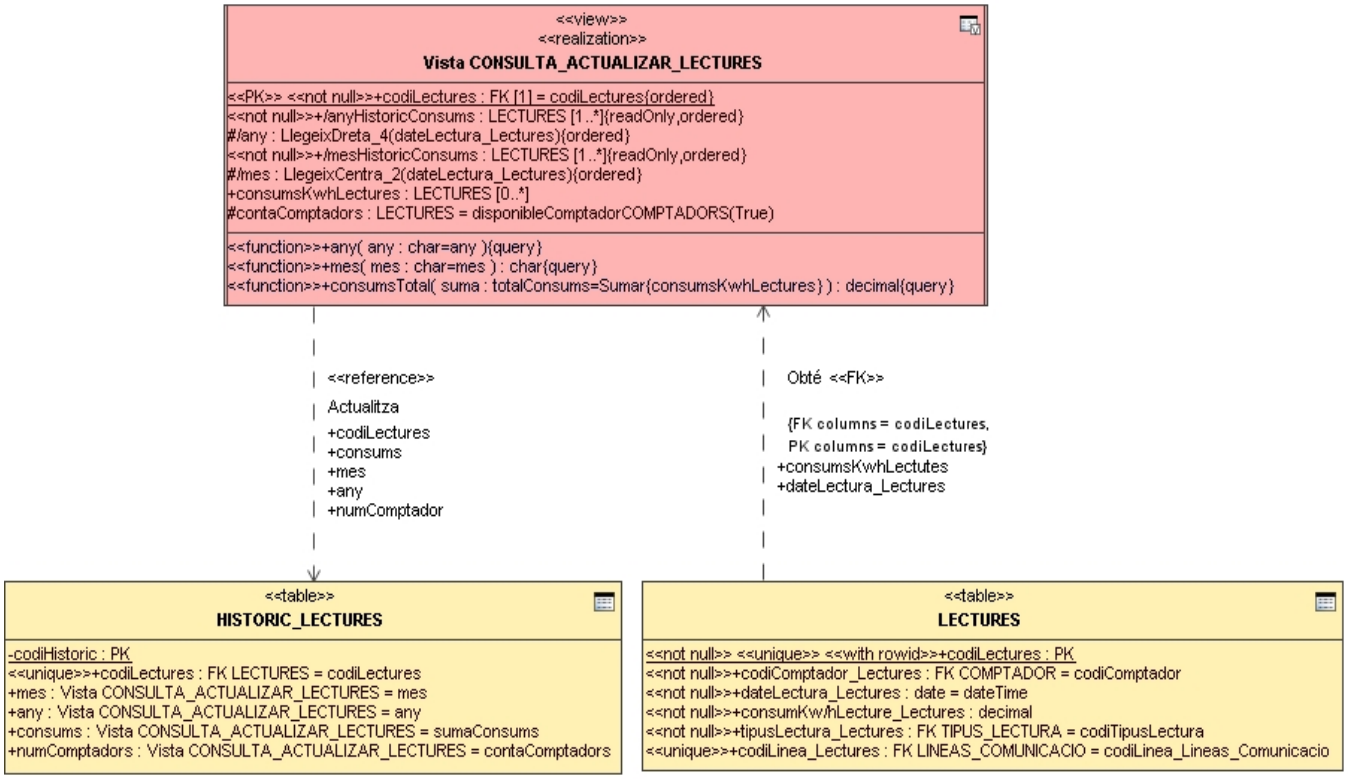


Ilustración 95 Actualizar Lecturas.

2.3 Diseño Lógico.

Transformación del modelo Entidad/Relación a un modelo relacional.

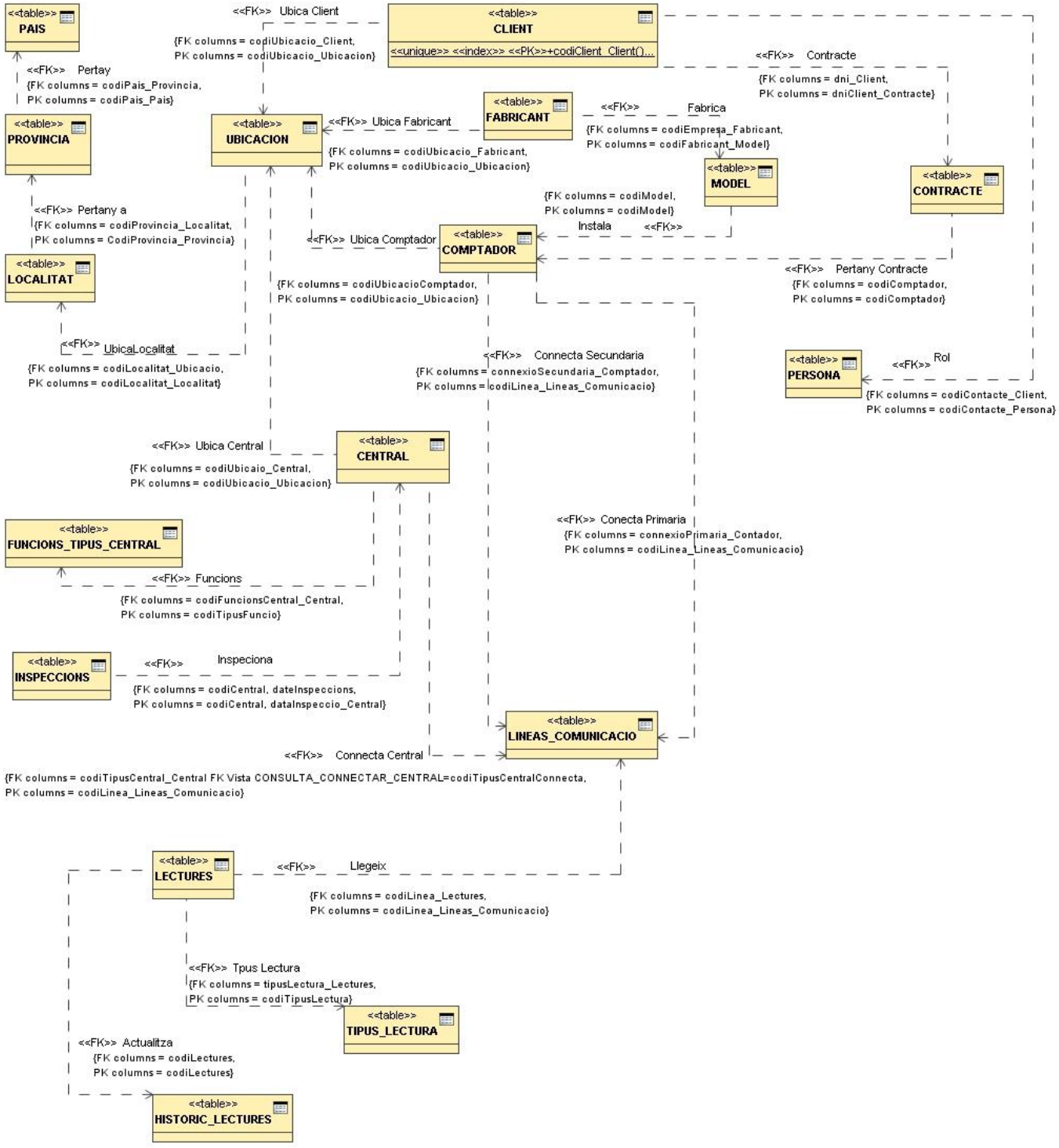


Ilustración 96 Estructura Entidad Relación Base de Datos Relacional.

2.3.1 Localización e identificación del las Claves y su rol.

Del modelo Entidad/Relación se ha identificado el conjunto de claves primarias y foranas, las cuales constituyen el punto esencialmente básico para llevar a término la integridad referencial y la normalización de la base de datos para conseguir durante el proceso de implementación la eficiencia, consistencia y optimización del SGBD.

Las claves primarias identidad, son únicas por lo cual durante la implementación del sistema serán utilizadas en la generación de los índices con el fin de optimizar el acceso productivo de la Base de Datos durante las extracciones que se generen y utilicen.

2.3.2 Refinamiento y depuración del Modelo Entidad/Relación.

Bajo el principio de cascada se procede a una revisión exhaustiva de las posibles optimizaciones posibles siguiendo escrupulosamente la normalización de las Bases de Datos Relacionales. Tal hecho se procede a la fase siguiente, revisando la arquitectura que en un principio salvo mejoras propuestas el diseño generado queda como definitivo.

2.4 Diseño Físico.

El diseño físico es el apartado donde se define la creación de la Base de Datos Relacional, es la creación estructurada del conjunto de tablas con sus atributos, donde se define la intención y la extensión de sus competencias dentro del sistema.

En este punto la detección de claves primarias PK externas y las interrelaciones entre ellas que se hallan sujetas a la integridad referencial, configuran las características que tienen estas entidades.

Para conseguir dicho objetivo, se generan el conjunto de scripts (código) SQL, siempre siguiendo el guión preestablecido por el enunciado huyendo en todo momento de cualquier herramienta automatizada.

Esta fase, también conocida como productiva se realizará los siguientes procesos.

- Métrica optimización tiempo de respuestas.

  Disminución del tiempo de respuesta de la consultas, análisis de la métrica.

- Optimización del espacio físico del SGBD.

  Minimización del espacio físico que ocupa en la unidad de disco físico.

- Políticas de seguridad de uso.

  Crear las políticas de seguridad de los datos críticos que contiene la Base de Datos.

- Optimización de recursos.

  Optimización de los recursos del Sistema Gestor de Base de Datos, a nivel físico dependiendo de las necesidades del Hardware.

- Aplicación del Diseño Lógico.

  Como punto de partida seguiremos el diseño lógico diseñado, aplicando los conocimientos concretos adquiridos sobre la gestión de SGBD.

- Creación de las Tablas.

  Conjunto de código en lenguaje SQL donde aparecerá el archivo CREAR_TABLAS.sql que contiene este código en SQL.

- Creación de los Usuarios.

  En un principio creamos un usuario con los permisos correspondientes y limitados mediante el Tableespace que el código nos ofrece y el sistema comercial usado nos permite.

3 Implementación SQL PL/SQL.

3.1 Descripción de carga:

El fichero que contiene el código hay un fichero LEAME.HTLM el cual esta definido todo el código que hay en el proyecto.

La implementación sea definido por módulos, el motivo es para estructurar en la implementación del producto final que se desarrollara en la fase posterior.

Sea considerado a nivel de usuario que el uso del aplicativo tendrá una estructuración orientada la mismo por lo cual sea definido de la siguiente forma:

3.1.1 MODULO USUARIOS.

3.1.2 MODULO ADMINISTRACION.

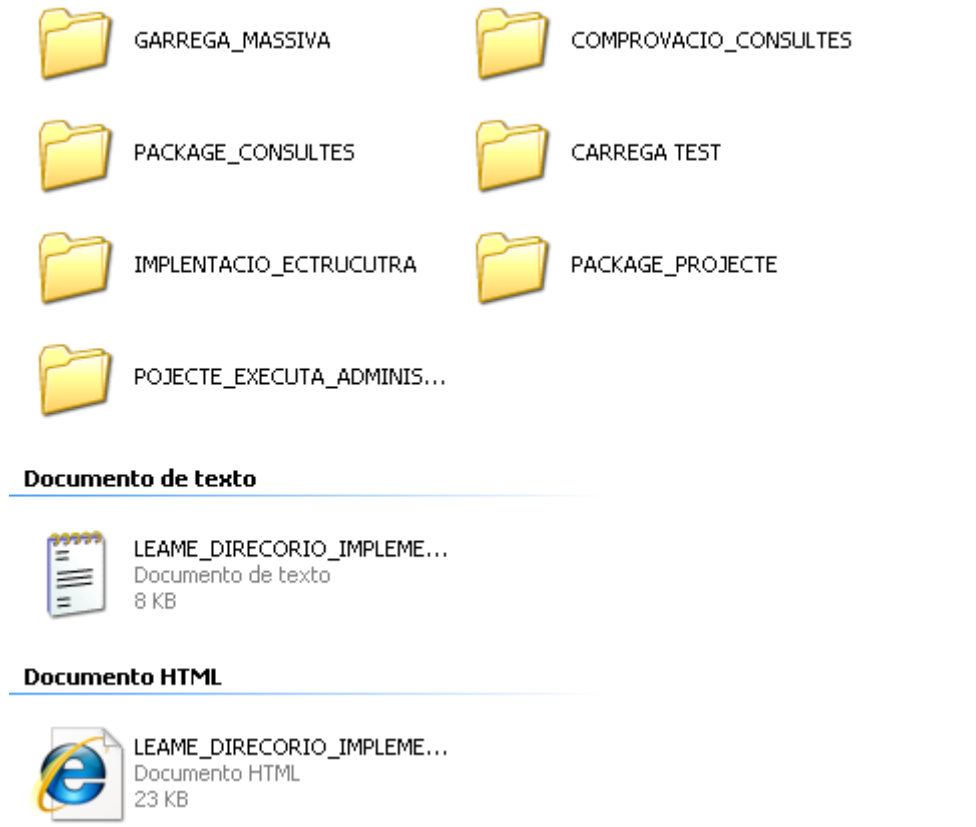
3.1.3 MODULO MANTENIMIENTO.

3.1.4 MODULO PRODUCCIÓN.

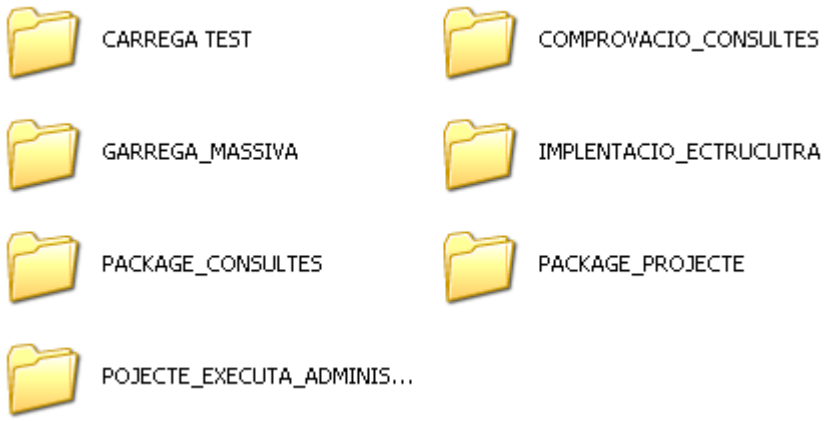
3.1.5 MODULO CONSULTAS.

3.1.6 MODULO ESTADISTICO.

3.1.7 MODULO GENERAL.



El sistema esta construido por partes, tal como se puede visualizar en la figura siguiente.



La primea carpeta IMPLEMATACIO_ESTRUCURA los siguientes ficheros.

1.**1_LOAD_USER.sql**

2. DROP_GENERAL.sql

3. CREAR_TABLAS.sql

4. CREAR_SEQUENCIAS.sql

5. CREAR_TRIGGERS.sql

2. CREAR_FUNCIONES.sql

El orden de creación debe de ser el siguiente.

- Primero debe de crearse los permisos mediante el fichero 1_LOADER USER.sql para elusuario TFC.
- Segundo generar el fichero DROP_GENERAL ejecuta el borrado de todo el proyecto dentro de la Base de Datos.
- Tercero debe de ejecutarse el fichero CREAR_TABLAS.sql que genera las tablas en el sistema.
- Cuarto debe de crearse las secuencias para que las tablas generen automáticamente lo códigos identificativos o claves primarias. CREAR_SEQUENCIAS.sql
- Quinto debe de crearse los disparadores encargados de ejecutar las secuencias cada vez que se inserta un registro.
- Sexto el fichero de funciones que regula conversiones dentro del sistema cuando se utilizan campos numéricos o de fecha y deben de ser manipulados en formato texto o viceversa. CREAR_FUNCIONES.sql

3.2 Creación de la Base de datos.

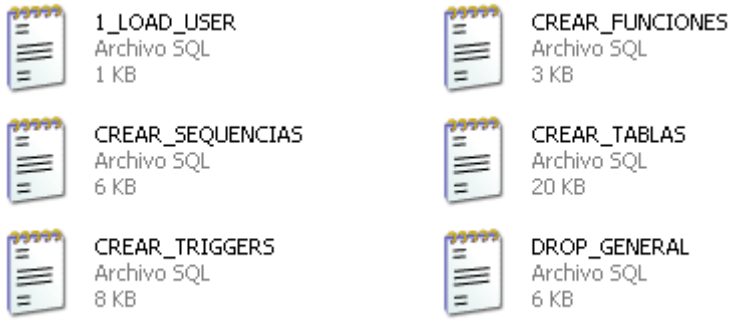
La base de datos esta creada y gestionada por paquetes integrados que ejecutan las extracciones. Son los llamados PACKAGE donde contiene el diseño del código a compilar el cual ejecuta los ficheros GESTION que efectúan las diferentes funcionalidades exigidas.

3.3 Creación de usuarios.

El nivel de usuarios tal como sea comentado ha definido el usuario TFC, sea considerado de que el proyecto es un prototipo por lo cual el usuario SYSTEM pose todos los permisos para gestionar las funcionalidades del proyecto. El fichero 1_LOAD_USER especifica los permisos y la creación del usuario.

3.4 Creación de Tablas.

Las tablas están generadas siguiendo el diseño preestablecido el código del mismo puede visualizarse en el fichero CREA_TABLAS.



3.5 Creación de Índices.

Los índices son generados de forma automática al crear las tablas con las diferentes restricciones que el diseño especifica para conservar la integridad referencia entre las entidades.

Los índices son esenciales para generar las extracciones de forma rápida y eficiente.

3.6 Carga de datos de Test:

En el fichero contiene el conjunto de ficheros para ejecutar la carga de datos mediante las instrucciones generadas en los procedimientos. Como por ejemplo:

```
-- EXECUTA INSERTAR ESTAT
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_ESTAT.PRC_ALTA_ESTAT('ALTA',sortida);
GESTION_ESTAT.PRC_ALTA_ESTAT('BAIXA',sortida);
END;
/
```

Que inserta los valores Alta y Baja en la tabla ESTAT.

En el directorio carga masiva, sean generado dos ficheros para efectuar cargas masivas o migraciones de otros sistemas de Base de Datos.

El ejemplo de carga de Clientes su código del fichero Client.CTL

```
LOAD DATA
CHARACTERSET 'WE8MSWIN1252'
INFILE 'CLIENTS.txt'
INTO TABLE CLIENT
FIELDS TERMINATED BY ';'
OPTIONALLY ENCLOSED BY '"'
TRAILING NULLCOLS
(
idClient,
idtipoClient,
estatClient,
dniClient,
nomClient,
cognom1Client,
cognom2Client,
idUbicaClient,
dataAltaClient,
dataModificacioClient,
observacioClient
)
```

Este código debe de ejecutarse el modo DOS con la instrucción

c:\sqlldr control=client

El fichero una vez ejecutado solicita el usuario y su contraseña.

3.7 Descripción de control.

La tabla LOG.TFC sea creado con la finalidad de recoger el conjunto de procesos en las extracciones y modificaciones que se generan en el sistema.

El procedimiento de altas de Contratos, genera de forma automatizada la inicialización de lecturas de contador en la tabla LECTURA.

El procedimiento de gestión de lecturas, de forma automatizada inserta valores en la tabla de LECTURA y la de HIT_LECTURA con la finalidad de poder hacer las extracciones de consultas de a tiempo real y disponer de un histórico de la información de lecturas que se generan en los contadores. Sea considerado de que un contrato implica un contador inicializado siempre a lecturas a cero.

Todo el sistema funciona a modo de procedures almacenadas en los PACKAGE para cada tratamiento que se efectúa.

La ejecución de cualquier proceso o manipulación de algún procedimiento siempre desencadena la ejecución del procedimiento PK_GENERALS.GRAVAR.LOG_PROCEDRURE

Que le envía los resultados del proceso e inserta los mismo en la taula LOG_TFC

```
create or replace PACKAGE body          PKG_GENERAL AS


PROCEDURE      gravar_log_procedure (
 c_procesLog        LOG_TFC.procesLog%TYPE,
  c_dataHoraLog      LOG_TFC.dataHoraLog%TYPE,
  c_entradaLog       LOG_TFC.entradaLog%TYPE,
  c_sortidaLog       LOG_TFC.sortidaLog%TYPE,
  c_rspLog             LOG_TFC.rspLog%TYPE
)
AS
  PRAGMA AUTONOMOUS_TRANSACTION;

BEGIN
  INSERT INTO LOG_TFC(procesLog, dataHoraLog, entradaLog, sortidaLog, rspLog)
     VALUES (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, c_rspLog);
     COMMIT;     -- Aquest COMMIT nomÃ©s afecta a la transacciÃ³ autÃ²noma

END gravar_log_procedure;
END PKG_GENERAL;
```

4 APENDICE 1 CREACION DE TABLAS.

```
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
  NOM:      CREACIO DE TAULAS DEL SISTEMA

*****************************************************************************/
/
CREATE TABLE ESTAT /* 1 */
(
        idEstat             INT CONSTRAINT PK_idEstat PRIMARY KEY,
        descripcioEstat     VARCHAR2(20 CHAR) CONSTRAINT NN_Estat_descripcio NOT NULL
);
/
--      TAULA VIA
CREATE TABLE VIA /* 2 */
(
        idVia                       INT CONSTRAINT PK_idVia PRIMARY KEY,
        descripcioVia               VARCHAR2(20 CHAR) CONSTRAINT NN_Via_descripcio NOT NULL
);
/
--      TAULA PERSONA
CREATE TABLE PERSONA /* 3 */
(
        idPersona                   INT CONSTRAINT PK_idPersona PRIMARY KEY,
        descripcioPersona           VARCHAR2(20 CHAR) CONSTRAINT NN_Persona_descripcio NOT NULL
);
/
--      TAULA CLASSE CENTRAL
CREATE TABLE CLASSE_CENTRAL  /* 4 */
(
        idClasse            INT CONSTRAINT PK_idClasse PRIMARY KEY,
        descripcioClasse    VARCHAR2(20 CHAR) CONSTRAINT NN_Classe_Central_descripcio NOT NULL
);
/
--      TAULA TIPUS FUNCIONS
CREATE TABLE TIPUS_FUNCIONS  /* 5 */
(
        idFuncio                    INT CONSTRAINT PK_funcio PRIMARY KEY,
        descripcionFuncio           VARCHAR2(50 CHAR) CONSTRAINT NN_Tipus_Funcions_descripcion
NOT NULL
 );
/
--      TIPO LECTURA
CREATE TABLE TIPO_LECTURA /* 6 */
(
        idTipoLectura               INT CONSTRAINT PK_idTipoLectura PRIMARY KEY,
        descripcioLectura           VARCHAR2(50 CHAR) CONSTRAINT NN_Tipo_Lectura_descripcio NOT
NULL
);
/
--      TAULA TIPUS CENTRAL
CREATE TABLE TIPO_CENTRAL  /* 7 */
(
        idTipoCentral               INT CONSTRAINT PK_idTipoCentral PRIMARY KEY,
        descripcioTipoCentral       VARCHAR2(20 CHAR) CONSTRAINT NN_TIPO_CENTRAL_descripcio NOT
NULL,
        idFuncioTipoCentral         CONSTRAINT FK_Tipo_Central_idfuncio         REFERENCES
TIPUS_FUNCIONS(idFuncio) NOT NULL,
        quantitatTipoCentral        DECIMAL(9,2) CONSTRAINT NN_Tipo_Central_quantitat NOT NULL,
        estatTipoCentral    CONSTRAINT FK_tipo_central_estado           REFERENCES ESTAT(idEstat) NOT
NULL,
--      1=Alta OR 2=Baixa
        dataEstatTipoCentral        DATE CONSTRAINT NN_tCentrales_dataEstat  NOT NULL,
        observacionsTipoCentral VARCHAR2(250 CHAR)
);
/
--      TAULA PAIS  /* 8 */
CREATE TABLE PAIS
```

```
(
        idPais          INT CONSTRAINT PK_idPais PRIMARY KEY,
        descripcioPais  VARCHAR2(50 CHAR) CONSTRAINT NN_Pais_nomPais NOT NULL,
        codiPais        VARCHAR2(5 CHAR) CONSTRAINT NN_Pais_codiPais NOT NULL
 );
/
--      TAULA PROVINCIA
CREATE TABLE PROVINCIA  /* 9 */
(
        idProvincia     INT  CONSTRAINT PK_IdProvincia PRIMARY KEY,
        descripcioProvincia     VARCHAR2(50 CHAR) CONSTRAINT NN_provincia_NomProvincia NOT
NULL,
        idPaisProvincia         CONSTRAINT FK_Provincia_idiPais  REFERENCES PAIS (idPais)NOT
NULL
 );
/
--      TAULA LOCALITAT
CREATE TABLE LOCALITAT  /* 10 */
(
        idLocalitat     INT CONSTRAINT PK_idLocalitat PRIMARY KEY,
        descripcioLocalitat     VARCHAR2(50 CHAR) CONSTRAINT NN_Localitat_NomLocalitat NOT
NULL,
        idProvinciaLocalitat    CONSTRAINT FK_Localitat_idiProvincia     REFERENCES PROVINCIA
(idProvincia)NOT NULL
);
/
--      TAULA UBICACIO
CREATE TABLE UBICACIO  /* 11 */
(
        idUbica         INT CONSTRAINT PK_Ubicacio PRIMARY KEY,
        idViaUbicacio   CONSTRAINT FK1_Ubicacio_codiVia         REFERENCES VIA (idVia) NOT
NULL,
        direccionUbicacio       VARCHAR2(50 CHAR) CONSTRAINT NN_Ubicacio_direccio NOT NULL,
        numeroUbicacio          VARCHAR2(5 CHAR) CONSTRAINT NN_Ubicacio_Numero NOT NULL,
        pisUbicacio     VARCHAR2(10 CHAR) CONSTRAINT NN_Ubicacio_Pis  NULL,
        portaUbicacio           VARCHAR2(10 CHAR) CONSTRAINT NN_Ubicacio_Porta NULL,
        codiPostal      VARCHAR2(10 CHAR) CONSTRAINT NN_Ubicacio_CP NOT NULL,
        idLocalitatUbicacio     CONSTRAINT FK2_Ubicacio_idLocalitat     REFERENCES LOCALITAT
(idLocalitat) NOT NULL
);
/
--      TAULA TIPUS INSPECCIO
CREATE TABLE TIPO_INSPECCIO  /* 13 */
(
        idTInspccio             INT CONSTRAINT PK_idTInspeccio PRIMARY KEY,
        descripcioTInspeccio    VARCHAR2(50 CHAR)CONSTRAINT NN_Tipo_Inspeccio_descripcio NOT
NULL
);
/
--      TAULA INSPECCIO
CREATE TABLE INSPECCIO  /* 14 */
(
        idInspeccio     INT CONSTRAINT PK_Inspeccio_idInspeccio PRIMARY KEY,
    dataInspeccio               DATE CONSTRAINT NN_Inspeccio_dataInspecio NOT NULL,
        codiInspector           VARCHAR2(20 CHAR) CONSTRAINT NN_Inspeccio_inspector NOT NULL,
        idsuperaInspeccio       CONSTRAINT FK_Tipo_Inspeccio_idSupera    REFERENCES
TIPO_INSPECCIO(idTInspccio) NOT NULL,
        observacionsInspeccio   VARCHAR2(250 CHAR)
);
/
--      TAULA CENTRAL
CREATE TABLE CENTRAL  /* 12 */
(
        idCentral       INT CONSTRAINT PK_idCent PRIMARY KEY,
        dataAltaCentral DATE CONSTRAINT NN_Central_dataAltaCent  NOT NULL,
        estatCentral            CONSTRAINT FK1_Central_estatCent REFERENCES ESTAT(idEstat)
NOT NULL,
        dataModificaCentral     DATE CONSTRAINT NN_Central_dataModifCent  NULL,
        idclasseCentral         CONSTRAINT FK2_Central_idclasseCent     REFERENCES
CLASSE_CENTRAL(idClasse) NOT NULL,
        idtipoCentral   CONSTRAINT FK3_Central_idtipoCent       REFERENCES
TIPO_CENTRAL(idTipoCentral) NULL,
```

```
        dataUltimaInspeccioCentral  DATE CONSTRAINT NN_Central_dataUltimaInsCent  NOT NULL,
    idInspeccioCentral      CONSTRAINT FK4_Central_idInspeccioCent REFERENCES
TIPO_INSPECCIO(idTInspccio) NOT NULL,
        energiaMaxima           DECIMAL(9,2) CONSTRAINT NN_Central_energiaMax NOT NULL,
        energiaMinima           DECIMAL(9,2) CONSTRAINT NN_Central_energiaMin  NOT NULL,
    idUbicaCentral          CONSTRAINT FK5_Central_idUbicaCent      REFERENCES UBICACIO(idUbica)
NOT NULL,
        observacionsCentral     VARCHAR2(250 CHAR)
);
/
--      TAULA FABRICANT
CREATE TABLE FABRICANT  /* 15 */
(
        idFabricant             INT CONSTRAINT PK_idFabricant PRIMARY KEY,
        nifFabricant            VARCHAR2(15 CHAR) CONSTRAINT NN_Fabricant_nif NOT NULL,
        dataAltaFabricant  DATE CONSTRAINT NN_Fabricant_dataAlta NOT NULL,
        estatFabricant          CONSTRAINT FK1_Fabricant_estat          REFERENCES
ESTAT(idEstat) NOT NULL,
        dataModifcaFabricant    DATE CONSTRAINT NN_Fabricant_fechaModifica  NULL,
        nomComercialFabricant   VARCHAR2(70 CHAR) CONSTRAINT NN_Fabricant_nomComercial NOT
NULL,
        idUbicaFabricant  CONSTRAINT FK2_Fabricant_idUbica      REFERENCES UBICACIO(idUbica)
NOT NULL,
        CONSTRAINT UN_Fabricant_nif UNIQUE (nifFabricant)
);
/
--      TAULA CLIENT
CREATE TABLE CLIENT /* 16 */
(
        idClient                INT CONSTRAINT PK_Client PRIMARY KEY,
        dniClient               VARCHAR2(10 CHAR) CONSTRAINT AK_Client_dni UNIQUE,
        dataAltaClient      DATE DEFAULT(sysdate) CONSTRAINT NN_Client_dataAlta  NOT NULL,
        estatClient             CONSTRAINT FK1_Client_estat REFERENCES ESTAT(idEstat) NOT NULL,
        dataModificacioClient   DATE,
        nomClient               VARCHAR2(50 CHAR) CONSTRAINT NN_Client_nom NOT NULL,
        cognom1Client           VARCHAR2(50 CHAR) CONSTRAINT NN_Client_cognom1 NOT NULL,
        cognom2Client           VARCHAR2(50 CHAR),
        idUbicaClient       CONSTRAINT FK2_Client_idUbica REFERENCES UBICACIO(idUbica) NOT NULL,
        idtipoClient            CONSTRAINT FK3_Client_idtipoClient REFERENCES
PERSONA(idPersona) NOT NULL,
        observacioClient   VARCHAR2(250 CHAR)
);
/
--      TAULA MODEL
CREATE TABLE MODELO  /* 17 */
(
        idModelo            INT  CONSTRAINT PK_idModelo PRIMARY KEY,
        descripcioModelo        VARCHAR2(50 CHAR),
        anyoFabricaModelo       VARCHAR2(4 CHAR),
        idFabricantModelo       CONSTRAINT FK1_Modelo_idiFabricant      REFERENCES
FABRICANT(idFabricant) NOT NULL,
        dataAltaModelo          DATE CONSTRAINT NN_Modelo_dataAlta NOT NULL,
        estatModelo             CONSTRAINT FK2_Modelo_estat             REFERENCES
ESTAT(idEstat) NOT NULL,
        dataModificacioModelo   DATE CONSTRAINT NN_Modelo_fechaModifica  NULL,
        observacionsModelo      VARCHAR2(50 CHAR)
);
/
--      TAULA COMPTADOR
CREATE TABLE COMPTADOR  /* 18 */
(
        idComptador             INT  CONSTRAINT PK_idComptador PRIMARY KEY,
        idModeloComptador       CONSTRAINT FK1_Comptador_idModel        REFERENCES
MODELO(idModelo) NOT NULL,
        idUbicaComptador    CONSTRAINT FK2_Comptador_idUbica    REFERENCES UBICACIO(idUbica)
NOT NULL,
        dataAltaComptador       DATE CONSTRAINT NN_Comptador_fechaAlta  NOT NULL,
        idEstadoCompator        CONSTRAINT FK3_Comptador_idEstat        REFERENCES
ESTAT(idEstat) NOT NULL,
        dataModificacioCompator DATE CONSTRAINT NN_Comptador_fechaEstado   NULL,
        numSerieComptador       VARCHAR2(20 CHAR) CONSTRAINT NN_Comptador_numSerie NOT
NULL,
```

```
      observacionsComptador   VARCHAR2(70 CHAR)
);
/
--          TAULA CONTRACTE
CREATE TABLE CONTRACTE  /* 19 */
(
          idContracte          INT CONSTRAINT PK_idContracte PRIMARY KEY,
          dniClienteContracte            CONSTRAINT FK1_Contracte_dniCliente   REFERENCES
CLIENT(dnicLIENT) NOT NULL,
          idContadorContracte          CONSTRAINT FK2_Contracte_idContador   REFERENCES
COMPTADOR(idComptador) NOT NULL,
          dataAltaContracte  DATE CONSTRAINT NN_Contracte_fechaAlta  NOT NULL,
          idEstatContracte    CONSTRAINT FK3_Contracte_idEstat      REFERENCES ESTAT(idEstat) NOT NULL,
          dataModificacioContracte DATE CONSTRAINT NN_Contracte_fechaModifica  NULL,
          potenciaContracte              DECIMAL(9,2)   CONSTRAINT NN_Contracte_potenciaContrato NOT
NULL,
          observacioContracte          VARCHAR2(100 CHAR)
);
/
--  TAULA TIPUS DE LINEA
CREATE TABLE LINEA_TIPUS /* 20 */
(
 idTlinea        INT CONSTRAINT PK_Tlinea  PRIMARY KEY,
 idEstatTlinea     CONSTRAINT FK_Tlinea_idEstado  REFERENCES ESTAT(idEstat) NOT NULL,
 dataEstatTLinea  DATE CONSTRAINT NN_linea_fechaEstat NOT NULL,
 consumMaximTLinea DECIMAL(9,2) CONSTRAINT NN_Tlinea_consumMaxim NOT NULL,
 observacioTlinea  VARCHAR(50 CHAR)
);
/
--          TAULA LINEA CONNECTAR  CENTRALS
CREATE TABLE LINEA_CONECTA_CENTRAL /* 30 */
(
          idConecta       INT CONSTRAINT PK_idConecta   PRIMARY KEY,
          dataAltaC       DATE  DEFAULT(sysdate)CONSTRAINT NN_CONECTA_fechaAlta  NOT NULL,

          idEstatC         CONSTRAINT FK1_CONECTA_idEstat REFERENCES ESTAT(idEstat) NOT NULL,
          dataModificacioC              DATE CONSTRAINT NN_CONECTA_fechaModifica  NULL,
          connectaCProdu       CONSTRAINT FK2_CONECTA_connectaCentProdu  REFERENCES
CENTRAL(idCentral) NOT NULL,
          connectaCDistri    CONSTRAINT FK3_CONECTA_connectaCentDistri  REFERENCES
CENTRAL(idCentral) NOT NULL,
          observacioC        VARCHAR(50 CHAR)
);
/
--          TAULA LINEA CONNECTAR COMPTADOR AMB CENTRAL DISTRIBUCIO
CREATE TABLE LINEA_CONECTAR_COMPTADOR /* 21 */
(
          idLineaConecta                INT CONSTRAINT PK_idLinea  PRIMARY KEY,
          idLineaTipoLineaConecta    CONSTRAINT FK2_lineaConecta_idLineaTLinea REFERENCES
LINEA_TIPUS(idTlinea) NOT NULL,
      dataAltaLineaConecta    DATE  DEFAULT(sysdate)CONSTRAINT NN_Linea_fechaAlta  NOT NULL,

          idEstatLineaConecta     CONSTRAINT FK1_Linea_idEstat REFERENCES ESTAT(idEstat) NOT NULL,
          dataModificacioLineaConecta           DATE CONSTRAINT NN_Linea_fechaModifica  NULL,
          connecta1_CentralConecta       CONSTRAINT FK2_Linea_connectaCentral              REFERENCES
LINEA_CONECTA_CENTRAL(idConecta) NOT NULL,
          connecta2_ComptadorConecta          CONSTRAINT FK3_Linea_connectaComptador  REFERENCES
COMPTADOR(idComptador) NOT NULL,
          parLineaComptadorConecta       NUMBER NOT NULL,
          observacioLineaConecta         VARCHAR(50 CHAR)
);
/
--          TAULA LECTURA
CREATE TABLE LECTURA /* 22 */
(
          idLectura             INT CONSTRAINT PK_idLectura  PRIMARY KEY,
          --idLineaLectura    CONSTRAINT FK1_Lectura_idLinea REFERENCES
LINEA_CONECTAR_COMPTADOR(idLineaConecta) NOT NULL,
          idComptadorLectura      CONSTRAINT FK1_Lectures_idcomptador REFERENCES
COMPTADOR(idComptador) NOT NULL,
      confirmaLectura          INT CONSTRAINT FK2_Lectura_Comfirma CHECK(confirmaLectura='0' OR
confirmaLectura='1')NOT NULL,
```

```
-- estat opcional per dona per validar 0=No validada OR 1=Validada
            dataLectura                 DATE CONSTRAINT NN_Lectura_fechaLectura NOT NULL,
            idTipoLectura       CONSTRAINT FK3_Lectura_tipusLectura REFERENCES
TIPO_LECTURA(idTipoLectura) NOT NULL,                       -- tipo lectura 0 telematica, 1 presencial
            consumLectura               DECIMAL(9,2) CONSTRAINT NN_Lectura_Consumo  NULL,
      lecturaActual         DECIMAL(9,2) CONSTRAINT NN_Lectura_Consumo  NULL,
      lecturaAnterior       DECIMAL(9,2) CONSTRAINT NN_Lectura_Consumo  NULL
);
/
--          TAULA HIST_LECTURA
CREATE TABLE HIST_LECTURA  /* 23 */
(
            regHistLectura              INT CONSTRAINT  PK_numReg PRIMARY KEY,
      idLineaHistLectura      CONSTRAINT FK2_Hist_Lectura_idLinea REFERENCES
LINEA_CONECTAR_COMPTADOR(idLineaConecta) NOT NULL,
            idComptadorHistLectura      CONSTRAINT FK1_Hist_Lectura_idContador  REFERENCES
COMPTADOR(idComptador) NOT NULL,
            anyoHistLectura         VARCHAR2(4 CHAR) CONSTRAINT NN_Hist_Lectura_anyo NOT NULL,
      mesHistLectura          VARCHAR2(2 CHAR) CONSTRAINT NN_Hist_Lectura_mes  NOT NULL,
            totalConsumoHistLectura DECIMAL(9,2)
);
-- TAULA LOG_TFC
CREATE TABLE LOG_TFC
(
 idLog        NUMBER CONSTRAINT PK_log PRIMARY KEY,
 procesLog    VARCHAR2(1000 BYTE)   CONSTRAINT NN_procesoLog_Log NOT NULL,
 dataHoraLog   DATE  CONSTRAINT NN_fechaHoraLog_Log NOT NULL,
 entradaLog    VARCHAR2(1000 BYTE)  CONSTRAINT NN_entradaLog_Log NOT NULL,
 sortidaLog    VARCHAR2(1000 BYTE)  CONSTRAINT NN_salidaLog_Log NOT NULL,
 rspLog       VARCHAR2(1000 BYTE)  CONSTRAINT NN_RSPLog_log NOT NULL
);
--          MODUL ESTADISTIC
/
--          TAULA E 1
CREATE TABLE E_1 /* 24 */
(
 idE_1      INT CONSTRAINT PK_idE_1 PRIMARY KEY,
 idCentral   CONSTRAINT FK1_E_1_idCentral REFERENCES CENTRAL(idCentral) NOT NULL,
 idComptador_E_1 CONSTRAINT FK2_E_1 REFERENCES COMPTADOR(idComptador) NOT NULL,
 sumaConsums DECIMAL(9,2)
);

/
--          TAULA E 2 /* 23 */
-- Donada una línia de comunicació i un any concret, el valor mitjà de l'energia consumida,
-- tenint en compte que aquest
-- consum depèn dels comptadors que s'alimenten mitjançant  aquesta línia.
-- Si un comptador pot fer servir dues línies, suposeu que les dues línies computen el consum per
-- garantir que les línies estan ben dimensionades en cas de caiguda d'alguna d'elles.
/
CREATE TABLE E_2 /*25*/
(
 idE_2 INT CONSTRAINT PK_idE_2 PRIMARY KEY,
 idLineaConectaE_2 CONSTRAINT FK_E_2_idLineaConectaE_2 REFERENCES
LINEA_CONECTAR_COMPTADOR(idLineaConecta) NOT NULL,
 anyoE_2    VARCHAR2(4 CHAR) CONSTRAINT NN_e2_anyo NOT NULL,
 consumoContadoresE_2    NUMBER(9,2) CONSTRAINT NN_e2_consumoContadores  NOT NULL,
 consumoEnergiaMedioE_2  NUMBER(9,2) CONSTRAINT NN_e2_consumoEnergiaMedio  NOT NULL
);
/
--  TAULA E_3
-- E3_Línia que ha estat més carregada a nivell d'energia consumida.
CREATE TABLE E_3 /*26*/
(
id_E3     NUMBER   CONSTRAINT      PK_e_3_ID_E3 PRIMARY KEY,
 linea_E3     CONSTRAINT      FK_e_3_linea_E3  REFERENCES
LINEA_CONECTAR_COMPTADOR(idLineaConecta) NOT NULL,
 consumoContadores_E3    NUMBER(9,2) CONSTRAINT      NN_e_3_consumoContadores_E3  NULL,
 any_E3             VARCHAR2(4 CHAR) CONSTRAINT     NN_e_3_any_E3  NOT NULL
);

/
```

```
-- TAULA E_4
-- E4 Donat un any concret: percentatge de línies que superen el 50% d'energia consumida.
CREATE TABLE E_4 /*27*/
(
idE_4 NUMBER  CONSTRAINT PK_idE4   PRIMARY KEY,
lineaE_4  CONSTRAINT      FK_e_4_linea_E4  REFERENCES
LINEA_CONECTAR_COMPTADOR(idLineaConecta) NOT NULL,
anyoE_4   VARCHAR2(4 CHAR) CONSTRAINT  NN_E4_anyo  NOT NULL,
percentageE_4 NUMBER(9,2) CONSTRAINT NN_E4_percentageE_4  NOT NULL
);
/
-- TAULA E_5
-- E5 Donat un any concret: el nombre de centrals de producció que generen menys del 30% de producció.
CREATE TABLE E_5 /*28*/
(
idE_5 NUMBER  CONSTRAINT PK_idE5   PRIMARY KEY,
lineaE_5  CONSTRAINT      FK_e_5_linea_E5  REFERENCES
LINEA_CONECTAR_COMPTADOR(idLineaConecta) NOT NULL,
anyoE_5   VARCHAR2(4 CHAR) CONSTRAINT  NN_E5_anyo  NOT NULL,
centProducMenos30 NUMBER(9,2) CONSTRAINT NN_E5_centProduMenos30 NOT NULL
);
/
-- TAULA E_6
-- E6 Top-10 de comptadors que històricament han tingut més consum.
CREATE TABLE E_6 /*29*/
(
idE_6 NUMBER  CONSTRAINT PK_idE6   PRIMARY KEY,
idcontadorE_6 CONSTRAINT FK_E6_idContadorE_6 REFERENCES  COMPTADOR(idComptador)  NOT NULL,
consumoE_6     NUMBER(9,2) CONSTRAINT NN_E6_consumoE_6 NOT NULL
);
/
-- TAULA E_7
-- E7 Consum mig de tots dels clients.
CREATE TABLE E_7
(
 idE_7 NUMBER CONSTRAINT PK_E7 PRIMARY KEY,
 dniClient  CONSTRAINT FK_E7_clientDni  REFERENCES  CLIENT(dniClient)  NOT NULL,
 consumoMedioCli  NUMBER(9,2) CONSTRAINT  NN_e7_consumoMedio NOT NULL
);
/

/************** FINAL PROCES CREACIO TAULES **************************/
```

5 APENDICE 2 CREACION DE SECUENCIAS

```
/*****************************************************************************
Autor: Eduard Monzonis Hierro                          UOC
18/05/2012                                  TFC: CONTROL ENERGIA.
  NOM:       CREACIO DE SEQUENCIES

*****************************************************************************/
/
CREATE SEQUENCE SEQ_CENTRAL
            START WITH 1
            INCREMENT BY 1
            NOMAXVALUE
            NOCACHE
;
/
-- Secuencia CLASSE CENTRAL
CREATE SEQUENCE SEQ_CLASSE_CENTRAL
            START WITH 1
            INCREMENT BY 1
            NOMAXVALUE
            NOCACHE
;
/
-- Secuencias CLIENT
CREATE SEQUENCE SEQ_CLIENT
```

```
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencias COMPTADOR
CREATE SEQUENCE SEQ_COMPTADOR
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia CONTRACTE
CREATE SEQUENCE SEQ_CONTRACTE
		START WITH 1
		INCREMENT BY 1
		NOMAXVALUE
		NOCACHE
;
/
-- Secuencia CLASSE ESTAT
CREATE SEQUENCE SEQ_ESTAT
		START WITH 1
		INCREMENT BY 1
		NOMAXVALUE
		NOCACHE
;
/
-- Secuencia FABRICANT
CREATE SEQUENCE SEQ_FABRICANT
		START WITH 1
		INCREMENT BY 1
		NOMAXVALUE
		NOCACHE
;
/
-- Secuencia INSPECCIO
CREATE SEQUENCE SEQ_INSPECCIO
		START WITH 1
		INCREMENT BY 1
		NOMAXVALUE
		NOCACHE
;
/
-- Secuencia LECTURA
CREATE SEQUENCE SEQ_LECTURA
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia LINEA_CONECTAR
CREATE SEQUENCE SEQ_LINEA_CONECTAR
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;

-- Secuencia LINEA_TIPUS
CREATE SEQUENCE SEQ_LINEA_TIPUS
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia LOG_TFC
CREATE SEQUENCE SEQ_LOG_TFC
```

```
    START WITH 1
    INCREMENT BY 1
    NOMAXVALUE
    NOCACHE
;
/
-- Secuencia MODELO
CREATE SEQUENCE SEQ_MODELO
    START WITH 1
    INCREMENT BY 1
    NOMAXVALUE
    NOCACHE
;
/
-- Secuencia PAIS
CREATE SEQUENCE SEQ_PAIS
    START WITH 1
    INCREMENT BY 1
    NOMAXVALUE
    NOCACHE
;
/
-- Secuencia PROVINCIA
CREATE SEQUENCE SEQ_PROVINCIA
    START WITH 1
    INCREMENT BY 1
    NOMAXVALUE
    NOCACHE
;
/
-- Secuencia LOCALITAT
CREATE SEQUENCE SEQ_LOCALITAT
    START WITH 1
    INCREMENT BY 1
    NOMAXVALUE
    NOCACHE
;
/
-- Secuencia TIPO_CENTRAL
CREATE SEQUENCE SEQ_TIPO_CENTRAL
    START WITH 1
    INCREMENT BY 1
    NOMAXVALUE
    NOCACHE
;
/
-- Secuencia TIPO_INSPECCIO
CREATE SEQUENCE SEQ_TIPO_INSPECCIO
    START WITH 1
    INCREMENT BY 1
    NOMAXVALUE
    NOCACHE
;
/
-- Secuencia TIPO_LECTURA
CREATE SEQUENCE SEQ_TIPO_LECTURA
    START WITH 1
    INCREMENT BY 1
    NOMAXVALUE
    NOCACHE
;
/
-- Secuencia HIST_LECTURA
CREATE SEQUENCE SEQ_HIST_LECTURA
    START WITH 1
    INCREMENT BY 1
    NOMAXVALUE
    NOCACHE
;
/
-- Secuencia TIPO_FUNCIONES
CREATE SEQUENCE SEQ_TIPUS_FUNCIONS
```

```
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia UBICACIO
CREATE SEQUENCE SEQ_UBICACIO
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia VIA
CREATE SEQUENCE SEQ_VIA
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia PERSONA
CREATE SEQUENCE SEQ_PERSONA
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- ESTADISTIQUES
-- Secuencia E_1
CREATE SEQUENCE SEQ_E_1
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia E_2
CREATE SEQUENCE SEQ_E_2
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia E_3
CREATE SEQUENCE SEQ_E_3
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia E_4
CREATE SEQUENCE SEQ_E_4
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia E_5
CREATE SEQUENCE SEQ_E_5
	START WITH 1
	INCREMENT BY 1
	NOMAXVALUE
	NOCACHE
;
/
-- Secuencia E_6
```

```
CREATE SEQUENCE SEQ_E_6
      START WITH 1
      INCREMENT BY 1
      NOMAXVALUE
      NOCACHE
;
/
-- Secuencia E_7
CREATE SEQUENCE SEQ_E_7
      START WITH 1
      INCREMENT BY 1
      NOMAXVALUE
      NOCACHE
;
/
-- Secuencia conectar linea a centrals
CREATE SEQUENCE SEQ_LINEA_CONECTA_CENTRAL
      START WITH 1
      INCREMENT BY 1
      NOMAXVALUE
      NOCACHE
;
/********************FI PROCES CREACIO SEQUENCIES ****************************/
```

6 APENDICE 3 CREACION DE FUNCIONES.

```
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
  NOM:       CREACIO DE FUNCIONS

*****************************************************************************/

create or replace FUNCTION  es_data
/*****************************************************************************
  NOM:       Es_Data
  DESCRIPCIÓ:
          Comprova si el parámetro c_data te el format de data especificat
          pel paràmetre c_format. Per defecte el format de la data será DD/MM/AAAA

  RETORNA:    Un booleà que serà vertader si compleix el format i fals si no compleix.

  EXEMPLE:    Es_Data('22/04/2008 08:00', 'DD/MM/YYYY hh24:mi:ss')
*****************************************************************************/
(
  c_data    VARCHAR2,
  c_format  VARCHAR2 := 'DD/MM/YYYY'
)
    RETURN BOOLEAN
  AS
    d_data_text  DATE;
  BEGIN
    d_data_text := TO_DATE (c_data, c_format);
    RETURN TRUE;
  EXCEPTION
```

```
    WHEN OTHERS
    THEN
      RETURN FALSE;
END es_data;

/
create or replace
FUNCTION  quinTrimestre(n_mes number) RETURN number
/*****************************************************************************
  NOM:       quinTrimestre

  RETORNA:    Un numero de mes de la data.

  EXEMPLE:    quinTrimestre(11) retorna 4
*****************************************************************************/
  AS
    trimestre NUMBER;
  BEGIN
     IF ( n_mes > 0 AND n_mes < 4 ) THEN
       trimestre := 1;
     END IF;
     IF ( n_mes > 3 AND n_mes < 7 ) THEN
       trimestre := 1;
     END IF;
     IF ( n_mes > 6 AND n_mes < 10 ) THEN
       trimestre := 1;
     END IF;
     IF ( n_mes > 10 AND n_mes < 13 ) THEN
       trimestre := 1;
     END IF;
     RETURN trimestre;

  EXCEPTION
    WHEN OTHERS
    THEN
      RETURN 0;
END quinTrimestre;
/

create or replace FUNCTION es_numero(s_param VARCHAR2) RETURN BOOLEAN IS
/*****************************************************************************
  NOM:       Es_Numero
  DESCRIPCIÓ:
          Comprova si el paràmetre s_param es numèric o no.
          Es comprova amb la funció TO_NUMBER, a la que si es passa un
          paràmetre no numèric provoca una excepció que ens indica que
          no és numèrico, en cas contrari sabrem que sí ho és.

  RETORNA:    Un booleà que serà vertader si el paràmetre s_param és un número
          o fals si no ho és.

  EXEMPLE:    Es_Numero(25)-> Retorna True, Es_Numero('TENIS')-> Retorna False
```

```
**************************************************************************/

 n NUMBER;
BEGIN
  n := TO_NUMBER(s_param);
  RETURN TRUE;
EXCEPTION
  WHEN OTHERS THEN
    RETURN FALSE;
END es_numero;
/
```

7 APENDICE 4 TRIGGERS.

```
/**************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
-- CREACIO DE DISPARADORS PER GESTIONAR ELS INSERTS
-- creacio de TRIGGERS
**************************************************************************/
-- VIA
CREATE OR REPLACE TRIGGER TR_VIA_KEY BEFORE INSERT ON VIA
FOR EACH ROW
BEGIN
  SELECT SEQ_VIA.NEXTVAL INTO :NEW.idVia FROM DUAL;
END;
/
--  LOG_TFC
CREATE OR REPLACE TRIGGER TR_LOG_TFC_KEY BEFORE INSERT ON LOG_TFC
FOR EACH ROW
BEGIN
  SELECT SEQ_LOG_TFC.NEXTVAL INTO :NEW.idLog FROM DUAL;
END;
/
--  CENTRAL
CREATE OR REPLACE TRIGGER TR_CENTRAL_KEY BEFORE INSERT ON CENTRAL
FOR EACH ROW
BEGIN
  SELECT SEQ_CENTRAL.NEXTVAL INTO :NEW.idCentral FROM DUAL;
END;
/
--  CLASSE CENTRAL
CREATE OR REPLACE TRIGGER TR_CALSSE_CENTRAL_KEY BEFORE INSERT ON CLASSE_CENTRAL
FOR EACH ROW
BEGIN
  SELECT SEQ_CLASSE_CENTRAL.NEXTVAL INTO :NEW.idClasse FROM DUAL;
END;
/
--  CLIENT
CREATE OR REPLACE TRIGGER TR_CLIENT_KEY BEFORE INSERT ON CLIENT
FOR EACH ROW
BEGIN
  SELECT SEQ_CLIENT.NEXTVAL INTO :NEW.idClient FROM DUAL;
END;
/
--  COMPTADOR
CREATE OR REPLACE TRIGGER TR_COMPTADOR_KEY BEFORE INSERT ON COMPTADOR
FOR EACH ROW
BEGIN
  SELECT SEQ_COMPTADOR.NEXTVAL INTO :NEW.idComptador FROM DUAL;
END;
/
--  CONTRACTE
CREATE OR REPLACE TRIGGER TR_CONTRACTE_KEY BEFORE INSERT ON CONTRACTE
FOR EACH ROW
BEGIN
```

```
  SELECT SEQ_CONTRACTE.NEXTVAL INTO :NEW.idContracte FROM DUAL;
END;
/
-- ESTAT
CREATE OR REPLACE TRIGGER TR_ESTAT_KEY BEFORE INSERT ON ESTAT
FOR EACH ROW
BEGIN
  SELECT SEQ_ESTAT.NEXTVAL INTO :NEW.idEstat FROM DUAL;
END;
/
-- FABRICANT
CREATE OR REPLACE TRIGGER TR_FABRICANT_KEY BEFORE INSERT ON FABRICANT
FOR EACH ROW
BEGIN
  SELECT SEQ_FABRICANT.NEXTVAL INTO :NEW.idFabricant FROM DUAL;
END;
/
-- HIST_LECTURA
CREATE OR REPLACE TRIGGER TR_HIST_LECTURA_KEY BEFORE INSERT ON HIST_LECTURA
FOR EACH ROW
BEGIN
  SELECT SEQ_HIST_LECTURA.NEXTVAL INTO :NEW.regHistLectura FROM DUAL;
END;
/
-- INSPECCIO
CREATE OR REPLACE TRIGGER TR_INSPECCIO_KEY BEFORE INSERT ON INSPECCIO
FOR EACH ROW
BEGIN
  SELECT SEQ_INSPECCIO.NEXTVAL INTO :NEW.idInspeccio FROM DUAL;
END;
/
-- LECTURA
CREATE OR REPLACE TRIGGER TR_LECTURA_KEY BEFORE INSERT ON LECTURA
FOR EACH ROW
BEGIN
  SELECT SEQ_LECTURA.NEXTVAL INTO :NEW.idLectura FROM DUAL;
END;
/
-- LINEA_TIPUS
CREATE OR REPLACE TRIGGER TR_LINEA_TIPUS_KEY BEFORE INSERT ON LINEA_TIPUS
FOR EACH ROW
BEGIN
  SELECT SEQ_LINEA_TIPUS.NEXTVAL INTO :NEW.idTLinea FROM DUAL;
END;
/
-- MODELO
CREATE OR REPLACE TRIGGER TR_MODELO_KEY BEFORE INSERT ON MODELO
FOR EACH ROW
BEGIN
  SELECT SEQ_MODELO.NEXTVAL INTO :NEW.idModelo FROM DUAL;
END;
/
-- PERSONA
CREATE OR REPLACE TRIGGER TR_PERSONA_KEY BEFORE INSERT ON PERSONA
FOR EACH ROW
BEGIN
  SELECT SEQ_PERSONA.NEXTVAL INTO :NEW.idPersona FROM DUAL;
END;
/
-- PAIS
CREATE OR REPLACE TRIGGER TR_PAIS_KEY BEFORE INSERT ON PAIS
FOR EACH ROW
BEGIN
  SELECT SEQ_PAIS.NEXTVAL INTO :NEW.idPais FROM DUAL;
END;
/
-- PROVINCIA
CREATE OR REPLACE TRIGGER TR_PROVINCIA_KEY BEFORE INSERT ON PROVINCIA
FOR EACH ROW
BEGIN
  SELECT SEQ_PROVINCIA.NEXTVAL INTO :NEW.idProvincia FROM DUAL;
END;
```

```
/
-- LOCALITAT
CREATE OR REPLACE TRIGGER TR_LOCALITAT_KEY BEFORE INSERT ON LOCALITAT
FOR EACH ROW
BEGIN
  SELECT SEQ_LOCALITAT.NEXTVAL INTO :NEW.idLocalitat FROM DUAL;
END;
/
-- TIPO_CENTRAL
CREATE OR REPLACE TRIGGER TR_TIPO_CENTRAL_KEY BEFORE INSERT ON TIPO_CENTRAL
FOR EACH ROW
BEGIN
  SELECT SEQ_TIPO_CENTRAL.NEXTVAL INTO :NEW.idTipoCentral FROM DUAL;
END;
/
-- TIPO_INSPECCIO
CREATE OR REPLACE TRIGGER TR_TIPO_INSPECCIO_KEY BEFORE INSERT ON TIPO_INSPECCIO
FOR EACH ROW
BEGIN
  SELECT SEQ_TIPO_INSPECCIO.NEXTVAL INTO :NEW.idTInspccio FROM DUAL;
END;
/
-- TIPO_LECTURA
CREATE OR REPLACE TRIGGER TR_TIPO_LECTURA_KEY BEFORE INSERT ON TIPO_LECTURA
FOR EACH ROW
BEGIN
  SELECT SEQ_TIPO_LECTURA.NEXTVAL INTO :NEW.idTipoLectura FROM DUAL;
END;
/
-- UBICACIO
CREATE OR REPLACE TRIGGER TR_UBICACIO_KEY BEFORE INSERT ON UBICACIO
FOR EACH ROW
BEGIN
  SELECT SEQ_UBICACIO.NEXTVAL INTO :NEW.idUbica FROM DUAL;
END;
/
-- LINEA CONECTAR COMPTADORS AMB CENTRALS
CREATE OR REPLACE TRIGGER TR_LINEA_CONECTAR BEFORE INSERT ON
LINEA_CONECTAR_COMPTADOR
FOR EACH ROW
BEGIN
  SELECT SEQ_LINEA_CONECTAR.NEXTVAL INTO :NEW.idLineaConecta FROM DUAL;
END;
/
-- LINEA TIPUS FUNCIONS
CREATE OR REPLACE TRIGGER TR_TIPUS_FUNCIONS BEFORE INSERT ON TIPUS_FUNCIONS
FOR EACH ROW
BEGIN
  SELECT SEQ_TIPUS_FUNCIONS.NEXTVAL INTO :NEW.IdFuncio FROM DUAL;
END;
/
-- LINEA CONECTAR CENTRALS
CREATE OR REPLACE TRIGGER TR_SEQ_LINEA_CONECTA_CENTRAL BEFORE INSERT ON
LINEA_CONECTA_CENTRAL
FOR EACH ROW
BEGIN
  SELECT SEQ_LINEA_CONECTA_CENTRAL.NEXTVAL INTO :NEW.idConecta FROM DUAL;
END;
/*****************************************************************************
ESTADISTIQUES
*****************************************************************************/
-- E_1
CREATE OR REPLACE TRIGGER TR_E_1_KEY BEFORE INSERT ON E_1
FOR EACH ROW
BEGIN
  SELECT SEQ_E_1.NEXTVAL INTO :NEW.idE_1 FROM DUAL;
END;
/
-- E_2
CREATE OR REPLACE TRIGGER TR_E_2_KEY BEFORE INSERT ON E_2
FOR EACH ROW
BEGIN
```

```
  SELECT SEQ_E_2.NEXTVAL INTO :NEW.idE_2 FROM DUAL;
END;
/
--  E_3
CREATE OR REPLACE TRIGGER TR_E_3_KEY BEFORE INSERT ON E_3
FOR EACH ROW
BEGIN
  SELECT SEQ_E_3.NEXTVAL INTO :NEW.id_E3  FROM DUAL;
END;
/
--  E_4
CREATE OR REPLACE TRIGGER TR_E_4_KEY BEFORE INSERT ON E_4
FOR EACH ROW
BEGIN
  SELECT SEQ_E_4.NEXTVAL INTO :NEW.idE_4 FROM DUAL;
END;
/
--  E_5
CREATE OR REPLACE TRIGGER TR_E_5_KEY BEFORE INSERT ON E_5
FOR EACH ROW
BEGIN
  SELECT SEQ_E_5.NEXTVAL INTO :NEW.idE_5 FROM DUAL;
END;
/
--  E_6
CREATE OR REPLACE TRIGGER TR_E_6_KEY BEFORE INSERT ON E_6
FOR EACH ROW
BEGIN
  SELECT SEQ_E_6.NEXTVAL INTO :NEW.idE_6 FROM DUAL;
END;
/
--  E_7
CREATE OR REPLACE TRIGGER TR_E_7_KEY BEFORE INSERT ON E_7
FOR EACH ROW
BEGIN
  SELECT SEQ_E_7.NEXTVAL INTO :NEW.idE_7 FROM DUAL;
END;
/
/************** FI DEL PROCES ****************************************/
```

8 APENDICE 5 MODUL GENERAL.

Modulo de control de la tabla LOG_TFC donde queda gravado cada una de las acciones que se ejecutan mediante la procedures.

```
CREATE OR REPLACE
PACKAGE        "PKG_GENERAL" AS

procedure gravar_log_procedure (
        c_procesLog       LOG_TFC.procesLog%TYPE,
        c_dataHoraLog     LOG_TFC.dataHoraLog%TYPE,
        c_entradaLog      LOG_TFC.entradaLog%TYPE,
        c_sortidaLog      LOG_TFC.sortidaLog%TYPE,
        c_rspLog          LOG_TFC.rspLog%TYPE);

END PKG_GENERAL ;

CREATE OR REPLACE
PACKAGE BODY         "PKG_GENERAL" AS

PROCEDURE       gravar_log_procedure (
```

```
 c_procesLog        LOG_TFC.procesLog%TYPE,
  c_dataHoraLog      LOG_TFC.dataHoraLog%TYPE,
  c_entradaLog       LOG_TFC.entradaLog%TYPE,
  c_sortidaLog       LOG_TFC.sortidaLog%TYPE,
  c_rspLog             LOG_TFC.rspLog%TYPE
)
AS
  PRAGMA AUTONOMOUS_TRANSACTION;

BEGIN
  INSERT INTO LOG_TFC(procesLog, dataHoraLog, entradaLog, sortidaLog, rspLog)
      VALUES (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, c_rspLog);
      COMMIT;
-- Aquest COMMIT nomes afecta a la transaccio autonoma

END gravar_log_procedure;
END PKG_GENERAL;
```

9 APENDICE 6 MODUL ADMINISTRACIO.

Este modulo contiene los paquetes par gestionar las tablas de CLIENTES, FABRICANTES, Y CONTRATOS.

Sea considerado definirlo así por ser las acciones más usuales concernientes a un departamento de administración o comercial de un empresa.

Definición de variables y estructura de las procedures dentro el PACKAGE.

Se define:

9.1 TRACTAMIENTO DE CLIENTES

DEFINICION	PROCEDIMIENTO
Alta de Clientes	PRC_ALTA_CLIENT
Modificar Clientes	PRC_MODIFICAR_CLIENT
Baja de un cliente	PRC_BAIXA_CLIENT
Consultar un cliente según DNI	PRC_CONSULTAR_DNI_CLIENT

```
CREATE OR REPLACE
PACKAGE        "GESTION_CLIENT" AS
PROCEDURE PRC_ALTA_CLIENT(
   p_dniClient in CLIENT.dniClient%Type,
   p_dataAltaCLient in CLIENT.dataAltaCLient%Type,
   p_estatClient in CLIENT.estatClient%Type,
   p_nomClient in CLIENT.nomClient%Type,
   p_cognom1Client in CLIENT.cognom1Client%Type,
   p_cognom2Client in CLIENT.cognom2Client%Type,
   p_idUbicaClient in CLIENT.idUbicaClient%Type,
   p_idTipoClient in CLIENT.idTipoClient%Type,
   p_observacioClient in CLIENT.observacioClient%Type,
   s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_MODIFICAR_CLIENT(
   p_idTipoClient in CLIENT.idTipoClient%Type,
   p_dniClient in CLIENT.dniClient%Type,
   p_nomClient in CLIENT.nomClient%Type,
   p_cognom1Client in CLIENT.cognom1Client%Type,
   p_cognom2Client in CLIENT.cognom2Client%Type,
   p_estatClient in CLIENT.estatClient%Type,
   p_idUbicaClient in CLIENT.idUbicaClient%Type,
   p_dataAltaCLient in CLIENT.dataAltaCLient%Type,
   p_dataModificacioClient in CLIENT.dataModificacioCLient%Type,
   p_observacioClient in CLIENT.observacioClient%Type,
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_BAIXA_CLIENT(
```

```
    p_idClient in CLIENT.idClient%TYPE,
    p_estatClient in CLIENT.estatClient%Type,
    p_dataModificacioClient in CLIENT.dataModificacioCLient%Type,
    p_dniClient in CLIENT.dniClient%Type,
    s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_CONSULTAR_DNI_CLIENT(
  p_dniClient in CLIENT.dniClient%Type,
  s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_CONSULTAR_CLIENT(
  s_rsp out NOCOPY VARCHAR2);
END GESTION_CLIENT;
```

9.1.1 PROCEDIMIENTO SPL PACKAGE CLIENTE.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_CLIENT" AS
   n_registres      NUMBER;
  s_sql          VARCHAR2 (2000);
  n_NUM_ERR          NUMBER(10);

  c_procesLog          LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp            LOG_TFC.rspLog%TYPE;

  n_texte1 VARCHAR2(30);
  n_texte2 VARCHAR2(30);
   n_DESCRIPCIOESTAT ESTAT.DESCRIPCIOESTAT%TYPE;
  n_DESCRIPCIOPERSONA PERSONA.DESCRIPCIOPERSONA%TYPE;
 n_DESCRIPCIOVIA VIA.DESCRIPCIOVIA%TYPE;
 n_DIRECCIONUBICACIO UBICACIO.DIRECCIONUBICACIO%TYPE;
 n_NUMEROUBICACIO UBICACIO.NUMEROUBICACIO%TYPE;
 n_PISUBICACIO UBICACIO.PISUBICACIO%TYPE;
 n_PORTAUBICACIO UBICACIO.PORTAUBICACIO%TYPE;
 n_CODIPOSTAL UBICACIO.CODIPOSTAL%TYPE;
 n_DESCRIPCIOLOCALITAT LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
 n_DESCRIPCIOPROVINCIA PROVINCIA.DESCRIPCIOPROVINCIA%TYPE;
 n_DESCRIPCIOPAIS PAIS.DESCRIPCIOPAIS%TYPE;

n_dataAltaClient      CLIENT.dataAltaClient%TYPE;
n_dataModificacioClient CLIENT.dataModificacioClient%TYPE;
n_estatClient CLIENT.estatClient%TYPE;
n_dniClient CLIENT.dniClient%TYPE;
n_nomClient CLIENT.nomClient%TYPE;
n_cognom1Client CLIENT.cognom1Client%TYPE;
n_cognom2Client CLIENT.cognom2Client%TYPE;
n_idUbicaClient CLIENT.idUbicaClient%TYPE;
n_idTipoClient CLIENT.idTipoClient%TYPE;
```

```
n_observacioClient CLIENT.observacioClient%TYPE;
n_idClient CLIENT.idClient%TYPE;


 n_via VIA.DESCRIPCIOVIA%TYPE;
 n_direccio UBICACIO.DIRECCIONUBICACIO%TYPE;
 n_numero UBICACIO.NUMEROUBICACIO%TYPE;
 n_pis UBICACIO.PISUBICACIO%TYPE;
 n_porta UBICACIO.PORTAUBICACIO%TYPE;
 n_cp UBICACIO.CODIPOSTAL%TYPE;
 n_localitat LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;

sortida VARCHAR2(1000):='';
e_client       EXCEPTION;

/*************************************************************************
    NOM:        PRC_ALTA_CLIENT
    DESCRIPCIÃƒâ€œ:
            Procediment encarregat de l'alta les dades d'un Client.

*************************************************************************/
PROCEDURE PRC_ALTA_CLIENT(
   p_dniClient in CLIENT.dniClient%Type,
   p_dataAltaCLient in CLIENT.dataAltaCLient%Type,
   p_estatClient in CLIENT.estatClient%Type,
   p_nomClient in CLIENT.nomClient%Type,
   p_cognom1Client in CLIENT.cognom1Client%Type,
   p_cognom2Client in CLIENT.cognom2Client%Type,
   p_idUbicaClient in CLIENT.idUbicaClient%Type,
   p_idTipoClient in CLIENT.idTipoClient%Type,
   p_observacioClient in CLIENT.observacioClient%Type,
   s_rsp out NOCOPY VARCHAR2)
 AS
 n_controlEstat number;
  BEGIN

 --Parametres per la taula LOG
    c_procesLog := 'PRC_ALTA_CLIENT';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :=   'dniClient: ' || p_dniClient ||
                ', data Alta Client: ' || p_dataAltaClient ||
                --', estat Client: ' || p_estatClient ||
                ', nom Client: ' || p_nomClient ||
                ', cognom1 Client: ' || p_cognom1Client ||
                ', cognom2 Client: ' || p_cognom2Client ||
                ', id Ubica Client: ' || p_idUbicaClient ||
                ', id Tipo Client: ' || p_idTipoClient ||
                 --, p_dataModificacioClient: ' || p_dataModificacioClient ||
                ', observacioClient: ' || p_observacioClient;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('        ALTA DE CLIENTS       ');
```

```
    DBMS_OUTPUT.PUT_LINE('-------------------------------- ');
    DBMS_OUTPUT.PUT_LINE(' dniClient: ' || p_dniClient ||
                ', data Alta Client: ' || p_dataAltaClient ||
                --', estat Client: ' || p_estatClient ||
                ', nom Client: ' || p_nomClient ||
                ', cognom1 Client: ' || p_cognom1Client ||
                ', cognom2 Client: ' || p_cognom2Client ||
                ', id Ubica Client: ' || p_idUbicaClient ||
                ', id Tipo Client: ' || p_idTipoClient ||
                 --, p_dataModificacioClient: ' || p_dataModificacioClient ||
                ', observacioClient: ' || p_observacioClient);
  c_sortidalog := 's_rsp';

  n_controlEstat:=1;
-- Comprovem que el dni client  no sigui NULL
  If p_dniClient IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA DE CLIENTS: ');
    s_rsp := 'Falta especificar el DNI';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_client;
  END IF;
  SELECT COUNT(PERSONA.IDPERSONA) INTO n_idTipoClient
  FROM PERSONA
  WHERE PERSONA.IDPERSONA=p_idTipoClient;
 -- Comprovem els parÃƒÂ metres NULL i advertim si falta algun parÃƒÂ metre
  IF n_idTipoClient=0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA DE CLIENTS: ');
    s_rsp := 'Falta especificar el tipu de client.';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_client;
  END IF;
  If p_nomClient  IS NULL THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('ALTA DE CLIENTS: ');
   s_rsp := 'Falta especificar el nom ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_client;
  END IF;
  If p_cognom1Client   IS NULL THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('ALTA DE CLIENTS: ');
   s_rsp := 'Falta especificar cognom1 ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_client;
  END IF;
  If p_cognom2Client   IS NULL THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('ALTA DE CLIENTS: ');
   s_rsp := 'Falta especificar cognom2 ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
```

```
    RAISE e_client;
  END IF;
  If p_idUbicaClient   IS NULL THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA DE CLIENTS: ');
    s_rsp := 'Falta especificar el codi ubicacio on conte la direccio. ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_client;
  END IF;

-- Comprovem si existeix el dni client a  donar d'alta
  SELECT COUNT (*) INTO n_registres
  FROM CLIENT
  WHERE DNICLIENT=p_dniClient;

  If n_registres=0 then
 -- El donem d'alta a Persona i a CLIENT
  INSERT INTO CLIENT(
                dniClient,
                dataAltaClient,
                estatClient,
                nomClient,
                cognom1Client,
                cognom2Client,
                idUbicaClient,
                idTipoClient,
                observacioClient)
                      VALUES(
                              p_dniClient,
                              p_dataAltaClient,
                              n_controlEstat,
                              p_nomClient,
                              p_cognom1Client,
                              p_cognom2Client,
                              p_idUbicaClient,
                              p_idTipoClient,
                              p_observacioClient);

    SELECT PERSONA.DESCRIPCIOPERSONA INTO n_texte1
    FROM PERSONA
    WHERE PERSONA.IDPERSONA=p_idTipoClient;

    SELECT ESTAT.DESCRIPCIOESTAT INTO n_texte2
    FROM ESTAT
    WHERE ESTAT.IDESTAT=1;

    SELECT DISTINCT
    VIA.DESCRIPCIOVIA,
    UBICACIO.DIRECCIONUBICACIO,
    UBICACIO.NUMEROUBICACIO,
    UBICACIO.PISUBICACIO,
    UBICACIO.PORTAUBICACIO,
```

```
    UBICACIO.CODIPOSTAL,
    LOCALITAT.DESCRIPCIOLOCALITAT
    INTO n_via,n_direccio,n_numero,n_pis,n_porta,n_cp,n_localitat
    FROM VIA,UBICACIO,LOCALITAT
    WHERE UBICACIO.IDUBICA=p_idUbicaClient
    AND VIA.IDVIA= UBICACIO.IDVIAUBICACIO
    AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT;

       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('Insertat el Client:
        Dni: '||p_dniClient||
       ' Data alta:  '||p_dataAltaClient||
       ' Estat:  '||n_texte2||
       ' Tipus Client: '||n_texte1||
       ' Nom:  '||p_nomClient||
       ' Cognom1:  '||p_cognom1Client||
       ' Cognom2:  '||p_cognom2Client||
       ' Direccio:  '||n_direccio||
       ' Numero:  '||n_numero||
       ' Pis:  '||n_pis||
       ' Porta::  '||n_porta||
       ' Codi Postal:  '||n_cp||
       ' Poblacio:  '||n_localitat||
       ' Observacions: '||p_observacioClient);
       DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
       s_rsp :='Ok: Insertat el Clinet:  Insertat el Client:
        Dni: '||p_dniClient||
       ' Data alta:  '||p_dataAltaClient||
       ' Estat:  '||n_texte2||
       ' Nom:  '||p_nomClient||
       ' Cognom1:  '||p_cognom1Client||
       ' Cognom2:  '||p_cognom2Client;
       pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
       DBMS_OUTPUT.PUT_LINE(s_rsp);
      ELSE
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('     ALTA DE CLIENTS      ');
       DBMS_OUTPUT.PUT_LINE('_____________________________');
       s_rsp := 'DADES duplicades';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_client;
 END IF;

COMMIT;

  EXCEPTION
    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
      ELSE
        -- L'error si ha estat controlat per codi
```

```
        s_rsp := 'Error: ' || s_rsp;
      END IF;
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE(' ');
       DBMS_OUTPUT.put_line(s_rsp);
       pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
END PRC_ALTA_CLIENT;

/*************************************************************************
   NOM:      PRC_MODIFICAR_CLIENT
   DESCRIPCIÃƒâ€œ:
          Procediment encarregat de modificar les dades d'un contacte.
          Les dades es modificaran en la taula persona i el identificador de
          Persona apuntarÃƒÂ  a la clau primaria del contacte de la taula CLIENT
            o contacte_Proveidor.
*************************************************************************/
PROCEDURE PRC_MODIFICAR_CLIENT(
   p_idTipoClient in CLIENT.idTipoClient%Type,
   p_dniClient in CLIENT.dniClient%Type,
   p_nomClient in CLIENT.nomClient%Type,
   p_cognom1Client in CLIENT.cognom1Client%Type,
   p_cognom2Client in CLIENT.cognom2Client%Type,
   p_estatClient in CLIENT.estatClient%Type,
   p_idUbicaClient in CLIENT.idUbicaClient%Type,
   p_dataAltaCLient in CLIENT.dataAltaCLient%Type,
   p_dataModificacioClient in CLIENT.dataModificacioCLient%Type,
   p_observacioClient in CLIENT.observacioClient%Type,
   s_rsp out NOCOPY VARCHAR2)

AS
 n_dni VARCHAR2(15);
BEGIN
--Parametres per la taula LOG-----------------------------------------------------------
  c_procesLog := 'PRC_MODIFICAR_CLIENT';
  c_dataHoraLog := SYSDATE;
  c_entradaLog := 'dni Client: ' || p_dniClient ||
                          ', nom Client: ' || p_nomClient ||
             ', cognom1 Client: ' || p_cognom1Client ||
              ', cognom2 Client: ' || p_cognom2Client ||
                          --', estat Client: ' || p_estatClient ||
              ', id Ubica Client: ' || p_idUbicaClient ||
             ', id Tipo Client: ' || p_idTipoClient ||
              --', data Alta Client: ' || p_dataAltaClient ||
                          ', data Modificacio Client: ' || p_dataModificacioClient;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('         MODIFICAR UN CLIENT        ');
  DBMS_OUTPUT.PUT_LINE('------------------------------------ ');
  DBMS_OUTPUT.PUT_LINE(' dni Client: ' || p_dniClient ||
                          ', nom Client: ' || p_nomClient ||
             ', cognom1 Client: ' || p_cognom1Client ||
```

```
            ', cognom2 Client: ' || p_cognom2Client ||
                    ', estat Client: ' || p_estatClient ||
            ', id Ubica Client: ' || p_idUbicaClient ||
           ', id Tipo Client: ' || p_idTipoClient ||
            ', data Alta Client: ' || p_dataAltaClient ||
                    ', data Modificacio Client: ' || p_dataModificacioClient);
  c_sortidalog := 's_rsp';

-- Comprovem que p_dniClient no sigui NULL----------------------------------------------
  If p_dniClient IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UN CLIENT: ');
    s_rsp := 'Falta especificarel NIF de Client';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_client;
  END IF;
  --1er. Comprovem que el client existeix------------------------------
  SELECT COUNT (*) INTO n_registres
  FROM CLIENT
  WHERE dniClient= p_dniClient;
  If n_registres=0 then
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR CLIENT:  ');
     s_rsp := 'Persona no existent a la BBDD';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_client;
  END IF;
  -- Comprovem els parÃƒÂ metres NULL i advertim si falta algun parÃƒÂ metre
  If (p_nomClient  IS NULL) or (p_cognom1Client IS NULL) or (p_cognom2Client  IS NULL)
      or (p_idUbicaClient IS NULL) or (p_dataModificacioClient IS NULL)
      OR (p_idTipoClient IS NULL) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR CLIENT: ');
     s_rsp := 'Falta especificar algun pararametre del client.  ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_client;
   END IF;

   SELECT CLIENT.DNICLIENT INTO n_dni
   FROM CLIENT
   WHERE CLIENT.DNICLIENT=p_dniClient;

   s_sql := 'UPDATE CLIENT SET ';
--Construirem la sentencia UPDATE segons si hi ha valor en els parÃƒÂ metres-
    IF p_nomClient IS NOT NULL THEN
    -- Modifiquem el nom
       s_sql := s_sql || 'NOMCLIENT='' || p_nomClient || '',';
    END IF;
    IF p_cognom1Client IS NOT NULL THEN
    -- Modifiquem el cognom1----------------------------------------
      s_sql := s_sql || 'COGNOM1CLIENT='' || p_cognom1Client || '',';
    END IF;
```

```
    IF p_cognom2Client IS NOT NULL THEN
    -- Modifiquem el cognom2---------------------------------------
       s_sql := s_sql || 'COGNOM2CLIENT=''' || p_cognom2Client || ''',';
    END IF;
    -- Modificar data modificacio
    IF p_dataModificacioClient IS NOT NULL THEN
      s_sql := s_sql || 'DATAMODIFICACIOCLIENT=''' ||p_dataModificacioClient|| ''',';
    END IF;
    IF p_idUbicaClient IS NOT NULL THEN
    -- Modifiquem l'adreca-----------------------------------------
      s_sql := s_sql || 'IDUBICACLIENT=''' || p_idUbicaClient || ''',';
    END IF;
    IF p_idTipoClient IS NOT NULL THEN
    -- Modifiquem id Tipo Client----------------------------------------
      s_sql := s_sql || 'IDTIPOCLIENT=''' || p_idTipoClient || ''',';
    END IF;
-- Eliminem la coma final de la sentencia SQL-------------------------
    s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
    s_sql := s_sql || ' WHERE DNICLIENT=''' ||n_dni || '''';

EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0 THEN
--L'error serÃƒÂ  que el DNI ja el tenim en la taula i no admet duplicats--
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR CLIENT: ');
     s_rsp := 'Actualitzacio no realitzada';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_client;
   ELSE
      SELECT PERSONA.DESCRIPCIOPERSONA INTO n_texte1
      FROM PERSONA
      WHERE PERSONA.IDPERSONA=p_idTipoClient;

      SELECT ESTAT.DESCRIPCIOESTAT INTO n_texte2
      FROM ESTAT
      WHERE ESTAT.IDESTAT=1;

      SELECT
      VIA.DESCRIPCIOVIA,
      UBICACIO.DIRECCIONUBICACIO,
      UBICACIO.NUMEROUBICACIO,
      UBICACIO.PISUBICACIO,
      UBICACIO.PORTAUBICACIO,
      UBICACIO.CODIPOSTAL,
      LOCALITAT.DESCRIPCIOLOCALITAT
      INTO n_via,n_direccio,n_numero,n_pis,n_porta,n_cp,n_localitat
      FROM VIA,UBICACIO,LOCALITAT
      WHERE UBICACIO.IDUBICA=p_idUbicaClient
      AND VIA.IDVIA= UBICACIO.IDVIAUBICACIO
      AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT;
```

```
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('Modificacio feta al Client:
          Dni: '||p_dniClient||
         ' Data Modificacio:  '||p_dataModificacioClient||
         ' Estat:  '||n_texte2||
         ' Tipus Client: '||n_texte1||
         ' Nom:  '||p_nomClient||
         ' Cognom1:  '||p_cognom1Client||
         ' Cognom2:  '||p_cognom2Client||
         ' Direccio:  '||n_direccio||
         ' Numero:  '||n_numero||
         ' Pis:  '||n_pis||
         ' Porta::  '||n_porta||
         ' Codi Postal:  '||n_cp||
         ' Poblacio:  '||n_localitat||
         ' Observacions: '||p_observacioClient);
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
        s_rsp := 'OK Modificacio al Client:
         Dni: '||p_dniClient||
         ' Data alta:  '||p_dataAltaClient||
         ' Estat:  '||n_texte2||
         ' Nom:  '||p_nomClient||
         ' Cognom1:  '||p_cognom1Client||
         ' Cognom2:  '||p_cognom2Client;
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;
EXCEPTION
    WHEN DUP_VAL_ON_INDEX THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR CLIENT:  ');
      s_rsp := 'Error: NIF duplicat. No es poden fer les modificacions';
      DBMS_OUTPUT.put_line (s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;

    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
        -- L'error si ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.put_line (s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
end PRC_MODIFICAR_CLIENT;

/*************************************************************************
```

```
      NOM:        PRC_BAIXA_CLIENT
      DESCRIPCIÃƒâ€œ:
            Procediment encarregat de donar de baixa  un CLIENT.
*********************************************************************/
PROCEDURE PRC_BAIXA_CLIENT(
        p_idClient in CLIENT.idClient%TYPE,
        p_estatClient in CLIENT.estatClient%Type,
        p_dataModificacioClient in CLIENT.dataModificacioCLient%Type,
        p_dniClient in CLIENT.dniClient%Type,
        s_rsp out NOCOPY VARCHAR2)
AS
  n_controlEstat number;
  n_dni VARCHAR2(15);
BEGIN
--Parametres per la taula LOG-----------------------------------------------------------
  c_procesLog := 'PRC_BAIXA_CLIENT';
  c_dataHoraLog := SYSDATE;
  c_entradaLog :=' dni Client: ' || p_dniClient||
          ' data Modificacio Client: ' || p_dataModificacioClient;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('        BAIXA CLIENT         ');
  DBMS_OUTPUT.PUT_LINE('--------------------------------');
  DBMS_OUTPUT.PUT_LINE(' dni Client: ' || p_dniClient||
          ' data Modificacio Client: ' || p_dataModificacioClient);
  c_sortidalog := 's_rsp';
  n_controlEstat:=2;
-- Comprovem que el idClient
  If  p_dniClient IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('BAIXA CLIENT: ');
    s_rsp := 'Falta especificar el DNI Client.  ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_client;
  END IF;
-- Comprovem que la data modificacio CLIENT no sigui NULL-------------------------------------------
  If p_dataModificacioClient IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('BAIXA CLIENT: ');
    s_rsp := 'Falta especificar el NIF del Client';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_client;
  END IF;
--1er. Comprovem que l'estat del Client sigui actiu=1-----------------------------
  SELECT COUNT (*) INTO n_registres
  FROM CLIENT
  WHERE dniClient=p_dniClient
  AND CLIENT.estatClient=1;
-- MIRO SI ESTA EN SITUACIO ALTA
 If n_registres=0 then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('BAIXA_CLIENT: ');
```

```
  s_rsp := 'Client no existent a la BBDD';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_client;
END IF;
-- RECUPERO EL DNI SI ESTA EN SITUACIO 1=ALTA
  SELECT dniClient INTO n_texte2
  FROM CLIENT
  WHERE dniClient=p_dniClient
  AND CLIENT.estatClient=1;

  IF p_dniClient IS NOT NULL  THEN
    s_sql := 'UPDATE CLIENT SET dniClient = ''' ||p_dniClient || '''';
    s_sql := 'UPDATE CLIENT SET estatClient='''||n_controlEstat||'''';
    s_sql := 'UPDATE CLIENT SET dataModificacioClient='''||p_dataModificacioClient||'''';
-- Eliminem la coma final de la sentencia SQL-------------------------
    s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
    s_sql := s_sql || ' WHERE DNICLIENT=''' ||n_dni || '''';
  END IF;
EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('BAIXA_CLIENT:  ');
    s_rsp := 'Actualitzacio no realitzada';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_client;
  ELSE
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Client donat de Baixa:
    DNI: '||p_dniClient||
    ' Data Modificacio: '||p_dataModificacioClient );
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  s_rsp := 'OK: Client donat de Baixa:
    DNI: '||p_dniClient||
    ' Data Modificacio: '||p_dataModificacioClient;
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;

COMMIT;
 EXCEPTION

  WHEN OTHERS THEN
   IF s_rsp IS NULL THEN
--L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
-- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
   END IF;
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE(' ');
       DBMS_OUTPUT.put_line (s_rsp);
```

```
        pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
ROLLBACK;
end PRC_BAIXA_CLIENT;

/*************************************************************************
     NOM:       PRC_CONSULTAR_DNI_CLIENT
     DESCRIPCIÃƒâ€œ:
              Procediment encarregat de consultar les dades d'un contacte
              i mostrarles per pantalla.
*************************************************************************/
PROCEDURE PRC_CONSULTAR_DNI_CLIENT(
 p_dniClient in CLIENT.dniClient%Type,
 s_rsp out NOCOPY VARCHAR2)
AS
n_IDCLIENT number;
BEGIN
 --Parametres per la taula LOG-----------------------------------------------------
     c_procesLog := 'PRC_CONSULTAR_CLIENT';
     c_dataHoraLog := SYSDATE;
     c_entradaLog :=' Dni Client: ' || p_dniClient;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('        CONSULTA CLIENT         ');
    DBMS_OUTPUT.PUT_LINE('-----------------------------------');
    DBMS_OUTPUT.PUT_LINE('Dni Client: ' || p_dniClient);
    c_sortidalog := 's_rsp';

 -- Comprovem que p_dniClient no sigui NULL-----------------------------------------
   If p_dniClient IS NULL then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('CONSULTA CLIENT : ');
      s_rsp := 'Falta especificar el DNI del Client';
      DBMS_OUTPUT.put_line (s_rsp);
      RAISE e_client;
   END IF;
 --1er. Comprovem que existeixi la persona-----------------------------
        SELECT COUNT (*) INTO n_registres
        FROM CLIENT
        WHERE CLIENT.DNICLIENT= p_dniClient;

 If n_registres=0 then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('CONSULTAR CLIENT: ');
         s_rsp := 'Client no existent a la BBDD';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
         RAISE e_client;
    END IF;

 -- Recuperem les dades de la taula Client-------------------------
SELECT
    CLIENT.IDCLIENT,
    CLIENT.DNICLIENT,
    CLIENT.DATAALTACLIENT,
```

```
    ESTAT.DESCRIPCIOESTAT,
    CLIENT.DATAMODIFICACIOCLIENT,
    CLIENT.NOMCLIENT,
    CLIENT.COGNOM1CLIENT,
    CLIENT.COGNOM2CLIENT,
    PERSONA.DESCRIPCIOPERSONA,
    VIA.DESCRIPCIOVIA,
    UBICACIO.DIRECCIONUBICACIO,
    UBICACIO.NUMEROUBICACIO,
    UBICACIO.PISUBICACIO,
    UBICACIO.PORTAUBICACIO,
    UBICACIO.CODIPOSTAL,
    LOCALITAT.DESCRIPCIOLOCALITAT,
    PROVINCIA.DESCRIPCIOPROVINCIA,
    PAIS.DESCRIPCIOPAIS
    INTO
    n_IDCLIENT,
    n_DNICLIENT,
    n_DATAALTACLIENT,
    n_DESCRIPCIOESTAT,
    n_DATAMODIFICACIOCLIENT,
    n_NOMCLIENT,
    n_COGNOM1CLIENT,
    n_COGNOM2CLIENT,
    n_DESCRIPCIOPERSONA,
    n_DESCRIPCIOVIA,
    n_DIRECCIONUBICACIO,
    n_NUMEROUBICACIO,
    n_PISUBICACIO,
    n_PORTAUBICACIO,
    n_CODIPOSTAL,
    n_DESCRIPCIOLOCALITAT,
    n_DESCRIPCIOPROVINCIA,
    n_DESCRIPCIOPAIS
FROM CLIENT,VIA,UBICACIO,LOCALITAT,PROVINCIA,PAIS,ESTAT,PERSONA
WHERE CLIENT.DNICLIENT=p_dniClient
AND CLIENT.IDTIPOCLIENT=PERSONA.IDPERSONA
AND CLIENT.ESTATCLIENT=ESTAT.IDESTAT
AND CLIENT.IDUBICACLIENT=UBICACIO.IDUBICA
AND UBICACIO.IDVIAUBICACIO=VIA.IDVIA
AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT
AND LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS;

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTAR CLIENT: ');
   DBMS_OUTPUT.PUT_LINE(
   '   Codi Client: '
|| n_IDCLIENT||
' Dni: '
||n_DNICLIENT||
' Data alta: '
```

```
||n_DATAALTACLIENT||
' Estat: '
||n_DESCRIPCIOESTAT||
' Data ultima modificacio: '
||n_DATAMODIFICACIOCLIENT||
' Nom: '
||n_NOMCLIENT||
' Cognom1: '
||n_COGNOM1CLIENT||
' Cognom2: '
||n_COGNOM2CLIENT||
' Tipus de Client: '
||n_DESCRIPCIOPERSONA||
/*****************************
' Via Publica: '
||n_DESCRIPCIOVIA||
' Direccio: '
||n_DIRECCIONUBICACIO||
' Numero:  '
||n_NUMEROUBICACIO||
' Pis:  '
||n_PISUBICACIO||
' Porta:  '
||n_PORTAUBICACIO||
' Codi Postal:  '
||n_CODIPOSTAL||
 *******************************/
' Poblacio: '
||n_DESCRIPCIOLOCALITAT||
' Provincia: '
||n_DESCRIPCIOPROVINCIA||
' Pais:  '
||n_DESCRIPCIOPAIS);

DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
 s_rsp := ('OK: CONSULTA CLIENT: '||p_dniClient);
 DBMS_OUTPUT.put_line (s_rsp);
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

COMMIT;

  EXCEPTION

    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
-- L'error no ha estat controlat per codi----------------------------------------
         s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
-- L'error si ha estat controlat per codi----------------------------------------
          s_rsp := 'Error: ' || s_rsp;
      END IF;
        DBMS_OUTPUT.ENABLE;
```

```
        DBMS_OUTPUT.PUT_LINE(' ');
        DBMS_OUTPUT.put_line (s_rsp);
        pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
        ROLLBACK;
end PRC_CONSULTAR_DNI_CLIENT;

-- CONSULTA DELS CLIENTS
PROCEDURE PRC_CONSULTAR_CLIENT(
  s_rsp out NOCOPY VARCHAR2)
AS
n_IDCLIENT number;

CURSOR C_CLIENT IS
SELECT
    CLIENT.IDCLIENT,
    CLIENT.DNICLIENT,
    CLIENT.DATAALTACLIENT,
    ESTAT.DESCRIPCIOESTAT,
    CLIENT.DATAMODIFICACIOCLIENT,
    CLIENT.NOMCLIENT,
    CLIENT.COGNOM1CLIENT,
    CLIENT.COGNOM2CLIENT,
    PERSONA.DESCRIPCIOPERSONA,
    VIA.DESCRIPCIOVIA,
    UBICACIO.DIRECCIONUBICACIO,
    UBICACIO.NUMEROUBICACIO,
    UBICACIO.PISUBICACIO,
    UBICACIO.PORTAUBICACIO,
    UBICACIO.CODIPOSTAL,
    LOCALITAT.DESCRIPCIOLOCALITAT,
    PROVINCIA.DESCRIPCIOPROVINCIA,
    PAIS.DESCRIPCIOPAIS
FROM CLIENT,VIA,UBICACIO,LOCALITAT,PROVINCIA,PAIS,ESTAT,PERSONA
WHERE CLIENT.DNICLIENT=CLIENT.DNICLIENT
AND CLIENT.IDTIPOCLIENT=PERSONA.IDPERSONA
AND CLIENT.ESTATCLIENT=ESTAT.IDESTAT
AND CLIENT.IDUBICACLIENT=UBICACIO.IDUBICA
AND UBICACIO.IDVIAUBICACIO=VIA.IDVIA
AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT
AND LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS;

BEGIN
  c_procesLog := 'PRC_CONSULTA_VIAS';
  c_dataHoraLog := SYSDATE;
  c_entradaLog :='CONSULTA CLIENTS';
  c_sortidalog := 's_rsp';
 SELECT DISTINCT COUNT(*) INTO n_registres
 FROM CLIENT
 WHERE CLIENT.DNICLIENT=CLIENT.DNICLIENT
 AND CLIENT.IDCLIENT=CLIENT.IDCLIENT;
```

```
IF n_registres=0 THEN
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('CONSULTA CLIENT: ');
  s_rsp := 'No hi han clients donades en al BDD. ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_client;
ELSE
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE('          CONSULTA DE CLIENTS                    ');
DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------------');
OPEN C_CLIENT;
FETCH C_CLIENT INTO
    n_IDCLIENT,
    n_DNICLIENT,
    n_DATAALTACLIENT,
    n_DESCRIPCIOESTAT,
    n_DATAMODIFICACIOCLIENT,
    n_NOMCLIENT,
    n_COGNOM1CLIENT,
    n_COGNOM2CLIENT,
    n_DESCRIPCIOPERSONA,
    n_DESCRIPCIOVIA,
    n_DIRECCIONUBICACIO,
    n_NUMEROUBICACIO,
    n_PISUBICACIO,
    n_PORTAUBICACIO,
    n_CODIPOSTAL,
    n_DESCRIPCIOLOCALITAT,
    n_DESCRIPCIOPROVINCIA,
    n_DESCRIPCIOPAIS;
 WHILE C_CLIENT%FOUND LOOP
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(
  '   Codi Client: '
|| n_IDCLIENT||
' Dni: '
||n_DNICLIENT||
' Data alta: '
||n_DATAALTACLIENT||
' Estat: '
||n_DESCRIPCIOESTAT||
' Data ultima modificacio: '
||n_DATAMODIFICACIOCLIENT||
' Nom: '
||n_NOMCLIENT||
' Cognom1: '
||n_COGNOM1CLIENT||
' Cognom2: '
||n_COGNOM2CLIENT||
' Tipus de Client: '
||n_DESCRIPCIOPERSONA||
/*********************************
```

```
' Via Publica: '
||n_DESCRIPCIOVIA||
' Direccio: '
||n_DIRECCIONUBICACIO||
' Numero:  '
||n_NUMEROUBICACIO||
' Pis:  '
||n_PISUBICACIO||
' Porta:  '
||n_PORTAUBICACIO||
' Codi Postal:  '
||n_CODIPOSTAL||
**************************/
' Poblacio: '
||n_DESCRIPCIOLOCALITAT||
' Provincia: '
||n_DESCRIPCIOPROVINCIA||
' Pais:  '
||n_DESCRIPCIOPAIS);
 s_rsp := 'OK  CONSULTA CLIENTS
 Codi Client: '
|| n_IDCLIENT
||' Dni: '
||n_DNICLIENT;
 FETCH C_CLIENT INTO
    n_IDCLIENT,
    n_DNICLIENT,
    n_DATAALTACLIENT,
    n_DESCRIPCIOESTAT,
    n_DATAMODIFICACIOCLIENT,
    n_NOMCLIENT,
    n_COGNOM1CLIENT,
    n_COGNOM2CLIENT,
    n_DESCRIPCIOPERSONA,
    n_DESCRIPCIOVIA,
    n_DIRECCIONUBICACIO,
    n_NUMEROUBICACIO,
    n_PISUBICACIO,
    n_PORTAUBICACIO,
    n_CODIPOSTAL,
    n_DESCRIPCIOLOCALITAT,
    n_DESCRIPCIOPROVINCIA,
    n_DESCRIPCIOPAIS;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total clients: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_CLIENT;
END IF;
COMMIT;
  EXCEPTION
```

```
      WHEN OTHERS THEN
       IF s_rsp IS NULL THEN
 -- L'error no ha estat controlat per codi----------------------------------------
          s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
       ELSE
 -- L'error si ha estat controlat per codi----------------------------------------
           s_rsp := 'Error: ' || s_rsp;
       END IF;
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE(' ');
         DBMS_OUTPUT.put_line (s_rsp);
         pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
         ROLLBACK;
END PRC_CONSULTAR_CLIENT;
end GESTION_CLIENT;
```

9.2 TRATAMIENTO FABRICANTES:

DEFINICION	PROCEDIMIENTO
Alta de Fabricante	PRC_ALTA_FABRICANT
Modificar Fabricante	PRC_MODIFICAR_FABRICANT
Baja de un Fabricante	PRC_BAIXA_FABRICANT
Consultar un Fabricante	PRC_CONSULTAR_UN_FABRICANT
Consulta de los Fabricantes	PRC_CONSULTAR_FABRICANTS

```
CREATE OR REPLACE
PACKAGE        "GESTION_FABRICANT" AS
PROCEDURE PRC_ALTA_FABRICANT(
   p_NIFFabricant in FABRICANT.NIFFABRICANT%Type,
   p_dataAltaFabricant in FABRICANT.DATAALTAFABRICANT%Type,
   p_estatFabricant in FABRICANT.ESTATFABRICANT%Type,
   p_nomComercialFabricant in FABRICANT.NOMCOMERCIALFABRICANT%Type,
   p_idUbicaFabricant in FABRICANT.IDUBICAFABRICANT%Type,
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_MODIFICAR_FABRICANT(
   p_idFabricant in FABRICANT.IDFABRICANT%Type,
   p_NIFFabricant in FABRICANT.NIFFABRICANT%Type,
   p_dataModificaFabricant in FABRICANT.DATAMODIFCAFABRICANT%Type,
   p_estatFabricant in FABRICANT.ESTATFABRICANT%Type,
   p_nomComercialFabricant in FABRICANT.NOMCOMERCIALFABRICANT%Type,
   p_idUbicaFabricant in FABRICANT.IDUBICAFABRICANT%Type,
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_BAIXA_FABRICANT(
   p_idFabricant in FABRICANT.IDFABRICANT%Type,
   p_NIFFabricant in FABRICANT.NIFFABRICANT%Type,
   p_estatFabricant in FABRICANT.ESTATFABRICANT%Type,
   p_dataModificaFabricant in FABRICANT.DATAMODIFCAFABRICANT%Type,
   p_nomComercialFabricant in FABRICANT.NOMCOMERCIALFABRICANT%Type,
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_CONSULTAR_UN_FABRICANT(
  p_NIFFabricant in FABRICANT.NIFFABRICANT%Type,
  s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_CONSULTAR_FABRICANTS(
   s_rsp out NOCOPY VARCHAR2);
END "GESTION_FABRICANT";
```

9.2.1 PROCEDIMIENTO SPL PACKAGE FABRICANTE.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_FABRICANT" AS
  n_registres     NUMBER;
  s_sql           VARCHAR2 (2000);
  n_NUM_ERR           NUMBER(10);

  c_procesLog            LOG_TFC.procesLog%TYPE;
  c_dataHoraLog     LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog      LOG_TFC.entradaLog%TYPE;
  c_sortidaLog      LOG_TFC.sortidaLog%TYPE;
  s_rsp               LOG_TFC.rspLog%TYPE;
  --s_rsp VARCHAR2 (2000);
  n_estat NUMBER;
  n_controlEstat FABRICANT.ESTATFABRICANT%TYPE;

  n_idFabricant  FABRICANT.IDFABRICANT%Type;
  n_nifFabricant  FABRICANT.NIFFABRICANT%TYPE;
  n_dataAltaFabricant      FABRICANT.DATAALTAFABRICANT%TYPE;
```

```
  n_estatFabricant  FABRICANT.ESTATFABRICANT%TYPE;
  n_dataModificacioFabricant  FABRICANT.DATAMODIFCAFABRICANT%TYPE;
  n_nomComercialFabricant  FABRICANT.NOMCOMERCIALFABRICANT%Type;
  n_idUbicaFabricant FABRICANT.IDUBICAFABRICANT%TYPE;

  n_via VIA.DESCRIPCIOVIA%TYPE;
  n_direccioFabricant UBICACIO.DIRECCIONUBICACIO%TYPE;
  n_numeroFabricant UBICACIO.NUMEROUBICACIO%TYPE;
  n_pisFabricant UBICACIO.PISUBICACIO%TYPE;
  n_portaFabricant UBICACIO.PORTAUBICACIO%TYPE;
  n_codiPostalFabricant UBICACIO.CODIPOSTAL%TYPE;
  n_poblacioFabricant LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
  n_provinvciaFabricant PROVINCIA.DESCRIPCIOPROVINCIA%TYPE;
  n_paisFabricant PAIS.DESCRIPCIOPAIS%TYPE;
  n_descEstat ESTAT.DESCRIPCIOESTAT%TYPE;

  sortida VARCHAR2(2000):='';
  e_fabricant       EXCEPTION;
/*****************************************************************************
  ALTA DE FABRICANT
  Procediment per donar d'alta al Fabricant de models de comptadors.
*****************************************************************************/
PROCEDURE PRC_ALTA_FABRICANT(
    p_NIFFabricant in FABRICANT.NIFFABRICANT%Type,
    p_dataAltaFabricant in FABRICANT.DATAALTAFABRICANT%Type,
    p_estatFabricant in FABRICANT.ESTATFABRICANT%Type,
    p_nomComercialFabricant in FABRICANT.NOMCOMERCIALFABRICANT%Type,
    p_idUbicaFabricant in FABRICANT.IDUBICAFABRICANT%Type,
    s_rsp out NOCOPY VARCHAR2)
    AS
    BEGIN
--Parametres per la taula FABRICANT
    c_procesLog := 'PRC_ALTA_FABRICANT';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :=   ' NIF : ' ||p_NIFFabricant||
                  ', Data alta: ' ||p_dataAltaFabricant||
                  ', Nom Comercial: ' ||p_nomComercialFabricant||
                  ', Direccio: ' ||p_idUbicaFabricant;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('   ALTA FABRICANT     ');
    DBMS_OUTPUT.PUT_LINE('_____________________');
    DBMS_OUTPUT.PUT_LINE(' NIF : ' ||p_NIFFabricant||
                    ', Data alta: ' ||p_dataAltaFabricant||
                    ', Nom Comercial: ' ||p_nomComercialFabricant||
                    ', Direccio: ' ||p_idUbicaFabricant);
    c_sortidalog := 's_rsp';

-- Comprovem que el NIF no sigui NULL
  If p_NIFFabricant IS NULL then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ALTA FABRICANT: ');
```

```
      s_rsp := 'Falta especificar el NIF ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_fabricant;
   END IF;
-- Comprovem els parÃƒÂ metres NULL i advertim si falta algun parÃƒÂ metre
   if p_dataAltaFabricant  IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ALTA FABRICANT: ');
      s_rsp := 'Falta especificar la dta alta  ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_fabricant;
   END IF;
   IF p_nomComercialFabricant IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ALTA FABRICANT: ');
      s_rsp := 'Falta especificar nom comercial  ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_fabricant;
   END IF;
   IF p_idUbicaFabricant IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ALTA FABRICANT: ');
      s_rsp := 'Falta especificar el codi ubicacio   ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_fabricant;
   END IF;

-- Comprovem si existeix el NIF a  donar d'alta
   SELECT COUNT (*) INTO n_registres
   FROM FABRICANT
   WHERE FABRICANT.NIFFABRICANT=p_NIFFabricant;

   If n_registres=0 then
-- El donem d'alta a FABRICANT
   n_estat:=1;
   INSERT INTO FABRICANT(NIFFABRICANT,
                         DATAALTAFABRICANT,
                         ESTATFABRICANT,
                         NOMCOMERCIALFABRICANT,
                         IDUBICAFABRICANT)
   VALUES(p_NIFFabricant,
          p_dataAltaFabricant,
          n_estat,
          p_nomComercialFabricant,
          p_idUbicaFabricant);

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('Insertat el Fabricant: NIF: '
                        ||p_NIFFabricant||
                        ' Data alta: '
                        ||p_dataAltaFabricant||
                        ' Nom Fabricant: '
```

```
                ||p_nomComercialFabricant||
                ' Codi Ubicacio: '
                ||p_idUbicaFabricant);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  s_rsp :='Ok: Insertat el Fabricant: NIF: '
                ||p_NIFFabricant||
                ' Data alta: '
                ||p_dataAltaFabricant||
                ' Nom Fabricant: '
                ||p_nomComercialFabricant||
                ' Codi Ubicacio: '
                ||p_idUbicaFabricant;
  pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  DBMS_OUTPUT.PUT_LINE(s_rsp);
 ELSE
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('ALTA FABRICANT: ');
  s_rsp := 'NIF duplicat';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_fabricant;
END IF;
COMMIT;
  EXCEPTION
     WHEN OTHERS THEN
       IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
       ELSE
         -- L'error si ha estat controlat per codi
         s_rsp := 'Error: ' || s_rsp;
       END IF;
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE(' ');
       DBMS_OUTPUT.put_line(s_rsp);
       pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;
END PRC_ALTA_FABRICANT;
/*****************************************************************
   PROCEDIMENT MODIFICAR FABRICANT
*****************************************************************/
PROCEDURE PRC_MODIFICAR_FABRICANT(
   p_idFabricant in FABRICANT.IDFABRICANT%Type,
   p_NIFFabricant in FABRICANT.NIFFABRICANT%Type,
   p_dataModificaFabricant in FABRICANT.DATAMODIFCAFABRICANT%Type,
   p_estatFabricant in FABRICANT.ESTATFABRICANT%Type,
   p_nomComercialFabricant in FABRICANT.NOMCOMERCIALFABRICANT%Type,
   p_idUbicaFabricant in FABRICANT.IDUBICAFABRICANT%Type,
   s_rsp out NOCOPY VARCHAR2)
  AS
  BEGIN
   c_procesLog := 'PRC_MODIFICAR_FABRICANT';
   c_dataHoraLog := SYSDATE;
```

```
    c_entradaLog := 'Nif Fabricant: ' || p_NIFFabricant ||
                        ', Data Modificacio Fabricant: ' ||p_dataModificaFabricant ||
              ', Nom Comercial Fabricant: ' || p_nomComercialFabricant ||
                        ', Id Ubicacio Fabricant: ' || p_idUbicaFabricant;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('     MODIFICAR FABRICANT     ');
    DBMS_OUTPUT.PUT_LINE('_____________________________');
    DBMS_OUTPUT.PUT_LINE(' NIF Fabricant: '
                    || p_NIFFabricant ||
                    '  Data Modificacio Fabricant: '
                    ||p_dataModificaFabricant ||
                    '  Nom Comercial Fabricant: '
                    || p_nomComercialFabricant ||
                    '  Codi Ubicacio Fabricant: '
                    || p_idUbicaFabricant);
    c_sortidalog := 's_rsp';
  If p_NIFFabricant  IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR FABRICANT: ');
    s_rsp := 'Falta especificarel NIF de Fabricant';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_fabricant;
  END IF;
  If p_dataModificaFabricant IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR FABRICANT: ');
    s_rsp := 'Falta especificar la data de Fabricant ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_fabricant;
  END IF;
  If p_nomComercialFabricant IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR FABRICANT: ');
    s_rsp := 'Falta especificar Nom Comercial de Fabricant ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_fabricant;
  END IF;
  If p_idUbicaFabricant IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR FABRICANT: ');
    s_rsp := 'Falta especificar el Codi ubicacio de Fabricant ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_fabricant;
  END IF;
--1er. Comprovem  Fabricant------------------------------
  SELECT COUNT (*) INTO n_registres
  FROM FABRICANT
  WHERE FABRICANT.NIFFABRICANT=p_NIFFabricant;
-- si exiteix el FABRICANT
  If n_registres=0 then
    DBMS_OUTPUT.ENABLE;
```

```
    DBMS_OUTPUT.PUT_LINE('MODIFICAR_FABRICANT:  ');
    s_rsp := 'Fabricant no existent a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_fabricant;
  END IF;
-- recuperem dades
  SELECT FABRICANT.NIFFABRICANT INTO n_NIFFabricant
  FROM FABRICANT
  WHERE FABRICANT.NIFFABRICANT=p_NIFFabricant;
  s_sql := 'UPDATE FABRICANT SET ';
--Construirem la sentencia UPDATE segons si hi ha valor en els parÃƒÂ metres-
  IF p_NIFFabricant  IS NOT NULL THEN
-- Modifiquem el nif Fabricant
      s_sql := s_sql || 'NIFFABRICANT=''' || p_NIFFabricant || ''',';
  END IF;
  IF p_dataModificaFabricant IS NOT NULL THEN
-- Modifiquem els data Modifica Fabricant---------------------------------------
    s_sql := s_sql || 'DATAMODIFCAFABRICANT =''' || p_dataModificaFabricant || ''',';
  END IF;
  IF p_nomComercialFabricant IS NOT NULL THEN
-- Modifiquem nom Comercial Fabricant-------------------------------------------
    s_sql := s_sql || 'NOMCOMERCIALFABRICANT=''' ||p_nomComercialFabricant || ''',';
  END IF;
  IF p_idUbicaFabricant IS NOT NULL THEN
-- Modifiquem l'adreca----------------------------------------
     s_sql := s_sql || 'idUbicaFabricant=''' || p_idUbicaFabricant || ''',';
   END IF;
-- Eliminem la coma final de la sentencia SQL-------------------------
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
   s_sql := s_sql || ' WHERE NIFFABRICANT  =''' ||n_NIFFabricant|| '''';

EXECUTE IMMEDIATE s_sql;

    IF SQL%ROWCOUNT = 0  THEN
--L'error serÃƒÂ  que el DNI ja el tenim en la taula i no admet duplicats--
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR_FABRICANT: ');
      s_rsp := 'Actualitzacio no realitzada  ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_fabricant;
    ELSE
      DBMS_OUTPUT.PUT_LINE('Modificacio feta del Fabricant amd NIF: '
      ||n_NIFFabricant||
      '  Data modificacio: '||p_dataModificaFabricant||
      '  Nom Comercial: '||p_nomComercialFabricant||
      '  Codi Ubicacio: '||p_idUbicaFabricant);
      DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
      DBMS_OUTPUT.PUT_LINE('MODIFICAR_FABRICANT: ');
      s_rsp := 'OK: Modificacio feta del Fabricant amd NIF: '
            ||n_NIFFabricant||
            '  Data modificacio: '||p_dataModificaFabricant||
```

```
            ' Nom Comercial: '||p_nomComercialFabricant||
            ' Codi Ubicacio: '||p_idUbicaFabricant;
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;
COMMIT;
 EXCEPTION
  WHEN OTHERS THEN
   IF s_rsp IS NULL THEN
-- L'error no ha estat controlat per codi
     s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
   ELSE
-- L'error si ha estat controlat per codi
    s_rsp := 'Error: ' || s_rsp;
   END IF;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('  ');
   DBMS_OUTPUT.put_line(s_rsp);
   pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 ROLLBACK;
END PRC_MODIFICAR_FABRICANT;

/*******************************************************************************
PROCEDIMENT PER DONA DE BAIXA A UN FABRICANT
Segon el NIF si existeix l'estat canvia a situacio 2=BAIXA
*******************************************************************************/
PROCEDURE PRC_BAIXA_FABRICANT(
   p_idFabricant in FABRICANT.IDFABRICANT%Type,
   p_NIFFabricant in FABRICANT.NIFFABRICANT%Type,
   p_estatFabricant in FABRICANT.ESTATFABRICANT%Type,
   p_dataModificaFabricant in FABRICANT.DATAMODIFCAFABRICANT%Type,
   p_nomComercialFabricant in FABRICANT.NOMCOMERCIALFABRICANT%Type,
   s_rsp out NOCOPY VARCHAR2)
   AS
   BEGIN
   c_procesLog := 'PRC_BAIXA_FABRICANT';
   c_dataHoraLog := SYSDATE;
   c_entradaLog := ' NIF Fabricant: ' || p_NIFFabricant ||
                        '  Data de la baixa del Fabricant: '
          ||p_dataModificaFabricant;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('    BAIXA UN FABRICANT      ');
   DBMS_OUTPUT.PUT_LINE('_____________________________');
   DBMS_OUTPUT.PUT_LINE(' NIF del Fabricant: '
         || p_NIFFabricant ||
                    ', Data de la baixa del Fabricant: '
          ||p_dataModificaFabricant );
           c_sortidalog := 's_rsp';
-- DONEM EL VALOR DE BAIXA A LA VARIABLE
   n_estat:=2;
-- comprovem si el fabricant existeix a la BDD
SELECT COUNT  (*) INTO n_registres
FROM FABRICANT
```

```
WHERE FABRICANT.NIFFABRICANT=p_NIFFabricant;

 IF n_registres=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('BAIXA_FABRICANT: ');
   s_rsp := 'El Fabricant no exiteix en la BDD ha de fer-se un alta.  ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_fabricant;
 END IF;

-- Comprovem que el Nif Fabricant no estigui en blancs o null
 If (p_NIFFabricant IS NULL) then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('BAIXA_FABRICANT: ');
   s_rsp := 'Falta especificar el Nif Fabricant ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_fabricant;
 END IF;

-- si esta donat alta
SELECT DISTINCT FABRICANT.ESTATFABRICANT INTO n_controlEstat
FROM FABRICANT
WHERE FABRICANT.NIFFABRICANT=p_NIFFabricant;

 IF n_controlEstat=2 THEN
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('BAIXA FABRICANT: ');
          s_rsp := 'El fabricant: '||p_NIFFabricant||' ja esta donant de baixa. ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_fabricant;
 END IF;
-- recuperem dades del fabricant
 SELECT FABRICANT.NIFFABRICANT INTO n_NIFFabricant
 FROM FABRICANT
 WHERE FABRICANT.NIFFABRICANT=p_NIFFabricant;

 IF p_NIFFabricant IS NOT NULL  THEN
    s_sql := 'UPDATE FABRICANT SET  estatFabricant='''||n_estat||'''';
    s_sql := 'UPDATE FABRICANT SET  dataModificacioClient='''|| p_dataModificaFabricant||'''';
-- Eliminem la coma final de la sentencia SQL-------------------------
     s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
   s_sql := s_sql || ' WHERE NifFabricant=''' ||n_NIFFabricant|| '''';
 END IF;
EXECUTE IMMEDIATE s_sql;
 IF SQL%ROWCOUNT = 0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('BAIXA FABRICANT:  ');
    s_rsp := 'Actualitzacio no realitzada';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_fabricant;
 ELSE
```

```
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('BAIXA FABRICANT:  ');
    DBMS_OUTPUT.PUT_LINE('Actualitzacio de baixa del Fabricant: '||n_NIFFabricant||
    ' data de la baixa: '||p_dataModificaFabricant||
    ' estat: '||n_estat||'= Baixa ha estat satisfactoria.');
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
    s_rsp := 'OK Actualitzacio de baixa del Fabricant: '||n_NIFFabricant||
    ' data de la baixa: '||p_dataModificaFabricant||
    ' estat: '||n_estat||'= Baixa ha estat satisfactoria.';
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;
COMMIT;
EXCEPTION
     WHEN OTHERS THEN
       IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
       ELSE
         -- L'error si ha estat controlat per codi
         s_rsp := 'Error: ' || s_rsp;
       END IF;
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE(' ');
       DBMS_OUTPUT.put_line(s_rsp);
       pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
       ROLLBACK;
END PRC_BAIXA_FABRICANT;
/*****************************************************************************
PROCEDIMENT CONSULTAR AMB UN NIF FABRICANT
Segon el NIF si existeix l'estat canvia a situacio 2=BAIXA
*****************************************************************************/
PROCEDURE PRC_CONSULTAR_UN_FABRICANT(
  p_NIFFabricant in FABRICANT.NIFFABRICANT%Type,
  s_rsp out NOCOPY VARCHAR2)
 AS
 CURSOR C_FABRIANT IS
SELECT  DISTINCT   FABRICANT.IDFABRICANT AS CODI,
         FABRICANT.NIFFABRICANT AS NIF,
         FABRICANT.DATAALTAFABRICANT AS DATA_ALTA,
         ESTAT.DESCRIPCIOESTAT AS ESTAT,
         FABRICANT.DATAMODIFCAFABRICANT AS DATA_MODIFICACIO,
         FABRICANT.NOMCOMERCIALFABRICANT AS NOM_COMERCIAL,
         VIA.DESCRIPCIOVIA AS VIA_PUBLICA,
         UBICACIO.DIRECCIONUBICACIO AS DIRECCIO,
         UBICACIO.NUMEROUBICACIO AS NUMERO,
         UBICACIO.PISUBICACIO AS PIS,
         UBICACIO.PORTAUBICACIO AS PORTA,
         UBICACIO.CODIPOSTAL AS CODI_POSTAL,
         LOCALITAT.DESCRIPCIOLOCALITAT AS POBLACIO,
         PROVINCIA.DESCRIPCIOPROVINCIA AS PROVINCIA,
         PAIS.DESCRIPCIOPAIS AS PAIS
    FROM FABRICANT,ESTAT,UBICACIO,LOCALITAT,PROVINCIA,PAIS,VIA
```

```
    WHERE FABRICANT.NIFFABRICANT=p_NIFFabricant
        AND FABRICANT.ESTATFABRICANT=ESTAT.IDESTAT
        AND FABRICANT.IDUBICAFABRICANT=UBICACIO.IDUBICA
        AND LOCALITAT.IDLOCALITAT=UBICACIO.IDLOCALITATUBICACIO
        AND LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
        AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS
        AND UBICACIO.IDVIAUBICACIO=VIA.IDVIA
       AND FABRICANT.IDFABRICANT=FABRICANT.IDFABRICANT
       ORDER BY FABRICANT.IDFABRICANT;

BEGIN
 c_procesLog := 'PRC_CONSULTAR_UN_FABRICANT';
 c_dataHoraLog := SYSDATE;
 c_entradaLog := 'NIF Fabricant: ' || p_NIFFabricant;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('    CONSULTA DADES UN FABRICANT    ');
  DBMS_OUTPUT.PUT_LINE('_____________________________________');
  DBMS_OUTPUT.PUT_LINE(' NIF Fabricant: '
                || p_NIFFabricant);
  c_sortidalog := 's_rsp';
        -- Comprovem que Nif Fabricant no sigui NULL----------------------------------------------
 If p_NIFFabricant IS NULL then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('CONSULTA DADES UN FABRICANT : ');
  s_rsp := 'Falta especificar el Nif Fabricant ';
  DBMS_OUTPUT.put_line (s_rsp);
  RAISE  e_fabricant;
 END IF;
--1er. Comprovem que existeixi Fabricant------------------------------
 SELECT DISTINCT COUNT (FABRICANT.NIFFABRICANT) INTO n_registres
 FROM FABRICANT
 WHERE FABRICANT.NIFFABRICANT=p_NIFFabricant;

 If n_registres=0 then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('CONSULTAR FABRICANT:  ');
  s_rsp := 'Fabricant no existent a la BBDD';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_fabricant;
 END IF;
OPEN C_FABRIANT;
  FETCH C_FABRIANT INTO
  n_idFabricant,
  n_nifFabricant,
  n_dataAltaFabricant,
  n_descEstat,
  n_dataModificacioFabricant,
  n_nomComercialFabricant,
  n_via,n_direccioFabricant,
  n_numeroFabricant,
  n_pisFabricant,
```

```
    n_portaFabricant,
    n_codiPostalFabricant,
    n_poblacioFabricant,
    n_provinvciaFabricant,
    n_paisFabricant;
  WHILE C_FABRIANT%FOUND LOOP
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('
   Codi Fabricant:'||n_idFabricant||
  ' NIF: '||n_nifFabricant||
  ' Data Alta: '||n_dataAltaFabricant||
  ' Estat: '||n_descEstat ||
  ' Data ultima modificacio: '||n_dataModificacioFabricant||
  ' Nom Comercial: '||n_nomComercialFabricant);
    s_rsp :='OK  CONSULTAR UN FABRICANT:
    Codi Fabricant:'||n_idFabricant||
  ' NIF: '||n_nifFabricant||
  ' Data Alta: '||n_dataAltaFabricant||
  ' Estat: '||n_descEstat||
  ' Data ultima modificacio: '||n_dataModificacioFabricant||
  ' Nom Comercial: '||n_nomComercialFabricant;
FETCH C_FABRIANT INTO
    n_idFabricant,
    n_nifFabricant,
    n_dataAltaFabricant,
    n_descEstat,
    n_dataModificacioFabricant,
    n_nomComercialFabricant,
    n_via,n_direccioFabricant,
    n_numeroFabricant,
    n_pisFabricant,
    n_portaFabricant,
    n_codiPostalFabricant,
    n_poblacioFabricant,
    n_provinvciaFabricant,
    n_paisFabricant;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total fabricants: '||n_nifFabricant);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_FABRIANT;

COMMIT;
 EXCEPTION
     WHEN OTHERS THEN
       IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
       ELSE
         -- L'error si ha estat controlat per codi
         s_rsp := 'Error: ' || s_rsp;
```

```
        END IF;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE(' ');
        DBMS_OUTPUT.put_line(s_rsp);
        pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
        ROLLBACK;
END PRC_CONSULTAR_UN_FABRICANT;

PROCEDURE PRC_CONSULTAR_FABRICANTS(
  s_rsp out NOCOPY VARCHAR2)
 AS
 CURSOR C_FABRIANT IS
SELECT  DISTINCT   FABRICANT.IDFABRICANT AS CODI,
         FABRICANT.NIFFABRICANT AS NIF,
         FABRICANT.DATAALTAFABRICANT AS DATA_ALTA,
         ESTAT.DESCRIPCIOESTAT AS ESTAT,
         FABRICANT.DATAMODIFCAFABRICANT AS DATA_MODIFICACIO,
         FABRICANT.NOMCOMERCIALFABRICANT AS NOM_COMERCIAL,
         VIA.DESCRIPCIOVIA AS VIA_PUBLICA,
         UBICACIO.DIRECCIONUBICACIO AS DIRECCIO,
         UBICACIO.NUMEROUBICACIO AS NUMERO,
         UBICACIO.PISUBICACIO AS PIS,
         UBICACIO.PORTAUBICACIO AS PORTA,
         UBICACIO.CODIPOSTAL AS CODI_POSTAL,
         LOCALITAT.DESCRIPCIOLOCALITAT AS POBLACIO,
         PROVINCIA.DESCRIPCIOPROVINCIA AS PROVINCIA,
         PAIS.DESCRIPCIOPAIS AS PAIS
     FROM FABRICANT,ESTAT,UBICACIO,LOCALITAT,PROVINCIA,PAIS,VIA
     WHERE FABRICANT.NIFFABRICANT=FABRICANT.NIFFABRICANT
         AND FABRICANT.ESTATFABRICANT=ESTAT.IDESTAT
         AND FABRICANT.IDUBICAFABRICANT=UBICACIO.IDUBICA
         AND LOCALITAT.IDLOCALITAT=UBICACIO.IDLOCALITATUBICACIO
         AND LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
         AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS
         AND UBICACIO.IDVIAUBICACIO=VIA.IDVIA
         AND FABRICANT.IDFABRICANT=FABRICANT.IDFABRICANT
         ORDER BY FABRICANT.NIFFABRICANT;
 BEGIN
 c_procesLog := 'PRC_CONSULTAR_UN_FABRICANT';
 c_dataHoraLog := SYSDATE;
 c_entradaLog := '     CONSULTA DADES UN FABRICANT       ';
 c_sortidalog := 's_rsp';

 --1er. Comprovem que existeixi Fabricant------------------------------
 SELECT  DISTINCT COUNT (FABRICANT.IDFABRICANT) INTO n_registres
 FROM FABRICANT
 WHERE FABRICANT.NIFFABRICANT=FABRICANT.NIFFABRICANT
 AND FABRICANT.IDFABRICANT=FABRICANT.IDFABRICANT;

 If n_registres=0 then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTAR FABRICANT:  ');
```

```
    s_rsp := 'Fabricant no existent a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_fabricant;
END IF;

    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          CONSULTA TIPUS DE LINEAS ACTIVADES          ');
    DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------------');

OPEN C_FABRIANT;
    FETCH C_FABRIANT INTO
    n_idFabricant,
    n_nifFabricant,
    n_dataAltaFabricant,
    n_descEstat,
    n_dataModificacioFabricant,
    n_nomComercialFabricant,
    n_via,n_direccioFabricant,
    n_numeroFabricant,
    n_pisFabricant,
    n_portaFabricant,
    n_codiPostalFabricant,
    n_poblacioFabricant,
    n_provinvciaFabricant,
    n_paisFabricant;
  WHILE C_FABRIANT%FOUND LOOP
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('
   Codi Fabricant:'||n_idFabricant||' '||
  ' NIF: '||n_nifFabricant||
  ' Data Alta: '||n_dataAltaFabricant||' '||
  ' Estat: '||n_descEstat ||
  ' Data ultima modificacio: '||n_dataModificacioFabricant||' '||
  ' Nom Comercial: '||n_nomComercialFabricant);
   s_rsp :='OK  CONSULTAR UN FABRICANT:
    Codi Fabricant:'||n_idFabricant||
  ' NIF: '||n_nifFabricant||
  ' Data Alta: '||n_dataAltaFabricant||
  ' Estat: '||n_descEstat||
  ' Data ultima modificacio: '||n_dataModificacioFabricant||
  ' Nom Comercial: '||n_nomComercialFabricant;
FETCH C_FABRIANT INTO
    n_idFabricant,
    n_nifFabricant,
    n_dataAltaFabricant,
    n_descEstat,
    n_dataModificacioFabricant,
    n_nomComercialFabricant,
    n_via,n_direccioFabricant,
    n_numeroFabricant,
    n_pisFabricant,
```

```
    n_portaFabricant,
    n_codiPostalFabricant,
    n_poblacioFabricant,
    n_provinvciaFabricant,
    n_paisFabricant;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' Total fabricants: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_FABRIANT;
COMMIT;

EXCEPTION
   WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
      ELSE
         -- L'error si ha estat controlat per codi
         s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;

  END PRC_CONSULTAR_FABRICANTS;
END "GESTION_FABRICANT";
```

9.3 TRATAMIENTO CONTRATOS.

En este proceso se inicializa en lecturas a cero, hace un apunte del comptador en el día de alta con la lectura actual inicializada a cero.

DEFINICION	PROCEDIMIENTO
Alta de Contrato	PRC_ALTA_CONTRACTE
Modificar Contrato	PRC_MODIFICAR_CONTRACTE

Baja de un Contrato	PRC_CANVIAR_ALTA_CONTRACTE(
Alta de un Contrato	PRC_CANVIAR_ALTA_CONTRACTE
Consultar un Contrato DNI	PRC_CONSULTA_CONTRA_DNI
Consulta de los Contratos	PRC_CONSULTA_CONTRACTE

```
CREATE OR REPLACE
PACKAGE        "GESTION_CONTRACTE" AS
PROCEDURE PRC_ALTA_CONTRACTE(
            p_dniClientContracte IN CONTRACTE.DNICLIENTECONTRACTE%TYPE,
            p_idComptadorContracte IN CONTRACTE.IDCONTADORCONTRACTE%TYPE,
            p_dataAltaContracte IN CONTRACTE.DATAALTACONTRACTE%TYPE,
            p_potenciaContracte IN CONTRACTE.POTENCIACONTRACTE%TYPE,
            p_observacioContracte IN CONTRACTE.OBSERVACIOCONTRACTE%TYPE,
            s_rsp out NOCOPY VARCHAR);

PROCEDURE PRC_MODIFICAR_CONTRACTE(
            p_idContracte IN CONTRACTE.IDCONTRACTE%TYPE,
            p_idComptadorContracte       IN CONTRACTE.IDCONTADORCONTRACTE%TYPE,
            p_dataModificacioContracte IN CONTRACTE.DATAMODIFICACIOCONTRACTE%TYPE,
            p_potenciaContracte  IN CONTRACTE.POTENCIACONTRACTE%TYPE,
            p_observacioContracte         IN CONTRACTE.OBSERVACIOCONTRACTE%TYPE,
      s_rsp out NOCOPY VARCHAR);

PROCEDURE PRC_CANVIAR_ALTA_CONTRACTE(
            p_idContracte IN CONTRACTE.IDCONTRACTE%TYPE,
            p_dataModificacioContracte IN CONTRACTE.DATAMODIFICACIOCONTRACTE%TYPE,
            s_rsp out NOCOPY VARCHAR);

PROCEDURE PRC_CANVIAR_BAIXA_CONTRACTE(
            p_idContracte IN CONTRACTE.IDCONTRACTE%TYPE,
            p_dataModificacioContracte IN CONTRACTE.DATAMODIFICACIOCONTRACTE%TYPE,
            s_rsp out NOCOPY VARCHAR);

PROCEDURE PRC_CONSULTA_CONTRACTE(
            s_rsp out NOCOPY VARCHAR);

PROCEDURE PRC_CONSULTA_CONTRA_DNI(
            p_dniClientContracte IN CONTRACTE.DNICLIENTECONTRACTE%TYPE,
            s_rsp out NOCOPY VARCHAR);
```

```
END "GESTION_CONTRACTE";
```

9.3.1 PROCEDIMIENTO SPL PACKAGE CONTRATOS.

```
CREATE OR REPLACE PACKAGE BODY        "GESTION_CONTRACTE" AS
 n_registres     NUMBER;
 s_sql         VARCHAR2 (2000);
 n_NUM_ERR          NUMBER(10);

 n_idContracte  NUMBER;
 n_dniClientContracte VARCHAR2(10);
 n_idComptadorContracte  NUMBER;
-- n_fechaEstado_ins     VARCHAR2(11 CHAR);

 n_idComptadorLectura number;
 n_confirmaLectura  number;
 n_dataLectura VARCHAR2(11 CHAR);
 n_idTipoLectura  number;
 n_consumLectura  number;
 n_lecturaActual  number;
 n_lecturaAnterior  number;

 n_estat NUMBER;

 lec_idComptadorLectura number;
 lec_confirmaLectura  number;
 lec_dataLectura VARCHAR2(11 CHAR);
 lec_idTipoLectura  number;
 lec_consumLectura  number;
 lec_lecturaActual  number;
 lec_lecturaAnterior  number;

 n_descEstat ESTAT.DESCRIPCIOESTAT%TYPE;
 n_controlEstat ESTAT.IDESTAT%TYPE;

 n_dniContracte CONTRACTE.DNICLIENTECONTRACTE%TYPE;
 n_observacionsContracte CONTRACTE.OBSERVACIOCONTRACTE%TYPE;
 n_potenciaContracte CONTRACTE.POTENCIACONTRACTE%TYPE;
 n_dataAltaContracte CONTRACTE.DATAALTACONTRACTE%TYPE;
 n_dataModificacioContracte CONTRACTE.DATAMODIFICACIOCONTRACTE%TYPE;

 n_nomClient CLIENT.NOMCLIENT%TYPE;
 n_cognom1Client CLIENT.COGNOM1CLIENT%TYPE;
 n_cognom2Client CLIENT.COGNOM2CLIENT%TYPE;

 c_procesLog                LOG_TFC.procesLog%TYPE;
 c_dataHoraLog     LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog      LOG_TFC.entradaLog%TYPE;
 c_sortidaLog      LOG_TFC.sortidaLog%TYPE;
 s_rsp                      LOG_TFC.RSPLOG%TYPE;
```

```
 sortida VARCHAR2(2000):='';
 e_contracte   EXCEPTION;

-- ALTA DE UN CONTRACTE A UN CLIENT SEGONS DNI
PROCEDURE PRC_ALTA_CONTRACTE(
            p_dniClientContracte IN CONTRACTE.DNICLIENTECONTRACTE%TYPE,
            p_idComptadorContracte IN CONTRACTE.IDCONTADORCONTRACTE%TYPE,
            p_dataAltaContracte IN CONTRACTE.DATAALTACONTRACTE%TYPE,
            p_potenciaContracte IN CONTRACTE.POTENCIACONTRACTE%TYPE,
            p_observacioContracte IN CONTRACTE.OBSERVACIOCONTRACTE%TYPE,
            s_rsp out NOCOPY VARCHAR)
 AS
 BEGIN

   c_procesLog := 'PRC_ALTA_CONTRACTE';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Dni Client: ' ||p_dniClientContracte||
   ', Codi Comptador: ' ||p_idComptadorContracte||
   ', Data Alta Contracte: ' ||p_dataAltaContracte||
   ', Potencia Contractada: ' ||p_potenciaContracte||
   ', Observacions Contracte: ' ||p_observacioContracte;

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('      ASSIGNAR CONTRACTES A CLIENTS     ');
   DBMS_OUTPUT.PUT_LINE('----------------------------------------');
   DBMS_OUTPUT.PUT_LINE('    Dni Client: ' ||p_dniClientContracte||
   ', Codi Comptador: ' ||p_idComptadorContracte||
   ', Data Alta Contracte: ' ||p_dataAltaContracte||
   ', Potencia Contractada: ' ||p_potenciaContracte||
   ', Observacions Contracte: ' ||p_observacioContracte);
    c_sortidalog := 's_rsp';

   n_dataLectura     := SUBSTR (p_dataAltaContracte , 1, 11);
   n_controlEstat:=1;

 -- CONSULTEM EL CLIENT SI EXISTEIX
   SELECT CLIENT.DNICLIENT INTO n_dniClientContracte
   FROM CLIENT
   WHERE CLIENT.DNICLIENT=p_dniClientContracte;

   IF n_dniClientContracte IS NULL THEN
    DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('ASSIGNAR CONTRACTE A CLIENT:   ');
        s_rsp := 'El DNI no existeix, el client no esta  en alta ';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_contracte ;
    END IF;

 --CONSULTEM SI EL COMPTADOR ES DIPONIBLE
  SELECT COMPTADOR.IDCOMPTADOR INTO n_idComptadorContracte
   FROM COMPTADOR
```

```
WHERE COMPTADOR.IDCOMPTADOR=p_idComptadorContracte
AND COMPTADOR.IDESTADOCOMPATOR=n_controlEstat;

 IF p_idComptadorContracte=0 THEN
 DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ASSIGNAR CONTRACTE A CLIENT:   ');
      s_rsp := 'El Comptador no existeix o no esta disponible ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_contracte ;
END IF;

IF p_dataAltaContracte IS NULL THEN
DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ASSIGNAR CONTRACTE A CLIENT:   ');
      s_rsp := 'Falta la Data alta ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_contracte ;
END IF;
 IF (p_potenciaContracte IS NULL OR p_potenciaContracte=0
 OR p_potenciaContracte<0) THEN
DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ASSIGNAR CONTRACTE A CLIENT:   ');
      s_rsp := 'La potencia contractada no es correcte ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_contracte ;
END IF;
-- controlem si exiteix el contracte
SELECT COUNT (*)INTO n_registres
FROM CONTRACTE
WHERE CONTRACTE.DNICLIENTECONTRACTE=n_dniClientContracte
AND CONTRACTE.IDCONTADORCONTRACTE=n_idComptadorContracte;

 IF n_registres= 0 THEN
  n_dataLectura:= SUBSTR (p_dataAltaContracte , 1, 11);
  n_controlEstat:=1;
 lec_idComptadorLectura:=p_idComptadorContracte;
 lec_dataLectura:=n_dataLectura;

 SELECT ESTAT.DESCRIPCIOESTAT INTO n_descEstat
 FROM ESTAT
 WHERE ESTAT.IDESTAT=n_controlEstat;

 INSERT INTO CONTRACTE(DNICLIENTECONTRACTE,
                 IDCONTADORCONTRACTE,
                 DATAALTACONTRACTE,
                 IDESTATCONTRACTE,
                 POTENCIACONTRACTE,
                 OBSERVACIOCONTRACTE)
      VALUES(p_dniClientContracte,
           p_idComptadorContracte,
           p_dataAltaContracte,
           n_controlEstat,
```

```
            p_potenciaContracte,
            p_observacioContracte);

  GESTION_LECTURA.PRC_INICIA_LECTURA(
            lec_idComptadorLectura,
            lec_dataLectura,
            s_rsp);

    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('  Contracte realitzat Dni Client: '||p_dniClientContracte||
    ' Estat : '||n_descEstat||
    ' en la data:' ||lec_dataLectura||
    ' amb el Comptador: '||lec_idComptadorLectura);
    --Gravem en la taula log-----------------------------------------------------
    s_rsp :='Ok Contracte realitzat Dni Client: '||p_dniClientContracte||
    ' Estat : '||n_descEstat||
    ' en la data:' ||lec_dataLectura||
    ' amb el Comptador: '||lec_idComptadorLectura;
    pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('Inicialitzat el comptador amb lectura a zero:
     Contracte realitzat '||p_dniClientContracte||
   ' Estat : '||n_descEstat||
   ' amb el Comptador: '||p_idComptadorContracte||
    'Lectura: inicial: '||lec_idComptadorLectura||
    ' Data inicialitzada: '||lec_dataLectura);
   DBMS_OUTPUT.PUT_LINE('********* FI DE LES OPERACIONS *********');
    --Gravem en la taula log-----------------------------------------------------
    s_rsp :='Ok Inicialitzat el comptador amb lectura a zero:
     Contracte realitzat '||p_dniClientContracte||
   ' Estat : '||n_descEstat||
   ' amb el Comptador: '||p_idComptadorContracte||
    'Lectura: inicial: '||lec_idComptadorLectura||
    ' Data inicialitzada: '||lec_dataLectura;
   pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

    ELSE
    DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('ASSIGNAR CONTRACTE A CLIENT: ');
       s_rsp := 'Contracte de Client duplicat';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_contracte ;
    END IF;
 COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
```

```
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 ROLLBACK;
END PRC_ALTA_CONTRACTE;


PROCEDURE PRC_MODIFICAR_CONTRACTE(
            p_idContracte IN CONTRACTE.IDCONTRACTE%TYPE,
            p_idComptadorContracte       IN CONTRACTE.IDCONTADORCONTRACTE%TYPE,
            p_dataModificacioContracte IN CONTRACTE.DATAMODIFICACIOCONTRACTE%TYPE,
            p_potenciaContracte  IN CONTRACTE.POTENCIACONTRACTE%TYPE,
            p_observacioContracte          IN CONTRACTE.OBSERVACIOCONTRACTE%TYPE,
      s_rsp out NOCOPY VARCHAR)
AS
BEGIN
   c_procesLog := 'PRC_MODIFICAR_CONTRACTE';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Numero Contracte: '||p_idContracte||
   ', Codi Comptador: ' ||p_idComptadorContracte||
   ', Data modificacio Contracte: ' ||p_dataModificacioContracte||
   ', Potencia Contractada: ' ||p_potenciaContracte||
   ', Observacions Contracte: ' ||p_observacioContracte;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('    MODIFICAR UN CONTRACTE     ');
   DBMS_OUTPUT.PUT_LINE('------------------------------ ');
   DBMS_OUTPUT.PUT_LINE('    Numero Contracte: '||p_idContracte||
   --', Dni Client: ' ||p_dniClientContracte||
   ', Codi Comptador: ' ||p_idComptadorContracte||
   ', Data Alta Contracte: ' ||p_dataModificacioContracte||
   ', Potencia Contractada: ' ||p_potenciaContracte||
   ', Observacions Contracte: ' ||p_observacioContracte);
   c_sortidalog := 's_rsp';

 SELECT CONTRACTE.IDESTATCONTRACTE INTO n_controlEstat
 FROM CONTRACTE
 WHERE CONTRACTE.IDCONTRACTE=p_idContracte;

 IF p_idContracte IS NULL THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('MODIFICAR UN CONTRACTE:  ');
 s_rsp := 'Falta entrar el num.Contracte  ';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_contracte ;
 END IF;

 -- CONSULTEM EL CLIENT SI EXISTEIX
   SELECT CLIENT.DNICLIENT INTO n_dniClientContracte
   FROM CLIENT, CONTRACTE
```

```
  WHERE CONTRACTE.IDCONTRACTE=p_idContracte
  AND CLIENT.ESTATCLIENT=n_controlEstat;

  IF n_dniClientContracte IS NULL THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR UN CONTRACTE:  ');
   s_rsp := 'El DNI no existeix no té cap contracte assignat ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_contracte ;
   END IF;

 --CONSULTEM SI EL COMPTADOR ES DIPONIBLE
  SELECT DISTINCT COUNT (COMPTADOR.IDCOMPTADOR) INTO n_idComptadorContracte
   FROM COMPTADOR
   WHERE COMPTADOR.IDCOMPTADOR=p_idContracte
   AND COMPTADOR.IDESTADOCOMPATOR=n_controlEstat;

   IF p_idComptadorContracte=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR UN CONTRACTE:    ');
    s_rsp := 'El Comptador no existeix o no esta disponible ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_contracte ;
   END IF;

   IF p_dataModificacioContracte IS NULL THEN
   DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('MODIFICAR UN CONTRACTE:    ');
         s_rsp := 'Falta la Data en que modifica les dades  ';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_contracte ;
   END IF;
    IF (p_potenciaContracte IS NULL OR p_potenciaContracte=0
    OR p_potenciaContracte<0) THEN
   DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UN CONTRACTE:     ');
    s_rsp := 'La potencia contractada no es correcte ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_contracte ;
   END IF;
   -- controlem si exiteix el contracte
   SELECT COUNT (*)INTO n_registres
   FROM CONTRACTE
   WHERE CONTRACTE.DNICLIENTECONTRACTE=n_dniClientContracte
   AND CONTRACTE.IDCONTRACTE=p_idContracte;

  IF n_registres=0 THEN
     DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UN CONTRACTE:     ');
    s_rsp := 'No existeix Id del contracte ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_contracte ;
```

```
    END IF;
   --recuperem el contracte
   SELECT CONTRACTE.IDCONTRACTE INTO n_idContracte
         FROM CONTRACTE
         WHERE CONTRACTE.IDCONTRACTE=p_idContracte;

   SELECT ESTAT.DESCRIPCIOESTAT INTO n_descEstat
   FROM ESTAT
   WHERE ESTAT.IDESTAT=n_controlEstat;

    s_sql := 'UPDATE CONTRACTE SET ';
    IF p_idComptadorContracte IS NOT NULL THEN
   -- Modifiquem  id Comptador Contracte
     s_sql := s_sql || 'IDCONTADORCONTRACTE=''' ||p_idComptadorContracte|| ''',';
     END IF;
    IF p_dataModificacioContracte IS NOT NULL THEN
    -- Modifiquem data Modificacio Contracte
      s_sql := s_sql || 'DATAMODIFICACIOCONTRACTE=''' ||p_dataModificacioContracte|| ''',';
      END IF;
    IF n_controlEstat IS NOT NULL THEN
      -- Modifiquem id Estat Contracte
      s_sql := s_sql || 'IDESTATCONTRACTE=''' ||n_controlEstat|| ''',';
     END IF;
    IF p_potenciaContracte IS NOT NULL THEN
  -- Modifiquem potencia Contracte
    s_sql := s_sql || 'POTENCIACONTRACTE=''' ||p_potenciaContracte|| ''',';
    END IF;
    IF p_observacioContracte IS NOT NULL THEN
  -- Modifiquem Observacions Tipo Linea
    s_sql := s_sql || 'OBSERVACIOCONTRACTE=''' ||p_observacioContracte|| ''',';
    END IF;
 -- Eliminem la coma final de la sentencia SQL-------------------------
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÆ'Ã‚Â³ del WHERE-----------------------
   s_sql := s_sql || ' WHERE IDCONTRACTE=''' ||p_idContracte|| '''';
EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0 THEN
 --L'error serÃƒÂ  que Tipo Linea ja el tenim en la taula i no admet duplicats--
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR_CONTRACTE:  ');
  s_rsp := 'Actualitzacio no realitzada';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_contracte ;
  ELSE
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' La modificacio del Contracte numero: '||p_idContracte||
  ' en la data: '||p_dataModificacioContracte||
  ' en estat: '||n_descEstat||
  ' amb el comptador: '||p_idComptadorContracte||
  ' amb la Potecnia contratade de: '||p_potenciaContracte||
  ' Onservacions: '||p_observacioContracte|| ' modificacio realitzada');
  DBMS_OUTPUT.PUT_LINE('********* FI DE LES OPERACIONS *********');
```

```
    s_rsp := 'OK MODIFICAR CONTRACTE: El Contracte numero: '||p_idContracte||
  ' en la data: '||p_dataModificacioContracte||
  ' en estat: '||n_descEstat||
  ' amb el comptador: '||p_idComptadorContracte||
  ' amb la Potecnia contratade de: '||p_potenciaContracte||
  ' Onservacions: '||p_observacioContracte|| ' modificacio realitzada';
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;
COMMIT;
  EXCEPTION
    WHEN DUP_VAL_ON_INDEX THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR CONTRACTE:  ');
      s_rsp := 'Error: CONTRACTE duplicat. No es poden fer les modificacions';
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        ----------------------------------------
        s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
        -- L'error si ha estat controlat per codi
        ----------------------------------------
        s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('  ');
      DBMS_OUTPUT.put_line (s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;

END PRC_MODIFICAR_CONTRACTE;

PROCEDURE PRC_CANVIAR_ALTA_CONTRACTE(
              p_idContracte IN CONTRACTE.IDCONTRACTE%TYPE,
              p_dataModificacioContracte IN CONTRACTE.DATAMODIFICACIOCONTRACTE%TYPE,
              s_rsp out NOCOPY VARCHAR)
AS
BEGIN
    c_procesLog := 'PRC_CANVIAR_ALTA_CONTRACTE';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :=' Numero Contracte: '||p_idContracte||
    ', Data modificacio Contracte: ' ||p_dataModificacioContracte;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('       CANVIAR ALTA A UN CONTRACTE      ');
    DBMS_OUTPUT.PUT_LINE('---------------------------------------- ');
    DBMS_OUTPUT.PUT_LINE('   Numero Contracte: '||p_idContracte||
    ', Data modificacio Contracte: ' ||p_dataModificacioContracte);
    c_sortidalog := 's_rsp';
```

```
IF p_idContracte IS NULL THEN
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE('CANVIAR ALTA A UN CONTRACTE:  ');
s_rsp := 'Falta el numero de contracte. ';
DBMS_OUTPUT.PUT_LINE(s_rsp);
RAISE e_contracte ;
END IF;
IF p_dataModificacioContracte IS NULL THEN
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE('CANVIAR ALTA A UN CONTRACTE:  ');
s_rsp := 'Falta la data de la modificacio. ';
DBMS_OUTPUT.PUT_LINE(s_rsp);
RAISE e_contracte ;
END IF;

SELECT COUNT (p_idContracte) INTO n_idContracte
FROM CONTRACTE
WHERE CONTRACTE.IDCONTRACTE=p_idContracte;

IF n_idContracte=0 THEN
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE('CANVIAR ALTA A UN CONTRACTE:  ');
s_rsp := 'El contracte no exiteix. ';
DBMS_OUTPUT.PUT_LINE(s_rsp);
RAISE e_contracte ;
END IF;
n_controlEstat:=1;
SELECT CONTRACTE.IDESTATCONTRACTE INTO n_estat
FROM CONTRACTE
WHERE CONTRACTE.IDESTATCONTRACTE=n_controlEstat
AND CONTRACTE.IDCONTRACTE=p_idContracte;

IF n_estat=n_controlEstat THEN
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE('CANVIAR ALTA A UN CONTRACTE:  ');
s_rsp := 'El contracte ja esta en ALTA. ';
DBMS_OUTPUT.PUT_LINE(s_rsp);
RAISE e_contracte ;
END IF;

SELECT CONTRACTE.IDCONTRACTE INTO n_idContracte
FROM CONTRACTE
WHERE CONTRACTE.IDCONTRACTE=p_idContracte
AND CONTRACTE.IDESTATCONTRACTE=n_controlEstat;

s_sql := 'UPDATE CONTRACTE SET ';
   IF p_dataModificacioContracte IS NOT NULL THEN
   -- Modifiquem data Modificacio Contracte
     s_sql := s_sql || 'DATAMODIFICACIOCONTRACTE="' ||p_dataModificacioContracte|| '",';
     END IF;
   IF n_controlEstat IS NOT NULL THEN
     -- Modifiquem id Estat Contracte
```

```
      s_sql := s_sql || 'IDESTATCONTRACTE='" ||n_controlEstat|| "',';
     END IF;
 -- Eliminem la coma final de la sentencia SQL-------------------------
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÆ'Ã‚Â³ del WHERE----------------------
   s_sql := s_sql || ' WHERE IDCONTRACTE='" ||p_idContracte|| "';

EXECUTE IMMEDIATE s_sql;
 IF SQL%ROWCOUNT = 0 THEN
 --L'error serÃƒÂ  que Tipo Linea ja el tenim en la taula i no admet duplicats--
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('CANVIAR ALTA A UN CONTRACTE: ');
 s_rsp := 'Actualitzacio no realitzada';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_contracte ;
 ELSE
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(' Alta del Contracte numero: '||p_idContracte||
 ' en la data: '||p_dataModificacioContracte||
 ' en estat: '||n_descEstat||
 ' modificacio realitzada');
 DBMS_OUTPUT.PUT_LINE('********* FI DE LES OPERACIONS *********');
   s_rsp := 'OK Alta del Contracte numero: '||p_idContracte||
 ' en la data: '||p_dataModificacioContracte||
 ' en estat: '||n_descEstat||
 ' modificacio realitzada';
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;
 EXCEPTION
   WHEN DUP_VAL_ON_INDEX THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('CANVIAR ALTA A UN CONTRACTE:  ');
     s_rsp := 'Error: CONTRACTE duplicat. No es poden fer les modificacions';
     DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;
  WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi
       ----------------------------------------
       s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi
       ----------------------------------------
       s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('  ');
     DBMS_OUTPUT.put_line (s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;
```

```
END PRC_CANVIAR_ALTA_CONTRACTE;

PROCEDURE PRC_CANVIAR_BAIXA_CONTRACTE(
        p_idContracte IN CONTRACTE.IDCONTRACTE%TYPE,
        p_dataModificacioContracte IN CONTRACTE.DATAMODIFICACIOCONTRACTE%TYPE,
        s_rsp out NOCOPY VARCHAR)
AS
BEGIN
  c_procesLog := 'PRC_CANVIAR_BAIXA_CONTRACTE';
  c_dataHoraLog := SYSDATE;
  c_entradaLog :=' Numero Contracte: '||p_idContracte||
  ', Data modificacio Contracte: ' ||p_dataModificacioContracte;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('      BAIXA DE CONTRACTE          ');
  DBMS_OUTPUT.PUT_LINE('---------------------------------------- ');
  DBMS_OUTPUT.PUT_LINE('   Numero Contracte: '||p_idContracte||
  ', Data modificacio Contracte: ' ||p_dataModificacioContracte);
  c_sortidalog := 's_rsp';

  IF p_idContracte IS NULL THEN
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('BAIXA DE CONTRACTE:  ');
  s_rsp := 'Falta el numero de contracte. ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_contracte ;
  END IF;
  IF p_dataModificacioContracte IS NULL THEN
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('BAIXA DE CONTRACTE:  ');
  s_rsp := 'Falta la data de la modificacio. ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_contracte ;
  END IF;

  SELECT COUNT (p_idContracte) INTO n_idContracte
  FROM CONTRACTE
  WHERE CONTRACTE.IDCONTRACTE=p_idContracte;

  IF n_idContracte=0 THEN
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('BAIXA DE CONTRACTE:  ');
  s_rsp := 'El contracte no exiteix. ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_contracte ;
  END IF;
  n_controlEstat:=2;
  SELECT CONTRACTE.IDESTATCONTRACTE INTO n_estat
  FROM CONTRACTE
  WHERE CONTRACTE.IDESTATCONTRACTE=n_controlEstat
  AND CONTRACTE.IDCONTRACTE=p_idContracte;
```

```
   IF n_estat=n_controlEstat THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('BAIXA DE CONTRACTE:  ');
   s_rsp := 'El contracte ja esta en Baixa. ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_contracte ;
   END IF;

   SELECT CONTRACTE.IDCONTRACTE INTO n_idContracte
   FROM CONTRACTE
   WHERE CONTRACTE.IDCONTRACTE=p_idContracte
   AND CONTRACTE.IDESTATCONTRACTE=n_controlEstat;

 s_sql := 'UPDATE CONTRACTE SET ';
     IF p_dataModificacioContracte IS NOT NULL THEN
     -- Modifiquem data Modificacio Contracte
       s_sql := s_sql || 'DATAMODIFICACIOCONTRACTE='' ||p_dataModificacioContracte|| '',';
       END IF;
     IF n_controlEstat IS NOT NULL THEN
       -- Modifiquem id Estat Contracte
       s_sql := s_sql || 'IDESTATCONTRACTE='' ||n_controlEstat|| '',';
      END IF;
 -- Eliminem la coma final de la sentencia SQL-------------------------
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÆ'Ã‚Â³ del WHERE-----------------------
   s_sql := s_sql || ' WHERE IDCONTRACTE='' ||p_idContracte|| '';

EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0 THEN
 --L'error serÃƒÂ  que Tipo Linea ja el tenim en la taula i no admet duplicats--
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('BAIXA DE CONTRACTE: ');
  s_rsp := 'Actualitzacio no realitzada';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_contracte ;
  ELSE
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' Baixa del Contracte numero: '||p_idContracte||
  ' en la data: '||p_dataModificacioContracte||
  ' en estat: '||n_descEstat||
  ' modificacio realitzada');
  DBMS_OUTPUT.PUT_LINE('********* FI DE LES OPERACIONS *********');
    s_rsp := 'OK Baixa del Contracte numero: '||p_idContracte||
  ' en la data: '||p_dataModificacioContracte||
  ' en estat: '||n_descEstat||
  ' modificacio realitzada';
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;
COMMIT;
  EXCEPTION
    WHEN DUP_VAL_ON_INDEX THEN
```

```
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('BAIXA DE CONTRACTE:  ');
    s_rsp := 'Error: CONTRACTE duplicat. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_CANVIAR_BAIXA_CONTRACTE;

PROCEDURE PRC_CONSULTA_CONTRACTE(
            s_rsp out NOCOPY VARCHAR)
AS

CURSOR C_CONTRACTE IS
SELECT
CONTRACTE.IDCONTRACTE,
CONTRACTE.DATAALTACONTRACTE,
ESTAT.DESCRIPCIOESTAT,
CONTRACTE.DATAMODIFICACIOCONTRACTE,
CONTRACTE.DNICLIENTECONTRACTE,
CLIENT.NOMCLIENT,
CLIENT.COGNOM1CLIENT,
CLIENT.COGNOM2CLIENT,
CONTRACTE.IDCONTADORCONTRACTE,
CONTRACTE.POTENCIACONTRACTE,
CONTRACTE.OBSERVACIOCONTRACTE
FROM ESTAT, CLIENT,CONTRACTE
WHERE CONTRACTE.IDCONTRACTE=CONTRACTE.IDCONTRACTE
AND CONTRACTE.DNICLIENTECONTRACTE=CLIENT.DNICLIENT
AND CONTRACTE.IDESTATCONTRACTE=ESTAT.IDESTAT;

BEGIN
   c_procesLog := 'PRC_CONSULTAR_CONTRACTE';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='CONSULTAR CONTRACTES';
   c_sortidalog := 's_rsp';
```

```
SELECT DISTINCT COUNT (*) INTO n_registres
FROM CONTRACTE
WHERE CONTRACTE.IDCONTRACTE=CONTRACTE.IDCONTRACTE;

IF n_registres=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTAR CONTRACTES:  ');
   s_rsp := 'No hi han contractes. ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_contracte ;
  ELSE

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('      CONSULTA CONTRACTES        ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------------');

OPEN C_CONTRACTE;
FETCH C_CONTRACTE INTO
n_idContracte,
n_dataAltaContracte,
n_descEstat,
n_dataModificacioContracte,
n_dniContracte,
n_nomClient,
n_cognom1Client,
n_cognom2Client,
n_idComptadorContracte,
n_potenciaContracte,
n_observacionsContracte;
WHILE C_CONTRACTE%FOUND LOOP
DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(
'   Numero contracte:  '
||n_idContracte||
' Data alta Contracte:  '
||n_dataAltaContracte||
' Estat:  '
||n_descEstat||
' Data Modificacio: '
||n_dataModificacioContracte||
' Dni Client: '
||n_dniContracte||
' Nom: '
||n_nomClient||
' Cognom1: '
||n_cognom1Client||
' Cognom2: '
||n_cognom2Client||
' Codi Comptador: '
||n_idComptadorContracte||
' Potencia Contractada: '
||n_potenciaContracte||
```

```
' Observacions: '
||n_observacionsContracte);
 s_rsp := 'OK  CONSULTA CONTRACTES
   Numero contracte:  '
||n_idContracte||
--' Data alta Contracte:  '
--||n_dataAltaContracte||
--' Estat:  '
--||n_descEstat||
/****************************
' Data Modificacio: '
||n_dataModificacioContracte||
' Dni Client: '
||n_dniContracte||
' Nom: '
||n_nomClient||
' Cognom1: '
||n_cognom1Client||
' Cognom2: '
||n_cognom2Client||
' Codi Comptador: '
||n_idComptadorContracte||
' Potencia Contractada: '
||n_potenciaContracte||
*****************************/
' Observacions: '
||n_observacionsContracte;
FETCH C_CONTRACTE INTO n_idContracte,
n_dataAltaContracte,
n_descEstat,
n_dataModificacioContracte,
n_dniContracte,
n_nomClient,
n_cognom1Client,
n_cognom2Client,
n_idComptadorContracte,
n_potenciaContracte,
n_observacionsContracte;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total contractes: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');

CLOSE C_CONTRACTE;
 END IF;
COMMIT;
 EXCEPTION

  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
```

```
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_CONSULTA_CONTRACTE;

PROCEDURE PRC_CONSULTA_CONTRACTE_DNI(
            p_dniClientContracte IN CONTRACTE.DNICLIENTECONTRACTE%TYPE,
            s_rsp out NOCOPY VARCHAR)
AS

CURSOR C_CONTRACTE_DNI IS
SELECT CONTRACTE.IDCONTRACTE,
CONTRACTE.DATAALTACONTRACTE,
ESTAT.DESCRIPCIOESTAT,
CONTRACTE.DATAMODIFICACIOCONTRACTE,
CONTRACTE.DNICLIENTECONTRACTE,
CLIENT.NOMCLIENT,
CLIENT.COGNOM1CLIENT,
CLIENT.COGNOM2CLIENT,
CONTRACTE.IDCONTADORCONTRACTE,
CONTRACTE.POTENCIACONTRACTE,
CONTRACTE.OBSERVACIOCONTRACTE
FROM ESTAT, CLIENT,CONTRACTE
WHERE CONTRACTE.IDCONTRACTE=CONTRACTE.IDCONTRACTE
AND CONTRACTE.DNICLIENTECONTRACTE=CLIENT.DNICLIENT
AND CONTRACTE.IDESTATCONTRACTE=ESTAT.IDESTAT
AND CONTRACTE.DNICLIENTECONTRACTE=p_dniClientContracte;
BEGIN
   c_procesLog := 'PRC_CONSULTA_CONTRACTE_DNI';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' DNI Client: '||p_dniClientContracte;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('      CONSULTAR CONTRACTE UN CLIENT           ');
   DBMS_OUTPUT.PUT_LINE('---------------------------------------------- ');
   DBMS_OUTPUT.PUT_LINE('   DNI Client: '||p_dniClientContracte);
   c_sortidalog := 's_rsp';

 IF p_dniClientContracte IS NULL THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' CONSULTAR CONTRACTE UN CLIENT:  ');
   s_rsp := 'Falta DNI del Client. ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_contracte ;
```

```
   END IF;
SELECT COUNT(*) INTO n_registres
FROM CONTRACTE
WHERE CONTRACTE.DNICLIENTECONTRACTE=p_dniClientContracte;

IF n_registres=0 THEN
 DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' CONSULTAR CONTRACTE UN CLIENT:  ');
   s_rsp := 'No hi han contractes del client aquest. ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_contracte ;
   END IF;

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('       CONSULTA CONTRACTES UN CLIENT          ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------');

OPEN C_CONTRACTE_DNI;
FETCH C_CONTRACTE_DNI INTO n_idContracte,
n_dataAltaContracte,
n_descEstat,
n_dataModificacioContracte,
n_dniContracte,
n_nomClient,
n_cognom1Client,
n_cognom2Client,
n_idComptadorContracte,
n_potenciaContracte,
n_observacionsContracte;
WHILE C_CONTRACTE_DNI%FOUND LOOP
DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(
 '   Numero contracte:  '
 ||n_idContracte||
 ' Data alta Contracte:  '
 ||n_dataAltaContracte||
 ' Estat:  '
 ||n_descEstat||
 ' Data Modificacio: '
 ||n_dataModificacioContracte||
 ' Dni Client: '
 ||n_dniContracte||
 ' Nom: '
 ||n_nomClient||
 ' Cognom1: '
 ||n_cognom1Client||
 ' Cognom2: '
 ||n_cognom2Client||
 ' Codi Comptador: '
 ||n_idComptadorContracte||
 ' Potencia Contractada: '
 ||n_potenciaContracte||
```

```
 ' Observacions: '
 ||n_observacionsContracte);
  s_rsp := 'OK  CONSULTA CONTRACTES UN CLIENT
   Numero contracte:  '
 ||n_idContracte||
 ' Data alta Contracte:  '
 ||n_dataAltaContracte||
 ' Estat:  '
 ||n_descEstat||
 ' Data Modificacio: '
 ||n_dataModificacioContracte||
 ' Dni Client: '
 ||n_dniContracte;
FETCH C_CONTRACTE_DNI INTO n_idContracte,
n_dataAltaContracte,
n_descEstat,
n_dataModificacioContracte,
n_dniContracte,
n_nomClient,
n_cognom1Client,
n_cognom2Client,
n_idComptadorContracte,
n_potenciaContracte,
n_observacionsContracte;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total contractes: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');

CLOSE C_CONTRACTE_DNI;
COMMIT;
 EXCEPTION

   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi----------------------------------------
       s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi----------------------------------------
       s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line (s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;

END PRC_CONSULTA_CONTRACTE_DNI;

END "GESTION_CONTRACTE";
```

10 APENDICE 7 MODULO MANTENIMIENTO.

Este modulo hace el tratamiento de todas las entidades necesarias para que los demás módulos puedan efectuar sus funcionalidades asignadas.

```
PACKAGE_GESTIO_ESTAT.sql

PACKAGE_GESTIO_LOCALITAT.sql

PACKAGE_GESTIO_PAIS.sql

PACKAGE_GESTIO_PERSONA.sql

PACKAGE_GESTIO_PROVINCIA.sql

PACKAGE_GESTIO_UBICACIO.sql

PACKAGE_GESTIO_VIA.sql

GESTIO_ESTAT.sql

GESTIO_LOCALITAT.sql

GESTIO_PAIS.sql

GESTIO_PERSONA.sql

GESTIO_PROVINCIA.sql

GESTIO_UBICACIO.sql

GESTIO_VIA.sql
```

10.1 TRATAMIENTO DE ESTADOS O SITUACIONES.

DEFINICION	PROCEDIMIENTO
Alta de Estados	PRC_ALTA_ESTAT
Modificar Estados	PRC_MODIFICAR_ESTAT
Consulta de Estados	PRC_CONSULTA_CONTRACTE

```
create or replace PACKAGE         "GESTION_ESTAT" AS
PROCEDURE PRC_ALTA_ESTAT(
p_descripcioEstat in ESTAT.DESCRIPCIOESTAT%TYPE,
s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_MODIFICAR_ESTAT(
p_descripcioEstat in ESTAT.DESCRIPCIOESTAT%TYPE,
p_idEstat in ESTAT.IDESTAT%TYPE,
s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_CONSULTA_ESTAT(
   s_rsp out NOCOPY VARCHAR2);
END "GESTION_ESTAT";
```

10.1.1 PROCEDIMIENTO SPL PACKAGE ESTADOS.

```
CREATE OR REPLACE
PACKAGE BODY          "GESTION_ESTAT" AS
  c_procesLog           LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp               LOG_TFC.rspLog%TYPE;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);
    n_descripcioEstat ESTAT.DESCRIPCIOESTAT%TYPE;
  n_idEstat   ESTAT.IDESTAT%TYPE;
  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_estat     EXCEPTION;
-------------------------------------------------------------------
-- create procedure for table "LOG_TFC i ESTAT"
   procedure PRC_ALTA_ESTAT(
    p_descripcioEstat in ESTAT.DESCRIPCIOESTAT%TYPE,
    s_rsp out NOCOPY VARCHAR2)
   AS

   BEGIN
    c_procesLog := 'PRC_ALTA_ESTAT';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='Decripcio Estat: ' || p_descripcioEstat;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.PUT_LINE('ALTA DE ESTAT O SITUACIO');
     DBMS_OUTPUT.PUT_LINE('_____________________');
        DBMS_OUTPUT.PUT_LINE('Descripcio: '|| p_descripcioEstat);
    c_sortidalog := 's_rsp';
    If (p_descripcioEstat IS NULL) then
```

```
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('ALTA_ESTAT: ');
         s_rsp := 'Falta especificar estat';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_estat;
    END IF;
     -- Comprovem si existeix el estat  donar d'alta
       SELECT COUNT (*)INTO n_registres
         FROM ESTAT
         WHERE descripcioEstat = p_descripcioEstat;
    If n_registres=0 then
       -- El donem d'alta a la estat
         INSERT INTO ESTAT(descripcioEstat)
         VALUES(p_descripcioEstat);
         --Gravem en la taula log----------------------------------------------------
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('Insertat estat: '||p_descripcioEstat);
            DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
         s_rsp :='Ok: Insertat estat: '||p_descripcioEstat;
         pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        ELSE
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('ALTA_ESTAT:  ');
        s_rsp := 'DESCRICIO Estat  duplicat';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_estat;
    END IF;
  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  end PRC_ALTA_ESTAT;

 PROCEDURE PRC_MODIFICAR_ESTAT(
/*****************************************************************
  NOM:        PRC_MODIFICAR_PAIS
  DESCRIPCIÃ“:
            Procediment encarregat de modificar les dades d'un pais.
            Les dades es modificaran en la taula persona i el identificador de
            Via apuntarÃ  a la clau primaria del contacte de la taula Pais.
```

```
**********************************************************************/
    p_descripcioEstat in ESTAT.DESCRIPCIOESTAT%TYPE,
    p_idEstat in ESTAT.IDESTAT%TYPE,
     s_rsp out NOCOPY VARCHAR2)
AS

BEGIN
 c_procesLog := 'PRC_MODIFICAR_ESTAT';
 c_dataHoraLog := SYSDATE;
 c_entradaLog :='Decripcio Estat: ' || p_descripcioEstat;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
 DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT');
  DBMS_OUTPUT.PUT_LINE('_____________________');
 c_sortidalog := 's_rsp';
 -- ComprovaciÃ³ del identificador Estat-----

  If  (p_descripcioEstat IS NULL) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR_ESTAT: ');
           s_rsp := 'Falta especificar el Estat';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_estat;
    ELSE

      SELECT COUNT (*)INTO n_registres
      FROM ESTAT
      WHERE DESCRIPCIOESTAT= p_descripcioEstat;

     If n_registres=0 then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR_ESTAT: ');
        s_rsp := 'ESTAT no existeix a la BBDD';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
         RAISE e_estat;
      END IF;

    -- recuperem el codi ESTAT
SELECT ESTAT.IDESTAT, ESTAT.DESCRIPCIOESTAT INTO n_idEstat,n_descripcioEstat
FROM ESTAT
WHERE ESTAT.IDESTAT=ESTAT.IDESTAT;

   --s_sql :='UPDATE ESTAT SET';

    s_sql := 'UPDATE ESTAT SET ';

       -- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
       IF p_descripcioEstat IS NOT NULL THEN
            -- Modifiquem el descripcio ESTAT
            s_sql := s_sql || 'DESCRIPCIOESTAT=''' || p_descripcioEstat || ''',';
            --s_sql := s_sql || ' Codi Estat=''' || p_idEstat || ''',';
        END IF;
```

```
        -- Eliminem la coma final de la sentencia SQL
        s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
        -- Afegim a la sentencia la condiciÃ³ del WHERE
        s_sql := s_sql || ' WHERE idEstat=''' || n_idEstat || '''';


        EXECUTE IMMEDIATE s_sql;

        IF SQL%ROWCOUNT = 0
        THEN
          --L'error serÃ  que ESTAT ja el tenim en la taula i no admet duplicats--
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICAR_ESTAT:  ');
          s_rsp := 'ActualitzaciÃ³ no realitzada';
          RAISE e_estat;
        ELSE
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('Modificacio feta de ESTAT: codi: '||n_idEstat||
         ' Descripcio: '||p_descripcioEstat );
         DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
          s_rsp := 'OK Modificacio feta de ESTAT: codi: '||n_idEstat||
         ' Descripcio: '||p_descripcioEstat ;
         pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
        END IF;
    END IF;
  COMMIT;
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    s_rsp := 'Error: Estat duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_MODIFICAR_ESTAT;
```

```
PROCEDURE PRC_CONSULTA_ESTAT(
   s_rsp out NOCOPY VARCHAR2)
AS
CURSOR C_ESTAT IS
SELECT ESTAT.IDESTAT, ESTAT.DESCRIPCIOESTAT
FROM ESTAT
WHERE ESTAT.IDESTAT=ESTAT.IDESTAT;

BEGIN
  c_procesLog := 'PRC_CONSULTA_ESTAT';
   c_dataHoraLog := SYSDATE;
    c_entradaLog :='CONSULTA ESTAT O SITUACIO';
   c_sortidalog := 's_rsp';

   SELECT COUNT(ESTAT.IDESTAT) INTO n_registres
   FROM ESTAT
   WHERE ESTAT.IDESTAT=ESTAT.IDESTAT;

   IF n_registres=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTA ESTAT: ');
   s_rsp := 'No hi han ESTATS donades d´alta ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_estat;
   END IF;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('          CONSULTA DE ESTATS          ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------');
   OPEN C_ESTAT;
          FETCH C_ESTAT INTO n_idEstat,n_descripcioEstat;
 WHILE C_ESTAT%FOUND LOOP
  DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(
'   Codi Estat '
|| n_idEstat
||' Descripcio '
|| n_descripcioEstat);
 s_rsp := 'OK  CONSULTA ESTATS
 Codi Estat:  '
|| n_idEstat
||' Descripcio: '
||n_descripcioEstat;
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  FETCH C_ESTAT INTO n_idEstat,n_descripcioEstat;
  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total estats: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_ESTAT;
COMMIT;
 EXCEPTION
```

```
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_CONSULTA_ESTAT;
END GESTION_ESTAT;
```

10.2 TRATAMIENTO DE VIAS PÚBLICAS.

DEFINICION	PROCEDIMIENTO
Alta de Vías Públicas	PRC_ALTA_VIA
Modificar Vías Públicas	PRC_MODIFICAR_VIA
Consulta de Vías Públicas	PRC_CONSULTA_VIA

```
create or replace PACKAGE        "GESTION_VIA" AS
procedure PRC_ALTA_VIA(
      p_descripcioVia in VIA.DESCRIPCIOVIA%TYPE,

      s_rsp out NOCOPY VARCHAR2);
procedure PRC_MODIFICAR_VIA(
      p_idVia IN VIA.idVia%TYPE,
      p_descripcioVia IN VIA.DESCRIPCIOVIA%TYPE,
      s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_CONSULTA_VIA(
   s_rsp out NOCOPY VARCHAR2
   );

end GESTION_VIA;
```

10.2.1 PROCEDIMIENTO SPL PACKAGE VIAS PUBLICAS.

```
CREATE OR REPLACE
PACKAGE BODY         "GESTION_VIA" AS
  c_procesLog          LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp            LOG_TFC.rspLog%TYPE;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);
  n_descripcioVia VIA.descripcioVia%TYPE;
  n_idVia VIA.idVia%TYPE;
  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_via     EXCEPTION;
-------------------------------------------------------------
-- create procedure for table "LOG_TFC i VIA"


/*****************************************************************************
Autor: Eduard Monzonis Hierro                          UOC
18/05/2012                                  TFC: CONTROL ENERGIA.
  NOM:        PRC_ALTA_VIA
  DESCRIPCIÃ“:
            Procediment encarregat donar alta d'un via pÃºblica.

*****************************************************************************/
  procedure PRC_ALTA_VIA(
    p_descripcioVia in VIA.descripcioVia%TYPE,
   s_rsp out NOCOPY VARCHAR2)
  AS

  BEGIN
   c_procesLog := 'PRC_ALTA_VIA';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Decripcio Via: ' || p_descripcioVia;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('ALTA DE LA VIA PUBLICA');
   DBMS_OUTPUT.PUT_LINE('_____________________');
   DBMS_OUTPUT.PUT_LINE('ALTA VIA: '|| p_descripcioVia);
   c_sortidalog := 's_rsp';
   If (p_descripcioVia IS NULL) then
        DBMS_OUTPUT.ENABLE;
           DBMS_OUTPUT.PUT_LINE('ALTA_VIA:  ');
        s_rsp := 'Falta especificar el la Via PÃºblica';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_via;
    END IF;
     -- Comprovem si existeix el VIA a  donar d'alta
```

```
        SELECT COUNT (*)INTO n_registres
          FROM VIA
          WHERE descripcioVia = p_descripcioVia;
      If n_registres=0 then
        -- El donem d'alta a la Via Publica
         -- Insertem en la taula Persona----------------------------------------------
          INSERT INTO VIA(descripcioVia)
          VALUES(p_descripcioVia);
          --Gravem en la taula log-----------------------------------------------------
         DBMS_OUTPUT.ENABLE;
           DBMS_OUTPUT.PUT_LINE('Insertada la Via Pública: '|| p_descripcioVia);
           DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
          s_rsp :='Ok: En la Taula VIA Insertada la Via Pública:  '|| p_descripcioVia;
          pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
          DBMS_OUTPUT.PUT_LINE(s_rsp);
        ELSE
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('ALTA_VIA: ');
        s_rsp := 'DESCRICIO VIA  duplicat';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_via;
      END IF;
    COMMIT;
  EXCEPTION
    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
      ELSE
        -- L'error si ha estat controlat per codi
        s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;

    end PRC_ALTA_VIA;

   PROCEDURE PRC_MODIFICAR_VIA(
  /*****************************************************************************
  Autor: Eduard Monzonis Hierro                    UOC
  18/05/2012                          TFC: CONTROL ENERGIA.
    NOM:       PRC_MODIFICAR_VIA
    DESCRIPCIÃ“:
              Procediment encarregat de modificar les  vias pÃºblicas.
              Les dades es modificaran en la taula persona i el identificador de
              Via apuntarÃ  a la clau primaria del contacte de la taula Via.
  *****************************************************************************/
       p_idVia IN VIA.idVia%TYPE,
       p_descripcioVia in VIA.descripcioVia%TYPE,
```

```
    s_rsp out NOCOPY VARCHAR2)
AS

BEGIN
 c_procesLog := 'PRC_MODIFICAR_VIA';
 c_dataHoraLog := SYSDATE;
 c_entradaLog :='Decripcio Via: ' || p_descripcioVia;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(' ');
 DBMS_OUTPUT.PUT_LINE('MODIFICAR LA VIA PUBLICA');
 DBMS_OUTPUT.PUT_LINE('_____');
 DBMS_OUTPUT.PUT_LINE('Nom de la Via Publica: '|| p_descripcioVia);
 c_sortidalog := 's_rsp';
 -- ComprovaciÃ³ del identificador de la Via
  IF  p_descripcioVia IS NULL THEN
   DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('MODIFICAR_VIA: ');
  s_rsp := 'Falta especificar el Nom de la Via Publica: ';
   DBMS_OUTPUT.put_line(s_rsp);
    RAISE e_via;

  -- 1er. Comprovem LA DESCRIPCIO
     SELECT COUNT (*)INTO n_registres
       FROM VIA
       WHERE DESCRIPCIOVIA =p_descripcioVia;
      If n_registres=0 then
       DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('MODIFICAR_VIA: ');
       s_rsp := 'VIA no existent a la BBDD';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_via;
      END IF;
   -- recuperem el codi de la via
    SELECT VIA.idVia,VIA.DESCRIPCIOVIA INTO n_idVia,n_descripcioVia
        FROM VIA
        WHERE DESCRIPCIOVIA=p_descripcioVia;

    s_sql := 'UPDATE VIA SET ';

    -- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
         IF p_descripcioVia IS NOT NULL THEN
             -- Modifiquem el nom
             s_sql := s_sql || 'descripcioVia='' || p_descripcioVia || '',';

         END IF;

         -- Eliminem la coma final de la sentencia SQL
         -------------------------
         s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
         -- Afegim a la sentencia la condiciÃ³ del WHERE
         ----------------------
         s_sql := s_sql || ' WHERE descripcioVia='' || p_descripcioVia|| '';
```

```
        EXECUTE IMMEDIATE s_sql;

        IF SQL%ROWCOUNT = 0
        THEN
          --L'error serÃ  que el la via ja el tenim en la taula i no admet duplicats--
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICAR_VIA: ');
          s_rsp := 'ActualitzaciÃ³ no realitzada';
          DBMS_OUTPUT.PUT_LINE(s_rsp);
          RAISE e_via;
        ELSE
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('Modificacio feta de la Via Pública: codi: '||p_idVia||
         ' Descripcio: '|| p_descripcioVia);
         DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
          s_rsp := 'OK Modificacio feta de la Via Pública: codi: '||n_idVia||
         ' Descripcio: '|| p_descripcioVia;
         pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
        END IF;
  END IF;
 COMMIT;
 EXCEPTION
 WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_MODIFICAR_VIA;
/*****************************************************************************
CONSULTA DE TOTES LES VIES PUBLIQUES DONADES D´ALTA
*****************************************************************************/
PROCEDURE PRC_CONSULTA_VIA(
   s_rsp out NOCOPY VARCHAR2)
AS
CURSOR C_VIAS IS
 SELECT VIA.IDVIA,VIA.DESCRIPCIOVIA
 FROM VIA
 WHERE VIA.IDVIA=VIA.IDVIA;

BEGIN
   c_procesLog := 'PRC_CONSULTA_VIAS';
   c_dataHoraLog := SYSDATE;
```

```
    c_entradaLog :='CONSULTA DE LES VIES PUBLIQUES';
    c_sortidalog := 's_rsp';

    SELECT COUNT(VIA.IDVIA)INTO n_registres
    FROM VIA
    WHERE VIA.IDVIA=VIA.IDVIA;

    IF n_registres=0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTA DE VIES PUBLIQUES: ');
    s_rsp := 'No hi han vies publiques donades d´alta ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_via;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          CONSULTA DE LES VIES PUBLIQUES          ');
    DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------');
  OPEN C_VIAS;
   FETCH C_VIAS INTO n_idVia,n_descripcioVia;
  WHILE C_VIAS%FOUND LOOP
   DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(
'   Codi Via '
|| n_idVia
||' Descripcio '
|| n_descripcioVia);
  s_rsp := 'OK  CONSULTA DE LES VIES PUBLIQUES
  Codi Via:  '
|| n_idVia
||' Descripcio: '
|| n_descripcioVia;

  FETCH C_VIAS INTO n_idVia,n_descripcioVia;

  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total vies públiques: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_VIAS;
COMMIT;
 EXCEPTION

   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi----------------------------------------
       s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi----------------------------------------
       s_rsp := 'Error: ' || s_rsp;
     END IF;
```

```
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_CONSULTA_VIA;
END GESTION_VIA;
```

10.3 TRATAMIENTO DE PAIS.

DEFINICION	PROCEDIMIENTO
Alta de Pais	PRC_ALTA_PAIS
Modificar Pais	PRC_MODIFICAR_PAIS
Consulta de Pais	PRC_CONSULTAR_PAIS

```
CREATE OR REPLACE
PACKAGE        "GESTION_PAIS" AS
  procedure PRC_ALTA_PAIS(
      p_descripcioPais in PAIS.descripcioPais%TYPE,
      p_codiPais in PAIS.codiPais%TYPE,
      s_rsp out NOCOPY VARCHAR2);
   --Modificar un Pais
   procedure PRC_MODIFICAR_PAIS(
      p_idPais IN PAIS.IDPAIS%TYPE,
      p_descripcioPais in PAIS.descripcioPais%TYPE,
      p_codiPais in PAIS.codiPais%TYPE,
      s_rsp out NOCOPY VARCHAR2);
    --Consultar un Pais
   procedure PRC_CONSULTAR_PAIS(
    s_rsp out NOCOPY VARCHAR2);
end GESTION_PAIS;
```

10.3.1 PROCEDIMIENTO SPL PACKAGE DE PAIS.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_PAIS" AS
  c_procesLog          LOG_TFC.PROCESLOG%TYPE;
  c_dataHoraLog    LOG_TFC.DATAHORALOG%TYPE;
  c_entradaLog     LOG_TFC.ENTRADALOG%TYPE;
  c_sortidaLog     LOG_TFC.SORTIDALOG%TYPE;
  s_rsp              LOG_TFC.RSPLOG%TYPE;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);

  n_descripcioPais PAIS.descripcioPais%TYPE;
  n_codiPais PAIS.codiPais%TYPE;
  n_idPais   PAIS.IDPAIS%TYPE;
  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_pais     EXCEPTION;
-------------------------------------------------------------
-- create procedure for table "LOG_TFC i PAIS"
/*****************************************************************************
Autor: Eduard Monzonis Hierro                          UOC
18/05/2012                                TFC: CONTROL ENERGIA.
              PROCEDIEMENT GESTIO PAIS
DESCRIPCIO:
La funcionalitat d'aquest es donar d'alta el paisos que necesita el sitema per
gestionar PROVICIES, LOCALITAT i les UBICACIONS que es on hi han les dades
de les direccions del CLIENTS, FABRICANTS, COMPTADORS, CENTRALS
*****************************************************************************/
   procedure PRC_ALTA_PAIS(
    p_descripcioPais in PAIS.descripcioPais%TYPE,
    p_codiPais in PAIS.codiPais%TYPE,
    s_rsp out NOCOPY VARCHAR2)
   AS

   BEGIN
    c_procesLog := 'PRC_ALTA_PAIS';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='Decripcio Pais: ' || p_descripcioPais;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('     ALTA PAIS     ');
    DBMS_OUTPUT.PUT_LINE('_____________________________');
    DBMS_OUTPUT.PUT_LINE(' Decripcio Pais: ' || p_descripcioPais||
    ' codi de Pais: '||p_codiPais);
    c_sortidalog := 's_rsp';
    If (p_descripcioPais IS NULL) then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ALTA_PAIS: ');
      s_rsp := 'Falta especificar el País ';
      RAISE e_pais;
```

```
    END IF;
     -- Comprovem si existeix el pais a  donar d'alta
       SELECT COUNT (*)INTO n_registres
         FROM PAIS
         WHERE PAIS.descripcioPais = p_descripcioPais;
     If n_registres=0 then
        -- El donem d'alta  PaÃs
         INSERT INTO PAIS(descripcioPais,codiPais)
         VALUES(p_descripcioPais,p_codiPais);
         --Gravem en la taula log----------------------------------------------------
        DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('Insertat País: '||p_descripcioPais||
         ' Codi ISO del País:  '||p_codiPais);
         DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
         s_rsp :='Ok: Insertada País: '||p_descripcioPais||
         ' Codi ISO del País:  '||p_codiPais;
         pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        ELSE

         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('ALTA_PAIS: ');
        s_rsp := 'DESCRICIO Pais  duplicat';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_pais;
     END IF;
   COMMIT;
EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
     ELSE
        -- L'error si ha estat controlat per codi
        s_rsp := 'Error: ' || s_rsp;
     END IF;
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE(' ');
         DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;

   end PRC_ALTA_PAIS;

  PROCEDURE PRC_MODIFICAR_PAIS(
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
   NOM:       PRC_MODIFICAR_PAIS
   DESCRIPCIÃ“:
            Procediment encarregat de modificar les dades d'un pais.
            Les dades es modificaran en la taula persona i el identificador de
```

```
            Via apuntarÃ  a la clau primaria del contacte de la taula Pais.
*********************************************************************************/
      p_idPais IN PAIS.idPais%TYPE,
      p_descripcioPais in PAIS.descripcioPais%TYPE,
      p_codiPais in PAIS.codiPais%TYPE,
      s_rsp out NOCOPY VARCHAR2)
   AS

   BEGIN
    c_procesLog := 'PRC_MODIFICAR_PAIS';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='Decripcio Pais: ' || p_descripcioPais||
                ', Codi_Pais: ' || p_codiPais;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('      MODIFICAR UN PAÍS       ');
    DBMS_OUTPUT.PUT_LINE('_____________________________');
    DBMS_OUTPUT.PUT_LINE('Decripcio Pais: ' || p_descripcioPais||
                ', Codi_Pais: ' || p_codiPais);
    c_sortidalog := 's_rsp';
    -- ComprovaciÃ³ del identificador Pais-----
    IF p_descripcioPais IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR UN PAÍS: ');
      s_rsp := 'Falta especificar el país a modificar';
      DBMS_OUTPUT.put_line(s_rsp);
        RAISE e_pais;
    END IF;
    SELECT PAIS.IDPAIS INTO n_registres
    FROM PAIS
    WHERE PAIS.DESCRIPCIOPAIS = p_descripcioPais;

    If n_registres=0 then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR_PAIS: ');
      s_rsp := 'PAIS no existeix a la BBDD';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_pais;
    END IF;

    --CERQUEM EL PAIS
        -- recuperem el codi Pais
    SELECT PAIS.idPais INTO n_idPais
    FROM PAIS
    WHERE PAIS.DESCRIPCIOPAIS= p_descripcioPais;

   s_sql := 'UPDATE PAIS SET ';

-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
   IF p_descripcioPais IS NOT NULL THEN
-- Modifiquem el descripcio pasis
      s_sql := s_sql || 'DESCRIPCIOPAIS=''' || p_descripcioPais || ''',';
```

```
 END IF;
   IF p_codiPais IS NOT NULL THEN
-- Modifiquem el codi paÃs
    s_sql := s_sql || 'CODIPAIS="' || p_codiPais || '",';
  END IF;

-- Eliminem la coma final de la sentencia SQL
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃ³ del WHERE
   s_sql := s_sql || ' WHERE IDPAIS="' ||n_idPais|| '"';

 EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0  THEN
--L'error serÃ  que el la via ja el tenim en la taula i no admet duplicats--
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR_PAIS: ');
     s_rsp := 'Actualitzacio no realitzada';
     RAISE e_pais;
   ELSE
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('Modificacio feta del País: codi: '||n_idPais||
             ' Descripcio: '||p_descripcioPais||
             ' codi ISO: '||p_codiPais);
             DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
              s_rsp := 'OKModificacio feta del País: codi: '||n_idPais||
             ' Descripcio: '||p_descripcioPais||
             ' codi ISO: '||p_codiPais;

     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
   END IF;

COMMIT;
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE(' MODIFICAR_PAIS:  ');
    s_rsp := 'Error: Pais duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('  ');
```

```
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_MODIFICAR_PAIS;

/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
  NOM:       PRC_CONSULTAR_PAIS
  DESCRIPCIÃ“:
            Procediment encarregat de consultar les dades d'un pais.
            Les dades a consultar en la taula pais i el identificador de
            pais apuntarÃ  a la clau primaria del Pais de la taula Pais.
*****************************************************************************/
PROCEDURE PRC_CONSULTAR_PAIS(
   s_rsp out NOCOPY VARCHAR2)
AS

CURSOR C_PAIS IS
SELECT PAIS.IDPAIS,PAIS.DESCRIPCIOPAIS,
      PAIS.CODIPAIS
FROM PAIS
WHERE PAIS.IDPAIS=PAIS.IDPAIS;

BEGIN
  c_procesLog := 'PRC_CONSULTAR_PAIS';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='   CONSULTA PAISOS   ';
   c_sortidalog := 's_rsp';
-- Comprovem que p_descripcioPais no sigui NULL----------------------------------------------
   SELECT COUNT(*)INTO n_registres
   FROM PAIS
   WHERE PAIS.IDPAIS=PAIS.IDPAIS;
  IF n_registres=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTAR_PAIS: ');
      s_rsp := 'No hi han paisos donats en alta.  ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_pais;
  END IF;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('          CONSULTA PAISOS          ');
  DBMS_OUTPUT.PUT_LINE('-----------------------------------');

OPEN C_PAIS;
  FETCH C_PAIS INTO n_idPais,n_descripcioPais,n_codiPais;
  WHILE C_PAIS%FOUND LOOP
   DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(
 '   Codi Pais '
 ||n_idPais
 ||' Descripcio '
```

```
|| n_descripcioPais||
' codi ISO pais: '||n_codiPais);
 s_rsp := 'OK  CONSULTA DELS PAISOS
  Codi Pais '
||n_idPais
||' Descripcio '
|| n_descripcioPais||
' codi ISO pais: '||n_codiPais;

 FETCH C_PAIS INTO n_idPais,n_descripcioPais,n_codiPais;

 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total paisos donat en alta: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_PAIS;

 COMMIT;
EXCEPTION

 WHEN OTHERS THEN
   IF s_rsp IS NULL THEN
     -- L'error no ha estat controlat per codi----------------------------------------
     s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
   ELSE
     -- L'error si ha estat controlat per codi----------------------------------------
     s_rsp := 'Error: ' || s_rsp;
   END IF;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.put_line (s_rsp);
   pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
   ROLLBACK;

 END PRC_CONSULTAR_PAIS;
END GESTION_PAIS;
```

10.4 TRATAMIENTO DE PROVINCIA.

DEFINICION	PROCEDIMIENTO
Alta de Provincias	PRC_ALTA_PROVINCIA
Modificar Provincias	PRC_MODIFICAR_PROVINCIA
Consulta de una Provincia	PRC_CONSULTA_UNA_PROVINCIA
Consulta de Provincias	PRC_CONSULTA_PROVINCIA

```
CREATE OR REPLACE
PACKAGE        "GESTION_PROVINCIA" AS
-- GESTIO DE PROVINCIES
PROCEDURE PRC_ALTA_PROVINCIA(
   p_descripcioProvincia in PROVINCIA.DESCRIPCIOPROVINCIA%TYPE,
   p_idPaisProvincia in PROVINCIA.IDPAISPROVINCIA%TYPE,
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_MODIFICAR_PROVINCIA(
   p_idProvincia in PROVINCIA.IDPROVINCIA%TYPE,
   p_descripcioProvincia in PROVINCIA.DESCRIPCIOPROVINCIA%TYPE,
   p_idPaisProvincia in PROVINCIA.IDPAISPROVINCIA%TYPE,
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_CONSULTA_UNA_PROVINCIA(
  p_descripcioProvincia in PROVINCIA.DESCRIPCIOPROVINCIA%TYPE,
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_CONSULTA_PROVINCIA(
   s_rsp out NOCOPY VARCHAR2);
END GESTION_PROVINCIA;
```

10.4.1 PROCEDIMIENTO SPL PACKAGE DE PROVINCIAS.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_PROVINCIA" AS
  c_procesLog         LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp           LOG_TFC.rspLog%TYPE;
```

```
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);
  n_descripcioProvincia PROVINCIA.DESCRIPCIOPROVINCIA%TYPE;
  n_idPaisProvincia PROVINCIA.IDPAISPROVINCIA%TYPE;
  n_idProvincia  PROVINCIA.IDPROVINCIA%TYPE;
  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_provincia     EXCEPTION;
  ------------------------------------------------------------------
-- create procedure for table "LOG_TFC i PROVINCIA"
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
   NOM:        PRC_ALTA_PROVINCIA
   DESCRIPCIÃ“:
            Procediment encarregat donar d'alta provincias onn el id de pais
            direcciona a la taula PAIS
*****************************************************************************/
PROCEDURE PRC_ALTA_PROVINCIA(
    p_descripcioProvincia in PROVINCIA.DESCRIPCIOPROVINCIA%TYPE,
    p_idPaisProvincia in PROVINCIA.IDPAISPROVINCIA%TYPE,
    s_rsp out NOCOPY VARCHAR2)
    AS
    BEGIN
    c_procesLog := 'PRC_ALTA_PROVINCIA';
    c_dataHoraLog := SYSDATE;
    c_procesLog := 'PRC_ALTA_PROVINCIA';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='Decripcio Via: ' ||p_descripcioProvincia;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('   ALTA PROVINCIA     ');
    DBMS_OUTPUT.PUT_LINE('_____________________');
    DBMS_OUTPUT.PUT_LINE('Provincia: '||p_descripcioProvincia||
    ' codi Pais_ '||p_idPaisProvincia);
    c_sortidalog := 's_rsp';

    If (p_descripcioProvincia IS NULL) OR (p_idPaisProvincia IS NULL ) then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE(' ALTA PROVINCIA: ');
         s_rsp := 'Falta especificar la provincia i el codi de Pais ';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE  e_provincia;
     END IF;
-- Comprovem si existeix el la provincia a  donar d'alta
        SELECT COUNT (*)INTO n_registres
          FROM PROVINCIA
          WHERE DESCRIPCIOPROVINCIA =p_descripcioProvincia
          AND PROVINCIA.IDPAISPROVINCIA=p_idPaisProvincia;

        If n_registres=0 then
        -- El donem d'alta Provincia
```

```
        INSERT INTO PROVINCIA(PROVINCIA.DESCRIPCIOPROVINCIA,PROVINCIA.IDPAISPROVINCIA)
        VALUES(p_descripcioProvincia,p_idPaisProvincia);
        --Gravem en la taula log----------------------------------------------------
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('Insertada la Provincia: '||p_descripcioProvincia||
      ' amb el codi de Pais: ' ||p_idPaisProvincia);
      DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
      s_rsp :='Ok: Insertada la Provincia: '||p_descripcioProvincia||
      ' amb el codi de Pais: ' ||p_idPaisProvincia;
      pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
       DBMS_OUTPUT.PUT_LINE(s_rsp);
    ELSE
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ALTA PROVINCIA: ');
      s_rsp := 'Dades de la provincia duplicades.  ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_provincia;
    END IF;
COMMIT;
  EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_ALTA_PROVINCIA;
/*****************************************************************************
 Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
  NOM:       PRC_MODIFICAR_PROVINCIA
  DESCRIPCIÃ“:
          Procediment encarregat donar de modificar una o vaies provincies

*****************************************************************************/
PROCEDURE PRC_MODIFICAR_PROVINCIA(
   p_idProvincia in PROVINCIA.IDPROVINCIA%TYPE,
   p_descripcioProvincia in PROVINCIA.DESCRIPCIOPROVINCIA%TYPE,
   p_idPaisProvincia in PROVINCIA.IDPAISPROVINCIA%TYPE,
   s_rsp out NOCOPY VARCHAR2)
   AS
   BEGIN
   c_procesLog := 'PRC_MODIFICAR_PROVINCIA';
   c_dataHoraLog := SYSDATE;
```

```
    c_entradaLog :='Id Provincia: ' ||p_idProvincia||
            ', Decripcio Provincia: ' ||p_descripcioProvincia ||
            ', Codi Pais: ' || p_idPaisProvincia;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('  MODIFICAR UNA PROVINCIA   ');
    DBMS_OUTPUT.PUT_LINE('_____________________________');
    DBMS_OUTPUT.PUT_LINE('Codi Provincia: ' ||p_idProvincia||
            ', Decripcio Provincia: ' ||p_descripcioProvincia ||
            ', Codi Pais: ' || p_idPaisProvincia);
    c_sortidalog := 's_rsp';

    -- ComprovaciÃ³ del identificador Provincia-----
    IF p_idProvincia IS NULL THEN
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICAR UNA PROVINCIA : ');
          s_rsp := 'Falta especificar el codi Provincia';
          DBMS_OUTPUT.put_line(s_rsp);
        RAISE e_provincia;
    END IF;
    If  p_descripcioProvincia IS NULL THEN
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICAR UNA PROVINCIA: ');
                s_rsp := 'Falta especificar la Provincia';
          DBMS_OUTPUT.put_line(s_rsp);
          RAISE e_provincia;
    END IF;
    If   p_idPaisProvincia IS NULL THEN
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICAR UNA PROVINCIA: ');
                s_rsp := 'Falta especificar el codi del Pais. ';
          DBMS_OUTPUT.put_line(s_rsp);
          RAISE e_provincia;
    END IF;
  SELECT COUNT (*)INTO n_registres
  FROM PROVINCIA
  WHERE PROVINCIA.IDPROVINCIA = p_idProvincia
  AND PROVINCIA.IDPAISPROVINCIA=p_idPaisProvincia;

   If n_registres=0 then
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR UNA PROVINCIA: ');
     s_rsp := 'PROVINCIA no existeix a la BBDD';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_provincia;
    END IF;

  -- recuperem el codi Provincia
    SELECT PROVINCIA.IDPROVINCIA INTO n_idProvincia
    FROM PROVINCIA
    WHERE PROVINCIA.IDPROVINCIA = p_idProvincia
    AND PROVINCIA.IDPAISPROVINCIA=p_idPaisProvincia;
```

```
 s_sql := 'UPDATE PROVINCIA SET ';
-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
 IF p_descripcioProvincia IS NOT NULL THEN
-- Modifiquem el descripcio PROVINCIA
  s_sql := s_sql || 'DESCRIPCIOPROVINCIA=''' || p_descripcioProvincia || ''',';
 END IF;
 IF p_descripcioProvincia IS NOT NULL THEN
  s_sql := s_sql || 'IDPAISPROVINCIA=''' || p_idPaisProvincia || ''',';
 END IF;
-- Eliminem la coma final de la sentencia SQL
 s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃ³ del WHERE
 s_sql := s_sql || ' WHERE IDPROVINCIA=''' || n_idProvincia|| '''';

  EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0  THEN
--L'error serÃ  que el la via ja el tenim en la taula i no admet duplicats--
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR_PROVINCIA:  ');
     s_rsp := 'Actualitzacio no realitzada';
     RAISE e_provincia;
   ELSE
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('Modificacio de la Provincia
     feta satisfactoriament : codi Provincia: '||n_idProvincia||
              ' Descripcio: ' ||p_idPaisProvincia ||
              ' codi del Pais: '||p_idPaisProvincia );
     DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
   END IF;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
     ELSE
        -- L'error si ha estat controlat per codi
        s_rsp := 'Error: ' || s_rsp;
     END IF;
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;
END PRC_MODIFICAR_PROVINCIA;

/*****************************************************************************
 Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
  NOM:       CONSULTA_PROVINCIA
  DESCRIPCIÃ“:
```

```
            Procediment encarregat de consultar les dades d'una Provincia.
            Les dades a consultar en la taula Provincia i el identificador de
            Provincia apuntarÃ  a la clau primaria  de la taula Pais.
*********************************************************************************/
PROCEDURE PRC_CONSULTA_UNA_PROVINCIA(
  p_descripcioProvincia in PROVINCIA.DESCRIPCIOPROVINCIA%TYPE,
    s_rsp out NOCOPY VARCHAR2)
AS
n_descripcioPais  PAIS.DESCRIPCIOPAIS%TYPE;
n_codiPais PAIS.CODIPAIS%TYPE;
BEGIN
 c_procesLog := 'PRC_CONSULTA_PROVINCIA';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :=' Decripcio Provincia: ' ||p_descripcioProvincia;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('  CONSULTA D''UNA PROVINCIA ');
    DBMS_OUTPUT.PUT_LINE('_____');
    DBMS_OUTPUT.PUT_LINE('Decripcio Provincia: ' ||p_descripcioProvincia);
    c_sortidalog := 's_rsp';
 -- Comprovem que p_descripcioProvincia  no sigui NULL----------------------------------------------
      If  p_descripcioProvincia   IS NULL then
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('CONSULTA D''UNA PROVINCIA: ');
                s_rsp := 'Falta especificar Provincia';
          DBMS_OUTPUT.put_line(s_rsp);
                  RAISE  e_provincia;
      ELSE
  --1er. Comprovem que existeixi la Provincia------------------------------

     SELECT COUNT (*)INTO n_registres
     FROM PROVINCIA
     WHERE PROVINCIA.IDPROVINCIA =PROVINCIA.IDPROVINCIA
     AND PROVINCIA.DESCRIPCIOPROVINCIA=p_descripcioProvincia;

  If n_registres=0 then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTA D''UNA PROVINCIA: ');
    s_rsp := 'PROVINCIA no existent a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE  e_provincia;
  END IF;
       -- Recuperem les dades de la taula PROVINCIA-------------------------
   SELECT PROVINCIA.IDPROVINCIA,
      PROVINCIA.DESCRIPCIOPROVINCIA,
      PAIS.DESCRIPCIOPAIS,
      PAIS.CODIPAIS INTO n_idProvincia,n_descripcioProvincia,
      n_descripcioPais,n_codiPais
      FROM PROVINCIA, PAIS
      WHERE  PROVINCIA.DESCRIPCIOPROVINCIA= p_descripcioProvincia
      AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS;

      DBMS_OUTPUT.ENABLE;
```

```
    DBMS_OUTPUT.PUT_LINE('Consulta de la provincia: codi: '||n_idProvincia||
            ' Descripcio: '||n_descripcioProvincia||
            ' Pais: '||n_descripcioPais||
            ' codi Pais ISO: '||n_codiPais);
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
    s_rsp := 'OK Consulta de la provincia: codi: '||n_idProvincia||
            ' Descripcio: '||n_descripcioProvincia||
            ' Pais: '||n_descripcioPais||
            ' codi Pais ISO: '||n_codiPais;
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
end PRC_CONSULTA_UNA_PROVINCIA;

-- LLISTAT DE PROVINCIES AMB PAIS
PROCEDURE PRC_CONSULTA_PROVINCIA(
   s_rsp out NOCOPY VARCHAR2)
AS
n_descripcioPais  PAIS.DESCRIPCIOPAIS%TYPE;
n_codiPais PAIS.CODIPAIS%TYPE;
CURSOR C_PROVINCIA IS
 SELECT PROVINCIA.IDPROVINCIA,
    PROVINCIA.DESCRIPCIOPROVINCIA,
    PAIS.DESCRIPCIOPAIS,
    PAIS.CODIPAIS
    FROM PROVINCIA, PAIS
    WHERE   PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS;

BEGIN
   c_procesLog := 'PRC_CONSULTA_PROVINCIES';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='CONSULTA PROVINCIAS';
   c_sortidalog := 's_rsp';

 SELECT COUNT (*)  INTO n_registres
 FROM PROVINCIA
 WHERE PROVINCIA.IDPROVINCIA= PROVINCIA.IDPROVINCIA;
```

```
IF n_registres=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTA de PROVINCIES: ');
   s_rsp := 'No hi han provinicies donades d´alta ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_provincia;
END IF;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('          CONSULTA DE LES PROVINCIES              ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------------');
 OPEN C_PROVINCIA;
  FETCH C_PROVINCIA INTO n_idProvincia,n_descripcioProvincia,n_descripcioPais,n_codiPais;
 WHILE C_PROVINCIA%FOUND LOOP
  DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(
'   Codi Provincia '
|| n_idProvincia||
' Descripcio '
||n_descripcioProvincia||
' Pais: '||n_descripcioPais||
' Codi ISO pais: '||n_codiPais);
 s_rsp := 'OK  CONSULTA PROVINCIES
 Codi Provincia '
|| n_idProvincia||
' Descripcio '
||n_descripcioProvincia||
' Pais: '||n_descripcioPais||
' Codi ISO pais: '||n_codiPais;

 FETCH C_PROVINCIA INTO n_idProvincia,
 n_descripcioProvincia,n_descripcioPais,n_codiPais;

  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total provincies: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_PROVINCIA;
COMMIT;
 EXCEPTION

  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
```

```
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;


end PRC_CONSULTA_PROVINCIA;
END "GESTION_PROVINCIA";
```

10.5 TRATAMIENTO DE LOCALIDADES.

DEFINICION	PROCEDIMIENTO
Alta de Localidades	PRC_ALTA_LOCALITAT
Modificar Localidades	PRC_MODIFICAR_LOCALITAT
Consulta de una Localidades	PRC_CONSULTA_UNA_LOCALITAT
Consulta de Localidades	PRC_CONSULTA_LOCALITATS

```
CREATE OR REPLACE
PACKAGE       "GESTION_LOCALITAT" AS
-- GESTIO DE LOCALITAT
PROCEDURE PRC_ALTA_LOCALITAT(
   p_descripcioLocalitat in LOCALITAT.DESCRIPCIOLOCALITAT%TYPE,
   p_idProvinciaLocalitat in LOCALITAT.IDPROVINCIALOCALITAT%TYPE,
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_MODIFICAR_LOCALITAT(
   p_idLocalitat in LOCALITAT.IDLOCALITAT%TYPE,
   p_descripcioLocalitat in LOCALITAT.DESCRIPCIOLOCALITAT%TYPE,
   p_idProvinciaLocalitat in LOCALITAT.IDPROVINCIALOCALITAT%TYPE,
   s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_CONSULTA_LOCALITATS(
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_CONSULTA_UNA_LOCALITAT(
   p_descripcioLocalitat in LOCALITAT.DESCRIPCIOLOCALITAT%TYPE,
   s_rsp out NOCOPY VARCHAR2);
END "GESTION_LOCALITAT";
```

10.5.1 PROCEDIMIENTO SPL PACKAGE DE LOCALITATS.

```
CREATE OR REPLACE
PACKAGE BODY         "GESTION_LOCALITAT" AS
  c_procesLog           LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp               LOG_TFC.rspLog%TYPE;
  n_registres  NUMBER;
  n_idLocalitat LOCALITAT.IDLOCALITAT%TYPE;
  n_descripcioLocalitat LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
  n_descripcioProvincia PROVINCIA.DESCRIPCIOPROVINCIA%TYPE;
  n_descripcioPais PAIS.DESCRIPCIOPAIS%TYPE;

  s_sql VARCHAR2 (2000);
  n_descripcioLocalitat LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
  n_idpProvinciaLocalitat LOCALITAT.IDPROVINCIALOCALITAT%TYPE;

  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_localitat    EXCEPTION;
  ------------------------------------------------------------------
-- create procedure for table "LOG_TFC i LOCALITAT"

/*****************************************************************************
Autor: Eduard Monzonis Hierro                          UOC
18/05/2012                                TFC: CONTROL ENERGIA.
            PROCEDIMENT ALTA DE LOCALITAT
DESCRIPCIO:
Procediment per gestionar les localitats on estan geograficamnt el COMPTADORS,
CENTRALS, CLIENTS, FABRICANTS. Totes aquelles entitats que necessiten per
motius de negoci esta ubicades en un lloc fÃsic.
D'aquest procediement depent PROVINCIa  i PAÃ?S, que l'entitat UBICACIO hereda
dades d'aquesta entitat per poder gestionar les direcciÃ³n de les entitats
que precisen esta ubicades de forma fÃsica per motius del negoci qe el sistema
en el rojecte precisa.
*****************************************************************************/
PROCEDURE PRC_ALTA_LOCALITAT(
   p_descripcioLocalitat in LOCALITAT.DESCRIPCIOLOCALITAT%TYPE,
   p_idProvinciaLocalitat in LOCALITAT.IDPROVINCIALOCALITAT%TYPE,
   s_rsp out NOCOPY VARCHAR2)
   AS

   BEGIN
   c_procesLog :='PRC_ALTA_LOCALITAT';
   c_dataHoraLog := SYSDATE;
   c_entradaLog := 'Decripcio LOCALITAT: '|| p_descripcioLocalitat ||
            ', Codi de Provincia: '|| p_idProvinciaLocalitat;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
```

```
    DBMS_OUTPUT.PUT_LINE('    ALTA LOCALITAT    ');
    DBMS_OUTPUT.PUT_LINE('_____________________');
    DBMS_OUTPUT.PUT_LINE('Decripcio LOCALITAT: '|| p_descripcioLocalitat ||
                ', Codi de Provincia: '|| p_idProvinciaLocalitat);
    c_sortidalog := 's_rsp';

  If (p_descripcioLocalitat IS NULL) OR (p_idProvinciaLocalitat IS NULL ) then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ALTA LOCALITAT: ');
      s_rsp := 'Falta especificar la localitat i el codi de Provincia ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_localitat;
  END IF;
     -- Comprovem si existeix  la localitat a  donar d'alta
  SELECT COUNT (*)INTO n_registres
    FROM LOCALITAT
  WHERE DESCRIPCIOLOCALITAT = p_descripcioLocalitat;
  If n_registres=0 then
    -- El donem d'alta a la Via Publica
  INSERT INTO LOCALITAT(LOCALITAT.DESCRIPCIOLOCALITAT,
    LOCALITAT.IDPROVINCIALOCALITAT)
  VALUES(p_descripcioLocalitat,p_idProvinciaLocalitat);
--Gravem en la taula log-----------------------------------------------------
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('Insertada la Localitat: '||p_descripcioLocalitat||
        ' amb el codi de Provincia: ' ||p_idProvinciaLocalitat);
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
    s_rsp :='Ok: Insertada la Localitat: '||p_descripcioLocalitat||
        ' amb el codi de Provincia: ' ||p_idProvinciaLocalitat;
    pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    DBMS_OUTPUT.PUT_LINE(s_rsp);
  ELSE
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA LOCALITAT: ');
    s_rsp := 'Dades de la Localitat duplicades.  ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_localitat;
  END IF;
COMMIT;

   EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
     ELSE
        -- L'error si ha estat controlat per codi
        s_rsp := 'Error: ' || s_rsp;
     END IF;
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('ALTA_LOCALITAT: ');
          DBMS_OUTPUT.put_line(s_rsp);
```

```
        pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_ALTA_LOCALITAT;

/*******************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
              PROCEDURE MODIFICAR LOCALITAT
DESCRIPCIO:
Procediment per gestionar les modificacions de l'entitat.
*******************************************************************************/
PROCEDURE PRC_MODIFICAR_LOCALITAT(
   p_idLocalitat in LOCALITAT.IDLOCALITAT%TYPE,
   p_descripcioLocalitat in LOCALITAT.DESCRIPCIOLOCALITAT%TYPE,
   p_idProvinciaLocalitat in LOCALITAT.IDPROVINCIALOCALITAT%TYPE,
   s_rsp out NOCOPY VARCHAR2)
   AS

   BEGIN
   c_procesLog := 'PRC_MODIFICAR_LOCALITAT';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Codi Localitat: ' || p_idLocalitat||
              ', Descripcio Localitat: ' || p_descripcioLocalitat||
              ', Codi Provincia: ' ||p_idProvinciaLocalitat;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('  MODIFICAR UNA LOCALITAT   ');
  DBMS_OUTPUT.PUT_LINE('_____________________________');
  DBMS_OUTPUT.PUT_LINE(' Codi Localitat: ' || p_idLocalitat||
              ', Descripcio Localitat: ' || p_descripcioLocalitat||
              ', Codi Provincia: ' ||p_idProvinciaLocalitat);
   c_sortidalog := 's_rsp';
   -- ComprovaciÃ³ del identificador Localitat-----
 IF p_idLocalitat IS NULL OR p_idLocalitat=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR UNA LOCALITAT: ');
   s_rsp := 'Falta especificar  el codi Localitat';
   DBMS_OUTPUT.put_line(s_rsp);
   RAISE e_localitat;
 END IF;
 If p_descripcioLocalitat IS NULL THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR UNA LOCALITAT: ');
   s_rsp := 'Falta especificar Localitat';
   DBMS_OUTPUT.put_line(s_rsp);
   RAISE e_localitat;
 END IF;
 If p_idProvinciaLocalitat IS NULL OR p_idProvinciaLocalitat=0 THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('ODIFICAR UNA LOCALITAT: ');
     s_rsp := 'Falta especificar codi de la Provincia de la Localitat';
     DBMS_OUTPUT.put_line(s_rsp);
     RAISE e_localitat;
```

```
 END IF;
 -- MIREM SI ESTA DONAT ALTA
 SELECT COUNT (*)INTO n_registres
 FROM LOCALITAT
 WHERE LOCALITAT.IDLOCALITAT =p_idLocalitat;

 If n_registres=0 then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR UNA LOCALITAT: ');
  s_rsp := 'LOCALITAT no existeix a la BBDD';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_localitat;
 END IF;
-- recuperem el codi de la Localitat
 SELECT LOCALITAT.IDLOCALITAT INTO n_idLocalitat
 FROM LOCALITAT
 WHERE IDLOCALITAT = p_idLocalitat;

 s_sql := 'UPDATE LOCALITAT SET ';

-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
 IF  p_descripcioLocalitat IS NOT NULL  THEN
-- Modifiquem el codi Localitat
  s_sql := s_sql || 'DESCRIPCIOLOCALITAT='' ||p_descripcioLocalitat|| ''','';
 END IF;
 IF p_idProvinciaLocalitat IS NOT NULL  THEN
-- Modifiquem CODI PROVINCIA
   s_sql := s_sql || 'IDPROVINCIALOCALITAT='' ||p_idProvinciaLocalitat|| ''','';
 END IF;
-- Eliminem la coma final de la sentencia SQL
 s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
 -- Afegim a la sentencia la condiciÃ³ del WHERE
 s_sql := s_sql || ' WHERE IDLOCALITAT'' ||n_idLocalitat|| '''';

 EXECUTE IMMEDIATE s_sql;

 IF SQL%ROWCOUNT = 0  THEN
--L'error serÃ  que el la via ja el tenim en la taula i no admet duplicats--
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR UNA LOCALITAT: ');
   s_rsp := 'Actualitzacio no realitzada';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_localitat;
 ELSE
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('Modificacio de la Localitat
    feta satisfactoriament : codi Localitat: '||n_idLocalitat||
            ' Descripcio: ' ||p_descripcioLocalitat||
            ' codi del Provincia: '||p_idProvinciaLocalitat);
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
```

```
COMMIT;
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UNA LOCALITAT: ');
    s_rsp := 'Error: Localitat duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
 END PRC_MODIFICAR_LOCALITAT;
/*****************************************************************************
 Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
            CONSULTA DE LES LOCALITATS
DESCRIPCIO:
Procediment per consultar una Localitat i que per herencia i integritat
referencial en cascade obtenim PROVINCIA i PAIS de la LOCALITAT.
*****************************************************************************/

PROCEDURE PRC_CONSULTA_UNA_LOCALITAT(
   p_descripcioLocalitat in LOCALITAT.DESCRIPCIOLOCALITAT%TYPE,
   s_rsp out NOCOPY VARCHAR2)
 AS
 n_idLocalitat LOCALITAT.IDLOCALITAT%TYPE;
 n_descripcioLocalitat LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
 n_descripcioProvincia PROVINCIA.DESCRIPCIOPROVINCIA%TYPE;
 n_descripcioPais PAIS.DESCRIPCIOPAIS%TYPE;
 BEGIN
  c_procesLog := 'PRC_CONSULTA_UNA_LOCALITAT';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Descripcio Localitat: ' || p_descripcioLocalitat;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('  CONSULTA UNA LOCALITAT    ');
  DBMS_OUTPUT.PUT_LINE('_____________________________');
  DBMS_OUTPUT.PUT_LINE(' Descripcio Localitat: ' || p_descripcioLocalitat);
-- Comprovem que  p_descripcioLocalitat no sigui NULL-----------------------------------------
 If  p_descripcioLocalitat IS NULL then
```

```
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('CONSULTA UNA LOCALITAT: ');
 s_rsp := 'Falta especificar LOCALITAT';
 DBMS_OUTPUT.put_line (s_rsp);
 RAISE e_localitat;
END IF;
--1er. Comprovem que existeixi la persona------------------------------
 SELECT COUNT (*) INTO n_registres
 FROM LOCALITAT
 WHERE LOCALITAT.DESCRIPCIOLOCALITAT= p_descripcioLocalitat;

 If n_registres=0 then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('CONSULTA UNA LOCALITAT: ');
  s_rsp := 'LOCALITAT no existent a la BBDD';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_localitat;
 END IF;
-- Recuperem les dades de la taula localitat-------------------------
 SELECT LOCALITAT.IDLOCALITAT,
       LOCALITAT.DESCRIPCIOLOCALITAT,
       PROVINCIA.DESCRIPCIOPROVINCIA,
       PAIS.DESCRIPCIOPAIS  INTO n_idLocalitat,n_descripcioLocalitat,
       n_descripcioProvincia,n_descripcioPais
 FROM LOCALITAT,PROVINCIA,PAIS
 WHERE LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
 AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS
 AND LOCALITAT.DESCRIPCIOLOCALITAT=p_descripcioLocalitat;

 DBMS_OUTPUT.PUT_LINE(s_sql);
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('CONSULTA UNA LOCALITAT:
  codi Localitat: '||n_idLocalitat||
 ' Descripcio:    '||n_descripcioLocalitat||
 ' Provincia:     '||n_descripcioProvincia||
 ' Pais:          '||n_descripcioPais);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
        s_rsp := 'OK: CONSULTA UNA LOCALITAT:
  codi Localitat: '||n_idLocalitat||
 ' Descripcio:    '||n_descripcioLocalitat||
 ' Provincia:     '||n_descripcioProvincia||
 ' Pais:          '||n_descripcioPais;
 DBMS_OUTPUT.put_line (s_rsp);
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);


COMMIT;
 EXCEPTION

  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
```

```
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

  END PRC_CONSULTA_UNA_LOCALITAT;
/*****************************************************************************
 Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
-- CONSULTA DE TOTES LES LOCALITATS AMB PROVINCIA I PAIS
*****************************************************************************/
PROCEDURE PRC_CONSULTA_LOCALITATS(
   s_rsp out NOCOPY VARCHAR2)
AS
  n_idLocalitat LOCALITAT.IDLOCALITAT%TYPE;
  n_descripcioLocalitat LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
  n_descripcioProvincia PROVINCIA.DESCRIPCIOPROVINCIA%TYPE;
  n_descripcioPais PAIS.DESCRIPCIOPAIS%TYPE;

CURSOR C_LOCALITAT IS
SELECT LOCALITAT.IDLOCALITAT AS CODI,
      LOCALITAT.DESCRIPCIOLOCALITAT AS LOCALITAT,
      PROVINCIA.DESCRIPCIOPROVINCIA AS PROVINCIA,
      PAIS.DESCRIPCIOPAIS AS PAIS
  INTO n_idLocalitat,n_descripcioLocalitat,
      n_descripcioProvincia,n_descripcioPais
FROM LOCALITAT,PROVINCIA,PAIS
WHERE LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS
AND LOCALITAT.IDLOCALITAT=LOCALITAT.IDLOCALITAT
ORDER BY PAIS.DESCRIPCIOPAIS,PROVINCIA.DESCRIPCIOPROVINCIA,
LOCALITAT.DESCRIPCIOLOCALITAT ;

BEGIN
   c_procesLog := 'PRC_CONSULTA_LOCALITATS';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='CONSULTA LOCALITATS ';
   c_sortidalog := 's_rsp';

  SELECT COUNT (*)  INTO n_registres
  FROM LOCALITAT
  WHERE LOCALITAT.IDLOCALITAT=LOCALITAT.IDLOCALITAT;
IF n_registres=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTA DE LOCALITATS: ');
   s_rsp := 'No hi han localitats donades d´alta ';
```

```
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_localitat;
  END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          CONSULTA DE LOCALITATS             ');
    DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------');
  OPEN C_LOCALITAT;
   FETCH C_LOCALITAT INTO n_idLocalitat,n_descripcioLocalitat,
      n_descripcioProvincia,n_descripcioPais;
  WHILE C_LOCALITAT%FOUND LOOP
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(
 ' Codi Localitat: '
 ||n_idLocalitat||
 ' Localitat: '
 ||n_descripcioLocalitat||
 ' Provincia: '
 ||n_descripcioProvincia||
 ' Pais: '
 ||n_descripcioPais);
  s_rsp := 'OK  CONSULTA LOCALITATS
  Codi Localitat: '
 ||n_idLocalitat||
 ' Descripcio:    '
 ||n_descripcioLocalitat||
 ' Provincia:     '
 ||n_descripcioProvincia||
 ' Pais:          '
 ||n_descripcioPais;

  FETCH C_LOCALITAT INTO n_idLocalitat,n_descripcioLocalitat,
      n_descripcioProvincia,n_descripcioPais;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total localitats: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_LOCALITAT;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi----------------------------------------
       s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi----------------------------------------
       s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line (s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
```

```
    ROLLBACK;

END PRC_CONSULTA_LOCALITATS;
END "GESTION_LOCALITAT";
```

10.6 TRATAMIENTO DE UBICACIONES.

DEFINICION	PROCEDIMIENTO
Alta de Ubicaciones	PRC_ALTA_UBICACIO
Modificar Ubicaciones	PRC_MODIFICAR_UBICACIO
Consulta de una Ubicacion	PRC_CONSULTAR_UNA_UBICACIO_CP
Consulta de Ubicaciones	PRC_CONSULTAR_UBICACIONS

```
CREATE OR REPLACE
PACKAGE       "GESTION_UBICACIO" AS
-- Alta Ubicacio
PROCEDURE PRC_ALTA_UBICACIO(
      p_idViaUbica in UBICACIO.IDVIAUBICACIO%TYPE,
      p_direccionUbica in UBICACIO.DIRECCIONUBICACIO%TYPE,
      p_numeroUbica in UBICACIO.NUMEROUBICACIO%TYPE,
      p_pisUbica in UBICACIO.PISUBICACIO%TYPE,
      p_portaUbica in UBICACIO.PORTAUBICACIO%TYPE,
      p_codiPostalUbica in UBICACIO.CODIPOSTAL%TYPE,
      p_idLocalitat in UBICACIO.IDLOCALITATUBICACIO%TYPE,
      s_rsp out NOCOPY VARCHAR2);

-- Modificar Ubicacio
PROCEDURE PRC_MODIFICAR_UBICACIO(
      p_idUbica IN UBICACIO.IDUBICA%TYPE,
      p_idViaUbica IN UBICACIO.IDVIAUBICACIO%TYPE,
      p_direccionUbica in UBICACIO.DIRECCIONUBICACIO%TYPE,
      p_numeroUbica in UBICACIO.NUMEROUBICACIO%TYPE,
      p_pisUbica in UBICACIO.PISUBICACIO%TYPE,
      p_portaUbica in UBICACIO.PORTAUBICACIO%TYPE,
      p_codiPostalUbica in UBICACIO.CODIPOSTAL%TYPE,
      p_idLocalitat in UBICACIO.IDLOCALITATUBICACIO%TYPE,
      s_rsp out NOCOPY VARCHAR2);
```

```
    --Consultar un Ubicacio

PROCEDURE PRC_CONSULTAR_UNA_UBICACIO_CP(
    p_cpUbica in UBICACIO.CODIPOSTAL%TYPE,
    s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_CONSULTAR_UBICACIONS(
    s_rsp out NOCOPY VARCHAR2);

END "GESTION_UBICACIO";
```

10.6.1 PROCEDIMIENTO SPL PACKAGE DE UBICACIONES.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_UBICACIO" AS
  c_procesLog LOG_TFC.procesLog%TYPE;
  c_dataHoraLog LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog LOG_TFC.entradaLog%TYPE;
  c_sortidaLog LOG_TFC.sortidaLog%TYPE;
  s_rsp LOG_TFC.rspLog%TYPE;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);

  n_descripcioVia VIA.DESCRIPCIOVIA%TYPE;
  n_direccioUbicacio UBICACIO.DIRECCIONUBICACIO%TYPE;
  n_numeroUbicacio UBICACIO.NUMEROUBICACIO%TYPE;
  n_pisUbicacio UBICACIO.PISUBICACIO%TYPE;
  n_portaUbicacio UBICACIO.PISUBICACIO%TYPE;
  n_codiPostalUbicacio UBICACIO.CODIPOSTAL%TYPE;
  n_localitatUbicacio LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
  n_provinciaUbicacio PROVINCIA.DESCRIPCIOPROVINCIA%TYPE;
  n_paisUbicacio PAIS.DESCRIPCIOPAIS%TYPE;
  n_idUbicacio UBICACIO.IDUBICA%TYPE;

  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_ubicacio    EXCEPTION;

/*****************************************************************************
ALTA UBICACIO
Procediment per donar altes d'ubicacions per poder assignar a client o fabricant
o centrsl o comptador. Totes aquelles taules que depenent d'una ubicaciÃ³
fÃsicament geografica.
*****************************************************************************/
PROCEDURE PRC_ALTA_UBICACIO(
      p_idViaUbica in UBICACIO.IDVIAUBICACIO%TYPE,
      p_direccionUbica in UBICACIO.DIRECCIONUBICACIO%TYPE,
      p_numeroUbica in UBICACIO.NUMEROUBICACIO%TYPE,
      p_pisUbica in UBICACIO.PISUBICACIO%TYPE,
      p_portaUbica in UBICACIO.PORTAUBICACIO%TYPE,
      p_codiPostalUbica in UBICACIO.CODIPOSTAL%TYPE,
```

```
    p_idLocalitat in UBICACIO.IDLOCALITATUBICACIO%TYPE,
    s_rsp out NOCOPY VARCHAR2)
  AS

  BEGIN
  c_procesLog := 'PRC_ALTA_UBICACIO';
  c_dataHoraLog := SYSDATE;
  c_entradaLog :='Codi Via Publica: ' ||p_idViaUbica ||
            ', Direccio : ' ||p_direccionUbica ||
            ', Numero: ' ||p_numeroUbica ||
            ', Pis: ' ||p_pisUbica ||
            ', Porta: ' ||p_portaUbica ||
            ', Codi Postal: ' ||p_codiPostalUbica  ||
            ', Codi Localitat: ' ||p_idLocalitat;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('      ALTA UBICACIONS      ');
  DBMS_OUTPUT.PUT_LINE('_____');
  DBMS_OUTPUT.PUT_LINE(' Codi Via Publica: ' ||p_idViaUbica ||
            ', Direccio : ' ||p_direccionUbica ||
            ', Numero: ' ||p_numeroUbica ||
            ', Pis: ' ||p_pisUbica ||
            ', Porta: ' ||p_portaUbica ||
            ', Codi Postal: ' ||p_codiPostalUbica  ||
            ', Codi Localitat: ' ||p_idLocalitat);
  c_sortidalog := 's_rsp';
If (p_idViaUbica IS NULL) then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('ALTA UBICACIONS : ');
  s_rsp := 'Falta especificar Codi Via Publica  ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_ubicacio ;
 END IF;
If (p_direccionUbica IS NULL) then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('ALTA UBICACIONS: ');
  s_rsp := 'Falta especificar direccion  ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_ubicacio ;
 END IF;
 If (p_numeroUbica IS NULL) then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA UBICACIONS: ');
    s_rsp := 'Falta especificar numero  ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_ubicacio ;
 END IF;
 If (p_codiPostalUbica  IS NULL) then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA UBICACIONS: ');
    s_rsp := 'Falta especificar codi postal  ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_ubicacio ;
```

```
 END IF;
 If (p_idLocalitat  IS NULL) then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA UBICACIONS: ');
    s_rsp := 'Falta especificar codi localitat  ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_ubicacio ;
 END IF;
-- Comprovem si existeix UBICACIO a  donar d'alta
 SELECT COUNT (*) INTO n_registres
 FROM UBICACIO
 WHERE UBICACIO.IDVIAUBICACIO=p_idViaUbica
      AND UBICACIO.DIRECCIONUBICACIO=p_direccionUbica
      AND UBICACIO.NUMEROUBICACIO=p_numeroUbica
      AND UBICACIO.PISUBICACIO=p_pisUbica
      AND UBICACIO.PORTAUBICACIO=p_portaUbica
      AND UBICACIO.CODIPOSTAL=p_codiPostalUbica
      AND UBICACIO.IDLOCALITATUBICACIO=p_idLocalitat;

 If n_registres=0 then
-- El donem d'alta UBICACIO
 INSERT INTO UBICACIO(IDVIAUBICACIO,
                      DIRECCIONUBICACIO,
                      NUMEROUBICACIO,
                      PORTAUBICACIO,
                      PISUBICACIO,
                      UBICACIO.CODIPOSTAL,
                      IDLOCALITATUBICACIO)
 VALUES(p_idViaUbica,
        p_direccionUbica,
        p_numeroUbica,
        p_pisUbica,
        p_portaUbica,
        p_codiPostalUbica,
        p_idLocalitat);

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('Insertada ubicacio:
   Codi via publica:'||p_idViaUbica||
   ' Direccio: '||p_direccionUbica||
   ' Numero: '||p_numeroUbica||
   ' Pis: '||p_pisUbica||
   ' Porta: '||p_portaUbica||
   ' Codi POstal: '||p_codiPostalUbica||
   ' Codi Localitat: '||p_idLocalitat);
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');

--Gravem en la taula log-----------------------------------------------------
   s_rsp :='Ok: Insertada ubicacio:
   Codi via publica:'||p_idViaUbica||
   ' Direccio: '||p_direccionUbica||
   ' Numero: '||p_numeroUbica||
```

```
    ' Pis: '||p_pisUbica||
    ' Porta: '||p_portaUbica||
    ' Codi POstal: '||p_codiPostalUbica||
    ' Codi Localitat: '||p_idLocalitat;
    pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    DBMS_OUTPUT.PUT_LINE(s_rsp);
  ELSE
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA UBICACIONS: ');
    s_rsp := 'Dades Ubicacio duplicades ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_ubicacio ;
  END IF;
  COMMIT;
  EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

  END PRC_ALTA_UBICACIO;
/*****************************************************************************
  NOM:       PRC_MODIFICAR_UBICACIO
  DESCRIPCIÃ“:
    Procediment encarregat de modificar les dades d'una UBICACIO
    Les dades es modificaran en la taula UBICACIO i el identificador de
    tipus UBICACIO apuntarÃ  a la clau primaria del codi de la taula
    UBICACIO.
*****************************************************************************/
PROCEDURE PRC_MODIFICAR_UBICACIO(
      p_idUbica IN UBICACIO.IDUBICA%TYPE,
      p_idViaUbica IN UBICACIO.IDVIAUBICACIO%TYPE,
      p_direccionUbica in UBICACIO.DIRECCIONUBICACIO%TYPE,
      p_numeroUbica in UBICACIO.NUMEROUBICACIO%TYPE,
      p_pisUbica in UBICACIO.PISUBICACIO%TYPE,
      p_portaUbica in UBICACIO.PORTAUBICACIO%TYPE,
      p_codiPostalUbica in UBICACIO.CODIPOSTAL%TYPE,
      p_idLocalitat in UBICACIO.IDLOCALITATUBICACIO%TYPE,
      s_rsp out NOCOPY VARCHAR2)
  AS
  BEGIN
  c_procesLog := 'PRC_MODIFICAR_UBICACIO';
  c_dataHoraLog := SYSDATE;
```

```
  c_entradaLog := ' Codi Ubicacio: ' ||p_idUbica||
            '  Codi Via Publica: ' ||p_idViaUbica ||
            '  Direccio : ' ||p_direccionUbica ||
            '  Numero: ' ||p_numeroUbica ||
            '  Pis: ' ||p_pisUbica ||
            '  Porta: ' ||p_portaUbica ||
            '  Codi Postal: ' ||p_codiPostalUbica  ||
            '  Codi Localitat: ' ||p_idLocalitat;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('    MODIFICAR UBICACIO     ');
  DBMS_OUTPUT.PUT_LINE('_____________________________');
  DBMS_OUTPUT.PUT_LINE('Codi Ubicacio:  '||p_idUbica ||
            '  Codi Via Publica: ' ||p_idViaUbica ||
            '  Direccio : ' ||p_direccionUbica ||
            '  Numero: ' ||p_numeroUbica ||
            '  Pis: ' ||p_pisUbica ||
            '  Porta: ' ||p_portaUbica ||
            '  Codi Postal: ' ||p_codiPostalUbica  ||
            '  Codi Localitat: ' ||p_idLocalitat);
  c_sortidalog := 's_rsp';
  -- ComprovaciÃ³ del identificador Tipus Funcions de Central-----
 If p_idUbica IS NULL then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR UBICACIO: ');
  s_rsp := 'Falta especificar Codi  ubicacio. ';
  DBMS_OUTPUT.put_line(s_rsp);
  RAISE e_ubicacio ;
 END IF;

If p_idViaUbica IS NULL then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR UBICACIO: ');
  s_rsp := 'Falta especificar Codi Via Publica.   ';
  DBMS_OUTPUT.put_line(s_rsp);
  RAISE e_ubicacio;
 END IF;
 If p_direccionUbica IS NULL then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR UBICACIO: ');
  s_rsp := 'Falta especificar direccion.  ';
  DBMS_OUTPUT.put_line(s_rsp);
  RAISE e_ubicacio;
 END IF;
If p_numeroUbica IS NULL then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR UBICACIO: ');
  s_rsp := 'Falta especificar numero.  ';
  DBMS_OUTPUT.put_line(s_rsp);
  RAISE e_ubicacio;
 END IF;
 If p_codiPostalUbica  IS NULL then
```

```
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UBICACIO: ');
    s_rsp := 'Falta especificar codi postal.  ';
    DBMS_OUTPUT.put_line(s_rsp);
    RAISE e_ubicacio;
  END IF;
  If p_idLocalitat  IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UBICACIO: ');
    s_rsp := 'Falta especificar codi localitat UBICACIO ';
    DBMS_OUTPUT.put_line(s_rsp);
    RAISE e_ubicacio;
  END IF;
-- cerquem si existeix
  SELECT COUNT (*) INTO n_registres
  FROM UBICACIO
  WHERE UBICACIO.IDUBICA=p_idUbica;

  If n_registres=0 then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UBICACIO: ');
    s_rsp := 'UBICACIO no existeix a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_ubicacio;
  END IF;
-- recuperem el codi UBICACIO
  SELECT UBICACIO.IDUBICA INTO n_idUbicacio
  FROM UBICACIO
  WHERE UBICACIO.IDUBICA=p_idUbica;

  s_sql := 'UPDATE UBICACIO SET ';

-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF p_idViaUbica IS NOT NULL THEN
-- Modifiquem el descripcio UBICACIO
    s_sql := s_sql || 'IDVIAUBICACIO=''' || p_idViaUbica|| ''',';
  END IF;
  IF p_direccionUbica IS NOT NULL THEN
   s_sql := s_sql || 'DIRECCIONUBICACIO=''' || p_direccionUbica|| ''',';
  END IF;
  IF p_numeroUbica IS NOT NULL THEN
    s_sql := s_sql || 'NUMEROUBICACIO=''' || p_numeroUbica || ''',';
  END IF;
  IF p_pisUbica IS NOT NULL THEN
    s_sql := s_sql || 'PISUBICACIO=''' || p_pisUbica|| ''',';
  END IF;
  IF p_portaUbica IS NOT NULL THEN
     s_sql := s_sql || 'PORTAUBICACIO=''' || p_portaUbica|| ''',';
  END IF;
  IF p_codiPostalUbica  IS NOT NULL THEN
    s_sql := s_sql || 'CODIPOSTAL=''' || p_codiPostalUbica || ''',';
  END IF;
```

```
 IF p_idLocalitat  IS NOT NULL THEN
   s_sql := s_sql || 'IDLOCALITATUBICACIOt='' || p_idLocalitat || '','';
 END IF;
-- Eliminem la coma final de la sentencia SQL
 s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃ³ del WHERE
 s_sql := s_sql || ' WHERE IDUBICA='' ||n_idUbicacio|| ''';

 EXECUTE IMMEDIATE s_sql;

 IF SQL%ROWCOUNT = 0  THEN
--L'error serÃ  que el la via ja el tenim en la taula i no admet duplicats--
   s_rsp := 'Actualitzacio no realitzada. ';
   RAISE e_ubicacio;
 ELSE
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('Modificacio feta en Ubicacio:
                Codi Via Publica: '||p_idViaUbica||
               '  Direccio : ' ||p_direccionUbica ||
               '  Numero: ' ||p_numeroUbica ||
               '  Pis: ' ||p_pisUbica ||
               '  Porta: ' ||p_portaUbica ||
               '  Codi Postal: ' ||p_codiPostalUbica  ||
               '  Codi Localitat: ' ||p_idLocalitat);
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
             s_rsp := 'OK Modificacio feta en Ubicacio:
                Codi Via Publica: '||p_idViaUbica||
               '  Direccio : ' ||p_direccionUbica ||
               '  Numero: ' ||p_numeroUbica ||
               '  Pis: ' ||p_pisUbica ||
               '  Porta: ' ||p_portaUbica ||
               '  Codi Postal: ' ||p_codiPostalUbica  ||
               '  Codi Localitat: ' ||p_idLocalitat;
   pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    s_rsp := 'Error: Ubicacio duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
```

```
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

 END PRC_MODIFICAR_UBICACIO;
/*****************************************************************************
CONSULTA UNA UBICACIO segons la direcciÃ³ i el codi de poblacio
*****************************************************************************/
PROCEDURE PRC_CONSULTAR_UNA_UBICACIO_CP(
    p_cpUbica in UBICACIO.CODIPOSTAL%TYPE,
   s_rsp out NOCOPY VARCHAR2)
  AS

CURSOR C_CP IS
SELECT UBICACIO.IDUBICA AS CODI,
    VIA.DESCRIPCIOVIA AS VIA_PUBLICA,
    UBICACIO.DIRECCIONUBICACIO AS DIRECCIO,
    UBICACIO.NUMEROUBICACIO AS NUMERO,
    UBICACIO.PISUBICACIO AS PIS,
    UBICACIO.PORTAUBICACIO AS PORTA,
    UBICACIO.CODIPOSTAL AS CODI_POSTAL,
    LOCALITAT.DESCRIPCIOLOCALITAT AS POBLACIO,
    PROVINCIA.DESCRIPCIOPROVINCIA AS PROVINCIA,
    PAIS.DESCRIPCIOPAIS AS PAIS
FROM UBICACIO,VIA,LOCALITAT,PROVINCIA,PAIS
WHERE
VIA.IDVIA=UBICACIO.IDVIAUBICACIO
AND UBICACIO.IDUBICA=UBICACIO.IDUBICA
AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT
AND LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS
AND UBICACIO.CODIPOSTAL= p_cpUbica
ORDER BY UBICACIO.CODIPOSTAL;
   BEGIN
   c_procesLog := 'PRC_CONSULTAR_UBICACIO_CP';
   c_dataHoraLog := SYSDATE;
   c_entradaLog := ' Codi Postal : ' ||p_cpUbica ;
   c_sortidalog := 's_rsp';
   -- Comprovem que  dades codi postal
   SELECT DISTINCT COUNT(UBICACIO.CODIPOSTAL)INTO n_registres
   FROM UBICACIO
   WHERE UBICACIO.CODIPOSTAL=p_cpUbica
   AND UBICACIO.IDUBICA=UBICACIO.IDUBICA;

 If  n_registres=0  then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTA UBICACIO PER CODI POSTAL:  ');
   s_rsp := 'Les dades no existeixen en la BDD.  ';
   DBMS_OUTPUT.put_line(s_rsp);
   RAISE  e_ubicacio;
```

```
 END IF;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('          CONSULTA UBICACIO                    ');
 DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------');
OPEN C_CP;
  FETCH C_CP INTO n_idUbicacio,n_descripcioVia,n_direccioUbicacio,
  n_numeroUbicacio,n_pisUbicacio,n_portaUbicacio,
  n_codiPostalUbicacio,n_localitatUbicacio,
  n_provinciaUbicacio,n_paisUbicacio;
WHILE C_CP%FOUND LOOP
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(
' Codi Ubicacio: '
||n_idUbicacio||
'   Via Publica: '
||n_descripcioVia||
'   Direccio: '
||n_direccioUbicacio||
'   Numero: '
|| n_numeroUbicacio||
'   Pis: '
||n_pisUbicacio||
'   Porta: '
||n_portaUbicacio||
'   Codi Postal: '
||n_codiPostalUbicacio||
'   Poblacio: '
||n_localitatUbicacio||
'   Provincia: '
|| n_provinciaUbicacio||
'   Pais: '
||n_paisUbicacio);
 s_rsp := 'OK  CONSULTA UBICACIO
  Codi Ubicacio: '
||n_idUbicacio||
'    Via Publica: '
||n_descripcioVia||
'   Direccio: '
||n_direccioUbicacio||
'   Numero: '
|| n_numeroUbicacio||
'   Pis: '
||n_pisUbicacio||
'   Porta: '
||n_portaUbicacio||
'   Codi Postal: '
||n_codiPostalUbicacio||
'   Poblacio: '
||n_localitatUbicacio||
'   Provincia: '
|| n_provinciaUbicacio||
'   Pais: '
```

```
 ||n_paisUbicacio;

   FETCH C_CP INTO n_idUbicacio,n_descripcioVia,n_direccioUbicacio,
   n_numeroUbicacio,n_pisUbicacio,n_portaUbicacio,
   n_codiPostalUbicacio,n_localitatUbicacio,
   n_provinciaUbicacio,n_paisUbicacio;

  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total ubicacions: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_CP;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
      ELSE
         -- L'error si ha estat controlat per codi
         s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('  ');
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;

END PRC_CONSULTAR_UNA_UBICACIO_CP;

PROCEDURE PRC_CONSULTAR_UBICACIONS(
       s_rsp out NOCOPY VARCHAR2)
AS
  CURSOR C_CP IS
SELECT VIA.DESCRIPCIOVIA AS VIA_PUBLICA,
      UBICACIO.DIRECCIONUBICACIO AS DIRECCIO,
      UBICACIO.NUMEROUBICACIO AS NUMERO,
      UBICACIO.PISUBICACIO AS PIS,
      UBICACIO.PORTAUBICACIO AS PORTA,
      UBICACIO.CODIPOSTAL AS CODI_POSTAL,
      LOCALITAT.DESCRIPCIOLOCALITAT AS POBLACIO,
      PROVINCIA.DESCRIPCIOPROVINCIA AS PROVINCIA,
      PAIS.DESCRIPCIOPAIS AS PAIS
FROM UBICACIO,VIA,LOCALITAT,PROVINCIA,PAIS
WHERE
VIA.IDVIA=UBICACIO.IDVIAUBICACIO
AND UBICACIO.IDUBICA=UBICACIO.IDUBICA
AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT
AND LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS
```

```
AND UBICACIO.CODIPOSTAL=UBICACIO.CODIPOSTAL
ORDER BY UBICACIO.CODIPOSTAL;
   BEGIN
   c_procesLog := 'PRC_CONSULTAR_UBICACIOS';
   c_dataHoraLog := SYSDATE;
   c_entradaLog := ' CONSULTA UBICACIONS ';
   c_sortidalog := 's_rsp';
   -- Comprovem que  dades codi postal
   SELECT DISTINCT COUNT(UBICACIO.CODIPOSTAL)INTO n_registres
   FROM UBICACIO
   WHERE UBICACIO.CODIPOSTAL=UBICACIO.CODIPOSTAL
   AND UBICACIO.IDUBICA=UBICACIO.IDUBICA;
    If  n_registres=0  then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTA UBICACIONS:  ');
   s_rsp := 'Les dades no existeixen en la BDD.  ';
   DBMS_OUTPUT.put_line(s_rsp);
   RAISE  e_ubicacio;
 END IF;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('          CONSULTA UBICACIONS                    ');
 DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------');
OPEN C_CP;
   FETCH C_CP INTO n_descripcioVia,n_direccioUbicacio,
   n_numeroUbicacio,n_pisUbicacio,n_portaUbicacio,
   n_codiPostalUbicacio,n_localitatUbicacio,
   n_provinciaUbicacio,n_paisUbicacio;
 WHILE C_CP%FOUND LOOP
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(
'   Via Publica: '
||n_descripcioVia||
'   Direccio: '
||n_direccioUbicacio||
'   Numero: '
|| n_numeroUbicacio||
'   Pis: '
||n_pisUbicacio||
'   Porta: '
||n_portaUbicacio||
'   Codi Postal: '
||n_codiPostalUbicacio||
'   Poblacio: '
||n_localitatUbicacio||
'   Provincia: '
|| n_provinciaUbicacio||
'   Pais: '
||n_paisUbicacio);
 s_rsp := 'OK  CONSULTA UBICACIONS
    Via Publica: '
||n_descripcioVia||
'   Direccio: '
```

```
||n_direccioUbicacio||
'   Numero: '
|| n_numeroUbicacio||
'   Pis: '
||n_pisUbicacio||
'   Porta: '
||n_portaUbicacio||
'   Codi Postal: '
||n_codiPostalUbicacio||
'   Poblacio: '
||n_localitatUbicacio||
'   Provincia: '
|| n_provinciaUbicacio||
'   Pais: '
||n_paisUbicacio;
 FETCH C_CP INTO n_descripcioVia,n_direccioUbicacio,
  n_numeroUbicacio,n_pisUbicacio,n_portaUbicacio,
  n_codiPostalUbicacio,n_localitatUbicacio,
  n_provinciaUbicacio,n_paisUbicacio;
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);  END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total ubicacions: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_CP;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  END PRC_CONSULTAR_UBICACIONS;
END "GESTION_UBICACIO";
```

10.7 TRATAMIENTO DE PERSONAS.

DEFINICION	PROCEDIMIENTO
Alta de Localidades	PRC_ALTA_LOCALITAT
Modificar Localidades	PRC_MODIFICAR_LOCALITAT
Consulta de una Localidades	PRC_CONSULTA_UNA_LOCALITAT
Consulta de Localidades	PRC_CONSULTA_LOCALITATS

```
create or replace PACKAGE        "GESTION_PERSONA" AS
PROCEDURE PRC_ALTA_PERSONA(
  p_descripcioPersona in PERSONA.DESCRIPCIOPERSONA%TYPE,
  s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_MODIFICAR_PERSONA(
  p_descripcioPersona in PERSONA.DESCRIPCIOPERSONA%TYPE,
  p_idPersona in PERSONA.IDPERSONA%TYPE,
  s_rsp out NOCOPY VARCHAR2);

procedure PRC_CONSULTAR_PERSONA(
   s_rsp out NOCOPY VARCHAR2);
END GESTION_PERSONA;
```

10.7.1 PROCEDIMIENTO SPL PACKAGE DE PERSONAS.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_PERSONA" AS
  c_procesLog LOG_TFC.procesLog%TYPE;
  c_dataHoraLog LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog LOG_TFC.entradaLog%TYPE;
  c_sortidaLog LOG_TFC.sortidaLog%TYPE;
  s_rsp LOG_TFC.rspLog%TYPE;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);
```

```
 n_descripcioPersona PERSONA.DESCRIPCIOPERSONA%TYPE;
 n_idPersona   PERSONA.IDPERSONA%TYPE;
 n_NUM_ERR NUMBER(10);
 sortida varchar2(500):='';
 e_persona     EXCEPTION;
-----------------------------------------------------------------
-- create procedure for table "LOG_TFC i PERSONA"
/*************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
            PROCEDIMENT DONAR ALTA PERSONA
DESCRIPCIO:
Es un procediement per poder donar d'alta els tipus de persones  que el seu
rol dins de sistema i el negoci forman part del conjunt de CLIENTS.
S'ha fet estandard pel motiu de que pugui ser reutilitzat per altres
entitats en cas de una millora del sistema que precisi el desenvolupament
de la fase de l'aplicaciÃ³ al usuari final.
*************************************************************************/
  procedure PRC_ALTA_PERSONA(
   p_descripcioPersona in PERSONA.DESCRIPCIOPERSONA%TYPE,
   s_rsp out NOCOPY VARCHAR2)
  AS

  BEGIN
   c_procesLog := 'PRC_ALTA_PERSONA';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Decripcio Persona: ' || p_descripcioPersona;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('     CLASSE DE CLIENT     ');
    DBMS_OUTPUT.PUT_LINE('_____________________');
   c_sortidalog := 's_rsp';
   If (p_descripcioPersona IS NULL) then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('ALTA_PERSONA: ');
         s_rsp := 'Falta especificar persona';
        RAISE e_persona;
    END IF;
     -- Comprovem si existeix el NIF a  donar d'alta
       SELECT COUNT (*)INTO n_registres
          FROM PERSONA
          WHERE DESCRIPCIOPERSONA =p_descripcioPersona;
     If n_registres=0 then
        INSERT INTO PERSONA(descripcioPersona)
          VALUES(p_descripcioPersona);
          --Gravem en la taula log-----------------------------------------------------
          DBMS_OUTPUT.ENABLE;
            DBMS_OUTPUT.PUT_LINE('Insertada Classe de client: '||p_descripcioPersona);
            DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
          s_rsp :='Ok: En la Taula PERSONA Insertada persona o classe de client:  '|| p_descripcioPersona;
          pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
          DBMS_OUTPUT.PUT_LINE(s_rsp);
        ELSE
```

```
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('ALTA_PERSONA: ');
       s_rsp := 'DESCRICIO Persona  duplicat';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_persona;
    END IF;
  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

  end PRC_ALTA_PERSONA;

 PROCEDURE PRC_MODIFICAR_PERSONA(
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
  NOM:       PRC_MODIFICAR_PERSONA
  DESCRIPCIÃ“:
            Procediment encarregat de modificar les dades d'un pais.
            Les dades es modificaran en la taula persona i el identificador de
            Via apuntarÃ  a la clau primaria del contacte de la taula Pais.
*****************************************************************************/
      p_descripcioPersona in PERSONA.DESCRIPCIOPERSONA%TYPE,
      p_idPersona in PERSONA.IDPERSONA%TYPE,
       s_rsp out NOCOPY VARCHAR2)
  AS

  BEGIN
   c_procesLog := 'PRC_MODIFICAR_PERSONA';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Decripcio Persona: ' || p_descripcioPersona;
       DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('  MODIFICAR CLASSE DE CLIENT  ');
    DBMS_OUTPUT.PUT_LINE('_____________________________');
   DBMS_OUTPUT.PUT_LINE('Nom de la Via Publica: '|| p_descripcioPersona);
   c_sortidalog := 's_rsp';
   -- ComprovaciÃ³ del identificador CLASSE CLIENT-----

    If  (p_descripcioPersona IS NULL) THEN
       DBMS_OUTPUT.ENABLE;
```

```
        DBMS_OUTPUT.PUT_LINE('MODIFICAR_PERSONA: ');
              s_rsp := 'Falta especificar el persona';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
         RAISE e_persona;
       ELSE

         SELECT COUNT (*)INTO n_registres
         FROM PERSONA
         WHERE DESCRIPCIOPERSONA = p_descripcioPersona;
        If n_registres=0 then
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICAR_PERSONA: ');
          s_rsp := 'PERSONA no existeix a la BBDD';
          DBMS_OUTPUT.PUT_LINE(s_rsp);
           RAISE e_persona;
         END IF;

      -- recuperem el codi de la persona
 SELECT PERSONA.IDPERSONA,PERSONA.DESCRIPCIOPERSONA INTO n_idPersona,n_descripcioPersona
 FROM PERSONA
 WHERE DESCRIPCIOPERSONA=p_descripcioPersona;

      s_sql := 'UPDATE PERSONA SET ';

   -- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
        IF p_descripcioPersona IS NOT NULL THEN
             -- Modifiquem el descripcio persona
       s_sql := s_sql || 'descripcioPersona=''' || p_descripcioPersona || ''',';
       END IF;

          -- Eliminem la coma final de la sentencia SQL
          s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
          -- Afegim a la sentencia la condiciÃ³ del WHERE
          s_sql := s_sql || ' WHERE idPersona=''' || n_idPersona|| '''';

     EXECUTE IMMEDIATE s_sql;

          IF SQL%ROWCOUNT = 0
          THEN
--L'error serÃ  que  la PERSONA ja el tenim en la taula i no admet duplicats--
            DBMS_OUTPUT.ENABLE;
            DBMS_OUTPUT.PUT_LINE('MODIFICAR_PERSONA: ');
            s_rsp := 'ActualitzaciÃ³ no realitzada';
            DBMS_OUTPUT.PUT_LINE(s_rsp);
            RAISE e_persona;
          ELSE
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('Modificacio feta de la Classe de Client
    : codi: '|| n_idPersona||
  ' Descripcio: '|| p_descripcioPersona);
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   s_rsp := 'OK Modificacio feta de la PERSONA: codi: '||n_idPersona||
```

```
          ' Descripcio: '||p_descripcioPersona;
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;
END IF;
COMMIT;
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    s_rsp := 'Error: Pais duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  END PRC_MODIFICAR_PERSONA;

/*******************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
  NOM:       CONSULTAR_PERSONA
  DESCRIPCIÃ“:
            Procediment encarregat de consultar les dades d'una persona.
            Les dades a consultar en la taula persona i el identificador de
            persona apuntarÃ  a la clau primaria  de la taula Persona.
*******************************************************************************/

 procedure PRC_CONSULTAR_PERSONA(
   s_rsp out NOCOPY VARCHAR2)
AS

  CURSOR C_PERSONA IS
  SELECT PERSONA.IDPERSONA,PERSONA.DESCRIPCIOPERSONA
  FROM PERSONA
  WHERE PERSONA.IDPERSONA=PERSONA.IDPERSONA;
  e_persona EXCEPTION;
BEGIN
   c_procesLog := 'PRC_CONSULTA_PERSONA';
    c_dataHoraLog := SYSDATE;
```

```
    c_entradaLog :='CONSULTA DE CLASSE DE CLIENT';
   c_sortidalog := 's_rsp';
-- Comprovem que p_descripcioPersona no sigui NULL----------------------------------------------
    --1er. Comprovem que existeixi la persona------------------------------
     SELECT COUNT (*) INTO n_registres
     FROM PERSONA
     WHERE  PERSONA.IDPERSONA=PERSONA.IDPERSONA;

   If n_registres=0 then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTAR_PERSONA: ');
   s_rsp := 'Classe de Client  no existent a la BBDD';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_persona;
   END IF;

DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE('          CONSULTA CLASSR DE CLIENT          ');
DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------');
-- Recuperem les dades de la taula PERSONA-------------------------

 OPEN C_PERSONA;
    FETCH C_PERSONA INTO n_idPersona ,n_descripcioPersona;
 WHILE C_PERSONA%FOUND LOOP
  DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(
'   Codi Classe Client '
||n_idPersona
||' Descripcio '
|| n_descripcioPersona);
 s_rsp := 'OK  CONSULTA PERSONA
 Codi Persona:  '
||n_idPersona
||' Descripcio: '
|| n_descripcioPersona;

 FETCH C_PERSONA INTO n_idPersona,n_descripcioPersona;

 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total Classes de Clients: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_PERSONA;
COMMIT;
 EXCEPTION

  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
```

```
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

  END PRC_CONSULTAR_PERSONA;
END "GESTION_PERSONA";
```

11 APENDICE 8 MODUL PRODUCCION.

Modulo de producción, conjunto de entidades que controlan las funcionalidades del proyecto.

PACKAGE_GESTIO_CENTRALS.sql

PACKAGE_GESTIO_CLASSE_CENTRAL.sql

PACKAGE_GESTIO_COMPTADOR.sql

PACKAGE_GESTIO_CONECTA_CENTRALS.sql

PACKAGE_GESTIO_CONECTA_COMPTADORS.sql

PACKAGE_GESTIO_HIST_LECTURES.sql

PACKAGE_GESTIO_INSPECCIONS.sql

PACKAGE_GESTIO_LECTURES.sql

PACKAGE_GESTIO_LINEA_TIPUS.sql

PACKAGE_GESTIO_MODEL.sql

PACKAGE_GESTIO_TIPO_CENTRAL.sql

PACKAGE_GESTIO_TIPUS_FUNCIONS.sql

PACKAGE_GESTIO_TIPUS_INSPECCIONS.sql

PACKAGE_GESTIO_TIPUS_LECTURA.sql GESTIO_CENTRAL.sql

GESTIO_CLASSE_CENTRAL.sql

GESTIO_COMPTADOR.sql

GESTIO_CONECTA_CENTRALS.sql

GESTIO_CONECTAR_COMPTADOR.sql

GESTIO_HIST_LECTURES.sql

GESTIO_INSPECCIONS.sql

GESTIO_LECTURES.sql

GESTIO_LINEA_TIPUS.sql

GESTIO_MODEL.sql

GESTIO_TIPUS_CENTRALS.sql

GESTIO_TIPUS_FUNCIONS.sql

GESTIO_TIPUS_INSPECCIONS.sql

GESTIO_TIPUS_LECTURA.sql

11.1 TRATAMIENTO DE GESTION TIPUS LECTURA.

Gestiona el tipo de lectura si es Presencial o bien Telemática.

DEFINICION	PROCEDIMIENTO
Alta de Tipo de Lectura	PRC_ALTA_TIPO_LECTURA
Modificar el Tipo de Lectura	PRC_MODIFICAR_TIPO_LECTURA
Consulta el Tipo de Lectura	PRC_CONSULTAR_TIPO_LECTURA

```
CREATE OR REPLACE
PACKAGE        "GESTION_TIPO_LECTURA" AS
  PROCEDURE PRC_ALTA_TIPO_LECTURA(
    p_descripcioTLectura in TIPO_LECTURA.DESCRIPCIOLECTURA%TYPE,
    s_rsp out NOCOPY VARCHAR2);
  PROCEDURE PRC_MODIFICAR_TIPO_LECTURA(
    p_descripcioTLectura in TIPO_LECTURA.DESCRIPCIOLECTURA%TYPE,
    p_idTLectura in TIPO_LECTURA.IDTIPOLECTURA%TYPE,
    s_rsp out NOCOPY VARCHAR2);
procedure PRC_CONSULTAR_TIPO_LECTURA(
    s_rsp out NOCOPY VARCHAR2);
END GESTION_TIPO_LECTURA;
```

11.1.1 PROCEDIMIENTO SPL PACKAGE TIPUS LECTURA.

```
CREATE OR REPLACE PACKAGE BODY        "GESTION_TIPO_CENTRAL" AS
  n_registres     NUMBER;
  s_sql         VARCHAR2 (2000);
  n_NUM_ERR          NUMBER(10);
  c_procesLog          LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp      LOG_TFC.RSPLOG%TYPE;
  n_idTipoCentral TIPO_CENTRAL.IDTIPOCENTRAL%TYPE;
  n_descripcioTipoCentral TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL%TYPE;
  n_idFuncionCentral  TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL%TYPE;
  n_quantitatTipoCentral TIPO_CENTRAL.QUANTITATTIPOCENTRAL%TYPE;
  n_estatTiposCentral TIPO_CENTRAL.ESTATTIPOCENTRAL%TYPE;
  n_dataEstatTipoCentral TIPO_CENTRAL.DATAESTATTIPOCENTRAL%TYPE;
  n_observacioTiposCentral TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL%TYPE;
  n_estat ESTAT.DESCRIPCIOESTAT%TYPE;
  n_funcions TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE;

  n_descEstat ESTAT.DESCRIPCIOESTAT%TYPE;
  n_funcio TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE;

  sortida VARCHAR2(1000):='';
  e_tipo_central   EXCEPTION;
/*************************************************************************
Autor: Eduard Monzonis Hierro                         UOC
18/05/2012                                TFC: CONTROL ENERGIA.
          PROCEDIMENT PCR_ALTA_TIPO_CENTRAL
Procediment per donar d'alta el tipus de central de produccIÃ³ d'energia.
*************************************************************************/
PROCEDURE PRC_ALTA_TIPO_CENTRAL(
   p_descripcioTipoCentral in TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL%TYPE,
   p_idFuncionCentral in TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL%TYPE,
   p_quantitatTipoCentral in TIPO_CENTRAL.QUANTITATTIPOCENTRAL%TYPE,
   p_estatTiposCentral in TIPO_CENTRAL.ESTATTIPOCENTRAL%TYPE,
   p_dataEstatTipoCentral in TIPO_CENTRAL.DATAESTATTIPOCENTRAL%TYPE,
   p_observacioTiposCentral in TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL%TYPE,
   s_rsp out NOCOPY VARCHAR2)
   AS

   BEGIN
    c_procesLog := 'PCR_ALTA_TIPO_CENTRAL';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :=
    'Descripcio Tipo Central: ' || p_descripcioTipoCentral||
    ', Id Funcion Central: ' || p_idFuncionCentral||
    ', Quantitat Tipo Central: ' || p_quantitatTipoCentral||
    --', Estat Tipos Central: ' || p_estatTiposCentral||
    ', Data Estat Tipo Central: ' || p_dataEstatTipoCentral||
```

```
', Observacio Tipos Central: ' || p_observacioTiposCentral;
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(' ');
DBMS_OUTPUT.PUT_LINE('      ALTA TIPUS DE CENTRALS       ');
DBMS_OUTPUT.PUT_LINE('---------------------------------------');
DBMS_OUTPUT.PUT_LINE(' Descripcio Tipo Central: ' || p_descripcioTipoCentral||
', Id Funcion Central: ' || p_idFuncionCentral||
', Quantitat Tipo Central: ' || p_quantitatTipoCentral||
--', Estat Tipos Central: ' || p_estatTiposCentral||
', Data Estat Tipo Central: ' || p_dataEstatTipoCentral||
', Observacio Tipos Central: ' || p_observacioTiposCentral);
c_sortidalog := 's_rsp';

n_estatTiposCentral:=1;

SELECT ESTAT.DESCRIPCIOESTAT INTO n_estat
FROM ESTAT
WHERE ESTAT.IDESTAT=n_estatTiposCentral;

SELECT TIPUS_FUNCIONS.DESCRIPCIONFUNCIO INTO n_funcions
FROM TIPUS_FUNCIONS
WHERE TIPUS_FUNCIONS.IDFUNCIO=p_idFuncionCentral;

 If p_descripcioTipoCentral IS NULL then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE CENTRALS: ');
      s_rsp := 'Falta especificar Tipo Central';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_tipo_central;
 END IF;
If (p_idFuncionCentral IS NULL) OR (p_idFuncionCentral=0) OR (p_idFuncionCentral<0) then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE CENTRALS: ');
      s_rsp := 'Falta especificar Id Funcion CentraL. ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_tipo_central;
 END IF;
 If (p_quantitatTipoCentral IS NULL OR p_quantitatTipoCentral<0) then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE CENTRALS:  ');
      s_rsp := 'Falta especificar Quantitat Tipo Central o el vaLor es mes petit que 0 ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_tipo_central;
 END IF;
-- Comprovem si existeix el el tipus de lectura a  donar d'alta
 SELECT COUNT (*)INTO n_registres
 FROM TIPO_CENTRAL
 WHERE DESCRIPCIOTIPOCENTRAL = p_descripcioTipoCentral
 AND TIPO_CENTRAL.ESTATTIPOCENTRAL=n_estatTiposCentral;

If n_registres=0 then
-- El donem d'alta a la tipus lectura
```

```
INSERT INTO TIPO_CENTRAL (DESCRIPCIOTIPOCENTRAL,
                IDFUNCIOTIPOCENTRAL,
                QUANTITATTIPOCENTRAL,
                ESTATTIPOCENTRAL,
                DATAESTATTIPOCENTRAL,
                OBSERVACIONSTIPOCENTRAL)
        VALUES(p_descripcioTipoCentral,
            p_idFuncionCentral,
            p_quantitatTipoCentral,
            n_estatTiposCentral,
            p_dataEstatTipoCentral,
            p_observacioTiposCentral);
        --Gravem en la taula log----------------------------------------------------

 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Insertada satisfatoriament el tipus de
 Central '||p_descripcioTipoCentral||
 ' Data alta:'||p_dataEstatTipoCentral||
 ' Estat: '||n_estat||
  ' Unitats: '||p_quantitatTipoCentral||
 ' Cacteristiques: '||n_funcions||
 ' Observacions:'||p_observacioTiposCentral);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
 s_rsp :='Ok: Insertada satisfatoriament el tipus de
 Central '||p_descripcioTipoCentral||
 ' Data alta:'||p_dataEstatTipoCentral||
 ' Estat: '||n_estat||
 ' Unitats: '||p_quantitatTipoCentral||
 ' Cacteristiques: '||n_funcions||
 ' Observacions:'||p_observacioTiposCentral;
 pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 ELSE
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE CENTRALS: ');
   s_rsp := 'DESCRICIO Tipus Central duplicat';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_tipo_central;
 END IF;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA_TIPO_CENTRAL:  ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
```

```
    ROLLBACK;

 END PRC_ALTA_TIPO_CENTRAL;
/**********************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
        PROCEDIMENT PCR_MODIFICA_TIPO_CENTRAL
Procediment per modificar el tipus de central de producciÃ³ d'energia.
**********************************************************************/
procedure PRC_MODIFICAR_TIPO_CENTRAL(
  p_idTipoCentral in TIPO_CENTRAL.IDTIPOCENTRAL%TYPE,
  p_descripcioTipoCentral in TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL%TYPE,
  p_idFuncionCentral in TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL%TYPE,
  p_quantitatTipoCentral in TIPO_CENTRAL.QUANTITATTIPOCENTRAL%TYPE,
  p_estatTiposCentral in TIPO_CENTRAL.ESTATTIPOCENTRAL%TYPE,
  p_dataEstatTipoCentral in TIPO_CENTRAL.DATAESTATTIPOCENTRAL%TYPE,
  p_observacioTiposCentral in TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL%TYPE,
  s_rsp out NOCOPY VARCHAR2)
  AS

  BEGIN
   c_procesLog := 'PCR_MODIFICA_TIPO_CENTRAL';
   c_dataHoraLog := SYSDATE;
   c_entradaLog := 'Id Tipo Central Tipo Central: ' ||p_idTipoCentral||
   ', Descripcio Tipo Central: ' || p_descripcioTipoCentral||
   ', Id Funcion Central: ' || p_idFuncionCentral||
   ', Quantitat Tipo Central: ' || p_quantitatTipoCentral||
   --', Estat Tipos Central: ' || p_estatTiposCentral||
   ', Data Estat Tipo Central: ' || p_dataEstatTipoCentral||
   ', Observacio Tipos Central: ' || p_observacioTiposCentral;
      DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('      MODIFICACION TIPUS DE CENTRALS       ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------------------------');
      DBMS_OUTPUT.PUT_LINE(' Codi Tipo Central: ' ||p_idTipoCentral||
   ', Descripcio Tipo Central: ' || p_descripcioTipoCentral||
   ', Id Funcion Central: ' || p_idFuncionCentral||
   ', Quantitat Tipo Central: ' || p_quantitatTipoCentral||
   --', Estat Tipos Central: ' || p_estatTiposCentral||
   ', Data Estat Tipo Central: ' || p_dataEstatTipoCentral||
   ', Observacio Tipos Central: ' || p_observacioTiposCentral);
   c_sortidalog := 's_rsp';
    -- ComprovaciÃ³ del identificador tipus Central i dades a modificar-----
   IF p_idTipoCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICACION TIPUS DE CENTRALS: ');
      s_rsp := 'Falta especificar el codi tipus Central  ';
      DBMS_OUTPUT.put_line(s_rsp);
      RAISE e_tipo_central;

    If  p_descripcioTipoCentral IS NULL THEN
        DBMS_OUTPUT.ENABLE;
```

```
        DBMS_OUTPUT.PUT_LINE('MODIFICACION TIPUS DE CENTRALS:  ');
              s_rsp := 'Falta especificar tipus Central  ';
        DBMS_OUTPUT.put_line(s_rsp);
        RAISE e_tipo_central;
    END IF;
     If  p_idFuncionCentral IS NULL THEN
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('MODIFICACION TIPUS DE CENTRALS:  ');
              s_rsp := 'Falta especificar id Funcion Central';
        DBMS_OUTPUT.put_line(s_rsp);
        RAISE e_tipo_central;
    END IF;
     If  (p_quantitatTipoCentral IS NULL) OR (p_quantitatTipoCentral<0) THEN
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('MODIFICACION TIPUS DE CENTRALS:  ');
              s_rsp := 'Falta especificar Quantitat Tipo Central o el vaLor es mÃ©s petit que 0';
        DBMS_OUTPUT.put_line(s_rsp);
        RAISE e_tipo_central;
    END IF;
     If  (p_estatTiposCentral IS NULL OR p_estatTiposCentral<0 OR p_estatTiposCentral<2) THEN
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
              s_rsp := 'Falta especificar Estat Tipos Central, 1=ALTA o 2=BAIXA';
        DBMS_OUTPUT.put_line(s_rsp);
        RAISE e_tipo_central;
    END IF;
    If (p_dataEstatTipoCentral IS NULL) then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
        s_rsp := 'Falta especificar Data Estat Tipo Central';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_tipo_central;
   END IF;
     ELSE
  SELECT COUNT (*) INTO n_registres
        FROM TIPO_CENTRAL
        WHERE IDTIPOCENTRAL = p_idTipoCentral;
       If n_registres=0 then
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
          s_rsp := 'TIPO_CENTRAL: no existeix a la BBDD';
          DBMS_OUTPUT.PUT_LINE(s_rsp);
           RAISE e_tipo_central;
        END IF;
      -- Comprovem els parÃ metres NULL i advertim si falta algun parÃ metre
    If  (p_descripcioTipoCentral IS NULL) THEN
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
              s_rsp := 'Falta especificar tipus Central';
        DBMS_OUTPUT.put_line(s_rsp);
        RAISE e_tipo_central;
     END IF;
```

```
 If  (p_idFuncionCentral IS NULL) THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
          s_rsp := 'Falta especificar id Funcion Central';
    DBMS_OUTPUT.put_line(s_rsp);
    RAISE e_tipo_central;
END IF;
 If  (p_quantitatTipoCentral IS NULL OR p_quantitatTipoCentral<0) THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
          s_rsp := 'Falta especificar Quantitat Tipo Central o el vaLor es mÃ©s petit que 0';
    DBMS_OUTPUT.put_line(s_rsp);
    RAISE e_tipo_central;
END IF;
 If  (p_estatTiposCentral IS NULL OR p_estatTiposCentral<0 OR p_estatTiposCentral>1) THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
          s_rsp := 'Falta especificar Estat Tipos Central, 1=ALTA o 0=BAIXA';
    DBMS_OUTPUT.put_line(s_rsp);
    RAISE e_tipo_central;
END IF;
If (p_dataEstatTipoCentral IS NULL) then
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
    s_rsp := 'Falta especificar Data Estat Tipo Central';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_tipo_central;
END IF;
-- recuperem el codi de la TIPO_CENTRAL
   SELECT IDTIPOCENTRAL  INTO n_idTipoCentral
       FROM TIPO_CENTRAL
      WHERE IDTIPOCENTRAL=p_idTipoCentral;

   s_sql := 'UPDATE TIPO_CENTRAL SET ';

  -- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
     IF p_descripcioTipoCentral IS NOT NULL THEN
          -- Modifiquem el descripcio
          s_sql := s_sql || 'DESCRIPCIOTIPOCENTRAL='' ||p_descripcioTipoCentral|| ''',';
      END IF;
      IF p_idFuncionCentral IS NOT NULL THEN
          s_sql := s_sql || 'IDFUNCIOTIPOCENTRAL='' ||p_idFuncionCentral|| ''',';
       END IF;
      IF p_quantitatTipoCentral IS NOT NULL THEN
          s_sql := s_sql || 'QUANTITATTIPOCENTRAL='' ||p_quantitatTipoCentral|| ''',';
       END IF;
      IF p_estatTiposCentral IS NOT NULL THEN
          s_sql := s_sql || 'ESTATTIPOCENTRAL='' ||p_estatTiposCentral|| ''',';
       END IF;
       IF p_dataEstatTipoCentral IS NOT NULL THEN
          s_sql := s_sql || 'DATAESTATTIPOCENTRAL='' ||p_dataEstatTipoCentral|| ''',';
       END IF;
```

```
        IF p_observacioTiposCentral IS NOT NULL THEN
            s_sql := s_sql || 'OBSERVACIONSTIPOCENTRAL="' ||p_observacioTiposCentral|| '",';
        END IF;

        -- Eliminem la coma final de la sentencia SQL
        s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
        -- Afegim a la sentencia la condiciÃ³ del WHERE
        s_sql := s_sql || ' WHERE IDTIPOCENTRAL="' ||n_idTipoCentral|| '"';

        EXECUTE IMMEDIATE s_sql;
       IF SQL%ROWCOUNT = 0  THEN
          --L'error serÃ  que Tipo Central ja el tenim en la taula i no admet duplicats--
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
          s_rsp := 'ActualitzaciÃ³ no realitzada';
          DBMS_OUTPUT.PUT_LINE(s_rsp);
          RAISE e_tipo_central;
        ELSE
      SELECT TIPUS_FUNCIONS.DESCRIPCIONFUNCIO INTO n_funcio
      FROM TIPUS_FUNCIONS, TIPO_CENTRAL
      WHERE TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL=p_idFuncionCentral;

      SELECT ESTAT.DESCRIPCIOESTAT INTO n_estat
      FROM ESTAT, TIPO_CENTRAL
      WHERE TIPO_CENTRAL.ESTATTIPOCENTRAL=p_estatTiposCentral;

          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICA TIPO CENTRAL:
          Codi tipo central: '||n_idTipoCentral||
          ' Descripcio: '||p_descripcioTipoCentral||
          ' Unitats: '||p_quantitatTipoCentral||
          ' Caracteristiques: '||n_funcio||
          ' Estat: '||n_estat||
          ' Data Estat Tipo Central: '||p_dataEstatTipoCentral||
          ' Observacions: '||p_observacioTiposCentral||' Modificacio satisfactoria. ');
          DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
          s_rsp := 'OK  Codi tipo central: '||n_idTipoCentral||
          ' Descripcio: '||p_descripcioTipoCentral||
          ' Unitats: '||p_quantitatTipoCentral||
          ' Caracteristiques: '||n_funcio||
          ' Estat: '||n_estat||
          ' Data Estat Tipo Central: '||p_dataEstatTipoCentral||
          ' Observacions: '||p_observacioTiposCentral||' Modificacio satisfactoria. ';

          pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      END IF;
    END IF;
  COMMIT;
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR TIPO CENTRAL: ');
```

```
    s_rsp := 'Error: Tipo Central duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_MODIFICAR_TIPO_CENTRAL;
/*********************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
          PROCEDIMENT PCR_CONSULTA_TIPO_CENTRAL
Procediment per consultar el tipus de central de producciÃ³ d'energia.
*********************************************************************/
PROCEDURE PRC_CONSULTAR_TIPO_CENTRAL(
  s_rsp out NOCOPY VARCHAR2)
  AS

  CURSOR C_TIPO_CENTRAL IS

  SELECT  DISTINCT TIPO_CENTRAL.IDTIPOCENTRAL,
  TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL,
  TIPO_CENTRAL.DATAESTATTIPOCENTRAL,
  ESTAT.DESCRIPCIOESTAT,
  TIPO_CENTRAL.QUANTITATTIPOCENTRAL,
  TIPUS_FUNCIONS.DESCRIPCIONFUNCIO,
  TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL
  FROM TIPO_CENTRAL,TIPUS_FUNCIONS,ESTAT
  WHERE TIPO_CENTRAL.IDTIPOCENTRAL=TIPO_CENTRAL.IDTIPOCENTRAL
  AND TIPO_CENTRAL.ESTATTIPOCENTRAL=ESTAT.IDESTAT
  AND TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL=TIPUS_FUNCIONS.IDFUNCIO;

  BEGIN
  c_procesLog := 'CONSULTAR_TIPO_CENTRAL';
  c_dataHoraLog := SYSDATE;
  c_entradaLog :=' CONSULTA TIPUS DE CENTRALS' ;
  c_sortidalog := 's_rsp';

  SELECT COUNT (*) INTO n_registres
  FROM TIPO_CENTRAL
```

```
 WHERE TIPO_CENTRAL.IDTIPOCENTRAL=TIPO_CENTRAL.IDTIPOCENTRAL;

 IF n_registres=0 THEN
    DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTAR TIPO CENTRAL ');
   s_rsp := 'Tipo de Central no existent a la BBDD';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_tipo_central;
   END IF;

  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('          CONSULTA TIPO CENTRAL              ');
  DBMS_OUTPUT.PUT_LINE('-------------------------------------------------');

OPEN C_TIPO_CENTRAL;
   FETCH C_TIPO_CENTRAL INTO n_idTipoCentral,n_descripcioTipoCentral,
   n_dataEstatTipoCentral,n_descEstat,n_quantitatTipoCentral,n_funcio,
   n_observacioTiposCentral;
 WHILE C_TIPO_CENTRAL%FOUND LOOP
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(
'   Codi Tipo Central:  '
||n_idTipoCentral
||' Descripcio '
||n_descripcioTipoCentral||
' Data estat tipo centras: '||n_dataEstatTipoCentral||
' Estat Central: '||n_descEstat||
' Unitats: '||n_quantitatTipoCentral||
' Caracteristiques: '||n_funcio||
' Observacions: '||n_observacioTiposCentral );
 s_rsp := 'OK  CONSULTA TIPO CENTRAL:
 Codi Tipo Central:  '
||n_idTipoCentral
||' Descripcio '
||n_descripcioTipoCentral||
' Data estat tipo centras: '||n_dataEstatTipoCentral||
' Estat Central: '||n_descEstat||
' Unitats: '||n_quantitatTipoCentral||
' Caracteristiques: '||n_funcio||
' Observacions: '||n_observacioTiposCentral ;
 FETCH C_TIPO_CENTRAL INTO n_idTipoCentral,n_descripcioTipoCentral,
   n_dataEstatTipoCentral,n_estat,n_quantitatTipoCentral,n_funcio,
   n_observacioTiposCentral;
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total tipo de Centrals: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');

CLOSE C_TIPO_CENTRAL;
```

```
COMMIT;
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTAR TIPO DE CENTRAL: ');
    s_rsp := 'Error: Tipo Central duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
 END PRC_CONSULTAR_TIPO_CENTRAL;
END "GESTION_TIPO_CENTRAL";
```

11.2 TRATAMIENTO DE GESTION CLASES DE CENTRALES.

Se gestiona si son centrales de Producción o bine de Distribución.

DEFINICION	PROCEDIMIENTO
Alta de la Clase de Central	PRC_ALTA_CLASSE_CENTRAL
Modificar la Clase de Central	PRC_MODIFICA_CLASSE_CENTRAL
Consulta la Clase de Central	PRC_CONSULTAR_CLASSE_CENTRAL

```
CREATE OR REPLACE
PACKAGE        "GESTION_CLASSE_CENTRAL" AS

PROCEDURE PRC_ALTA_CLASSE_CENTRAL (
                        p_descripcioClasse in CLASSE_CENTRAL.descripcioClasse%TYPE,
                        s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_MODIFICA_CLASSE_CENTRAL(
                        p_descripcioClasse in CLASSE_CENTRAL.descripcioClasse%TYPE,
                        p_idClasse in CLASSE_CENTRAL.idClasse%TYPE,
                        s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_CONSULTAR_CLASSE_CENTRAL(
                        s_rsp out NOCOPY VARCHAR2);

END "GESTION_CLASSE_CENTRAL";
```

11.2.1 PROCEDIMIENTO SPL PACKAGE CLASE DE CENTRALES.

```
CREATE OR REPLACE
PACKAGE BODY         "GESTION_CLASSE_CENTRAL" AS
  c_procesLog           LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp               LOG_TFC.rspLog%TYPE;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);
  n_descripcioClasse CLASSE_CENTRAL.descripcioClasse%TYPE;
  n_idClasse   CLASSE_CENTRAL.idClasse%TYPE;
  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_classe_central   EXCEPTION;

/*****************************************************************************
   NOM:       PCR_ALTA_CLASSE_CENTRAL
   DESCRIPCIÃ“:
            Procediment encarregat donar alta de Classe de Central.
            La inserciÃ³ de dades esta controlada per un disparador o trigger
                        el qual tÃ© associat una seqÃºÃ¨ncia SEQ_CLASSE_CENTRAL per generar
                        la clau primaria n format numÃ©rica de forma autoincrementable
                        les dades es modificaran en la taula CENTRAL i el identificador de
            clau primaria del de la taula Central.
*****************************************************************************/

PROCEDURE PRC_ALTA_CLASSE_CENTRAL(
p_descripcioClasse in CLASSE_CENTRAL.descripcioClasse%TYPE,
    S_RSP OUT VARCHAR2)
AS
BEGIN
    c_procesLog := 'PRC_ALTA_CLASSE_CENTRAL';
```

```
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='Decripcio Classe de Central: ' || p_descripcioClasse;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE(' ALTA CLASSES DE CENTRALS ');
    DBMS_OUTPUT.PUT_LINE('_____');
    DBMS_OUTPUT.PUT_LINE('ALTA CLASSES DE CENTRALS : Decripcio Classe de Central: '
    || p_descripcioClasse);
    c_sortidalog := 's_rsp';
    If p_descripcioClasse IS NULL then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE(' ALTA CLASSES DE CENTRALS ');
       s_rsp := 'Falta especificar la classe de central';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_classe_central;
     END IF;
-- Comprovem si existeix el  donar d'alta
  SELECT COUNT (*) INTO n_registres
  FROM CLASSE_CENTRAL
  WHERE CLASSE_CENTRAL.DESCRIPCIOCLASSE=p_descripcioClasse;

  If n_registres=0 then
-- El donem d'alta a la Classe de Central
      INSERT INTO CLASSE_CENTRAL(descripcioClasse)
      VALUES(p_descripcioClasse);
 --Gravem en la taula log------------------------------------------------------
      DBMS_OUTPUT.PUT_LINE('Insertada la classe de central: '
      ||p_descripcioClasse);
      DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
      s_rsp :='Ok: Insertat satisfactoriament'|| p_descripcioClasse;
      pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    ELSE
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ALTA CLASSES DE CENTRALS ');
      s_rsp := 'Tipus Inspeccio duplicat ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_classe_central;
    END IF;
   COMMIT;
EXCEPTION
   WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
      ELSE
         -- L'error si ha estat controlat per codi
         s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('  ');
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
```

```
    ROLLBACK;
end PRC_ALTA_CLASSE_CENTRAL;

/*******************************************************************************
  NOM:       PRC_MODIFICA_CLASSE_CENTRAL
  DESCRIPCIÃ“:
            Procediment encarregat de modificar la classe de central.
            Controla la descripcio i el id de la classe de central.
*******************************************************************************/
PROCEDURE PRC_MODIFICA_CLASSE_CENTRAL(
    p_descripcioClasse IN CLASSE_CENTRAL.descripcioClasse%TYPE,
    P_IDCLASSE IN CLASSE_CENTRAL.idClasse%TYPE,
S_RSP OUT VARCHAR2)
as
begin
   c_procesLog := 'PRC_MODIFICAR_CALSSE_CENTRAL';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Decripcio Central: ' || p_descripcioClasse;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('MODIFICAR CLASSES DE CENTRALS');
   DBMS_OUTPUT.PUT_LINE('_____________________');
   DBMS_OUTPUT.PUT_LINE('Classe e Central: '||p_descripcioClasse);
   c_sortidalog := 's_rsp';

-- ComprovaciÃ³ del identificador Classe Central
 If  (p_descripcioClasse IS NULL) THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR CLASSES DE CENTRALS  ');
    s_rsp := 'Falta especificar la central';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_classe_central;
 END IF;
    SELECT COUNT (*)INTO n_registres
    FROM CLASSE_CENTRAL
    WHERE CLASSE_CENTRAL.DESCRIPCIOCLASSE=p_descripcioClasse;

 If n_registres=0 then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR CLASSES DE CENTRALS ');
    s_rsp := 'CLASSE_CENTRAL no existeix a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_classe_central;
 END IF;

-- recuperem el codi de la central
 SELECT CLASSE_CENTRAL.idClasse,
 CLASSE_CENTRAL.DESCRIPCIOCLASSE INTO n_idClasse,n_descripcioClasse
 FROM CLASSE_CENTRAL
 WHERE DESCRIPCIOCLASSE=p_descripcioClasse;

 s_sql := 'UPDATE CLASSE_CENTRAL SET ';
```

```
-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF p_descripcioClasse IS NOT NULL THEN
-- Modifiquem el descripcio classe central
     s_sql := s_sql || 'DESCRIPCIOCLASSE='' ||p_descripcioClasse|| '',';
  END IF;
-- Eliminem la coma final de la sentencia SQL
  s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃ³ del WHERE
  s_sql := s_sql || ' WHERE idClasse='' ||n_idClasse|| '''';

  EXECUTE IMMEDIATE s_sql;

  IF SQL%ROWCOUNT = 0  THEN
--L'error serÃ  que el la via ja el tenim en la taula i no admet duplicats--
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR_CALSSE_CENTRAL ');
     DBMS_OUTPUT.PUT_LINE('**********************');
     s_rsp := 'ActualitzaciÃ³ no realitzada';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_classe_central;
  ELSE
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('Modificacio Calsse de Central feta amb el
      Codi Classe de Central: '||n_idClasse||
     ' Descripcio: '||p_descripcioClasse);
     DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
     DBMS_OUTPUT.put_line ('OK: Modificacio Calsse de Central feta amb el
      Codi Classe de Central: '||n_idClasse||
     ' Descripcio: '||p_descripcioClasse);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;
EXCEPTION
   WHEN DUP_VAL_ON_INDEX THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR CLASSE DE CENTRAL ');
      s_rsp := 'Error: Tipus Inspeccions duplicada. No es poden fer les modificacions';
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
   WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         ----------------------------------------
         s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
         -- L'error si ha estat controlat per codi
         ----------------------------------------
         s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
```

```
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

end PRC_MODIFICA_CLASSE_CENTRAL;
/*****************************************************************************
  NOM:       PRC_CONSULTAR_CLASE_CENTRAL
  DESCRIPCIÃ“:
          Procediment encarregat de consultar les dades d'una Classe de
                  Central.

*****************************************************************************/
PROCEDURE PRC_CONSULTAR_CLASSE_CENTRAL(
      s_rsp out NOCOPY VARCHAR2)
AS

CURSOR C_CLASSE_CENTRAL IS
SELECT CLASSE_CENTRAL.IDCLASSE,
CLASSE_CENTRAL.DESCRIPCIOCLASSE
FROM CLASSE_CENTRAL
WHERE CLASSE_CENTRAL.IDCLASSE=CLASSE_CENTRAL.IDCLASSE;

BEGIN
   c_procesLog := 'CONSULTAR_CLASE_CENTRAL';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='CONSULTAR_CLASE_CENTRAL';
   c_sortidalog := 's_rsp';
-- Comprovem que p_descripcioClasse  no sigui NULL---------------------------------------------

 --1er. Comprovem que existeixi la classe de central
   SELECT COUNT (*)     INTO n_registres
    FROM CLASSE_CENTRAL
    WHERE  CLASSE_CENTRAL.IDCLASSE=CLASSE_CENTRAL.IDCLASSE;


 If n_registres=0 then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTAR_CLASSE_CENTRAL ');
    s_rsp := 'CLASSE CENTRAL no existent a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_classe_central;
    END IF;

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('          CONSULTA CLASSES DE CENTRALS          ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------------------------');

   OPEN C_CLASSE_CENTRAL;
   FETCH C_CLASSE_CENTRAL INTO n_idClasse,n_descripcioClasse;
  WHILE C_CLASSE_CENTRAL%FOUND LOOP
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(
```

```
 '   Codi Classe de Central:  '
 ||n_idClasse
 ||' Descripcio '
 ||n_descripcioClasse);
  s_rsp := 'OK  CONSULTA CLASSES DE CENTRALS
  Codi Classe de Central:  '
 ||n_idClasse
 ||' Descripcio: '
 ||n_descripcioClasse;
  FETCH C_CLASSE_CENTRAL INTO n_idClasse,n_descripcioClasse;

  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total classes de centrals: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');

CLOSE C_CLASSE_CENTRAL;

COMMIT;
 EXCEPTION

   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi----------------------------------------
       s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi----------------------------------------
       s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line (s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;

END PRC_CONSULTAR_CLASSE_CENTRAL;
END "GESTION_CLASSE_CENTRAL";
```

11.3 TRATAMIENTO DE GESTION TIPO DE FUNCIONALIDAES.

Este procedimiento es para tratar el tipo de centrales en temas de emsiones y residuos.

DEFINICION	PROCEDIMIENTO
Alta de la Clase de Central	PRC_ALTA_TIPUS_FUNCIONS
Modificar la Clase de Central	PRC_MODIFICAR_TIPUS_FUNCIONS
Consulta la Clase de Central	PRC_CONSULTAR_TIPUS_FUNCIONS

```
CREATE OR REPLACE
PACKAGE        "GESTION_TIPUS_FUNCIONS" AS
PROCEDURE PRC_ALTA_TIPUS_FUNCIONS(
   p_descripcioTFuncions in TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE,
   s_rsp out NOCOPY VARCHAR2);

 PROCEDURE PRC_MODIFICAR_TIPUS_FUNCIONS(
   p_descripcioTFuncions in TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE,
   p_idTFuncions in TIPUS_FUNCIONS.IDFUNCIO%TYPE,
   s_rsp out NOCOPY VARCHAR2);

  procedure PRC_CONSULTAR_TIPUS_FUNCIONS(
   s_rsp out NOCOPY VARCHAR2);
END GESTION_TIPUS_FUNCIONS;
```

11.3.1 PROCEDIMIENTO SPL PACKAGE GESTION TIPO DE FUNCIONALIDAES.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_TIPUS_FUNCIONS" AS
c_procesLog LOG_TFC.procesLog%TYPE;
 c_dataHoraLog LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog LOG_TFC.entradaLog%TYPE;
 c_sortidaLog LOG_TFC.sortidaLog%TYPE;
 s_rsp LOG_TFC.rspLog%TYPE;
 n_registres  NUMBER;
 s_sql VARCHAR2 (2000);

 n_descripcioTFuncions TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE;
```

```
 n_idTFuncions   TIPUS_FUNCIONS.IDFUNCIO%TYPE;
 n_NUM_ERR NUMBER(10);
 sortida varchar2(500):='';
 e_tipo_Funcions    EXCEPTION;
---------------------------------------------------------------
-- create procedure for table "LOG_TFC i TIPUS_FUNCIONS"
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
  NOM:       PRC_ALTA_TIPUS_FUNCIONS
  DESCRIPCIÃ“:
    Procediment encarregat de dades d'alta a la taula Tipus Funcions de Central
    DistribuciÃ³.

*****************************************************************************/
  procedure PRC_ALTA_TIPUS_FUNCIONS(
   p_descripcioTFuncions in TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE,
   s_rsp out NOCOPY VARCHAR2)
  AS

  BEGIN
   c_procesLog := 'PRC_ALTA_TIPUS_FUNCIONS';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Decripcio Tipo Funcions de Central Produccio: ' || p_descripcioTFuncions;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE FUNCIONS DE LA CENTRAL DE PRODUCCIO');
   DBMS_OUTPUT.PUT_LINE('_____________________________');
   DBMS_OUTPUT.PUT_LINE('Tipus de Funcio: '|| p_descripcioTFuncions);
   c_sortidalog := 's_rsp';
   If (p_descripcioTFuncions IS NULL) then
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE FUNCIONS DE LA CENTRAL DE PRODUCCIO  ');
        s_rsp := 'Falta especificar Tipo Funcions de Central Produccio³';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_tipo_Funcions;
    END IF;
     -- Comprovem si existeix el Tipo Funcions de Central DistribuciÃ³ a  donar d'alta
       SELECT COUNT (*) INTO n_registres
         FROM TIPUS_FUNCIONS
         WHERE DESCRIPCIONFUNCIO=p_descripcioTFuncions;
    If n_registres=0 then
       -- El donem d'alta a la tipus lectura
         INSERT INTO TIPUS_FUNCIONS(DESCRIPCIONFUNCIO)
         VALUES(p_descripcioTFuncions);
         --Gravem en la taula log-----------------------------------------------------
           DBMS_OUTPUT.ENABLE;
           DBMS_OUTPUT.PUT_LINE('Insertada el tipus de Funcio de la
           Central de Produccio: '||p_descripcioTFuncions);
           DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
           s_rsp :='Ok: En la Taula TIPUS DE FUNCIONS Insertada
           Tipus de funcio de la Central de Produccio:  '||p_descripcioTFuncions;
           pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
```

```
            DBMS_OUTPUT.PUT_LINE(s_rsp);
          ELSE
            DBMS_OUTPUT.ENABLE;
            DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE FUNCIONS DE LA CENTRAL DE PRODUCCIO ');
            s_rsp := 'DESCRICIO Tipus de funcio duplicada ';
            DBMS_OUTPUT.PUT_LINE(s_rsp);
          RAISE e_tipo_Funcions;
        END IF;
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        DBMS_OUTPUT.ENABLE;
            DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE FUNCIONS DE LA CENTRAL DE PRODUCCIO ');
        s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
        DBMS_OUTPUT.PUT_LINE(s_rsp);
      ELSE
        -- L'error si ha estat controlat per codi
       DBMS_OUTPUT.ENABLE;
            DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE FUNCIONS DE LA CENTRAL DE PRODUCCIO');
        s_rsp := 'Error: ' || s_rsp;
        DBMS_OUTPUT.PUT_LINE(s_rsp);
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('  ');
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;

END PRC_ALTA_TIPUS_FUNCIONS;

PROCEDURE PRC_MODIFICAR_TIPUS_FUNCIONS(
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
  NOM:       PRC_MODIFICAR_TIPUS_FUNCIONS
  DESCRIPCIÃ“:
    Procediment encarregat de modificar les dades d'un Tipus Funcions de Central
    DistribuciÃ³.
    Les dades es modificaran en la taula Tipus_Funcions i el identificador de
    tipus Funcions apuntarÃ  a la clau primaria del codi de la taula
    Tipus_Funcions.
*****************************************************************************/
   p_descripcioTFuncions in TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE,
   p_idTFuncions in TIPUS_FUNCIONS.IDFUNCIO%TYPE,
   s_rsp out NOCOPY VARCHAR2)
  AS

  BEGIN
   c_procesLog := 'PRC_MODIFICAR_TIPUS_FUNCIONS';
   c_dataHoraLog := SYSDATE;
```

```
  c_entradaLog :='Decripcio Tipus Funcions de Central: ' || p_descripcioTFuncions;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('MODIFICAR EL TIPUS DE FUNCIONS DE LA CENTRAL DE PRODUCCIO');
  DBMS_OUTPUT.PUT_LINE('_____');
  DBMS_OUTPUT.PUT_LINE('Tipus de Funcio: '|| p_descripcioTFuncions);
  c_sortidalog := 's_rsp';
  -- ComprovaciÃ³ del identificador Tipus Funcions de Central-----

  If  p_descripcioTFuncions IS NULL THEN
       DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('MODIFICAR EL TIPUS DE FUNCIONS DE LA CENTRAL DE PRODUCCIO:  ');
             s_rsp := 'Falta especificar Tipus Funcions de Central';
       DBMS_OUTPUT.put_line(s_rsp);
       RAISE e_tipo_Funcions;
     END IF;

        SELECT COUNT (*) INTO n_registres
        FROM TIPUS_FUNCIONS
        WHERE TIPUS_FUNCIONS.DESCRIPCIONFUNCIO=p_descripcioTFuncions;
   If n_registres=0 then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('MODIFICAR EL TIPUS DE FUNCIONS DE LA CENTRAL DE PRODUCCIO:  ');
       s_rsp := 'TIPUS FUNCIONS no existeix a la BBDD';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_tipo_Funcions;
     END IF;

      -- recuperem el codi de la tipus lectura
  SELECT TIPUS_FUNCIONS.IDFUNCIO,
  TIPUS_FUNCIONS.DESCRIPCIONFUNCIO INTO n_idTFuncions,n_descripcioTFuncions
  FROM TIPUS_FUNCIONS
  WHERE TIPUS_FUNCIONS.IDFUNCIO=p_descripcioTFuncions;

  s_sql := 'UPDATE TIPUS_FUNCIONS SET ';

-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF p_descripcioTFuncions IS NOT  NULL THEN
  -- Modifiquem el descripcio Tipus Funcions de Central
  s_sql := s_sql || 'DESCRIPCIONFUNCIO=''' || p_descripcioTFuncions || ''',';
  END IF;
 -- Eliminem la coma final de la sentencia SQL
  s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
 -- Afegim a la sentencia la condiciÃ³ del WHERE
  s_sql := s_sql || ' WHERE IDFUNCIO=''' ||n_idTFuncions|| '''';

 EXECUTE IMMEDIATE s_sql;

 IF SQL%ROWCOUNT = 0 THEN
--L'error serÃ  que el la via ja el tenim en la taula i no admet duplicats--
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR EL TIPUS DE FUNCIONS DE LA CENTRAL DE PRODUCCIO:  ');
    s_rsp := 'Actualitzaci0 no realitzada';
```

```
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_tipo_Funcions;
  ELSE
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('Modificacio feta del Tipus de Funcio
    de la Central de Produccio: codi: '||n_idTFuncions||
    ' Descripcio: '||p_descripcioTFuncions);
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
    DBMS_OUTPUT.put_line ('OK: Modificacio feta del Tipus de Funcio
    de la Central de Produccio: codi: '||n_idTFuncions||
    ' Descripcio: '||n_descripcioTFuncions);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;
EXCEPTION
   WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR_TIPUS_FUNCIONS: ');
    s_rsp := 'Error: Tipus Funcions de Central duplicada.
    No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi
       ----------------------------------------
       s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi
       ----------------------------------------
       s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line (s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;
END PRC_MODIFICAR_TIPUS_FUNCIONS;
/*******************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
   NOM:       PRC_CONSULTAR_TIPUS_FUNCIONS
   DESCRIPCIÃ“:
     Procediment encarregat de modificar les dades d'un Tipus Funcions de Central
     DistribuciÃ³.
     Les dades es consulten en la taula Tipus_Funcions i el identificador de
     tipus Funcions apuntarÃ  a la clau primaria del codi de la taula
     Tipus_Funcions.
*******************************************************************************/
procedure PRC_CONSULTAR_TIPUS_FUNCIONS(
   s_rsp out NOCOPY VARCHAR2)
```

```
AS

CURSOR C_FUNCIONS IS
  SELECT TIPUS_FUNCIONS.IDFUNCIO,TIPUS_FUNCIONS.DESCRIPCIONFUNCIO
  FROM TIPUS_FUNCIONS
  WHERE TIPUS_FUNCIONS.IDFUNCIO=TIPUS_FUNCIONS.IDFUNCIO;

BEGIN
    c_procesLog := 'PRC_CONSULTAR_TIPUS_FUNCIONS';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='CONSULTA TIPUS DE FUNCIONS DE LES CENTRALS DE PRODUCCIO';
    c_sortidalog := 's_rsp';

  --1er. Comprovem que existeixi Tipus de Funcion------------------------------
    SELECT COUNT (*) INTO n_registres
    FROM TIPUS_FUNCIONS
    WHERE  TIPUS_FUNCIONS.IDFUNCIO=TIPUS_FUNCIONS.IDFUNCIO;

    IF n_registres=0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTA TIPUS DE FUNCIONS DE LES CENTRALS DE PRODUCCIO: ');
    s_rsp := 'No hi han tipus de funcions donades d´alta ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_tipo_Funcions;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          CONSULTA TIPUS DE FUNCIONS DE LES CENTRALS DE PRODUCCIO          ');
    DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------------------------------');
    OPEN C_FUNCIONS;
    FETCH C_FUNCIONS INTO n_idTFuncions,n_descripcioTFuncions;
  WHILE C_FUNCIONS%FOUND LOOP
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(
'   Codi Tipus Funcio:  '
|| n_idTFuncions
||' Descripcio '
|| n_descripcioTFuncions);
  s_rsp := 'OK  CONSULTA TIPUS FUNCIONS DE LES CENTRALS
  Codi Tipus Funcio:  '
||n_idTFuncions
||' Descripcio: '
||n_descripcioTFuncions;
  FETCH C_FUNCIONS INTO n_idTFuncions,n_descripcioTFuncions;

  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total fucions tipus: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_FUNCIONS;
```

```
  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_CONSULTAR_TIPUS_FUNCIONS;

END "GESTION_TIPUS_FUNCIONS";
```

11.4 TRATAMIENTO DE GESTION TIPO CENTRAL.

DEFINICION	PROCEDIMIENTO
Alta Tipo de Central	PRC_ALTA_TIPO_CENTRAL
Modificar Tipo de Central	PRC_MODIFICAR_TIPO_CENTRAL
Consulta Tipo de Central	PRC_CONSULTAR_TIPO_CENTRAL

```
CREATE OR REPLACE PACKAGE        "GESTION_TIPO_CENTRAL" AS
procedure PRC_ALTA_TIPO_CENTRAL (
  p_descripcioTipoCentral in TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL%TYPE,
  p_idFuncionCentral in TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL%TYPE,
  p_quantitatTipoCentral in TIPO_CENTRAL.QUANTITATTIPOCENTRAL%TYPE,
  p_estatTiposCentral in TIPO_CENTRAL.ESTATTIPOCENTRAL%TYPE,
  p_dataEstatTipoCentral in TIPO_CENTRAL.DATAESTATTIPOCENTRAL%TYPE,
  p_observacioTiposCentral in TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL%TYPE,
  s_rsp out NOCOPY VARCHAR2);
procedure PRC_MODIFICAR_TIPO_CENTRAL(
  p_idTipoCentral in TIPO_CENTRAL.IDTIPOCENTRAL%TYPE,
  p_descripcioTipoCentral in TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL%TYPE,
  p_idFuncionCentral in TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL%TYPE,
```

```
  p_quantitatTipoCentral in TIPO_CENTRAL.QUANTITATTIPOCENTRAL%TYPE,
  p_estatTiposCentral in TIPO_CENTRAL.ESTATTIPOCENTRAL%TYPE,
  p_dataEstatTipoCentral in TIPO_CENTRAL.DATAESTATTIPOCENTRAL%TYPE,
  p_observacioTiposCentral in TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL%TYPE,
  s_rsp out NOCOPY VARCHAR2);
procedure PRC_CONSULTAR_TIPO_CENTRAL(
 s_rsp out NOCOPY VARCHAR2);
END  "GESTION_TIPO_CENTRAL";
```

11.4.1 PROCEDIMIENTO SPL PACKAGE GESTION TIPO CENTRAL.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_TIPO_CENTRAL" AS
 n_registres     NUMBER;
 s_sql         VARCHAR2 (2000);
 n_NUM_ERR          NUMBER(10);
 c_procesLog         LOG_TFC.procesLog%TYPE;
 c_dataHoraLog   LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog    LOG_TFC.entradaLog%TYPE;
 c_sortidaLog    LOG_TFC.sortidaLog%TYPE;
 s_rsp     LOG_TFC.RSPLOG%TYPE;
 n_idTipoCentral TIPO_CENTRAL.IDTIPOCENTRAL%TYPE;
 n_descripcioTipoCentral TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL%TYPE;
 n_idFuncionCentral  TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL%TYPE;
 n_quantitatTipoCentral TIPO_CENTRAL.QUANTITATTIPOCENTRAL%TYPE;
 n_estatTiposCentral TIPO_CENTRAL.ESTATTIPOCENTRAL%TYPE;
 n_dataEstatTipoCentral TIPO_CENTRAL.DATAESTATTIPOCENTRAL%TYPE;
 n_observacioTiposCentral TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL%TYPE;
 n_estat ESTAT.DESCRIPCIOESTAT%TYPE;
 n_funcions TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE;

 n_descEstat ESTAT.DESCRIPCIOESTAT%TYPE;
 n_funcio TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE;

 sortida VARCHAR2(1000):='';
 e_tipo_central   EXCEPTION;
/*************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
        PROCEDIMENT PCR_ALTA_TIPO_CENTRAL
Procediment per donar d'alta el tipus de central de produccÃ³ d'energia.
*************************************************************************/
PROCEDURE PRC_ALTA_TIPO_CENTRAL(
  p_descripcioTipoCentral in TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL%TYPE,
  p_idFuncionCentral in TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL%TYPE,
  p_quantitatTipoCentral in TIPO_CENTRAL.QUANTITATTIPOCENTRAL%TYPE,
  p_estatTiposCentral in TIPO_CENTRAL.ESTATTIPOCENTRAL%TYPE,
  p_dataEstatTipoCentral in TIPO_CENTRAL.DATAESTATTIPOCENTRAL%TYPE,
  p_observacioTiposCentral in TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL%TYPE,
  s_rsp out NOCOPY VARCHAR2)
  AS
```

```
BEGIN
 c_procesLog := 'PCR_ALTA_TIPO_CENTRAL';
 c_dataHoraLog := SYSDATE;
 c_entradaLog :=
 'Descripcio Tipo Central: ' || p_descripcioTipoCentral||
 ', Id Funcion Central: ' || p_idFuncionCentral||
 ', Quantitat Tipo Central: ' || p_quantitatTipoCentral||
 --', Estat Tipos Central: ' || p_estatTiposCentral||
 ', Data Estat Tipo Central: ' || p_dataEstatTipoCentral||
 ', Observacio Tipos Central: ' || p_observacioTiposCentral;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(' ');
 DBMS_OUTPUT.PUT_LINE('      ALTA TIPUS DE CENTRALS       ');
 DBMS_OUTPUT.PUT_LINE('--------------------------------------');
 DBMS_OUTPUT.PUT_LINE(' Descripcio Tipo Central: ' || p_descripcioTipoCentral||
 ', Id Funcion Central: ' || p_idFuncionCentral||
 ', Quantitat Tipo Central: ' || p_quantitatTipoCentral||
 --', Estat Tipos Central: ' || p_estatTiposCentral||
 ', Data Estat Tipo Central: ' || p_dataEstatTipoCentral||
 ', Observacio Tipos Central: ' || p_observacioTiposCentral);
 c_sortidalog := 's_rsp';

 n_estatTiposCentral:=1;

 SELECT ESTAT.DESCRIPCIOESTAT INTO n_estat
 FROM ESTAT
 WHERE ESTAT.IDESTAT=n_estatTiposCentral;

 SELECT TIPUS_FUNCIONS.DESCRIPCIONFUNCIO INTO n_funcions
 FROM TIPUS_FUNCIONS
 WHERE TIPUS_FUNCIONS.IDFUNCIO=p_idFuncionCentral;

  If p_descripcioTipoCentral IS NULL then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE CENTRALS: ');
       s_rsp := 'Falta especificar Tipo Central';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_tipo_central;
  END IF;
 If (p_idFuncionCentral IS NULL) OR (p_idFuncionCentral=0) OR (p_idFuncionCentral<0) then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE CENTRALS: ');
       s_rsp := 'Falta especificar Id Funcion CentraL. ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_tipo_central;
  END IF;
  If (p_quantitatTipoCentral IS NULL OR p_quantitatTipoCentral<0) then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE CENTRALS:  ');
       s_rsp := 'Falta especificar Quantitat Tipo Central o el vaLor es mes petit que 0 ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
```

```
        RAISE e_tipo_central;
    END IF;
-- Comprovem si existeix el el tipus de lectura a  donar d'alta
   SELECT COUNT (*)INTO n_registres
   FROM TIPO_CENTRAL
   WHERE DESCRIPCIOTIPOCENTRAL = p_descripcioTipoCentral
   AND TIPO_CENTRAL.ESTATTIPOCENTRAL=n_estatTiposCentral;

  If n_registres=0 then
-- El donem d'alta a la tipus lectura
  n_estatTiposCentral:=1;
  INSERT INTO TIPO_CENTRAL (DESCRIPCIOTIPOCENTRAL,
                  IDFUNCIOTIPOCENTRAL,
                  QUANTITATTIPOCENTRAL,
                  ESTATTIPOCENTRAL,
                  DATAESTATTIPOCENTRAL,
                  OBSERVACIONSTIPOCENTRAL)
          VALUES(p_descripcioTipoCentral,
              p_idFuncionCentral,
              p_quantitatTipoCentral,
              n_estatTiposCentral,
              p_dataEstatTipoCentral,
              p_observacioTiposCentral);
          --Gravem en la taula log-----------------------------------------------------

  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Insertada satisfatoriament el tipus de
  Central '||p_descripcioTipoCentral||
  ' Data alta:'||p_dataEstatTipoCentral||
  ' Estat: '||n_estat||
   ' Unitats: '||p_quantitatTipoCentral||
  ' Cacteristiques: '||n_funcions||
  ' Observacions:'||p_observacioTiposCentral);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  s_rsp :='Ok: Insertada satisfatoriament el tipus de
  Central '||p_descripcioTipoCentral||
  ' Data alta:'||p_dataEstatTipoCentral||
  ' Estat: '||n_estat||
  ' Unitats: '||p_quantitatTipoCentral||
  ' Cacteristiques: '||n_funcions||
  ' Observacions:'||p_observacioTiposCentral;
  pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ELSE
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE CENTRALS: ');
    s_rsp := 'DESCRICIO Tipus Central duplicat';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_tipo_central;
  END IF;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
```

```
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA_TIPO_CENTRAL:  ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

  END PRC_ALTA_TIPO_CENTRAL;
/**************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
          PROCEDIMENT PCR_MODIFICA_TIPO_CENTRAL
Procediment per modificar el tipus de central de produccià³ d'energia.
**************************************************************************/
procedure PRC_MODIFICAR_TIPO_CENTRAL(
   p_idTipoCentral in TIPO_CENTRAL.IDTIPOCENTRAL%TYPE,
   p_descripcioTipoCentral in TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL%TYPE,
   p_idFuncionCentral in TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL%TYPE,
   p_quantitatTipoCentral in TIPO_CENTRAL.QUANTITATTIPOCENTRAL%TYPE,
   p_estatTiposCentral in TIPO_CENTRAL.ESTATTIPOCENTRAL%TYPE,
   p_dataEstatTipoCentral in TIPO_CENTRAL.DATAESTATTIPOCENTRAL%TYPE,
   p_observacioTiposCentral in TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL%TYPE,
   s_rsp out NOCOPY VARCHAR2)
   AS

   BEGIN
    c_procesLog := 'PCR_MODIFICA_TIPO_CENTRAL';
    c_dataHoraLog := SYSDATE;
    c_entradaLog := 'Id Tipo Central Tipo Central: ' ||p_idTipoCentral||
    ', Descripcio Tipo Central: ' || p_descripcioTipoCentral||
    ', Id Funcion Central: ' || p_idFuncionCentral||
    ', Quantitat Tipo Central: ' || p_quantitatTipoCentral||
    --', Estat Tipos Central: ' || p_estatTiposCentral||
    ', Data Estat Tipo Central: ' || p_dataEstatTipoCentral||
    ', Observacio Tipos Central: ' || p_observacioTiposCentral;
       DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('      MODIFICACION TIPUS DE CENTRALS       ');
    DBMS_OUTPUT.PUT_LINE('-----------------------------------------------');
       DBMS_OUTPUT.PUT_LINE(' Codi Tipo Central: ' ||p_idTipoCentral||
    ', Descripcio Tipo Central: ' || p_descripcioTipoCentral||
    ', Id Funcion Central: ' || p_idFuncionCentral||
    ', Quantitat Tipo Central: ' || p_quantitatTipoCentral||
    --', Estat Tipos Central: ' || p_estatTiposCentral||
    ', Data Estat Tipo Central: ' || p_dataEstatTipoCentral||
    ', Observacio Tipos Central: ' || p_observacioTiposCentral);
```

```
c_sortidalog := 's_rsp';
 -- ComprovaciÃ³ del identificador tipus Central i dades a modificar-----
IF p_idTipoCentral IS NULL THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICACION TIPUS DE CENTRALS: ');
   s_rsp := 'Falta especificar el codi tipus Central  ';
   DBMS_OUTPUT.put_line(s_rsp);
   RAISE e_tipo_central;

 If  p_descripcioTipoCentral IS NULL THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICACION TIPUS DE CENTRALS:  ');
          s_rsp := 'Falta especificar tipus Central  ';
     DBMS_OUTPUT.put_line(s_rsp);
     RAISE e_tipo_central;
 END IF;
  If  p_idFuncionCentral IS NULL THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICACION TIPUS DE CENTRALS:  ');
          s_rsp := 'Falta especificar id Funcion Central';
     DBMS_OUTPUT.put_line(s_rsp);
     RAISE e_tipo_central;
 END IF;
  If  (p_quantitatTipoCentral IS NULL) OR (p_quantitatTipoCentral<0) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICACION TIPUS DE CENTRALS:  ');
          s_rsp := 'Falta especificar Quantitat Tipo Central o el vaLor es mÃ©s petit que 0';
     DBMS_OUTPUT.put_line(s_rsp);
     RAISE e_tipo_central;
 END IF;
  If  (p_estatTiposCentral IS NULL OR p_estatTiposCentral<0 OR p_estatTiposCentral<2) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
          s_rsp := 'Falta especificar Estat Tipos Central, 1=ALTA o 2=BAIXA';
     DBMS_OUTPUT.put_line(s_rsp);
     RAISE e_tipo_central;
 END IF;
 If (p_dataEstatTipoCentral IS NULL) then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
     s_rsp := 'Falta especificar Data Estat Tipo Central';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_tipo_central;
END IF;
  ELSE
SELECT COUNT (*) INTO n_registres
      FROM TIPO_CENTRAL
      WHERE IDTIPOCENTRAL = p_idTipoCentral;
     If n_registres=0 then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
       s_rsp := 'TIPO_CENTRAL: no existeix a la BBDD';
```

```
        DBMS_OUTPUT.PUT_LINE(s_rsp);
         RAISE e_tipo_central;
       END IF;
     -- Comprovem els parÃ metres NULL i advertim si falta algun parÃ metre
  If  (p_descripcioTipoCentral IS NULL) THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
            s_rsp := 'Falta especificar tipus Central';
      DBMS_OUTPUT.put_line(s_rsp);
      RAISE e_tipo_central;
   END IF;
   If  (p_idFuncionCentral IS NULL) THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
            s_rsp := 'Falta especificar id Funcion Central';
      DBMS_OUTPUT.put_line(s_rsp);
      RAISE e_tipo_central;
  END IF;
   If  (p_quantitatTipoCentral IS NULL OR p_quantitatTipoCentral<0) THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
            s_rsp := 'Falta especificar Quantitat Tipo Central o el vaLor es mÃ©s petit que 0';
      DBMS_OUTPUT.put_line(s_rsp);
      RAISE e_tipo_central;
  END IF;
   If  (p_estatTiposCentral IS NULL OR p_estatTiposCentral<0 OR p_estatTiposCentral>1) THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
            s_rsp := 'Falta especificar Estat Tipos Central, 1=ALTA o 0=BAIXA';
      DBMS_OUTPUT.put_line(s_rsp);
      RAISE e_tipo_central;
  END IF;
  If (p_dataEstatTipoCentral IS NULL) then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
      s_rsp := 'Falta especificar Data Estat Tipo Central';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_tipo_central;
 END IF;
  -- recuperem el codi de la TIPO_CENTRAL
     SELECT IDTIPOCENTRAL  INTO n_idTipoCentral
         FROM TIPO_CENTRAL
        WHERE IDTIPOCENTRAL=p_idTipoCentral;

     s_sql := 'UPDATE TIPO_CENTRAL SET ';

    -- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
       IF p_descripcioTipoCentral IS NOT NULL THEN
            -- Modifiquem el descripcio
            s_sql := s_sql || 'DESCRIPCIOTIPOCENTRAL='' ||p_descripcioTipoCentral|| '',';
       END IF;
       IF p_idFuncionCentral IS NOT NULL THEN
```

```
        s_sql := s_sql || 'IDFUNCIOTIPOCENTRAL='' ||p_idFuncionCentral|| ''','';
    END IF;
    IF p_quantitatTipoCentral IS NOT NULL THEN
        s_sql := s_sql || 'QUANTITATTIPOCENTRAL='' ||p_quantitatTipoCentral|| ''','';
    END IF;
    IF p_estatTiposCentral IS NOT NULL THEN
        s_sql := s_sql || 'ESTATTIPOCENTRAL='' ||p_estatTiposCentral|| ''','';
    END IF;
    IF p_dataEstatTipoCentral IS NOT NULL THEN
        s_sql := s_sql || 'DATAESTATTIPOCENTRAL='' ||p_dataEstatTipoCentral|| ''','';
    END IF;
    IF p_observacioTiposCentral IS NOT NULL THEN
        s_sql := s_sql || 'OBSERVACIONSTIPOCENTRAL='' ||p_observacioTiposCentral|| ''','';
    END IF;

    -- Eliminem la coma final de la sentencia SQL
    s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
    -- Afegim a la sentencia la condiciÃ³ del WHERE
    s_sql := s_sql || ' WHERE IDTIPOCENTRAL='' ||n_idTipoCentral|| ''';

    EXECUTE IMMEDIATE s_sql;
    IF SQL%ROWCOUNT = 0  THEN
       --L'error serÃ  que Tipo Central ja el tenim en la taula i no admet duplicats--
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('MODIFICA_TIPO_CENTRAL:  ');
       s_rsp := 'ActualitzaciÃ³ no realitzada';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_tipo_central;
    ELSE
    SELECT TIPUS_FUNCIONS.DESCRIPCIONFUNCIO INTO n_funcio
    FROM TIPUS_FUNCIONS, TIPO_CENTRAL
    WHERE TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL=p_idFuncionCentral;

    SELECT ESTAT.DESCRIPCIOESTAT INTO n_estat
    FROM ESTAT, TIPO_CENTRAL
    WHERE TIPO_CENTRAL.ESTATTIPOCENTRAL=p_estatTiposCentral;

       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('MODIFICA TIPO CENTRAL:
       Codi tipo central: '||n_idTipoCentral||
       ' Descripcio: '||p_descripcioTipoCentral||
       ' Unitats: '||p_quantitatTipoCentral||
       ' Caracteristiques: '||n_funcio||
       ' Estat: '||n_estat||
       ' Data Estat Tipo Central: '||p_dataEstatTipoCentral||
       ' Observacions: '||p_observacioTiposCentral||' Modificacio satisfactoria. ');
       DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
       s_rsp := 'OK  Codi tipo central: '||n_idTipoCentral||
       ' Descripcio: '||p_descripcioTipoCentral||
       ' Unitats: '||p_quantitatTipoCentral||
       ' Caracteristiques: '||n_funcio||
       ' Estat: '||n_estat||
```

```
            ' Data Estat Tipo Central: '||p_dataEstatTipoCentral||
            ' Observacions: '||p_observacioTiposCentral||' Modificacio satisfactoria. ';

            pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      END IF;
    END IF;
  COMMIT;
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR TIPO CENTRAL: ');
    s_rsp := 'Error: Tipo Central duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_MODIFICAR_TIPO_CENTRAL;
/*********************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
          PROCEDIMENT PCR_CONSULTA_TIPO_CENTRAL
Procediment per consultar el tipus de central de produccióÃ³ d'energia.
*********************************************************************/
PROCEDURE PRC_CONSULTAR_TIPO_CENTRAL(
  s_rsp out NOCOPY VARCHAR2)
  AS

  CURSOR C_TIPO_CENTRAL IS

  SELECT  DISTINCT TIPO_CENTRAL.IDTIPOCENTRAL,
  TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL,
  TIPO_CENTRAL.DATAESTATTIPOCENTRAL,
  ESTAT.DESCRIPCIOESTAT,
  TIPO_CENTRAL.QUANTITATTIPOCENTRAL,
  TIPUS_FUNCIONS.DESCRIPCIONFUNCIO,
  TIPO_CENTRAL.OBSERVACIONSTIPOCENTRAL
  FROM TIPO_CENTRAL,TIPUS_FUNCIONS,ESTAT
  WHERE TIPO_CENTRAL.IDTIPOCENTRAL=TIPO_CENTRAL.IDTIPOCENTRAL
```

```
 AND TIPO_CENTRAL.ESTATTIPOCENTRAL=ESTAT.IDESTAT
 AND TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL=TIPUS_FUNCIONS.IDFUNCIO;

 BEGIN
 c_procesLog := 'CONSULTAR_TIPO_CENTRAL';
 c_dataHoraLog := SYSDATE;
 c_entradaLog :=' CONSULTA TIPUS DE CENTRALS' ;
 c_sortidalog := 's_rsp';

 SELECT COUNT (*) INTO n_registres
 FROM TIPO_CENTRAL
 WHERE TIPO_CENTRAL.IDTIPOCENTRAL=TIPO_CENTRAL.IDTIPOCENTRAL;

 IF n_registres=0 THEN
    DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTAR TIPO CENTRAL ');
   s_rsp := 'Tipo de Central no existent a la BBDD';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_tipo_central;
   END IF;

  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('          CONSULTA TIPO CENTRAL              ');
  DBMS_OUTPUT.PUT_LINE('-------------------------------------------------');

OPEN C_TIPO_CENTRAL;
   FETCH C_TIPO_CENTRAL INTO n_idTipoCentral,n_descripcioTipoCentral,
   n_dataEstatTipoCentral,n_descEstat,n_quantitatTipoCentral,n_funcio,
   n_observacioTiposCentral;
 WHILE C_TIPO_CENTRAL%FOUND LOOP
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(
'   Codi Tipo Central:  '
||n_idTipoCentral
||' Descripcio '
||n_descripcioTipoCentral||
' Data estat tipo centras: '||n_dataEstatTipoCentral||
' Estat Central: '||n_descEstat||
' Unitats: '||n_quantitatTipoCentral||
' Caracteristiques: '||n_funcio||
' Observacions: '||n_observacioTiposCentral );
 s_rsp := 'OK  CONSULTA TIPO CENTRAL:
 Codi Tipo Central:  '
||n_idTipoCentral
||' Descripcio '
||n_descripcioTipoCentral||
' Data estat tipo centras: '||n_dataEstatTipoCentral||
' Estat Central: '||n_descEstat||
' Unitats: '||n_quantitatTipoCentral||
' Caracteristiques: '||n_funcio||
' Observacions: '||n_observacioTiposCentral ;
 FETCH C_TIPO_CENTRAL INTO n_idTipoCentral,n_descripcioTipoCentral,
```

```
    n_dataEstatTipoCentral,n_estat,n_quantitatTipoCentral,n_funcio,
    n_observacioTiposCentral;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total tipo de Centrals: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');

CLOSE C_TIPO_CENTRAL;

COMMIT;
EXCEPTION
   WHEN DUP_VAL_ON_INDEX THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('CONSULTAR TIPO DE CENTRAL: ');
      s_rsp := 'Error: Tipo Central duplicada. No es poden fer les modificacions';
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
   WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         ----------------------------------------
         s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
         -- L'error si ha estat controlat per codi
         ----------------------------------------
         s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.put_line (s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
   END PRC_CONSULTAR_TIPO_CENTRAL;
END "GESTION_TIPO_CENTRAL";
```

11.5 TRATAMIENTO DE GESTION LINEA TIPUS.

Este procedimiento establece el tipo de línea que se desea conectar contiene las tolerancias máximas que soportan.

DEFINICION	PROCEDIMIENTO
Alta Linea Tipus	PRC_ALTA_LINEA_TIPUS
Modificar Linea Tipus	PRC_MODIFICAR_LINEA_TIPUS
Consulta Linea Tipus	PRC_CONSULTAR_TIPUS_FUNCIONS
Consulta línea Activas	PRC_CONSULTAR_LINEA_ACTIVA
Consulta línea Pasiva	PRC_CONSULTAR_LINEA_NO_ACTIVA

```
CREATE or replace PACKAGE        "GESTION_LINEA_TIPUS" AS
--- ALTA LINEA TIPUS
PROCEDURE PRC_ALTA_LINEA_TIPUS(
    p_dataEstatTipoLinea in LINEA_TIPUS.DATAESTATTLINEA%Type,
   p_consumMaximTipoLinea in LINEA_TIPUS.CONSUMMAXIMTLINEA%Type,
   p_observacionTipoLinea in LINEA_TIPUS.OBSERVACIOTLINEA%Type,
   s_rsp out NOCOPY VARCHAR);
--- MODIFICAR ESTAT O DADES DE LA LINEA TIPUS

PROCEDURE PRC_MODIFICAR_LINEA_TIPUS(
   p_idTipoLinea in LINEA_TIPUS.IDTLINEA%Type,
   p_estatTipoLinea in LINEA_TIPUS.IDESTATTLINEA%Type,
   p_dataEstatTipoLinea in LINEA_TIPUS.DATAESTATTLINEA%Type,
   p_consumMaximTipoLinea in LINEA_TIPUS.CONSUMMAXIMTLINEA%Type,
   p_observacionTipoLinea in LINEA_TIPUS.OBSERVACIOTLINEA%Type,
   s_rsp out NOCOPY VARCHAR2);

-- CONSULTA LINEA_TIPUS ESTAT ACTIVA
PROCEDURE PRC_CONSULTAR_LINEA_ACTIVA(
  s_rsp out NOCOPY VARCHAR2);
-- CONSULTA LINEA TIPUS ESTAT LINEA NO ACTIVA
```

```
PROCEDURE PRC_CONSULTAR_LINEA_NO_ACTIVA(
  s_rsp out NOCOPY VARCHAR2);
END "GESTION_LINEA_TIPUS";
```

11.5.1 PROCEDIMIENTO SPL PACKAGE GESTION LINEA TIPUS.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_LINEA_TIPUS" AS
 n_registres     NUMBER;
 s_sql          VARCHAR2 (2000);
 n_NUM_ERR          NUMBER(10);

 c_procesLog          LOG_TFC.procesLog%TYPE;
 c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog     LOG_TFC.entradaLog%TYPE;
 c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
 s_rsp      LOG_TFC.RSPLOG%TYPE;

 n_estatLinea NUMBER;
 n_estatTipoLinea LINEA_TIPUS.IDESTATTLINEA%TYPE;
 n_dataEstatTipoLinea   LINEA_TIPUS.DATAESTATTLINEA%Type;
 n_consumMaximTipoLinea   LINEA_TIPUS.CONSUMMAXIMTLINEA%TYPE;
 n_observacionTipoLinea  LINEA_TIPUS.OBSERVACIOTLINEA%TYPE;
 n_idTipoLinea  LINEA_TIPUS.IDTLINEA%Type;
 n_estat ESTAT.DESCRIPCIOESTAT%TYPE;
 n_idEstat LINEA_TIPUS.IDESTATTLINEA%TYPE;

 sortida VARCHAR2(1000):='';
 e_tipus_linea  EXCEPTION;
---------------------------------------------------------------------
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
          POCEDIMENT ALTA LINEA TIPUS
DESCRIPCIO:
La funcionalitat es de guardar els tipus de lineas on cada una d'elles tolera
una quantitat maxima de fluxe energeric.
Te dependencies amb LINEAS CONECTAR que hereda les dades per controlar
el nivell de sobrecarrega de les lineas.
*****************************************************************************/
PROCEDURE PRC_ALTA_LINEA_TIPUS(
   p_dataEstatTipoLinea in LINEA_TIPUS.DATAESTATTLINEA%Type,
   p_consumMaximTipoLinea in LINEA_TIPUS.CONSUMMAXIMTLINEA%Type,
   p_observacionTipoLinea in LINEA_TIPUS.OBSERVACIOTLINEA%Type,
   s_rsp out NOCOPY VARCHAR)
   AS

   BEGIN
   c_procesLog := 'PRC_LINEA_ ALTA_TIPUS';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Data Estat Tipus Linea: ' ||p_dataEstatTipoLinea||
```

```
', Cosum Maxim Tolerat Linea Tipus: ' ||p_consumMaximTipoLinea||
', Observacions Tipus Linea: ' ||p_observacionTipoLinea;
DBMS_OUTPUT.PUT_LINE(' ');
DBMS_OUTPUT.PUT_LINE('     ALTA TIPUS DE LINEA     ');
DBMS_OUTPUT.PUT_LINE('_____________________________');
DBMS_OUTPUT.PUT_LINE('Data Estat Tipus Linea: ' ||p_dataEstatTipoLinea||
', Cosum Maxim Tolerat Linea Tipus: ' ||p_consumMaximTipoLinea||
', Observacions Tipus Linea: ' ||p_observacionTipoLinea);
c_sortidalog := 's_rsp';

-- comprovacio de dades entrades per donar d'alta
  n_estatLinea:=1;
If (p_dataEstatTipoLinea IS NULL) then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('     ALTA TIPUS DE LINEA     ');
       DBMS_OUTPUT.PUT_LINE('_____________________________');
       s_rsp := 'Falta especificar data Estat Tipus Linea ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_tipus_linea;
 END IF;
 If (p_consumMaximTipoLinea IS NULL) then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('     ALTA TIPUS DE LINEA     ');
       DBMS_OUTPUT.PUT_LINE('_____________________________');
       s_rsp := 'Falta especificar el Cosum Maxim tolerable Tipus Linea ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_tipus_linea;
 END IF;
If (p_consumMaximTipoLinea<0 OR  p_consumMaximTipoLinea=0) then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('     ALTA TIPUS DE LINEA     ');
       DBMS_OUTPUT.PUT_LINE('_____________________________');
       s_rsp := 'Falta especificar el Cosum Maxim tolerable Tipus Linea no pot ser 0 menor de 0 ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_tipus_linea;
 END IF;
 -- Comprovem si existeix el estat  donar d'alta
    SELECT COUNT (*)      INTO n_registres
       FROM LINEA_TIPUS
       WHERE IDESTATTLINEA =  n_estatLinea
       AND DATAESTATTLINEA=p_dataEstatTipoLinea
       AND LINEA_TIPUS.IDTLINEA= LINEA_TIPUS.IDTLINEA
       AND CONSUMMAXIMTLINEA=p_consumMaximTipoLinea;

  If n_registres=0 then
  n_estatLinea:=1;
     -- El donem d'alta Tipus Linea
  INSERT INTO LINEA_TIPUS(IDESTATTLINEA,
                    DATAESTATTLINEA,
                    CONSUMMAXIMTLINEA,
                    OBSERVACIOTLINEA)
       VALUES(n_estatLinea,
```

```
            p_dataEstatTipoLinea,
            p_consumMaximTipoLinea,
            p_observacionTipoLinea);
        --Gravem en la taula log----------------------------------------------------


         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('Insertada el tipus de linea:  en la data: '||p_dataEstatTipoLinea||
         ' Consum maxim tolerable de:  '||p_consumMaximTipoLinea||
         ' Observacions: '||p_observacionTipoLinea);
         DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
         s_rsp :='Ok: Insertada el tipus de linea:  en la data: '||p_dataEstatTipoLinea||
         ' Consum maxim tolerable de:  '||p_consumMaximTipoLinea||
         ' Observacions: '||p_observacionTipoLinea;
         pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        ELSE
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('     ALTA TIPUS DE LINEA     ');
         DBMS_OUTPUT.PUT_LINE('_____________________________');
         s_rsp := 'DADES duplicades';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_tipus_linea;
      END IF;
    COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_ALTA_LINEA_TIPUS;
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
          MODIFICAR  DADES DE LA LINEA TIPUS
DESCRIPCIO:
Procediment per modificar les dades del tipus de linea.
*****************************************************************************/

PROCEDURE PRC_MODIFICAR_LINEA_TIPUS(
   p_idTipoLinea in LINEA_TIPUS.IDTLINEA%Type,
   p_estatTipoLinea in LINEA_TIPUS.IDESTATTLINEA%Type,
   p_dataEstatTipoLinea in LINEA_TIPUS.DATAESTATTLINEA%Type,
```

```
    p_consumMaximTipoLinea in LINEA_TIPUS.CONSUMMAXIMTLINEA%Type,
    p_observacionTipoLinea in LINEA_TIPUS.OBSERVACIOTLINEA%Type,
    s_rsp out NOCOPY VARCHAR2)
    AS

    BEGIN
    c_procesLog := 'PRC_MODIFICAR_LINEA_TIPUS';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='Id Tipus Linea: ' ||p_idTipoLinea||
    --', Estat Tipus Linea: ' || p_estatTipoLinea||
    ', Data Estat Tipus Linea: ' ||p_dataEstatTipoLinea||
    ', Cosum Maxim Tolerat Linea Tipus: ' ||p_consumMaximTipoLinea||
    ', Observacions Tipus Linea: ' ||p_observacionTipoLinea;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('    MODIFICAR LINEA TIPUS     ');
    DBMS_OUTPUT.PUT_LINE('_____');
    DBMS_OUTPUT.PUT_LINE('Id Tipus Linea: ' ||p_idTipoLinea||
    --', Estat Tipus Linea: ' || p_estatTipoLinea||
    ', Data Estat Tipus Linea: ' ||p_dataEstatTipoLinea||
    ', Cosum Maxim Tolerat Linea Tipus: ' ||p_consumMaximTipoLinea||
    ', Observacions Tipus Linea: ' ||p_observacionTipoLinea);
    c_sortidalog := 's_rsp';
-- comprovacio de dades
     IF p_idTipoLinea IS NULL THEN
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('MODIFICAR LINEA TIPUS: ');
       s_rsp := 'Falta especificar el codi Tipus Linea ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_tipus_linea;
        END IF;
    If (p_dataEstatTipoLinea IS NULL) then
            DBMS_OUTPUT.ENABLE;
            DBMS_OUTPUT.PUT_LINE('MODIFICAR LINEA TIPUS: ');
            s_rsp := 'Falta especificar data Estat Tipus Linea ';
            DBMS_OUTPUT.PUT_LINE(s_rsp);
          RAISE e_tipus_linea;
         END IF;
      If (p_consumMaximTipoLinea IS NULL) then
            DBMS_OUTPUT.ENABLE;
            DBMS_OUTPUT.PUT_LINE('MODIFICAR LINEA TIPUS: ');
            s_rsp := 'Falta especificar el Cosum Maxin tolerable Tipus Linea ';
            DBMS_OUTPUT.PUT_LINE(s_rsp);
          RAISE e_tipus_linea;
       END IF;
    If (p_consumMaximTipoLinea<0 OR  p_consumMaximTipoLinea=0) then
            DBMS_OUTPUT.ENABLE;
            DBMS_OUTPUT.PUT_LINE('MODIFICAR_LINEA_TIPUS ');
            s_rsp := 'Falta especificar el Cosum Maxim tolerable Tipus Linea ';
            DBMS_OUTPUT.PUT_LINE(s_rsp);
          RAISE e_tipus_linea;
     END IF;
     -- recuperem el codi Tipus Linea, estat i data estat
```

```
        SELECT LINEA_TIPUS.IDTLINEA,LINEA_TIPUS.IDESTATTLINEA,ESTAT.DESCRIPCIOESTAT
        INTO n_idTipoLinea,n_estatTipoLinea,n_estat
             FROM LINEA_TIPUS,ESTAT
             WHERE LINEA_TIPUS.IDTLINEA=1
             AND ESTAT.IDESTAT=LINEA_TIPUS.IDESTATTLINEA;


        SELECT COUNT (*)INTO n_registres
          FROM LINEA_TIPUS
          WHERE LINEA_TIPUS.IDTLINEA=p_idTipoLinea;

            If n_registres=0 then
             DBMS_OUTPUT.ENABLE;
             DBMS_OUTPUT.PUT_LINE('MODIFICAR LINEA TIPUS: ');
             s_rsp := 'Tipus Linea no existent a la BBDD';
             DBMS_OUTPUT.PUT_LINE(s_rsp);
             RAISE e_tipus_linea;
            END IF;

      s_sql := 'UPDATE LINEA_TIPUS SET ';
     -- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
     IF p_dataEstatTipoLinea IS NOT NULL THEN
      -- Modifiquem Data Estat Tipo Linea
      s_sql := s_sql || ' DATAESTATTLINEA='' ||p_dataEstatTipoLinea|| '',';
     END IF;
     IF p_consumMaximTipoLinea IS NOT NULL THEN
      -- Modifiquem el Consum Maxim Tipo Linea
      s_sql := s_sql || 'CONSUMMAXIMTLINEA='' ||p_consumMaximTipoLinea|| '',';
    END IF;
    IF p_observacionTipoLinea IS NOT NULL THEN
    -- Modifiquem Observacions Tipo Linea
    s_sql := s_sql || 'OBSERVACIOTLINEA='' ||p_observacionTipoLinea|| '',';
    END IF;
  -- Eliminem la coma final de la sentencia SQL-------------------------
    s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE-----------------------
  s_sql := s_sql || ' WHERE IDTLINEA='' ||n_idTipoLinea || '';
    EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0 THEN
--L'error serÃ  que Tipo Linea ja el tenim en la taula i no admet duplicats--
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR LINEA TIPUS: ');
    s_rsp := 'Actualitzacio no realitzada';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_tipus_linea;
   ELSE
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR_LINEA_TIPUS ');
    DBMS_OUTPUT.PUT_LINE('Actualitzacio de la linea: '||p_idTipoLinea||
    ' en data '||p_dataEstatTipoLinea||
    ' en un consum maxim tolerable de '||p_consumMaximTipoLinea||
    ' Observacions. '||p_observacionTipoLinea||' ha estat satisfactoria.');
```

```
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
    s_rsp := 'OK Actualitzacio de la linea: '||p_idTipoLinea||
    ' en data '||p_dataEstatTipoLinea||
    ' en un consum maxim tolerable de '||p_consumMaximTipoLinea||
    ' Observacions. '||p_observacionTipoLinea||' ha estat satisfactoria.';
    DBMS_OUTPUT.put_line ('OK Actualitzacio de la linea: '||p_idTipoLinea||
    ' en data '||p_dataEstatTipoLinea||
    ' en un consum maxim tolerable de '||p_consumMaximTipoLinea||
    ' Observacions. '||p_observacionTipoLinea||' ha estat satisfactoria.');
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;
COMMIT;
  EXCEPTION
    WHEN DUP_VAL_ON_INDEX THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR_LINEA_TIPUS ');
      s_rsp := 'Error: Tipo Linea duplicada. No es poden fer les modificacions';
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        ----------------------------------------
        s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
        -- L'error si ha estat controlat per codi
        ----------------------------------------
        s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.put_line (s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
END PRC_MODIFICAR_LINEA_TIPUS;
/*******************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
--  BAIXA LINEA
*******************************************************************************/
PROCEDURE PRC_BAIXA_LINEA_TIPUS(
    p_idTipoLinea in LINEA_TIPUS.IDTLINEA%Type,
    p_estatTipoLinea in LINEA_TIPUS.IDESTATTLINEA%Type,
    p_dataEstatTipoLinea in LINEA_TIPUS.DATAESTATTLINEA%Type,
    p_observacionTipoLinea in LINEA_TIPUS.OBSERVACIOTLINEA%Type,
    s_rsp out NOCOPY VARCHAR2)
 AS

 BEGIN
    c_procesLog := 'PRC_BAIXA_LINEA_TIPUS';
    c_dataHoraLog := SYSDATE;
```

```
    c_entradaLog :='Id Tipus Linea: ' ||p_idTipoLinea||
    ' Data Estat Tipus Linea: ' ||p_dataEstatTipoLinea;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('    BAIXA LINEA TIPUS    ');
    DBMS_OUTPUT.PUT_LINE('_____________________________');
    DBMS_OUTPUT.PUT_LINE('Codi Tipus Linea: ' ||p_idTipoLinea||
    ' Data Estat Tipus Linea: ' ||p_dataEstatTipoLinea);
    c_sortidalog := 's_rsp';
   IF p_idTipoLinea IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('BAIXA LINEA TIPUS  : ');
      s_rsp := 'Falta especificar el codi Tipo Linea ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_tipus_linea;
   END IF;
   IF p_dataEstatTipoLinea IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('BAIXA LINEA TIPUS  : ');
      s_rsp := 'Falta especificar la data de la modificacio de estat de Tipus Linea ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_tipus_linea;
   END IF;
--1er. Comprovem que existeixi Tipo Linea-------------------------------
    SELECT COUNT (*) INTO n_registres
    FROM LINEA_TIPUS
    WHERE  LINEA_TIPUS.IDTLINEA=p_idTipoLinea;
  IF n_registres =0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('BAIXA LINEA TIPUS  : '||p_idTipoLinea||
   ' ha de fer-se una alta de la linea tipus. ');
   s_rsp := 'BAIXA LINEA TIPUS  : '||p_idTipoLinea||
   ' ha de fer-se una alta de la linea tipus. ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_tipus_linea;
  ELSE
  --RECUPEREM LA LINEA TIPUS
  SELECT LINEA_TIPUS.IDTLINEA,
  LINEA_TIPUS.DATAESTATTLINEA,
  LINEA_TIPUS.IDESTATTLINEA
  ESTAT.DESCRIPCIOESTAT,
  LINEA_TIPUS.CONSUMMAXIMTLINEA,
  LINEA_TIPUS.OBSERVACIOTLINEA
  INTO
n_idTipoLinea,n_dataEstatTipoLinea,n_idEstat,n_estatTipoLinea,n_consumMaximTipoLinea,n_observacionTipoLinea
  FROM LINEA_TIPUS,ESTAT
  WHERE LINEA_TIPUS.IDTLINEA=p_idTipoLinea
  AND LINEA_TIPUS.IDESTATTLINEA= ESTAT.IDESTAT;
  n_estatLinea:=2;
      s_sql := 'UPDATE LINEA_TIPUS SET ';
     -- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF (n_idEstat IS NOT NULL) AND (n_idEstat=1) THEN
-- Modifiquem Data Estat Tipo Linea
```

```
  s_sql := s_sql || 'IDESTATTLINEA='' ||n_estatLinea|| '',';
 END IF;
 IF p_dataEstatTipoLinea IS NOT NULL THEN
-- Modifiquem Data Estat Tipo Linea
  s_sql := s_sql || ' DATAESTATTLINEA='' ||p_dataEstatTipoLinea|| '',';
 END IF;

 -- Eliminem la coma final de la sentencia SQL-------------------------
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
 s_sql := s_sql || ' WHERE IDTLINEA='' ||n_idTipoLinea || '''';
   EXECUTE IMMEDIATE s_sql;
IF SQL%ROWCOUNT = 0 THEN
--L'error serÃ  que Tipo Linea ja el tenim en la taula i no admet duplicats--
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('BAIXA DE LA LINEA TIPUS: ');
   s_rsp := 'Actualitzacio no realitzada';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_tipus_linea;
  ELSE
 SELECT LINEA_TIPUS.IDTLINEA,LINEA_TIPUS.DATAESTATTLINEA,
 LINEA_TIPUS.IDESTATTLINEA,ESTAT.DESCRIPCIOESTAT,
 LINEA_TIPUS.CONSUMMAXIMTLINEA,LINEA_TIPUS.OBSERVACIOTLINEA
 INTO  n_idTipoLinea,n_dataEstatTipoLinea,n_estat,n_dataEstatTipoLinea,
 n_consumMaximTipoLinea,n_observacionTipoLinea
 FROM LINEA_TIPUS,ESTAT
 WHERE LINEA_TIPUS.IDTLINEA=p_idTipoLinea
 AND LINEA_TIPUS.IDESTATTLINEA= ESTAT.IDESTAT;

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('BAIXA LINEA TIPUS: ');
   DBMS_OUTPUT.PUT_LINE('Actualitzacio de baixa de la linea: '||p_idTipoLinea||
   ' data de la baixa: '||n_dataEstatTipoLinea||
   ' estat: '||n_estat||
   '=Baixa amb consum maxim tolerable de '||n_consumMaximTipoLinea||
   ' Observacions. '||n_observacionTipoLinea||' ha estat satisfactoria.');
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   s_rsp := 'OK Actualitzacio de la linea: '||p_idTipoLinea||
   ' en data '||n_dataEstatTipoLinea||
   ' en un consum maxim tolerable de '||n_consumMaximTipoLinea||
   ' Observacions. '||n_observacionTipoLinea||' ha estat satisfactoria.';
   DBMS_OUTPUT.put_line ('OK Actualitzacio de la linea: '||p_idTipoLinea||
   ' en data '||n_dataEstatTipoLinea||
   ' en un consum maxim tolerable de '||n_consumMaximTipoLinea||
   ' Observacions. '||n_observacionTipoLinea||' ha estat satisfactoria.');
   pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;
  END IF;
COMMIT;
 EXCEPTION
   WHEN DUP_VAL_ON_INDEX THEN
     DBMS_OUTPUT.ENABLE;
```

```
    DBMS_OUTPUT.PUT_LINE('BAIXA LINEA TIPUS: ');
    s_rsp := 'Error: Tipo Linea duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_BAIXA_LINEA_TIPUS;
--
-- CONSULTA LINEA_TIPUS ESTAT ACTIVAS
PROCEDURE PRC_CONSULTAR_LINEA_ACTIVA(
  s_rsp out NOCOPY VARCHAR2)
AS
CURSOR C_LIENAS_TIPUS_ON IS
   SELECT DISTINCT LINEA_TIPUS.IDTLINEA AS CODI,
        LINEA_TIPUS.DATAESTATTLINEA AS FECHA,
        ESTAT.DESCRIPCIOESTAT AS ESTAT,
        LINEA_TIPUS.CONSUMMAXIMTLINEA AS COSUM_MAXIM_TOLERABLE,
       LINEA_TIPUS.OBSERVACIOTLINEA AS OBSERVACIONS
   FROM LINEA_TIPUS,ESTAT
   WHERE LINEA_TIPUS.IDTLINEA=
   LINEA_TIPUS.IDTLINEA
   AND LINEA_TIPUS.IDESTATTLINEA=1
   AND LINEA_TIPUS.IDESTATTLINEA=ESTAT.IDESTAT
   ORDER BY LINEA_TIPUS.IDTLINEA;

BEGIN
 c_procesLog := 'PRC_CONSULTAR_LINEA_ACTIVA';
 c_dataHoraLog := SYSDATE;
 c_entradaLog :='CONSULTA LINEAS ACTIVADES ';
 c_sortidalog := 's_rsp';
-- Comprovem que id Tipo Linea ESTIGUI EN ESTAT 1=Alta
  SELECT COUNT(LINEA_TIPUS.IDTLINEA) INTO n_idTipoLinea
    FROM LINEA_TIPUS
    WHERE LINEA_TIPUS.IDTLINEA=LINEA_TIPUS.IDTLINEA
    AND LINEA_TIPUS.IDESTATTLINEA=1;
  IF n_idTipoLinea=0 THEN
   DBMS_OUTPUT.ENABLE;
```

```
   DBMS_OUTPUT.PUT_LINE('CONSULTA TIPUS DE LINEAS ACTIVADES ');
   s_rsp := 'No hi han tipus de lineas que estiguin actives ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_tipus_linea;
  END IF;

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('          CONSULTA TIPUS DE LINEAS ACTIVADES          ');
   DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------------');

  OPEN C_LIENAS_TIPUS_ON;
   FETCH C_LIENAS_TIPUS_ON INTO n_idTipoLinea,n_dataEstatTipoLinea,
   n_estat,n_consumMaximTipoLinea,
   n_observacionTipoLinea ;
  WHILE C_LIENAS_TIPUS_ON%FOUND LOOP
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(
 ' Codi Tipus linea:  '
 ||n_idTipoLinea
 ||' Data Activacio: '
 ||n_dataEstatTipoLinea||
 ' Estat: '||n_estat||
 ' Consum maxim tolerable: '
 ||n_consumMaximTipoLinea||
 'Observacions: '||n_observacionTipoLinea);
  s_rsp := 'OK  CONSULTA DE LINEQAS TIPUS ACTIVADES
   Codi Tipus linea:  '
 ||n_idTipoLinea
 ||' Data Activacio: '
 ||n_dataEstatTipoLinea||
 ' Estat: '||n_estat||
 ' Consum maxim tolerable: '
 ||n_consumMaximTipoLinea||
 ' Observacions: '||n_observacionTipoLinea;
  FETCH C_LIENAS_TIPUS_ON INTO n_idTipoLinea,n_dataEstatTipoLinea,
    n_estat,n_consumMaximTipoLinea,
    n_observacionTipoLinea ;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total tipus lineas en estat ON: '||n_idTipoLinea);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_LIENAS_TIPUS_ON;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
     ELSE
        -- L'error si ha estat controlat per codi
```

```
        s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
END PRC_CONSULTAR_LINEA_ACTIVA;

-- CONSULTA LINEA TIPUS ESTAT LINEA NO ACTIVA
PROCEDURE PRC_CONSULTAR_LINEA_NO_ACTIVA(
  s_rsp out NOCOPY VARCHAR2)
AS
    CURSOR C_LIENAS_TIPUS_OFF IS
    SELECT DISTINCT LINEA_TIPUS.IDTLINEA AS CODI,
          LINEA_TIPUS.DATAESTATTLINEA AS FECHA,
          ESTAT.DESCRIPCIOESTAT AS ESTAT,
          LINEA_TIPUS.CONSUMMAXIMTLINEA AS COSUM_MAXIM_TOLERABLE,
         LINEA_TIPUS.OBSERVACIOTLINEA AS OBSERVACIONS
    FROM LINEA_TIPUS,ESTAT
    WHERE LINEA_TIPUS.IDTLINEA=LINEA_TIPUS.IDTLINEA
    AND LINEA_TIPUS.IDESTATTLINEA=2
    AND LINEA_TIPUS.IDESTATTLINEA=ESTAT.IDESTAT
    ORDER BY LINEA_TIPUS.IDTLINEA;

BEGIN
 c_procesLog := 'PRC_CONSULTAR_LINEA_NO_ACTIVA';
 c_dataHoraLog := SYSDATE;
 c_entradaLog :='CONSULTA TIPUS DE LINEAS DESACTIVADES ';
 c_sortidalog := 's_rsp';
-- Comprovem que id Tipo Linea ESTIGUI EN ESTAT 1=Alta
   SELECT COUNT(LINEA_TIPUS.IDTLINEA) INTO n_idTipoLinea
     FROM LINEA_TIPUS
      WHERE LINEA_TIPUS.IDTLINEA=LINEA_TIPUS.IDTLINEA
      AND LINEA_TIPUS.IDESTATTLINEA=2;
   IF n_idTipoLinea=0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTA TIPUS DE LINEAS DESACTIVADES ');
    s_rsp := 'No hi han tipus de lineas que estiguin desactives ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_tipus_linea;
   END IF;

    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          CONSULTA TIPUS DE LINEAS DESACTIVADES          ');
    DBMS_OUTPUT.PUT_LINE('----------------------------------------------------------------');

  OPEN C_LIENAS_TIPUS_OFF;
    FETCH C_LIENAS_TIPUS_OFF INTO n_idTipoLinea,
    n_estat,n_dataEstatTipoLinea,n_consumMaximTipoLinea,
    n_observacionTipoLinea ;
  WHILE C_LIENAS_TIPUS_OFF%FOUND LOOP
```

```
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(
' Codi Tipus linea:  '
||n_idTipoLinea
||' Data Desactivacio: '
||n_dataEstatTipoLinea||
' Estat: '||n_estat||
' Consum maxim tolerable: '
||n_consumMaximTipoLinea||
'Observacions: '||n_observacionTipoLinea);
 s_rsp := 'OK  CONSULTA LINEAS TIPUS NO ACTIVADES
  Codi Tipus linea:  '
||n_idTipoLinea
||' Data Activacio: '
||n_dataEstatTipoLinea||
' Estat: '||n_estat||
' Consum maxim tolerable: '
||n_consumMaximTipoLinea||
' Observacions: '||n_observacionTipoLinea;
 FETCH C_LIENAS_TIPUS_OFF INTO n_idTipoLinea,n_dataEstatTipoLinea,
  n_estat,n_consumMaximTipoLinea,
  n_observacionTipoLinea ;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total tipus lineas en estat OF o BAIXA: '||n_idTipoLinea);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_LIENAS_TIPUS_OFF;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_CONSULTAR_LINEA_NO_ACTIVA;
END "GESTION_LINEA_TIPUS";
```

11.6 TRATAMIENTO DE GESTION TIPO DE INSPECCIONES.

DEFINICION	PROCEDIMIENTO
Alta de Tipo de Inspección	PRC_ALTA_TIPO_INSPECCIO
Modificar Tipo de Inspección	PRC_MODIFICAR_TIPO_INSPECCIO
Consulta Tipo de Inspección	PRC_CONSULTAR_TIPO_INSPECCIO

```
CREATE OR REPLACE
PACKAGE        "GESTION_TIPO_INSPECCIO" AS
PROCEDURE PRC_ALTA_TIPO_INSPECCIO(
    p_descripcioInspeccio in TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%TYPE,
    s_rsp out NOCOPY VARCHAR2);

   PROCEDURE PRC_MODIFICAR_TIPO_INSPECCIO(
    p_descripcioInspeccio in TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%TYPE,
    p_idInspeccio in TIPO_INSPECCIO.IDTINSPCCIO%TYPE,
    s_rsp out NOCOPY VARCHAR2);

  procedure PRC_CONSULTAR_TIPO_INSPECCIO(
    s_rsp out NOCOPY VARCHAR2);
END "GESTION_TIPO_INSPECCIO" ;
```

11.6.1 PROCEDIMIENTO SPL PACKAGE GESTION TIPO DE INSPECCION.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_TIPO_INSPECCIO" AS
 c_procesLog LOG_TFC.procesLog%TYPE;
  c_dataHoraLog LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog LOG_TFC.entradaLog%TYPE;
  c_sortidaLog LOG_TFC.sortidaLog%TYPE;
  s_rsp LOG_TFC.rspLog%TYPE;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);

  n_descripcioInspeccio TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%TYPE;
  n_idInspeccio   TIPO_INSPECCIO.IDTINSPCCIO%TYPE;
  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
```

```
 e_tipo_inspeccio    EXCEPTION;
/*******************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
          NOM:
                    PROCEDIMENT PER DONAR ALTA_TIPO_INSPECCIO

          DESCRIPCIÃ“:
                    Segons el disseny expressat en el model realcional.

*******************************************************************************/
PROCEDURE PRC_ALTA_TIPO_INSPECCIO(
  p_descripcioInspeccio in TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%TYPE,
  s_rsp out NOCOPY VARCHAR2)
 AS

  BEGIN
   c_procesLog := 'PRC_ALTA_TIPO_INSPECCIO';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Decripcio Tipo Inspeccio: ' ||p_descripcioInspeccio;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE INSPECCIONS');
   DBMS_OUTPUT.PUT_LINE('_____________________');
   DBMS_OUTPUT.PUT_LINE('ALTA_TIPO_INSPECCIO: Decripcio Tipo Inspeccio: '
   ||p_descripcioInspeccio);
   c_sortidalog := 's_rsp';
 If (p_descripcioInspeccio IS NULL) then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('ALTA_TIPO_INSPECCIO: ');
        s_rsp := 'Falta especificar tipus dÂ´Inspeccio';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_tipo_inspeccio;
    END IF;
     -- Comprovem si existeix el el tipus de lectura a  donar d'alta
       SELECT COUNT (*)INTO n_registres
         FROM TIPO_INSPECCIO
         WHERE DESCRIPCIOTINSPECCIO = p_descripcioInspeccio;
    If n_registres=0 then
       -- El donem d'alta a la tipus lectura
         INSERT INTO TIPO_INSPECCIO(DESCRIPCIOTINSPECCIO)
         VALUES(p_descripcioInspeccio);
         --Gravem en la taula log-----------------------------------------------------
         DBMS_OUTPUT.PUT_LINE('Insertada el tipus inspeccio: '
         ||p_descripcioInspeccio);
         DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
         s_rsp :='Ok: Insertat satisfactoriament'|| p_descripcioInspeccio;
         pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        ELSE
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('ALTA_TIPO_INSPECCIO: ');
        s_rsp := 'Tipus Inspeccio duplicat ';
```

```
        DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_tipo_inspeccio;
    END IF;
  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_ALTA_TIPO_INSPECCIO;
/*******************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
            NOM:  PRC_MODIFICAR_TIPO_INSPECCIO
          PROCEDIMENT PERMODIFICAR TIPO INSPECCIO

            DESCRIPCIÃ“:
                    Segons el disseny expressat en el model realcional.

*******************************************************************************/
PROCEDURE PRC_MODIFICAR_TIPO_INSPECCIO(
   p_descripcioInspeccio in TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%TYPE,
   p_idInspeccio in TIPO_INSPECCIO.IDTINSPCCIO%TYPE,
   s_rsp out NOCOPY VARCHAR2)
 AS

  BEGIN
   c_procesLog := 'PRC_MODIFICAR_TIPO_INSPECCIO';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Decripcio Tipo Inspeccio: ' ||p_descripcioInspeccio;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('MODIFICAR EL TIPUS DE INSPECCIONS');
   DBMS_OUTPUT.PUT_LINE('_____________________');
   DBMS_OUTPUT.PUT_LINE('Tipus de Inspeccio: '||p_descripcioInspeccio);

   c_sortidalog := 's_rsp';
   -- ComprovaciÃ³ Tipus de Inspeccio-----

   If  (p_descripcioInspeccio  IS NULL) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR EL TIPUS DE INSPECCIONS ');
```

```
      s_rsp := 'Falta especificar tipus Inspeccio ';
      DBMS_OUTPUT.put_line(s_rsp);
      RAISE e_tipo_inspeccio;
  END IF;

  SELECT COUNT (*) INTO n_registres
  FROM TIPO_INSPECCIO
  WHERE TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO=p_descripcioInspeccio;

  If n_registres=0 then
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR_TIPO_INSPECCIO:  ');
     s_rsp := 'TIPO_INSPECCIO no existeix a la BBDD';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_tipo_inspeccio;
  END IF;

-- recuperem el codi de la tipus lectura
  SELECT TIPO_INSPECCIO.IDTINSPCCIO,
  TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO INTO n_idInspeccio,n_descripcioInspeccio
  FROM TIPO_INSPECCIO
  WHERE TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO=p_descripcioInspeccio;

  s_sql := 'UPDATE TIPO_INSPECCIO SET ';
-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF p_descripcioInspeccio  IS NOT NULL THEN
 -- Modifiquem el descripcio pasis
  s_sql := s_sql || 'DESCRIPCIOTINSPECCIO=''' ||p_descripcioInspeccio || ''',';
  END IF;
  -- Eliminem la coma final de la sentencia SQL
  s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
  -- Afegim a la sentencia la condiciÃ³ del WHERE
 s_sql := s_sql || ' WHERE IDTINSPECCIO=''' ||n_idInspeccio|| '''';

  EXECUTE IMMEDIATE s_sql;

 IF SQL%ROWCOUNT = 0 THEN
  --L'error serÃ  que el la via ja el tenim en la taula i no admet duplicats--
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' MODIFICAR_TIPO_INSPECCIO ');
    s_rsp := 'Actualitzacio no realitzada';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_tipo_inspeccio;
 ELSE
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('Modificacio feta del Tipus Inspeccio
      Codi Tipus Inspeccio: '||n_idInspeccio||
     ' Descripcio: '||p_descripcioInspeccio );
     DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
     DBMS_OUTPUT.put_line ('OK: Modificacio feta del Tipus de Inspeccio
     de la Central de Produccio: codi: '||n_idInspeccio||
     ' Descripcio: '||p_descripcioInspeccio);
```

```
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;

EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR_TIPO_INSPECCIO ');
    s_rsp := 'Error: Tipus Inspeccions duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
 END PRC_MODIFICAR_TIPO_INSPECCIO;
/*******************************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
            NOM:  PRC_CONSULTAR_TIPO_INSPECCIO
          PROCEDIMENT PER CONSULTAR TIPO INSPECCIO

            DESCRIPCIÃ“:
                    Segons el disseny expressat en el model realcional.

*******************************************************************************************/
procedure PRC_CONSULTAR_TIPO_INSPECCIO(
      s_rsp out NOCOPY VARCHAR2)
AS
CURSOR C_TIPO_INSPECCIO IS
SELECT TIPO_INSPECCIO.IDTINSPCCIO,
TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO
FROM TIPO_INSPECCIO
WHERE TIPO_INSPECCIO.IDTINSPCCIO=TIPO_INSPECCIO.IDTINSPCCIO;

BEGIN
   c_procesLog := 'PRC_CONSULTAR_TIPO_INSPECCIO';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='CONSULTA TIPUS DE FUNCIONS DE LES CENTRALS DE PRODUCCIO';
   c_sortidalog := 's_rsp';
```

```
--1er. Comprovem que existeixi Tipus de Funcion------------------------------
  SELECT COUNT (*) INTO n_registres
  FROM TIPO_INSPECCIO
  WHERE TIPO_INSPECCIO.IDTINSPCCIO=TIPO_INSPECCIO.IDTINSPCCIO;

  IF n_registres=0 THEN
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('CONSULTA TIPUS INSPECCIONS ');
  s_rsp := 'No hi han tipus inspeccions donades d´alta ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_tipo_inspeccio;
  END IF;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('          CONSULTA TIPUS INSPECCIONS           ');
  DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------');
  OPEN C_TIPO_INSPECCIO;
  FETCH C_TIPO_INSPECCIO INTO n_idInspeccio,n_descripcioInspeccio;
 WHILE C_TIPO_INSPECCIO%FOUND LOOP
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(
'   Codi Tipus Inspeccions:  '
|| n_idInspeccio
||' Descripcio '
|| n_descripcioInspeccio);
 s_rsp := 'OK  CONSULTA INSPECCIONS
 Codi Tipus Inspeccio:  '
|| n_idInspeccio
||' Descripcio: '
||n_descripcioInspeccio;
 FETCH C_TIPO_INSPECCIO INTO  n_idInspeccio,n_descripcioInspeccio;

 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total tipus Inspeccions: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_TIPO_INSPECCIO;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
```

```
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  END PRC_CONSULTAR_TIPO_INSPECCIO;
END "GESTION_TIPO_INSPECCIO";
```

11.7 TRATAMIENTO DE GESTION INSPECCIONES.

DEFINICION	PROCEDIMIENTO
Alta de Inspección	PRC_ALTA_INSPECCIONS
Modificar Inspección	PRC_MODIFICAR_INSPECCIONS
Consulta Inspección	PRC_CONSULTAR_INSPECCIONS

```
CREATE OR REPLACE
PACKAGE        "GESTION_INPECCIONS" AS
--- ALTA INPECCIONS
PROCEDURE PRC_ALTA_INSPECCIONS(
   p_dataInspeccio in INSPECCIO.DATAINSPECCIO%Type,
   p_codiInspector in INSPECCIO.CODIINSPECTOR%Type,
   p_resultatInspeccio in INSPECCIO.IDSUPERAINSPECCIO%TYPE,
   p_observacionsInspeccio in INSPECCIO.OBSERVACIONSINSPECCIO%Type,
   s_rsp out NOCOPY VARCHAR);

--- MODIFICAR INPECCIONS
PROCEDURE PRC_MODIFICAR_INSPECCIONS(
   p_idInspeccio in INSPECCIO.IDINSPECCIO%Type,
   p_dataInspeccio in INSPECCIO.DATAINSPECCIO%Type,
   p_codiInspector in INSPECCIO.CODIINSPECTOR%Type,
    p_resultatInspeccio in INSPECCIO.IDSUPERAINSPECCIO%TYPE,
   p_observacionsInspeccio in INSPECCIO.OBSERVACIONSINSPECCIO%Type,
   s_rsp out NOCOPY VARCHAR2);

-- CONSULTA INPECCIONS
PROCEDURE PRC_CONSULTAR_INSPECCIONS(
  s_rsp out NOCOPY VARCHAR2);
END "GESTION_INPECCIONS";
```

11.7.1 PROCEDIMIENTO SPL PACKAGE GESTION INSPECCIONES.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_TIPO_INSPECCIO" AS
 c_procesLog LOG_TFC.procesLog%TYPE;
  c_dataHoraLog LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog LOG_TFC.entradaLog%TYPE;
  c_sortidaLog LOG_TFC.sortidaLog%TYPE;
  s_rsp LOG_TFC.rspLog%TYPE;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);

  n_descripcioInspeccio TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%TYPE;
  n_idInspeccio   TIPO_INSPECCIO.IDTINSPCCIO%TYPE;
  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_tipo_inspeccio     EXCEPTION;
/*******************************************************************************
Autor: Eduard Monzonis Hierro                       UOC
18/05/2012                              TFC: CONTROL ENERGIA.
            NOM:
                        PROCEDIMENT PER DONAR ALTA_TIPO_INSPECCIO

            DESCRIPCIÃ“:
                        Segons el disseny expressat en el model realcional.

*******************************************************************************/
PROCEDURE PRC_ALTA_TIPO_INSPECCIO(
   p_descripcioInspeccio in TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%TYPE,
   s_rsp out NOCOPY VARCHAR2)
 AS

  BEGIN
   c_procesLog := 'PRC_ALTA_TIPO_INSPECCIO';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Decripcio Tipo Inspeccio: ' ||p_descripcioInspeccio;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('ALTA TIPUS DE INSPECCIONS');
   DBMS_OUTPUT.PUT_LINE('_____________________');
   DBMS_OUTPUT.PUT_LINE('ALTA_TIPO_INSPECCIO: Decripcio Tipo Inspeccio: '
   ||p_descripcioInspeccio);
   c_sortidalog := 's_rsp';
 If (p_descripcioInspeccio IS NULL) then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('ALTA_TIPO_INSPECCIO: ');
        s_rsp := 'Falta especificar tipus dÂ´Inspeccio';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_tipo_inspeccio;
    END IF;
     -- Comprovem si existeix el el tipus de lectura a  donar d'alta
       SELECT COUNT (*)INTO n_registres
```

```
        FROM TIPO_INSPECCIO
        WHERE DESCRIPCIOTINSPECCIO = p_descripcioInspeccio;
    If n_registres=0 then
      -- El donem d'alta a la tipus lectura
        INSERT INTO TIPO_INSPECCIO(DESCRIPCIOTINSPECCIO)
        VALUES(p_descripcioInspeccio);
        --Gravem en la taula log----------------------------------------------------
        DBMS_OUTPUT.PUT_LINE('Insertada el tipus inspeccio: '
        ||p_descripcioInspeccio);
        DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
        s_rsp :='Ok: Insertat satisfactoriament'|| p_descripcioInspeccio;
        pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       ELSE
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('ALTA_TIPO_INSPECCIO: ');
       s_rsp := 'Tipus Inspeccio duplicat ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_tipo_inspeccio;
    END IF;
  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_ALTA_TIPO_INSPECCIO;
/*******************************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
          NOM:  PRC_MODIFICAR_TIPO_INSPECCIO
         PROCEDIMENT PERMODIFICAR TIPO INSPECCIO

          DESCRIPCIÃ“:
                Segons el disseny expressat en el model realcional.

*******************************************************************************************/
PROCEDURE PRC_MODIFICAR_TIPO_INSPECCIO(
   p_descripcioInspeccio in TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%TYPE,
   p_idInspeccio in TIPO_INSPECCIO.IDTINSPCCIO%TYPE,
   s_rsp out NOCOPY VARCHAR2)
 AS
```

```
 BEGIN
  c_procesLog := 'PRC_MODIFICAR_TIPO_INSPECCIO';
  c_dataHoraLog := SYSDATE;
  c_entradaLog :='Decripcio Tipo Inspeccio: ' ||p_descripcioInspeccio;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('MODIFICAR EL TIPUS DE INSPECCIONS');
  DBMS_OUTPUT.PUT_LINE('_____________________');
  DBMS_OUTPUT.PUT_LINE('Tipus de Inspeccio: '||p_descripcioInspeccio);

  c_sortidalog := 's_rsp';
  -- ComprovaciÃ³ Tipus de Inspeccio-----

  If  (p_descripcioInspeccio  IS NULL) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR EL TIPUS DE INSPECCIONS ');
     s_rsp := 'Falta especificar tipus Inspeccio ';
     DBMS_OUTPUT.put_line(s_rsp);
     RAISE e_tipo_inspeccio;
 END IF;

 SELECT COUNT (*) INTO n_registres
 FROM TIPO_INSPECCIO
 WHERE TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO=p_descripcioInspeccio;

 If n_registres=0 then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR_TIPO_INSPECCIO:  ');
    s_rsp := 'TIPO_INSPECCIO no existeix a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_tipo_inspeccio;
 END IF;

-- recuperem el codi de la tipus lectura
 SELECT TIPO_INSPECCIO.IDTINSPCCIO,
 TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO INTO n_idInspeccio,n_descripcioInspeccio
 FROM TIPO_INSPECCIO
 WHERE TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO=p_descripcioInspeccio;

 s_sql := 'UPDATE TIPO_INSPECCIO SET ';
-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
 IF p_descripcioInspeccio  IS NOT NULL THEN
 -- Modifiquem el descripcio pasis
 s_sql := s_sql || 'DESCRIPCIOTINSPECCIO=''' ||p_descripcioInspeccio || ''',';
 END IF;
 -- Eliminem la coma final de la sentencia SQL
 s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
 -- Afegim a la sentencia la condiciÃ³ del WHERE
s_sql := s_sql || ' WHERE IDTINSPECCIO=''' ||n_idInspeccio|| '''';

 EXECUTE IMMEDIATE s_sql;
```

```
 IF SQL%ROWCOUNT = 0 THEN
  --L'error serÃ  que el la via ja el tenim en la taula i no admet duplicats--
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' MODIFICAR_TIPO_INSPECCIO ');
    s_rsp := 'Actualitzacio no realitzada';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_tipo_inspeccio;
 ELSE
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('Modificacio feta del Tipus Inspeccio
      Codi Tipus Inspeccio: '||n_idInspeccio||
     ' Descripcio: '||p_descripcioInspeccio );
     DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
     DBMS_OUTPUT.put_line ('OK: Modificacio feta del Tipus de Inspeccio
     de la Central de Produccio: codi: '||n_idInspeccio||
     ' Descripcio: '||p_descripcioInspeccio);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;

EXCEPTION
   WHEN DUP_VAL_ON_INDEX THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR_TIPO_INSPECCIO ');
      s_rsp := 'Error: Tipus Inspeccions duplicada. No es poden fer les modificacions';
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
   WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         ----------------------------------------
         s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
         -- L'error si ha estat controlat per codi
         ----------------------------------------
         s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.put_line (s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;
  END PRC_MODIFICAR_TIPO_INSPECCIO;
/*******************************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
            NOM:  PRC_CONSULTAR_TIPO_INSPECCIO
          PROCEDIMENT PER CONSULTAR TIPO INSPECCIO

            DESCRIPCIÃ“:
```

```
                    Segons el disseny expressat en el model realcional.

*********************************************************************************************/
procedure PRC_CONSULTAR_TIPO_INSPECCIO(
      s_rsp out NOCOPY VARCHAR2)
AS
CURSOR C_TIPO_INSPECCIO IS
SELECT TIPO_INSPECCIO.IDTINSPCCIO,
TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO
FROM TIPO_INSPECCIO
WHERE TIPO_INSPECCIO.IDTINSPCCIO=TIPO_INSPECCIO.IDTINSPCCIO;

BEGIN
   c_procesLog := 'PRC_CONSULTAR_TIPO_INSPECCIO';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='CONSULTA TIPUS DE FUNCIONS DE LES CENTRALS DE PRODUCCIO';
   c_sortidalog := 's_rsp';

 --1er. Comprovem que existeixi Tipus de Funcion------------------------------
   SELECT COUNT (*) INTO n_registres
   FROM TIPO_INSPECCIO
   WHERE TIPO_INSPECCIO.IDTINSPCCIO=TIPO_INSPECCIO.IDTINSPCCIO;

   IF n_registres=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTA TIPUS INSPECCIONS ');
   s_rsp := 'No hi han tipus inspeccions donades d´alta ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_tipo_inspeccio;
   END IF;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('          CONSULTA TIPUS INSPECCIONS           ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------');
   OPEN C_TIPO_INSPECCIO;
   FETCH C_TIPO_INSPECCIO INTO n_idInspeccio,n_descripcioInspeccio;
 WHILE C_TIPO_INSPECCIO%FOUND LOOP
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(
'   Codi Tipus Inspeccions:  '
|| n_idInspeccio
||' Descripcio '
|| n_descripcioInspeccio);
 s_rsp := 'OK  CONSULTA INSPECCIONS
 Codi Tipus Inspeccio:  '
|| n_idInspeccio
||' Descripcio: '
||n_descripcioInspeccio;
 FETCH C_TIPO_INSPECCIO INTO  n_idInspeccio,n_descripcioInspeccio;

 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 END LOOP;
```

```
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total tipus Inspeccions: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_TIPO_INSPECCIO;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  END PRC_CONSULTAR_TIPO_INSPECCIO;
END "GESTION_TIPO_INSPECCIO";
```

11.8 TRATAMIENTO DE GESTION TIPO DE MODELO.

DEFINICION	PROCEDIMIENTO
Alta de Modelo	PRC_ALTA_MODELO
Modificar Modelo	PRC_MODIFICAR_MODELO
Baixa de Modelo	PRC_BAIXA_MODELO
Consulta Modelo	PRC_CONSULTAR_MODELO
Consulta línea Pasiva	PRC_CONSULTAR_LINEA_NO_ACTIVA

```
CREATE OR REPLACE
PACKAGE        "GESTION_MODELO" AS
PROCEDURE PRC_ALTA_MODELO(
   p_descripcioModelo in MODELO.DESCRIPCIOMODELO%Type,
   p_anyModelo in MODELO.ANYOFABRICAMODELO%Type,
   p_idFabricantModelo in MODELO.IDFABRICANTMODELO%Type,
   p_dataAltaModelo in MODELO.DATAALTAMODELO%Type,
   p_estatModelo in MODELO.ESTATMODELO%Type,
   p_observacionsModelo in MODELO.OBSERVACIONSMODELO%Type,
   s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_MODIFICAR_MODELO(
   p_idModelo in MODELO.IDMODELO%Type,
   p_descripcioModelo in MODELO.DESCRIPCIOMODELO%Type,
   p_anyModelo in MODELO.ANYOFABRICAMODELO%Type,
   p_idFabricantModelo in MODELO.IDFABRICANTMODELO%Type,
   p_dataModificaModelo in MODELO.DATAMODIFICACIOMODELO%Type,
   p_estatModelo in MODELO.ESTATMODELO%Type,
   p_observacionsModelo in MODELO.OBSERVACIONSMODELO%Type,
   s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_BAIXA_MODELO(
   p_idModelo in MODELO.IDMODELO%Type,
   p_descripcioModelo in MODELO.DESCRIPCIOMODELO%Type,
   p_estatModelo in MODELO.ESTATMODELO%Type,
   p_dataModificaModelo in MODELO.DATAMODIFICACIOMODELO%Type,
   s_rsp out NOCOPY VARCHAR2);
PROCEDURE PRC_CONSULTAR_MODELO(
  s_rsp out NOCOPY VARCHAR2);
END  "GESTION_MODELO";
```

11.8.1 PROCEDIMIENTO SPL PACKAGE GESTION MODELO.

```
CREATE OR REPLACE
PACKAGE BODY         "GESTION_MODELO" AS
c_procesLog             LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp               LOG_TFC.rspLog%TYPE;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);

  n_idModelo  MODELO.IDMODELO%Type;
  n_descripcioModelo MODELO.DESCRIPCIOMODELO%Type;
  n_estatModelo  MODELO.ESTATMODELO%Type;
  n_anyModelo  MODELO.ANYOFABRICAMODELO%Type;
  n_idFabricantModelo  MODELO.IDFABRICANTMODELO%Type;
  n_observacionsModelo  MODELO.OBSERVACIONSMODELO%Type;
  n_dataModificacioModel MODELO.DATAMODIFICACIOMODELO%TYPE;
  n_dataAltaModel MODELO.DATAALTAMODELO%TYPE;
```

```
  n_fabricant FABRICANT.NOMCOMERCIALFABRICANT%TYPE;
  n_idEstatModelo MODELO.ESTATMODELO%TYPE;
  n_controlEstat NUMBER;
  n_descModel ESTAT.DESCRIPCIOESTAT%TYPE;

  n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_model     EXCEPTION;
----------------------------------------------------------------
/*****************************************************************************
Autor: Eduard Monzonis Hierro                          UOC
18/05/2012                                  TFC: CONTROL ENERGIA.
          PROCEDIMENT ALTA DE MODEL DE COMPTADOR
DESCRIPCIO.
Gestiona els models de COMPTADORS i depen de FABRICANTS. Els comptadors
tene definides les propietats fÃsiques que aquest procediment insereix.
*****************************************************************************/
PROCEDURE PRC_ALTA_MODELO(
    p_descripcioModelo in MODELO.DESCRIPCIOMODELO%Type,
    p_anyModelo in MODELO.ANYOFABRICAMODELO%Type,
    p_idFabricantModelo in MODELO.IDFABRICANTMODELO%Type,
    p_dataAltaModelo in MODELO.DATAALTAMODELO%Type,
    p_estatModelo in MODELO.ESTATMODELO%Type,
    p_observacionsModelo in MODELO.OBSERVACIONSMODELO%Type,
    s_rsp out NOCOPY VARCHAR2)
AS

BEGIN
    c_procesLog := 'PRC_ALTA_MODELO';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='Decripcio Model: ' ||p_descripcioModelo ||
    ' , Any Model: ' ||p_anyModelo||
    ' , id Fabricant Model: ' || p_idFabricantModelo ||
    ' , Data Alta Model: ' || p_dataAltaModelo||
    ' , Observacions Modelo : ' ||p_observacionsModelo;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.PUT_LINE('     ALTA MODELS            ');
      DBMS_OUTPUT.PUT_LINE('_____________________________');
      DBMS_OUTPUT.PUT_LINE(' Decripcio Model: ' ||p_descripcioModelo ||
    ' , Any Model: ' ||  p_anyModelo||
    ' , id Fabricant Model: ' || p_idFabricantModelo ||
    ' , Data Alta Model: ' || p_dataAltaModelo||
    ' , Observacions Modelo : ' ||p_observacionsModelo);
    c_sortidalog := 's_rsp';
    n_estatModelo:=1;
 If p_descripcioModelo IS NULL then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ALTA MODELS: ');
      s_rsp := 'Falta especificar Decripcio Model ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
```

```
    RAISE e_model;
  END IF;
  If p_anyModelo IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('ALTA MODELS: ');
    s_rsp := 'Falta especificar Any Model ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_model;
  END IF;
-- Comprovem si existeix el estat  donar d'alta
  SELECT COUNT (*)INTO n_registres
  FROM MODELO
  WHERE MODELO.DESCRIPCIOMODELO = p_descripcioModelo
  AND MODELO.ANYOFABRICAMODELO= p_anyModelo
  AND MODELO.DATAALTAMODELO=p_dataAltaModelo
  AND MODELO.IDFABRICANTMODELO=p_idFabricantModelo;

  If n_registres=0 then
-- El donem d'alta a la estat
--POSEM L'ESTAT A 1=ALTA

  INSERT INTO MODELO (DESCRIPCIOMODELO,
          ANYOFABRICAMODELO,
          IDFABRICANTMODELO,
          DATAALTAMODELO,
          ESTATMODELO,
          OBSERVACIONSMODELO)
          VALUES(p_descripcioModelo,
          p_anyModelo,
          p_idFabricantModelo,
          p_dataAltaModelo,
          n_estatModelo,
          p_observacionsModelo);
--Gravem en la taula log----------------------------------------------------
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Insertada model:
          Descripcion: '||p_descripcioModelo||
          ' Data alta: '||p_dataAltaModelo||
          ' Any model:  '||p_anyModelo||
          ' Observacions: '||p_observacionsModelo);
          DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
          s_rsp :='Ok: Insertada model:
          Descripcion: '||p_descripcioModelo||
          ' Data alta: '||p_dataAltaModelo||
          ' Any model:  '||p_anyModelo||
          ' Observacions: '||p_observacionsModelo;
          pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
          DBMS_OUTPUT.PUT_LINE(s_rsp);
        ELSE
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('     ALTA MODELS           ');
          DBMS_OUTPUT.PUT_LINE('_____________________________');
```

```
        s_rsp := 'DADES duplicades';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_model;
    END IF;
COMMIT;
 EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
 END PRC_ALTA_MODELO;
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
          PRCEDIMENT MODIFICAR MODEL COMPTADOR
DESCRIPCIO:
Procediement per modificar les dades del model de comptador.
*****************************************************************************/
 PROCEDURE PRC_MODIFICAR_MODELO(
   p_idModelo in MODELO.IDMODELO%Type,
   p_descripcioModelo in MODELO.DESCRIPCIOMODELO%Type,
   p_anyModelo in MODELO.ANYOFABRICAMODELO%Type,
   p_idFabricantModelo in MODELO.IDFABRICANTMODELO%Type,
   p_dataModificaModelo in MODELO.DATAMODIFICACIOMODELO%Type,
   p_estatModelo in MODELO.ESTATMODELO%Type,
   p_observacionsModelo in MODELO.OBSERVACIONSMODELO%Type,
   s_rsp out NOCOPY VARCHAR2)
AS

BEGIN
   c_procesLog := 'PRC_MODIFICAR_MODELO';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Codi Model: ' ||p_idModelo||
   ', Descripcio Model: ' ||p_descripcioModelo||
   ', Any Model: ' ||p_anyModelo||
   ', Data Modifica Model: ' ||p_dataModificaModelo||
   ', Observacions Modelo  '||p_observacionsModelo;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(' ');
 DBMS_OUTPUT.PUT_LINE('     MODIFICAR UN MODEL            ');
 DBMS_OUTPUT.PUT_LINE('_____________________________________');
 DBMS_OUTPUT.PUT_LINE(' Codi Modelo: ' ||p_idModelo||
   ', Descripcio Modelo: ' ||p_descripcioModelo||
```

```
    ', Any Modelo: ' ||p_anyModelo||
    ', Data Modifica Modelo: ' ||p_dataModificaModelo||
    ', Observacions Modelo  '||p_observacionsModelo);
      c_sortidalog := 's_rsp';
-- comprovacio de dades
  IF p_idModelo IS NULL THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR UN MODEL: ');
   s_rsp := 'Falta especificar el el codi Modelo ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_model;
  END IF;
  If p_descripcioModelo IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UN MODEL: ');
    s_rsp := 'Falta especificar descripcio Modelo ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_model;
  END IF;
  If p_anyModelo IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UN MODEL: ');
    s_rsp := 'Falta especificar any Modelo ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_model;
  END IF;
  If p_dataModificaModelo IS NULL then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UN MODEL: ');
    s_rsp := 'Falta especificar data Modificacio Modelo ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_model;
   END IF;
-- recuperem el codi MODELO
  SELECT MODELO.IDMODELO INTO n_idModelo
  FROM MODELO
  WHERE MODELO.IDMODELO=p_idModelo;

  If n_registres=0 then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR UN MODEL: ');
    s_rsp := 'MODELO no existent a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_model;
  END IF;
  s_sql := 'UPDATE MODELO SET ';

-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF p_descripcioModelo IS NOT NULL THEN
-- Modifiquem descripcio Modelo
    s_sql := s_sql || 'DESCRIPCIOMODELO="' ||p_descripcioModelo|| '",';
  END IF;
```

```
 IF p_anyModelo IS NOT NULL THEN
-- Modifiquem any Modelo
  s_sql := s_sql || 'ANYOFABRICAMODELO='' ||p_anyModelo|| '','';
 END IF;
 IF p_dataModificaModelo IS NOT NULL THEN
-- Modifiquem data Modificacio Modelo
  s_sql := s_sql || 'DATAMODIFICACIOMODELO='' ||p_dataModificaModelo|| '','';
 END IF;
 IF p_observacionsModelo IS NOT NULL THEN
-- Modifiquem Observacions Modelo
  s_sql := s_sql || 'OBSERVACIONSMODELO='' ||p_observacionsModelo|| '','';
 END IF;
-- Eliminem la coma final de la sentencia SQL-------------------------
 s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
 s_sql := s_sql || ' WHERE IDMODELO='' ||p_idModelo|| '''';

EXECUTE IMMEDIATE s_sql;

 IF SQL%ROWCOUNT = 0 THEN
--L'error serÃ  que Tipo Linea ja el tenim en la taula i no admet duplicats--
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR UN MODEL: ');
  s_rsp := 'Actualitzacio no realitzada';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_model;
 ELSE
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR UN MODEL: ');
  DBMS_OUTPUT.PUT_LINE('Actualitzacio Model amb el
  Codi Model: '||p_idModelo||
  ' Descripcio del Model: '||p_descripcioModelo||
  ' Any del Model: '||p_anyModelo||
  ' Data de la modificacio: '||p_dataModificaModelo||
  ' Observacions. '||p_observacionsModelo||' ha estat satisfactoria.');
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  s_rsp := 'OK :Actualitzacio Model amb el
  Codi Model: '||p_idModelo||
  ' Descripcio del Model: '||p_descripcioModelo||
  ' Any del Model: '||p_anyModelo||
  ' Data de la modificacio: '||p_dataModificaModelo||
  ' Observacions. '||p_observacionsModelo||' ha estat satisfactoria.';
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;
 EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
```

```
        s_rsp := 'Error: ' || s_rsp;
      END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.put_line(s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;
END PRC_MODIFICAR_MODELO;
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
              PROCEDIMENT BAIXA MODEL COMPTADOR
DESCRIPCIO:
Consulta d'un model de comptador.
*****************************************************************************/
PROCEDURE PRC_BAIXA_MODELO(
    p_idModelo in MODELO.IDMODELO%Type,
    p_descripcioModelo in MODELO.DESCRIPCIOMODELO%Type,
    p_estatModelo in MODELO.ESTATMODELO%Type,
    p_dataModificaModelo in MODELO.DATAMODIFICACIOMODELO%Type,
    s_rsp out NOCOPY VARCHAR2)
AS

BEGIN
  c_procesLog := 'PRC_BAIXA_MODELO';
  c_dataHoraLog := SYSDATE;
  c_entradaLog :='Id Tipus Linea: ' ||p_idModelo||
  ' Data Estat Tipus Linea: ' ||p_dataModificaModelo;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('      BAIXA  MODEL        ');
  DBMS_OUTPUT.PUT_LINE('_____________________________');
  DBMS_OUTPUT.PUT_LINE('Codi Model: ' ||p_idModelo||
  ' Data modificacio: ' ||p_dataModificaModelo);
  c_sortidalog := 's_rsp';
--controlem dades
 IF (p_idModelo IS NULL) OR (p_idModelo=0) THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('BAIXA  MODEL : ');
    s_rsp := 'Falta especificar el codi del model o bé el model s´ha posat 0  ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_model;
  END IF;
  IF p_dataModificaModelo IS NULL THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('BAIXA  MODEL : ');
    s_rsp := 'Falta especificar la data de la modificacio. ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_model;
  END IF;
--1er. Comprovem que existeixi el Model------------------------------
   SELECT COUNT (*) INTO n_registres
   FROM MODELO
```

```
   WHERE MODELO.IDMODELO=p_idModelo;
-- ISERTEM SI NO EXISTEIX
 IF p_idModelo=0 THEN
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('BAIXA  MODEL: '||p_idModelo||
  ' ha de fer-se una alta del model. ');
  s_rsp := 'BAIXA  MODEL: '||p_idModelo||
  ' ha de fer-se una alta del model. ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_model;
 END IF;
  --RECUPEREM EL MODEL
 SELECT
 MODELO.IDMODELO,
 MODELO.DATAMODIFICACIOMODELO,
 MODELO.ANYOFABRICAMODELO,
 MODELO.ESTATMODELO,
 MODELO.DESCRIPCIOMODELO,
 MODELO.IDFABRICANTMODELO,
 MODELO.OBSERVACIONSMODELO
 INTO n_idModelo,
     n_dataModificacioModel,
     n_anyModelo,
     n_idEstatModelo,
     n_descripcioModelo,
     n_idFabricantModelo,
     n_observacionsModelo
 FROM MODELO
 WHERE MODELO.IDMODELO=p_idModelo;

 SELECT FABRICANT.NOMCOMERCIALFABRICANT INTO n_fabricant
 FROM FABRICANT, MODELO
 WHERE FABRICANT.IDFABRICANT= MODELO.IDFABRICANTMODELO;

 -- posem el control Estat e 2=BAIXA
 n_controlEstat:=2;
-- Actualitzem
 s_sql := 'UPDATE MODELO SET ';
    -- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF (n_idEstatModelo IS NOT NULL) AND (n_idEstatModelo=1) THEN
-- Modifiquem Data Estat model
    s_sql := s_sql || 'ESTATMODELO='' ||n_controlEstat|| '',';
  END IF;
  IF p_dataModificaModelo IS NOT NULL THEN
-- Modifiquem Data Estat model
   s_sql := s_sql || 'DATAMODIFICACIOMODELO='' ||p_dataModificaModelo|| '',';
  END IF;
 -- Eliminem la coma final de la sentencia SQL-------------------------
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
 s_sql := s_sql || ' WHERE IDMODELO='' ||p_idModelo|| '';
```

```
EXECUTE IMMEDIATE s_sql;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('BAIXA  MODEL: ');
 DBMS_OUTPUT.PUT_LINE('Actualitzacio de baixa model: '||p_idModelo||
   ' data de la baixa: '||p_dataModificaModelo||
   ' estat: '||n_controlEstat||
   '=Baixa del Model: '||n_descripcioModelo||
   ' del Fabricant: '||n_fabricant||
   ' Observacions. '||n_observacionsModelo||' ha estat satisfactoria.');
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   s_rsp := 'OK Actualitzacio de baixa model: '||p_idModelo||
   ' data de la baixa: '||p_dataModificaModelo||
   ' estat: '||n_controlEstat||
   '=Baixa del Model: '||n_descripcioModelo||
   ' del Fabricant: '||n_fabricant||
   ' Observacions. '||n_observacionsModelo||' ha estat satisfactoria.';
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
COMMIT;
 EXCEPTION
   WHEN DUP_VAL_ON_INDEX THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('BAIXA LINEA TIPUS: ');
     s_rsp := 'Error: Tipo Linea duplicada. No es poden fer les modificacions';
     DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;
  WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        ----------------------------------------
        s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
        -- L'error si ha estat controlat per codi
        ----------------------------------------
        s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line (s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;

END PRC_BAIXA_MODELO;

PROCEDURE PRC_CONSULTAR_MODELO(
  s_rsp out NOCOPY VARCHAR2)
AS

CURSOR C_MODELO IS
SELECT MODELO.IDMODELO AS CODI,
      MODELO.DATAALTAMODELO AS DATA_ALTA,
      ESTAT.DESCRIPCIOESTAT AS ESTAT,
```

```
    MODELO.DATAMODIFICACIOMODELO AS DATA_MODIFICACIO,
    MODELO.ANYOFABRICAMODELO AS ANY_FABRICACIO,
    MODELO.DESCRIPCIOMODELO AS DESCRIPCIO,
    FABRICANT.NOMCOMERCIALFABRICANT AS FABRICANT,
    MODELO.OBSERVACIONSMODELO AS OBSERVACIONS
FROM MODELO,FABRICANT,ESTAT
WHERE MODELO.IDFABRICANTMODELO=FABRICANT.IDFABRICANT
AND MODELO.ESTATMODELO=ESTAT.IDESTAT
AND MODELO.IDMODELO=MODELO.IDMODELO;

BEGIN
 c_procesLog := 'PRC_CONSULTAR_MODEL';
 c_dataHoraLog := SYSDATE;
 c_entradaLog :='CONSULTA MODELS ';
 c_sortidalog := 's_rsp';
 SELECT COUNT(*)INTO n_registres
   FROM MODELO
   WHERE MODELO.IDMODELO=MODELO.IDMODELO;
 IF n_registres=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTA MODELS: ');
   s_rsp := 'No hi han models donats en alta.  ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_model;
 END IF;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('      CONSULTA MODELS DE COPMTADORS          ');
 DBMS_OUTPUT.PUT_LINE('-----------------------------------------------');
 OPEN C_MODELO;
 FETCH C_MODELO  INTO n_idModelo,
    n_dataAltaModel,
    n_descModel,
    n_dataModificacioModel,
    n_anyModelo,
    n_descripcioModelo,
    n_fabricant,
    n_observacionsModelo;
 WHILE C_MODELO%FOUND LOOP
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(
'   Codi Model: '
||n_idModelo||
' Data alta: '
||n_dataModificacioModel||
' Esta: '
||n_descModel||
' Data ultima modificacio: '
||n_dataModificacioModel||
' Any fabricacio: '
||n_anyModelo||
' Descripcio: '
||n_descripcioModelo||
```

```
 ' Fabricant: '
 ||n_fabricant||
 ' Observacions: '
 ||n_observacionsModelo );
  s_rsp := 'OK CONSULTA MODELS Codi Model: '
 ||n_idModelo||
 ' Data alta: '
 ||n_dataModificacioModel||
 ' Esta: '
 ||n_descModel||
 ' Data ultima modificacio: '
 ||n_dataModificacioModel||
 ' Any fabricacio: '
 ||n_anyModelo||
 ' Descripcio: '
 ||n_descripcioModelo||
 ' Fabricant: '
 ||n_fabricant||
 ' Observacions: '
 ||n_observacionsModelo ;

FETCH C_MODELO  INTO n_idModelo,
      n_dataAltaModel,
      n_descModel,
      n_dataModificacioModel,
      n_anyModelo,
      n_descripcioModelo,
      n_fabricant,
      n_observacionsModelo ;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
   END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total models donats en alta: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_MODELO;
COMMIT;
    EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi
        s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
     ELSE
        -- L'error si ha estat controlat per codi
        s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;
   END PRC_CONSULTAR_MODELO;
END "GESTION_MODELO";
```

11.9 TRATAMIENTO DE GESTION CENTRALES.

DEFINICION	PROCEDIMIENTO
Alta de Central de Producción	PRC_ALTA_CENTRAL_PRODUCCIO
Alta de Central de Distribución	PRC_ALTA_CENTRAL_DISTRIBUCIO
Alta de una Central	PROCEDURE PRC_ALTA_CENTRAL
Baja de una Central	PRC_BAIXA_CENTRAL
Modificar Central de Producción	PRC_MODIFICAR_CENTRAL_PRODUC
Modificar Central de Distribución	PRC_MODIFICAR_CENTRAL_DISTRIB
Consulta de las Centrales de Producción	PRC_CONSULTAR_CENTRAL_PRODU
Consulta de las Centrales de Distribución	PRC_CONSULTAR_CENTRAL_DISTRIB

```
CREATE OR REPLACE
PACKAGE        "GESTION_CENTRAL" AS
PROCEDURE PRC_ALTA_CENTRAL_PRODUCCIO(
      p_dataAltaCentral in CENTRAL.DATAALTACENTRAL%Type,
      p_idTipoCentral in CENTRAL.IDTIPOCENTRAL%Type,
      p_dataUltmimaInspeccioCentral in CENTRAL.DATAULTIMAINSPECCIOCENTRAL%TYPE,
      p_idInspeccioCentral in CENTRAL.IDINSPECCIOCENTRAL%Type,
      p_energiaMax in CENTRAL.ENERGIAMAXIMA%Type,
      p_energiaMin in CENTRAL.ENERGIAMINIMA%Type,
      p_idUbicaCentral in CENTRAL.IDUBICACENTRAL%Type,
      p_observacionsCentral in CENTRAL.OBSERVACIONSCENTRAL%Type,
      s_rsp out NOCOPY VARCHAR2);
```

```
PROCEDURE PRC_ALTA_CENTRAL_DISTRIBUCIO(
      p_dataAltaCentral in CENTRAL.DATAALTACENTRAL%Type,
       p_dataUltmimaInspeccioCentral in CENTRAL.DATAULTIMAINSPECCIOCENTRAL%TYPE,
       p_idInspeccioCentral in CENTRAL.IDINSPECCIOCENTRAL%Type,
       p_energiaMax in CENTRAL.ENERGIAMAXIMA%Type,
       p_energiaMin in CENTRAL.ENERGIAMINIMA%Type,
       p_idUbicaCentral in CENTRAL.IDUBICACENTRAL%Type,
       p_observacionsCentral in CENTRAL.observacionsCentral%Type,
       s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_ALTA_CENTRAL(
       p_idCentral in CENTRAL.idCentral%TYPE,
       p_estatCentral in CENTRAL.ESTATCENTRAL%Type,
       p_dataModificacioCentral in CENTRAL.DATAMODIFICACENTRAL%Type,
       s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_BAIXA_CENTRAL(
       p_idCentral in CENTRAL.idCentral%TYPE,
       p_estatCentral in CENTRAL.estatCentral%Type,
       p_dataModificacioCentral in CENTRAL.DATAMODIFICACENTRAL%Type,
       s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_MODIFICAR_CENTRAL_PRODUC(
       p_idCentral in CENTRAL.IDCENTRAL%TYPE,
       p_estatCentral in CENTRAL.ESTATCENTRAL%Type,
       p_dataModificacioCentral in CENTRAL.DATAMODIFICACENTRAL%Type,
       p_idClasseCentral in CENTRAL.IDCLASSECENTRAL%Type,
       p_idTipoCentral in CENTRAL.IDTIPOCENTRAL%Type,
       p_dataUltmimaInspeccioCentral in CENTRAL.DATAULTIMAINSPECCIOCENTRAL%TYPE,
       p_idInspeccioCentral in CENTRAL.IDINSPECCIOCENTRAL%Type,
       p_energiaMax in CENTRAL.ENERGIAMAXIMA%Type,
       p_energiaMin in CENTRAL.ENERGIAMINIMA%Type,
       p_idUbicaCentral in CENTRAL.IDUBICACENTRAL%Type,
       p_observacionsCentral in CENTRAL.OBSERVACIONSCENTRAL%Type,
       s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_MODIFICAR_CENTRAL_DISTRIB(
       p_idCentral in CENTRAL.IDCENTRAL%TYPE,
       p_estatCentral in CENTRAL.estatCentral%Type,
       p_dataModificacioCentral in CENTRAL.DATAMODIFICACENTRAL%Type,
       p_idClasseCentral in CENTRAL.IDCLASSECENTRAL%Type,
       p_idTipoCentral in CENTRAL.idTipoCentral%Type,
       p_idInspeccioCentral in CENTRAL.IDINSPECCIOCENTRAL%Type,
        p_dataUltmimaInspeccioCentral in CENTRAL.DATAULTIMAINSPECCIOCENTRAL%TYPE,
       p_energiaMax in CENTRAL.ENERGIAMAXIMA%Type,
       p_energiaMin in CENTRAL.ENERGIAMINIMA%Type,
       p_idUbicaCentral in CENTRAL.idUbicaCentral%Type,
       p_observacionsCentral in CENTRAL.observacionsCentral%Type,
       s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_CONSULTAR_CENTRAL_PRODU(
```

```
        s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_CONSULTAR_CENTRAL_DISTRIB(
        s_rsp out NOCOPY VARCHAR2);

END "GESTION_CENTRAL";
```

11.9.1 PROCEDIMIENTO SPL PACKAGE DE CENTRALES.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_CENTRAL" AS
  n_registres     NUMBER;
  s_sql         VARCHAR2 (2000);
  n_NUM_ERR          NUMBER(10);
  c_procesLog          LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp      LOG_TFC.RSPLOG%TYPE;


  n_idCentral CENTRAL.IDCENTRAL%TYPE;
  n_idTipoCentral CENTRAL.IDTIPOCENTRAL%TYPE;
  n_resulatInspeccio TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%TYPE;


  n_idUbicaCentral CENTRAL.IDUBICACENTRAL%TYPE;
  n_dataAltaCentral CENTRAL.DATAALTACENTRAL%TYPE;
  n_dataModificacioCentral CENTRAL.DATAMODIFICACENTRAL%TYPE;
  n_ultimaInspeccioCentral CENTRAL.DATAULTIMAINSPECCIOCENTRAL%TYPE;
  n_maxEnergia CENTRAL.ENERGIAMAXIMA%TYPE;
  n_minEnergia CENTRAL.ENERGIAMINIMA%TYPE;
  n_observacions CENTRAL.OBSERVACIONSCENTRAL%TYPE;

  n_descResultatInspeccio TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO%type;
  n_descEstat  ESTAT.DESCRIPCIOESTAT%TYPE;
  n_descTipoCentral TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL%TYPE;
  n_descClasseCentral CLASSE_CENTRAL.DESCRIPCIOCLASSE%TYPE;
  n_localitatCentral LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
  n_descTipusFuncioCentral TIPUS_FUNCIONS.DESCRIPCIONFUNCIO%TYPE;
  n_descClasseCenral CLASSE_CENTRAL.DESCRIPCIOCLASSE%TYPE;
  n_controlEstatCentral ESTAT.IDESTAT%TYPE;
  n_controlTipoClasse NUMBER;

  n_estatCentral number;
  n_idClasseCentral number;
  p_estatCentral  CENTRAL.ESTATCENTRAL%Type;
  p_idClasseCentral CENTRAL.IDCLASSECENTRAL%TYPE;
  p_idTipoCentral  CENTRAL.IDTIPOCENTRAL%Type;

  sortida VARCHAR2(1000):=";
```

```
 e_central   EXCEPTION;
/*****************************************************************************
  NOM:       PRC_ALTA_CENTRAL_PRODUCCIO
  DESCRIPCIÃ“:
          Procediment encarregat donar alta  Central Produccio.
          La inserciÃ³ de dades esta controlada per un disparador o trigger
                  el qual tÃ© associat una seqÃºÃ¨ncia SEQ_CENTRAL per generar
                  la clau primaria en format numÃ©rica de forma autoincrementable
                  les dades es modificaran en la taula CENTRAL i el identificador de
          clau primaria del de la taula Central.
*****************************************************************************/
PROCEDURE PRC_ALTA_CENTRAL_PRODUCCIO(
       p_dataAltaCentral in CENTRAL.DATAALTACENTRAL%Type,
      -- p_estatCentral in CENTRAL.ESTATCENTRAL%Type,
      -- p_idClasseCentral in CENTRAL.IDCLASSECENTRAL%Type,
       p_idTipoCentral in CENTRAL.IDTIPOCENTRAL%Type,
       p_dataUltmimaInspeccioCentral in CENTRAL.DATAULTIMAINSPECCIOCENTRAL%TYPE,
       p_idInspeccioCentral in CENTRAL.IDINSPECCIOCENTRAL%Type,
       p_energiaMax in CENTRAL.ENERGIAMAXIMA%Type,
       p_energiaMin in CENTRAL.ENERGIAMINIMA%Type,
       p_idUbicaCentral in CENTRAL.IDUBICACENTRAL%Type,
       p_observacionsCentral in CENTRAL.OBSERVACIONSCENTRAL%Type,
       s_rsp out NOCOPY VARCHAR2)
 AS

 BEGIN
n_idClasseCentral:=1;
n_estatCentral:=1;
   --Parametres per la taula LOG
    c_procesLog := 'PRC_ALTA_CENTRAL_PRODUCCIO';
    c_dataHoraLog := SYSDATE;
    c_entradaLog := 'Data Alta Central: ' ||p_dataAltaCentral||
                --', Estat Central Produccio: ' ||n_estatCentral||
               -- ', Id Classe Central: ' ||p_idClasseCentral||
                ', Id Tipo Central: ' ||p_idTipoCentral ||
                ', Data Inspeccio Central Produccio: ' ||p_dataUltmimaInspeccioCentral||
                ', Resultat Inspeccio Produccio: ' ||p_idInspeccioCentral||
                ', Energiaa Maxima Produida: ' ||p_energiaMax||
                ', Energia Minima Produida: ' ||p_energiaMin||
                ', Direccio Central Produccio: ' ||p_idUbicaCentral||
                ', Observacions Central Produccio: ' ||p_observacionsCentral;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('      ALTA CENTRAL PRODUCCIO          ');
   DBMS_OUTPUT.PUT_LINE('----------------------------------------');
   DBMS_OUTPUT.PUT_LINE('Data Alta Central: ' ||p_dataAltaCentral||
               --', Estat Central ProduccciÃ³: ' ||n_estatCentral||
               -- ', Id Classe Central: ' ||p_idClasseCentral||
                ', Id Tipo Central: ' ||p_idTipoCentral ||
                ', Data Inspeccio Central Produccio: ' ||p_dataUltmimaInspeccioCentral||
                ', Resultat Inspeccio Produccio: ' ||p_idInspeccioCentral||
                ', Energia Maxima Produida: ' ||p_energiaMax||
```

```
                ', Energia Minima Produida: ' ||p_energiaMin||
                ', Direccio Central Produccio: ' ||p_idUbicaCentral||
                ', Observacions Central Produccio³: ' ||p_observacionsCentral);
  c_sortidalog := 's_rsp';

 n_estatCentral:=1;
 n_idClasseCentral:=1;

-- Comprovem que Classe de Central sigui 1=Produccio i les dades que es volen inserir
  IF p_dataAltaCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL PRODUCCIO        ');
       DBMS_OUTPUT.PUT_LINE('---------------------------------------');
       s_rsp := 'Falta especificar la Data alta  ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
  END IF;
  IF  p_dataUltmimaInspeccioCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL PRODUCCIO        ');
       DBMS_OUTPUT.PUT_LINE('---------------------------------------');
       s_rsp := 'Falta especificar la Data de la Inspeccio  ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
  END IF;
  IF  p_idInspeccioCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL PRODUCCIO        ');
       DBMS_OUTPUT.PUT_LINE('---------------------------------------');
      s_rsp := 'Falta especificar el Id del resultat Inspeccio ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
  END IF;
  IF  p_energiaMax IS NULL THEN
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL PRODUCCIO        ');
       DBMS_OUTPUT.PUT_LINE('---------------------------------------');
       s_rsp := 'Falta especificar EnergÃa MÃ¡xima que pot Produir ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE  e_central;
  END IF;
  IF  p_energiaMin IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL PRODUCCIO        ');
      DBMS_OUTPUT.PUT_LINE('---------------------------------------');
      s_rsp := 'Falta especificar EnergÃa MÃnima que pot Produir ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
  END IF;
  IF  p_idUbicaCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL PRODUCCIO        ');
```

```
      DBMS_OUTPUT.PUT_LINE('----------------------------------------');
      s_rsp := 'Falta especificar el codi Ubicacio que identifica la situacio
         geografica de la Central de produccio. ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
   END IF;

    SELECT  DISTINCT TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO INTO n_resulatInspeccio
  FROM TIPO_INSPECCIO
  WHERE TIPO_INSPECCIO.IDTINSPCCIO=p_idInspeccioCentral;

  SELECT DISTINCT ESTAT.DESCRIPCIOESTAT INTO n_descEstat
  FROM ESTAT
  WHERE ESTAT.IDESTAT=n_estatCentral;
     SELECT  DISTINCT TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL  INTO n_descTipoCentral
  FROM TIPO_CENTRAL
  WHERE TIPO_CENTRAL.IDTIPOCENTRAL=p_idTipoCentral;

    SELECT  DISTINCT CLASSE_CENTRAL.DESCRIPCIOCLASSE INTO n_descClasseCentral
  FROM CLASSE_CENTRAL
  WHERE CLASSE_CENTRAL.IDCLASSE=n_idClasseCentral;

     SELECT DISTINCT LOCALITAT.DESCRIPCIOLOCALITAT  INTO n_localitatCentral
  FROM  UBICACIO, LOCALITAT
  WHERE UBICACIO.IDUBICA=p_idUbicaCentral
  AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT;


  SELECT DISTINCT COUNT (*) INTO n_registres
  FROM CENTRAL
  WHERE CENTRAL.IDUBICACENTRAL=p_idUbicaCentral
  AND CENTRAL.DATAULTIMAINSPECCIOCENTRAL=p_dataUltmimaInspeccioCentral
  AND CENTRAL.DATAALTACENTRAL=p_dataAltaCentral
  AND CENTRAL.ESTATCENTRAL=n_estatCentral;

 If n_registres=0 then
  n_estatCentral:=1;
  n_idClasseCentral:=1;
 -- El donem d'alta a la Classe de Central

 INSERT INTO CENTRAL(DATAALTACENTRAL,
                 ESTATCENTRAL,
                 IDCLASSECENTRAL,
                 IDTIPOCENTRAL,
                 DATAULTIMAINSPECCIOCENTRAL,
                 IDINSPECCIOCENTRAL,
                 ENERGIAMAXIMA,
                 ENERGIAMINIMA,
                 IDUBICACENTRAL,
                 OBSERVACIONSCENTRAL)
          VALUES(p_dataAltaCentral,
              n_estatCentral,
```

```
            n_idClasseCentral,
            p_idTipoCentral,
            p_dataUltmimaInspeccioCentral,
            p_idInspeccioCentral,
            p_energiaMax,
            p_energiaMin,
            p_idUbicaCentral,
            p_observacionsCentral);

 DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE( 'Insertada la Central amb
 Data Alta Central: ' ||p_dataAltaCentral||
 ', Estat Central Produccio: '||n_descEstat||
 ', Classe Central: '||n_descClasseCenral||
 '  Tipo Central: '||n_descTipoCentral||
 ', Data Inspeccio Central Produccio: ' ||p_dataUltmimaInspeccioCentral||
 ', Resultat Inspeccio Produccio: '||n_resulatInspeccio||
 ', EnergÃa MÃ¡xima Produida: ' ||p_energiaMax||
 ', EnergÃa Minima Produida: ' ||p_energiaMin||
 ', Direccio Central Produccio: ' ||p_idUbicaCentral||' '||n_localitatCentral||
 ', Observacions Central Produccio: ' ||p_observacionsCentral);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
 --Gravem en la taula log-------------------------------------------------------
 s_rsp :='Ok: Insertada la Central amb
 Data Alta Central: ' ||p_dataAltaCentral||
 ', Estat Central Produccio: '||n_descEstat||
 ', Classe Central: '||n_descClasseCentral||
 '  Tipo Central: '||n_descTipoCentral||
   ', EnergÃa MÃ¡xima Produida: ' ||p_energiaMax||
 ', EnergÃa Minima Produida: ' ||p_energiaMin||
 ', Direccio Central Produccio: ' ||n_localitatCentral||
 ', Observacions Central Produccio: ' ||p_observacionsCentral;
 pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 ELSE
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('ALTA CENTRAL PRODUCCIO: ');
 DBMS_OUTPUT.PUT_LINE('Central Produccio Duplicada' );
 s_rsp := 'Central duplicada Data Alta Central: ' ||p_dataAltaCentral||
 ', Estat Central Produccio: '||n_descEstat||
 ', Classe Central: '||n_descClasseCenral||
 '  Tipo Central: '||n_descTipoCentral;
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_central;
 END IF;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
```

```
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
ROLLBACK;
END PRC_ALTA_CENTRAL_PRODUCCIO;
/*****************************************************************************
  NOM:       PRC_ALTA_CENTRAL_DISTRIBUCIO
  DESCRIPCIÃ“:
          Procediment encarregat donar alta  Central Distribucio.
          La inserciÃ³ de dades esta controlada per un disparador o trigger
                  el qual tÃ© associat una seqÃºÃ¨ncia SEQ_CENTRAL per generar
                  la clau primaria en format numÃ©rica de forma autoincrementable
                  les dades es modificaran en la taula CENTRAL i el identificador de
          clau primaria del de la taula Central.
*****************************************************************************/
PROCEDURE PRC_ALTA_CENTRAL_DISTRIBUCIO(
       p_dataAltaCentral in CENTRAL.DATAALTACENTRAL%Type,
       p_dataUltmimaInspeccioCentral in CENTRAL.DATAULTIMAINSPECCIOCENTRAL%TYPE,
       p_idInspeccioCentral in CENTRAL.IDINSPECCIOCENTRAL%Type,
       p_energiaMax in CENTRAL.ENERGIAMAXIMA%Type,
       p_energiaMin in CENTRAL.ENERGIAMINIMA%Type,
       p_idUbicaCentral in CENTRAL.IDUBICACENTRAL%Type,
       p_observacionsCentral in CENTRAL.observacionsCentral%Type,
       s_rsp out NOCOPY VARCHAR2)
 AS

 BEGIN

  --Parametres per la taula LOG
   c_procesLog := 'PRC_ALTA_CENTRAL_DISTRIBUCIO';
   c_dataHoraLog := SYSDATE;
   c_entradaLog := 'Data Alta Central: ' ||p_dataAltaCentral||
               --', Estat Central ProduccciÃ³: ' ||p_estatCentral||
               --', Id Classe Central: ' ||p_idClasseCentral||
               --', Id Tipo Central: ' ||p_idTipoCentral||
               ', Data Inspeccio Central Distribucio: ' ||p_dataUltmimaInspeccioCentral||
               ', Resultat Inspeccio Distribucio: ' ||p_idInspeccioCentral||
               ', Energiaa Maxima Distribucio: ' ||p_energiaMax||
               ', Energia Minima Distribucio: ' ||p_energiaMin||
               ', Direccio Central Distribucio: ' ||p_idUbicaCentral||
               ', Observacions Central Distribucio: ' ||p_observacionsCentral;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('      ALTA CENTRAL DISTRIBUCIO          ');
  DBMS_OUTPUT.PUT_LINE('----------------------------------------');
  DBMS_OUTPUT.PUT_LINE('Data Alta Central: ' ||p_dataAltaCentral||
               ', Data Inspeccio Central Distribucio: ' ||p_dataUltmimaInspeccioCentral||
               ', Resultat Inspeccio Distribucio: ' ||p_idInspeccioCentral||
               ', Energia Maxima Distribucio: ' ||p_energiaMax||
```

```
                ', Energia Minima Distribucio: ' ||p_energiaMin||
                ', Direccio Central Distribucio: ' ||p_idUbicaCentral||
                ', Observacions Central Distribucio: ' ||p_observacionsCentral);
  c_sortidalog := 's_rsp';
 -- Comprovem que Classe de Central sigui 2=Distribucio i les dades que es volen inserir
  n_idTipoCentral:=2;
  n_estatCentral:=1;
  n_idClasseCentral:=0;

   IF  p_dataUltmimaInspeccioCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL DISTRIBUCIO       ');
       DBMS_OUTPUT.PUT_LINE('----------------------------------------');
       s_rsp := 'Falta especificar la Data de la Inspeccio  ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
   END IF;
   IF  p_idInspeccioCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL DISTRIBUCIO       ');
       DBMS_OUTPUT.PUT_LINE('----------------------------------------');
      s_rsp := 'Falta especificar el Id del resultat Inspeccio ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
   END IF;
   IF  p_energiaMax IS NULL THEN
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL DISTRIBUCIO       ');
       DBMS_OUTPUT.PUT_LINE('----------------------------------------');
       s_rsp := 'Falta especificar EnergÃa MÃ¡xima que pot Distribuir ';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE  e_central;
   END IF;
   IF  p_energiaMin IS NULL THEN
      DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL DISTRIBUCIO       ');
       DBMS_OUTPUT.PUT_LINE('----------------------------------------');
      s_rsp := 'Falta especificar EnergÃa MÃnima que pot Distribuir';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
   END IF;
   IF  p_idUbicaCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL DISTRIBUCIO       ');
       DBMS_OUTPUT.PUT_LINE('----------------------------------------');
      s_rsp := 'Falta especificar el codi Ubicacio que identifica la situacio
         geografica de la Central de Distribucio. ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
   END IF;
  n_idTipoCentral:=6;
  n_estatCentral:=1;
```

```
 n_idClasseCentral:=2;
SELECT  DISTINCT TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO INTO n_resulatInspeccio
 FROM TIPO_INSPECCIO
 WHERE TIPO_INSPECCIO.IDTINSPCCIO=p_idInspeccioCentral;

 SELECT DISTINCT ESTAT.DESCRIPCIOESTAT INTO n_descEstat
 FROM ESTAT
 WHERE ESTAT.IDESTAT=n_estatCentral;

  SELECT  DISTINCT CLASSE_CENTRAL.DESCRIPCIOCLASSE INTO n_descClasseCentral
 FROM CLASSE_CENTRAL
 WHERE CLASSE_CENTRAL.IDCLASSE=n_idClasseCentral;

   SELECT DISTINCT LOCALITAT.DESCRIPCIOLOCALITAT  INTO n_localitatCentral
 FROM  UBICACIO, LOCALITAT
 WHERE UBICACIO.IDUBICA=p_idUbicaCentral
 AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT;


-- Comprovem si existeix el

 SELECT COUNT (*) INTO n_registres
 FROM CENTRAL
 WHERE IDUBICACENTRAL=p_idUbicaCentral
 AND  CENTRAL.IDCLASSECENTRAL=n_idTipoCentral;


 If n_registres=0 then
-- El donem d'alta a la Classe de Central
n_idTipoCentral:=6;
 n_estatCentral:=1;
 n_idClasseCentral:=2;
INSERT INTO CENTRAL(DATAALTACENTRAL,
                ESTATCENTRAL,
                IDCLASSECENTRAL,
                IDTIPOCENTRAL,
                DATAULTIMAINSPECCIOCENTRAL,
                IDINSPECCIOCENTRAL,
                ENERGIAMAXIMA,
                ENERGIAMINIMA,
                IDUBICACENTRAL,
                OBSERVACIONSCENTRAL)
        VALUES(p_dataAltaCentral,
            n_estatCentral,
            n_idClasseCentral,
            n_idTipoCentral,
            p_dataUltmimaInspeccioCentral,
            p_idInspeccioCentral,
            p_energiaMax,
            p_energiaMin,
            p_idUbicaCentral,
            p_observacionsCentral);
```

```
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE( 'Insertada la Central amb
 Data Alta Central: ' ||p_dataAltaCentral||
 ', Estat Central Distribucio: ' ||p_estatCentral||' '||n_descEstat||
 '  Tipo Central: '||n_descTipoCentral||
 ', Data Inspeccio Central Distribucio: ' ||p_dataUltmimaInspeccioCentral||
 ', Resultat Inspeccio Distribucio: ' ||p_idInspeccioCentral||' '||n_resulatInspeccio||
 ', EnergÃa MÃ¡xima Distribucio: ' ||p_energiaMax||
 ', EnergÃa Minima Distribucio: ' ||p_energiaMin||
 ', Direccio Central Distribucio: ' ||p_idUbicaCentral||' '||n_localitatCentral||
 ', Observacions Central Distribucio: ' ||p_observacionsCentral);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
 --Gravem en la taula log----------------------------------------------------
  DBMS_OUTPUT.put_line ('OK: Insertada' || p_idTipoCentral);
   s_rsp :='Ok: insertada Central Distribucio: Data Alta Central: ' ||p_dataAltaCentral||
 ', Estat Central Distribucio: '||n_descEstat||
 '  Tipo Central: '||n_descTipoCentral||
  ', EnergÃa MÃ¡xima Distribucio: ' ||p_energiaMax||
 ', EnergÃa Minima Distribucio: ' ||p_energiaMin;
 pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 ELSE
 DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('       ALTA CENTRAL DISTRIBUCIO         ');
      DBMS_OUTPUT.PUT_LINE('-----------------------------------------');
 DBMS_OUTPUT.PUT_LINE('Central Distribucio Duplicada ');
 s_rsp := 'Central Distribucio duplicada Data Alta Central: ' ||p_dataAltaCentral||
 ', Estat Central Distribucio: ' ||p_estatCentral||' '||n_descEstat||
 '  Tipo Central: '||n_descTipoCentral||
 ', EnergÃa MÃ¡xima Distribucio: ' ||p_energiaMax||
 ', EnergÃa Minima Distribucio: ' ||p_energiaMin||
 ', Direccio Central Distribucio: '||n_localitatCentral;
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_central;
 END IF;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
ROLLBACK;
END PRC_ALTA_CENTRAL_DISTRIBUCIO;
```

```
/*****************************************************************************
  NOM:       PRC_BAIXA_CENTRAL
  DESCRIPCIÃ“:
             Procediment encarregat donar BAIXA  Central.
             Es una modificacio de estat de la Central.
             0=BAIXA
             1=ALTA
             El procediment actualitza estatCentral a 0
*****************************************************************************/
PROCEDURE PRC_BAIXA_CENTRAL(
        p_idCentral in CENTRAL.idCentral%TYPE,
        p_estatCentral in CENTRAL.estatCentral%Type,
        p_dataModificacioCentral in CENTRAL.DATAMODIFICACENTRAL%Type,
        s_rsp out NOCOPY VARCHAR2)
AS
BEGIN
 --Parametres per la taula LOG
    c_procesLog := 'PRC_BAIXA_CENTRAL';
    c_dataHoraLog := SYSDATE;
    c_entradaLog := 'Id Central: ' ||p_idCentral||
                ', Data Modificacio Central: ' ||p_dataModificacioCentral;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          BAIXA CENTRAL               ');
    DBMS_OUTPUT.PUT_LINE('--------------------------------------------');
    DBMS_OUTPUT.PUT_LINE(' Id Central: ' ||p_idCentral||
    ', Data Modificacio Central: ' ||p_dataModificacioCentral);
    c_sortidalog := 's_rsp';
     -- Comprovem que Classe de Central sigui 1=Produccio i les dades que es volen inserir
    SELECT CENTRAL.ESTATCENTRAL INTO n_controlEstatCentral
    FROM CENTRAL
    WHERE CENTRAL.IDCENTRAL=p_idCentral;

    IF n_controlEstatCentral=2 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          BAIXA CENTRAL               ');
    DBMS_OUTPUT.PUT_LINE('--------------------------------------------');
    s_rsp := 'La Central ja esta de Baixa.  ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE  e_central;
    END IF;

    IF  p_idCentral IS NULL THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          BAIXA CENTRAL               ');
    DBMS_OUTPUT.PUT_LINE('--------------------------------------------');
    s_rsp := 'Falta en id de la Central ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE  e_central;
    END IF;

    IF  p_dataModificacioCentral IS NULL THEN
    DBMS_OUTPUT.ENABLE;
```

```
    DBMS_OUTPUT.PUT_LINE('          BAIXA CENTRAL                ');
    DBMS_OUTPUT.PUT_LINE('-------------------------------------------');
    s_rsp := 'Falta la Data de Modificacio ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE  e_central;
   END IF;

   n_estatCentral:=2;

   SELECT CENTRAL.IDCENTRAL INTO n_idCentral
   FROM CENTRAL
   WHERE IDCENTRAL= p_idCentral;

  s_sql := 'UPDATE CENTRAL SET ESTATCENTRAL=n_estatCentral';

-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF p_dataModificacioCentral IS NOT NULL THEN
-- Modifiquem el descripcio classe central
  s_sql := s_sql || 'DATAMODIFICACENTRAL='' ||p_dataModificacioCentral|| '',';
  END IF;

  -- Eliminem la coma final de la sentencia SQL
 s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃ³ del WHER
 s_sql := s_sql || ' WHERE IDCENTRAL='' || n_idCentral|| '''';
EXECUTE IMMEDIATE s_sql;

IF SQL%ROWCOUNT = 0 THEN
--L'error serÃ  que id Central ja el tenim en la taula i no admet duplicats--
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('          BAIXA CENTRAL                ');
  DBMS_OUTPUT.PUT_LINE('-------------------------------------------');
  s_rsp := 'Actualitzacio no realitzada '||p_idCentral;
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_central;
ELSE

SELECT CLASSE_CENTRAL.DESCRIPCIOCLASSE INTO n_descClasseCentral
FROM CLASSE_CENTRAL,CENTRAL
WHERE CENTRAL.IDCLASSECENTRAL=CLASSE_CENTRAL.IDCLASSE
AND CENTRAL.IDCENTRAL=p_idCentral;

SELECT ESTAT.DESCRIPCIOESTAT INTO n_descEstat
FROM ESTAT
WHERE ESTAT.IDESTAT=n_estatCentral;

  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Central ha estat donada de baixa:
  Codi Central: '||p_idCentral||' '||n_descClasseCentral||
  ' Estat: ' ||n_descEstat||
  ', Data Modificacio Central: ' ||p_dataModificacioCentral);
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
```

```
  s_rsp := 'OK Central ha estat donada de baixa:
  Codi Central: '||p_idCentral||' '||n_descClasseCentral||
  ' Estat: ' ||n_descEstat||
  ', Data Modificacio Central: ' ||p_dataModificacioCentral;
   pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;
 EXCEPTION

   WHEN DUP_VAL_ON_INDEX THEN
     DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('          BAIXA CENTRAL                ');
  DBMS_OUTPUT.PUT_LINE('-----------------------------------------');
     s_rsp := 'Error: Central duplicada. No es poden fer les modificacions';
     DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi
       ----------------------------------------
       s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi
       ----------------------------------------
       s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line (s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;
END PRC_BAIXA_CENTRAL;

/*****************************************************************************
   NOM:       PRC_ALTA_CENTRAL
   DESCRIPCIÃ“:
              Procediment encarregat donar BAIXA  Central.
              Es una modificacio de estat de la Central.
              0=BAIXA
              1=ALTA
              El procediment actualitza estatCentral a 0
*****************************************************************************/
PROCEDURE PRC_ALTA_CENTRAL(
        p_idCentral in CENTRAL.idCentral%TYPE,
        p_estatCentral in CENTRAL.estatCentral%Type,
        p_dataModificacioCentral in CENTRAL.DATAMODIFICACENTRAL%Type,
        s_rsp out NOCOPY VARCHAR2)
AS

BEGIN
  --Parametres per la taula LOG
```

```
    c_procesLog := 'PRC_ALTA_CENTRAL';
    c_dataHoraLog := SYSDATE;
    c_entradaLog := 'Id Central: ' ||p_idCentral||
                ', Data Modificacio Central: ' ||p_dataModificacioCentral;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          ALTA CENTRAL               ');
    DBMS_OUTPUT.PUT_LINE('-----------------------------------------');
    DBMS_OUTPUT.PUT_LINE(' Id Central: ' ||p_idCentral||
    ', Data Modificacio Central: ' ||p_dataModificacioCentral);
    c_sortidalog := 's_rsp';
     -- Comprovem que Classe de Central sigui 1=Produccio i les dades que es volen inserir
    SELECT CENTRAL.ESTATCENTRAL INTO n_controlEstatCentral
    FROM CENTRAL
    WHERE CENTRAL.IDCENTRAL=p_idCentral;

    IF n_controlEstatCentral=1 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          ALTA CENTRAL               ');
    DBMS_OUTPUT.PUT_LINE('-----------------------------------------');
    s_rsp := 'La Central ja esta en ALTA.  ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE  e_central;
    END IF;

    IF  p_idCentral IS NULL THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          ALTA CENTRAL               ');
    DBMS_OUTPUT.PUT_LINE('-----------------------------------------');
    s_rsp := 'Falta en id de la Central ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE  e_central;
    END IF;

    IF  p_dataModificacioCentral IS NULL THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          ALTA CENTRAL               ');
    DBMS_OUTPUT.PUT_LINE('-----------------------------------------');
    s_rsp := 'Falta la Data de Modificacio ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE  e_central;
   END IF;

   n_estatCentral:=1;

   SELECT CENTRAL.IDCENTRAL INTO n_idCentral
   FROM CENTRAL
   WHERE IDCENTRAL= p_idCentral;

  s_sql := 'UPDATE CENTRAL SET ESTATCENTRAL=n_estatCentral';

-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF p_dataModificacioCentral IS NOT NULL THEN
```

```
-- Modifiquem el descripcio classe central
  s_sql := s_sql || 'DATAMODIFICACENTRAL='' ||p_dataModificacioCentral|| '',';
  END IF;

  -- Eliminem la coma final de la sentencia SQL
 s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃ³ del WHER
 s_sql := s_sql || ' WHERE IDCENTRAL='' || n_idCentral|| '''';
EXECUTE IMMEDIATE s_sql;

IF SQL%ROWCOUNT = 0 THEN
--L'error serÃ  que id Central ja el tenim en la taula i no admet duplicats--
  DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('           ALTA CENTRAL                ');
     DBMS_OUTPUT.PUT_LINE('---------------------------------------');
  s_rsp := 'Actualitzacio no realitzada '||p_idCentral;
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_central;
ELSE

SELECT CLASSE_CENTRAL.DESCRIPCIOCLASSE INTO n_descClasseCentral
FROM CLASSE_CENTRAL,CENTRAL
WHERE CENTRAL.IDCLASSECENTRAL=CLASSE_CENTRAL.IDCLASSE
AND CENTRAL.IDCENTRAL=p_idCentral;

SELECT ESTAT.DESCRIPCIOESTAT INTO n_descEstat
FROM ESTAT
WHERE ESTAT.IDESTAT=n_estatCentral;

  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Central ha estat donada en alta:
  Codi Central: '||p_idCentral||' '||n_descClasseCentral||
  ' Estat: ' ||n_descEstat||
  ', Data Modificacio Central: ' ||p_dataModificacioCentral);
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  s_rsp := 'OK Central ha estat donada de alta:
  Codi Central: '||p_idCentral||' '||n_descClasseCentral||
  ' Estat: ' ||n_descEstat||
  ', Data Modificacio Central: ' ||p_dataModificacioCentral;
   pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END IF;
COMMIT;
 EXCEPTION

   WHEN DUP_VAL_ON_INDEX THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('           ALTA CENTRAL                ');
     DBMS_OUTPUT.PUT_LINE('---------------------------------------');
     s_rsp := 'Error: Central duplicada. No es poden fer les modificacions';
     DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;
```

```
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 ROLLBACK;
END PRC_ALTA_CENTRAL;
/*****************************************************************************
  NOM:       PRC_MODIFICAR_CENTRAL_PRODUC
  DESCRIPCIÃ“:
            Procediment encarregat modificar la Central de Produccio.
            Es crea un cursor per controlar si el valor de  la Classe de
            Central = 1 si es aixi modifiquem l'estructura de dades
            de la Central de ProduccIÃ³.

*****************************************************************************/
PROCEDURE PRC_MODIFICAR_CENTRAL_PRODUC(
        p_idCentral in CENTRAL.IDCENTRAL%TYPE,
        p_estatCentral in CENTRAL.ESTATCENTRAL%Type,
        p_dataModificacioCentral in CENTRAL.DATAMODIFICACENTRAL%Type,
        p_idClasseCentral in CENTRAL.IDCLASSECENTRAL%Type,
        p_idTipoCentral in CENTRAL.IDTIPOCENTRAL%Type,
        p_dataUltmimaInspeccioCentral in CENTRAL.DATAULTIMAINSPECCIOCENTRAL%TYPE,
        p_idInspeccioCentral in CENTRAL.IDINSPECCIOCENTRAL%Type,
        p_energiaMax in CENTRAL.ENERGIAMAXIMA%Type,
        p_energiaMin in CENTRAL.ENERGIAMINIMA%Type,
        p_idUbicaCentral in CENTRAL.IDUBICACENTRAL%Type,
        p_observacionsCentral in CENTRAL.OBSERVACIONSCENTRAL%Type,
        s_rsp out NOCOPY VARCHAR2)
AS

BEGIN
 --Parametres per la taula LOG
    c_procesLog := 'PRC_MODIFICAR_CENTRAL_PRODUC';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='Id Central Produccio: ' ||p_idCentral||
                        ', Data Modificacio Central Produccio: ' ||p_dataModificacioCentral||
            --', Id Classe Central Produccio: ' ||p_idClasseCentral ||
            ', Id Tipo Central Produccio: ' ||p_idTipoCentral||
                        ', Id Inspeccio Central Produccio: ' ||p_idInspeccioCentral||
            ', Data Ultima Inspeccio Central Produccio: '||p_dataUltmimaInspeccioCentral||
            ', Energia Maxima Central Produccio: ' ||p_energiaMax ||
            ', Energia Minma Central Produccio: ' || p_energiaMin||
```

```
            ', Id Ubicacio Central Produccio: ' ||p_idUbicaCentral ||
                        ', observacions Central Produccio: ' ||p_observacionsCentral;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('    MODIFICAR LA CENTRAL DE PRODUCCIO    ');
    DBMS_OUTPUT.PUT_LINE('---------------------------------------- ');
    DBMS_OUTPUT.PUT_LINE('Id Central Produccio: ' ||p_idCentral||
                        ', Data Modificacio Central Produccio: ' ||p_dataModificacioCentral||
            --', Id Classe Central Produccio: ' ||p_idClasseCentral ||
            ', Id Tipo Central Produccio: ' ||p_idTipoCentral||
                        ', Id Inspeccio Central Produccio: ' ||p_idInspeccioCentral||
            ', Data Ultima Inspeccio Central Produccio: '||p_dataUltmimaInspeccioCentral||
            ', Energia Maxima Central Produccio: ' ||p_energiaMax ||
            ', Energia Minma Central Produccio: ' || p_energiaMin||
            ', Id Ubicacio Central Produccio: ' ||p_idUbicaCentral ||
                        ', observacions Central Produccio: ' ||p_observacionsCentral);
  c_sortidalog := 's_rsp';

SELECT  CENTRAL.IDCLASSECENTRAL INTO n_idClasseCentral
FROM  CENTRAL
WHERE CENTRAL.IDCLASSECENTRAL=1;

SELECT CLASSE_CENTRAL.DESCRIPCIOCLASSE INTO n_descClasseCentral
FROM  CLASSE_CENTRAL
WHERE CLASSE_CENTRAL.IDCLASSE=1;

-- MIREM SI LA CENTRAL ES DE PRODUCCIO
 IF n_idClasseCentral<>1 THEN
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:    ');
  s_rsp := 'La central no es de Produccio es de '||n_descClasseCentral;
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_central;
  END IF;

-- ComprovaciÃ³ del identificador Central
  IF p_idCentral IS NULL THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:    ');
    s_rsp := 'Falta especificar el el codi de la central ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_central;
  END IF;
  IF  p_dataModificacioCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:    ');
      s_rsp := 'Falta la Data de Modificacio ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
  END IF;
 IF  p_idTipoCentral IS NULL THEN
     DBMS_OUTPUT.ENABLE;
```

```
    DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:     ');
    s_rsp := 'Falta Id Tipo Central a Modificar ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE  e_central;
  END IF;
  IF  p_idInspeccioCentral IS NULL THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:     ');
     s_rsp := 'Falta Id Inspeccio a Modificar ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE  e_central;
  END IF;
  IF  p_dataUltmimaInspeccioCentral IS NULL THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:     ');
     s_rsp := 'Falta Data Ultima Inspeccio a Modificar ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE  e_central;
  END IF;
  IF  (p_energiaMax IS NULL)  OR (p_energiaMax<0) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:     ');
     s_rsp := 'Falta Energia Maxima a Modificar o el valor es mes petit que 0 ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE  e_central;
  END IF;
  IF  (p_energiaMin IS NULL) OR (p_energiaMin<0) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:     ');
     s_rsp := 'Falta Energia Minima a Modificar o el valor es mes petit que 0  ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE  e_central;
  END IF;
  IF  p_idUbicaCentral  IS NULL THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:     ');
     s_rsp := 'Falta id Ubicacio Central a Modificar ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE  e_central;
  END IF;
  SELECT COUNT (*) INTO n_registres
  FROM CENTRAL
  WHERE idCentral = p_idCentral
  AND CENTRAL.IDCLASSECENTRAL=n_idClasseCentral;

  If n_registres=0 then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:     ');
    s_rsp := 'CENTRAL_PRODUCCIO no existeix a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_central;
  END IF;
```

```
-- recuperem el codi de la central
  SELECT CENTRAL.IDCENTRAL INTO n_idCentral
  FROM CENTRAL
  WHERE CENTRAL.IDCENTRAL= p_idCentral;

            s_sql := 'UPDATE CENTRAL SET ';

-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF p_dataModificacioCentral IS NOT NULL THEN
-- Modifiquem el data modificacio central
    s_sql := s_sql || 'DATAMODIFICACENTRAL='' ||p_dataModificacioCentral|| '',';
  END IF;
  IF p_idTipoCentral IS NOT NULL THEN
-- Modifiquem el id Tipo Central
    s_sql := s_sql || 'IDTIPOCENTRAL='' ||p_idTipoCentral|| '',';
  END IF;
  IF p_idInspeccioCentral IS NOT NULL THEN
-- Modifiquem el id Tipo Central central
     s_sql := s_sql || 'IDINSPECCIOCENTRAL='' ||p_idInspeccioCentral|| '',';
  END IF;
  IF p_dataUltmimaInspeccioCentral IS NOT NULL THEN
-- Modifiquem data Ultmima Inspeccio Central
     s_sql := s_sql || 'DATAULTIMAINSPECCIOCENTRAL='' ||p_dataUltmimaInspeccioCentral|| '',';
  END IF;
  IF p_energiaMax IS NOT NULL THEN
-- Modifiquem energia Maxima central
    s_sql := s_sql || 'ENERGIAMAXIMA='' ||p_energiaMax|| '',';
  END IF;
  IF p_energiaMin IS NOT NULL THEN
-- Modifiquem energia Minima central
    s_sql := s_sql || 'ENERGIAMINIMA='' ||p_energiaMin|| '',';
  END IF;
  IF p_idUbicaCentral IS NOT NULL THEN
-- Modifiquem Id Ubicacio Central
    s_sql := s_sql || 'IDUBICACENTRAL='' ||p_idUbicaCentral|| '',';
  END IF;
  IF p_observacionsCentral IS NOT NULL THEN
-- Modifiquem el observacions central
    s_sql := s_sql || 'OBSERVACIONSCENTRAL='' ||p_observacionsCentral|| '',';
  END IF;
-- Eliminem la coma final de la sentencia SQL
    s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃ³ del WHERE
    s_sql := s_sql || ' WHERE IDCENTRAL='' ||n_idCentral|| '';

EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0 THEN
--L'error serÃ  que id Central ja el tenim en la taula i no admet duplicats--
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:     ');
    s_rsp := 'Actualitzacio no realitzada';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
```

```
  RAISE e_central;
 ELSE
 SELECT TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL INTO n_descTipoCentral
 FROM CENTRAL, TIPO_CENTRAL
 WHERE CENTRAL.IDTIPOCENTRAL=p_idTipoCentral
 AND CENTRAL.IDTIPOCENTRAL=TIPO_CENTRAL.IDTIPOCENTRAL;

 SELECT LOCALITAT.DESCRIPCIOLOCALITAT INTO n_localitatCentral
 FROM UBICACIO, LOCALITAT
 WHERE UBICACIO.IDUBICA=p_idUbicaCentral
 AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT;

  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Actualitzacio satisfactoria Id Central Produccio: ' ||p_idCentral||
                       ', Data Modificacio Central Produccio: ' ||p_dataModificacioCentral||
           ', Classe Central: ' ||n_descClasseCentral||
           ', Id Tipo Central Produccio: ' ||p_idTipoCentral||' '||n_descTipoCentral||
                       ', Id Inspeccio Central Produccio: ' ||p_idInspeccioCentral||
           ', Data Ultima Inspeccio Central Produccio: '||p_dataUltmimaInspeccioCentral||
           ', Energia Maxima Central Produccio: ' ||p_energiaMax ||
           ', Energia Minma Central Produccio: ' || p_energiaMin||
           ', Ubicacio Central Produccio: ' ||n_localitatCentral||
                       ', observacions Central Produccio: ' ||p_observacionsCentral);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  s_rsp := 'OK Actualitzacio satisfactoria Actualitzacio satisfactoria
           Id Central Produccio: ' ||p_idCentral||
                       ', Data Modificacio Central Produccio: ' ||p_dataModificacioCentral||
           ', Classe Central: ' ||n_descClasseCentral||
           ', Id Tipo Central Produccio: ' ||p_idTipoCentral||' '||n_descTipoCentral||
                       ', Id Inspeccio Central Produccio: ' ||p_idInspeccioCentral||
           ', Data Ultima Inspeccio Central Produccio: '||p_dataUltmimaInspeccioCentral||
           ', Energia Maxima Central Produccio: ' ||p_energiaMax ||
           ', Energia Minma Central Produccio: ' || p_energiaMin||
           ', Ubicacio Central Produccio: ' ||n_localitatCentral||
                       ', observacions Central Produccio: ' ||p_observacionsCentral;
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;

COMMIT;
 EXCEPTION

  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
```

```
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_MODIFICAR_CENTRAL_PRODUC;
/*******************************************************************************
  NOM:       PRC_MODIFICAR_CENTRAL_DISTRIB
  DESCRIPCIÃ“:
          Procediment encarregat modificar la Central de DistribuciÃ³.
          Es crea un cursor per controlar si el valor de  la Classe de
          Central = 1 si es aixi modifiquem l'estructura de dades
          de la Central de DistribuciÃ³.

*******************************************************************************/
PROCEDURE PRC_MODIFICAR_CENTRAL_DISTRIB(
p_idCentral in CENTRAL.IDCENTRAL%TYPE,
      p_estatCentral in CENTRAL.estatCentral%Type,
      p_dataModificacioCentral in CENTRAL.DATAMODIFICACENTRAL%Type,
      p_idClasseCentral in CENTRAL.IDCLASSECENTRAL%Type,
      p_idTipoCentral in CENTRAL.idTipoCentral%Type,
      p_idInspeccioCentral in CENTRAL.IDINSPECCIOCENTRAL%Type,
      p_dataUltmimaInspeccioCentral in CENTRAL.DATAULTIMAINSPECCIOCENTRAL%TYPE,
      p_energiaMax in CENTRAL.ENERGIAMAXIMA%Type,
      p_energiaMin in CENTRAL.ENERGIAMINIMA%Type,
      p_idUbicaCentral in CENTRAL.idUbicaCentral%Type,
      p_observacionsCentral in CENTRAL.observacionsCentral%Type,
      s_rsp out NOCOPY VARCHAR2)

AS

BEGIN
 --Parametres per la taula LOG
   c_procesLog := 'PRC_MODIFICAR_CENTRAL_PRODUC';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Id Central Produccio: ' ||p_idCentral||
                  ', Data Modificacio Central Produccio: ' ||p_dataModificacioCentral||
          --', Id Classe Central Produccio: ' ||p_idClasseCentral ||
          --', Id Tipo Central Produccio: ' ||p_idTipoCentral||
                  ', Id Inspeccio Central Produccio: ' ||p_idInspeccioCentral||
          ', Data Ultima Inspeccio Central Produccio: '||p_dataUltmimaInspeccioCentral||
          ', Energia Maxima Central Produccio: ' ||p_energiaMax ||
          ', Energia Minma Central Produccio: ' || p_energiaMin||
          ', Id Ubicacio Central Produccio: ' ||p_idUbicaCentral ||
                  ', observacions Central Produccio: ' ||p_observacionsCentral;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('    MODIFICAR LA CENTRAL DE DISTRIBUCIO    ');
    DBMS_OUTPUT.PUT_LINE('---------------------------------------- ');
    DBMS_OUTPUT.PUT_LINE('Id Central Produccio: ' ||p_idCentral||
                  ', Data Modificacio Central Produccio: ' ||p_dataModificacioCentral||
          --', Id Classe Central Produccio: ' ||p_idClasseCentral ||
          --', Id Tipo Central Produccio: ' ||p_idTipoCentral||
                  ', Id Inspeccio Central Produccio: ' ||p_idInspeccioCentral||
```

```
            ', Data Ultima Inspeccio Central Produccio: '||p_dataUltmimaInspeccioCentral||
            ', Energia Maxima Central Produccio: ' ||p_energiaMax ||
            ', Energia Minma Central Produccio: ' || p_energiaMin||
            ', Id Ubicacio Central Produccio: ' ||p_idUbicaCentral ||
                        ', observacions Central Produccio: ' ||p_observacionsCentral);
   c_sortidalog := 's_rsp';

SELECT  CENTRAL.IDCLASSECENTRAL INTO n_idClasseCentral
FROM  CENTRAL
WHERE CENTRAL.IDCLASSECENTRAL=2;

SELECT CLASSE_CENTRAL.DESCRIPCIOCLASSE INTO n_descClasseCentral
FROM  CLASSE_CENTRAL
WHERE CLASSE_CENTRAL.IDCLASSE=2;

-- MIREM SI LA CENTRAL ES DE PRODUCCIO
 IF n_idClasseCentral<>1 THEN
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE DISTRIBUCIO:    ');
  s_rsp := 'La central no es de DISTRIBUCIO es de '||n_descClasseCentral;
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_central;
  END IF;

-- ComprovaciÃ³ del identificador Central
  IF p_idCentral IS NULL THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE DISTRIBUCIO:    ');
    s_rsp := 'Falta especificar el el codi de la central ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_central;
  END IF;
  IF  p_dataModificacioCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE DISTRIBUCIO:    ');
      s_rsp := 'Falta la Data de Modificacio ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
  END IF;

  IF  p_idInspeccioCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE DISTRIBUCIO:    ');
      s_rsp := 'Falta Id Inspeccio a Modificar ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE  e_central;
  END IF;
  IF  p_dataUltmimaInspeccioCentral IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE DISTRIBUCIO:    ');
      s_rsp := 'Falta Data Ultima Inspeccio a Modificar ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
```

```
    RAISE  e_central;
  END IF;
  IF  (p_energiaMax IS NULL)  OR (p_energiaMax<0) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE DISTRIBUCIO:     ');
     s_rsp := 'Falta Energia Maxima a Modificar o el valor es mes petit que 0 ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE  e_central;
  END IF;
  IF  (p_energiaMin IS NULL) OR (p_energiaMin<0) THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE DISTRIBUCIO:     ');
     s_rsp := 'Falta Energia Minima a Modificar o el valor es mes petit que 0  ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE  e_central;
  END IF;
  IF  p_idUbicaCentral  IS NULL THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE DISTRIBUCIO:     ');
     s_rsp := 'Falta id Ubicacio Central a Modificar ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE  e_central;
  END IF;
  SELECT COUNT (*) INTO n_registres
  FROM CENTRAL
  WHERE idCentral = p_idCentral
  AND CENTRAL.IDCLASSECENTRAL=n_idClasseCentral;

  If n_registres=0 then
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE DISTRIBUCIO:     ');
    s_rsp := 'CENTRAL DISTRIBUCIO no existeix a la BBDD';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_central;
  END IF;
-- recuperem el codi de la central
  SELECT CENTRAL.IDCENTRAL INTO n_idCentral
  FROM CENTRAL
  WHERE CENTRAL.IDCENTRAL= p_idCentral;

              s_sql := 'UPDATE CENTRAL SET ';

-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
  IF p_dataModificacioCentral IS NOT NULL THEN
-- Modifiquem el data modificacio central
    s_sql := s_sql || 'DATAMODIFICACENTRAL="' ||p_dataModificacioCentral|| '",';
  END IF;
 IF p_idInspeccioCentral IS NOT NULL THEN
-- Modifiquem el id Tipo Central central
     s_sql := s_sql || 'IDINSPECCIOCENTRAL="' ||p_idInspeccioCentral|| '",';
  END IF;
  IF p_dataUltmimaInspeccioCentral IS NOT NULL THEN
```

```
-- Modifiquem data Ultmima Inspeccio Central
    s_sql := s_sql || 'DATAULTIMAINSPECCIOCENTRAL=''' ||p_dataUltmimaInspeccioCentral|| ''',';
 END IF;
 IF p_energiaMax IS NOT NULL THEN
-- Modifiquem energia Maxima central
   s_sql := s_sql || 'ENERGIAMAXIMA=''' ||p_energiaMax|| ''',';
 END IF;
 IF p_energiaMin IS NOT NULL THEN
-- Modifiquem energia Minima central
   s_sql := s_sql || 'ENERGIAMINIMA=''' ||p_energiaMin|| ''',';
 END IF;
 IF p_idUbicaCentral IS NOT NULL THEN
-- Modifiquem Id Ubicacio Central
   s_sql := s_sql || 'IDUBICACENTRAL=''' ||p_idUbicaCentral|| ''',';
 END IF;
 IF p_observacionsCentral IS NOT NULL THEN
-- Modifiquem el observacions central
   s_sql := s_sql || 'OBSERVACIONSCENTRAL=''' ||p_observacionsCentral|| ''',';
 END IF;
-- Eliminem la coma final de la sentencia SQL
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃ³ del WHERE
   s_sql := s_sql || ' WHERE IDCENTRAL=''' ||n_idCentral|| '''';

EXECUTE IMMEDIATE s_sql;
 IF SQL%ROWCOUNT = 0 THEN
--L'error serÃ  que id Central ja el tenim en la taula i no admet duplicats--
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CENTRAL DE PRODUCCIO:     ');
   s_rsp := 'Actualitzacio no realitzada';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_central;
 ELSE

 SELECT LOCALITAT.DESCRIPCIOLOCALITAT INTO n_localitatCentral
 FROM CENTRAL, UBICACIO, LOCALITAT
 WHERE UBICACIO.IDUBICA=p_idUbicaCentral
 AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT;

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('Actualitzacio satisfactoria Id Central Produccio: ' ||p_idCentral||
                        ', Data Modificacio Central Produccio: ' ||p_dataModificacioCentral||
           ', Classe Central: ' ||n_descClasseCentral||
                        ', Id Inspeccio Central Produccio: ' ||p_idInspeccioCentral||
           ', Data Ultima Inspeccio Central Produccio: '||p_dataUltmimaInspeccioCentral||
           ', Energia Maxima Central Produccio: ' ||p_energiaMax ||
           ', Energia Minma Central Produccio: ' || p_energiaMin||
           ', Ubicacio Central Produccio: ' ||n_localitatCentral||
                        ', observacions Central Produccio: ' ||p_observacionsCentral);
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  s_rsp := 'OK Actualitzacio satisfactoria Actualitzacio satisfactoria
           Id Central Produccio: ' ||p_idCentral||
```

```
                    ', Data Modificacio Central Produccio: ' ||p_dataModificacioCentral||
            ', Classe Central: ' ||n_descClasseCentral||
                    ', Id Inspeccio Central Produccio: ' ||p_idInspeccioCentral||
            ', Data Ultima Inspeccio Central Produccio: '||p_dataUltmimaInspeccioCentral||
            ', Energia Maxima Central Produccio: ' ||p_energiaMax ||
            ', Energia Minma Central Produccio: ' || p_energiaMin||
            ', Ubicacio Central Produccio: ' ||n_localitatCentral||
                    ', observacions Central Produccio: ' ||p_observacionsCentral;
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
   END IF;
COMMIT;
 EXCEPTION

  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_MODIFICAR_CENTRAL_DISTRIB;

PROCEDURE PRC_CONSULTAR_CENTRAL_PRODU(
       s_rsp out NOCOPY VARCHAR2)
AS
CURSOR C_CENTRAL_PRODU IS
SELECT  DISTINCT CENTRAL.IDCENTRAL,
CENTRAL.DATAALTACENTRAL,
ESTAT.DESCRIPCIOESTAT,
CENTRAL.DATAMODIFICACENTRAL,
CLASSE_CENTRAL.DESCRIPCIOCLASSE,
TIPUS_FUNCIONS.DESCRIPCIONFUNCIO,
TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL,
TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO,
CENTRAL.DATAULTIMAINSPECCIOCENTRAL,
CENTRAL.ENERGIAMAXIMA,
CENTRAL.ENERGIAMINIMA,
LOCALITAT.DESCRIPCIOLOCALITAT,
CENTRAL.OBSERVACIONSCENTRAL
FROM
CENTRAL,ESTAT,CLASSE_CENTRAL,TIPO_CENTRAL,TIPO_INSPECCIO,UBICACIO,LOCALITAT,TIPUS_FUNCIONS
WHERE ESTAT.IDESTAT=CENTRAL.ESTATCENTRAL
AND CENTRAL.IDCLASSECENTRAL=1
AND CENTRAL.IDTIPOCENTRAL=TIPO_CENTRAL.IDTIPOCENTRAL
AND CENTRAL.IDINSPECCIOCENTRAL=TIPO_INSPECCIO.IDTINSPCCIO
AND CENTRAL.IDUBICACENTRAL=UBICACIO.IDUBICA
```

```
AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT
AND CENTRAL.IDUBICACENTRAL=UBICACIO.IDUBICA
AND TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL=TIPUS_FUNCIONS.IDFUNCIO
AND CENTRAL.IDCLASSECENTRAL=CLASSE_CENTRAL.IDCLASSE
ORDER BY CENTRAL.IDCENTRAL;

BEGIN
--Parametres per la taula LOG-----------------------------------------------------------
    c_procesLog :=  'PRC_CONSULTAR_CENTRAL PRODUCCIO';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='CONSULTAR CENTRALS DE PRODUCCIO  ';
    c_sortidalog := 's_rsp';
 SELECT COUNT (*)  INTO n_registres
 FROM CENTRAL
 WHERE CENTRAL.IDCENTRAL=CENTRAL.IDCENTRAL
 AND CENTRAL.IDCLASSECENTRAL=1;

 IF n_registres=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('CONSULTAR CENTRALS DE PRODUCCIO: ');
   s_rsp := 'No hi han CENTRALS DE PRODUCCIO donades d´alta ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_central;
END IF;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('          CONSULTAR CENTRALS DE PRODUCCIO              ');
   DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------------');
 OPEN C_CENTRAL_PRODU;
 FETCH C_CENTRAL_PRODU INTO n_idCentral,
    n_dataAltaCentral,
    n_descEstat,
    n_dataModificacioCentral,
    n_descClasseCentral,
    n_descTipusFuncioCentral,
    n_descTipoCentral,
    n_descResultatInspeccio,
    n_ultimaInspeccioCentral,
    n_maxEnergia,
    n_minEnergia,
    n_localitatCentral,
    n_observacions;

WHILE C_CENTRAL_PRODU%FOUND LOOP
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(
'  Codi Central: '
||n_idCentral||
'  Classe de Central: '
||n_descClasseCentral||
  '  Data Alta: '
||n_descClasseCentral||
'  Estat: '
```

```
||n_descEstat||
'  Tipus de Central: '
||n_descTipoCentral||
 '  Caractristiques: '
||n_descTipusFuncioCentral||
 '  Energia Maxima:  '
||n_maxEnergia||
 '  Energia Minima: '
||n_minEnergia||
 '  Poblacio: '
||n_localitatCentral);
-- '  Orservacions: '
--||n_observacions);
 DBMS_OUTPUT.PUT_LINE(' ');
 s_rsp := 'OK  CONSULTA CENTRALS PRODUCCIO:
   Codi Central: '
||n_idCentral;
FETCH C_CENTRAL_PRODU  INTO n_idCentral,
    n_dataAltaCentral,
    n_descEstat,
    n_dataModificacioCentral,
    n_descClasseCentral,
    n_descTipusFuncioCentral,
    n_descTipoCentral,
    n_descResultatInspeccio,
    n_ultimaInspeccioCentral,
    n_maxEnergia,
    n_minEnergia,
    n_localitatCentral,
    n_observacions;
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total Centrals de Produccio: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_CENTRAL_PRODU;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
```

```
END PRC_CONSULTAR_CENTRAL_PRODU;

PROCEDURE PRC_CONSULTAR_CENTRAL_DISTRIB(
        s_rsp out NOCOPY VARCHAR2)
AS
CURSOR C_CENTRAL_DISTRIBU IS
SELECT DISTINCT CENTRAL.IDCENTRAL AS CODI,
    CLASSE_CENTRAL.DESCRIPCIOCLASSE AS CLASSE_CENTRAL,
    TIPO_CENTRAL.DESCRIPCIOTIPOCENTRAL AS TIPUS_CENTRAL,
    TIPUS_FUNCIONS.DESCRIPCIONFUNCIO AS CARACTERISTIQUES,
    CENTRAL.DATAALTACENTRAL AS DATA_ALTA,
    ESTAT.DESCRIPCIOESTAT AS ESTAT,
    CENTRAL.DATAMODIFICACENTRAL AS DATA_MODIFICACIO,
    TIPO_INSPECCIO.DESCRIPCIOTINSPECCIO AS RESULTAT_INSPECCIO,
    CENTRAL.DATAULTIMAINSPECCIOCENTRAL AS DATA_ULTIMA_INSPECCIO,
    CENTRAL.ENERGIAMAXIMA AS ENERGIA_MAXIMA,
    CENTRAL.ENERGIAMINIMA AS ENERGIA_MINIMA,
    LOCALITAT.DESCRIPCIOLOCALITAT AS LOCALITAT,
    CENTRAL.OBSERVACIONSCENTRAL
FROM CENTRAL,TIPO_CENTRAL,TIPUS_FUNCIONS,ESTAT,TIPO_INSPECCIO,
LOCALITAT,CLASSE_CENTRAL,UBICACIO
WHERE CENTRAL.IDCENTRAL=CENTRAL.IDCENTRAL
AND CENTRAL.IDCLASSECENTRAL=2
AND CENTRAL.IDTIPOCENTRAL=TIPO_CENTRAL.IDTIPOCENTRAL
AND TIPO_CENTRAL.IDFUNCIOTIPOCENTRAL=TIPUS_FUNCIONS.IDFUNCIO
AND CENTRAL.ESTATCENTRAL=ESTAT.IDESTAT
AND CENTRAL.IDINSPECCIOCENTRAL=TIPO_INSPECCIO.IDTINSPCCIO
AND CENTRAL.IDUBICACENTRAL=UBICACIO.IDUBICA
AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT
AND CENTRAL.IDCLASSECENTRAL=CLASSE_CENTRAL.IDCLASSE;


BEGIN
--Parametres per la taula LOG-----------------------------------------------------------
    c_procesLog :=  'PRC_CONSULTAR_CENTRAL DISTRIBUCIO';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='CONSULTAR CENTRALS DE  DISTRIBUCIO  ';
    c_sortidalog := 's_rsp';
  SELECT COUNT (*)  INTO n_registres
  FROM CENTRAL
  WHERE CENTRAL.IDCENTRAL=CENTRAL.IDCENTRAL
  AND CENTRAL.IDCLASSECENTRAL=2;

  IF n_registres=0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTAR CENTRALS DE  DISTRIBUCIO: ');
    s_rsp := 'No hi han CENTRALS DE  DISTRIBUCIO donades d´alta ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_central;
 END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('          CONSULTAR CENTRALS DE  DISTRIBUCIO            ');
```

```
   DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------------');
  OPEN  C_CENTRAL_DISTRIBU;
  FETCH  C_CENTRAL_DISTRIBU
     INTO n_idCentral,
     n_descClasseCentral,
     n_descTipoCentral,
     n_descTipusFuncioCentral,
     n_dataAltaCentral,
     n_descEstat,
     n_dataModificacioCentral,
     n_descResultatInspeccio,
     n_ultimaInspeccioCentral,
     n_maxEnergia,
     n_minEnergia,
     n_localitatCentral,
     n_observacions;
WHILE C_CENTRAL_DISTRIBU%FOUND LOOP
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(
 '   Codi Central: '
||n_idCentral||
'   Classe de Central: '
||n_descClasseCentral||
 '   Data Alta: '
||n_dataAltaCentral||
'   Estat: '
||n_descEstat||
'   Data Modificacio: '
||n_dataModificacioCentral||
'   Resultat Inspeccio: '
||n_descResultatInspeccio||
'   Data Ultima Inspeccio: '
||n_ultimaInspeccioCentral||
 '   Energia Maxima:  '
||n_maxEnergia||
 '   Energia Minima: '
||n_minEnergia||
 '   Poblacio: '
||n_localitatCentral);
-- '   Observacions: '
--||n_observacions);
 s_rsp := 'OK  CENTRALS DE PRODUCCIO:
    Codi Central: '
||n_idCentral||
'   Classe de Central: '
||n_descClasseCentral||
 'Energia Maxima:  '
||n_maxEnergia||
 '   Energia Minima: '
||n_minEnergia||
 '   Poblacio: '
||n_localitatCentral;
```

```
 FETCH  C_CENTRAL_DISTRIBU
     INTO n_idCentral,
     n_descClasseCentral,
     n_descTipoCentral,
     n_descTipusFuncioCentral,
     n_dataAltaCentral,
     n_descEstat,
     n_dataModificacioCentral,
     n_descResultatInspeccio,
     n_ultimaInspeccioCentral,
     n_maxEnergia,
     n_minEnergia,
     n_localitatCentral,
     n_observacions;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total Centrals de Distribucio: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE  C_CENTRAL_DISTRIBU;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi
       s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi
       s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;
END PRC_CONSULTAR_CENTRAL_DISTRIB;

END "GESTION_CENTRAL";
```

11.10 TRATAMIENTO DE GESTION CENTRALES.

DEFINICION	PROCEDIMIENTO
Alta de Contador	PRC_ALTA_COMPTADOR
Cambiar el estado del Contador a Alta	PRC_PASSAR_COMPTADOR_ALTA
Cambiar el estado del Contador a Baja	PRC_PASSAR_COMPTADOR_BAIXA
Modificar un Contador	PRC_MODIFICA_COMPTADOR
Consultar Contadores	PRC_CONSULTAR_COMPTADOR

```
CREATE OR REPLACE
PACKAGE        "GESTION_COMPTADOR" AS
procedure PRC_ALTA_COMPTADOR(
  p_idModelComptador in COMPTADOR.IDMODELOCOMPTADOR%TYPE,
  p_idUbicacioComptador in COMPTADOR.IDUBICACOMPTADOR%TYPE,
  p_dataAltaComptador in COMPTADOR.DATAALTACOMPTADOR%TYPE,
  --p_estatComptador in COMPTADOR.IDESTADOCOMPATOR%TYPE,
  p_numeroSerieComptador in COMPTADOR.NUMSERIECOMPTADOR%TYPE,
  p_observacionsComptador in COMPTADOR.OBSERVACIONSCOMPTADOR%TYPE,
  s_rsp out NOCOPY VARCHAR2);

procedure PRC_PASSAR_COMPTADOR_ALTA(
  p_idComptador in COMPTADOR.IDCOMPTADOR%TYPE,
  p_dataModificacioComptador in COMPTADOR.DATAMODIFICACIOCOMPATOR%TYPE,
  p_estatComptador in COMPTADOR.IDESTADOCOMPATOR%TYPE,
   s_rsp out NOCOPY VARCHAR2);

procedure PRC_PASSAR_COMPTADOR_BAIXA(
  p_idComptador in COMPTADOR.IDCOMPTADOR%TYPE,
  p_dataModificacioComptador in COMPTADOR.DATAMODIFICACIOCOMPATOR%TYPE,
  p_estatComptador in COMPTADOR.IDESTADOCOMPATOR%TYPE,
  s_rsp out NOCOPY VARCHAR2);

procedure PRC_MODIFICA_COMPTADOR(
  p_idComptador in COMPTADOR.IDCOMPTADOR%TYPE,
```

```
  p_idModelComptador in COMPTADOR.IDMODELOCOMPTADOR%TYPE,
  p_idUbicacioComptador in COMPTADOR.IDUBICACOMPTADOR%TYPE,
  p_dataModificacioComptador in COMPTADOR.DATAMODIFICACIOCOMPATOR%TYPE,
  p_estatComptador in COMPTADOR.IDESTADOCOMPATOR%TYPE,
  p_numeroSerieComptador in COMPTADOR.NUMSERIECOMPTADOR%TYPE,
  p_observacionsComptador in COMPTADOR.OBSERVACIONSCOMPTADOR%TYPE,
  s_rsp out NOCOPY VARCHAR2);

procedure PRC_CONSULTAR_COMPTADOR(
  s_rsp out NOCOPY VARCHAR2);
END "GESTION_COMPTADOR";
```

11.10.1 PROCEDIMIENTO SPL PACKAGE DE CONTADORES.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_COMPTADOR" AS
 c_procesLog          LOG_TFC.procesLog%TYPE;
 c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog     LOG_TFC.entradaLog%TYPE;
 c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
 s_rsp              LOG_TFC.rspLog%TYPE;
 n_registres  NUMBER;
 s_sql VARCHAR2 (2000);

 n_idComptador  COMPTADOR.IDCOMPTADOR%TYPE;
 n_idModelComptador COMPTADOR.IDMODELOCOMPTADOR%TYPE;
 n_idUbicacioComptador COMPTADOR.IDUBICACOMPTADOR%TYPE;
 n_dataAltaComptador COMPTADOR.DATAALTACOMPTADOR%TYPE;
 n_estatComptador  COMPTADOR.IDESTADOCOMPATOR%TYPE;
 n_numeroSerieComptador  COMPTADOR.NUMSERIECOMPTADOR%TYPE;
 n_observacionsComptador  COMPTADOR.OBSERVACIONSCOMPTADOR%TYPE;

 n_estat  ESTAT.DESCRIPCIOESTAT%TYPE;
 n_localitat  LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
 n_provincia PROVINCIA.DESCRIPCIOPROVINCIA%TYPE;
 n_pais PAIS.DESCRIPCIOPAIS%TYPE;
 n_modelo MODELO.DESCRIPCIOMODELO%TYPE;
 n_dataModificacioComptador COMPTADOR.DATAMODIFICACIOCOMPATOR%TYPE;
 n_via VIA.DESCRIPCIOVIA%TYPE;
 n_direccio UBICACIO.NUMEROUBICACIO%TYPE;
 n_numero UBICACIO.NUMEROUBICACIO%TYPE;
 n_pis UBICACIO.PISUBICACIO%TYPE;
 n_porta UBICACIO.PORTAUBICACIO%TYPE;
 n_codiPostal UBICACIO.CODIPOSTAL%TYPE;

 n_estatComptadorControl NUMBER;

 n_NUM_ERR NUMBER(10);
 sortida varchar2(2000):='';
 e_comptador   EXCEPTION;
```

```
PROCEDURE PRC_ALTA_COMPTADOR(
  p_idModelComptador in COMPTADOR.IDMODELOCOMPTADOR%TYPE,
  p_idUbicacioComptador in COMPTADOR.IDUBICACOMPTADOR%TYPE,
  p_dataAltaComptador in COMPTADOR.DATAALTACOMPTADOR%TYPE,
  p_numeroSerieComptador in COMPTADOR.NUMSERIECOMPTADOR%TYPE,
  p_observacionsComptador in COMPTADOR.OBSERVACIONSCOMPTADOR%TYPE,
  s_rsp out NOCOPY VARCHAR2)
  AS
  BEGIN
   c_procesLog := 'PRC_ALTA_COMPTADOR';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Id Model: ' || p_idModelComptador||
   ' Id Ubicacio Comptador: ' || p_idUbicacioComptador||
   ' , Data Alta Comptador: ' || p_dataAltaComptador||
   --' , Estat Comptador: ' || p_estatComptador ||
   ' , Numero Serie Comptador: ' || p_numeroSerieComptador||
   ' , Observacions Comptador: ' || p_observacionsComptador;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('      ALTA COMPTADOR     ');
   DBMS_OUTPUT.PUT_LINE('------------------------ ');
   DBMS_OUTPUT.PUT_LINE('Id Model: ' || p_idModelComptador||
   ' Id Ubicacio Comptador: ' || p_idUbicacioComptador||
   ' , Data Alta Comptador: ' || p_dataAltaComptador||
  -- ' , Estat Comptador: ' || p_estatComptador ||
   ' , Numero Serie Comptador: ' || p_numeroSerieComptador||
   ' , Observacions Comptador: ' || p_observacionsComptador);
   c_sortidalog := 's_rsp';
   --Comprovar dades
     If p_idModelComptador IS NULL then
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('ALTA_COMPTADOR: ');
         s_rsp := 'Falta especificar Id Model Comptador ';
          DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_comptador;
      END IF;
     If p_idUbicacioComptador IS NULL then
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('ALTA_COMPTADOR: ');
          s_rsp := 'Falta especificar Id Ubicacio Comptador ';
          DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_comptador;
      END IF;
    If p_dataAltaComptador IS NULL then
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('ALTA_COMPTADOR: ');
         s_rsp := 'Falta especificar Data Alta Comptador ';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_comptador;
      END IF;
   If p_numeroSerieComptador IS NULL then
          DBMS_OUTPUT.ENABLE;
```

```
        DBMS_OUTPUT.PUT_LINE('ALTA_COMPTADOR: ');
        s_rsp := 'Falta especificar Numero Serie Comptador ';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_comptador;
     END IF;
  -- Comprovem si existeix el  donar el comptador
      SELECT COUNT (*) INTO n_registres
        FROM COMPTADOR
        WHERE COMPTADOR.NUMSERIECOMPTADOR = p_numeroSerieComptador ;
   If n_registres=0 then

  n_estatComptadorControl:=1;

   SELECT LOCALITAT.DESCRIPCIOLOCALITAT,
   PROVINCIA.DESCRIPCIOPROVINCIA,
   PAIS.DESCRIPCIOPAIS INTO n_localitat,n_provincia,n_pais
   FROM LOCALITAT, UBICACIO, PROVINCIA,PAIS
   WHERE LOCALITAT.IDLOCALITAT=UBICACIO.IDLOCALITATUBICACIO
   AND UBICACIO.IDUBICA=p_idUbicacioComptador
   AND LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
   AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS;

   SELECT ESTAT.DESCRIPCIOESTAT INTO n_estat
   FROM ESTAT
   WHERE ESTAT.IDESTAT=n_estatComptadorControl;

      -- El donem d'alta COMPTADOR
        INSERT INTO COMPTADOR(IDMODELOCOMPTADOR,
        IDUBICACOMPTADOR,
        DATAALTACOMPTADOR,
        IDESTADOCOMPATOR,
        NUMSERIECOMPTADOR,
        OBSERVACIONSCOMPTADOR)
        VALUES(p_idModelComptador,
            p_idUbicacioComptador,
            p_dataAltaComptador,
            n_estatComptadorControl,
            p_numeroSerieComptador,
            p_observacionsComptador);
  --Gravem en la taula log----------------------------------------------------
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('ALTA_COMPTADOR: ');
        DBMS_OUTPUT.PUT_LINE(' Comptador inserit: Id Model: ' || p_idModelComptador||
            '  Ciutat: ' ||n_localitat||
            '  Provincia: '||n_provincia||
            '  Pais: '||n_pais||
            '  Data Alta Comptador: ' || p_dataAltaComptador||
            '  Estat Comptador: ' || n_estat ||
            '  Numero Serie Comptador: ' || p_numeroSerieComptador||
            '  Observacions Comptador: ' || p_observacionsComptador);
           DBMS_OUTPUT.PUT_LINE('********** FI DE LES OPERACIONS ********** ');
        s_rsp :='Ok Comptador inserit: Id Model: ' || p_idModelComptador||
```

```
            --' Ciutat: ' ||n_localitat||
            --' Provincia: '||n_provincia||
            -- ' Pais: '||n_pais||
            ' , Data Alta Comptador: ' || p_dataAltaComptador||
            ' , Estat Comptador: ' || n_estat ||
            ' , Numero Serie Comptador: ' || p_numeroSerieComptador||
            ' , Observacions Comptador: ' || p_observacionsComptador;
        pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
        DBMS_OUTPUT.PUT_LINE(s_rsp);
   ELSE
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('ALTA COMPTADOR: ');
   s_rsp := 'Comptador duplicada';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_comptador;
    END IF;
  COMMIT;
 EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;
END PRC_ALTA_COMPTADOR;

PROCEDURE PRC_MODIFICA_COMPTADOR(
   p_idComptador in COMPTADOR.IDCOMPTADOR%TYPE,
   p_idModelComptador in COMPTADOR.IDMODELOCOMPTADOR%TYPE,
   p_idUbicacioComptador in COMPTADOR.IDUBICACOMPTADOR%TYPE,
   p_dataModificacioComptador in COMPTADOR.DATAMODIFICACIOCOMPATOR%TYPE,
   p_estatComptador in COMPTADOR.IDESTADOCOMPATOR%TYPE,
   p_numeroSerieComptador in COMPTADOR.NUMSERIECOMPTADOR%TYPE,
   p_observacionsComptador in COMPTADOR.OBSERVACIONSCOMPTADOR%TYPE,
   s_rsp out NOCOPY VARCHAR2)
   AS
   BEGIN
    --Parametres per la taula LOG-----------------------------------------------------------
     c_procesLog := 'PRC_MODIFICA_COMPTADOR';
     c_dataHoraLog := SYSDATE;
     c_entradaLog := 'Id ComptadorModel: ' || p_idComptador||
     ',Id Model: ' || p_idModelComptador||
    ' Id Ubicacio Comptador: ' || p_idUbicacioComptador||
    ' , Data modificacio Comptador: ' ||p_dataModificacioComptador||
    ' , Estat Comptador: ' || p_estatComptador ||
```

```
  ' , Numero Serie Comptador: ' || p_numeroSerieComptador||
  ' , Observacions Comptador: ' || p_observacionsComptador;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('     MODIFICAR COMPTADOR     ');
  DBMS_OUTPUT.PUT_LINE('---------------------------- ');
  DBMS_OUTPUT.PUT_LINE(' Id ComptadorModel: ' || p_idComptador||
   ',Id Model Comptador: ' || p_idModelComptador||
  ' Id Ubicacio Comptador: ' || p_idUbicacioComptador||
  ' , Data modificacio  Comptador: ' ||p_dataModificacioComptador||
  ' , Estat Comptador: ' || p_estatComptador ||
  ' , Numero Serie Comptador: ' || p_numeroSerieComptador||
  ' , Observacions Comptador: ' || p_observacionsComptador);
  c_sortidalog := 's_rsp';

     -- Comprovem que p_idComptador no sigui NULL----------------------------------------------
     If p_idComptador IS NULL then
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE('MODIFICA_COMPTADOR: ');
             s_rsp := 'Falta especificar id Comptador';
       DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_comptador;
     END IF;
     --1er. Comprovem que l'estat de la Client sigui actiu=1------------------------------
       SELECT COUNT (*) INTO n_registres
         FROM COMPTADOR
         WHERE COMPTADOR.IDCOMPTADOR= p_idComptador;

        If n_registres=0 then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('MODIFICA_COMPTADOR:  ');
         s_rsp := 'COMPTADOR no existent a la BBDD';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
         RAISE e_comptador;
        END IF;

        SELECT COMPTADOR.IDCOMPTADOR INTO n_idComptador
         FROM COMPTADOR
         WHERE COMPTADOR.IDCOMPTADOR= p_idComptador;


 s_sql := 'UPDATE COMPTADOR SET ';

 --Construirem la sentencia UPDATE segons si hi ha valor en els parÃƒÂ metres-
 IF p_idModelComptador IS NOT NULL THEN
-- Modifiquem el nom
   s_sql := s_sql || 'IDMODELOCOMPTADOR=''' || p_idModelComptador || ''',';
 END IF;
 IF p_idUbicacioComptador IS NOT NULL THEN
-- Modifiquem id Ubicacio Comptador---------------------------------------
   s_sql := s_sql || 'DUBICACOMPTADOR=''' || p_idUbicacioComptador || ''',';
 END IF;
```

```
 IF p_dataModificacioComptador IS NOT NULL THEN
-- Modifiquem DATA MODIFICACIO---------------------------------------
   s_sql := s_sql || 'DATAMODIFICACIOCOMPATOR=''' ||p_dataModificacioComptador|| ''',';
 END IF;

   IF p_estatComptador IS NOT NULL THEN
-- Modifiquem estat Comptado-----------------------------------
    s_sql := s_sql || 'IDESTADOCOMPATOR=''' || p_estatComptador || ''',';
   END IF;
   IF  p_numeroSerieComptador  IS NOT NULL THEN
-- Modifiquem numero Serie Comptador-------------------------------------------
    s_sql := s_sql || 'NUMSERIECOMPTADOR=''' || p_numeroSerieComptador || ''',';
   END IF;

   IF  p_observacionsComptador IS NOT NULL THEN

    s_sql := s_sql || 'OBSERVACIONSCOMPTADOR=''' || p_observacionsComptador || ''',';
   END IF;

-- Eliminem la coma final de la sentencia SQL-------------------------
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
   s_sql := s_sql || ' WHERE IDCOMPTADOR=''' ||n_idComptador|| '''';

EXECUTE IMMEDIATE s_sql;

 IF SQL%ROWCOUNT = 0  THEN
--L'error serÃƒÂ  que el DNI ja el tenim en la taula i no admet duplicats--
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICA COMPTADOR: ');
    s_rsp := 'Actualitzacio no realitzada';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_comptador;
 ELSE
    SELECT LOCALITAT.DESCRIPCIOLOCALITAT,
    PROVINCIA.DESCRIPCIOPROVINCIA,
    PAIS.DESCRIPCIOPAIS INTO n_localitat,n_provincia,n_pais
    FROM LOCALITAT, UBICACIO, PROVINCIA,PAIS
    WHERE LOCALITAT.IDLOCALITAT=UBICACIO.IDLOCALITATUBICACIO
    AND UBICACIO.IDUBICA=p_idUbicacioComptador
    AND LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
    AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS;

    SELECT MODELO.DESCRIPCIOMODELO INTO n_modelo
    FROM MODELO
    WHERE MODELO.IDMODELO=p_idModelComptador;

    SELECT ESTAT.DESCRIPCIOESTAT INTO n_estat
    FROM ESTAT
    WHERE ESTAT.IDESTAT=p_estatComptador;

    DBMS_OUTPUT.ENABLE;
```

```
    DBMS_OUTPUT.PUT_LINE('MODIFICA COMPTADOR: ');
    DBMS_OUTPUT.PUT_LINE(' Codi comptador: '||n_idComptador||
    ' Codi Modelo: '||p_idModelComptador ||
    ' Modelo: '||n_modelo||
    ' Data modificacio: '||p_dataModificacioComptador||
    ' Estat: '||n_estat||
    ' Numero de serie: '||p_numeroSerieComptador||
    ' Localitat: '||n_localitat||
    ' Provincia: '||n_provincia||
    ' Pais: '||n_pais||
    ' Observacions: '||p_observacionsComptador);
   DBMS_OUTPUT.PUT_LINE('********** FI DE LES OPERACIONS **********');
    s_rsp := 'OK: Modificacio satisfactoria per : Codi comptador: '||n_idComptador||
    ' Codi Modelo: '||p_idModelComptador||
    ' Modelo: '||n_modelo||
    ' Data modificacio: '||p_dataModificacioComptador||
    ' Estat: '||n_estat||
    ' Numero de serie: '||p_numeroSerieComptador||
    ' Observacions: '||p_observacionsComptador;
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END IF;
COMMIT;
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    s_rsp := 'Error: COMPTADOR duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;

END PRC_MODIFICA_COMPTADOR;

PROCEDURE PRC_CONSULTAR_COMPTADOR(
   s_rsp out NOCOPY VARCHAR2)
  AS
  CURSOR C_COMPTADOR IS
  SELECT DISTINCT COMPTADOR.IDCOMPTADOR AS CODI,
  MODELO.DESCRIPCIOMODELO AS MODELO,
```

```
COMPTADOR.DATAALTACOMPTADOR AS FECHA_ALTA,
ESTAT.DESCRIPCIOESTAT AS ESTAT,
COMPTADOR.DATAMODIFICACIOCOMPATOR AS ULTIMA_MODIFICACIO,
COMPTADOR.NUMSERIECOMPTADOR AS NUMERO_DE_SERIE,
--VIA.DESCRIPCIOVIA,
--UBICACIO.DIRECCIONUBICACIO AS DIRECCIO,
--UBICACIO.NUMEROUBICACIO,
--UBICACIO.PISUBICACIO,
--UBICACIO.PORTAUBICACIO,
--UBICACIO.CODIPOSTAL AS CODIPOSTAL,
--LOCALITAT.DESCRIPCIOLOCALITAT,
COMPTADOR.OBSERVACIONSCOMPTADOR AS OBSERVACIONS
FROM COMPTADOR, MODELO,ESTAT,UBICACIO,LOCALITAT, VIA
WHERE COMPTADOR.IDCOMPTADOR=COMPTADOR.IDCOMPTADOR
AND COMPTADOR.IDMODELOCOMPTADOR=MODELO.IDMODELO
AND COMPTADOR.IDESTADOCOMPATOR=ESTAT.IDESTAT
AND COMPTADOR.IDUBICACOMPTADOR=UBICACIO.IDUBICA
AND LOCALITAT.IDLOCALITAT=UBICACIO.IDLOCALITATUBICACIO;



 BEGIN
c_procesLog := 'PRC_CONSULTAR_COMPTADOR';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='CONSULTA COPMTADORS ' ;
    c_sortidalog := 's_rsp';

--COMPROVEM EXITEIX COMPTADOR
 SELECT COUNT (*) INTO n_registres
 FROM COMPTADOR
 WHERE COMPTADOR.IDCOMPTADOR=COMPTADOR.IDCOMPTADOR ;


 IF n_registres=0 THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('CONSULTAR ELS COMPTADORS: ');
 s_rsp := 'No hi han comptadors. ';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_comptador;

 ELSE

 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(' ');
 DBMS_OUTPUT.PUT_LINE('     CONSULTAR ELS COMPTADORS          ');
 DBMS_OUTPUT.PUT_LINE('---------------------------------------- ');

 OPEN C_COMPTADOR;
 FETCH C_COMPTADOR  INTO n_idComptador,
 n_modelo,
 n_dataAltaComptador,
 n_estat,
```

```
 n_dataModificacioComptador,
 n_numeroSerieComptador,
 n_observacionsComptador;

 WHILE C_COMPTADOR%FOUND LOOP
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(' Codi comptador: '||n_idComptador||
    ' Modelo: '||n_modelo||
    ' Data Alta: '||n_dataAltaComptador||
    ' Estat: '||n_estat||
    ' Data modificacio: '||n_dataModificacioComptador||
    ' Numero de serie: '||n_numeroSerieComptador||
    --' Via: ' ||n_via||
    --' Direccio: '||n_direccio||
    --' Numero: '||n_numero||
    --' Pis: '||n_pis||
    --' Porta: '||n_porta||
    --' Codi Postal: '||n_codiPostal||
    --' Localitat: '||n_localitat||
    ' Observacions: '||n_observacionsComptador);
     s_rsp := 'OK  CONSULTA COMPTADORS
     Codi comptador: '||n_idComptador||
    ' Modelo: '||n_modelo||
    ' Data Alta: '||n_dataAltaComptador||
    ' Estat: '||n_estat||
    ' Data modificacio: '||n_dataModificacioComptador||
    ' Numero de serie: '||n_numeroSerieComptador||
    --' Via: ' ||n_via||
    --' Direccio: '||n_direccio||
    --' Numero: '||n_numero||
    --' Pis: '||n_pis||
    --' Porta: '||n_porta||
    --' Codi Postal: '||n_codiPostal||
    --' Localitat: '||n_localitat||
    ' Observacions: '||n_observacionsComptador;
 FETCH C_COMPTADOR  INTO n_idComptador,
 n_modelo,
 n_dataAltaComptador,
 n_estat,
 n_dataModificacioComptador,
 n_numeroSerieComptador,
 n_observacionsComptador;

 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total comptador: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_COMPTADOR;
END IF;
COMMIT;
EXCEPTION
```

```
  WHEN DUP_VAL_ON_INDEX THEN
    s_rsp := 'Error: COMPTADOR duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;
END PRC_CONSULTAR_COMPTADOR;

procedure PRC_PASSAR_COMPTADOR_ALTA(
  p_idComptador in COMPTADOR.IDCOMPTADOR%TYPE,
  p_dataModificacioComptador in COMPTADOR.DATAMODIFICACIOCOMPATOR%TYPE,
  p_estatComptador in COMPTADOR.IDESTADOCOMPATOR%TYPE,
  s_rsp out NOCOPY VARCHAR2)
AS
BEGIN
    c_procesLog := 'PRC PASSAR COMPTADOR ALTA';
    c_dataHoraLog := SYSDATE;
    c_entradaLog := 'Id ComptadorModel: ' || p_idComptador||
  ' , Data modificacio Comptador: ' ||p_dataModificacioComptador;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('    MODIFICAR ESTAT COMPTADOR A ALTA    ');
   DBMS_OUTPUT.PUT_LINE('---------------------------------------- ');
   DBMS_OUTPUT.PUT_LINE(' Id ComptadorModel: ' || p_idComptador||
   ' , Data modificacio Comptador: ' ||p_dataModificacioComptador);
   c_sortidalog := 's_rsp';
 --- MIRO LA SITUACIO DEL COMPTADOR SI JA ESTA EN  ALTA AVISO
 SELECT COMPTADOR.IDESTADOCOMPATOR INTO n_estatComptadorControl
 FROM COMPTADOR
 WHERE COMPTADOR.IDCOMPTADOR=p_idComptador;

 IF n_estatComptadorControl=1 THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A ALTA: ');
 s_rsp := 'El comptador ja esta en situacio ALTA. ';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_comptador;
 END IF;
```

```
 IF p_dataModificacioComptador IS NULL THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A ALTA: ');
 s_rsp := 'Falta especificar id Comptador';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_comptador;
 END IF;
 IF p_idComptador IS NULL THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('MMODIFICAR ESTAT COMPTADOR A ALTA: ');
 s_rsp := 'Falta especificar id Comptador';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_comptador;
 END IF;
SELECT COUNT (*) INTO n_registres
FROM COMPTADOR
WHERE COMPTADOR.IDCOMPTADOR= p_idComptador;

If n_registres=0 then
  DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A ALTA:  ');
   s_rsp := 'COMPTADOR no existent a la BBDD';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_comptador;
 END IF;

 n_estatComptadorControl:=1;

 SELECT COMPTADOR.IDCOMPTADOR INTO n_idComptador
 FROM COMPTADOR
 WHERE COMPTADOR.IDCOMPTADOR= p_idComptador;

 SELECT ESTAT.DESCRIPCIOESTAT INTO n_estat
 FROM ESTAT
 WHERE ESTAT.IDESTAT=n_estatComptadorControl;

s_sql := 'UPDATE COMPTADOR SET ';

  --Construirem la sentencia UPDATE segons si hi ha valor en els parÃƒÂ metres-
   IF p_dataModificacioComptador IS NOT NULL THEN
-- Modifiquem el nom
   s_sql := s_sql || 'DATAMODIFICACIOCOMPATOR=''' ||p_dataModificacioComptador|| ''',';
 END IF;
 IF n_estatComptadorControl IS NOT NULL THEN
-- Modifiquem el nom
   s_sql := s_sql || 'IDESTADOCOMPATOR=''' ||n_estatComptadorControl|| ''',';
 END IF;

-- Eliminem la coma final de la sentencia SQL-------------------------
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
```

```
   s_sql := s_sql || ' WHERE IDCOMPTADOR="' ||n_idComptador|| '"';

EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A ALTA: ');
    s_rsp := 'Actualitzacio no realitzada';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_comptador;
   ELSE
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A ALTA:
    Codi Comptador: '||n_idComptador||
    ' Data Modificacio: '||p_dataModificacioComptador||
    ' Nou estat: '||n_estat);
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   s_rsp := 'OK: ODIFICAR ESTAT COMPTADOR A ALTA:
    Codi Comptador: '||n_idComptador||
    ' Data Modificacio: '||p_dataModificacioComptador||
    ' Nou estat: '||n_estat;
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
   END IF;

COMMIT;
 EXCEPTION

   WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
--L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
-- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
   END IF;
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE(' ');
       DBMS_OUTPUT.put_line (s_rsp);
       pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
ROLLBACK;
END PRC_PASSAR_COMPTADOR_ALTA;

procedure PRC_PASSAR_COMPTADOR_BAIXA(
  p_idComptador in COMPTADOR.IDCOMPTADOR%TYPE,
  p_dataModificacioComptador in COMPTADOR.DATAMODIFICACIOCOMPATOR%TYPE,
  p_estatComptador in COMPTADOR.IDESTADOCOMPATOR%TYPE,
  s_rsp out NOCOPY VARCHAR2)
AS
BEGIN
    c_procesLog := 'PRC PASSAR COMPTADOR BAIXA';
    c_dataHoraLog := SYSDATE;
    c_entradaLog := 'Id ComptadorModel: ' || p_idComptador||
  ' , Data modificacio Comptador: ' ||p_dataModificacioComptador;
```

```
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('     MODIFICAR ESTAT COMPTADOR A BAIXA:     ');
  DBMS_OUTPUT.PUT_LINE('---------------------------------------- ');
  DBMS_OUTPUT.PUT_LINE(' Id ComptadorModel: ' || p_idComptador||
  ' , Data modificacio Comptador: ' ||p_dataModificacioComptador);
  c_sortidalog := 's_rsp';
 --- MIRO LA SITUACIO DEL COMPTADOR SI JA ESTA EN  BAIXA AVISO
 SELECT COMPTADOR.IDESTADOCOMPATOR INTO n_estatComptadorControl
 FROM COMPTADOR
 WHERE COMPTADOR.IDCOMPTADOR=p_idComptador;

 IF n_estatComptadorControl=2 THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A BAIXA: ');
 s_rsp := 'El comptador ja esta en situacio BAIXA. ';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_comptador;
 END IF;

 IF p_dataModificacioComptador IS NULL THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A BAIXA: ');
 s_rsp := 'Falta especificar id Comptador';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_comptador;
 END IF;
 IF p_idComptador IS NULL THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A BAIXA: ');
 s_rsp := 'Falta especificar id Comptador';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_comptador;
 END IF;
SELECT COUNT (*) INTO n_registres
FROM COMPTADOR
WHERE COMPTADOR.IDCOMPTADOR= p_idComptador;

If n_registres=0 then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A BAIXA:  ');
  s_rsp := 'COMPTADOR no existent a la BBDD';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_comptador;
 END IF;

 n_estatComptadorControl:=2;

 SELECT COMPTADOR.IDCOMPTADOR INTO n_idComptador
 FROM COMPTADOR
 WHERE COMPTADOR.IDCOMPTADOR= p_idComptador;
```

```
 SELECT ESTAT.DESCRIPCIOESTAT INTO n_estat
 FROM ESTAT
 WHERE ESTAT.IDESTAT=n_estatComptadorControl;

s_sql := 'UPDATE COMPTADOR SET ';

 --Construirem la sentencia UPDATE segons si hi ha valor en els parÃƒÂ metres-
  IF p_dataModificacioComptador IS NOT NULL THEN
-- Modifiquem el nom
   s_sql := s_sql || 'DATAMODIFICACIOCOMPATOR=''' ||p_dataModificacioComptador|| ''',';
 END IF;
 IF n_estatComptadorControl IS NOT NULL THEN
-- Modifiquem el nom
   s_sql := s_sql || 'IDESTADOCOMPATOR=''' ||n_estatComptadorControl|| ''',';
 END IF;

-- Eliminem la coma final de la sentencia SQL--------------------------
   s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
   s_sql := s_sql || ' WHERE IDCOMPTADOR=''' ||n_idComptador|| '''';

EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0 THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A BAIXA: ');
     s_rsp := 'Actualitzacio no realitzada';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_comptador;
   ELSE
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR ESTAT COMPTADOR A BAIXA:
     Codi Comptador: '||n_idComptador||
     ' Data Modificacio: '||p_dataModificacioComptador||
     ' Nou estat: '||n_estat);
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   s_rsp := 'OK: MODIFICAR ESTAT COMPTADOR A BAIXA:
     Codi Comptador: '||n_idComptador||
     ' Data Modificacio: '||p_dataModificacioComptador||
     ' Nou estat: '||n_estat;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    END IF;

COMMIT;
 EXCEPTION

   WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
--L'error no ha estat controlat per codi---------------------------------------
       s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
-- L'error si ha estat controlat per codi---------------------------------------
       s_rsp := 'Error: ' || s_rsp;
```

```
    END IF;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE(' ');
        DBMS_OUTPUT.put_line (s_rsp);
        pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
ROLLBACK;
END PRC_PASSAR_COMPTADOR_BAIXA;

END "GESTION_COMPTADOR";
```

11.11 TRATAMIENTO DE GESTION CONECTAR CENTRALES DE PRODUCCION A LAS LINEAS CON LAS CENTRALES DE DISTRIBUCION.

Gestiona la conectividad entre las centrales de Producción y Ditribución.

DEFINICION	PROCEDIMIENTO
Alta de Conexión Central	PRC_CONECTA_CENTRALS
Modificar la Conexión de la Central	PRC_MODIFICA_CONNEXIO
Consulta una Central Conectada o no Conectada	PRC_CONSULTAR_CONEXIO_CENT
Consulta de todas la Conexiones entre Centrales	PRC_CONSULTAR_CONEXIONS_CENT

```
CREATE OR REPLACE
PACKAGE        "GESTION_CONECTA_CENTRALS" AS

PROCEDURE PRC_CONECTA_CENTRALS(
   p_dataAltaConecta in LINEA_CONECTA_CENTRAL.DATAALTAC%Type,
   p_conectaCentralProduccio in LINEA_CONECTA_CENTRAL.CONNECTACPRODU%Type,
   p_conectaCentralDistribucio in LINEA_CONECTA_CENTRAL.CONNECTACDISTRI%Type,
   p_observacionsConecta in LINEA_CONECTA_CENTRAL.OBSERVACIOC%Type,
   s_rsp out NOCOPY VARCHAR);

--- MODIFICAR CONEXIONS DE CENTRALS

PROCEDURE PRC_MODIFICA_CONNEXIO(
   p_idConectaCentral in LINEA_CONECTA_CENTRAL.IDCONECTA%Type,
   p_estatConecta in LINEA_CONECTA_CENTRAL.IDESTATC%Type,
```

```
    p_dataModificaConectaCentral in LINEA_CONECTA_CENTRAL.DATAMODIFICACIOC%Type,
    p_conectaCentralProduccio in LINEA_CONECTA_CENTRAL.CONNECTACPRODU%Type,
    p_conectaCentralDistribucio in LINEA_CONECTA_CENTRAL.CONNECTACDISTRI%Type,
    p_observacionsConecta in LINEA_CONECTA_CENTRAL.OBSERVACIOC%Type,
    s_rsp out NOCOPY VARCHAR);

-- CONSULTA CONEXIONS DE CENTRALS

PROCEDURE PRC_CONSULTAR_CONEXIO_CENT(
  p_idConectaCentral in LINEA_CONECTA_CENTRAL.IDCONECTA%Type,
  s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_CONSULTAR_CONEXIONS_CENT(
   s_rsp out NOCOPY VARCHAR2);

END GESTION_CONECTA_CENTRALS;
```

11.11.1PROCEDIMIENTO SPL PACKAGE CONECTAR CENTRALES DE PRODUCCION A LAS LINEAS CON LAS CENTRALES DE DISTRIBUCION.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_CONECTA_CENTRALS" AS
  n_registres     NUMBER;
  s_sql         VARCHAR2 (2000);
  n_NUM_ERR          NUMBER(10);

  c_procesLog          LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp      LOG_TFC.RSPLOG%TYPE;

  -- VARIABLE PER OBTENIR CENTRALS DE PRODUCCIO
  n_tipusCentralProduccio NUMBER;
  -- VARIABLE PER OBTENIR CENTRALS DE DISTRIBUCIO
  n_tipusCentralDistribucio NUMBER;
  -- VARIABLE DE CONTROL CONNEXIO MÃ‰S DE DUES CENTRALS DE DISTRIBUCIO
  -- AMB UNA DE PRODUCCIO
  n_connexio NUMBER;
  n_conectarCentral NUMBER;
  n_idConectaCentral  LINEA_CONECTA_CENTRAL.IDCONECTA%Type;
  n_estatConecta LINEA_CONECTA_CENTRAL.IDESTATC%Type;
  n_dataAltaConecta   LINEA_CONECTA_CENTRAL.DATAALTAC%Type;
  n_conectaCentralProduccio   LINEA_CONECTA_CENTRAL.CONNECTACPRODU%TYPE;
  n_conectaCentralDistribucio  LINEA_CONECTA_CENTRAL.CONNECTACDISTRI%TYPE;
  n_dataModificacioConecta LINEA_CONECTA_CENTRAL.DATAMODIFICACIOC%TYPE;
  n_observacions LINEA_CONECTA_CENTRAL.OBSERVACIOC%TYPE;

  n_controlEstatPro ESTAT.IDESTAT%TYPE;
```

```
  n_produccio NUMBER;
  n_controlEstatDis ESTAT.IDESTAT%TYPE;
  n_distribucio NUMBER;
  n_descEstatConecta ESTAT.DESCRIPCIOESTAT%TYPE;
  n_lineaProduccio NUMBER;
  n_lineaDistribucio NUMBER;

  sortida VARCHAR2(1000):='';
  e_conceta_centrals   EXCEPTION;
-----------------------------------------------------------------
/*****************************************************************************
      NOM: PRC_CONECTA_CENTRALS
      DESCRIPCIO: procediment per conectar una central de Produccio amb
      una o dues de Distribucio.
*****************************************************************************/
PROCEDURE PRC_CONECTA_CENTRALS(
   p_dataAltaConecta in LINEA_CONECTA_CENTRAL.DATAALTAC%Type,
   p_conectaCentralProduccio in LINEA_CONECTA_CENTRAL.CONNECTACPRODU%Type,
   p_conectaCentralDistribucio in LINEA_CONECTA_CENTRAL.CONNECTACDISTRI%Type,
   p_observacionsConecta in LINEA_CONECTA_CENTRAL.OBSERVACIOC%Type,
   s_rsp out NOCOPY VARCHAR)
AS
BEGIN
   c_procesLog := 'PRC_CONECTA_CENTRALS';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Data Alta: ' ||p_dataAltaConecta||
   ', Codi Central Produccio: ' ||p_conectaCentralProduccio||
   ', Codi Central Distribucio: ' ||p_conectaCentralDistribucio||
   ', Observacions Connexio: ' ||p_observacionsConecta;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('     ASSIGNAR CENTRAL UNA CONNEXIO      ');
   DBMS_OUTPUT.PUT_LINE('----------------------------------------');
   DBMS_OUTPUT.PUT_LINE('     Data Alta connexio: ' ||p_dataAltaConecta||
   ', Codi Central Produccio: ' ||p_conectaCentralProduccio||
   ', Codi Central Distribucio: ' ||p_conectaCentralDistribucio||
   ', Observacions Connexio: ' ||p_observacionsConecta);
   c_sortidalog := 's_rsp';

-- mirem si les dades no son nules
if p_dataAltaConecta IS NULL then
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('ASSIGNAR CENTRAL UNA CONNEXIO:  ');
  s_rsp := 'Falta especificar Data Alta. ';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_conceta_centrals;
END IF;
if p_conectaCentralProduccio IS NULL then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('ASSIGNAR CENTRAL CONEXIO:  ');
   s_rsp := 'Falta el codi de la central de produccio';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
```

```
    RAISE e_conceta_centrals;
END IF;
if p_conectaCentralDistribucio IS NULL then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('ASSIGNAR CENTRAL CONEXIO:  ');
   s_rsp := 'Falta el codi de la central de distribucio';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_conceta_centrals;
END IF;
-- mirem si hi ha mes d'un connexio de central de distribucio amb una de produccio
SELECT DISTINCT COUNT(p_conectaCentralProduccio) INTO n_connexio
FROM LINEA_CONECTA_CENTRAL
WHERE LINEA_CONECTA_CENTRAL.CONNECTACDISTRI=p_conectaCentralDistribucio;

-- si hi ha mes de dues llavors no pot fer-se la conexio en cas
-- contrari continua en sequencia.
IF n_connexio>2 THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('ASSIGNAR CENTRAL UNA CONNEXIO:  ');
 s_rsp := 'La central de distribucio esta connectada amb dues Centrals de produccio,
 no pot connectar-se amb aquesta central de produccio. ';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_conceta_centrals;
ELSE
n_conectarCentral:=1;
END IF;
-- EXAMINEN la central de produccio.
-- n_produccio si hi ha 1 llavors exiteix la central de produccio
-- sino no es central de produccio i emet excepcio.
SELECT DISTINCT COUNT (CENTRAL.ESTATCENTRAL) INTO n_produccio
FROM CENTRAL, CLASSE_CENTRAL, ESTAT
WHERE CENTRAL.IDCLASSECENTRAL=1
AND CLASSE_CENTRAL.IDCLASSE=CENTRAL.ESTATCENTRAL
AND CENTRAL.ESTATCENTRAL=ESTAT.IDESTAT
AND CENTRAL.IDTIPOCENTRAL<>6
AND CENTRAL.IDCENTRAL=p_conectaCentralProduccio;
-- si n_produccio=1 llavors a n_controlEstatPro hi ha 1=ALTA
-- o 2=Baixa en cas contrari no es una central de produccio o
-- bé no existeix
IF n_produccio=1 THEN
SELECT DISTINCT CENTRAL.ESTATCENTRAL INTO n_controlEstatPro
FROM CENTRAL, CLASSE_CENTRAL, ESTAT
WHERE CENTRAL.IDCLASSECENTRAL=1
AND CLASSE_CENTRAL.IDCLASSE=CENTRAL.ESTATCENTRAL
AND CENTRAL.ESTATCENTRAL=ESTAT.IDESTAT;
ELSE
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('ASSIGNAR CENTRAL UNA CONNEXIO:  ');
 s_rsp := 'La central de produccio no exiteix a la BDD. ';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_conceta_centrals;
END IF;
```

```
-- EXAMINEN la central de distribucio.
-- n_distribucio si hi ha 1 llavors exiteix la central de distribucio
-- sino no es central de distribucio i emet excepcio.
SELECT DISTINCT COUNT (CENTRAL.ESTATCENTRAL) INTO n_distribucio
FROM CENTRAL, CLASSE_CENTRAL, ESTAT
WHERE CENTRAL.IDCLASSECENTRAL=2
AND CLASSE_CENTRAL.IDCLASSE=CENTRAL.ESTATCENTRAL
AND CENTRAL.ESTATCENTRAL=ESTAT.IDESTAT
AND CENTRAL.IDTIPOCENTRAL=6
AND CENTRAL.IDCENTRAL=p_conectaCentralDistribucio;
IF n_distribucio=1 THEN
SELECT DISTINCT CENTRAL.ESTATCENTRAL INTO n_controlEstatDis
FROM CENTRAL, CLASSE_CENTRAL, ESTAT
WHERE CENTRAL.IDCLASSECENTRAL=2
AND CLASSE_CENTRAL.IDCLASSE=CENTRAL.ESTATCENTRAL
AND CENTRAL.ESTATCENTRAL=ESTAT.IDESTAT
AND CENTRAL.IDTIPOCENTRAL=6
AND CENTRAL.IDCENTRAL=p_conectaCentralDistribucio;
ELSE
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('ASSIGNAR CENTRAL UNA CONNEXIO:  ');
 s_rsp := 'La central de distribucio no exiteix a la BDD. ';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_conceta_centrals;
END IF;

IF n_conectarCentral>2 AND n_distribucio=0 AND n_produccio=0 THEN
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('ASSIGNAR CENTRAL UNA CONNEXIO:  ');
   s_rsp := 'No pot rfer-se la conexio. ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_conceta_centrals;
ELSE

 SELECT COUNT(*) INTO n_registres
 FROM LINEA_CONECTA_CENTRAL
 WHERE LINEA_CONECTA_CENTRAL.CONNECTACPRODU=p_conectaCentralProduccio
 AND LINEA_CONECTA_CENTRAL.CONNECTACDISTRI=p_conectaCentralDistribucio
 AND LINEA_CONECTA_CENTRAL.DATAALTAC=p_dataAltaConecta;
END IF;

IF n_registres=0 THEN
n_estatConecta:=1;
-- El donem d'alta la Linea de conexio entre les centrals de produccio i distribucio
 INSERT INTO LINEA_CONECTA_CENTRAL(
                        DATAALTAC,
                        IDESTATC,
                        CONNECTACPRODU,
                        CONNECTACDISTRI,
                        OBSERVACIOC)
          VALUES(p_dataAltaConecta,
              n_estatConecta,
```

```
            p_conectaCentralProduccio,
            p_conectaCentralDistribucio,
            p_observacionsConecta);
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('ASSIGNAR CENTRAL UNA CONNEXIO:
  La Central de Produccio: '||p_conectaCentralProduccio||
  ' amb la Central de Distribucio: '||p_conectaCentralDistribucio||
  ' Connectada: '||n_controlEstatPro||
  ' ha estat inserida i conectada satisfactiament.');
  DBMS_OUTPUT.PUT_LINE(' ********** FIN DE LES OPERACIONS **********');
  --Gravem en la taula log----------------------------------------------------
  s_rsp :='Ok La Central de Produccio: '||p_conectaCentralProduccio||
  ' amb la Central de Distribucio: '||p_conectaCentralDistribucio||
  ' Connectada: '||n_controlEstatPro||
  ' ha estat inserida i conectada satisfactiament.';
  pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  DBMS_OUTPUT.PUT_LINE(s_rsp);
ELSE
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('ASSIGNAR CENTRAL UNA CONNEXIO: ');
  s_rsp := 'Linea duplicada';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_conceta_centrals;
END IF;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi
       s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi
       s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;

END PRC_CONECTA_CENTRALS;
/*****************************************************************************
NOM: PRC_MODIFICA_CONNEXIO
DESCRIPCIO: Procediment per modificar una connexio.
*****************************************************************************/

PROCEDURE PRC_MODIFICA_CONNEXIO(
    p_idConectaCentral in LINEA_CONECTA_CENTRAL.IDCONECTA%Type,
    p_estatConecta in LINEA_CONECTA_CENTRAL.IDESTATC%Type,
    p_dataModificaConectaCentral in LINEA_CONECTA_CENTRAL.DATAMODIFICACIOC%Type,
    p_conectaCentralProduccio in LINEA_CONECTA_CENTRAL.CONNECTACPRODU%Type,
    p_conectaCentralDistribucio in LINEA_CONECTA_CENTRAL.CONNECTACDISTRI%Type,
```

```
   p_observacionsConecta in LINEA_CONECTA_CENTRAL.OBSERVACIOC%Type,
   s_rsp out NOCOPY VARCHAR)
AS
BEGIN
   c_procesLog := ' PRC_MODIFICA_CONNEXIO';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' CCodi Linea conexio centrals: '||p_idConectaCentral||
   ', Data Modificacio: ' ||p_dataModificaConectaCentral||
   ', Codi Central Produccio: ' ||p_conectaCentralProduccio||
   ', Codi Central Distribucio: ' ||p_conectaCentralDistribucio||
   ', Codi Estat Linea: ' ||p_estatConecta||
   ', Observacions Connexio: ' ||p_observacionsConecta;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE(' MODIFICAR LA CONEXIO DE LES CENTRALS ');
   DBMS_OUTPUT.PUT_LINE('------------------------------------ ');
   DBMS_OUTPUT.PUT_LINE('
     Codi Linea conexio centrals: '||p_idConectaCentral||
   ', Data Modificacio: ' ||p_dataModificaConectaCentral||
   ', Codi Central Produccio: ' ||p_conectaCentralProduccio||
   ', Codi Central Distribucio: ' ||p_conectaCentralDistribucio||
   ', Codi Estat Linea: ' ||p_estatConecta||
   ', Observacions Connexio: ' ||p_observacionsConecta);
   c_sortidalog := 's_rsp';

    if  p_dataModificaConectaCentral IS NULL then
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CONEXIO DE LES CENTRALS:  ');
          s_rsp := 'Falta especificar Data de la modificacio ';
          DBMS_OUTPUT.PUT_LINE(s_rsp);
         RAISE e_conceta_centrals;
    END IF;
    if p_conectaCentralProduccio IS NULL then
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CONEXIO DE LES CENTRALS::  ');
          s_rsp := 'Falta el codi de la central de produccio.';
          DBMS_OUTPUT.PUT_LINE(s_rsp);
         RAISE e_conceta_centrals;
    END IF;
     if p_conectaCentralDistribucio IS NULL then
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CONEXIO DE LES CENTRALS::  ');
          s_rsp := 'Falta el codi de la central de distribucio. ';
          DBMS_OUTPUT.PUT_LINE(s_rsp);
         RAISE e_conceta_centrals;
    END IF;
    -- obting el lietral d'estat
    SELECT ESTAT.DESCRIPCIOESTAT INTO n_descEstatConecta
    FROM ESTAT
    WHERE ESTAT.IDESTAT=p_estatConecta;
    -- OBTING EL CODI ESTAT DE LA LINEA CONECTADA DE CENTRALS
    SELECT  DISTINCT LINEA_CONECTA_CENTRAL.IDESTATC INTO n_estatConecta
```

```
     FROM LINEA_CONECTA_CENTRAL
     WHERE LINEA_CONECTA_CENTRAL.IDESTATC=p_estatConecta;
    -- MIRO SI LES DADES SON CORRECTES
    IF n_estatConecta>2 OR n_estatConecta<1 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CONEXIO DE LES CENTRALS:  ');
    s_rsp := 'El codi estat connexio no es correcte, te que ser 1=alta o 2=baixa. ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_conceta_centrals;
    END IF;
--miren si exiteixen les centrals en la BDD en la conectivitat
--CERCA CENTRAL PRODUCCIO RECUPEREM LA LINEA DE CONEXIO
    SELECT DISTINCT COUNT(LINEA_CONECTA_CENTRAL.IDCONECTA) INTO n_produccio
    FROM LINEA_CONECTA_CENTRAL
    WHERE LINEA_CONECTA_CENTRAL.CONNECTACPRODU=p_conectaCentralProduccio;

    SELECT DISTINCT LINEA_CONECTA_CENTRAL.IDCONECTA INTO n_lineaProduccio
    FROM LINEA_CONECTA_CENTRAL
    WHERE LINEA_CONECTA_CENTRAL.CONNECTACPRODU=p_conectaCentralProduccio;
If n_produccio=0  then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CONEXIO DE LES CENTRALS: ');
      s_rsp := 'No hi linea cenectada no existent a la BBDD';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_conceta_centrals;
    END IF;
  -- n_linea hi ha la linea ce conexio
  s_sql := 'UPDATE LINEA_CONECTA_CENTRAL SET ';
  -- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-

    -- Modifiquem Data Estat Linea
    IF p_estatConecta IS NOT NULL THEN
    s_sql := s_sql || 'IDESTAT=''' ||p_estatConecta || ''',';
    END IF;
    -- MODIFICO LA DATA
    IF p_dataModificaConectaCentral IS NOT NULL THEN
    -- Modifiquem data Linea
    s_sql := s_sql || 'DATAMODIFICACIOC=''' ||p_dataModificaConectaCentral|| ''',';
    END IF;
    IF p_conectaCentralProduccio IS NOT NULL THEN
    -- Modifiquem central produccio
      s_sql := s_sql || 'CONNECTACPRODU=''' ||p_conectaCentralProduccio|| ''',';
    END IF;
    IF p_conectaCentralDistribucio IS NOT NULL THEN
     -- Modifiquem central de distribucio
    s_sql := s_sql || 'CONNECTACDISTRI=''' ||p_conectaCentralDistribucio|| ''',';
    END IF;
    IF p_observacionsConecta IS NOT NULL THEN
    -- Modifiquem Observacions Inspeccio
    s_sql := s_sql || 'OBSERVACIOC=''' ||p_observacionsConecta || ''',';
    END IF;
  -- Eliminem la coma final de la sentencia SQL-------------------------
```

```
    s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
    -- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
    s_sql := s_sql || ' WHERE IDCONECTA=''' ||n_lineaProduccio|| '''';

EXECUTE IMMEDIATE s_sql;

  IF SQL%ROWCOUNT = 0 THEN
  --L'error serÃ  que Tipo Linea ja el tenim en la taula i no admet duplicats--
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICAR LA CONEXIO DE LES CENTRALS: ');
  s_rsp := 'Actualitzacio no realitzada';
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_conceta_centrals;
  ELSE
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('MODIFICACIO de CONEXIO feta en
  Codi Linea connexio Centrals: '||p_idConectaCentral||
  ' amb Data de Modificacio: '||p_dataModificaConectaCentral||
  ' Estat modificat: '||n_descEstatConecta||
  ' Central Produccio: '||p_conectaCentralProduccio||
  ' Central Distribucio:'||p_conectaCentralDistribucio||
  ' Observacions: '||p_observacionsConecta);
  DBMS_OUTPUT.PUT_LINE('********** FI DE LES OPERACIONS **********');
  s_rsp := 'OK: MODIFICACIO de CONEXIO feta en
  Codi Linea connexio Centrals: '||p_idConectaCentral||
  ' amb Data de Modificacio: '||p_dataModificaConectaCentral||
  ' Estat modificat: '||n_descEstatConecta||
  ' Central Produccio: '||p_conectaCentralProduccio||
  ' Central Distribucio:'||p_conectaCentralDistribucio||
  ' Observacions: '||p_observacionsConecta;
  pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
END IF;
COMMIT;
  EXCEPTION
   WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
-- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
-- L'error si ha estat controlat per codi
     s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;

END PRC_MODIFICA_CONNEXIO;


PROCEDURE PRC_CONSULTAR_CONEXIO_CENT(
```

```
 p_idConectaCentral in LINEA_CONECTA_CENTRAL.IDCONECTA%Type,
 s_rsp out NOCOPY VARCHAR2)
 AS
 CURSOR C_CONEXIONS_CENT IS
 SELECT LINEA_CONECTA_CENTRAL.IDCONECTA,
 LINEA_CONECTA_CENTRAL.DATAALTAC,
 ESTAT.DESCRIPCIOESTAT,
 LINEA_CONECTA_CENTRAL.DATAMODIFICACIOC,
 LINEA_CONECTA_CENTRAL.CONNECTACPRODU,
 LINEA_CONECTA_CENTRAL.CONNECTACDISTRI,
 LINEA_CONECTA_CENTRAL.OBSERVACIOC
 FROM LINEA_CONECTA_CENTRAL,ESTAT
 WHERE LINEA_CONECTA_CENTRAL.IDCONECTA=p_idConectaCentral
 AND LINEA_CONECTA_CENTRAL.IDESTATC=ESTAT.IDESTAT;
BEGIN
  c_procesLog := 'CONSULTAR_CONEXIO_CENT';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Codi Linea: '||p_idConectaCentral;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('    CONSULTA UNA CONEXIO DE CENTRALS    ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------------------');
   DBMS_OUTPUT.PUT_LINE(' Codi Linea: '||p_idConectaCentral);
   c_sortidalog := 's_rsp';

   SELECT COUNT(*) INTO n_registres
   FROM LINEA_CONECTA_CENTRAL
   WHERE LINEA_CONECTA_CENTRAL.IDCONECTA=p_idConectaCentral;

   IF n_registres=0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' CONSULTA UNA CONEXIO DE CENTRALS: ');
    s_rsp := 'No hi linea cenectada no existent a la BBDD';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_conceta_centrals;
   END IF;
OPEN C_CONEXIONS_CENT;
FETCH C_CONEXIONS_CENT INTO
n_idConectaCentral,
n_dataAltaConecta,
n_descEstatConecta,
n_dataModificacioConecta,
n_conectaCentralProduccio,
n_conectaCentralDistribucio,
n_observacions;
WHILE C_CONEXIONS_CENT%FOUND LOOP
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(
 '   Codi linea connexio Central:  '
 ||n_idConectaCentral||
 ' Data alta linea connexio:  '
 ||n_dataAltaConecta||
```

```
 ' Estat:  '
 ||n_descEstatConecta||
 ' Data Modificacio: '
 ||n_dataModificacioConecta||
 ' Codi Central Produccio: '
 ||n_conectaCentralProduccio||
 ' Codi Central Distribucio: '
 ||n_conectaCentralDistribucio||
  ' Observacions: '
 ||n_observacions);
  s_rsp := 'OK  CONSULTA UNA CONEXIO DE CENTRALS
   Codi linea connexio Central:  '
 ||n_idConectaCentral||
 ' Data alta linea connexio:  '
 ||n_dataAltaConecta||
 ' Estat:  '
 ||n_descEstatConecta||
 ' Data Modificacio: '
 ||n_dataModificacioConecta||
 ' Codi Central Produccio: '
 ||n_conectaCentralProduccio||
 ' Codi Central Distribucio: '
 ||n_conectaCentralDistribucio||
  ' Observacions: '
 ||n_observacions;
FETCH C_CONEXIONS_CENT INTO
n_idConectaCentral,
n_dataAltaConecta,
n_descEstatConecta,
n_dataModificacioConecta,
n_conectaCentralProduccio,
n_conectaCentralDistribucio,
n_observacions;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total connexio de centrals: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');

CLOSE C_CONEXIONS_CENT;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
      ELSE
         -- L'error si ha estat controlat per codi
         s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
```

```
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 ROLLBACK;
END PRC_CONSULTAR_CONEXIO_CENT;

PROCEDURE PRC_CONSULTAR_CONEXIONS_CENT(
  s_rsp out NOCOPY VARCHAR2)
AS
 CURSOR C_CONEXIO_CENT IS
 SELECT LINEA_CONECTA_CENTRAL.IDCONECTA,
 LINEA_CONECTA_CENTRAL.DATAALTAC,
 ESTAT.DESCRIPCIOESTAT,
 LINEA_CONECTA_CENTRAL.DATAMODIFICACIOC,
 LINEA_CONECTA_CENTRAL.CONNECTACPRODU,
 LINEA_CONECTA_CENTRAL.CONNECTACDISTRI,
 LINEA_CONECTA_CENTRAL.OBSERVACIOC
 FROM LINEA_CONECTA_CENTRAL,ESTAT
 WHERE LINEA_CONECTA_CENTRAL.IDCONECTA=LINEA_CONECTA_CENTRAL.IDCONECTA
 AND LINEA_CONECTA_CENTRAL.IDESTATC=ESTAT.IDESTAT;
BEGIN
  c_procesLog := 'PRC_CONSULTAR_CONEXIONS_CENT';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='CONSULTA CONNEXIOS DE CENTRALS  ';
   c_sortidalog := 's_rsp';

   SELECT COUNT(*) INTO n_registres
   FROM LINEA_CONECTA_CENTRAL
   WHERE LINEA_CONECTA_CENTRAL.IDCONECTA=LINEA_CONECTA_CENTRAL.IDCONECTA;

   IF n_registres=0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' CONSULTA CONNEXIOS DE CENTRALS: ');
    s_rsp := 'No hi linea cenectada no existent a la BBDD';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_conceta_centrals;
   END IF;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('    CONSULTA CONNEXIOS DE CENTRALS      ');
   DBMS_OUTPUT.PUT_LINE('----------------------------------------');

OPEN C_CONEXIO_CENT;
FETCH C_CONEXIO_CENT INTO
n_idConectaCentral,
n_dataAltaConecta,
n_descEstatConecta,
n_dataModificacioConecta,
n_conectaCentralProduccio,
n_conectaCentralDistribucio,
n_observacions;
WHILE C_CONEXIO_CENT%FOUND LOOP
```

```
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(
 '   Codi linea connexio Central:  '
 ||n_idConectaCentral||
 ' Data alta linea connexio:  '
 ||n_dataAltaConecta||
 ' Estat:  '
 ||n_descEstatConecta||
 ' Data Modificacio: '
 ||n_dataModificacioConecta||
 ' Codi Central Produccio: '
 ||n_conectaCentralProduccio||
 ' Codi Central Distribucio: '
 ||n_conectaCentralDistribucio||
  ' Observacions: '
 ||n_observacions);
  s_rsp := 'OK  CONSULTA UNA CONEXIO DE CENTRALS
   Codi linea connexio Central:  '
 ||n_idConectaCentral||
 ' Data alta linea connexio:  '
 ||n_dataAltaConecta||
 ' Estat:  '
 ||n_descEstatConecta||
 ' Data Modificacio: '
 ||n_dataModificacioConecta||
 ' Codi Central Produccio: '
 ||n_conectaCentralProduccio||
 ' Codi Central Distribucio: '
 ||n_conectaCentralDistribucio||
  ' Observacions: '
 ||n_observacions;
FETCH C_CONEXIO_CENT INTO
n_idConectaCentral,
n_dataAltaConecta,
n_descEstatConecta,
n_dataModificacioConecta,
n_conectaCentralProduccio,
n_conectaCentralDistribucio,
n_observacions;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total connexio de centrals: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');

CLOSE C_CONEXIO_CENT;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi
```

```
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('  ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;

END PRC_CONSULTAR_CONEXIONS_CENT;

END "GESTION_CONECTA_CENTRALS";
```

11.12 TRATAMIENTO DE GESTION CONECTAR CONTADORES A LAS LINEAS DE LAS CENTRALES DE DISTRIBUCION.

Gestiona la conectividad entre de los contadores con las líneas de las Centrales de Distribución.

DEFINICION	PROCEDIMIENTO
Alta de Conexión Central	PRC_CONECTA_CENTRALS
Modificar la Conexión de la Central	PRC_MODIFICA_CONNEXIO
Consulta una Central Conectada o no Conectada	PRC_CONSULTAR_CONEXIO_CENT
Consulta de todas la Conexiones entre Centrales	PRC_CONSULTAR_CONEXIONS_CENT

```
CREATE OR REPLACE PACKAGE        "GESTION_LINEA_CONECTAR" AS

PROCEDURE PRC_CONECTA_LINEA_COMPTADOR(
   p_idTipoLineaConecta in LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA%Type,
   p_dataAltaLineaConecta in LINEA_CONECTAR_COMPTADOR.DATAALTALINEACONECTA%Type,
   p_conectaCentralDistribucio in LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA%Type,
   p_conectaComptador in LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA%Type,
   p_parConecta in LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA%TYPE,
   p_observacionsLineaConecta in LINEA_CONECTAR_COMPTADOR.OBSERVACIOLINEACONECTA%Type,
```

```
   s_rsp out NOCOPY VARCHAR);

--- MODIFICAR CONEXIONS DE COMPTADOR

PROCEDURE PRC_MODIFICA_CONECTA_COMPTADOR(
   p_idLineaConecta in LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%TYPE,
   p_idTipoLineaConecta in LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA%Type,
   p_dataModificaLineaConecta in LINEA_CONECTAR_COMPTADOR.DATAMODIFICACIOLINEACONECTA%Type,
   p_estatLineaConecta in LINEA_CONECTAR_COMPTADOR.IDESTATLINEACONECTA%Type,
   p_conectaCentralDistribucio in LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA%Type,
   p_conectaComptador in LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA%Type,
   p_parConecta in LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA%TYPE,
   p_observacionsLineaConecta in LINEA_CONECTAR_COMPTADOR.OBSERVACIOLINEACONECTA%Type,
   s_rsp out NOCOPY VARCHAR);

-- CONSULTA CONEXIONS UNA CENTRAL A UN COMPTADOR

PROCEDURE PRC_CONSULTAR_CONEXIO_LINEA(
  p_idLineaConecta in LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%TYPE,
  s_rsp out NOCOPY VARCHAR2);

PROCEDURE PRC_CONSULTAR_LINEAS(
   s_rsp out NOCOPY VARCHAR2);

END "GESTION_LINEA_CONECTAR";
```

11.12.1 PROCEDIMIENTO SPL PACKAGE CONECTAR CONTADORES A LAS LINEAS DE LAS CENTRALES DE DISTRIBUCION.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_LINEA_CONECTAR" AS
 n_registres     NUMBER;
  s_sql          VARCHAR2 (2000);
  n_NUM_ERR           NUMBER(10);

  c_procesLog          LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp      LOG_TFC.RSPLOG%TYPE;

  n_dataAltaLinea LINEA_CONECTAR_COMPTADOR.DATAALTALINEACONECTA%TYPE;
  n_dataModificacioLinea LINEA_CONECTAR_COMPTADOR.DATAMODIFICACIOLINEACONECTA%TYPE;
  n_observacions LINEA_CONECTAR_COMPTADOR.OBSERVACIOLINEACONECTA%TYPE;
  n_idTpusLinea LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA%TYPE;
  n_idconectaComptador NUMBER;
  n_controlCentral NUMBER;
  n_idTipoLineaConecta  LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA%TYPE;
  n_conectaCentralDistribucio  LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA%TYPE;
  n_conectaComptador  LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA%TYPE;
```

```
  n_idLineaConecta   LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%Type;
  n_parConecta LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA%TYPE;
  n_controlEstat ESTAT.IDESTAT%TYPE;
  n_descEstat ESTAT.DESCRIPCIOESTAT%TYPE;
  n_controlTipoLinea LINEA_TIPUS.IDTLINEA%TYPE;
  n_consumToleratLinea LINEA_TIPUS.CONSUMMAXIMTLINEA%TYPE;
  n_primerPart NUMBER;
  n_segonPart NUMBER;
  n_lineaDis_1 NUMBER;
  n_lineaDis_2 NUMBER;
  n_conectaCentralDistribucio_1  LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA%TYPE;
  n_consumToleratLinea_1 NUMBER;
  n_conectaCentralDistribucio_2  LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA%TYPE;
  n_consumToleratLinea_2 NUMBER;
  n_centralMaxEner NUMBER;
  n_controlPar NUMBER;
  sortida VARCHAR2(1000):='';
  e_linea_conecta   EXCEPTION;

/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                              TFC: CONTROL ENERGIA.
PROCEDIMENT ALTA CONNEXIO LINEAS AMB COMPTADORS
*****************************************************************************/
PROCEDURE PRC_CONECTA_LINEA_COMPTADOR(
   p_idTipoLineaConecta in LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA%Type,
   p_dataAltaLineaConecta in LINEA_CONECTAR_COMPTADOR.DATAALTALINEACONECTA%Type,
   p_conectaCentralDistribucio in LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA%Type,
   p_conectaComptador in LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA%Type,
   --p_parConecta in LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA%TYPE,
   p_observacionsLineaConecta in LINEA_CONECTAR_COMPTADOR.OBSERVACIOLINEACONECTA%Type,
   s_rsp out NOCOPY VARCHAR)
AS

BEGIN
   c_procesLog := 'PRC_CONECTA_LINEA_COMPTADOR';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Tipus Linea: ' || p_idTipoLineaConecta||
   ', Data Alta connexio Linea: ' ||p_dataAltaLineaConecta ||
   ', Central Distribucio: ' ||p_conectaCentralDistribucio||
   ', Comptador: ' ||p_conectaComptador||
   ', Observacions  Linea: ' ||p_observacionsLineaConecta;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('    ALTA CONNEXIO COMPTADOR EN LINEAS   ');
    DBMS_OUTPUT.PUT_LINE('---------------------------------------- ');
    DBMS_OUTPUT.PUT_LINE(' Codi Tipus Linea: ' || p_idTipoLineaConecta||
   ', Data Alta connexio Linea: ' ||p_dataAltaLineaConecta ||
   ', Central Distribucio: ' ||p_conectaCentralDistribucio||
   ', Comptador: ' ||p_conectaComptador||
   ', Observacions  Linea: ' ||p_observacionsLineaConecta);
   c_sortidalog := 's_rsp';
```

```
   n_controlEstat:=1;

-- CONTROL DE PARAMETRES D'ENTRADA
   If p_idTipoLineaConecta IS NULL then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('ALTA CONNEXIO COMPTADOR EN LINEAS ');
         s_rsp := 'Falta especificar Id Tipo Linea Conectar ';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_linea_conecta;
    END IF;
   If p_dataAltaLineaConecta  IS NULL then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('ALTA CONNEXIO COMPTADOR EN LINEAS:  ');
         s_rsp := 'Falta especificar data Linea Conectar ';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_linea_conecta;
    END IF;

    If p_conectaCentralDistribucio IS NULL then
         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE('ALTA CONNEXIO COMPTADOR EN LINEAS:  ');
         s_rsp := 'Falta especificar  Id Central Distribucio a conectar  ';
         DBMS_OUTPUT.PUT_LINE(s_rsp);
        RAISE e_linea_conecta;
    END IF;
    --CONTROL DE CENTRAL DE DISTRIBUCIO SI EXISTEIX

     SELECT DISTINCT COUNT (CENTRAL.IDCENTRAL) INTO n_controlCentral
     FROM CENTRAL,CLASSE_CENTRAL
     WHERE CENTRAL.IDCENTRAL=p_conectaCentralDistribucio
     AND CLASSE_CENTRAL.IDCLASSE=2
     AND CENTRAL.ESTATCENTRAL=n_controlEstat;

    IF n_controlCentral=0 THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('ALTA CONNEXIO COMPTADOR EN LINEAS:  ');
     s_rsp := 'Falta especificar  Id Central Distribucio a conectar  ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_linea_conecta;
    END IF;

     SELECT DISTINCT CENTRAL.IDCENTRAL INTO n_conectaCentralDistribucio
     FROM CENTRAL,CLASSE_CENTRAL
     WHERE CENTRAL.IDCENTRAL=p_conectaCentralDistribucio
     AND CLASSE_CENTRAL.IDCLASSE=2
     AND CENTRAL.ESTATCENTRAL=n_controlEstat;

-- MIRO LAS LINEA DE DISTRIBUCIO QUE HI HA CONECTADES ENTRE ELLES

SELECT DISTINCT
     COUNT(LINEA_CONECTA_CENTRAL.IDCONECTA)INTO n_lineaDis_1
```

```
FROM CENTRAL, LINEA_CONECTA_CENTRAL
WHERE CENTRAL.IDCENTRAL=LINEA_CONECTA_CENTRAL.CONNECTACDISTRI
--AND CENTRAL.IDCENTRAL=p_conectaCentralDistribucio
--AND (SELECT MAX (CENTRAL.ENERGIAMAXIMA)
--FROM  CENTRAL
--WHERE CENTRAL.IDCENTRAL=p_conectaCentralDistribucio)>0
AND LINEA_CONECTA_CENTRAL.IDESTATC=1;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('LINEAS:  '||n_lineaDis_1);

 IF   n_lineaDis_1=0 THEN
  DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('ALTA CONNEXIO COMPTADOR EN LINEAS:
     La Central de distribucio no existeix: '||p_conectaCentralDistribucio);
        s_rsp := 'ALTA CONNEXIO COMPTADOR EN LINEAS:
     El comptador no te assignat un contracte o no esta
     en situacio alta com a disponible: '||p_conectaCentralDistribucio;
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_linea_conecta;
   END IF;

 SELECT CENTRAL.ENERGIAMAXIMA INTO n_centralMaxEner
 FROM CENTRAL
 WHERE CENTRAL.IDCENTRAL=p_conectaCentralDistribucio;

   --CONTROL COMPTADOR SI EXISTEIX I TE UN CONTRACTE EN VIGOR

   SELECT COUNT (COMPTADOR.IDCOMPTADOR) INTO n_registres
   FROM COMPTADOR, CONTRACTE
   WHERE COMPTADOR.IDCOMPTADOR=p_conectaComptador
   AND CONTRACTE.IDCONTADORCONTRACTE=COMPTADOR.IDCOMPTADOR
   AND CONTRACTE.IDESTATCONTRACTE=n_controlEstat;

 -- mirem el tipus de linea que vol conectar-se i que estigui en alta
   SELECT COUNT (LINEA_TIPUS.IDTLINEA)  INTO n_controlTipoLinea
   FROM LINEA_TIPUS
   WHERE LINEA_TIPUS.IDTLINEA=p_idTipoLineaConecta
   AND LINEA_TIPUS.IDESTATTLINEA=n_controlEstat;

-- MIRO SI ESTA ASSIGNAT EL COMPTADOR A LA LINEA QUE HI HA LA CENTRAL DE DISTRIBUCIO
 SELECT COUNT (*) INTO  n_conectaComptador
 FROM LINEA_CONECTAR_COMPTADOR
 WHERE  LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA=n_lineaDis_1
 AND LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA=p_conectaComptador
 AND LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA=
 LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA;

  -- si n_registres=0  el comptador no te contracte en situacio alta per
  -- connectar.
  -- si n_controlTipoLinea=0 el tipus de linea no exiteix
  -- si n_controlCentral=0 la central de distribucio no existeix o no esta
  -- disponible
```

```
  -- si n_conectaCentralDistribucio es igual a p_conectaCentralDistribucio
  -- a p_conectaCentralDistribucio conte el codi de la Central de
  -- Distribucio.
  --Llavors la connexio es viable i insertem els valors sino emet excepcio.
  -- n_conrolPar controla si el comptador te assignada la linea alternativa
  -- nomes podem haver 2 conectivitats de linea
 SELECT COUNT (LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA) INTO n_controlPar
FROM LINEA_CONECTAR_COMPTADOR
WHERE  LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA=p_conectaComptador
AND LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA=
LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA;

 If n_controlCentral=0 then
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('ALTA CONNEXIO COMPTADOR EN LINEAS:
     La Central de distribucio no existeix: '||p_conectaCentralDistribucio);
        s_rsp := 'ALTA CONNEXIO COMPTADOR EN LINEAS:
     El comptador no te assignat un contracte o no esta
     en situacio alta com a disponible: '||p_conectaCentralDistribucio;
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_linea_conecta;
   END IF;

SELECT DISTINCT CENTRAL.IDCENTRAL INTO n_conectaCentralDistribucio
FROM CENTRAL, LINEA_CONECTA_CENTRAL
WHERE LINEA_CONECTA_CENTRAL.IDCONECTA=n_lineaDis_1
AND CENTRAL.IDCENTRAL=
LINEA_CONECTA_CENTRAL.CONNECTACDISTRI;

   SELECT DISTINCT COUNT(PARLINEACOMPTADORCONECTA) INTO n_primerPart
   FROM LINEA_CONECTAR_COMPTADOR
   WHERE LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA=1
   AND LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA=p_conectaComptador;

 -- CONECTEM EL PRIMER PART SI PRIMER PART=0
If  n_conectaComptador=0  AND n_controlPar=0  then
    n_controlEstat:=1;
    n_primerPart:=1;
   -- El donem d'alta la connexio viable
INSERT INTO LINEA_CONECTAR_COMPTADOR(
         IDLINEATIPOLINEACONECTA,
         DATAALTALINEACONECTA,
         IDESTATLINEACONECTA,
         CONNECTA1_CENTRALCONECTA,
         CONNECTA2_COMPTADORCONECTA,
         PARLINEACOMPTADORCONECTA,
         OBSERVACIOLINEACONECTA)
VALUES(p_idTipoLineaConecta,
         p_dataAltaLineaConecta,
         n_controlEstat,
         n_lineaDis_1,
         p_conectaComptador,
```

```
        n_primerPart,
        p_observacionsLineaConecta);
--Gravem en la taula log----------------------------------------------------
 SELECT LINEA_TIPUS.CONSUMMAXIMTLINEA INTO n_consumToleratLinea
 FROM LINEA_TIPUS
 WHERE LINEA_TIPUS.IDTLINEA=p_idTipoLineaConecta;
 SELECT ESTAT.DESCRIPCIOESTAT INTO n_descEstat
 FROM ESTAT
 WHERE ESTAT.IDESTAT=n_controlEstat;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Connexio inserida PRIMERA connexio:
  Tipus de Linea Connectada: '||p_idTipoLineaConecta||
 --' amb una tolerancia de: '||n_consumToleratLinea||' Kw/h.
  ' en data de connexio: '||p_dataAltaLineaConecta||
   ' estat de linea: '||n_descEstat||
   ' conectat a la Central Distribucio: '||n_conectaCentralDistribucio||
   ' el comptador: '||p_conectaComptador||
   ' connexio primaria: '||n_primerPart||
   ' Observacions: '||p_observacionsLineaConecta);
    DBMS_OUTPUT.PUT_LINE('******** FI DE LES OPERACIONS ********');
         --Gravem en la taula log----------------------------------------------
  s_rsp :='Ok Connexio inserida:
  Tipus de Linea Connectada: '||p_idTipoLineaConecta||
   -- ' amb una tolerancia de: '||n_consumToleratLinea||' Kw/h.
  ' en data de connexio: '||p_dataAltaLineaConecta||
  -- ' estat de linea: '||n_descEstat||
   ' conectat a la Central Distribucio: '||n_conectaCentralDistribucio||
   ' el comptador: '||p_conectaComptador||
   ' connexio primaria: '||n_primerPart;

  pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  --DBMS_OUTPUT.PUT_LINE(s_rsp);

  ELSE

 If  n_conectaComptador>0 AND n_controlPar>0  then
    n_controlEstat:=1;
    n_segonPart:=2 ;
   -- El donem d'alta la connexio viable
INSERT INTO LINEA_CONECTAR_COMPTADOR(
        IDLINEATIPOLINEACONECTA,
        DATAALTALINEACONECTA,
        IDESTATLINEACONECTA,
        CONNECTA1_CENTRALCONECTA,
        CONNECTA2_COMPTADORCONECTA,
        PARLINEACOMPTADORCONECTA,
        OBSERVACIOLINEACONECTA)
VALUES(p_idTipoLineaConecta,
        p_dataAltaLineaConecta,
        n_controlEstat,
        n_lineaDis_1,
        p_conectaComptador,
```

```
        n_segonPart,
        p_observacionsLineaConecta);
--Gravem en la taula log----------------------------------------------------
 --11,'01/03/2012',1,16,11,'CONECTAT',sortida
 SELECT LINEA_TIPUS.CONSUMMAXIMTLINEA INTO n_consumToleratLinea
 FROM LINEA_TIPUS
 WHERE LINEA_TIPUS.IDTLINEA=p_idTipoLineaConecta;
 SELECT ESTAT.DESCRIPCIOESTAT INTO n_descEstat
 FROM ESTAT
 WHERE ESTAT.IDESTAT=n_controlEstat;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Connexio inserida SEGONA connexio:
   Tipus de Linea Connectada: '||p_idTipoLineaConecta||
 --' amb una tolerancia de: '||n_consumToleratLinea||' Kw/h.
 ' en data de connexio: '||p_dataAltaLineaConecta||

   ' estat de linea: '||n_descEstat||

   ' conectat a la Central Distribucio: '||n_conectaCentralDistribucio||
   ' el comptador: '||p_conectaComptador||
   ' connexio secundaria: '||n_segonPart||
   ' Observacions: '||p_observacionsLineaConecta);

   DBMS_OUTPUT.PUT_LINE('******** FI DE LES OPERACIONS ********');
        --Gravem en la taula log----------------------------------------------------
  s_rsp :='Ok Connexio inserida:
  Tipus de Linea Connectada: '||p_idTipoLineaConecta||
  ---' amb una tolerancia de: '||n_consumToleratLinea||' Kw/h.
   'en data de connexio: '||p_dataAltaLineaConecta||
   ' estat de linea: '||n_descEstat||
   ' conectat a la Central Distribucio: '||n_conectaCentralDistribucio||
   ' el comptador: '||p_conectaComptador||
   ' connexio secundaria: '||n_segonPart;
   --' Observacions: '||p_observacionsLineaConecta;
        pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
        --DBMS_OUTPUT.PUT_LINE(s_rsp);
  -- ELSE


   END IF;
   DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('ALTA CONNEXIO COMPTADOR EN LINEAS: ');
      s_rsp := ' Linea te el comptador te les dues connexios maximes
      autoritzades el nombre de par es de: '||n_segonPart|| ' pel comptador: '||p_conectaComptador ;
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_linea_conecta;
END IF;

COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
```

```
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_CONECTA_LINEA_COMPTADOR;
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.

--- MODIFICAR CONEXIONS DE COMPTADOR

*****************************************************************************/
PROCEDURE PRC_MODIFICA_CONECTA_COMPTADOR(
   p_idLineaConecta in LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%TYPE,
   p_idTipoLineaConecta in LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA%Type,
   p_dataModificaLineaConecta in LINEA_CONECTAR_COMPTADOR.DATAMODIFICACIOLINEACONECTA%Type,
   p_estatLineaConecta in LINEA_CONECTAR_COMPTADOR.IDESTATLINEACONECTA%Type,
   p_conectaCentralDistribucio in LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA%Type,
   p_conectaComptador in LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA%Type,
   p_parConecta in LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA%TYPE,
   p_observacionsLineaConecta in LINEA_CONECTAR_COMPTADOR.OBSERVACIOLINEACONECTA%Type,
   s_rsp out NOCOPY VARCHAR)
AS

BEGIN
   c_procesLog := 'PRC_MODIFICA_CONECTA_COMPTADOR';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Id Linea: '||p_idLineaConecta||
   ' Tipus Linea: ' || p_idTipoLineaConecta||
   ', Data Modificacio connexio Linea: ' ||p_dataModificaLineaConecta ||
   ', Linea Central Distribucio: ' ||p_conectaCentralDistribucio||
   ', Comptador: ' ||p_conectaComptador||
   ', Estat de la Linea: ' ||p_estatLineaConecta||
   ', Observacions  Linea: ' ||p_observacionsLineaConecta;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('     MODIFICAR CONNEXIO COMPTADOR A LA  LINEA   ');
    DBMS_OUTPUT.PUT_LINE('---------------------------------------------- ');
    DBMS_OUTPUT.PUT_LINE('
    Id Linea: '||p_idLineaConecta||
    ', Tipus Linea: ' || p_idTipoLineaConecta||
   ', Data Modificacio connexio Linea: ' ||p_dataModificaLineaConecta ||
   ', Linea Central Distribucio: ' ||p_conectaCentralDistribucio||
   ', Comptador: ' ||p_conectaComptador||
   ', Estat de la Linea: ' ||p_estatLineaConecta||
```

```
', Observacions  Linea: ' ||p_observacionsLineaConecta);
c_sortidalog := 's_rsp';

-- CONTROL DE DADES
IF p_idLineaConecta IS NULL THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA: ');
      s_rsp := 'Falta especificar Id Tipo Linea Conectar ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_linea_conecta;
 END IF;

If (p_idTipoLineaConecta IS NULL) then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA: ');
      s_rsp := 'Falta especificar Id Tipo Linea Conectar ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_linea_conecta;
 END IF;
If p_dataModificaLineaConecta  IS NULL then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA:  ');
      s_rsp := 'Falta especificar data Modificacio Linea Conectar ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_linea_conecta;
 END IF;
  If (p_estatLineaConecta IS NULL) then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA:  ');
      s_rsp := 'Falta especificar el Estat 1=ALTA o 0=BAIXA ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_linea_conecta;
 END IF;
 If (p_estatLineaConecta<0 OR p_estatLineaConecta>3
 OR p_estatLineaConecta=0) then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA:  ');
      s_rsp := 'Falta especificar el Estat 1=ALTA o 2=BAIXA ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_linea_conecta;
 END IF;
 If p_conectaCentralDistribucio IS NULL then
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA:  ');
      s_rsp := 'Falta especificar  ID Linea Central Distribucio  ';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_linea_conecta;
 END IF;
 --CONTROL LINEA DE CENTRAL DE DISTRIBUCIO SI EXISTEIX
 n_controlCentral:=0;
 SELECT DISTINCT LINEA_CONECTA_CENTRAL.IDCONECTA INTO n_controlCentral
FROM LINEA_CONECTA_CENTRAL,CENTRAL
```

```
   WHERE LINEA_CONECTA_CENTRAL.CONNECTACDISTRI=p_conectaCentralDistribucio
   AND CENTRAL.IDCENTRAL=p_conectaCentralDistribucio;


   -- control que el contador existeix en la linea, conectat
    If n_controlCentral=0 OR n_controlCentral IS NULL THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA:  '
     ||p_conectaCentralDistribucio||
     ' La linea de la Central Distribucio no existeix  ');
     s_rsp := ' La linea de la Central Distribucio no existeix  ';
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_linea_conecta;
    END IF;

    --CONTROL COMPTADOR SI EXISTEIX
    SELECT COMPTADOR.IDCOMPTADOR INTO n_idconectaComptador
    FROM COMPTADOR
    WHERE COMPTADOR.IDCOMPTADOR=p_conectaComptador;

If (n_idconectaComptador=0) then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA:
   no existeix '||p_conectaComptador);
   s_rsp := 'Id Comptador no existeix  ';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_linea_conecta;
END IF;
    -- recuperem el codi  Linea
   SELECT LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA INTO n_idLineaConecta
   FROM LINEA_CONECTAR_COMPTADOR
   WHERE LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=p_idLineaConecta;

 -- CONTROL SI EXISTEIX LA LINEA

 If n_registres=0 then
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA: ');
   s_rsp := 'Tipus Linea no existent a la BBDD';
   DBMS_OUTPUT.PUT_LINE(s_rsp);
   RAISE e_linea_conecta;
END IF;

s_sql := 'UPDATE LINEA_CONECTAR_COMPTADOR SET ';
-- Construirem la sentencia UPDATE segons si hi ha valor en els parÃ metres-
IF p_idTipoLineaConecta IS NOT NULL THEN
-- Modifiquem id Tipo LineaConecta
  s_sql := s_sql || 'IDLINEACONECTA=''' ||p_idTipoLineaConecta|| ''',';
 END IF;
IF p_dataModificaLineaConecta IS NOT NULL THEN
 -- Modifiquem data Modifica Linea Conecta
 s_sql := s_sql || 'DATAMODIFICACIOLINEACONECTA=''' ||p_dataModificaLineaConecta|| ''',';
```

```
END IF;
IF p_estatLineaConecta IS NOT NULL THEN
-- Modifiquem estat Linea Conecta
  s_sql := s_sql || 'IDESTATLINEACONECTA=''' ||p_estatLineaConecta|| ''',';
END IF;
IF p_conectaCentralDistribucio IS NOT NULL THEN
-- Modifiquem conecta Central Distribucio
s_sql := s_sql || 'CONNECTA1_CENTRALCONECTA=''' ||n_controlCentral|| ''',';
END IF;
IF p_conectaComptador IS NOT NULL THEN
-- Modifiquem conecta Central Distribucio
 s_sql := s_sql || 'CONNECTA2_COMPTADORCONECTA=''' ||p_conectaComptador|| ''',';
END IF;
IF p_parConecta IS NOT NULL THEN
 -- Modifiquem conecta par del comptador
  s_sql := s_sql || 'PARLINEACOMPTADORCONECTA=''' ||p_parConecta|| ''',';
END IF;
IF p_observacionsLineaConecta IS NOT NULL THEN
 -- Modifiquem conecta observacions
  s_sql := s_sql || 'OBSERVACIOLINEACONECTA=''' ||p_observacionsLineaConecta|| ''',';
END IF;
-- Eliminem la coma final de la sentencia SQL-------------------------
  s_sql := SUBSTR (s_sql, 1, LENGTH (s_sql) - 1);
-- Afegim a la sentencia la condiciÃƒÂ³ del WHERE----------------------
  s_sql := s_sql || ' WHERE IDLINEACONECTA=''' ||n_idLineaConecta|| '''';
 EXECUTE IMMEDIATE s_sql;
  IF SQL%ROWCOUNT = 0 THEN
  --L'error serÃ  que Tipo Linea ja el tenim en la taula i no admet duplicats--
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICA_CONECTA_COMPTADOR:');
    s_rsp := 'Actualitzacio no realitzada';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_linea_conecta;
ELSE

SELECT ESTAT.DESCRIPCIOESTAT INTO n_descEstat
FROM ESTAT
WHERE ESTAT.IDESTAT=p_estatLineaConecta;

 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('MODIFICAR CONNEXIO COMPTADOR A LA  LINEA:
 Modificat satisfactoriament en el Codi de Linea: '||p_idLineaConecta||
 '  data modificacio: '||p_dataModificaLineaConecta||
 ' estat linea: '||n_descEstat||
 ' central distribucio: '||n_controlCentral||
 ' comptador :'||p_conectaComptador||
 ' par modificat: ' ||p_parConecta||
 ' observacions: '||p_observacionsLineaConecta);
 DBMS_OUTPUT.put_line ('******** FI DE LES OPERACIONS ******** ');
 s_rsp := 'OK Modificat satisfactoriament en el Codi de Linea: '||p_idLineaConecta||
 '  data modificacio: '||p_dataModificaLineaConecta||
 ' estat linea: '||n_descEstat||
```

```
' central distribucio: '||n_controlCentral||
' comptador :'||p_conectaComptador||
' par modificat: ' ||p_parConecta||
' observacions: '||p_observacionsLineaConecta;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
          END IF;
    COMMIT;
 EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    s_rsp := 'Error:  Linea duplicada. No es poden fer les modificacions';
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      ----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('MODIFICA_CONECTA_COMPTADOR: ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
ROLLBACK;

END PRC_MODIFICA_CONECTA_COMPTADOR;

/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.

CONSULTA D'UNA LINEA

*****************************************************************************/

PROCEDURE PRC_CONSULTAR_CONEXIO_LINEA(
  p_idLineaConecta in LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%TYPE,
  s_rsp out NOCOPY VARCHAR2)
AS

CURSOR C_CONNEXIO_LINEA IS
SELECT LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA,
LINEA_CONECTAR_COMPTADOR.DATAALTALINEACONECTA,
ESTAT.DESCRIPCIOESTAT,
LINEA_CONECTAR_COMPTADOR.DATAMODIFICACIOLINEACONECTA,
LINEA_TIPUS.CONSUMMAXIMTLINEA,
```

```
CENTRAL.IDCENTRAL,
LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA,
LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA,
LINEA_CONECTAR_COMPTADOR.OBSERVACIOLINEACONECTA
FROM LINEA_CONECTAR_COMPTADOR,ESTAT,LINEA_CONECTA_CENTRAL,LINEA_TIPUS,CENTRAL
WHERE
LINEA_CONECTA_CENTRAL.IDCONECTA=LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA
AND LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=p_idLineaConecta
AND  LINEA_CONECTAR_COMPTADOR.IDESTATLINEACONECTA=ESTAT.IDESTAT
AND  LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA=LINEA_TIPUS.IDTLINEA
AND LINEA_CONECTA_CENTRAL.CONNECTACDISTRI=CENTRAL.IDCENTRAL;

BEGIN
   c_procesLog := 'PRC_CONSULTAR_CONEXIO_CENTRAL';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Codi Linea: ' ||p_idLineaConecta;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('    CONSULTA UNA LINEA CONNECTADA      ');
   DBMS_OUTPUT.PUT_LINE('----------------------------------------');
    DBMS_OUTPUT.PUT_LINE('CONSULTAR_CONEXIO_CENTRAL:
    Codi Linea: ' ||p_idLineaConecta);
   c_sortidalog := 's_rsp';
  IF  p_idLineaConecta is null THEN
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('CONSULTA UNA LINEA CONNECTADA: ');
        s_rsp := 'Falta especificar Codi Linea ';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_linea_conecta;
    END IF;
-- miro si la linea existeix
SELECT COUNT(*) INTO n_registres
FROM LINEA_CONECTAR_COMPTADOR
WHERE LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=p_idLineaConecta;

IF n_registres=0 THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('CONSULTA UNA LINEA CONNECTADA: ');
 s_rsp := 'La linea no existeix. ';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_linea_conecta;
END IF;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(' ');
 DBMS_OUTPUT.PUT_LINE('    CONSULTA UNA LINEA CONNECTADA      ');
 DBMS_OUTPUT.PUT_LINE('----------------------------------------');
OPEN C_CONNEXIO_LINEA;
FETCH C_CONNEXIO_LINEA
INTO n_idLineaConecta,
n_dataAltaLinea,
n_descEstat,
n_dataModificacioLinea,
```

```
n_centralMaxEner,
n_conectaCentralDistribucio,
n_conectaComptador,
n_parConecta,
n_observacions;
  WHILE C_CONNEXIO_LINEA%FOUND LOOP
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(
 '   Codi Linea: '
 ||n_idLineaConecta||
 ' Data Alta: '
 ||n_dataAltaLinea||
 ' Estat: '
 ||n_descEstat||
 ' Data Modificacio: '
 ||n_dataModificacioLinea||
 ' Energia Maxima torelable: '
 ||n_centralMaxEner||
' Central Distribucio: '
 ||n_conectaCentralDistribucio||
 ' Comptador: '
 ||n_conectaComptador||
 'Par de connexio: '
 ||n_parConecta||
 ' Observacions: '
 ||n_observacions);
  s_rsp := 'OK  CONSULTA UNA LINEA CONNECTADA
     Codi Linea: '
 ||n_idLineaConecta||
 ' Data Alta: '
 ||n_dataAltaLinea||
 ' Estat: '
 ||n_descEstat||
 ' Data Modificacio: '
 ||n_dataModificacioLinea||
 ' Energia Maxima torelable: '
 ||n_centralMaxEner||
' Central Distribucio: '
 ||n_conectaCentralDistribucio||
 ' Comptador: '
 ||n_conectaComptador||
 'Par de connexio: '
 ||n_parConecta||
 ' Observacions: '
 ||n_observacions;
FETCH C_CONNEXIO_LINEA
INTO n_idLineaConecta,
n_dataAltaLinea,
n_descEstat,
n_dataModificacioLinea,
n_centralMaxEner,
n_conectaCentralDistribucio,
```

```
n_conectaComptador,
n_parConecta,
n_observacions;

pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total connexions Linea: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_CONNEXIO_LINEA;
COMMIT;
  EXCEPTION

    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
  -- L'error no ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
   -- L'error si ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || s_rsp;
      END IF;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('  ');
        DBMS_OUTPUT.put_line (s_rsp);
        pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
ROLLBACK;
END PRC_CONSULTAR_CONEXIO_LINEA;

/*******************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.

CONSULTA LINEAS CONECTADES I NO CONECTADES

*******************************************************************************/

PROCEDURE PRC_CONSULTAR_LINEAS(
  s_rsp out NOCOPY VARCHAR2)
AS

CURSOR C_CONNEXIO_LINEA IS
SELECT LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA,
LINEA_CONECTAR_COMPTADOR.DATAALTALINEACONECTA,
ESTAT.DESCRIPCIOESTAT,
LINEA_CONECTAR_COMPTADOR.DATAMODIFICACIOLINEACONECTA,
LINEA_TIPUS.CONSUMMAXIMTLINEA,
CENTRAL.IDCENTRAL,
LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA,
LINEA_CONECTAR_COMPTADOR.PARLINEACOMPTADORCONECTA,
LINEA_CONECTAR_COMPTADOR.OBSERVACIOLINEACONECTA
FROM LINEA_CONECTAR_COMPTADOR,ESTAT,LINEA_CONECTA_CENTRAL,LINEA_TIPUS,CENTRAL
```

```
WHERE
LINEA_CONECTA_CENTRAL.IDCONECTA=LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA
AND LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA
AND  LINEA_CONECTAR_COMPTADOR.IDESTATLINEACONECTA=ESTAT.IDESTAT
AND  LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA=LINEA_TIPUS.IDTLINEA
AND LINEA_CONECTA_CENTRAL.CONNECTACDISTRI=CENTRAL.IDCENTRAL
ORDER BY IDLINEACONECTA;
BEGIN
   c_procesLog := 'PRC_CONSULTAR_LINEAS';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' CONSULTA LINEAS ';
      c_sortidalog := 's_rsp';

-- miro si hi ha lineas
SELECT COUNT(*) INTO n_registres
FROM LINEA_CONECTAR_COMPTADOR
WHERE LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA;

IF n_registres=0 THEN
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('CONSULTA LINEAS ');
 s_rsp := 'No hi han lineas connectadas. ';
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_linea_conecta;
END IF;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(' ');
 DBMS_OUTPUT.PUT_LINE('          CONSULTA LINEAS           ');
 DBMS_OUTPUT.PUT_LINE('----------------------------------------');
OPEN C_CONNEXIO_LINEA;
FETCH C_CONNEXIO_LINEA
INTO n_idLineaConecta,
n_dataAltaLinea,
n_descEstat,
n_dataModificacioLinea,
n_centralMaxEner,
n_conectaCentralDistribucio,
n_conectaComptador,
n_parConecta,
n_observacions;
  WHILE C_CONNEXIO_LINEA%FOUND LOOP
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(
 '   Codi Linea: '
 ||n_idLineaConecta||
 ' Data Alta: '
 ||n_dataAltaLinea||
 ' Estat: '
 ||n_descEstat||
 ' Data Modificacio: '
 ||n_dataModificacioLinea||
 ' Energia Maxima torelable: '
```

```
 ||n_centralMaxEner||
' Central Distribucio: '
 ||n_conectaCentralDistribucio||
 ' Comptador: '
 ||n_conectaComptador||
 ' Par de connexio: '
 ||n_parConecta||
 ' Observacions: '
 ||n_observacions);
 DBMS_OUTPUT.PUT_LINE('   ');
  s_rsp := 'OK   CONSULTA LINEAS
    Codi Linea: '
 ||n_idLineaConecta||
 ' Data Alta: '
 ||n_dataAltaLinea||
 ' Estat: '
 ||n_descEstat||
 ' Data Modificacio: '
 ||n_dataModificacioLinea||
 ' Energia Maxima torelable: '
 ||n_centralMaxEner||
' Central Distribucio: '
 ||n_conectaCentralDistribucio||
 ' Comptador: '
 ||n_conectaComptador||
 ' Par de connexio: '
 ||n_parConecta;
 --' Observacions: '
-- ||n_observacions;
FETCH C_CONNEXIO_LINEA
INTO n_idLineaConecta,
n_dataAltaLinea,
n_descEstat,
n_dataModificacioLinea,
n_centralMaxEner,
n_conectaCentralDistribucio,
n_conectaComptador,
n_parConecta,
n_observacions;

pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total Lineas: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  DBMS_OUTPUT.PUT_LINE('   ');
CLOSE C_CONNEXIO_LINEA;
COMMIT;
  EXCEPTION

    WHEN OTHERS THEN
```

```
    IF s_rsp IS NULL THEN
 -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
  -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('   ');
      DBMS_OUTPUT.put_line (s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
ROLLBACK;
END PRC_CONSULTAR_LINEAS;
END "GESTION_LINEA_CONECTAR";
```

11.13 TRATAMIENTO DE GESTION DE LECTURAS DE LOS CONTADORES.

Este procedimeinto recibe datos del procedimiento alta de contratos, cunado se ejecuta una alta de un contrato el procedimiento PRC_INICIA_LECTURA efectua un apune en la tabla de LECTURAS con el numero de contador la fecha y estado podinedo los campos de consumo, lectura actual y anterior a cero.

Cuando se ejecuta este procedimeinto para entrar lecturas de contadores le procedimiento PRC_LEER_LECTURAS automáticamente actualiza la tabla HIST_LECTURA con el año, el mes, la línea en que sea producido la lectura, el contador y el consumo efectuado.

DEFINICION	PROCEDIMIENTO
Alta de la lectura del contador en el mes, actualiza el Historico las lecturas realizadas.	PRC_LEER_LECTURA
Inicializa los valores de consumo de lescturas a cero cuando se da el alta de un Contrato	PRC_INICIA_LECTURA
Consulta de las Lecturas realizadas.	PRC_CONSULTAR_LECTURES

```
CREATE OR REPLACE
PACKAGE        "GESTION_LECTURA" AS
PROCEDURE PRC_LEER_LECTURA(
      p_idComptadorLectura IN LECTURA.IDCOMPTADORLECTURA%TYPE,
      p_confirmaLectura IN LECTURA.CONFIRMALECTURA%TYPE,
      p_dataLectura IN LECTURA.DATALECTURA%TYPE,
      p_idTipoLectura IN LECTURA.IDTIPOLECTURA%TYPE,
      p_lecturaActual IN LECTURA.LECTURAACTUAL%TYPE,
      s_rsp out NOCOPY VARCHAR);

PROCEDURE PRC_INICIA_LECTURA(
      lec_idComptadorLectura IN LECTURA.IDCOMPTADORLECTURA%TYPE,
      --lec_confirmaLectura in LECTURA.CONFIRMALECTURA%TYPE,
      lec_dataLectura IN LECTURA.DATALECTURA%TYPE,
      --lec_idTipoLectura IN LECTURA.IDTIPOLECTURA%TYPE,
      --lec_consumLectura IN LECTURA.CONSUMLECTURA%TYPE,
      --lec_lecturaActual IN LECTURA.LECTURAACTUAL%TYPE,
      --lec_lecturaAnterior IN LECTURA.LECTURAANTERIOR%TYPE,
      s_rsp out NOCOPY VARCHAR);

PROCEDURE PRC_CONSULTAR_LECTURES(
      s_rsp out NOCOPY VARCHAR);
END "GESTION_LECTURA";
```

11.13.1 PROCEDIMIENTO SPL PACKAGE LEER LECTURAS DE LOS CONTADORES.

```
CREATE OR REPLACE PACKAGE BODY        "GESTION_LECTURA" AS
  n_idComptadorLectura  number;
  n_lecturaAnterior  number;
  n_idTipoLectura    number;
  n_idLectura                  number;
  n_dataLectura        char(10);
  n_confirmaLectura     number;
  n_valorLecturaAnterior        NUMBER;
  n_consumLectura number;
  n_idContracte                CONTRACTE.IDCONTRACTE%TYPE;
  n_idLinea LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%TYPE;
  n_descTipusLectura TIPO_LECTURA.DESCRIPCIOLECTURA%TYPE;
  n_idLineaHistLectura LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%TYPE;
  n_lecturaActual NUMBER;

  g_idComptadorHist    NUMBER;
  g_idLineaHistLectura NUMBER;
  g_anyHistLectura     VARCHAR2(4);
  g_mesHistLectura     VARCHAR2(2);
  g_totalConsumHistLectura  NUMBER;

CURSOR C_LLEGIR_LINEA IS
```

```
SELECT LINEA_CONECTA_CENTRAL.CONNECTACDISTRI,
LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA,
CENTRAL.ENERGIAMAXIMA,
LINEA_TIPUS.CONSUMMAXIMTLINEA
FROM LINEA_CONECTA_CENTRAL,LINEA_CONECTAR_COMPTADOR,LINEA_TIPUS,CENTRAL
WHERE CENTRAL.IDCENTRAL=LINEA_CONECTA_CENTRAL.CONNECTACDISTRI
AND LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA=LINEA_TIPUS.IDTLINEA;

n_centralDistribucio LINEA_CONECTA_CENTRAL.CONNECTACDISTRI%TYPE;
n_lineaConectada LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%TYPE;
n_energiaMaximaCentral CENTRAL.ENERGIAMAXIMA%TYPE;
n_energiaMaximaLinea LINEA_TIPUS.CONSUMMAXIMTLINEA%TYPE;



 c_procesLog                LOG_TFC.procesLog%TYPE;
 c_dataHoraLog     LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog      LOG_TFC.entradaLog%TYPE;
 c_sortidaLog      LOG_TFC.sortidaLog%TYPE;

 s_rsp                      LOG_TFC.RSPLOG%TYPE;
 n_registres     NUMBER;
 s_sql         VARCHAR2 (2000);
 n_NUM_ERR          NUMBER(10);
 sortida VARCHAR2(2000):='';
 e_lectura   EXCEPTION;
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
            PROCEDIMENT LECTURA DE COMPTADORS.
DESCRIPCIO:
Les dades es generen cridant el procediment
GESTION_LECTURA.PRC_LEER_LECTURA(<id comptador>,<confirmacio de lectura>,
<data de la letura>,<id tipus de lectura>,
<valor de la lectura feta en el comptador>,sortida);
 On l el tipus de lectura 1=Presencial 2=Telematica.
 El procediment esta automatizat i carega dades calculades dels consums
 del mes i any i la linea per la que esta conectat el comptador.
 *****************************************************************************/
PROCEDURE PRC_LEER_LECTURA(
      p_idComptadorLectura IN LECTURA.IDCOMPTADORLECTURA%TYPE,
      p_confirmaLectura IN LECTURA.CONFIRMALECTURA%TYPE,
      p_dataLectura IN LECTURA.DATALECTURA%TYPE,
      p_idTipoLectura IN LECTURA.IDTIPOLECTURA%TYPE,
      p_lecturaActual IN LECTURA.LECTURAACTUAL%TYPE,
      s_rsp out NOCOPY VARCHAR)
AS

BEGIN

   c_procesLog := 'PRC_LEER_LECTURAS';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Id Comptador: ' ||p_idComptadorLectura||
```

```
            ', Confirma: ' ||p_confirmaLectura ||
            ', Data lectura: ' ||p_dataLectura||
            ', Tipus de Lectura: ' ||p_idTipoLectura||
            ', Lectura Comptador: ' ||p_lecturaActual;

    g_idComptadorHist:=p_idComptadorLectura;
    --g_idLineaHistLectura:=n_idLineaHistLectura;
    g_anyHistLectura:= SUBSTR (p_dataLectura, 7, 4);
    g_mesHistLectura:= SUBSTR (p_dataLectura, 4, 2);
    g_totalConsumHistLectura:=n_consumLectura;
      --g_totalConsumHistLectura:=n_consumLectura;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('     LLEGIR LECTURA COMPTADOR      ');
  DBMS_OUTPUT.PUT_LINE('-------------------------------------');
  DBMS_OUTPUT.PUT_LINE(': Id Comptador: '
            ||p_idComptadorLectura||
            ', Confirma: ' ||p_confirmaLectura ||
            ', Data Lectura: ' ||p_dataLectura||
            ', Tipus de Lectura: ' ||p_idTipoLectura||
            ', Lectura Comptador: ' ||p_lecturaActual);
  c_sortidalog := 's_rsp';


   -- cerco la linea que conecta
   SELECT LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA into n_idLineaHistLectura
   FROM LINEA_CONECTAR_COMPTADOR
   WHERE LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA=p_idComptadorLectura;

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('LLEGIR LECTURA COMPTADOR: Linea conectada: '
   ||n_idLineaHistLectura);
   g_idLineaHistLectura:=n_idLineaHistLectura;

   IF n_idLineaHistLectura<>n_idLineaHistLectura THEN
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('LLEGIR LECTURA COMPTADOR: Linea conectada
     no existeix : '
     ||n_idContracte);
     s_rsp := 'Comptador no te Linea '||p_idComptadorLectura;
     DBMS_OUTPUT.PUT_LINE(s_rsp);
     RAISE e_lectura;
     ELSE

   g_idLineaHistLectura:=n_idLineaHistLectura;

    END IF;

  --COMPROVACIO DEL COMPTADOR SI EXISTEIX EN CONTRACTE I ESTA ALTA
OPEN C_LLEGIR_LINEA;
FETCH C_LLEGIR_LINEA
INTO
```

```
n_centralDistribucio,
n_lineaConectada,
n_energiaMaximaCentral,
n_energiaMaximaLinea;

g_idLineaHistLectura:=n_lineaConectada;

    SELECT COUNT(*) INTO n_idContracte
          FROM CONTRACTE
          WHERE CONTRACTE.IDCONTADORCONTRACTE=p_idComptadorLectura
          AND CONTRACTE.IDESTATCONTRACTE=1;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('LLEGIR LECTURA COMPTADOR: Contracte: '
    ||n_idContracte);

   IF n_idContracte=0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('Comptador no te contracte: '||p_idComptadorLectura);
    s_rsp := 'Comptador no te contracte '||p_idComptadorLectura;
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_lectura;
   ELSE
         g_idComptadorHist:=p_idComptadorLectura;
   END IF;

    -- COMPROVACIO DEL TIPO DE LECTURA
    SELECT TIPO_LECTURA.IDTIPOLECTURA INTO n_idTipoLectura
    FROM TIPO_LECTURA
    WHERE TIPO_LECTURA.IDTIPOLECTURA=p_idTipoLectura;

    IF n_idTipoLectura IS NOT NULL  THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('Comptador te el tipus de lectura: '||n_idTipoLectura);
    --- EL VALOR DEL TIPUS DE LINEA LECTURA ESTA A t_lectura

    ELSE

    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('Comptador no te tipus de lectura: '||p_idComptadorLectura);
    s_rsp := 'Comptador no te tipus de lectura: '||p_idComptadorLectura;
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_lectura;
    END IF;

       -- CERQUEM LA LECTURA ANTERIOR FETA

    SELECT COUNT (DISTINCT LECTURA.LECTURAACTUAL)INTO n_lecturaAnterior
    FROM LECTURA
    WHERE LECTURA.IDCOMPTADORLECTURA=p_idComptadorLectura
    AND LECTURA.DATALECTURA<p_dataLectura;
    --AND  ROWNUM=1;
    --n_LecturaAnterior te la lectura anterior
```

```
      n_valorLecturaAnterior:=n_lecturaAnterior;

   IF n_lecturaAnterior=0 THEN
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE('Ultima lectura feta  del comptador: '||p_idComptadorLectura||
      '  no existeix es nul');
      s_rsp := 'Ultima lectura feta  del comptador: '||p_idComptadorLectura||
      '  no existeix es nul';
      DBMS_OUTPUT.PUT_LINE(s_rsp);
      RAISE e_lectura;
   END IF;

   -- CALCULEM EL CONSUM
   IF   (p_lecturaActual-n_valorLecturaAnterior)<0 THEN
     n_consumLectura:=n_valorLecturaAnterior-p_lecturaActual;
     g_totalConsumHistLectura:=n_consumLectura;
     ELSE
     n_consumLectura:=p_lecturaActual-n_valorLecturaAnterior;
     g_totalConsumHistLectura:=n_consumLectura;
   END IF;
   DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE('El consum de comptador: '||p_idComptadorLectura||
       ' en la data: '||p_dataLectura||' es de: '||n_consumLectura||' kw/h.');

     -- COMPROVEM DE QUE SI ES UNA LECTURA ADONAR D'ALTA
 /*************************************************************
    SELECT COUNT(*)INTO n_registres
      FROM LECTURA
      WHERE LECTURA.IDCOMPTADORLECTURA=p_idComptadorLectura
      AND LECTURA.DATALECTURA<p_dataLectura
      AND LECTURA.IDTIPOLECTURA=p_idTipoLectura
      AND LECTURA.LECTURAANTERIOR>n_lecturaAnterior;
   *************************************************************/

     SELECT DISTINCT COUNT(*) INTO n_registres
      FROM LECTURA
      WHERE LECTURA.IDCOMPTADORLECTURA=p_idComptadorLectura
      AND LECTURA.DATALECTURA=p_dataLectura
      AND LECTURA.IDTIPOLECTURA=p_idTipoLectura
      AND LECTURA.LECTURAACTUAL<>n_lecturaAnterior;

      IF n_registres=0 THEN

  /**********************
   -- WHILE n_registres=0 LOOP
   IF n_registres=0 THEN
   GESTION_HIST_CONSUMS.PRC_ACTUALIZAR_HIS_CONSUM(g_idComptadorHist,
g_idLineaHistLectura,g_anyHistLectura,g_mesHistLectura,
g_totalConsumHistLectura,s_rsp);
***********************************/
   -- isertem dades de la lectura
     INSERT INTO LECTURA(IDCOMPTADORLECTURA,
```

```
                    CONFIRMALECTURA,
                    DATALECTURA,
                    IDTIPOLECTURA,
                    CONSUMLECTURA,
                    LECTURAACTUAL,
                    LECTURAANTERIOR)
          VALUES(p_idComptadorLectura,
              p_confirmaLectura,
              p_dataLectura,
              p_idTipoLectura,
              n_consumLectura,
              p_lecturaActual,
              n_valorLecturaAnterior);

--  ACTUALITZA LA TAULA DE HISTORICS HIST_LECTURA

  GESTION_HIST_CONSUMS.PRC_ACTUALIZAR_HIS_CONSUM(
  g_idComptadorHist,
  g_idLineaHistLectura,
  g_anyHistLectura,
  g_mesHistLectura,
  g_totalConsumHistLectura,s_rsp);

--Gravem en la taula log----------------------------------------------------
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('El consum de comptador: '
  ||p_idComptadorLectura||
  ' en la data: '
  ||p_dataLectura||
  ' es de: '||n_consumLectura||' kw/h.');
  DBMS_OUTPUT.PUT_LINE('********** FI DE LES OPERACIONS ********** ');
  s_rsp :='Ok: insertada El consum de comptador: '
  ||p_idComptadorLectura||
  ' en la data: '
  ||p_dataLectura||
  ' es de: '||n_consumLectura||' kw/h.';
  pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 ELSE
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('LLEGIR LECTURA: duplicat ');
  s_rsp := 'Lectura ja feta pel comptador '||p_idComptadorLectura;
  DBMS_OUTPUT.PUT_LINE(s_rsp);
  RAISE e_lectura;
 END IF;
CLOSE C_LLEGIR_LINEA;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi
         s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
      ELSE
```

```
        -- L'error si ha estat controlat per codi
        s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 ROLLBACK;

END PRC_LEER_LECTURA;
/*****************************************************************************
Autor: Eduard Monzonis Hierro                    UOC
18/05/2012                          TFC: CONTROL ENERGIA.
        PROCEDIMENT INICIA LECTURES PER CONTRACTE D'ALTA FET.
DESCRIPCIO:
Aques procediment inicialitza els valor de lectura actual, lectura anterior
i consum.
Es un procediment automatizat que es cridat pel procediment de altes de
CONTRACTE de CLIENT.
*****************************************************************************/
PROCEDURE PRC_INICIA_LECTURA(
      lec_idComptadorLectura IN LECTURA.IDCOMPTADORLECTURA%TYPE,
      lec_dataLectura IN LECTURA.DATALECTURA%TYPE,
      s_rsp out NOCOPY VARCHAR)
AS

BEGIN
c_procesLog := 'PRC_INICIA_LECTURA';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Id Comptador Alta en CONTRACTE: ' ||lec_idComptadorLectura||
    ', Data  Alta Lectura Inicial: '||lec_dataLectura;
      -- passo valors a les variables
       --lec_confirmaLectura:=p_confirmaLectura;
      -- n_idComptadorLectura:=lec_idComptadorLectura;
       --n_confirmaLectura :=1;
       --n_dataLectura:=lec_dataLectura;
       --n_idTipoLectura:=lec_confirmaLectura;
       --n_consumLectura:=0;
       --n_lecturaActual:=0;
       --n_lecturaAnterior:=0;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('INICIAIITZATS LECTURAS COMPTADOR ALTA EN CONTRACTE FETA:
    Codi Comptador Alta en CONTRACTE: '||lec_idComptadorLectura||
     ', Data  Alta en CONTRACTE: '||lec_dataLectura);
   c_sortidalog := 's_rsp';

  IF  lec_idComptadorLectura<>0 THEN
-- POSA ELS CONSUMS DELS COMPTADORS A ZERO QUAN ES DONA ALTA D'UN CONTRACTE
              n_confirmaLectura:=1;
              n_idTipoLectura:=1;
```

```
            n_consumLectura:=0;
            n_lecturaActual:=0;
            n_lecturaAnterior:=0;
  --INSERT INICIALITZA VALORS DEL COMPTADOR DONAT ALTA A CONTRACTE
      INSERT INTO LECTURA(IDCOMPTADORLECTURA,
                  CONFIRMALECTURA,
                  DATALECTURA,
                  IDTIPOLECTURA,
                  CONSUMLECTURA,
                  LECTURAACTUAL,
                  LECTURAANTERIOR)
        VALUES(lec_idComptadorLectura,
            n_confirmaLectura,
            lec_dataLectura,
            n_idTipoLectura,
            n_consumLectura,
            n_lecturaActual,
            n_lecturaAnterior);
        --Gravem en la taula log----------------------------------------------------
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.put_line ('OK: INICIAlITZATS LECTURAS COMPTADOR ALTA EN CONTRACTE FETA:
        Id Comptador Alta en CONTRACTE: ' ||lec_idComptadorLectura||
            ', Confirma Alta en CONTRACTE: ' ||n_confirmaLectura||
            ', Data  Alta en CONTRACTE: ' ||n_dataLectura||
            ', Tipus de Lectura: ' ||n_idTipoLectura||
            ', Lectura inicialitzada: ' ||n_consumLectura||
     ', Consums inicialitzat:'||n_consumLectura||
     ', Lectura Anteior inicialitzada: '||n_LecturaAnterior);

 s_rsp :='OK: INICIAlITZATS LECTURAS COMPTADOR ALTA EN CONTRACTE FETA:
        Id Comptador Alta en CONTRACTE: ' ||lec_idComptadorLectura;
pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
 ELSE
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('COMPTADOR ALTA EN CONTRACTE FETA  duplicat ');
 s_rsp := 'COMPTADOR ALTA EN CONTRACTE FETA  duplicat'||lec_idComptadorLectura;
 DBMS_OUTPUT.PUT_LINE(s_rsp);
 RAISE e_lectura;
END IF;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('_____________');
    DBMS_OUTPUT.put_line(s_rsp);
```

```
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 ROLLBACK;
END PRC_INICIA_LECTURA;
/*****************************************************************************
CONSULTA DE LECTURES
ORDENADES PER DATA ALTA
*****************************************************************************/
PROCEDURE PRC_CONSULTAR_LECTURES(
      s_rsp out NOCOPY VARCHAR)
 AS

CURSOR C_LECTURES IS
SELECT LECTURA.IDLECTURA,
LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA,
LECTURA.IDCOMPTADORLECTURA,
LECTURA.CONFIRMALECTURA,
LECTURA.DATALECTURA,
TIPO_LECTURA.DESCRIPCIOLECTURA,
LECTURA.CONSUMLECTURA
FROM LECTURA,LINEA_CONECTAR_COMPTADOR,TIPO_LECTURA
WHERE LECTURA.IDLECTURA=LECTURA.IDLECTURA
AND LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA=LECTURA.IDCOMPTADORLECTURA
AND LECTURA.IDTIPOLECTURA=TIPO_LECTURA.IDTIPOLECTURA
ORDER BY LECTURA.DATALECTURA,LECTURA.IDLECTURA;

BEGIN
 c_procesLog := 'PRC_CONSULTAR_LECTURES';
  c_dataHoraLog := SYSDATE;
  c_entradaLog := 'PRC_CONSULTAR_LECTURES';
  c_sortidalog := 's_rsp';
 SELECT COUNT (*)  INTO n_registres
  FROM LECTURA
  WHERE LECTURA.IDLECTURA=LECTURA.IDLECTURA;

 IF n_registres=0 THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTA LECTURES: ');
    s_rsp := 'No hi han lectures donades d´alta ';
    DBMS_OUTPUT.PUT_LINE(s_rsp);
    RAISE e_lectura;
 END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('          CONSULTA LECTURES              ');
    DBMS_OUTPUT.PUT_LINE('-------------------------------------------------------');
  OPEN C_LECTURES;
  FETCH C_LECTURES
  INTO n_idLectura,
  n_idLinea,
  n_idComptadorLectura,
  n_confirmaLectura,
```

```
 n_dataLectura,
 n_descTipusLectura,
 n_consumLectura;
 WHILE C_LECTURES%FOUND LOOP
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(
'   Numero de Lectura: '
||n_idLectura||
' Linea de lectura: '
||n_idLinea||
' Codi Comptador: '
||n_idComptadorLectura||
' Lectura validada: '
||n_confirmaLectura||
 ' Data Lectura: '
||n_dataLectura||
 ' Tipus de lectura: '
||n_descTipusLectura||
' Consum Lectura: '||n_consumLectura||
 ' Kw/h del mes. ');
 DBMS_OUTPUT.PUT_LINE(' ');
 s_rsp := 'OK  Numero de Lectura: '
||n_idLectura||
' Linea de lectura: '
||n_idLinea||
' Codi Comptador: '
||n_idComptadorLectura||
' Lectura validada: '
||n_confirmaLectura||
 ' Data Lectura: '
||n_dataLectura||
 ' Tipus de lectura: '
||n_descTipusLectura||
' Consum Lectura: '||n_consumLectura||
' Kw/h del mes. ';

 FETCH C_LECTURES
 INTO n_idLectura,
 n_idLinea,
 n_idComptadorLectura,
 n_confirmaLectura,
 n_dataLectura,
 n_descTipusLectura,
 n_consumLectura;

 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

 END LOOP;
 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE('Total Lectures: '||n_registres);
 DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_LECTURES ;
```

```
COMMIT;

EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi
      s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
  END PRC_CONSULTAR_LECTURES;
END "GESTION_LECTURA";
```

11.14 TRATAMIENTO DE GESTION HITORICO DE CONSUMOS.

Este procedimiento solo contine PRC_ACTUALIZAR_HIS_CONSUM que es ejecutado por el porcedimiento PRC_LEER_LECTURA del PACKAGE GESTION_LECTURA el cual actualiza los datos de este HISTORICO DE LECTURAS

DEFINICION	PROCEDIMIENTO
Inserta los datos recibidos del procedimiento GESTION_LECTURA.PRC_LEE_LECTURA	PRC_ACTUALIZAR_HIS_CONSUM

```
CREATE OR REPLACE
PACKAGE        "GESTION_HIST_CONSUMS" AS

  PROCEDURE PRC_ACTUALIZAR_HIS_CONSUM(
  g_idComptadorHist  in  HIST_LECTURA.IDCOMPTADORHISTLECTURA%TYPE,
  g_idLineaHistLectura IN HIST_LECTURA.IDLINEAHISTLECTURA%TYPE,
  g_anyHistLectura  in   HIST_LECTURA.ANYOHISTLECTURA%TYPE,
  g_mesHistLectura  in   HIST_LECTURA.MESHISTLECTURA%TYPE,
  g_totalConsumHistLectura in  HIST_LECTURA.TOTALCONSUMOHISTLECTURA%TYPE,
 s_rsp out NOCOPY VARCHAR2);

END "GESTION_HIST_CONSUMS";
```

11.14.1 PROCEDIMIENTO SPL PACKAGE HISTORICO DE CONSUMOS.

```
CREATE OR REPLACE
PACKAGE BODY        "GESTION_HIST_CONSUMS" AS
  --DECLARACIO DE: varibles generals
 s_sql          VARCHAR2 (2000);
 n_NUM_ERR          NUMBER(10);
 n_lineas number;

 c_procesLog                LOG_TFC.procesLog%TYPE;
 c_dataHoraLog     LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog       LOG_TFC.entradaLog%TYPE;
 c_sortidaLog       LOG_TFC.sortidaLog%TYPE;
 s_rsp                      LOG_TFC.RSPLOG%TYPE;
 g_idComptadorHist    HIST_LECTURA.IDCOMPTADORHISTLECTURA%TYPE;
 g_idLineaHistLectura HIST_LECTURA.IDLINEAHISTLECTURA%TYPE;

 sortida VARCHAR2(1000):='';
 e_hist_consum   EXCEPTION;


-- CONSTRUCCIO DE LA PROCEDURE PRC ACTUALIZAR HISTORIC CONSUMS
PROCEDURE PRC_ACTUALIZAR_HIS_CONSUM(
 g_idComptadorHist  in  HIST_LECTURA.IDCOMPTADORHISTLECTURA%TYPE,
 g_idLineaHistLectura IN HIST_LECTURA.IDLINEAHISTLECTURA%TYPE,
 g_anyHistLectura  in   HIST_LECTURA.ANYOHISTLECTURA%TYPE,
 g_mesHistLectura  in   HIST_LECTURA.MESHISTLECTURA%TYPE,
 g_totalConsumHistLectura in  HIST_LECTURA.TOTALCONSUMOHISTLECTURA%TYPE,
 s_rsp out NOCOPY VARCHAR2)
 AS

BEGIN
   -- CREACIO DE VARIABLES ENTRADA I SORTIDA
   c_procesLog := 'GESTION_HIST_CONSUMS';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='GESTION_HIST_CONSUMS:
          Contador: '||g_idComptadorHist||
         ' Linea: '||g_idLineaHistLectura||
         ', any Hist Lectura: '||g_anyHistLectura ||
         ', mes Hist Lectura: '||g_mesHistLectura||
         ', total Consum Hist Lectura: '||g_totalConsumHistLectura;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE('      HISTORIC DE LECTURES        ');
    DBMS_OUTPUT.PUT_LINE('--------------------------------- ');
    DBMS_OUTPUT.PUT_LINE(' Contador: '||g_idComptadorHist||
         ' Linea: '||g_idLineaHistLectura||
         ', any Hist Lectura: '||g_anyHistLectura ||
         ', mes Hist Lectura: '||g_mesHistLectura||
         ', total Consum Hist Lectura: '||g_totalConsumHistLectura);
    c_sortidalog := 's_rsp';
```

```
    if (g_idComptadorHist IS NULL) then
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('GESTIO_HIST_CONSUMS: ');
        s_rsp := 'Falta especificar Codi Comptador: ';
        DBMS_OUTPUT.PUT_LINE(s_rsp);
       RAISE e_hist_consum;
    END IF;
/*************************************************************************
     SELECT DISTINCT LINEA_CONECTAR.CONNECTA1_CENTRALCONECTA into n_lineas
       FROM LINEA_CONECTAR
       WHERE LINEA_CONECTAR.CONNECTA2_COMPTADORCONECTA=g_idComptadorHist;
    g_idLineaHistLectura:=n_lineas;
*************************************************************************/
-- INSTRUCCIO INSERT PER INTRUDUIR VALORS ANY I MES DE CONSUMS
   INSERT INTO HIST_LECTURA(IDCOMPTADORHISTLECTURA,
                        IDLINEAHISTLECTURA,
                        ANYOHISTLECTURA,
                        MESHISTLECTURA,
                        TOTALCONSUMOHISTLECTURA)
          VALUES(g_idComptadorHist,
                g_idLineaHistLectura,
               g_anyHistLectura,
               g_mesHistLectura,
               g_totalConsumHistLectura);


  --Gravem en la taula log----------------------------------------------------
          DBMS_OUTPUT.ENABLE;
          DBMS_OUTPUT.put_line ('OK: Insertat Consum al Historic
          del comptador: ' ||g_idComptadorHist||' del any '||g_anyHistLectura||
          ' del mes '||g_mesHistLectura||' amb un consum de '
          ||g_totalConsumHistLectura||' kw/h.');
          DBMS_OUTPUT.PUT_LINE('******** FI DE LES OPERACIONS ********');
          s_rsp :='Ok: Insertat Consum al Historic del comptador: '
          ||g_idComptadorHist||
          ', '||g_idLineaHistLectura||
          ' del any '||g_anyHistLectura ||
          ' del mes ' ||g_mesHistLectura||
          ' amb un consum de '||g_totalConsumHistLectura||' kw/h.';
          pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
          DBMS_OUTPUT.PUT_LINE(s_rsp);
     -- END LOOP;

  COMMIT;
EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi
       s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
     ELSE
```

```
      -- L'error si ha estat controlat per codi
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line(s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  ROLLBACK;
  --- FI DE PROCES
  END PRC_ACTUALIZAR_HIS_CONSUM;
END "GESTION_HIST_CONSUMS";
```

12 APENDICE 9 MODUL ESTADISTICAS.

Este Modulo de acuerdo con las especificaciones establecidas inicialmente efectua siete tipos de estadísticas que están organizadas por PACKAGE.

```
GESTIO_E1.sql
GESTIO_E2.sql
GESTIO_E3.sql
GESTIO_E4.sql
GESTIO_E5.sql
PACKAGE_GESTIO_E1.sql
PACKAGE_GESTIO_E2.sql
PACKAGE_GESTIO_E3.sql
PACKAGE_GESTIO_E4.sql
PACKAGE_GESTIO_E5.sql
PACKAGE_GESTIO_E6.sql
```

12.1 TRATAMIENTO DE GESTION ESTADISTICA E 1.

Dada una central de producción , extraer los consumos de los contadores que dependen de esta Central.

DEFINICION	PROCEDIMIENTO
Alta de los consumos de los contadores en una central de Producción	PRC_ALTA_E1

```
CREATE or replace PACKAGE        "GESTION_E_1" AS
PROCEDURE PRC_ALTA_E1(
   p_idCentral in E_1.IDE_1%TYPE,
   s_rsp out NOCOPY VARCHAR);
END "GESTION_E_1";
```

12.1.1 PROCEDIMIENTO SPL PACKAGE GESTION ESTADISTICA E 1.

```
CREATE OR REPLACE
PACKAGE BODY         "GESTION_E_1" AS
  c_procesLog           LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp               LOG_TFC.rspLog%TYPE;
  n_mes NUMBER;
  n_any NUMBER;
  n_idCentral NUMBER;
  n_idContador NUMBER;
  n_produccio VARCHAR2(50);
  n_consumContadorCentProdu NUMBER;
  n_linea NUMBER;
  totals NUMBER;
  n_registres  NUMBER;
  s_sql VARCHAR2 (2000);
   n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_e_1     EXCEPTION;

PROCEDURE PRC_ALTA_E1(
    p_idCentral in E_1.IDE_1%TYPE,
    s_rsp out NOCOPY VARCHAR)
    AS
 CURSOR C_LINEA_E1 IS
 --LINEA DE LA CENTRAL CONECTA
 SELECT DISTINCT
LINEA_CONECTA_CENTRAL.IDCONECTA
FROM LINEA_CONECTA_CENTRAL
WHERE LINEA_CONECTA_CENTRAL.CONNECTACPRODU=p_idCentral;

--OBTE COMPTADORS ON HI HA LECTURES
CURSOR C_CALCULA_E1 IS
SELECT DISTINCT HIST_LECTURA.IDCOMPTADORHISTLECTURA
  FROM LINEA_CONECTAR_COMPTADOR, HIST_LECTURA, LECTURA
  WHERE LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA=n_linea
  AND (SELECT     SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)
     FROM HIST_LECTURA
     GROUP BY HIST_LECTURA.IDCOMPTADORHISTLECTURA
     HAVING SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)>-1
     AND  HIST_LECTURA.IDCOMPTADORHISTLECTURA=2)>0;

CURSOR C_CALCULA IS
       SELECT     SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA),
     HIST_LECTURA.MESHISTLECTURA,HIST_LECTURA.ANYOHISTLECTURA
     FROM  HIST_LECTURA, LECTURA
     GROUP BY HIST_LECTURA.IDCOMPTADORHISTLECTURA,HIST_LECTURA.MESHISTLECTURA,
     HIST_LECTURA.ANYOHISTLECTURA
```

```
    HAVING SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)>-1
    AND  HIST_LECTURA.IDCOMPTADORHISTLECTURA=n_idContador;

--OBTE LES CENTRALS PRODUCIO CONECTADES EN LINEA
CURSOR C_CENTRALS IS
 SELECT DISTINCT LINEA_CONECTA_CENTRAL.CONNECTACPRODU AS CENTRAL
FROM HIST_LECTURA, LINEA_CONECTAR_COMPTADOR, CENTRAL, LINEA_CONECTA_CENTRAL
WHERE LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=n_linea
AND LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA=
LINEA_CONECTA_CENTRAL.IDCONECTA
AND CENTRAL.IDCLASSECENTRAL=1;

  BEGIN
   c_procesLog := 'PRC_ALTA_E1';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Id Central: ' ||p_idCentral;
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE(' ');
       DBMS_OUTPUT.PUT_LINE('ESTADISTIQUES E_1');
       DBMS_OUTPUT.PUT_LINE(' Consulta del Codi de Central de Produccio:'||p_idCentral);
   c_sortidalog := 's_rsp';
   SELECT COUNT(*) INTO n_registres
FROM CENTRAL
WHERE CENTRAL.IDCLASSECENTRAL=1
AND CENTRAL.IDCENTRAL=p_idCentral;
IF n_registres>0 THEN

OPEN  C_LINEA_E1;
FETCH C_LINEA_E1 INTO
 n_linea;

OPEN  C_CENTRALS;
FETCH C_CENTRALS INTO
 n_idCentral;

 OPEN  C_CALCULA_E1;
FETCH C_CALCULA_E1 INTO
n_idContador;

 OPEN C_CALCULA;
 FETCH C_CALCULA INTO n_consumContadorCentProdu,n_mes,n_any;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('     ESTADISTIQUES E_1     ');
   DBMS_OUTPUT.PUT_LINE(' ---------------------------');
WHILE C_CENTRALS%FOUND AND C_LINEA_E1%FOUND AND C_CALCULA%FOUND
AND  C_CALCULA_E1%FOUND LOOP

INSERT INTO E_1 (idCentral,idComptador_E_1,sumaConsums)

VALUES(n_idCentral,n_idContador,n_consumContadorCentProdu);
```

```
 DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' En la linea: '||n_linea||
 ' amb la Central de produccio: '||n_idCentral||
 ' el consum que fet el comptador:' ||n_idContador||'
  es de: '||n_consumContadorCentProdu|| ' Kw/h.  en durant el 20'||n_any );
 s_rsp := 'OK En la linea: '||n_linea||
 ' amb la Central de produccio: '||n_idCentral||
 ' el consum que fet el comptador:' ||n_idContador||'
  es de: '||n_consumContadorCentProdu|| ' Kw/h. ';
FETCH C_CENTRALS INTO
 n_idCentral;
 FETCH C_LINEA_E1 INTO
 n_linea;
 FETCH C_CALCULA_E1 INTO
n_idContador;
FETCH C_CALCULA INTO n_consumContadorCentProdu,N_MES,N_ANY;
 pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  DBMS_OUTPUT.PUT_LINE(' ');

CLOSE C_CENTRALS;
CLOSE C_LINEA_E1;
CLOSE C_CALCULA_E1;
CLOSE C_CALCULA;
ELSE
  DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('     ESTADISTIQUES E_1:     ');
   DBMS_OUTPUT.PUT_LINE(' La Central  '||p_idCentral||' no es de Produccio');
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
 s_rsp := ' La Central no es de Produccio: '||p_idCentral;
  RAISE  e_e_1;
END IF;
COMMIT;
  EXCEPTION

    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
  -- L'error no ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
   -- L'error si ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || s_rsp;
      END IF;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('   ');
        DBMS_OUTPUT.put_line (s_rsp);
        pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
ROLLBACK;
 END PRC_ALTA_E1;
```

```
END "GESTION_E_1";
```

12.2 TRATAMIENTO DE GESTION ESTADISTICA E 2.

Dada una línea de comunicación y un año en concreto, retornar el valor medio de la energía consumida, teniendo encuenta de que los contadores que dependen de esta línea se conectan por dos pars y que el flujo puede venir de uno de los dos, la conexión dual se efectua paa evitar las cídas de tensión de líneas y no dejar sin suministro energético a los usuarios.

DEFINICION	PROCEDIMIENTO
Alta de los consumos de los contadores en las líneas activas en un año en concreto.	PRC_ALTA_E1

```
create or replace PACKAGE       "GESTION_E_2" AS
PROCEDURE PRC_ALTA_E1(
p_linea in LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%TYPE,
p_any in HIST_LECTURA.ANYOHISTLECTURA%TYPE,
s_rsp out NOCOPY VARCHAR);
END "GESTION_E_2";
```

12.2.1 PROCEDIMIENTO SPL PACKAGE GESTION ESTADISTICA E 2.

```
CREATE OR REPLACE PACKAGE BODY "GESTION_E_2" AS
c_procesLog             LOG_TFC.procesLog%TYPE;
 c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog     LOG_TFC.entradaLog%TYPE;
 c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
 s_rsp              LOG_TFC.rspLog%TYPE;

 n_lineaActiva NUMBER;
 --n_mes NUMBER;
 n_any NUMBER;
 n_promitg NUMBER;
 n_linea number;
 --n_registres  NUMBER;
 n_dimensio number;

 s_sql VARCHAR2 (2000);
  n_NUM_ERR NUMBER(10);
 sortida varchar2(500):='';
 e_e_2     EXCEPTION;
```

```
PROCEDURE PRC_ALTA_E_2(
p_linea in LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA%TYPE,
p_any in HIST_LECTURA.ANYOHISTLECTURA%TYPE,
s_rsp out NOCOPY VARCHAR)
AS

CURSOR C_E_2 IS
SELECT DISTINCT LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA,
 HIST_LECTURA.ANYOHISTLECTURA, AVG(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)
 FROM LINEA_CONECTAR_COMPTADOR,LINEA_TIPUS,HIST_LECTURA
 WHERE LINEA_TIPUS.IDTLINEA=LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA
 AND LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA=
HIST_LECTURA.IDCOMPTADORHISTLECTURA
 AND HIST_LECTURA.ANYOHISTLECTURA=p_any
  AND LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=p_linea
 GROUP BY LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA,HIST_LECTURA.ANYOHISTLECTURA
 ORDER BY LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA ASC;

BEGIN
 c_procesLog := 'PRC_ALTA_E2';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Codi linea: ' ||p_linea||' any a consultar: '||p_any;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE(' ');
        DBMS_OUTPUT.PUT_LINE('ESTADISTIQUES E_2');
        DBMS_OUTPUT.PUT_LINE(' Consulta la linea Codi:'||p_linea||
        ' any 20'||p_any);
   c_sortidalog := 's_rsp';

 -- COMPROVACIO DE LA LINEA ESTIGUI ACTIVA
 SELECT DISTINCT COUNT(LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA) INTO n_lineaActiva
 FROM LINEA_CONECTAR_COMPTADOR
 WHERE LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=p_linea;

 IF n_lineaActiva=0 THEN
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('     ESTADISTIQUES E_2     ');
  DBMS_OUTPUT.PUT_LINE(' La Linea:  '||p_linea||' no esta en funcionament
  o no esta activada');
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
 s_rsp := ' La Linea:  '||p_linea||' no esta en funcionament
  o no esta activada';
 RAISE  e_e_2;
END IF;

-- COMPROVO L'ANY
 SELECT DISTINCT COUNT(HIST_LECTURA.ANYOHISTLECTURA)INTO n_any
 FROM HIST_LECTURA
 WHERE HIST_LECTURA.ANYOHISTLECTURA= p_any;

-- MIREM LA DIMESSIO
 SELECT DISTINCT LINEA_TIPUS.CONSUMMAXIMTLINEA INTO n_dimensio
```

```
 FROM LINEA_TIPUS,LINEA_CONECTAR_COMPTADOR
 WHERE LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=3
 AND LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=LINEA_TIPUS.IDTLINEA;

 IF n_any=0 THEN
 DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('      ESTADISTIQUES E_2            ');
   DBMS_OUTPUT.PUT_LINE(' La Linea:  '||p_linea||' no té lectures
   en aquest any: 20'||p_any);
     DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
    s_rsp := ' La Linea:  '||p_linea||' no té lectures
   en aquest any: 20'||p_any;
  RAISE  e_e_2;
END IF;
  IF n_dimensio=0 THEN
 DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('      ESTADISTIQUES E_2            ');
   DBMS_OUTPUT.PUT_LINE(' La Linea:  '||p_linea||' no esta dimensionada per
   no tenir assignat un tipus de linea. ');
     DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
    s_rsp := ' La Linea:  '||p_linea||' no esta dimensionada per
   no tenir assignat un tipus de linea. ';
  RAISE  e_e_2;
END IF;

  DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('    ESTADISTIQUES E_2     ');
   DBMS_OUTPUT.PUT_LINE(' ----------------------------');

OPEN C_E_2;
FETCH C_E_2 INTO n_linea,n_any,n_promitg;

WHILE C_E_2%FOUND LOOP
IF n_promitg<n_dimensio THEN
  DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE(' Linea: '
   ||n_linea||
   ' any: '||n_any||
   ' promitg de '||n_promitg||' Kw/h tolerable dins
   dels maxims permes de '||n_dimensio||' Kw/h. en la linea. ');
   s_rsp := 'OK   ESTADISTIQUES E_2:
    Linea: '
   ||n_linea||
   ' any: '||n_any||
   ' promitg tolerable dins del maxims permes:
   ' ||n_promitg;
  INSERT INTO  E_2 (idLineaConectaE_2,ANYOE_2,
  CONSUMOCONTADORESE_2,
  CONSUMOENERGIAMEDIOE_2)
  VALUES(n_linea,n_any,n_dimensio,n_promitg);
```

```
END IF;
FETCH C_E_2 INTO
n_linea,
n_any,
n_promitg;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   DBMS_OUTPUT.PUT_LINE(' ');
CLOSE C_E_2;

COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;
END PRC_ALTA_E_2;
END "GESTION_E_2";
```

12.3 TRATAMIENTO DE GESTION ESTADISTICA E 3.

Extracción de datos de la línea que ha estado más sobrecargada a nivel de energía consumida.

DEFINICION	PROCEDIMIENTO
Alta de las líneas más sobrecargadas	PRC_ALTA_E_3

```
CREATE OR REPLACE
PACKAGE  "GESTION_E_3" AS
PROCEDURE PRC_ALTA_E_3(
  p_any in HIST_LECTURA.ANYOHISTLECTURA%TYPE,
  s_rsp out NOCOPY VARCHAR);

END;
```

12.3.1 PROCEDIMIENTO SPL PACKAGE GESTION ESTADISTICA E 3.

```
CREATE OR REPLACE
PACKAGE BODY "GESTION_E_3" AS
c_procesLog             LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp              LOG_TFC.rspLog%TYPE;
  n_registres NUMBER;
  n_lineaActiva NUMBER;
  n_central NUMBER;
  n_energiaTolerable NUMBER;
  n_comsumEnergia number;
  t_energiaTolerable number;

  s_sql VARCHAR2 (2000);
   n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_e_3     EXCEPTION;

PROCEDURE PRC_ALTA_E_3(
 p_any in HIST_LECTURA.ANYOHISTLECTURA%TYPE,
  s_rsp out NOCOPY VARCHAR)
 AS
 CURSOR C_CONSUMO_CENTRAL IS
 SELECT DISTINCT SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)
 FROM HIST_LECTURA, LINEA_CONECTAR_COMPTADOR,LINEA_CONECTA_CENTRAL,
 COMPTADOR,CENTRAL
 WHERE HIST_LECTURA.IDCOMPTADORHISTLECTURA=
 LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA
 AND LINEA_CONECTAR_COMPTADOR.IDESTATLINEACONECTA=1
 AND LINEA_CONECTA_CENTRAL.IDCONECTA=
 LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA
 AND LINEA_CONECTA_CENTRAL.IDESTATC=1
 AND COMPTADOR.IDESTADOCOMPATOR=1
 AND CENTRAL.ESTATCENTRAL=1
 AND COMPTADOR.IDCOMPTADOR=
 LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA
 AND CENTRAL.IDCENTRAL=
 LINEA_CONECTA_CENTRAL.CONNECTACPRODU
 AND COMPTADOR.IDCOMPTADOR=
```

```
 LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA
AND CENTRAL.IDCENTRAL=n_central;
CURSOR C_ENERGIA_TOLERA IS
 SELECT DISTINCT LINEA_CONECTA_CENTRAL.CONNECTACPRODU,
LINEA_TIPUS.IDTLINEA,LINEA_TIPUS.CONSUMMAXIMTLINEA
INTO n_central,n_lineaActiva,n_energiaTolerable
FROM LINEA_TIPUS, LINEA_CONECTAR_COMPTADOR,LINEA_CONECTA_CENTRAL
WHERE LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA=
LINEA_CONECTA_CENTRAL.IDCONECTA
AND LINEA_CONECTA_CENTRAL.IDESTATC=1
AND LINEA_CONECTAR_COMPTADOR.IDESTATLINEACONECTA=1
AND LINEA_TIPUS.IDESTATTLINEA=1
AND LINEA_TIPUS.IDTLINEA=
LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA;
 BEGIN
  c_procesLog := 'PRC_ALTA_E2';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Any a consultar: '||p_any;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE(' ');
        DBMS_OUTPUT.PUT_LINE('     ESTADISTIQUES E_3            ');
        DBMS_OUTPUT.PUT_LINE('_____________________________________');
        DBMS_OUTPUT.PUT_LINE(' Consulta  Any : 20'||p_any);
        DBMS_OUTPUT.PUT_LINE('_____________________________________ ');
        DBMS_OUTPUT.PUT_LINE(' ');
   c_sortidalog := 's_rsp';

-- NOMBRE LINEA CONECTADES
SELECT DISTINCT COUNT(LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA) INTO n_registres
FROM LINEA_CONECTAR_COMPTADOR, LINEA_TIPUS
WHERE LINEA_CONECTAR_COMPTADOR.IDLINEATIPOLINEACONECTA=
LINEA_TIPUS.IDTLINEA;
 IF n_registres=0 THEN
 DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('       ESTADISTIQUES E_3            ');
   DBMS_OUTPUT.PUT_LINE(' No hi han lineas connectades. ');
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   s_rsp := ' No hi han lineas connectades. ';
  RAISE  e_e_3;
END IF;
  DBMS_OUTPUT.PUT_LINE('   CONTROL LINEAS SOBRECARREGADES E_3  ');
   DBMS_OUTPUT.PUT_LINE('--------------------------------------');

OPEN C_ENERGIA_TOLERA;
FETCH C_ENERGIA_TOLERA INTO
n_central,n_lineaActiva,n_energiaTolerable;
 WHILE  C_ENERGIA_TOLERA%FOUND LOOP


SELECT DISTINCT SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)  INTO n_comsumEnergia
 FROM HIST_LECTURA, LINEA_CONECTAR_COMPTADOR,LINEA_CONECTA_CENTRAL,
 COMPTADOR,CENTRAL
```

```
 WHERE HIST_LECTURA.IDCOMPTADORHISTLECTURA=
 LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA
 AND LINEA_CONECTAR_COMPTADOR.IDESTATLINEACONECTA=1
 AND LINEA_CONECTA_CENTRAL.IDCONECTA=
 LINEA_CONECTAR_COMPTADOR.CONNECTA1_CENTRALCONECTA
 AND LINEA_CONECTA_CENTRAL.IDESTATC=1
 AND COMPTADOR.IDESTADOCOMPATOR=1
 AND CENTRAL.ESTATCENTRAL=1
 AND COMPTADOR.IDCOMPTADOR=
 LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA
 AND CENTRAL.IDCENTRAL=
 LINEA_CONECTA_CENTRAL.CONNECTACPRODU
 AND COMPTADOR.IDCOMPTADOR=
 LINEA_CONECTAR_COMPTADOR.CONNECTA2_COMPTADORCONECTA
AND CENTRAL.IDCENTRAL=n_central
AND HIST_LECTURA.ANYOHISTLECTURA=P_ANY
ORDER BY CENTRAL.IDCENTRAL;


IF n_comsumEnergia>n_energiaTolerable then
 DBMS_OUTPUT.PUT_LINE('      ');
    DBMS_OUTPUT.PUT_LINE('   LINEA SOBRECARREGADES   ');
   DBMS_OUTPUT.PUT_LINE('------------------------------');
   t_energiaTolerable:=n_comsumEnergia-n_energiaTolerable;

   DBMS_OUTPUT.PUT_LINE(' Linea '||n_lineaActiva);
    DBMS_OUTPUT.PUT_LINE(' Central  '||n_central);
   DBMS_OUTPUT.PUT_LINE(' Coconsum  '||n_comsumEnergia||' Kw/h.');
   DBMS_OUTPUT.PUT_LINE(' Tolerable  '||n_energiaTolerable||' Kw/h.' );
   DBMS_OUTPUT.PUT_LINE(' Sobrecarregada amb '||t_energiaTolerable||' Kw/h.');
    DBMS_OUTPUT.PUT_LINE(' ');
    s_rsp :='OK sobrecarregades Linea '||n_lineaActiva||
    ' Central  '||n_central||
   ' Coconsum  '||n_comsumEnergia||' Kw/h.  Tolerable  '
   ||n_energiaTolerable||' Kw/h.
   Sobrecarregada amb '||t_energiaTolerable||' Kw/h.';
END IF;
INSERT INTO E_3(linea_E3,consumoContadores_E3,any_E3)
 VALUES(n_lineaActiva,n_comsumEnergia,P_ANY);

 DBMS_OUTPUT.PUT_LINE('    ');
  DBMS_OUTPUT.PUT_LINE('   LINEA OPTIMITZADA  ');
   DBMS_OUTPUT.PUT_LINE('------------------------------');
  t_energiaTolerable:=n_comsumEnergia-n_energiaTolerable;
   DBMS_OUTPUT.PUT_LINE(' Linea '||n_lineaActiva);
   DBMS_OUTPUT.PUT_LINE(' Central '||n_central);
   DBMS_OUTPUT.PUT_LINE(' Cosum '||n_comsumEnergia||' Kw/h.');
   DBMS_OUTPUT.PUT_LINE(' Tolerable  '||t_energiaTolerable||' Kw/h.');
     DBMS_OUTPUT.PUT_LINE(' ');
s_rsp :='OK optimitzades: Linea '||n_lineaActiva||
' Central '||n_central||
' Cosum '||n_comsumEnergia||' Kw/h.
```

```
  Tolerable  '||t_energiaTolerable||' Kw/h.';

 --OMPLE LA TAULA
  IF n_comsumEnergia=0 THEN
  n_comsumEnergia:=0;
  END IF;
INSERT INTO E_3(linea_E3,consumoContadores_E3,any_E3)
 VALUES(n_lineaActiva,n_comsumEnergia,P_ANY);

FETCH C_ENERGIA_TOLERA INTO
n_central,n_lineaActiva,n_energiaTolerable;

pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('_____ ');
  DBMS_OUTPUT.PUT_LINE('Total Lineas: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
DBMS_OUTPUT.PUT_LINE(' ');
CLOSE C_ENERGIA_TOLERA;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
         -- L'error no ha estat controlat per codi----------------------------------------
         s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
         -- L'error si ha estat controlat per codi----------------------------------------
         s_rsp := 'Error: ' || s_rsp;
      END IF;
      DBMS_OUTPUT.ENABLE;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.put_line (s_rsp);
      pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
      ROLLBACK;

END PRC_ALTA_E_3;
END "GESTION_E_3";
```

12.4 TRATAMIENTO DE GESTION ESTADISTICA E 4.

Extracción de datos dado un año obtener el porcentaje de líneas que superan el 50% de la energía consmida.

DEFINICION	PROCEDIMIENTO
Alta de las líneas que han superado en su consumo el 50% con respecto al consumo global en un año.	PRC_ALTA_E_4

```
CREATE OR REPLACE
PACKAGE        "GESTION_E_4" AS
 PROCEDURE PRC_ALTA_E_4(
 p_any in HIST_LECTURA.ANYOHISTLECTURA%TYPE,
  s_rsp out NOCOPY VARCHAR);
END;
```

12.4.1 PROCEDIMIENTO SPL PACKAGE GESTION ESTADISTICA E 4.

```
CREATE OR REPLACE
PACKAGE BODY "GESTION_E_4" AS

  c_procesLog          LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp              LOG_TFC.rspLog%TYPE;
  n_aux NUMBER;
  n_percent NUMBER (9,2);
  n_linea NUMBER;
  n_numLineas NUMBER;
  n_central NUMBER;
  n_TotalComsumGlobal NUMBER;
  n_comsumEnergiaMax number;
  n_valorMitat number;
  n_maxValorLinea number;
  n_registres NUMBER;
  s_sql VARCHAR2 (2000);
   n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_e_4     EXCEPTION;
PROCEDURE PRC_ALTA_E_4(
 p_any in HIST_LECTURA.ANYOHISTLECTURA%TYPE,
  s_rsp out NOCOPY VARCHAR)
```

```
AS
CURSOR C_LINEAS IS
-- OBTING LES LINEAS
SELECT DISTINCT HIST_LECTURA.IDLINEAHISTLECTURA,
SUM( HIST_LECTURA.TOTALCONSUMOHISTLECTURA)
FROM  LINEA_CONECTAR_COMPTADOR,HIST_LECTURA
GROUP BY HIST_LECTURA.ANYOHISTLECTURA,HIST_LECTURA.IDLINEAHISTLECTURA
HAVING SUM( HIST_LECTURA.TOTALCONSUMOHISTLECTURA)>-1
and HIST_LECTURA.ANYOHISTLECTURA=p_any
AND LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=
HIST_LECTURA.IDLINEAHISTLECTURA
ORDER BY HIST_LECTURA.IDLINEAHISTLECTURA;
BEGIN
 c_procesLog := 'PRC_ALTA_E2';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Any a consultar: '||p_any;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE(' ');
        DBMS_OUTPUT.PUT_LINE('     ESTADISTIQUES E_4            ');
        DBMS_OUTPUT.PUT_LINE('_____________________________________');
        DBMS_OUTPUT.PUT_LINE(' Consulta  Any : 20'||p_any);
        DBMS_OUTPUT.PUT_LINE('_____________________________________ ');
        DBMS_OUTPUT.PUT_LINE(' ');
   c_sortidalog := 's_rsp';

--- NOMBRE TOTAL DE LINEAS
SELECT DISTINCT COUNT (HIST_LECTURA.IDLINEAHISTLECTURA) INTO n_numLineas
FROM  HIST_LECTURA,LINEA_CONECTAR_COMPTADOR
WHERE HIST_LECTURA.ANYOHISTLECTURA=p_any
AND LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=
HIST_LECTURA.IDLINEAHISTLECTURA;

IF n_numLineas=0 THEN
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE(' No hi han lineas que hagin consumit res o no hi han lineas en valors. ');
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   s_rsp := 'No hi han lineas que hagin consumit res o no hi han lineas en valors. ';
 RAISE  e_e_4;
END IF;
SELECT MAX(SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)) INTO n_comsumEnergiaMax
FROM HIST_LECTURA
GROUP BY HIST_LECTURA.ANYOHISTLECTURA,HIST_LECTURA.IDLINEAHISTLECTURA
HAVING  SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)>-1
AND HIST_LECTURA.ANYOHISTLECTURA=p_any;

-- total del consum
SELECT DISTINCT SUM( HIST_LECTURA.TOTALCONSUMOHISTLECTURA) INTO n_TotalComsumGlobal
FROM  HIST_LECTURA
WHERE HIST_LECTURA.ANYOHISTLECTURA=p_any;
-- del total n_valorMitat te el 50%
```

```
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('      ');
   DBMS_OUTPUT.PUT_LINE('   ESTADISTIQUES E_4     ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------');
OPEN C_LINEAS;
FETCH C_LINEAS INTO
n_linea,
n_maxValorLinea;

WHILE  C_LINEAS%FOUND   LOOP
n_valorMitat:=n_TotalComsumGlobal/2;
n_aux:=n_maxValorLinea-n_valorMitat;
n_percent:=(n_aux*100)/n_TotalComsumGlobal;
if n_valorMitat<n_maxValorLinea THEN
SELECT COUNT(HIST_LECTURA.IDLINEAHISTLECTURA) into n_registres
FROM HIST_LECTURA
WHERE HIST_LECTURA.IDLINEAHISTLECTURA=n_linea;
INSERT INTO E_4 (LINEAE_4,ANYOE_4,PERCENTAGEE_4)
VALUES(n_linea,p_any,n_percent);
DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' Linea: '||n_linea);
   DBMS_OUTPUT.PUT_LINE(' Te un exces de:  '||n_aux||' Kw/h.' );
   DBMS_OUTPUT.PUT_LINE(' Representa el  '||n_percent||'% sobre el 50% del total anual.');
   DBMS_OUTPUT.PUT_LINE(' Consum anual de la linea:  '||n_maxValorLinea||' Kw/h.');
      DBMS_OUTPUT.PUT_LINE(' ');
    s_rsp := 'OK linea que sobrepasa el 50% es la linea: '||n_linea||
    ' Representa el  '||n_percent||'% sobre el 50% del total anual.
     Te un exces de:  '||n_aux||' Kw/h.
     Consum:  '||n_maxValorLinea||' Kw/h.';
   ELSE
 DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.PUT_LINE(' La linea: '||n_linea||' no ha consumit per sobre del 50%. ');
     DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
    s_rsp := 'La linea: '||n_linea||' no ha consumit per sobre del 50%. ';
  RAISE  e_e_4;
END IF;
FETCH C_LINEAS INTO
n_linea,
n_maxValorLinea;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('_____________________________________ ');
  DBMS_OUTPUT.PUT_LINE('Total Lineas analitzades: '||n_numLineas);
  DBMS_OUTPUT.PUT_LINE('Total Genral consums: '||n_valorMitat||' Kw/h. any: 20'||p_any);
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
DBMS_OUTPUT.PUT_LINE(' ');
CLOSE C_LINEAS;
```

```
COMMIT;
COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

 END PRC_ALTA_E_4;
END "GESTION_E_4";
```

12.5 TRATAMIENTO DE GESTION ESTADISTICA E 5.

Extracción de datos dado un año obtener el porcentaje de líneas que no superen el 30% de la energía consmida.

DEFINICION	PROCEDIMIENTO
Alta de las líneas que no han superado en su consumo el 30% con respecto al consumo global en un año.	PRC_ALTA_E_5

```
create or replace PACKAGE        "GESTION_E_5" AS
 PROCEDURE PRC_ALTA_E_5(
 p_any in HIST_LECTURA.ANYOHISTLECTURA%TYPE,
  s_rsp out NOCOPY VARCHAR);
END;
```

12.5.1 PROCEDIMIENTO SPL PACKAGE GESTION ESTADISTICA E 5.

```
CREATE OR REPLACE
PACKAGE BODY "GESTION_E_5" AS
  c_procesLog           LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
```

```
 s_rsp              LOG_TFC.rspLog%TYPE;
 n_aux NUMBER;
 n_percent NUMBER (9,2);
 n_linea NUMBER;
 n_numLineas NUMBER;
 n_central NUMBER;
 n_TotalComsumGlobal NUMBER;
 n_comsumEnergiaMax number;
 n_valorMitat number;
 n_maxValorLinea number;
 n_registres NUMBER;
 s_sql VARCHAR2 (2000);
  n_NUM_ERR NUMBER(10);
 sortida varchar2(500):='';
 e_e_5     EXCEPTION;
PROCEDURE PRC_ALTA_E_5(
 p_any in HIST_LECTURA.ANYOHISTLECTURA%TYPE,
  s_rsp out NOCOPY VARCHAR)
AS
CURSOR C_LINEAS IS
-- OBTING LES LINEAS
SELECT DISTINCT HIST_LECTURA.IDLINEAHISTLECTURA,
SUM( HIST_LECTURA.TOTALCONSUMOHISTLECTURA)
FROM  LINEA_CONECTAR_COMPTADOR,HIST_LECTURA
GROUP BY HIST_LECTURA.ANYOHISTLECTURA,HIST_LECTURA.IDLINEAHISTLECTURA
HAVING SUM( HIST_LECTURA.TOTALCONSUMOHISTLECTURA)>-1
and HIST_LECTURA.ANYOHISTLECTURA=p_any
AND LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=
HIST_LECTURA.IDLINEAHISTLECTURA
ORDER BY HIST_LECTURA.IDLINEAHISTLECTURA;
BEGIN
 c_procesLog := 'ESTADISTIQUES E_5 ';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :=' Any a consultar: '||p_any;
       DBMS_OUTPUT.ENABLE;
       DBMS_OUTPUT.PUT_LINE(' ');
       DBMS_OUTPUT.PUT_LINE('     ESTADISTIQUES E_5           ');
       DBMS_OUTPUT.PUT_LINE('_____________________________________');
       DBMS_OUTPUT.PUT_LINE(' Consulta  Any : 20'||p_any);
       DBMS_OUTPUT.PUT_LINE('_____________________________________ ');
       DBMS_OUTPUT.PUT_LINE(' ');
   c_sortidalog := 's_rsp';

--- NOMBRE TOTAL DE LINEAS
SELECT DISTINCT COUNT (HIST_LECTURA.IDLINEAHISTLECTURA) INTO n_numLineas
FROM  HIST_LECTURA,LINEA_CONECTAR_COMPTADOR
WHERE HIST_LECTURA.ANYOHISTLECTURA=p_any
AND LINEA_CONECTAR_COMPTADOR.IDLINEACONECTA=
HIST_LECTURA.IDLINEAHISTLECTURA;

IF n_numLineas=0 THEN
DBMS_OUTPUT.ENABLE;
```

```
DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('      ESTADISTIQUES E_5           ');
  DBMS_OUTPUT.PUT_LINE(' No hi han lineas no han consumit res o no hi han lineas en valors. ');
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
  s_rsp := ' No hi han lineas no han consumit res o no hi han lineas en valors. ';
 RAISE  e_e_5;
END IF;
SELECT MIN(SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)) INTO n_comsumEnergiaMax
FROM HIST_LECTURA
GROUP BY HIST_LECTURA.ANYOHISTLECTURA,HIST_LECTURA.IDLINEAHISTLECTURA
HAVING  SUM(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)>-1
AND HIST_LECTURA.ANYOHISTLECTURA=p_any;

-- total del consum
SELECT DISTINCT SUM( HIST_LECTURA.TOTALCONSUMOHISTLECTURA) INTO n_TotalComsumGlobal
FROM  HIST_LECTURA
WHERE HIST_LECTURA.ANYOHISTLECTURA=p_any;
-- del total n_valorMitat te el 50%

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('     ');
   DBMS_OUTPUT.PUT_LINE('   ESTADISTIQUES E_5     ');
   DBMS_OUTPUT.PUT_LINE('------------------------------');
OPEN C_LINEAS;
FETCH C_LINEAS INTO
n_linea,
n_maxValorLinea;

WHILE  C_LINEAS%FOUND   LOOP
n_valorMitat:=n_TotalComsumGlobal*0.30;
n_aux:=(n_maxValorLinea-n_valorMitat)*(-1);
n_percent:=(n_aux*100)/n_TotalComsumGlobal;
if n_valorMitat>n_maxValorLinea THEN
SELECT COUNT(HIST_LECTURA.IDLINEAHISTLECTURA) into n_registres
FROM HIST_LECTURA
WHERE HIST_LECTURA.IDLINEAHISTLECTURA=n_linea;
--INSERT INTO E_4 (LINEAE_4,ANYOE_4,PERCENTAGEE_4)
--VALUES(n_linea,p_any,n_percent);
DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE(' Linea: '||n_linea);
  DBMS_OUTPUT.PUT_LINE(' Te defecte de:  '||n_aux||' Kw/h.' );
  DBMS_OUTPUT.PUT_LINE(' Representa el  '||n_percent||'% per sota del 30% del total anual.');
  DBMS_OUTPUT.PUT_LINE(' Consum anual de la linea:  '||n_maxValorLinea||' Kw/h.');
  DBMS_OUTPUT.PUT_LINE('per un total global de:'||n_TotalComsumGlobal||' Kw/h.');
     DBMS_OUTPUT.PUT_LINE(' ');
   s_rsp := 'OK linea que no arriba el 30% es la linea: '||n_linea||
   ' Representa el  '||n_percent||'% 30% del total anual.
    Te un exces de:  '||n_aux||' Kw/h.
    Consum:  '||n_maxValorLinea||' Kw/h.';
 end if;
FETCH C_LINEAS INTO
n_linea,
```

```
n_maxValorLinea;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('_____ ');
  DBMS_OUTPUT.PUT_LINE('Total Lineas analitzades: '||n_numLineas);
  DBMS_OUTPUT.PUT_LINE('Total Genral consums: '||n_TotalComsumGlobal||' Kw/h. any: 20'||p_any);
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
DBMS_OUTPUT.PUT_LINE(' ');
CLOSE C_LINEAS;
COMMIT;
COMMIT;
EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
        -- L'error no ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
     ELSE
        -- L'error si ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line (s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
     ROLLBACK;

 END PRC_ALTA_E_5;
END "GESTION_E_5";
```

12.6 TRATAMIENTO DE GESTION ESTADISTICA E 6.

Extracción de de los 10 contadores que históricamente han tenido más consumos.

DEFINICION	PROCEDIMIENTO
Alta de los contadores que historicamante han tenido más consumos.	PRC_ALTA_E_6

```
create or replace PACKAGE        "GESTION_E_6" AS
 PROCEDURE PRC_ALTA_E_6(
 p_any in HIST_LECTURA.ANYOHISTLECTURA%TYPE,
  s_rsp out NOCOPY VARCHAR);
END;
```

12.6.1 PROCEDIMIENTO SPL PACKAGE GESTION ESTADISTICA E 6.

```
CREATE OR REPLACE
PACKAGE BODY "GESTION_E_6" AS
  c_procesLog           LOG_TFC.procesLog%TYPE;
  c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
  c_entradaLog     LOG_TFC.entradaLog%TYPE;
  c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
  s_rsp               LOG_TFC.rspLog%TYPE;
  n_idComptador NUMBER;
  n_any NUMBER;
  n_comsumEnergiaMax number;
   n_registres NUMBER;
   n_registresAny NUMBER;
   n_numComptadors NUMBER;
  s_sql VARCHAR2 (2000);
   n_NUM_ERR NUMBER(10);
  sortida varchar2(500):='';
  e_e_6     EXCEPTION;

 PROCEDURE PRC_ALTA_E_6(
  s_rsp out NOCOPY VARCHAR)
AS
CURSOR C_E6_10 IS
-- ELS  10 COMPATODORS
SELECT  DISTINCT HIST_LECTURA.IDCOMPTADORHISTLECTURA,
MAX(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)
FROM HIST_LECTURA
GROUP BY HIST_LECTURA.IDCOMPTADORHISTLECTURA
HAVING MAX(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)>-1
AND (
SELECT DISTINCT COUNT(HIST_LECTURA.IDCOMPTADORHISTLECTURA)
FROM HIST_LECTURA
GROUP BY HIST_LECTURA.IDCOMPTADORHISTLECTURA
HAVING COUNT (HIST_LECTURA.IDCOMPTADORHISTLECTURA)>-1)<11
ORDER BY HIST_LECTURA.IDCOMPTADORHISTLECTURA;

CURSOR C_E6_10_ANY IS
-- ELS 10 COMPTADORS PER ANY
SELECT  DISTINCT HIST_LECTURA.IDCOMPTADORHISTLECTURA,
 HIST_LECTURA.ANYOHISTLECTURA,
MAX(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)
FROM HIST_LECTURA
GROUP BY HIST_LECTURA.IDCOMPTADORHISTLECTURA, HIST_LECTURA.ANYOHISTLECTURA
HAVING MAX(HIST_LECTURA.TOTALCONSUMOHISTLECTURA)>-1
AND (
SELECT DISTINCT COUNT(HIST_LECTURA.IDCOMPTADORHISTLECTURA)
FROM HIST_LECTURA
GROUP BY HIST_LECTURA.IDCOMPTADORHISTLECTURA
HAVING COUNT (HIST_LECTURA.IDCOMPTADORHISTLECTURA)>-1)<11
ORDER BY HIST_LECTURA.ANYOHISTLECTURA,HIST_LECTURA.IDCOMPTADORHISTLECTURA;
```

```
BEGIN

c_procesLog := 'ESTADISTIQUES E_6 ';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='ESTADITICA E_5';
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE(' ');
        DBMS_OUTPUT.PUT_LINE('     ESTADISTIQUES E_6           ');
        DBMS_OUTPUT.PUT_LINE('_____________________________________');
        DBMS_OUTPUT.PUT_LINE(' ');
   c_sortidalog := 's_rsp';

-- MIRO SI HI HAN LECTURES EN L'HISTORIC
SELECT DISTINCT COUNT(HIST_LECTURA.ANYOHISTLECTURA)INTO n_numComptadors
FROM HIST_LECTURA
WHERE HIST_LECTURA.ANYOHISTLECTURA>0;
IF n_numComptadors=0 THEN
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('      ESTADISTIQUES E_6           ');
  DBMS_OUTPUT.PUT_LINE(' No hi han comptadors en que analitzar per no hever lectures fetes. ');
    DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   s_rsp := ' No hi han comptadors en que analitzar per no hever lectures fetes. ';
 RAISE  e_e_6;
END IF;

 DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('     ');
   DBMS_OUTPUT.PUT_LINE('   ESTADISTIQUES E_5     ');
   DBMS_OUTPUT.PUT_LINE('----------------------------');
-- CERQUEM ELS VALORS DELS 10 AMB MAXIM CONSUM
OPEN C_E6_10;
FETCH C_E6_10 INTO
n_idComptador,
n_comsumEnergiaMax;

OPEN C_E6_10_ANY;
FETCH C_E6_10_any INTO
n_idComptador,
n_any,
n_comsumEnergiaMax;

n_registres:=0;
WHILE C_E6_10%FOUND  LOOP
n_registres:=n_registres+1;
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(' Comptador: '||n_idComptador);
DBMS_OUTPUT.PUT_LINE(' Cosums:    '||n_comsumEnergiaMax||' Kw/h.' );
DBMS_OUTPUT.PUT_LINE(' ');
s_rsp := 'OK  Comptador: '||n_idComptador||
      'Consum:  '||n_comsumEnergiaMax||' Kw/h.';
INSERT INTO E_6 (E_6.IDCONTADORE_6,E_6.CONSUMOE_6)
```

```
VALUES(n_idComptador,n_comsumEnergiaMax);
FETCH C_E6_10 INTO
n_idComptador,
n_comsumEnergiaMax;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

END LOOP;

 DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('      ');
   DBMS_OUTPUT.PUT_LINE('   ESTADISTIQUES E_5 PER ANYS    ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------');
WHILE C_E6_10_ANY%FOUND  LOOP
n_registresAny:=n_registresAny+1;
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(' Comptador: '||n_idComptador);
DBMS_OUTPUT.PUT_LINE(' Any: '||n_any);
DBMS_OUTPUT.PUT_LINE(' Cosums:    '||n_comsumEnergiaMax||' Kw/h.' );
DBMS_OUTPUT.PUT_LINE(' ');
s_rsp := 'OK  Comptador: '||n_idComptador||
  ' Any: '||n_any||
      ' Consum:  '||n_comsumEnergiaMax||' Kw/h.';
FETCH C_E6_10_any INTO
n_idComptador,
n_any,
n_comsumEnergiaMax;
pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);

END LOOP;

 DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('_____ ');
  DBMS_OUTPUT.PUT_LINE('Total comptadors analitzades: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('Total comptadors anlitzats amb tots els anys: '||n_registresAny);
  DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
DBMS_OUTPUT.PUT_LINE(' ');

COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
```

```
    ROLLBACK;
END PRC_ALTA_E_6;

END "GESTION_E_6";
```

12.7 TRATAMIENTO DE GESTION ESTADISTICA E 7.

Extracción del consumo medio de los clientes.

DEFINICION	PROCEDIMIENTO
Alta Consumo medio de los clientes.	PRC_ALTA_E_7

```
CREATE OR REPLACE
PACKAGE    "GESTION_E_7" AS
 PROCEDURE PRC_ALTA_E_7(
   s_rsp out NOCOPY VARCHAR);
END;
```

12.7.1 PROCEDIMIENTO SPL PACKAGE GESTION ESTADISTICA E 7.

```
CREATE OR REPLACE PACKAGE BODY "GESTION_E_7" AS
 c_procesLog         LOG_TFC.procesLog%TYPE;
 c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog     LOG_TFC.entradaLog%TYPE;
 c_sortidaLog     LOG_TFC.sortidaLog%TYPE;
 s_rsp            LOG_TFC.rspLog%TYPE;

 n_dni CLIENT.DNICLIENT%TYPE;
n_DNICLIENT CLIENT.DNICLIENT%TYPE;
n_COGNOM1CLIENT CLIENT.COGNOM1CLIENT%TYPE;
n_COGNOM2CLIENT CLIENT.COGNOM2CLIENT%TYPE;
 n_comsumEnergiaMax number;
 n_registres NUMBER;
 n_idClient CLIENT.idClient%TYPE;
 n_DESCRIPCIOESTAT ESTAT.DESCRIPCIOESTAT%TYPE;
 n_DESCRIPCIOPERSONA PERSONA.DESCRIPCIOPERSONA%TYPE;
n_DESCRIPCIOVIA VIA.DESCRIPCIOVIA%TYPE;
n_DIRECCIONUBICACIO UBICACIO.DIRECCIONUBICACIO%TYPE;
n_NUMEROUBICACIO UBICACIO.NUMEROUBICACIO%TYPE;
n_PISUBICACIO UBICACIO.PISUBICACIO%TYPE;
n_PORTAUBICACIO UBICACIO.PORTAUBICACIO%TYPE;
n_CODIPOSTAL UBICACIO.CODIPOSTAL%TYPE;
n_DESCRIPCIOLOCALITAT LOCALITAT.DESCRIPCIOLOCALITAT%TYPE;
n_DESCRIPCIOPROVINCIA PROVINCIA.DESCRIPCIOPROVINCIA%TYPE;
n_DESCRIPCIOPAIS PAIS.DESCRIPCIOPAIS%TYPE;
n_NOMCLIENT CLIENT.NOMCLIENT%TYPE;
 n_contracte NUMBER;
 n_comptador NUMBER;
 s_sql VARCHAR2 (2000);
  n_NUM_ERR NUMBER(10);
 sortida varchar2(500):='';
 e_e_7    EXCEPTION;
```

```
PROCEDURE PRC_ALTA_E_7(
  s_rsp out NOCOPY VARCHAR)
AS
CURSOR C_MITJA_CLI IS
SELECT DISTINCT AVG (HIST_LECTURA.TOTALCONSUMOHISTLECTURA),
CLIENT.DNICLIENT,CONTRACTE.IDCONTADORCONTRACTE,
HIST_LECTURA.IDCOMPTADORHISTLECTURA
FROM CONTRACTE, CLIENT,COMPTADOR, HIST_LECTURA
GROUP BY CLIENT.DNICLIENT,HIST_LECTURA.IDCOMPTADORHISTLECTURA,
CONTRACTE.DNICLIENTECONTRACTE,
CONTRACTE.IDCONTADORCONTRACTE,COMPTADOR.IDCOMPTADOR
HAVING AVG (HIST_LECTURA.TOTALCONSUMOHISTLECTURA)>-1
AND CLIENT.DNICLIENT=CONTRACTE.DNICLIENTECONTRACTE
AND COMPTADOR.IDCOMPTADOR=CONTRACTE.IDCONTADORCONTRACTE
AND COMPTADOR.IDCOMPTADOR=HIST_LECTURA.IDCOMPTADORHISTLECTURA;
BEGIN
   c_procesLog := 'ESTADISTIQUES E_7 ';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='ESTADISTICA E_7';
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE(' ');
        DBMS_OUTPUT.PUT_LINE('     ESTADISTIQUES E_7           ');
        DBMS_OUTPUT.PUT_LINE('_____________________________________________');
        DBMS_OUTPUT.PUT_LINE(' ');
   c_sortidalog := 's_rsp';
SELECT COUNT (CONTRACTE.DNICLIENTECONTRACTE) INTO n_registres
FROM CONTRACTE, CLIENT, COMPTADOR
WHERE CLIENT.DNICLIENT=
CONTRACTE.DNICLIENTECONTRACTE
AND COMPTADOR.IDCOMPTADOR=
CONTRACTE.IDCONTADORCONTRACTE
AND CONTRACTE.IDESTATCONTRACTE=1;
IF n_registres=0 THEN
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(' ');
  DBMS_OUTPUT.PUT_LINE('      ESTADISTIQUES E_7           ');
  DBMS_OUTPUT.PUT_LINE(' No hi han client en que analitzar per no hever lectures fetes. ');
   DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
   s_rsp := ' No hi han clients en que analitzar per no hever lectures fetes. ';
 RAISE  e_e_7;
END IF;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('     ');
   DBMS_OUTPUT.PUT_LINE('   ESTADISTIQUES E_7        ');
   DBMS_OUTPUT.PUT_LINE('-----------------------------');
OPEN C_MITJA_CLI;
FETCH C_MITJA_CLI INTO
n_comsumEnergiaMax,
n_dni,
n_contracte,
n_comptador;

WHILE C_MITJA_CLI%FOUND  LOOP
SELECT
    CLIENT.IDCLIENT,
    CLIENT.DNICLIENT,
    CLIENT.NOMCLIENT,
    CLIENT.COGNOM1CLIENT,
    CLIENT.COGNOM2CLIENT,
    PERSONA.DESCRIPCIOPERSONA,
    VIA.DESCRIPCIOVIA,
    UBICACIO.DIRECCIONUBICACIO,
    UBICACIO.NUMEROUBICACIO,
    UBICACIO.PISUBICACIO,
    UBICACIO.PORTAUBICACIO,
    UBICACIO.CODIPOSTAL,
    LOCALITAT.DESCRIPCIOLOCALITAT,
    PROVINCIA.DESCRIPCIOPROVINCIA,
    PAIS.DESCRIPCIOPAIS
    INTO
    n_IDCLIENT,
    n_DNICLIENT,
```

```
    n_NOMCLIENT,
    n_COGNOM1CLIENT,
    n_COGNOM2CLIENT,
    n_DESCRIPCIOPERSONA,
    n_DESCRIPCIOVIA,
    n_DIRECCIONUBICACIO,
    n_NUMEROUBICACIO,
    n_PISUBICACIO,
    n_PORTAUBICACIO,
    n_CODIPOSTAL,
    n_DESCRIPCIOLOCALITAT,
    n_DESCRIPCIOPROVINCIA,
    n_DESCRIPCIOPAIS
FROM CLIENT,VIA,UBICACIO,LOCALITAT,PROVINCIA,PAIS,PERSONA
WHERE CLIENT.DNICLIENT=n_dni
AND CLIENT.IDTIPOCLIENT=PERSONA.IDPERSONA
AND CLIENT.IDUBICACLIENT=UBICACIO.IDUBICA
AND UBICACIO.IDVIAUBICACIO=VIA.IDVIA
AND UBICACIO.IDLOCALITATUBICACIO=LOCALITAT.IDLOCALITAT
AND LOCALITAT.IDPROVINCIALOCALITAT=PROVINCIA.IDPROVINCIA
AND PROVINCIA.IDPAISPROVINCIA=PAIS.IDPAIS;
DBMS_OUTPUT.ENABLE;
DBMS_OUTPUT.PUT_LINE(' Client DNI: '||n_DNICLIENT||' Consum mitja de:'
||n_comsumEnergiaMax||' Kw/h.');
DBMS_OUTPUT.PUT_LINE('Nom:    '||n_NOMCLIENT||'  Cognom1: '||n_COGNOM1CLIENT||
' Cognom2:' ||n_COGNOM2CLIENT);
DBMS_OUTPUT.PUT_LINE(' Tipus de Client: '||n_DESCRIPCIOPERSONA);
DBMS_OUTPUT.PUT_LINE(' Direccio: '||n_DESCRIPCIOVIA||'
'||n_DIRECCIONUBICACIO||' num.'||n_NUMEROUBICACIO||
' pis '||n_PISUBICACIO||' porta '||n_PORTAUBICACIO);
DBMS_OUTPUT.PUT_LINE(' Codi postal:'||n_CODIPOSTAL);
DBMS_OUTPUT.PUT_LINE(' Localitat:  '||n_DESCRIPCIOLOCALITAT||
' Provincia '||n_DESCRIPCIOPROVINCIA );
DBMS_OUTPUT.PUT_LINE(' Pais:       '||n_DESCRIPCIOPAIS);
DBMS_OUTPUT.PUT_LINE(' ');
s_rsp := 'OK  Client DNI: '||n_dni||
      ' Consum mitja  '||n_comsumEnergiaMax||' Kw/h.';

FETCH C_MITJA_CLI INTO
n_comsumEnergiaMax,
n_dni,
n_contracte,
n_comptador;

pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
  END LOOP;
  DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('Total clients: '||n_registres);
  DBMS_OUTPUT.PUT_LINE('****** FI DE LES OPERACIONS *******');
CLOSE C_MITJA_CLI;

COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    IF s_rsp IS NULL THEN
      -- L'error no ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
    ELSE
      -- L'error si ha estat controlat per codi----------------------------------------
      s_rsp := 'Error: ' || s_rsp;
    END IF;
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' ');
    DBMS_OUTPUT.put_line (s_rsp);
    pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog, s_rsp);
    ROLLBACK;

END PRC_ALTA_E_7;

END "GESTION_E_7";
```

13 APENDICE 10 MODUL CONSULTAS.

DEFINICION	PROCEDIMIENTO
Consulta D	PRC_CONSULTA_C4
Consulta G	PRC_CONSULTA_C7

```
CREATE OR REPLACE
PACKAGE           "CONSULTAS" AS
PROCEDURE PRC_CONSULTA_C4(
p_dniClient in C4.DNICLIENT%TYPE,
p_idContracte in CONTRACTE.IDCONTRACTE%TYPE,
p_idModel in MODELO.IDMODELO%TYPE,
s_rsp out NOCOPY VARCHAR);
 PROCEDURE PRC_CONSULTA_C7(
p_any in MODELO.ANYOFABRICAMODELO%TYPE,
s_rsp out NOCOPY VARCHAR);
END CONSULTAS;
```

13.1.1 PROCEDIMIENTO SPL PACKAGE GESTION CONSULTAS.

```
CREATE OR REPLACE
PACKAGE BODY        "CONSULTAS" AS
n_registres     NUMBER;
 s_sql         VARCHAR2 (2000);
 n_NUM_ERR     NUMBER(10);
 n_any  VARCHAR2(4);
 valor number;
 n_comptador number;
 n_contador number;
 p_dniClient C4.IDC4%TYPE;
 c_procesLog     LOG_TFC.procesLog%TYPE;
 c_dataHoraLog    LOG_TFC.dataHoraLog%TYPE;
 c_entradaLog    LOG_TFC.entradaLog%TYPE;
 c_sortidaLog    LOG_TFC.sortidaLog%TYPE;
 n_idComptador COMPTADOR.IDCOMPTADOR%TYPE;
 n_anyModel MODELO.ANYOFABRICAMODELO%TYPE;
 n_descModel MODELO.DESCRIPCIOMODELO%TYPE;
 s_rsp   LOG_TFC.RSPLOG%TYPE;

 sortida VARCHAR2(1000):='';
  e_c7   EXCEPTION;
  e_c4   EXCEPTION;
PROCEDURE PRC_CONSULTA_C7(
p_any in MODELO.ANYOFABRICAMODELO%TYPE,
s_rsp out NOCOPY VARCHAR)
```

```
 AS
CURSOR C_C7 IS
SELECT COMPTADOR.IDCOMPTADOR,MODELO.ANYOFABRICAMODELO,
MODELO.DESCRIPCIOMODELO
from COMPTADOR, MODELO
where COMPTADOR.IDMODELOCOMPTADOR=
MODELO.IDMODELO
and   MODELO.ANYOFABRICAMODELO<p_any;
BEGIN
   c_procesLog := 'PRC_CONSULTA_C7';
   c_dataHoraLog := SYSDATE;
   c_entradaLog :='Any: ' ||p_any;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('CONSULTA_C7: Any de Fabricacio: ' ||p_any);
        DBMS_OUTPUT.PUT_LINE('_____');
   c_sortidalog := 's_rsp';

   IF p_any IS NULL  THEN
    DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTA_C7: any no correcta : '||p_any);
    DBMS_OUTPUT.PUT_LINE('Comptador any no correcta '||p_any);
        RAISE e_c7 ;
   END IF;
   -- MIRO SI HAN COMPTADORS SINO N'HI HN EMET EXCEPCIO
      SELECT COUNT(*) INTO valor
      FROM COMPTADOR,MODELO
      WHERE COMPTADOR.IDMODELOCOMPTADOR=modelo.IDMODELO
      AND MODELO.ANYOFABRICAMODELO<p_any;

 IF valor>0 THEN


 DBMS_OUTPUT.ENABLE;
 DBMS_OUTPUT.PUT_LINE(' ');
 DBMS_OUTPUT.PUT_LINE('   CONSULTA C7   ');
 DBMS_OUTPUT.PUT_LINE('---------- ');
OPEN  C_C7;
FETCH C_C7 INTO
n_idComptador,
n_anyModel,
n_descModel;
WHILE C_C7%FOUND  LOOP
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.PUT_LINE(' Comptador: '||n_idComptador);
     DBMS_OUTPUT.PUT_LINE(' Model : '||n_descModel);
     DBMS_OUTPUT.PUT_LINE(' Any Fabricacio : '||n_anyModel);
     s_rsp :='Ok C7 comptador: '||n_idComptador||' Model : '||n_descModel||' Any Fabricacio  :
'||n_anyModel;
FETCH C_C7 INTO
n_idComptador,
n_anyModel,
n_descModel;
     pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog,
s_rsp);
     DBMS_OUTPUT.PUT_LINE(s_rsp);

END LOOP;
    DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('_____');
   DBMS_OUTPUT.PUT_LINE('TOTAL COMPTADORS:  '||valor);
   DBMS_OUTPUT.PUT_LINE('******** FI DEL PROCES ********');
         DBMS_OUTPUT.PUT_LINE(' ');
CLOSE C_C7;
```

```
ELSE
DBMS_OUTPUT.ENABLE;
  DBMS_OUTPUT.PUT_LINE('  CONSULTA C7   ');
  DBMS_OUTPUT.PUT_LINE('no hi han comptadors. ');
  RAISE e_c7 ;
END IF;
COMMIT;
  EXCEPTION
    WHEN OTHERS THEN
      IF s_rsp IS NULL THEN
  -- L'error no ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || SQLCODE || SUBSTR(SQLERRM, 1, 100);
      ELSE
   -- L'error si ha estat controlat per codi----------------------------------------
        s_rsp := 'Error: ' || s_rsp;
      END IF;
        DBMS_OUTPUT.ENABLE;
        DBMS_OUTPUT.PUT_LINE('   ');
        DBMS_OUTPUT.put_line (s_rsp);
        pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog,
c_sortidaLog, s_rsp);
ROLLBACK;

  END PRC_CONSULTA_C7;

PROCEDURE PRC_CONSULTA_C4(
    p_dniClient in C4.DNICLIENT%TYPE,
    p_idContracte in CONTRACTE.IDCONTRACTE%TYPE,
    p_idModel in MODELO.IDMODELO%TYPE,
    s_rsp out NOCOPY VARCHAR)
 AS
 contadorContractes number;
 contadorDniClient number;
 contadorC4 number;
 n_ndiClient varchar2(10);
 n_idModel number;
 n_idContracte number;
 n_descripcio varchar2(30);
 s_dniclient varchar2(10);
 s_idcontracte number;
 s_idModel number;
  e_C4    EXCEPTION;
  BEGIN
  c_procesLog := 'PRC_CONSULTA_C4';
    c_dataHoraLog := SYSDATE;
    c_entradaLog :='CONSULTA CLIENTS amb contracte i model de
    comptador: introdueix DNI '||p_dniClient;

         DBMS_OUTPUT.ENABLE;
         DBMS_OUTPUT.PUT_LINE(' CONSULTA CLIENTS amb  DNI
         Client amb contracte i model de
         comptador '||p_dniClient);
         DBMS_OUTPUT.PUT_LINE('_____');
    c_sortidalog := 's_rsp';
/*******************************************************
    SELECT CONTRACTE.DNICLIENTECONTRACTE,
    CONTRACTE.IDCONTRACTE,
    MODELO.IDMODELO,
    MODELO.DESCRIPCIOMODELO
    FROM CONTRACTE,CLIENT,COMPTADOR,MODELO
    WHERE CONTRACTE.DNICLIENTECONTRACTE=CLIENT.DNICLIENT
    AND MODELO.IDMODELO=COMPTADOR.IDMODELOCOMPTADOR
    AND CONTRACTE.IDCONTADORCONTRACTE=COMPTADOR.IDCOMPTADOR;
```

```
   SELECT COUNT(DISTINCT CONTRACTE.DNICLIENTECONTRACTE)
    FROM CONTRACTE,CLIENT,COMPTADOR,MODELO
   WHERE CONTRACTE.DNICLIENTECONTRACTE=CLIENT.DNICLIENT
   AND MODELO.IDMODELO=COMPTADOR.IDMODELOCOMPTADOR;
 ************************************************************/
IF p_dniClient IS NULL THEN
   DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTA_C7: falta el dni : '||p_dniClient);
    DBMS_OUTPUT.PUT_LINE('CONSULTA_C7: falta el dni : '||p_dniClient);
        RAISE e_c7 ;
END IF;

-- SI TE CONTRACTE
SELECT COUNT(*) INTO n_registres
FROM CONTRACTE
WHERE CONTRACTE.DNICLIENTECONTRACTE=p_dniClient;

IF n_registres=0 THEN
 DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE('CONSULTA_C7: ciient sense contracte : '||p_dniClient);
    DBMS_OUTPUT.PUT_LINE('CONSULTA_C7: ciient sense contracte : '||p_dniClient);
        RAISE e_C4   ;
END IF;
 --OBTENIR ID CONTRACTE   n_idContracte
   SELECT CONTRACTE.IDCONTRACTE INTO n_idContracte
   FROM CONTRACTE
   where CONTRACTE.DNICLIENTECONTRACTE=p_dniClient;

-- COMPTADOR AMB CONTRACTE comptador Contracte
   SELECT CONTRACTE.IDCONTADORCONTRACTE INTO contadorContractes
   FROM COMPTADOR,CONTRACTE
   WHERE COMPTADOR.IDCOMPTADOR=CONTRACTE.IDCONTADORCONTRACTE
  AND CONTRACTE.DNICLIENTECONTRACTE=p_dniClient;

-- MODEL DEL COMPTADOR EN CONTRACTE id model
  SELECT COMPTADOR.IDMODELOCOMPTADOR INTO   n_idModel
  FROM COMPTADOR,MODELO
  WHERE MODELO.IDMODELO=COMPTADOR.IDCOMPTADOR
  AND COMPTADOR.IDCOMPTADOR=contadorContractes;
-- OBTENIM dni del contracte
SELECT DISTINCT CONTRACTE.DNICLIENTECONTRACTE,
   CONTRACTE.IDCONTRACTE,
   MODELO.IDMODELO          INTO s_dniClient,s_idContracte,s_idModel
    FROM CONTRACTE,CLIENT,COMPTADOR,MODELO
   WHERE CONTRACTE.DNICLIENTECONTRACTE=p_dniClient
   AND MODELO.IDMODELO= n_idModel
   AND CONTRACTE.IDCONTADORCONTRACTE=contadorContractes
   AND CONTRACTE.IDCONTRACTE=n_idContracte;


   IF n_registres>0 THEN
   contadorC4:=1;
   WHILE contadorC4<=contadorDniClient LOOP
   SELECT CONTRACTE.DNICLIENTECONTRACTE,
   CONTRACTE.IDCONTRACTE,
   MODELO.IDMODELO,
   MODELO.DESCRIPCIOMODELO INTO n_ndiClient,n_idContracte,n_idModel,n_descripcio
   FROM CONTRACTE,CLIENT,COMPTADOR,MODELO
   WHERE CONTRACTE.DNICLIENTECONTRACTE=p_dniClient
   AND MODELO.IDMODELO= n_idModel
   AND CONTRACTE.IDCONTADORCONTRACTE=COMPTADOR.IDCOMPTADOR
   AND CONTRACTE.IDCONTRACTE=contadorC4;
   contadorC4:=contadorC4+1;
   DBMS_OUTPUT.ENABLE;
```

```
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('DNI CLIENT:     '||n_ndiClient||
                  ' CODI CONTRACTE: '||n_idContracte||
                  ' CODI MODEL:     ' ||n_idModel||
                  ' DESCRIPCIO:     '||n_descripcio);

    c_sortidalog := 's_rsp';
    s_rsp :='Ok C4 DNI CLIENT:     '||s_dniClient||
                  ' CODI CONTRACTE: '||s_idContracte||
                  ' CODI MODEL:     ' ||n_idModel;
     pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog,
s_rsp);

      RAISE e_C4   ;
   END LOOP;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE(' ');
   DBMS_OUTPUT.PUT_LINE('Total : '||n_registres);
    DBMS_OUTPUT.PUT_LINE('******** FIN DE LES OPERACIONS ********');
  END IF;

   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.ENABLE;
   DBMS_OUTPUT.PUT_LINE('     CONSULTA CLIENTS C4   ');
   DBMS_OUTPUT.PUT_LINE('-------------------------- ');
   DBMS_OUTPUT.PUT_LINE('DNI CLIENT:  '||s_dniClient||
                  ' CODI CONTRACTE: '||s_idContracte||
                  ' CODI MODEL: '||n_idModel);
                  DBMS_OUTPUT.PUT_LINE(' Total : '||n_registres);
    s_rsp :='Ok C4 DNI CLIENT:     '||s_dniClient||
                  ' CODI CONTRACTE: '||s_idContracte||
                  ' CODI MODEL:     ' ||n_idModel;
     pkg_general.gravar_log_procedure(c_procesLog, c_dataHoraLog, c_entradaLog, c_sortidaLog,
s_rsp);
     DBMS_OUTPUT.ENABLE;
    DBMS_OUTPUT.PUT_LINE(' _____');
    DBMS_OUTPUT.PUT_LINE('Total : '||n_registres);
     DBMS_OUTPUT.PUT_LINE('******** FIN DE LES OPERACIONS ********');
     DBMS_OUTPUT.PUT_LINE(' ');
  COMMIT;
EXCEPTION
   WHEN OTHERS THEN
     IF s_rsp IS NULL THEN
       -- L'error no ha estat controlat per codi
       s_rsp := 'Error: ' || SQLCODE || SUBSTR (SQLERRM, 1, 100);
     ELSE
       -- L'error si ha estat controlat per codi
       s_rsp := 'Error: ' || s_rsp;
     END IF;
     DBMS_OUTPUT.ENABLE;
     DBMS_OUTPUT.PUT_LINE(' ');
     DBMS_OUTPUT.put_line(s_rsp);
     pkg_general.gravar_log_procedure (c_procesLog, c_dataHoraLog, c_entradaLog,
c_sortidaLog, s_rsp);
     ROLLBACK;

  END PRC_CONSULTA_C4;
END "CONSULTAS";
```

14 APENDICE 11 BANCO DE PRUEBAS Y CARGA DE DATOS.

Sea utilizado diferente información basada alguna de ella en la vida real con el fin de poder llevar acabo una simulación con extremo rigor.

En el fichero de Carga de datos CARREGA.sql hay todos lo procedimientos ordenados tal como se deben ejecutar, no obstante relaciona en este documento dicha información.

Al final de este Apendice hay algunas de las pantallas más siginificativas de las extracciones realizadas.

```
/*************************************************************************************************************************
Autor: Eduard Monzonis Hierro                                    TFC CONTROL ENERGIA
DATA:19/05/2012                                                                  UOC
                                          CARREGA DE DADES
*************************************************************************************************************************/
/
-- EXECUTA INSERTAR ESTAT
/*****************************************
GESTION_ESTAT.PRC_ALTA_ESTAT(
   P_DESCRIPCIOESTAT => P_DESCRIPCIOESTAT,
   sortida
 );
*****************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_ESTAT.PRC_ALTA_ESTAT('ALTA',sortida);
GESTION_ESTAT.PRC_ALTA_ESTAT('BAIXA',sortida);
END;
/
--CONSULTA
DECLARE
sortida VARCHAR(500):='';
BEGIN
 GESTION_ESTAT.PRC_CONSULTA_ESTAT(sortida);
END;
-- EXECUTA INSERTAR PERSONA
/*************************************************
 GESTION_PERSONA.PRC_ALTA_PERSONA(
   P_DESCRIPCIOPERSONA => P_DESCRIPCIOPERSONA,
   sortida
 );
*************************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_PERSONA.PRC_ALTA_PERSONA('Particular',sortida);
GESTION_PERSONA.PRC_ALTA_PERSONA('Empresa',sortida);
END;
```

```
/
-- EXECUTA INSERTAR TIPO LECTURA
/************************
 GESTION_TIPO_LECTURA.PRC_ALTA_TIPO_LECTURA(
   P_DESCRIPCIOTLECTURA => P_DESCRIPCIOTLECTURA,
   sortida
  );
*********************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_TIPO_LECTURA.PRC_ALTA_TIPO_LECTURA('Presencial',sortida);
GESTION_TIPO_LECTURA.PRC_ALTA_TIPO_LECTURA('Telemàtica',sortida);
END;
/
-- EXECUTA INSERTAR TIPUS FUNCIONS CENTRALS PRODUCCIO
--1         Kgrs. Emissions CO2
--2         Molins
--3         Residus Radiactius
--4         Plaques Solars
/*******************************
 GESTION_TIPUS_FUNCIONS.PRC_ALTA_TIPUS_FUNCIONS(
   P_DESCRIPCIOTFUNCIONS => P_DESCRIPCIOTFUNCIONS,
   sortida
  );
*********************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_TIPUS_FUNCIONS.PRC_ALTA_TIPUS_FUNCIONS('Kgrs. Emissions CO2',sortida);
GESTION_TIPUS_FUNCIONS.PRC_ALTA_TIPUS_FUNCIONS('Molins',sortida);
GESTION_TIPUS_FUNCIONS.PRC_ALTA_TIPUS_FUNCIONS('Residus Radiactius',sortida);
GESTION_TIPUS_FUNCIONS.PRC_ALTA_TIPUS_FUNCIONS('Plaques Solars',sortida);
END;
/
--CONSULTA TIPUS CENTRALS
DECLARE
sortida VARCHAR(1000):='';
BEGIN
  GESTION_TIPUS_FUNCIONS.PRC_CONSULTAR_TIPUS_FUNCIONS(sortida);
END;
/
-- TIPUS DE CENTRALS
/*********************************************
GESTION_TIPO_CENTRAL.PRC_ALTA_TIPO_CENTRAL(
   P_DESCRIPCIOTIPOCENTRAL => P_DESCRIPCIOTIPOCENTRAL,
   P_IDFUNCIONCENTRAL => P_IDFUNCIONCENTRAL,
   P_QUANTITATTIPOCENTRAL => P_QUANTITATTIPOCENTRAL,
   P_ESTATTIPOSCENTRAL => P_ESTATTIPOSCENTRAL,
   P_DATAESTATTIPOCENTRAL => P_DATAESTATTIPOCENTRAL,
   P_OBSERVACIOTIPOSCENTRAL => P_OBSERVACIOTIPOSCENTRAL,
   sortida
```

```
 );
************************************************************/
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_TIPO_CENTRAL.PRC_ALTA_TIPO_CENTRAL('Eolica',2,300,1,'12/04/2012','energia neta',sortida);
GESTION_TIPO_CENTRAL.PRC_ALTA_TIPO_CENTRAL('Nuclear',3,400,1,'12/04/2012','energia perillosa',sortida);
GESTION_TIPO_CENTRAL.PRC_ALTA_TIPO_CENTRAL('Carbó',1,500,1,'12/04/2012','energia bruta
contaminant',sortida);
GESTION_TIPO_CENTRAL.PRC_ALTA_TIPO_CENTRAL('Termica',1,50,1,'12/04/2012','energia bruta
contaminant',sortida);
GESTION_TIPO_CENTRAL.PRC_ALTA_TIPO_CENTRAL('Solar',4,1500,1,'12/04/2012','energia neta',sortida);
END;
/
--CONSULTA
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_TIPO_CENTRAL.PRC_CONSULTAR_TIPO_CENTRAL(sortida);
END;
-- EXECUTA INSERTAR CLASSE CENTRAL
/***********************************************
GESTION_CLASSE_CENTRAL.PRC_MODIFICA_CLASSE_CENTRAL(
   P_DESCRIPCIOCLASSE => P_DESCRIPCIOCLASSE,
   P_IDCLASSE => P_IDCLASSE,
  sortida
 );
 ***********************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_CLASSE_CENTRAL.PRC_ALTA_CLASSE_CENTRAL('Producció',sortida);
GESTION_CLASSE_CENTRAL.PRC_ALTA_CLASSE_CENTRAL('Distribució',sortida);
END;
/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_CLASSE_CENTRAL.PRC_CONSULTAR_CLASSE_CENTRAL(sortida);
END;
/

-- EXECUTA INSERTAR VIA
/*****************************************
 GESTION_VIA.PRC_ALTA_VIA(
   P_DESCRIPCIOVIA => P_DESCRIPCIOVIA,
   sortida
 );
 ***********************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
```

```
GESTION_VIA.PRC_ALTA_VIA('Avda._1',sortida);
GESTION_VIA.PRC_ALTA_VIA('C./',sortida);
GESTION_VIA.PRC_ALTA_VIA('Pge.',sortida);
GESTION_VIA.PRC_ALTA_VIA('Pol. Ind.',sortida);
GESTION_VIA.PRC_ALTA_VIA('Ctra.',sortida);
GESTION_VIA.PRC_ALTA_VIA('Passeig',sortida);
GESTION_VIA.PRC_ALTA_VIA('Cami',sortida);
GESTION_VIA.PRC_ALTA_VIA('Travessera',sortida);
GESTION_VIA.PRC_ALTA_VIA('Rbla.',sortida);
END;
/
--CONSULTA
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_VIA.PRC_CONSULTA_VIA(sortida);
END;
/
-- INSERT DELS TIPUS DE LINEA
/************************************************
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS(
   P_DATAESTATTIPOLINEA => P_DATAESTATTIPOLINEA,
   P_CONSUMMAXIMTIPOLINEA => P_CONSUMMAXIMTIPOLINEA,
   P_OBSERVACIONTIPOLINEA => P_OBSERVACIONTIPOLINEA,
   sortida
 );
************************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',500,'Fluxe subterrani',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/01/2012',1000,'Fluxe aeri',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/02/2012',1500,'Fluxe hibrit',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/03/2012',2000,'Fluxe Industrial Hibri',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/04/2012',2500,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/05/2012',3000,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',3500,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',4000,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',4500,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',5000,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',5500,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',6000,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',6500,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',7000,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',7500,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',8000,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',8500,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',9000,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',9500,'Alta Tensió',sortida);
GESTION_LINEA_TIPUS.PRC_ALTA_LINEA_TIPUS('15/12/2011',10000,'Alta Tensió',sortida);
END;
/
```

```
--CONSULTA NO ACTIVADES
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_LINEA_TIPUS.PRC_CONSULTAR_LINEA_NO_ACTIVA(sortida);
END;
/
--CONSULTA ACTIVADES
DECLARE
sortida VARCHAR(500):='';
BEGIN
  GESTION_LINEA_TIPUS.PRC_CONSULTAR_LINEA_ACTIVA(sortida);
END;
/
--  EXECUTA INSERTAR PAIS
/*******************************************
GESTION_PAIS.PRC_ALTA_PAIS(
    P_DESCRIPCIOPAIS => P_DESCRIPCIOPAIS,
    P_CODIPAIS => P_CODIPAIS,
    sortida
  );
***********************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_PAIS.PRC_ALTA_PAIS('ESPAÑA','ES',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('FRANÇA','FR',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('MARROC','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ALEMANIA','DE',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('DINAMARCA','DK',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('HOLANADA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('BELGICA','BE',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ITALIA','IT',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('GRAN BRETAÑA','GB',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('IRLANDA','IE',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ANDORRA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('SANT MARINO','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('LUXEMBURGO','LU',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ECUADOR','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('MEXICO','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ESTADOS UNIDOS','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('CANADA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('VENEZUELA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('COLOMBIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('TUNEZ','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ARGELIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('EGIPTO','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('LIBIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('LIBANO','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ISRAEL','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ETIOPÍA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('SUDAFRICA','.',sortida);
```

```
GESTION_PAIS.PRC_ALTA_PAIS('AUSTRLIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('SUIZA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('AUSTRIA','AT',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('RUSIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('LETONIA','LV',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ESTONIA','EE',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('LITUANIA','LT',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('FILANDIA','FI',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('NORUEGA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('SUECIA','SE',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ISLANDIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('POLONIA','PL',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('GRECIA','GR',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('TURQUIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('HUNGRIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('RUMANÍA','RO',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('VATICANO','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('MALTA','MT',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ARABIA SAUDÍ','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('INDIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('VIETMAN DEL NORTE','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('COREA DEL SUR','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('JAPON','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('CHINA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('PUERTO RICO','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('CUBA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('GUATEMALA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ARGENTINA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('BRASIL','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('GUAYANA FRANCESA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('PORTUGAL','PT',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ALBANIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('MONTENEGRO','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('SERBIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('PAKISTAN','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('VIETNAM DEL SUR','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('MONGOLIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('YEMEN','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('MAURITANIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('BOLIVIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('CHILE','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('PANAMA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('QATAR','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('OMAN','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('EMIRATOS ARBES','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('IRAN','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('IRAK','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('REPUBLICA DOMINICANA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('BELICE','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('CROACIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('URUGUAY','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('PARAGUAY','.',sortida);
```

```
GESTION_PAIS.PRC_ALTA_PAIS('HAITÍ','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('COREA DEL NORTE','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('TAHILANDIA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('NEVA ZELANDA','.',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('CHIPRE','CY',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('ESLOVENIA','SI',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('REPUBLICA CHECA','CZ',sortida);
GESTION_PAIS.PRC_ALTA_PAIS('PAISES BAJOS','NL',sortida);
END;
/
-- CONSULTA PAIS
DECLARE
sortida VARCHAR(500):='';
BEGIN
 GESTION_PAIS.PRC_CONSULTAR_PAIS(sortida);
END;
/
-- EXECUTA INSERTAR PROVINCIA
/***********************************************
 GESTION_PROVINCIA.PRC_ALTA_PROVINCIA(
   P_DESCRIPCIOPROVINCIA => P_DESCRIPCIOPROVINCIA,
   P_IDPAISPROVINCIA => P_IDPAISPROVINCIA,
   sortida
  );
***********************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LA MASSANA',11,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BARCELONA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VALENCIA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALICANTE',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GIRONA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LLEIDA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TARRAGONA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MADRID',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TERUEL',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SEGOVIA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ZARAGOZA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HUESCA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CUENCA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GUADALAJARA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LEON',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VALLADOLID',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SALAMANCA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PAMPLONA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BILBAO',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SAN SEBASTIAN',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LA CORUÑA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ORENSE',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PONTEVEDRA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LUGO',1,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OVIEDO',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GUIPUZCOA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BURGOS',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LA RIOJA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AVILA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TOLEDO',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BADAJOZ',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALBACETE',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MURCIA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CORDOBA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SEVILLA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('JAEN',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HUELVA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CADIZ',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DONOSTIA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CEUTA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MELILLA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GRANADA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SANTA CRUZ DE TENERIFE',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GRAN CANARIAS',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ISLAS BALEARES',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CASTELLÓ',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AIN',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SORIA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GREATER CASABLANCA',3,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AISNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALLIER',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALPES DE HAUTE PROVENCE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUTES ALPES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALPES MARITIMES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARDÈCHE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARDENNES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARIÈGE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AUBE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AUDE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AVEYRON',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BOUCHES DU RHÔNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CALVADOS',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CANTAL',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CHARENTE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CHARENTE MARITIME',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CHER',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CORRÈZE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CORSE DU SUD',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUTE CORSE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BAVARIA',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HESSE',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOWER SAXONY',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('WETSPHALIA',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PALATINATE',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BADEN WÜRTTEMBERG',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TURIGIAN',4,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SAXONY ANHALT',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SAXONY',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('POMERANIA',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BRANDENGURG',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SAARLAND',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HOLSTEIN',4,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VESZEPREN',42,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BARI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LES ESCALDES',11,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PESCARA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TERAMO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BARLETTA-ANDRIA-TRANI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BRINDISI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FOGGIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TARENTO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MATERA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LECCE',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('POTENZA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CATANZARO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CHIETI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('COSENZA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BENEVENTO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VIBO VALENTIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AVELLINO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('REGGIO CALABRIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CASERTA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CÔTE D´ARMOR',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CREUSE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TATABÁNYA',42,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DORDOGNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DOUBS',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DRÔME',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NORTH HOLLAND',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SOUTH HOLLAND',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GELDERLAND',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FLEVO-LAND',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FRIESLAND',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GRONINGEN',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OVERIJSSEL',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BRABANT',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ZEELAND',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('UTRECH',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LIMBURG',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DRENTHE',6,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AMBERES',7,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LIMBURGO',7,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BRABANTE FLAMENCO',7,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FLANDES ORIENTAL',7,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FLANDES OCCIDENTAL',7,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ÁTICA',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GRECIA CENTRAL',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MACEDONIA CENTRAL',40,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GRETA',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('EPIRO',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ISLAS JÓNICAS',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PELOPONESO',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('EGEO MERIDIONAL',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TESALIA',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GRECIA OCCIDENTAL',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MACEDONIA OCCIDENTAL',40,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('POMERANIA OCCIDENTAL',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('POMERANIA ORIENTAL',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LEBUS',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BAJA SILESIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OPOLE',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SILESIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GRAN POLONIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CUYAVIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LODZ',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VARMIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MASURIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MAZOVIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PODLAQUIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SANTA CRUZ',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LUBLIN',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PEQUEÑA POLONIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SUBCARPACIA',39,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALGAVRE',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALTO ALENTEJO',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BAIXO ALENTEJO',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BEIRA ALTA',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BEIRA BAIXA',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BEIRA LITORAL',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DOURO LITORAL',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ESTREMADURA',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MINHO',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RIBATEJO',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TRÀS-SO-MONTES',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALTO DOURO',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MADEIRA',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AZORES',58,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BURGENLAND',30,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CARINTINA',30,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BAJA AUSTRIA',30,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SALZBURG',30,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ESTIRIA',30,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TIROL',30,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VORALBERG',30,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VIENA',30,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ZURICH',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BERNA',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LUCERNA',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('URI',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SCHWYZ',29,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OBWALDEN',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NIDWALDEN',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GLARIS',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ZUG',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FRIBURG',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SOLEURA',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BASILEA CIUDAD',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BASILEA CAMPIÑA',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SCHAFFAUSEN',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('APPENZELL RODAS EXTER.',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('APPENZELL RODAS INTER.',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SAN GALO',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GRISONES',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARGOVIA',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TURGOVIA',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TESINO',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VAUD',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VALAIS',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NEUCHÂTEL',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GINEBRA',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('JURA SUIZA',29,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TEXAS',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NEBRASKA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LA VALL D´ARAN',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('EURE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('EURE ET LOIR',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FINISTÈRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GARD',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUTE GARONNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GERS',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GIRONDE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HÉRAULT',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ILLE ET VILAINE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('INDRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('INDRE ET LOIRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ISÈRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('JURA ',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LANDES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOIR ATLANTIQUE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOIRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUTE LOIRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOIRE ET CHER',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOT',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOT ET GARONNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOZÈRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MAIN ET LOIRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MANCHE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MARNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUTE MARNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MAYENNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MEURTHE ET MOSELLE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MEUSE',2,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MORBIHAN',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MOSELLE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NIÈVRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NORD',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OISE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ORNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PAS DE CALAIS',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PUY DE DÔME',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PYRÉNÉES ATLANTIQUES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUTES PYRÉNÉES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PYRÉNÉES ORIENTALES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BAS RHIN',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUT RHIN',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RHÔNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUTE SAONE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SAONE EY LOIRE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SARTHE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PARÍS',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SEINE MARITIME',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SEINE ET MARME',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('YVELINES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DEUX SÈVRES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SOMME',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TARN',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TARN ET GARONNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VAR',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VAUCLUSE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VENDÉE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VIENNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HUTE VIENNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VOSGES',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('YONNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TERRITORE DE BELFORT',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ESSONNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUTS DE SIENE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SEINE SANT DENIS',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VAL DE MARNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VAL D´OISE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOIRET',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SAVOIE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAUTE VIENNE',2,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NÁPOLES',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SALERMO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CALIGARI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CARBONIA-IGLESIAS',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MEDIO CAMPIDANO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NUORO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OGLIASTRA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OLBIA-TEMPIO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ORISTANO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SASSARI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BOLONIA',8,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FERRARA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FORLÌ-CESENA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MÓDENA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PARMA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PIACENZA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RÁVENA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('REGGIO EMILIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RÍMINI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GORIZIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PORDENONE',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TRIESTE',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('UDINE',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FROSIONE',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LATINA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RIETI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ROMA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VITERBO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GÉNOVA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('IMPERIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LA SPEZIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SAVONA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BÉRGAMO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BRESCIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('COMO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CREMONA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LECCO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LODI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MANTUA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MILÁN',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MONZA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PAVÍA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SONDRIO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VARESE',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ANCONA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ASCOLI PICENO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FERMO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MACERATA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PESARO Y URBINO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CAMPOBASSO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ISERNIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALESSANDRIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ASTI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BIELLA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CUNEO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NOVARA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TURÍN',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VERBANO-CUSIO-OSSOLA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VERCELLI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AGRIGENTO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CALTANISSETTA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CATANIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ENNA',8,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MESSINA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PALERMO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RAGUSA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SIRACUSA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TRAPANI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AREZZO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FLORENCIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GROSSETO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LIVORNO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LUCCA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MASA-CARRARA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PISA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PISTOIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PRATO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SIENA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BOLZANO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TRENTO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PERUGIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TERNI',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AOSTA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BELLUNO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PADUA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ROVIGO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TREVISO',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VENECIA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VERONA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VICENZA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('L´AQUILA',8,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CIUDAD REAL',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PLASENCIA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALMERÍA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MÁLAGA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ZAMORA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PALENCIA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CACERES',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NAVARRA',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ASTURIAS',1,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BÁCS-KISKUN',42,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALABAMA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALASKA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARIZONA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARKANSAS',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CALIFORNIA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CAROLINA DEL NORTE',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CAROLINA DEL SUR',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CHICAGO',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('COLUMBIA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONNECTICUT',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('COROLADO',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DAKOTA DEL NORTE',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DAKOTA DEL SUR',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DELAWARE',16,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FLORIDA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GEORGIA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAWAI',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('IDAHO',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ILLINOIS',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('INDIANA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('IOWA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('KANSAS',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('KENTUCKY',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LUISIANA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MAINE',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MARYNELAN',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MASSACHUSETTS',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MICHIGAN',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MINESOTA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MISISIPI',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MISURI',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MONTANA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('UTAH',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NEVADA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NEW YORK',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NUEVA JERSEY',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NUEVO HAMPSHIRE',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NUEVO MEXICO',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OHIO',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OKLAHOMA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OREGON',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PENSILVANIA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RHODE ISLAND',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TENNESSE',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VERMONT',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VIRGINIA',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VIRGINIA OCCIDENTAL',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('WASHINTONG D.C.',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('WISCONSIN',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('WYOMING',16,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ALBERTA',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('COLUMBIA BRITANICA',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ISLA DEL PRINCIPE EDUARDO',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MANITOBA',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NUEVA ESCOCIA',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NUEVO BRUNSWICK',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NUNAVUT',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ONTARIA',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('QUEBEC',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SASKATCHEWAN',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TERRANOVA Y LABRADOR',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TERRITORIOS DEL NOROESTE',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('YUKON',17,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BAJA CALIFORNIA',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BAJA CALIFORNIA DEL SUR',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CAMPECHE',15,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CHIAPAS',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CHIHUAHUA',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('COAHUILA DE ZARAGOZA',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('COLIMA',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DURANGO',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GUANAJUATO',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GUERRERO',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HIDALGO',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('JALISCO',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MEXICO',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MICHOACAN DE OCAMPO',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MORELOS',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NAYARIT',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NUEVO LEON',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OXACA',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PUEBLA',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('QUERETARO',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('QUINTANA ROO',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SANT LUIS POTOSI',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SINALOA',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SONORA',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VERACRUZ IGNACIO DE LA LLAVE',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('YUCATAN',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ZACATECAS',15,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARGYLL Y BUTE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AYRSHIRE Y ARRAN',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BANFFSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BERKS',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BERWICKSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BUCKINGHAM',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CAITHENESS',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CHESHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CLACKMANNAHSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CLWYD',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE AVERDEEN',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE ANTRIM',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE ARMAGH',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE BERFORD',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE BRISTOL',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE CAMBRIGE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE DUNDEE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE DURHAM',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE EDIMBURGO',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE GLAGOW',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE GLOUCESTER',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE HEREFORD',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE INVERNESS',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE LONDONDERRY',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE NAIRN',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE NORTHAMPTON',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE NOTTINGHAM',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE OXFORD',9,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE STAFFORD',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE WARWICK',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE WIGTOWN',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE WORCESTER',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DUNFRIES',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CORNUALLES E ISLAS SCILLY',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CUMBRIA',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DERBY',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DEVON',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DORSET',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DOWN',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DUNBARTONSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DYFED',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('EIFE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ESSEX',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ETTRICK Y LAUDERDALE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FEMANAGH',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GLAMORGAN CENTRAL',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GLAMORGAN OESTE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GLAMORGAN SUR',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GWENT',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GWYNEDD',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HAMPSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HEBRIDAS EXTERIORES',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ISLA SHETLAND',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ISLA WIGHT',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ISLAS ORKNEY',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('KENT',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('KINCARDINESHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('KIRKCUDBRIHHTSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LANARKSSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LANCASHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LEICESTER',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LINCOLNSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LONDRES',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOTHIAN CENTRAL',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOTHIAN ESTE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LOTHIAN OESTE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MANCHESTER',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MERSEYSIDE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MIDLANDS OCCIDENTALES',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MORAY',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NORFOLK',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NORTHUMBERLAND',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('PERTH Y KINROSS',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('POWYS',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('REFREWSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ROSS Y CROMARTY',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ROXBURG',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RUTLAND',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SHORPSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SOMERSET',9,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('STIRLING Y FALKIRK',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SUFFOLK',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SURREY',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SUSSEX OCCIDENTAL',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SUSSEX ORIENTAL',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SUTHERLAND',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TWEEDDALE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TYNE Y WEAR',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TYRONE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('WILTSHIRE',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('YORKSHIRE MERIDIONAL',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('YORKSHIRE OCCIDENTAL',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('YORKSHIRE ORIENTAL',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('YORKSHIRE SEPTENTRIONAL',9,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BORNHLM',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONDADO DE COPENAGUE',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FREDERIKSBERG',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('COPENAGE COMUNA',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARHUS',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FIONIA',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NÓGRÁD',42,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FREDEIKSBORG',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('JUTLANDIA MERIDIONAL',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('JUTLANDIA SEPTENTRIONAL',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RIBE',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('RINGKPBING',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ROSKILDE',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SELANDIA OCCIDENTAL',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('STORSTROM',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VEJLE',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VIBORG',5,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARRONDISSENMENT  DE ARLON',13,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARRONDISSENMENT DE BASTOGNE',13,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARRONDISSENMENT  DE MARCHE-EN-FEMENNE',13,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARRONDISSENMENT  DE NEUFCHÂTEAU',13,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ARRONDISSENMENT DE VIRTON',13,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NORRBOTTEN',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VÄSTERBOTTEN',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('JÄMTLAND',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VÄSTERNORRLAND',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GÄVLEBORG',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('DALARNA',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VÄRMLAND',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ÖREBRO',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VÄSTMANLAND',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('UPPSALA',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ESTOCOLMO',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SÖDERMANLAND',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('VÄSTRA GÖTALAND',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ÖSTERGÖTLAND',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('JÖNKÖPING',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('KALMAR',37,sortida);
```

```
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HALLAND',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('KRONOBERG',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BLEKINGE',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ESCANIA',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('GOTLAND',37,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('LEINSTER',10,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MUNSTER',10,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('CONNACHT',10,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ULSTER',10,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FINNMARK',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NORDLAND',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TROMS',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OPPLAND',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NORD-TRONDELAG',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SOR-TRONDELAG',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SOGN OG FJORDANE',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('HORDALAND',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('TELEMARK',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MORE OG ROMSDAL',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('BUSKERUD',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ROGALAND',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AUST_AGDER',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('AKERSHUS',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OSTFOLD',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OSLO',36,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('FIORDOS OCCIDENTALES',38,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('MERIDIONAL',38,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NOROCCIDENTAL',38,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('NORORIENTAL',38,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('OCCIDENTAL',38,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('ORIENTAL',38,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('REIKIAVIK',38,sortida);
GESTION_PROVINCIA.PRC_ALTA_PROVINCIA('SANT JULIA DE LORIA',11,sortida);
END;
/
-- CONSULTA PROVINCIA
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_PROVINCIA.PRC_CONSULTA_UNA_PROVINCIA('MADRID',sortida);
GESTION_PROVINCIA.PRC_CONSULTA_UNA_PROVINCIA('BARCELONA',sortida);
GESTION_PROVINCIA.PRC_CONSULTA_UNA_PROVINCIA('VALENCIA',sortida);
GESTION_PROVINCIA.PRC_CONSULTA_UNA_PROVINCIA('ZARAGOZA',sortida);
GESTION_PROVINCIA.PRC_CONSULTA_UNA_PROVINCIA('HUESCA',sortida);
END;
/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_PROVINCIA.PRC_CONSULTA_PROVINCIA(sortida);
END;
/
```

```
-- EXECUTA INSERTAR LOCALITATS
/*********************************************
 GESTION_LOCALITAT.PRC_ALTA_LOCALITAT(
   P_DESCRIPCIOLOCALITAT => P_DESCRIPCIOLOCALITAT,
   P_IDPROVINCIALOCALITAT => P_IDPROVINCIALOCALITAT,
   sortida
  );
*********************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOURG-EN-BRESSE',47,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LAON',50,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALBACETE',32,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALMANSA',32,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAUDETE',32,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COTILLAS',32,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HELLÍN',32,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MADRIGUERAS',32,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAROBLEDO',32,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OPORTO',154,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ELX',4,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CREVILLENTE',4,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BAÑERES DE MARIOLA',4,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOILINES (ALLIER)',51,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ADRA',367,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BÉJAR',367,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL EJIDO',367,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PURCHENA',367,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VELEZ RUBIO',367,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SERON',367,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DIGNE-LES-BAINS',52,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NIZA',54,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PRIVAS',55,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHARLEVILLE-MÉZIÈRES',56,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FOIX',57,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALLANDE',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALLER',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AMIEVA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AVILES',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BELMONTE DE MIRANDA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BIMENES',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOAL',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABRALES',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABRANES',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CANDAMO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CANGAS DE ONIS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CANGAS DEL NARCEA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARAVIA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARREÑO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTRILLON',373,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTROPOL',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COAÑA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COLUNGA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORVERA DE ASTURIAS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CUDILLERO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DEGAÑA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL FRANCO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GIJON',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GOZON',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GRADO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GRANDAS DE SALIME',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('IBIAS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ILLANO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ILLAS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LANGREO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LAVIANA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LENA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LLANERA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LLANES',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MIERES',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MORCIN',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MUROS DE NALON',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVIA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NOREÑA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ONIS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OVIEDO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PARRES',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PEÑAMELLERA ALTA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PEÑAMELLEDA BAJA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PESOZ',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PONGA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PRAVIA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PROAZA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('QUIROS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LAS REGUERAS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBADEDEVA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBADESELLA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBERA DE ARRIBA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIOSA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALAS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN MARTIN DE OSCOS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN MARTIN DEL REY AURELIO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANN TIRSO DE ARBRES',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTA EULAIA DE OSCOS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTO ADRIANO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SARIEGO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SIERO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SOBRESCOBIO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SOMIEDO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SOTO DE BARCO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TAPIA DE CASARIEGO',373,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TARAMUNDI',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TAVERGA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TINEO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALDES',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VEGADEO',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLANUEVA DE OSCOS',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAVIVIOSA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAYON',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('YERMES Y TAMEZA',373,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TROYES',58,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARCASONA',59,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RODEZ',60,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AVILA',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARENAS DE SAN PEDRO',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABEZAS DE POZO',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CANALES',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARDEÑOSA',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CONSTANZA',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FRESNEDILLA',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LOS LLANOS DE TORMES',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MIJARES',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVALMORAL',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PIEDRAHITA',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN JUAN DE GREDOS',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('UMBRIAS',29,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BADAJOZ',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ACEUCHAL',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALANGUE',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALBURQUERQUE',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCONCHEL',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALMENDRAL',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALMENDRALEJO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARROYO DE SAN SERVAN',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AZUAGA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BARCARROTA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BERLANGA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BIENVENIDA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BODONAL DE LA SIERRA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BURGUILLOS DE LA SIERRA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BURGUILLOS DEL CERRO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABEZA DE BUEY',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALERA DE LEON',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMPANARIO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMPILLO DE LLERENA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASAS DE DON PEDRO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTILBLANCO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTUERA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA CODOSERA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORTE DE PELEAS',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORDOBILLA DE LACARA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA CORONADA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHELES',31,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DON BENITO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESPARRALEJO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESPARRAGOSA DE LA SERENA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESPARRAGOSA DE LARES',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FERIA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FUENTE CANTOS',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA GARROVILLA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GRANJA DE TORREHERMOSA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GUAREÑA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA HABA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HERRERA DEL DUQUE',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HORNACHOS',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LOBON',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('JEREZ DE LOS CABALLEROS',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LLERENA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MAGUILLA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MEDELLIN',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MERIDA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONESTERIO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTIJO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVALVILLAR DE PELA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLIVA DE LA FRONTERA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLIVA DE MERIDA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLIVENZA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ORELLANA LA VIEJA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PUEBLA DE LA CALZADA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PUEBLA DE OBANDO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PUEBLA DE SANCHO PEREZ',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PUEBLONUEVO DE GUADIANA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('QUINTANA DE LA SERENA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBERA DE FRESNO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA ROCA DE LA SIERRA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALVALEON',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALVATIERRA DE LOS BARROS',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN VICENTE DE ALCANTARA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTA AMALIA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTA MARTA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LOS SANTOS DE MAIMONA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SIRUELA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SOLANA DE LOS BARROS',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TALAVERA LA REAL',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORREMEJIA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('USAGRE',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALDECABALLEROS',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALDELACALZADA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALENCIA DEL VENTOSO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALVERDE DE LEGANES',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALVERDE DE MERIDA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAFRANCA DE BARROS',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLANUEVA DE LA SERENA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLANUEVA DEL FRESNO',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAR DEL REY',31,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZAFRA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZAHINOS',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZLAMEA LA SERENA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA ZARZA',31,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BARCELONA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TERRASSA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SABADELL',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MANRESA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BERGA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTPEDOR',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RUBI',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MATADEPERA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT FRUITOS DE BAGES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTELLAR DEL VALLES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VACARISES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LES FONT DE TERRASSA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT CUGAT DEL VALLES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOLLET DEL VALLES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PREMIÀ DE MAR',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTGAT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT QUIRZE DEL VALLÈS',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLESA DE MONTSERRAT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONISTROL DE MONTSERRAT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONISTROL DE CALDERS',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT JOAN DE VILATORRADA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MATARO',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT CELONI',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PUIGCERDA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT ANDREU DE LA BARCA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARTORELL',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('L´HOSPITALE DE LLOBREGAT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT HILARI',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BADALONA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTA COLOMA DE GRAMANET',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORNELLA DE LLOBREGAT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT BOI DE LLOBREGAT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILANOVA I LA GELTRU',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILADECANS',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GRANOLLERS',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTELLDEFELS',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CERDANYOLA DEL VALLES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESPLUGAS DE LLOBREGAT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN SADURNI D´ANOIA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('IGUALADA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARTORELLES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASSRRES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT VICEÇ DE CASTELLET',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARTES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FONOLLOSA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT ADRIA DEL BESOS',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT RAMON',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT MIQUEL DEL FAI',2,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA FLORESTA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTELLBELL I EL VILAR',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ROCAFORT DE BAGES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PONT DE VILOMARA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOIA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RAJADELL',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTELLNOU',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GIRONELLA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('L´ESQUIROL',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTJUST DESVERN',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT PERE DE VILAMAJOR',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALLUS',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA COMA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALLENT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTELLARNAU',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RELLINAS',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ULLASTRELL',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILADECABALLS',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT QUIM',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BEGUDA ALTA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BEGUDA BAIXA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL PRAT DE LLOBREGAT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT MIQUEL DE CONTERES',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SETMENAT',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALDES DE MONTBUI',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLIVELLA',2,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BITONTO',84,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALTAMURA',84,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONOPOLI',84,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NOCI',84,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TURI',84,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALENZANO',84,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESTRASBURGO',244,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BERGAMO',306,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOLONIA',284,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANZOLA DELL´EMILIA',284,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARGELATO',284,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BUDRIO',284,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTEL GUELFO DI BOLONIA',284,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DOZZA',284,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTE SAN PRIETO',284,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VERGATO',284,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARSELLA',61,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BURGOS',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARANDA DE DUERO',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BELORADO',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BRIVIESCA',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COMTAT DE TREVIÑO',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESPINOSA DE LOS MONTEROS',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HUERTA DEL REY',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LERMA',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MEDINA DEL POMAR',27,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MIRANDA DE EBRO',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OÑA',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('QUINTANAR DE LA SIERRA',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ROA',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALAS DE LOS INFANTES',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SASAMON',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALLE DE MENA',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLADIEGO',27,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AHIGAL',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCUESCAR',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALDEANUEVA DE LA VERA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALISEDA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARROYO DE LA LUZ',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BROZAS',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABEZUELA DEL VALLE',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CACERES',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASAR DE CACERES',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CECLAVIN',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CILLEROS',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORIA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALISTEO',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GARROVILLAS DE ALCONETAR',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GUADALUPE',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HERVAS',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('JARANDILLA DE LA VERA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LOGROSAN',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LOSAR DE LA VERA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MADRILEJO',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MADROÑERA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MALPARTIDA DE CACERES',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MALPARTIDA DE PLASENCIA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MIAJADAS',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTANCHEZ',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTEHERMOSO',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MORALEJA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVAMORAL DE LA MATA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PLASENCIA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TALAYUELA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TRUJILLO',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALVERDE DE FRESNO',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZORITA',371,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CADIZ',38,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('JEREZ DE LA FRONTERA',38,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALGECIRAS',38,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL PUERTO DE SANTA MARIA',38,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAEN',62,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AURILLAC',63,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VINAROZ',46,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MORELLA',46,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALBI',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOTRICELLO',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BORGIA',95,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DAVOLI',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOTEPAONE',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PENTONE',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PLATANIA',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN FLORO',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TAVERNA',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TIRIOLO',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALLEFIORITA',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZAGARISE',95,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANGULEMA',64,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA ROCHELA',65,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOURGES',66,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHIETI',96,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FRANCA VILLA AL MARE',96,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BUCCHIANICO',96,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIPA TATIANA',96,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCAZAR DE SAN JUAN',365,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALMADEN',365,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARENAS DE SAN JUAN',365,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DAIMIEL',365,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MANZANARES',365,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MENBRILLA',365,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PUERTOLLANO',365,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TOMELLOSO',365,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAHERMOSA',365,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COMO',308,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORDOBA',34,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BAENA',34,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABRA',34,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL VISO',34,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LUCENA',34,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTILLA',34,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MORILES',34,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PUENTE GENIL',34,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAVICIOSA',34,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TULLE',67,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AJACCIO',68,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DIJON',103,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GUÉRET',104,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CUENCA',13,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NIORT',254,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PÉRIGUEUX',106,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BESANÇON',107,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALENCE',108,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ÉVRY',266,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LISBOA',161,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ÉVREUX',205,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHARTRES',206,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FERMO',320,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FERRARA',285,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('QUIMPER',207,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FLORENCIA',343,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FROSINONE',297,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NIMES',208,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GENOVA',302,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARENZANO',302,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BARGAGLI',302,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMOGLI',302,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DAVAGNA',302,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AUCH',210,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT ANTONI DE CALONGE',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT FELIU DE GUIXOLS',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PALAMOS',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PALS',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GIRONA',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBAS DE FRESER',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BLANES',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT ANONI DE COLONGE',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FIGUERES',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LLORET DE MAR',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLOT',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALT',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PALAFRUGELL',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ROSES',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BANYOLES',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTELLO D´EMPURIES',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTA COLOMA DE FARNES',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORROELLA DE MONTGRI',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIPOLL',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA BISBAL D´EMPORDA',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALONGE',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTELL-PLATJA D´ARO',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASSA DE LA SELVA',5,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BURDEOS',211,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LAS PALMAS',44,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TELDE',44,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALMUÑECAR',42,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BAZA',42,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GRANADA',42,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GUADIX',42,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LANJARÓN',42,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOTRIL',42,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NÍJAR',42,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TREVÉLEZ',42,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASABLANCA',49,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALGORA',14,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ATIENZA',14,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CIFUENTES',14,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HITA',14,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LUZÓN',14,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOLINA DE ARAGON',14,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PASTRANA',14,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PIQUERAS',14,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SIGÜENZA',14,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('UCEDA',14,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COLMAR',245,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BASTIA',69,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TOULOSE',209,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LE PUY-EN VELAY',221,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHAUMONT',229,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VESOUL',247,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANNECY',247,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LIMOGES',273,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GAP',53,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TARBES',242,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NANTERRE',267,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTPELLIER',212,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HUELVA',37,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AYAMONTE',37,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMPOFRÍO',37,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CUMBRES MAYORES',37,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('JABUGO',37,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANLUCAR DEL GUADIANA',37,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TRIGUEROS',37,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALVERDE DEL CAMINO',37,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTSO',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BARBASTRO',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FRAGA',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('JACA',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SABIÑANIGO',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BINEFAR',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SARIÑENA',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TAMARIT DE LA LLITERA',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GRAUS',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AINSA',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALABALATE DEL CINCA',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCOLEA DEL CINCA',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALMUDEVAR',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL TORRICO',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AYERDE',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BELLVER DE CINCA',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BENAVARRI',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BENASQUE',12,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RENNES',213,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHÂTEAUROUX',214,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GRENOBLE',216,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PALMA DE MALLORCA',45,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCALA LA REAL',36,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('JAEN',36,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANDÚJAR',36,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BAEZA',36,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BAILEN',36,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA CAROLINA',36,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LINARES',36,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ÚBEDA',36,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALDEPEÑAS',36,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LONS-LE-DAUNIER',217,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA CORUÑA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ABEGONDO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AMES',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARANGA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARES',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARTEIXO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARZUA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('A BAÑA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BERGONDO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BETANZOS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOIRO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOIMORTO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOQUEIXON',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('REION',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABANA DE BERGANTIÑOS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABANAS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMARIÑAS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMBRE',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('A CAPELA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARBALLO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARIÑO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARNOTA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARRAL',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CEDEIRA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CEE',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CERCEDA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CERDIDO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CESURAS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COIROS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORCUBION',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORISTANCO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CULLEREDO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CURTIS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DODRO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DUMBRIA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FENE',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FERROL',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FISTERRA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FRADES',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('IRIXOA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('A LARACHA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LAXE',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LOSAME',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MALPICA DE BERGANTIÑS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MAÑON',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MAZARICOS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MELIDE',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MESÍA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MIÑO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOECHE',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONFERO',21,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MUGARDOS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MUXIA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NARON',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NEDA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NEGREIRA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NOIA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLEIROS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ORDES',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OROSO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ORTIGUEIRA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OUTES',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OZA DOS RIOS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PADERNE',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PADRON',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('O PINO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('A POBRA DO CARAMIÑAL',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PONTECESO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PONTEDEUME',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AS PONTES DE CARGIA RODRIGUEZ',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PORTO DO SON',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBEIRA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ROIS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SADA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN SADURNIÑO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTIAGO DE COMPOSTELA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTISO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SOBRADO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AS SOMOZAS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TEO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TOQUES',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORDOIA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TOURO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TRAZO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VAL DO DUBRA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALDOVIÑO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VEDRA',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILAMIAOR',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VIMIANZO',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZAS',21,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARNEDO',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALAHORRA',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALFARO',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARNEDILLO',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOBADILLA',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HARO',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HERCE',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LARDERO',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LEIVA',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LUMBRERAS',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MEDRANO',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAJERA',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OCON',28,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PINILLOS',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PADREJON',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PRADILLO',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('QUEL',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RABANERA',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN ASENSIO',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTO DOMINGO DE LA CALZADA',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAREJO',28,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONT-DE-MARSAN',218,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('L´AQUILA',364,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PRATA',364,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MAGLIANO DE MARSI',364,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PACENTRO',364,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OFENA',364,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OCRE',364,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLA SANT´ANGELO',364,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VITTORIO',364,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LATINA',298,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LEON',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ASTORGA',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA BAÑEZA',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BEMBIBRE',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BENAVIDES',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOÑAR',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CACABELOS',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMPONARAYA',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARRACEDELO',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARRIZO DE LA RIBERA',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CISTIERNA',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CONGOSTO',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORULLON',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CUBILLOS DEL SIL',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHOZAS DE ABAJO',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FABERO',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PONFERRADA',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA ROBLA',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN ANDRES DEL RABANEDO',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN JUSTO DE LA VEGA',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTA MARIA DEL PARAMO',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTA MARINA DEL REY',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SARIEGOS',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORENO',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORRE DE BIERZO',15,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LIVORNO',345,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LLEIDA',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BALAGUER',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA SEU D´URGELL',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TARREGA',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOLLERUSSA',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CERVERA',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SOLSONA',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCARRAS',6,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LES BORGES BLANQUES',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TREMP',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GUISSONA',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALPICAT',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VIELHA',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GOSOL',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALMACELLES',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SORT',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIALP',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA GUINGUETA',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALENCIA D´ANEU',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESTERRI D´ANEU',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMARASA',6,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NANTES',219,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAINTE-ÉTIENNE',220,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BLOIS',222,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ORLEANS',271,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAHORS',223,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AGEN',224,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MENDE',225,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LUGO',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTFORTE DE LEMOS',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VIVEIRO',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILALBA',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SARRIA',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBADEO',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FOZ',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BURELA',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHANTADA',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GUITIRIZ',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CERVO',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONDOÑEDO',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AS PASTORIZA',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('O SAVIÑAO',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SOBER',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('O VALADOURO',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('XOVE',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('XERMADE',24,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVALCARNERO',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MADRID',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCOBENDAS',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCALA DE HENARES',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOSTOLES',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FUENLABRADA',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LEGANES',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCORCON',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORREJON DE ARDOZ',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GETAFE',8,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANGERS',226,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MÁLAGA',368,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANTEQUERA',368,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FUENGIROLA',368,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARBELLA',368,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MIJAS',368,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RONDA',368,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORREMOLINOS',368,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAINT-LÔ',227,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHÂLONS-EN-CHAMPAGNE',228,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LAVAL',230,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MESSINA',337,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NANCY',231,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BAR-LE-DUC',232,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MILÁN',313,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MODENA',287,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONZA',314,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VANNES',233,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('METZ',234,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CEHEGIN',33,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALASPARRAS',33,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARTAGENA',33,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MURCIA',33,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NÁPOLES',274,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORRE DEL GRECO',274,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('POZZUOLI',274,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASORIA',274,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTELLAMMARE DI STABIA',274,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AFRAGOLA',274,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARANO DI NAPOLI',274,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ACERRA',274,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PORTICI',274,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ABLITAS',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALTSASU',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALLO',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANDOSILLA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANTSOAIN',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AOIZ',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARANGUREN',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LOS ARCOS',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARGUEDAS',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARRONIZ',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARTAJONA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AYEGUI',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AZAGRA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BARAÑAIN',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VERA DE BIDASOA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BERIAIN',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BERRIOPLANO',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BERRIOZAR',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BUÑUEL',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BURLATA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABANILLAS',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CADREITA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAPARROSO',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARCAR',372,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARCASTILLO',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASCANTE',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASEDA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASTEJON',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CINTRUENIGO',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZIZUR ZENDEA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORELLA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CORTES',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DONEZTEBE',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EGÜES',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESTELLA LIZARRA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESTERIBAR',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ETXARRI-ARNATZ',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EZKABARTE',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FALCES',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FITERO',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FUNES',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FUSTIÑANA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GALAR',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('UHARTE',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('IRURTZUN',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LAKUNTZA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LARRAGA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LEITZA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LERIN',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LESAKA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LODOSA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LUMBIER',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARCILLA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MENDAVIA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MILAGRO',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MIRANDA DEARGA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONEAGUDO',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MURCHANTE',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NOAIN',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLAZAGUTIA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLZA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ORKOIEN',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PAMPLONA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PERALTA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PUENTE LA REINA (GARES)',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBAFORADA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN ADRIAN',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANGÜESA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTACARA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SARTAGUDA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SESMA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TAFALLA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TUDELA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ULTZAMA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALTIERRA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VIANA',372,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAFRANCA',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAVA (ATARRABIA)',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('YERRI',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZIZUR MAVOR',372,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NEVERS',235,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LILLE',236,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BEAUVAIS',237,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OURENSE',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('O BARCO DE VALDEORRAS',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('O CARBALLIÑO',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VERIN',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('XINZO DE LIMIA',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BARBADAS',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RUA',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CELANOVA',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBADAVIA',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALLARIZ',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MACEDA',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('POBRA DE TRIVES',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VIANA DO BOLO',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILOIRA',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SEIXALBO',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('IRINXO O',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MASIDE',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('A MERCA',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTERREI',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MUIÑOS',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OIMBRA',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PEREIRO DE AGUIAR',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RAMIRAS',22,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALENÇON',238,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PALENCIA',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AGUILAR DE CAMPOO',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALAR DE REY',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BALTANAS',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BARRUELO DE SANTULLAN',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARRION DE LOS CONDES',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CERVERA DEL PISUERGA',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DUEÑAS',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GUARDO',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HERRERA DEL PISUERGA',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PAREDES DE NAVA',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALDAÑA',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORQUEMADA',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VELILLA DEL RIO CARRION',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VENTA DE BAÑOS',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLADA',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAMURIEL DE CERRATO',370,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PALERMO',338,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PARÍS',250,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PARMA',288,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARRAS',239,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PESCARA',86,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AVEZZANO',86,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SULMONA',86,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOLOGNANO',86,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ELICE',86,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALLE',86,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VICOLI',86,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PIACENZA',289,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PISA',348,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PONTEVEDRA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AGOLADA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARBO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BAIONA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BUEU',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALDAS DE REIS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMPO LAMEIRO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CANGAS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('A CAÑIZA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CATOIRA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CERDERO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COTOBADE',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COVELO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CRECENTE',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CUNTIS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DOZON',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('A ESTRADA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FORCAREI',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FORNELOS DE MONTES',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GONDOMAR',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GROVE',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ISLA DE AROUSA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LALIN',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('A LAMA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARIN',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MEAÑO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MEIS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOAÑA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONDARIZ',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONDARIZ-BALNEARIO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MORAÑA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AS NEVES',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NIGRAN',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OIA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PAZOS CALDELAS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('POIO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PONTE CALDELAS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PONTEREAS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PORRIÑO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PORTAS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('REDONDELA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIBADUMIA',23,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ROSAL',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALCEDA DE CASELAS',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALVATERRA E MIÑO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANXENXO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SILLEDA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SOUTOMAIOR',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TOMIÑO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TUI',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALGA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VIGO',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILA DE CRUCES',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILABOA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILAGARCIA DE AROUSA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILANOVA DE AROUSA',23,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BRINDISI',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BARAGIANO',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FORENZA',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GINESTRA',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ATELLA',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BALVANO',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ABRIOLA',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ACERNZA',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANZI',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARATEA',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MISSANELLO',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MELFI',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('POTENZA',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SENISE',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TITO',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VENOSA',94,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PRATO',350,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CLERMONT-FERRAND',240,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PAU',241,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PERPIÑAN',243,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RAVENNA',290,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LYON',246,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIETI',299,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIMINI',292,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ROMA',300,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GUIDONIA MONTECELIO',300,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TÍVOLI',300,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('POMEZIA',300,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CIAMPINO',300,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARINO',300,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FIUMICINO',300,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALBA DE TORMES',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA ALBERCA',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BEJAR',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABRERIZOS',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CIUDAD RODRIGO',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GUIJUELO',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LUMBRALES',17,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PEÑARANDA DE BRACAMONTE',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTA MARTA DE TORMES',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TERRADILLOS',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLAMAYOR',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLARES DE LA REINA',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VITIGUDINO',17,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANSEBASTIAN',20,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TENERIFE',43,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MÂCON',248,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LE MANS',249,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CHAMBÉRY',272,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SEGOVIA',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AYLLON',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CANTALEJO',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CANTIMPALOS',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARBONERO MAYOR',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COCA',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CUELLAR',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL ESPINAR',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA LASTRILLA',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVA DE LA ASUNCION',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVALMANZANO',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVAS DE ORO',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PALAZUELOS DE ERESMA',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIAZA',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN CRISTOBAL DE SEGOVIA',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN IDELFONSO',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANTA MARIA LA REAL DE NIEVA',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SEPULVEDA',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TUREGANO',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLACASTIN',10,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MELUN',252,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RUAN',251,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOBIGNY',268,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SEVILLA',35,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCALA DE GUADAIRA',35,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARMONA',35,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ÉGIJA',35,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESTEPA',35,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LEBRIJA',35,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARCHENA',35,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OSUNA',35,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('UTRERA',35,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AMIENS',255,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BURGO DE OSMA',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SORIA',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AGREDA',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALMAZAN',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARCOS DE JALON',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BERLANGA DEL DUERO',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COVALEDA',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLVEGA',48,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN ESTENBAN DE GORMAZ',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAN LEONARDO DE YAGÜE',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VINUESA',48,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTAUBAN',257,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TARRAGONA',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('REUS',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORTOSA',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALLS',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL VENDRELL',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CAMBRILS',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SALOU',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALAFELL',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AMPOSTA',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILA-SECA',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT CARLES DE LA RAPITA',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORREDEMBARRA',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CUNIT',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOT-ROIG DEL CAMP',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DELTEBRE',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCANAR',7,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TERAMO',87,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BELFORT',265,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCAÑIZ',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TERUEL',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALBALATE DEL ARZOBISPO',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALBARRACIN',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALCORISA',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ANDORRA',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALAMOCHA',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALANDA',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CELLA',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESCUCHA',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('HIJAR',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MAS DE LAS MATAS',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONREAL DEL CAMPO',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTALBAN',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MORA DE RUBIOLS',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('UTRILLAS',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALL-DE-ROURES',9,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FUENSALIDA',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TOLEDO',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA GUARDIA',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ILLESCAS',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OCAÑA',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL ROMERAL',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TALAVERA DE LA REINA',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORRIJOS',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TURLEQUE',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('UGENA',30,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TRENTO',353,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TREVISO',360,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TRIESTE',295,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DUINO-AURISINA',295,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONRUPINO',295,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TURIN',330,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CRÉTEIL',269,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PONTOISE',270,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('IBI',3,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PUZOL',3,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SAGUNTO',3,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VALLADOLID',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SIMANCAS',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORDESILLA',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALAEJOS',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALDEAMAYOR DE SAN MARTIN',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARROYO DE LA ENCOMIENDA',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOECILLO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CABEZON DE PISUERGA',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARPIO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('COSTRONUÑO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CIGALES',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CISTÉRNIGA',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FRESNO VIEJO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FUENSALDAÑA',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ISCAR',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LAGUNA DE DUERO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MAYORGA',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MEDINA DE RIOSECO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MEDINA DEL CAMPO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MOJADOS',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MONTEMAYOR DE PILILLA',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NAVA DEL REY',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('OLMEDO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PEÑAFIEL',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PORTILLO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RUEDA',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORDESILLAS',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TUDELA DEL DUERO',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VIANA DEL CEGA',16,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TOULON',258,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AVIGNON',259,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA RCHE-SUR-YON',260,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VENECIA',361,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VERONA',362,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VICENZA',363,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BOLTAÑA',175,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('RIANXO',175,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('POITIERS',261,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ÉPINAL',263,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AUXERRE',264,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VERSALLES',253,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZAMORA',369,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BENAVENTE',369,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CUBILLOS',369,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL PEGO',369,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PINILLA DE TORO',369,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PRADO',369,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('QUINTANILLA DEL MONTE',369,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('REQUEJO',369,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORO',369,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZARAGOZA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TARAZONA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('AINZON',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALAGON',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALFARIN',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALFAMEN',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ALHAMA DE ARAGON',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA ALMUNIA DE DOÑA GODINA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ARIZA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ATECA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BORJA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BREA DE ARAGON',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('BUJARALOZ',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EL BURGO DE EBRO',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALATAIUD',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CALATORAO',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CARIÑENA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CASPE',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('CUARTE DE HUERVA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('DAROCA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EJEA DE LOS CABALLEROS',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('EPILA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ESCATRON',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FABARA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FIGUERUELAS',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('FUENTES DEL EBRO',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GALLUR',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('GELSA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ILLUECA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LECIÑENA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LUCENI',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MAELLA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MARIA DE HUERVA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MEQUINENZA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MORATA DE JALON',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('MUEL',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA MUELA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('NONASPE',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PASTRIZ',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PEDROLA',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PINA DE EBRO',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('PINSEQUE',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('LA PUEBLA DE ALFINDEN',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('QUINTO',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('REMOLINOS',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SADABA',11,sortida);
```

```
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SANT MATEO DE GALLEGO',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('SASTAGO',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('TORRES DE BERRELLEN',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('UTEBO',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('VILLANUEVA DE GALLEGO',11,sortida);
GESTION_LOCALITAT.PRC_ALTA_LOCALITAT('ZUERA',11,sortida);
END;
/
--CONSULTA LOCALITAT
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_LOCALITAT.PRC_CONSULTA_UNA_LOCALITAT('TERRASA',sortida);
GESTION_LOCALITAT.PRC_CONSULTA_UNA_LOCALITAT('SABADELL',sortida);
GESTION_LOCALITAT.PRC_CONSULTA_UNA_LOCALITAT('SALAMANCA',sortida);
GESTION_LOCALITAT.PRC_CONSULTA_UNA_LOCALITAT('MANRESA',sortida);
END;
/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_LOCALITAT.PRC_CONSULTA_LOCALITATS(sortida);
END;
/

--UBICACIONS
/*******************************************
GESTION_UBICACIO.PRC_ALTA_UBICACIO(
   P_IDVIAUBICA => P_IDVIAUBICA,
   P_DIRECCIONUBICA => P_DIRECCIONUBICA,
   P_NUMEROUBICA => P_NUMEROUBICA,
   P_PISUBICA => P_PISUBICA,
   P_PORTAUBICA => P_PORTAUBICA,
   P_CODIPOSTALUBICA => P_CODIPOSTALUBICA,
   P_IDLOCALITAT => P_IDLOCALITAT,
   sortida
  );
************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'ABAT ODO (L´) ',127,6,3,'08037',1,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'ACACIES',72,6,4,'08021',2,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'ACACIES',103,6,2,'08027',3,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'ACACIES',133,2,1,'08035',4,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'ACACIES',121,8,1,'08017',5,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'ACACIES',19,9,3,'08013',1,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'ACACIES',44,3,3,'08024',2,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'ACACIES',146,8,4,'08018',3,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(21,'ACTRIU TUBAU (L´)',83,7,1,'08024',4,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGREGACIO (L´)',144,3,3,'08013',5,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',75,1,3,'08024',1,sortida);
```

```
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',15,6,4,'08020',2,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',53,1,4,'08027',3,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',45,3,4,'08021',4,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',38,0,1,'08018',5,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',11,1,2,'08019',1,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',55,0,1,'08013',2,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',73,0,1,'08020',3,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',60,4,4,'08025',4,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',104,8,3,'08032',5,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGRICULTURA',29,4,1,'08031',1,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUDELLS',115,9,1,'08002',2,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUDELLS',71,4,4,'08003',3,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUDELLS',119,8,1,'08013',4,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUDES',8,1,1,'08034',5,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUDES',146,5,1,'08022',1,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUDES',27,8,3,'08030',2,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUDES',40,6,4,'08027',3,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUDES',127,4,2,'08025',4,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUDES',7,0,1,'08016',5,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUILAR',59,7,3,'08028',1,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUILAR',111,10,4,'08018',2,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUSTI I MILA',115,6,3,'08025',3,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AGUSTINA SARAGOSSA',137,9,3,'08015',4,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AIGUABLAVA',69,0,1,'08001',5,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AIGUABLAVA',23,8,3,'08037',1,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AIGUABLAVA',144,7,4,'08020',2,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AIGUABLAVA',106,6,2,'08024',3,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'AIGUABLAVA',87,5,2,'08032',4,sortida);
GESTION_UBICACIO.PRC_ALTA_UBICACIO(2,'ALBA (L´)',37,1,4,'08025',5,sortida);
END;
/
DECLARE
sortida VARCHAR(500):='';
BEGIN
-- CONSULTA CP
GESTION_UBICACIO.PRC_CONSULTAR_UNA_UBICACIO_CP('08025',sortida );
END;
/
--TOTES ELS UBICCIONS
DECLARE
sortida VARCHAR(500):='';
BEGIN
 GESTION_UBICACIO.PRC_CONSULTAR_UBICACIONS(sortida);
 END;
 /
-- EXECUTA INSERTAR CLIENTS
/*****************************************
 GESTION_CLIENT.PRC_ALTA_CLIENT(
   P_DNICLIENT => P_DNICLIENT,
   P_DATAALTACLIENT => P_DATAALTACLIENT,
   P_ESTATCLIENT => P_ESTATCLIENT,
   P_NOMCLIENT => P_NOMCLIENT,
```

```
   P_COGNOM1CLIENT => P_COGNOM1CLIENT,
   P_COGNOM2CLIENT => P_COGNOM2CLIENT,
   P_IDUBICACLIENT => P_IDUBICACLIENT,
   P_IDTIPOCLIENT => P_IDTIPOCLIENT,
   P_OBSERVACIOCLIENT => P_OBSERVACIOCLIENT,
   sortida
  );
**********************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_CLIENT.PRC_ALTA_CLIENT('11259485L','03/11/2001',1,'ABDELKRIM','FONTOUA','DUCREUX',2,1,'particular',
sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('49213673K','23/11/2001',1,'AARON','DOUHET','MALAGRAVA',1,1,'particular',sor
tida);
GESTION_CLIENT.PRC_ALTA_CLIENT('11259485L','20/07/2008',1,'ABDELKRIM','FONTOUA','DUCREUX',2,1,'particular',
sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('57248539K','07/11/2010',1,'ABAELKADER','FONZ','SALIP',3,1,'particular',sortida
);
GESTION_CLIENT.PRC_ALTA_CLIENT('74075914K','01/12/2001',1,'ABDERRAHMAN','CHAVARRIAS','PAXERAS',4,1,'par
ticular',sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('44331870Z','27/02/2007',1,'ABDELMALIK','HERETTE','MAIDEU',5,1,'particular',s
ortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('33624203C','03/03/2005',1,'ABDELKADER','BANTULA','BARRACA',6,1,'particula
r',sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('41929980Z','04/02/2008',1,'ABDEL','IBORRA','ALGORA',7,1,'particular',sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('39895167B','06/07/2005',1,'ABDELAH','BUTTICAZ','RECAMAN',8,1,'particular',s
ortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('39487303Y','14/06/2005',1,'ABDERRAMAN','NGOUNDO','ESPONADE',9,1,'partic
ular',sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('77784282F','15/03/2002',1,'ABDERRAMAN','TAMERON','PLAZA',10,1,'particular',
sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('27323876Z','22/03/2002',1,'ABDESELAN','YSNARD','ESCUER',11,1,'particular',s
ortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('71829137E','23/06/2009',1,'ABDESELAN','BLENNER','GURUPEGUI',12,1,'particul
ar',sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('22489285T','03/08/2003',1,'ABDULAI','PARDINAS','ARREGUS',13,1,'particular',s
ortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('95979492D','13/05/2009',1,'ABDERRAMAN','LAPIDO','ESTARRONA',14,1,'partic
ular',sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('24936715T','05/01/2005',1,'ABDESSELAM','LABAQUIOL','GARIDO',15,1,'particul
ar',sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('11434537H','19/05/2007',1,'ABDESSELAN','PUNYOLES','SALAGUREN',16,1,'part
icular',sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('74101919J','22/06/2009',1,'ABDO','AHIJADO','BEIXACONILL',17,1,'particular',so
rtida);
GESTION_CLIENT.PRC_ALTA_CLIENT('16682439X','26/11/2010',1,'ABDULIA','ARRAUT','SANTALUCIA',18,1,'particular',
sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('80687355C','26/03/2008',1,'ABDONA','CHADRADA','MARFAGON',19,1,'particular
',sortida);
GESTION_CLIENT.PRC_ALTA_CLIENT('10409647P','29/01/2007',1,'ABDESLAM','BELMEZ','RASHID',20,1,'particular',sort
ida);
```

```
GESTION_CLIENT.PRC_ALTA_CLIENT('32019954K','24/02/2010',1,'ABDESSELAN','TRUNAS','WICHI',21,1,'particular',sor
tida);
END;
/
-- CONSULTA DE UN CLIENTE POR DNI
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_CLIENT.PRC_CONSULTAR_DNI_CLIENT('11259485L',sortida);
end;
/
-- EXECUTA INSERTAR FABRICANT
/*******************************************************
 GESTION_FABRICANT.PRC_ALTA_FABRICANT(
   P_NIFFABRICANT => P_NIFFABRICANT,
   P_DATAALTAFABRICANT => P_DATAALTAFABRICANT,
   P_ESTATFABRICANT => P_ESTATFABRICANT,
   P_NOMCOMERCIALFABRICANT => P_NOMCOMERCIALFABRICANT,
   P_IDUBICAFABRICANT => P_IDUBICAFABRICANT,
   sortida
  );
************************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A08624203','18/01/2012',1,'ENDESA',41,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A08299802','19/01/2012',1,'FECSA',42,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A07975401','20/01/2012',1,'HIDROCARBUROS DEL
NORTE',43,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A07651000','18/01/2012',1,'CONTROL ENERGIA CC',44,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A07326599','19/01/2012',1,'CATALANA ENERGETICA',45,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A07002198','20/01/2012',1,'BBENERGY',46,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A06677797','18/01/2012',1,'ENHER',47,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A06353396','19/01/2012',1,'COMPANY ITT',48,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A06028995','20/01/2012',1,'HIDOELECTRIC DEL EBRO',49,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A05704594','18/01/2012',1,'SUNYER I ASSOCIATS',50,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A05380193','19/01/2012',1,'FRANACE CONTROLLER',51,sortida);
GESTION_FABRICANT.PRC_ALTA_FABRICANT('A05055792','20/01/2012',1,'HESSEN BBP & GERMANY',52,sortida);
END;
/
/*******************************************************
  GESTION_MODELO.PRC_ALTA_MODELO(
   P_DESCRIPCIOMODELO => P_DESCRIPCIOMODELO,
   P_ANYMODELO => P_ANYMODELO,
   P_IDFABRICANTMODELO => P_IDFABRICANTMODELO,
   P_DATAALTAMODELO => P_DATAALTAMODELO,
   P_ESTATMODELO => P_ESTATMODELO,
   P_OBSERVACIONSMODELO => P_OBSERVACIONSMODELO,
   sortida
  );
  *******************************************************/
DECLARE
```

```
sortida VARCHAR(500):='';
BEGIN
GESTION_MODELO.PRC_ALTA_MODEL('Esneider','2004','01/01/2004',1,'Actiu amb Suport'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('CKA','2005','01/01/2005',1,'Actiu amb Suport'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('TTTN','2006','01/01/2006',2,'Actiu amb Suport'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('FILERSENT','2007','01/01/2007',2,'Actiu amb Suport'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('IPORG','2008','01/01/2008',2,'Actiu amb Suport'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('GOLSTEN','2009','01/01/2009',3,'Actiu amb Suport'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('FIBRAL','2010','01/01/2010',3,'Actiu amb Suport'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('ENHER','2011','01/01/2011',4,'Actiu amb Suport'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('ENDESA','2012','01/01/2012',4,'Actiu amb Suport'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('TURINCORP','1995','01/01/1995',5,'Absolet fora de servei'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('FURTHER','1996','01/01/1996',6,'Absolet fora de servei'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('COLMAN','1997','01/01/1997',7,'Absolet fora de servei'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('MACII','1998','01/01/1998',8,'Absolet fora de servei'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('ROMY','1999','01/01/1999',9,'Absolet fora de servei'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('RHONEE','2000','01/01/2000',10,'Absolet fora de servei'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('POLENC','2001','01/01/2001',11,'Absolet fora de servei'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('FIBACCIITALY','2002','01/01/2002',12,'Absolet fora de servei'sortida);
GESTION_MODELO.PRC_ALTA_MODEL('HONNERDUMPER','2003','01/01/2003',12,'Absolet fora de servei'sortida);
END;
/
-- EXECUTA INSERTAR COMPTADOR
/************************************************
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(
   P_IDMODELCOMPTADOR => P_IDMODELCOMPTADOR,
   P_IDUBICACIOCOMPTADOR => P_IDUBICACIOCOMPTADOR,
   P_DATAALTACOMPTADOR => P_DATAALTACOMPTADOR,
   P_NUMEROSERIECOMPTADOR => P_NUMEROSERIECOMPTADOR,
   P_OBSERVACIONSCOMPTADOR => P_OBSERVACIONSCOMPTADOR,
  sortida
 );
 ************************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(1,1,'29/12/2011','123456789',Suport);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(2,2,'30/12/2011','123456790',Suport);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(3,3,'31/12/2011','123456791',Suport);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(4,4,'01/01/2012','123456792',Suport);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(5,5,'02/01/2012','123456793',Suport);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(6,6,'03/01/2012','123456794',Suport);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(7,7,'04/01/2012','123456795',Suport);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(8,8,'05/01/2012','123456796',Suport);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(9,9,'06/01/2012','123456797',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(1,10,'07/01/2012','123456798',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(2,11,'08/01/2012','123456799',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(3,12,'09/01/2012','123456800',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(4,13,'10/01/2012','123456801',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(5,14,'11/01/2012','123456802',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(6,15,'12/01/2012','123456803',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(7,16,'13/01/2012','123456804',sortida);
```

```
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(8,17,'14/01/2012','123456805',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(9,18,'15/01/2012','123456806',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(1,19,'16/01/2012','123456807',sortida);
GESTION_COMPTADOR.PRC_ALTA_COMPTADOR(2,20,'17/01/2012','123456808',sortida);
end;
/
-- consulta comptador
DECLARE
sortida VARCHAR(1000):='';
BEGIN
  GESTION_COMPTADOR.PRC_CONSULTAR_COMPTADOR(sortida);
  END;
/
-- EXECUTA  TIPUS INSPECCIO
/************************************************************
 GESTION_TIPO_INSPECCIO.PRC_ALTA_TIPO_INSPECCIO(
    P_DESCRIPCIOINSPECCIO => P_DESCRIPCIOINSPECCIO,
    sortida
  );
************************************************************/
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_TIPO_INSPECCIO.PRC_ALTA_TIPO_INSPECCIO('Superada',sortida);
GESTION_TIPO_INSPECCIO.PRC_ALTA_TIPO_INSPECCIO('Reservas',sortida);
GESTION_TIPO_INSPECCIO.PRC_ALTA_TIPO_INSPECCIO('No superada',sortida);
END;
/
-- EXECUTA INSPECCIO
/*****************************************************
  GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS(
    P_DATAINSPECCIO => P_DATAINSPECCIO,
    P_CODIINSPECTOR => P_CODIINSPECTOR,
    P_RESULTATINSPECCIO => P_RESULTATINSPECCIO,
    P_OBSERVACIONSINSPECCIO => P_OBSERVACIONSINSPECCIO,
    S_RSP => S_RSP
  );
*****************************************************/
-- EXECUTA INSPECCIO
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('10/04/2012','123456189',1,'revisió  de central Eolica',sortida);
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('11/04/2012','123456502',1,'revisió  de central Termica',sortida);
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('12/04/2012','123456815',1,'revisió  de central Termica',sortida);
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('13/04/2012','123457128',2,',revisió de central Solar',sortida);
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('14/04/2012','123457441',2,'revisió de central Nuclear',sortida);
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('15/04/2012','123457754',1,'revisió de central Nuclear',sortida);
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('17/04/2012','123458380',1,'revisio de central Termica',sortida);
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('17/04/2012','123458380',1,'revisio de central Termica',sortida);
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('17/04/2012','123458380',1,'revisio de central Carbó',sortida);
GESTION_INPECCIONS.PRC_ALTA_INSPECCIONS('17/04/2012','123458380',1,'revisio de central Carbó',sortida);
```

```
END;
/

-- EXECUTA INSERT CENTRALS DE PRODUCCIO
/
/*****************************************************************
  GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(
    P_DATAALTACENTRAL => P_DATAALTACENTRAL,
    P_IDTIPOCENTRAL => P_IDTIPOCENTRAL,
    P_DATAULTMIMAINSPECCIOCENTRAL => P_DATAULTMIMAINSPECCIOCENTRAL,
    P_IDINSPECCIOCENTRAL => P_IDINSPECCIOCENTRAL,
    P_ENERGIAMAX => P_ENERGIAMAX,
    P_ENERGIAMIN => P_ENERGIAMIN,
    P_IDUBICACENTRAL => P_IDUBICACENTRAL,
    P_OBSERVACIONSCENTRAL => P_OBSERVACIONSCENTRAL,
    S_RSP => S_RSP
  );
*****************************************************************/
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'12/04/2012',1,10000.00,1000.00,21,'PRODUCCIO
ELOICA',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'13/04/2012',1,20000.00,1000.00,22,'PRODUCCIO
NUCLEAR',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'14/04/2012',1,3000.00,1000.00,23,'PRODUCCIO
CARBÓ',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'15/04/2012',1,4000.00,1000.00,24,'PRODUCCIO
TERMICA',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'16/04/2012',1,1000.00,500.00,25,'PRODUCCIO
SOLAR',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'17/04/2012',1,2000.00,1000.00,26,'PRODUCCIO
ELOICA',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'18/04/2012',1,3000.00,1000.00,27,'PRODUCCIO
NUCLEAR',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'19/04/2012',1,4000.00,1000.00,28,'PRODUCCIO
CARBÓ',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'12/04/2012',1,2000.00,1000.00,29,'PRODUCCIO
TERMICA',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'13/04/2012',1,3000.00,500.00,30,'PRODUCCIO
SOLAR',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'14/04/2012',1,4000.00,1000.00,31,'DISTRIBUCIO',sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_PRODUCCIO(1,'15/04/2012',1,2000.00,1000.00,32,'DISTRIBUCIO',sortida);
/
--CENTRAL DE DISTRIBUCIO
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('16/04/2012','14/04/2012',1,3000.00,1000.00,33,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('17/04/2012','14/04/2012',1,4000.00,1000.00,34,'DISTRIBUCIO'
,sortida);
```

```
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('19/04/2012','15/04/2012',1,2000.00,500.00,35,'DISTRIBUCIO',
sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('12/04/2012','16/04/2012',1,3000.00,1000.00,36,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('13/04/2012','10/04/2012',1,4000.00,1000.00,37,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('14/04/2012','11/04/2012',1,2000.00,1000.00,38,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('15/04/2012','12/04/2012',1,3000.00,1000.00,39,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('16/04/2012','13/04/2012',1,4000.00,1000.00,40,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('17/04/2012','15/04/2012',1,2000.00,500.00,41,'DISTRIBUCIO',
sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('15/04/2012','16/04/2012',1,3000.00,1000.00,42,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('17/04/2012','10/04/2012',1,4000.00,1000.00,43,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('15/04/2012','11/04/2012',1,2000.00,1000.00,44,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('17/04/2012','12/04/2012',1,3000.00,1000.00,45,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('15/04/2012','13/04/2012',1,2000.00,500.00,46,'DISTRIBUCIO',
sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('16/04/2012','12/04/2012',1,3000.00,1000.00,47,'DISTRIBUCIO'
,sortida);
GESTION_CENTRAL.PRC_ALTA_CENTRAL_DISTRIBUCIO('15/04/2012','13/04/2012',1,4000.00,1000.00,48,'DISTRIBUCIO'
,sortida);
END;
/
--CONSULTA CENTRALS PRODUCCIO
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_CENTRAL.PRC_CONSULTAR_CENTRAL_PRODU(sortida);
END;
/
DECLARE
sortida VARCHAR(500):='';
BEGIN
--CONSULTA CENTRALS DITRIBUCIO
 GESTION_CENTRAL.PRC_CONSULTAR_CENTRAL_DISTRIB(sortida);
END;
/
-- EXECUTA CONECTA COMPTADORS A DISTRIBUCIO
/*************************************************************
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(
   P_IDTIPOLINEACONECTA => P_IDTIPOLINEACONECTA,
   P_DATAALTALINEACONECTA => P_DATAALTALINEACONECTA,
   P_CONECTACENTRALDISTRIBUCIO => P_CONECTACENTRALDISTRIBUCIO,
   P_CONECTACOMPTADOR => P_CONECTACOMPTADOR,
   P_OBSERVACIONSLINEACONECTA => P_OBSERVACIONSLINEACONECTA,
```

```
   sortida
 );
 ************************************/
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(1,'01/01/2012',1,1,1,'COMPTADOR 1 CONECTAT
CENTRAL 11',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(2,'01/01/2012',2,2,1,'COMPTADOR 2 CONECTAT
CENTRAL 12',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(3,'01/01/2012',3,3,1,'COMPTADOR 3 CONECTAT
CENTRAL 13',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(4,'01/01/2012',4,4,1,'COMPTADOR 4 CONECTAT
CENTRAL 14',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(5,'01/01/2012',5,5,1,'COMPTADOR 5 CONECTAT
CENTRAL 15',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(6,'01/01/2012',6,6,1,'COMPTADOR 6 CONECTAT
CENTRAL 16',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(7,'01/01/2012',7,7,1,'COMPTADOR 7 CONECTAT
CENTRAL 17',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(8,'01/01/2012',8,8,1,'COMPTADOR 8 CONECTAT
CENTRAL 18',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(9,'01/01/2012',9,9,1,'COMPTADOR 9 CONECTAT
CENTRAL 19',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(10,'01/01/2012',10,10,1,'COMPTADOR 10
CONECTAT CENTRAL 20',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(1,'01/01/2012',1,1,2,'COMPTADOR 1 CONECTAT
CENTRAL 11',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(2,'01/01/2012',2,2,2,'COMPTADOR 2 CONECTAT
CENTRAL 12',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(3,'01/01/2012',3,3,2,'COMPTADOR 3 CONECTAT
CENTRAL 13',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(4,'01/01/2012',4,4,2,'COMPTADOR 4 CONECTAT
CENTRAL 14',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(5,'01/01/2012',5,5,2,'COMPTADOR 5 CONECTAT
CENTRAL 15',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(6,'01/01/2012',6,6,2,'COMPTADOR 6 CONECTAT
CENTRAL 16',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(7,'01/01/2012',7,7,2,'COMPTADOR 7 CONECTAT
CENTRAL 17',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(8,'01/01/2012',8,8,2,'COMPTADOR 8 CONECTAT
CENTRAL 18',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(9,'01/01/2012',9,9,2,'COMPTADOR 9 CONECTAT
CENTRAL 19',sortida);
GESTION_LINEA_CONECTAR.PRC_CONECTA_LINEA_COMPTADOR(10,'01/01/2012',10,10,2,'COMPTADOR 10
CONECTAT CENTRAL 20',sortida);
END;
/
-- CONSULTAR LINEAS CONECTADES A COMPTADORS
DECLARE
sortida VARCHAR(1000):='';
BEGIN
```

```
 GESTION_LINEA_CONECTAR.PRC_CONSULTAR_LINEAS(sortida);
 END;
/
DECLARE
sortida VARCHAR(1000):='';
BEGIN
  GESTION_LINEA_CONECTAR.PRC_CONSULTAR_CONEXIO_LINEA(1,sortida);
  GESTION_LINEA_CONECTAR.PRC_CONSULTAR_CONEXIO_LINEA(2,sortida);
  GESTION_LINEA_CONECTAR.PRC_CONSULTAR_CONEXIO_LINEA(3,sortida);
  GESTION_LINEA_CONECTAR.PRC_CONSULTAR_CONEXIO_LINEA(4,sortida);
  GESTION_LINEA_CONECTAR.PRC_CONSULTAR_CONEXIO_LINEA(5,sortida);
  GESTION_LINEA_CONECTAR.PRC_CONSULTAR_CONEXIO_LINEA(6,sortida);
  GESTION_LINEA_CONECTAR.PRC_CONSULTAR_CONEXIO_LINEA(7,sortida);
 END;
/
-- EXECUTA ALTA CONTRACTE
/**********************************************************************
  GESTION_CONTRACTE.PRC_ALTA_CONTRACTE(
    P_DNICLIENTCONTRACTE => P_DNICLIENTCONTRACTE,
    P_IDCOMPTADORCONTRACTE => P_IDCOMPTADORCONTRACTE,
    P_DATAALTACONTRACTE => P_DATAALTACONTRACTE,
    P_POTENCIACONTRACTE => P_POTENCIACONTRACTE,
    P_OBSERVACIOCONTRACTE => P_OBSERVACIOCONTRACTE,
   sortida
  );
  **********************************************************************/
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('49213673K',1,'23/11/2001',500.00,'Paga per ventanilla',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('11259485L',2,'20/07/2008',200.00,'Paga per transferencia',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('57248539K',3,'07/11/2010',1000.00,'Reclamacio cobro',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('74075914K',4,'01/12/2001',250.00,'Paga per ventanilla',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('44331870Z',5,'27/02/2007',500.00,'Paga per transferencia',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('33624203C',6,'03/03/2005',200.00,'Reclamacio cobro',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('41929980Z',7,'04/02/2008',1000.00,'Paga per ventanilla',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('39895167B',8,'06/07/2005',250.00,'Paga per transferencia',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('39487303Y',9,'14/06/2005',500.00,'Reclamacio cobro',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('77784282F',10,'15/03/2002',200.00,'Paga per ventanilla',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('27323876Z',11,'22/03/2002',1000.00,'Paga per transferencia',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('71829137E',12,'23/06/2009',250.00,'Reclamacio cobro',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('22489285T',13,'03/08/2003',500.00,'Paga per ventanilla',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('95979492D',14,'04/08/2003',200.00,'Paga per transferencia',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('24936715T',15,'05/08/2003',1000.00,'Reclamacio cobro',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('11434537H',16,'06/08/2003',250.00,'Paga per ventanilla',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('74101919J',17,'07/08/2003',500.00,'Paga per transferencia',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('16682439X',18,'08/08/2003',200.00,'Reclamacio cobro',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('80687355C',19,'09/08/2003',1000.00,'Carrec BBVA',sortida);
GESTION_CONTRACTE.PRC_ALTA_CONTRACTE('10409647P',20,'10/08/2003',250.00,'Carrec Banesto',sortida);
END;
/
-- CONSULTA CONTRACTE EN UN DNI
```

```
DECLARE
sortida VARCHAR(1000):='';
BEGIN
  GESTION_CONTRACTE.PRC_CONSULTA_CONTRACTE_DNI('10409647P',sortida);
END;
/
--- CONSULTA CONTRACTE
DECLARE
sortida VARCHAR(1000):='';
BEGIN
   GESTION_CONTRACTE.PRC_CONSULTA_CONTRACTE(sortida);
END;
/
-- EXECUTA LLEGIR LECTURES
/*****************************************************
 GESTION_LECTURA.PRC_LEER_LECTURA(
   P_IDCOMPTADORLECTURA => P_IDCOMPTADORLECTURA,
   P_CONFIRMALECTURA => P_CONFIRMALECTURA,
   P_DATALECTURA => P_DATALECTURA,
   P_IDTIPOLECTURA => P_IDTIPOLECTURA,
   P_LECTURAACTUAL => P_LECTURAACTUAL,
   sortida
  );
*****************************************************/
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_LECTURA.PRC_LEER_LECTURAS(1,1,'01/02/2012',1,200,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(2,1,'01/02/2012',1,300,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(3,1,'01/02/2012',1,400,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(4,1,'01/02/2012',1,500,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(5,1,'01/02/2012',1,600,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(6,1,'01/02/2012',2,700,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(7,1,'01/02/2012',2,800,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(8,1,'01/02/2012',2,900,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(9,1,'01/02/2012',2,1000,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(10,1,'01/02/2012',2,1100,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(1,1,'01/03/2012',1,1500,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(2,1,'01/03/2012',1,500,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(3,1,'01/03/2012',1,800,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(4,1,'01/03/2012',1,250,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(5,1,'01/03/2012',1,100,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(6,1,'01/03/2012',2,125,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(7,1,'01/03/2012',2,75,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(8,1,'01/03/2012',2,100,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(9,1,'01/03/2012',2,50,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(10,1,'01/03/2012',2,130,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(1,1,'01/04/2012',1,1300,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(2,1,'01/04/2012',1,350,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(3,1,'01/04/2012',1,400,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(4,1,'01/04/2012',1,500,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(5,1,'01/04/2012',1,600,sortida);
```

```
GESTION_LECTURA.PRC_LEER_LECTURAS(6,1,'01/04/2012',2,700,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(7,1,'01/04/2012',2,800,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(8,1,'01/04/2012',2,900,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(9,1,'01/04/2012',2,1000,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(10,1,'01/04/2012',2,1100,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(1,1,'01/05/2012',1,600,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(2,1,'01/05/2012',1,700,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(3,1,'01/05/2012',1,800,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(4,1,'01/05/2012',1,900,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(5,1,'01/05/2012',1,1000,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(6,1,'01/05/2012',2,1100,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(7,1,'01/05/2012',2,200,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(8,1,'01/05/2012',2,300,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(9,1,'01/05/2012',2,400,sortida);
GESTION_LECTURA.PRC_LEER_LECTURAS(10,1,'01/05/2012',2,500,sortida);
END;
/
--CONSULTA LECTURES
/***********************************************
  GESTION_LECTURA.PRC_CONSULTAR_LECTURES(
    sortida
  );
***********************************************/
 DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_LECTURA.PRC_CONSULTAR_LECTURES(sortida);
END;
/
--- ESTADISTICA
--E1
DECLARE
sortida VARCHAR(500):='';
BEGIN
 GESTION_E_1.PRC_ALTA_E1(2,sortida);
END;
/
SELECT * FROM E_1;
/
--E2
DECLARE
sortida VARCHAR(500):='';
BEGIN
GESTION_E_2.PRC_ALTA_E_2(3,'12',sortida);
END;
/
SELECT * FROM E_2;
/
-- E3
DECLARE
sortida VARCHAR(500):='';
BEGIN
```

```
 GESTION_E_3.PRC_ALTA_E_3('11',sortida);
 GESTION_E_3.PRC_ALTA_E_3('12',sortida);
 GESTION_E_3.PRC_ALTA_E_3('10',sortida);
END;
/
SELECT * FROM E_3;
/
-- E_4
DECLARE
sortida VARCHAR(500):='';
BEGIN
 GESTION_E_4.PRC_ALTA_E_4('12',sortida);
 GESTION_E_4.PRC_ALTA_E_4('11',sortida);
END;
/
SELECT * FROM E_4;
/
-- E_5
DECLARE
sortida VARCHAR(500):='';
BEGIN
 GESTION_E_5.PRC_ALTA_E_5('12',sortida);
 GESTION_E_5.PRC_ALTA_E_5('11',sortida);
END;
/
SELECT * FROM E_5;
```

14.1 CONSULTAS Y EXTRACIONES SIGNIFICATIVAS.

CONSULTA TIPOS DE CENTRALES

```
--CONSULTA
DECLARE
sortida VARCHAR(1000):='';
BEGIN
GESTION_TIPO_CENTRAL.PRC_CONSULTAR_TIPO_CENTRAL(sortida);
END;
```

RESPUESTA A LA EJECUCIÓN:

```
anonymous block completed

        CONSULTA TIPO CENTRAL

--------------------------------------------------

  Codi Tipo Central:  4 Descripcio Termica Data estat tipo centras: 12/01/12 Estat Central: ALTA Unitats: 50
Caracteristiques: Kgrs. Emissions CO2 Observacions: energia bruta contaminant

  Codi Tipo Central:  6 Descripcio ---------- Data estat tipo centras: 12/01/12 Estat Central: ALTA Unitats: 0
Caracteristiques: ---------- Observacions: Dstribucio
```

Codi Tipo Central: 3 Descripcio Carbó Data estat tipo centras: 12/01/12 Estat Central: ALTA Unitats: 500 Caracteristiques: Kgrs. Emissions CO2 Observacions: energia bruta contaminant

Codi Tipo Central: 2 Descripcio Nuclear Data estat tipo centras: 12/01/12 Estat Central: ALTA Unitats: 400 Caracteristiques: Residus Radiactius Observacions: energia perillosa

Codi Tipo Central: 1 Descripcio Eolica Data estat tipo centras: 12/01/12 Estat Central: ALTA Unitats: 300 Caracteristiques: Molins Observacions: energia neta

Codi Tipo Central: 5 Descripcio Solar Data estat tipo centras: 12/01/12 Estat Central: ALTA Unitats: 1500 Caracteristiques: Plaques Solars Observacions: energia neta

Total tipo de Centrals: 6

******** FI DE LES OPERACIONS *********

14.1.1 CONSULTA DE UNA UBICACIÓN

--CONSULTA DE UNA UBICACIO SEGÚN EL CODIGO POSTAL

```
DECLARE

sortida VARCHAR(500):='';

BEGIN

-- CONSULTA CP

GESTION_UBICACIO.PRC_CONSULTAR_UNA_UBICACIO_CP('8037',sortida );

END;

/
```

RESULTADOS:

anonymous block completed

CONSULTA UBICACIO

Codi Ubicacio: 1 Via Publica: C./ Direccio: ABAT ODO (L') Numero: 127 Pis: 3 Porta: 6 Codi Postal: 8037 Poblacio: BOURG-EN-BRESSE Provincia: AIN Pais: FRANçA

Codi Ubicacio: 36 Via Publica: C./ Direccio: AIGUABLAVA Numero: 23 Pis: 3 Porta: 8 Codi Postal: 8037 Poblacio: BOURG-EN-BRESSE Provincia: AIN Pais: FRANçA

Total ubicacions: 2

******** FI DE LES OPERACIONS *********

CONSULTA DE LOS DATOS DE UN CLIENTE SEGÚN EL DNI

```
DECLARE
```

```
sortida VARCHAR(500):='';

BEGIN

GESTION_CLIENT.PRC_CONSULTAR_DNI_CLIENT('11259485L',sortida);

end;

/
```

RESULTADOS:

anonymous block completed

CONSULTA CLIENT

Dni Client: 11259485L

CONSULTAR CLIENT:

Codi Client: 2 Dni: 11259485L Data alta: 20/07/08 Estat: ALTA Data ultima modificacio: Nom: ABDELKRIM Cognom1: FONTOUA Cognom2: DUCREUX Tipus de Client: Particular Poblacio: LAON Provincia: AISNE Pais: FRANÇA

******** FI DE LES OPERACIONS *********

OK: CONSULTA CLIENT: 11259485L

14.1.2 CONSULTAS NIVEL ESTADISTICO.

```
--- ESTADISTICA

--E1

DECLARE

sortida VARCHAR(500):='';

BEGIN

 GESTION_E_1.PRC_ALTA_E1(2,sortida);

END;

/

SELECT * FROM E_1;

/

--E2

DECLARE

sortida VARCHAR(500):='';
```

```
BEGIN

GESTION_E_2.PRC_ALTA_E_2(3,'12',sortida);

END;

/

SELECT * FROM E_2;

/

-- E3

DECLARE

sortida VARCHAR(500):='';

BEGIN

 GESTION_E_3.PRC_ALTA_E_3('11',sortida);

 GESTION_E_3.PRC_ALTA_E_3('12',sortida);

 GESTION_E_3.PRC_ALTA_E_3('10',sortida);

END;

/

SELECT * FROM E_3;

/

-- E_4

DECLARE

sortida VARCHAR(500):='';

BEGIN

 GESTION_E_4.PRC_ALTA_E_4('12',sortida);

 GESTION_E_4.PRC_ALTA_E_4('11',sortida);

END;

/

SELECT * FROM E_4;

/

-- E_5

DECLARE

sortida VARCHAR(500):='';
```

```
BEGIN

 GESTION_E_5.PRC_ALTA_E_5('12',sortida);

 GESTION_E_5.PRC_ALTA_E_5('11',sortida);

END;

/

SELECT * FROM E_5;
```

RESULTADOS:

```
anonymous block completed

ESTADISTIQUES E_1
 Consulta del Codi de Central de Produccio:2

    ESTADISTIQUES E_1
 ----------------------------
 En la linea: 11 amb la Central de produccio: 2 el consum que fet el comptador:6
  es de: 44544 Kw/h.  en durant el 2012

****** FI DE LES OPERACIONS *******

IDE_1               IDCENTRAL           IDCOMPTADOR_E_1       SUMACONSUMS
---------------------- ---------------------- ---------------------- ----------------------
1                   2                   6                   44544
2                   2                   6                   44544

2 rows selected


line 10: SQLPLUS Command Skipped: /


anonymous block completed

ESTADISTIQUES E_2
 Consulta la linea Codi:3 any 2012

    ESTADISTIQUES E_2
 ----------------------------

 Linea: 3 any: 12 promitg de 75,5 Kw/h tolerable dins
   dels maxims permes de 1500 Kw/h. en la linea.

****** FI DE LES OPERACIONS *******

IDE_2               IDLINEACONECTAE_2     ANYOE_2 CONSUMOCONTADORESE_2   CONSUMOENERGIAMEDIOE_2
---------------------- ---------------------- ------- ---------------------- ----------------------
1                   3                   12     1500                80,2
2                   3                   12     1500                75,5

2 rows selected


line 19: SQLPLUS Command Skipped: /


anonymous block completed

    ESTADISTIQUES E_3
_____________________________________
 Consulta  Any : 2011
_____________________________________

  CONTROL LINEAS SOBRECARREGADES E_3
----------------------------------------
```

LINEA OPTIMITZADA

Linea 1
Central 4
Cosum 399 Kw/h.
Tolerable -101 Kw/h.

LINEA SOBRECARREGADES

Linea 1
Central 6
Coconsum 2799 Kw/h.
Tolerable 500 Kw/h.
Sobrecarregada amb 2299 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 6
Cosum 2799 Kw/h.
Tolerable 2299 Kw/h.

LINEA OPTIMITZADA

Linea 2
Central 1
Cosum 299 Kw/h.
Tolerable -701 Kw/h.

LINEA SOBRECARREGADES

Linea 1
Central 2
Coconsum 1096 Kw/h.
Tolerable 500 Kw/h.
Sobrecarregada amb 596 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 2
Cosum 1096 Kw/h.
Tolerable 596 Kw/h.

LINEA OPTIMITZADA

Linea 7
Central 2
Cosum 1096 Kw/h.
Tolerable -2404 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 5
Cosum 299 Kw/h.
Tolerable -201 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 1
Cosum 299 Kw/h.
Tolerable -201 Kw/h.

LINEA SOBRECARREGADES

Linea 1

Central 7
Coconsum 4999 Kw/h.
Tolerable 500 Kw/h.
Sobrecarregada amb 4499 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 7
Cosum 4999 Kw/h.
Tolerable 4499 Kw/h.

_____
Total Lineas: 15
******** FI DE LES OPERACIONS *********

ESTADISTIQUES E_3
_____
Consulta Any : 2012
_____

CONTROL LINEAS SOBRECARREGADES E_3

LINEA SOBRECARREGADES

Linea 1
Central 4
Coconsum 1792 Kw/h.
Tolerable 500 Kw/h.
Sobrecarregada amb 1292 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 4
Cosum 1792 Kw/h.
Tolerable 1292 Kw/h.

LINEA SOBRECARREGADES

Linea 1
Central 6
Coconsum 2367 Kw/h.
Tolerable 500 Kw/h.
Sobrecarregada amb 1867 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 6
Cosum 2367 Kw/h.
Tolerable 1867 Kw/h.

LINEA SOBRECARREGADES

Linea 2
Central 1
Coconsum 1192 Kw/h.
Tolerable 1000 Kw/h.
Sobrecarregada amb 192 Kw/h.

LINEA OPTIMITZADA

Linea 2
Central 1
Cosum 1192 Kw/h.
Tolerable 192 Kw/h.

LINEA SOBRECARREGADES

Linea 1
Central 2
Coconsum 4471 Kw/h.
Tolerable 500 Kw/h.
Sobrecarregada amb 3971 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 2
Cosum 4471 Kw/h.
Tolerable 3971 Kw/h.

LINEA SOBRECARREGADES

Linea 7
Central 2
Coconsum 4471 Kw/h.
Tolerable 3500 Kw/h.
Sobrecarregada amb 971 Kw/h.

LINEA OPTIMITZADA

Linea 7
Central 2
Cosum 4471 Kw/h.
Tolerable 971 Kw/h.

LINEA SOBRECARREGADES

Linea 1
Central 5
Coconsum 1292 Kw/h.
Tolerable 500 Kw/h.
Sobrecarregada amb 792 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 5
Cosum 1292 Kw/h.
Tolerable 792 Kw/h.

LINEA SOBRECARREGADES

Linea 1
Central 1
Coconsum 1192 Kw/h.
Tolerable 500 Kw/h.
Sobrecarregada amb 692 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 1
Cosum 1192 Kw/h.
Tolerable 692 Kw/h.

LINEA SOBRECARREGADES

Linea 1
Central 7
Coconsum 6689 Kw/h.
Tolerable 500 Kw/h.
Sobrecarregada amb 6189 Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 7
Cosum 6689 Kw/h.
Tolerable 6189 Kw/h.

_____
Total Lineas: 15
******** FI DE LES OPERACIONS *********

ESTADISTIQUES E_3
_____
Consulta Any : 2010
_____

CONTROL LINEAS SOBRECARREGADES E_3

LINEA OPTIMITZADA

Linea 1
Central 4
Cosum Kw/h.
Tolerable Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 6
Cosum Kw/h.
Tolerable Kw/h.

LINEA OPTIMITZADA

Linea 2
Central 1
Cosum Kw/h.
Tolerable Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 2
Cosum Kw/h.
Tolerable Kw/h.

LINEA OPTIMITZADA

Linea 7
Central 2
Cosum Kw/h.
Tolerable Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 5
Cosum Kw/h.
Tolerable Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 1
Cosum Kw/h.
Tolerable Kw/h.

LINEA OPTIMITZADA

Linea 1
Central 7
Cosum Kw/h.
Tolerable Kw/h.

_____
Total Lineas: 15
******** FI DE LES OPERACIONS *********

ID_E3	LINEA_E3	CONSUMOCONTADORES_E3	ANY_E3
2	1		11
3	1		11
4	1		11
5	1		11
6	2		11
7	2		11
8	1		11
9	1		11
10	7		11
11	7		11
12	1		11
13	1		11
14	1		11
15	1		11
16	1		11
17	1		11
18	1		12
19	1	2191	12
20	1	2191	12
21	1	3166	12
22	2	3166	12
23	2	1491	12
24	1	1491	12
25	1	5567	12
26	7	5567	12
27	7	5567	12
28	1	5567	12
29	1	1591	12
30	1	1591	12
31	1	1491	12
32	1	1491	12
33	1	8688	12
34	1	2191	12
35	1	3166	12
36	2	1491	12
37	1	5567	12
38	7	5567	12
39	1	1591	12
40	1	1491	12
41	1	8688	12
42	1	2191	12
43	1	2191	12
44	1	3166	12
45	1	3166	12
46	2	1491	12
47	2	1491	12
48	1	5567	12
49	1	5567	12
50	7	5567	12
51	7	5567	12
52	1	1591	12
53	1	1591	12
54	1	1491	12
55	1	1491	12
56	1	8688	12
57	1	8688	12
58	1	2191	12
59	1	2191	12
60	1	3166	12
61	1	3166	12
62	2	1491	12
63	2	1491	12
64	1	5567	12
65	1	5567	12

66	7	5567	12
67	7	5567	12
68	1	1591	12
69	1	1591	12
70	1	1491	12
71	1	1491	12
72	1	8688	12
73	1	8688	12
74	1		11
75	1		11
76	1		11
77	1		11
78	2		11
79	2		11
80	1		11
81	1		11
82	7		11
83	7		11
84	1		11
85	1		11
86	1		11
87	1		11
88	1		11
89	1		11
90	1	2191	12
91	1	2191	12
92	1	3166	12
93	1	3166	12
94	2	1491	12
95	2	1491	12
96	1	5567	12
97	1	5567	12
98	7	5567	12
99	7	5567	12
100	1	1591	12
101	1	1591	12
102	1	1491	12
103	1	1491	12
104	1	8688	12
105	1	8688	12
106	1		10
107	1		10
108	1		10
109	1		10
110	2		10
111	2		10
112	1		10
113	1		10
114	7		10
115	7		10
116	1		10
117	1		10
118	1		10
119	1		10
120	1		10
121	1		10
122	1	399	11
123	1	399	11
124	1	2799	11
125	1	2799	11
126	2	299	11
127	2	299	11
128	1	1096	11
129	1	1096	11
130	7	1096	11
131	7	1096	11
132	1	299	11
133	1	299	11
134	1	299	11
135	1	299	11
136	1	4999	11
137	1	4999	11
138	1	1792	12
139	1	1792	12
140	1	2367	12
141	1	2367	12
142	2	1192	12

```
143             2               1192            12
144             1               4471            12
145             1               4471            12
146             7               4471            12
147             7               4471            12
148             1               1292            12
149             1               1292            12
150             1               1192            12
151             1               1192            12
152             1               6689            12
153             1               6689            12
154             1                               10
155             1                               10
156             1                               10
157             1                               10
158             2                               10
159             2                               10
160             1                               10
161             1                               10
162             7                               10
163             7                               10
164             1                               10
165             1                               10
166             1                               10
167             1                               10
168             1                               10
169             1                               10

168 rows selected


line 30: SQLPLUS Command Skipped: /


anonymous block completed

    ESTADISTIQUES E_4
_____________________________________
Consulta  Any : 2012
_____________________________________


  ESTADISTIQUES E_4
------------------------------
Linea: 1
Te un exces de:  960,5 Kw/h.
Representa el  5,4% sobre el 50% del total anual.
Consum anual de la linea:  9862 Kw/h.


 La linea: 3 no ha consumit per sobre del 50%.
****** FI DE LES OPERACIONS *******

Error: La linea: 3 no ha consumit per sobre del 50%.

    ESTADISTIQUES E_4
_____________________________________
Consulta  Any : 2011
_____________________________________


  ESTADISTIQUES E_4
------------------------------

 La linea: 3 no ha consumit per sobre del 50%.
****** FI DE LES OPERACIONS *******

Error: La linea: 3 no ha consumit per sobre del 50%.

IDE_4               LINEAE_4            ANYOE_4 PERCENTAGEE_4
---------------------- ---------------------- ------- ----------------------
1                   1                   12    5,4
2                   1                   12    5,4

2 rows selected
```

```
line 40: SQLPLUS Command Skipped: /


anonymous block completed

    ESTADISTIQUES E_5
_____________________________________
Consulta  Any : 2012
_____________________________________


  ESTADISTIQUES E_5
------------------------------

Error: -1400ORA-01400: no se puede realizar una inserci?n NULL en ("SYSTEM"."LOG_TFC"."RSPLOG")

    ESTADISTIQUES E_5
_____________________________________
Consulta  Any : 2011
_____________________________________


  ESTADISTIQUES E_5
------------------------------
 Linea: 3
 Te defecte de:  2868,3 Kw/h.
 Representa el  29% per sota del 30% del total anual.
 Consum anual de la linea:  99 Kw/h.
per un total global de:9891 Kw/h.

 Linea: 4
 Te defecte de:  2568,3 Kw/h.
 Representa el  25,97% per sota del 30% del total anual.
 Consum anual de la linea:  399 Kw/h.
per un total global de:9891 Kw/h.

 Linea: 5
 Te defecte de:  2668,3 Kw/h.
 Representa el  26,98% per sota del 30% del total anual.
 Consum anual de la linea:  299 Kw/h.
per un total global de:9891 Kw/h.

 Linea: 6
 Te defecte de:  168,3 Kw/h.
 Representa el  1,7% per sota del 30% del total anual.
 Consum anual de la linea:  2799 Kw/h.
per un total global de:9891 Kw/h.

 Linea: 9
 Te defecte de:  2768,3 Kw/h.
 Representa el  27,99% per sota del 30% del total anual.
 Consum anual de la linea:  199 Kw/h.
per un total global de:9891 Kw/h.

 Linea: 10
 Te defecte de:  2668,3 Kw/h.
 Representa el  26,98% per sota del 30% del total anual.
 Consum anual de la linea:  299 Kw/h.
per un total global de:9891 Kw/h.

 Linea: 11
 Te defecte de:  2568,3 Kw/h.
 Representa el  25,97% per sota del 30% del total anual.
 Consum anual de la linea:  399 Kw/h.
per un total global de:9891 Kw/h.

 Linea: 12
 Te defecte de:  2568,3 Kw/h.
 Representa el  25,97% per sota del 30% del total anual.
 Consum anual de la linea:  399 Kw/h.
per un total global de:9891 Kw/h.

_____________________________________
Total Lineas analitzades: 9
Total Genral consums: 9891 Kw/h. any: 2011

****** FI DE LES OPERACIONS *******
```

```
IDE_5              LINEAE_5            ANYOE_5 CENTPRODUCMENOS30
---------- ---------- ---------- ------- ----------
1                  3                  11     29
2                  4                  11     25,97
3                  5                  11     26,98
4                  6                  11     1,7
5                  9                  11     27,99
6                  10                 11     26,98
7                  11                 11     25,97
8                  12                 11     25,97
9                  3                  11     29
10                 4                  11     25,97
11                 5                  11     26,98
12                 6                  11     1,7
13                 9                  11     27,99
14                 10                 11     26,98
15                 11                 11     25,97
16                 12                 11     25,97
17                 3                  11     29
18                 4                  11     25,97
19                 5                  11     26,98
20                 6                  11     1,7
21                 9                  11     27,99
22                 10                 11     26,98
23                 11                 11     25,97
24                 12                 11     25,97

24 rows selected
```

ESTADISTICA E_6

-- CONLSULTA E_6

```
DECLARE
sortida VARCHAR(500):='';
BEGIN
 GESTION_E_6.PRC_ALTA_E_6(sortida);
END;
```

RESULTADO

```
anonymous block completed

    ESTADISTIQUES E_6
_____________________________________


  ESTADISTIQUES E_5
------------------------------
 Comptador: 2
 Cosums:    99 Kw/h.

 Comptador: 4
 Cosums:    598 Kw/h.

 Comptador: 5
 Cosums:    398 Kw/h.

 Comptador: 6
 Cosums:    2799 Kw/h.

 Comptador: 7
 Cosums:    4999 Kw/h.

 Comptador: 8
 Cosums:    1198 Kw/h.
```

Comptador: 9
Cosums: 299 Kw/h.

Comptador: 10
Cosums: 498 Kw/h.

Comptador: 16
Cosums: 399 Kw/h.

ESTADISTIQUES E_5 PER ANYS

Comptador: 16
Any: 11
Cosums: 399 Kw/h.

Comptador: 4
Any: 11
Cosums: 399 Kw/h.

Comptador: 5
Any: 11
Cosums: 299 Kw/h.

Comptador: 6
Any: 11
Cosums: 2799 Kw/h.

Comptador: 7
Any: 11
Cosums: 4999 Kw/h.

Comptador: 8
Any: 11
Cosums: 199 Kw/h.

Comptador: 9
Any: 11
Cosums: 299 Kw/h.

Comptador: 10
Any: 11
Cosums: 399 Kw/h.

Comptador: 16
Any: 11
Cosums: 399 Kw/h.

Comptador: 2
Any: 12
Cosums: 98 Kw/h.

Comptador: 4
Any: 12
Cosums: 598 Kw/h.

Comptador: 5
Any: 12
Cosums: 398 Kw/h.

Comptador: 6
Any: 12
Cosums: 896 Kw/h.

Comptador: 7
Any: 12
Cosums: 2497 Kw/h.

Comptador: 8
Any: 12
Cosums: 1198 Kw/h.

Comptador: 9
Any: 12
Cosums: 298 Kw/h.

Comptador: 10
Any: 12
Cosums: 498 Kw/h.

Comptador: 16
Any: 12
Cosums: 398 Kw/h.

_____
Total comptadors analitzades: 9

******** FI DE LES OPERACIONS *********

CONSULTA ESTADISTICA E7

```
-- CONSULTA E_7
DECLARE
sortida VARCHAR(500):='';
BEGIN
  GESTION_E_7.PRC_ALTA_E_7(sortida);
END;
/
```

RESULTADO:
anonymous block completed

ESTADISTIQUES E_7

Client DNI: 11434537H Consum mitja de:217,8 Kw/h.
Nom: ABDESSELAN Cognom1: PUNYOLES Cognom2:SALAGUREN
Tipus de Client: Particular
Direccio: C./
AGRICULTURA num.11 pis 2 porta 1
Codi postal:8019
Localitat: BOURG-EN-BRESSE Provincia AIN
Pais: FRANçA

Client DNI: 39895167B Consum mitja de:397,6 Kw/h.
Nom: ABDELAH Cognom1: BUTTICAZ Cognom2:RECAMAN
Tipus de Client: Particular
Direccio: C./
AGRICULTURA num.73 pis 1 porta 0
Codi postal:8020
Localitat: ALBACETE Provincia ALBACETE
Pais: ESPAÑA

Client DNI: 39487303Y Consum mitja de:298,2 Kw/h.
Nom: ABDERRAMAN Cognom1: NGOUNDO Cognom2:ESPONADE
Tipus de Client: Particular
Direccio: C./
AGRICULTURA num.60 pis 4 porta 4
Codi postal:8025
Localitat: ALMANSA Provincia ALBACETE
Pais: ESPAÑA

Client DNI: 41929980Z Consum mitja de:2337,6 Kw/h.
Nom: ABDEL Cognom1: IBORRA Cognom2:ALGORA
Tipus de Client: Particular
Direccio: C./
AGRICULTURA num.55 pis 1 porta 0

Codi postal:8013
Localitat: LAON Provincia AISNE
Pais: FRANçA

Client DNI: 44331870Z Consum mitja de:318,2 Kw/h.
Nom: ABDELMALIK Cognom1: HERETTE Cognom2:MAIDEU
Tipus de Client: Particular
Direccio: C./
AGRICULTURA num.38 pis 1 porta 0
Codi postal:8018
Localitat: CAUDETE Provincia ALBACETE
Pais: ESPAÑA

Client DNI: 77784282F Consum mitja de:417,8 Kw/h.
Nom: ABDERRAMAN Cognom1: TAMERON Cognom2:PLAZA
Tipus de Client: Particular
Direccio: C./
AGRICULTURA num.104 pis 3 porta 8
Codi postal:8032
Localitat: CAUDETE Provincia ALBACETE
Pais: ESPAÑA

Client DNI: 33624203C Consum mitja de:1033,2 Kw/h.
Nom: ABDELKADER Cognom1: BANTULA Cognom2:BARRACA
Tipus de Client: Particular
Direccio: C./
AGRICULTURA num.11 pis 2 porta 1
Codi postal:8019
Localitat: BOURG-EN-BRESSE Provincia AIN
Pais: FRANçA

Client DNI: 11259485L Consum mitja de:80,2 Kw/h.
Nom: ABDELKRIM Cognom1: FONTOUA Cognom2:DUCREUX
Tipus de Client: Particular
Direccio: C./
AGRICULTURA num.15 pis 4 porta 6
Codi postal:8020
Localitat: LAON Provincia AISNE
Pais: FRANçA

Client DNI: 74075914K Consum mitja de:438,2 Kw/h.
Nom: ABDERRAHMAN Cognom1: CHAVARRIAS Cognom2:PAXERAS
Tipus de Client: Particular
Direccio: C./
AGRICULTURA num.45 pis 4 porta 3
Codi postal:8021
Localitat: ALMANSA Provincia ALBACETE
Pais: ESPAÑA

Total clients: 19
******** FI DE LES OPERACIONS *********

NIVEL DE CONSULTAS

```
--CONSULTA C_4
DECLARE
sortida VARCHAR(500):='';
BEGIN
CONSULTAS.PRC_CONSULTA_C4('11259485L',1,2,soritda);
END;
/

RESULTADO:

anonymous block completed
 CONSULTA CLIENTS amb  DNI
        Client amb contracte i model de
        comptador 11259485L
_____________________________

Total : 1
******** FIN DE LES OPERACIONS ********
    CONSULTA CLIENTS C4
---------------------------
DNI CLIENT:  11259485L CODI CONTRACTE: 1 CODI MODEL: 2
 Total : 1
_____________________________________
Total : 1
******** FIN DE LES OPERACIONS ********


-- CONSULTA C7
DECLARE
sortida VARCHAR(500):='';
BEGIN
  CONSULTAS.PRC_CONSULTA_C7('2012',sortida);
END;
/

RESULTADO:
anonymous block completed
CONSULTA_C7: Any de Fabricacio: 2012
_____________________________

  CONSULTA C7
----------------

 Comptador: 1
 Model : FURTHER
 Any Fabricacio : 2005
Ok C7 comptador: 1 Model : FURTHER Any Fabricacio  : 2005

 Comptador: 2
 Model : COLMAN
 Any Fabricacio : 2006
Ok C7 comptador: 2 Model : COLMAN Any Fabricacio  : 2006

 Comptador: 3
 Model : MACII
 Any Fabricacio : 2007
Ok C7 comptador: 3 Model : MACII Any Fabricacio  : 2007

 Comptador: 4
 Model : ROMY
 Any Fabricacio : 2008
Ok C7 comptador: 4 Model : ROMY Any Fabricacio  : 2008

 Comptador: 5
```

Model : RHONEE
Any Fabricacio : 2009
Ok C7 comptador: 5 Model : RHONEE Any Fabricacio : 2009

Comptador: 6
Model : POLENC
Any Fabricacio : 2010
Ok C7 comptador: 6 Model : POLENC Any Fabricacio : 2010

Comptador: 7
Model : FIBACCIITALY
Any Fabricacio : 2010
Ok C7 comptador: 7 Model : FIBACCIITALY Any Fabricacio : 2010

Comptador: 8
Model : HONNERDUMPER
Any Fabricacio : 2010
Ok C7 comptador: 8 Model : HONNERDUMPER Any Fabricacio : 2010

Comptador: 9
Model : TURINCORP
Any Fabricacio : 1995
Ok C7 comptador: 9 Model : TURINCORP Any Fabricacio : 1995

Comptador: 10
Model : FURTHER
Any Fabricacio : 2005
Ok C7 comptador: 10 Model : FURTHER Any Fabricacio : 2005

Comptador: 11
Model : COLMAN
Any Fabricacio : 2006
Ok C7 comptador: 11 Model : COLMAN Any Fabricacio : 2006

Comptador: 12
Model : MACII
Any Fabricacio : 2007
Ok C7 comptador: 12 Model : MACII Any Fabricacio : 2007

Comptador: 13
Model : ROMY
Any Fabricacio : 2008
Ok C7 comptador: 13 Model : ROMY Any Fabricacio : 2008

Comptador: 14
Model : RHONEE
Any Fabricacio : 2009
Ok C7 comptador: 14 Model : RHONEE Any Fabricacio : 2009

Comptador: 15
Model : POLENC
Any Fabricacio : 2010
Ok C7 comptador: 15 Model : POLENC Any Fabricacio : 2010

Comptador: 16
Model : FIBACCIITALY
Any Fabricacio : 2010
Ok C7 comptador: 16 Model : FIBACCIITALY Any Fabricacio : 2010

Comptador: 17
Model : HONNERDUMPER

Any Fabricacio : 2010
Ok C7 comptador: 17 Model : HONNERDUMPER Any Fabricacio : 2010

Comptador: 18
Model : TURINCORP
Any Fabricacio : 1995
Ok C7 comptador: 18 Model : TURINCORP Any Fabricacio : 1995

Comptador: 19
Model : FURTHER
Any Fabricacio : 2005
Ok C7 comptador: 19 Model : FURTHER Any Fabricacio : 2005

Comptador: 20
Model : COLMAN
Any Fabricacio : 2006
Ok C7 comptador: 20 Model : COLMAN Any Fabricacio : 2006
_____
TOTAL COMPTADORS: 20
********** FI DEL PROCES **********

15

Glosario

- **Algebra Relacional**

El álgebra relacional es un conjunto de operaciones que describen paso a paso como computar una respuesta sobre las relaciones, tal y como éstas son definidas en el modelo relacional. Denominada de tipo procedimental, a diferencia del Cálculo relacional que es de tipo declarativo.

- **AND**

Una puerta lógica, o compuerta lógica, es un dispositivo electrónico el cual es la expresión física de un operador booleano en la lógica de conmutación. Su valor {a Y b} disjunto.

- **Atributo**

Un atributo es una especificación que define una propiedad de un Objeto, elemento o archivo. También puede referirse o establecer el valor específico para una instancia determinada de los mismos.

Sin embargo, actualmente, el término atributo puede y con frecuencia se considera como si fuera una propiedad dependiendo de la tecnología que se use.

Para mayor claridad, los atributos deben ser considerados más correctamente como metadatos. Un atributo es con frecuencia y en general una característica de una propiedad.

Los atributos son las características que definen o identifican a una entidad. Estas pueden ser muchas, y el diseñador solo utiliza o implementa las que considere más relevantes. Los atributos son las propiedades que describen a cada entidad en un conjunto de entidades.

- **Booleano**

El tipo de dato lógico o booleano es en computación aquel que puede representar valores de lógica binaria, esto es 2 valores, valores que normalmente representan falso o verdadero.

- **Calve Alternativa**

Una clave alternativa es aquella clave candidata que no ha sido seleccionada como clave primaria, pero que también puede identificar de forma única a una fila dentro de una tabla.

- **Cardinalidad de las relaciones**

Tipo de cardinalidad se representa mediante una etiqueta en el exterior de la relación, respectivamente: "1:1", "1:N" y "N:M", aunque la notación depende del lenguaje utilizado, la que más se usa actualmente es el unificado. Otra forma de expresar la cardinalidad es situando un símbolo cerca de la línea que conecta una entidad con una relación:

"0" si cada instancia de la entidad no está obligada a participar en la relación.

"1" si toda instancia de la entidad está obligada a participar en la relación y, además, solamente participa una vez.

"N" , "M", ó "*" si cada instancia de la entidad no está obligada a participar en la relación y puede hacerlo cualquier número de veces.

Ejemplos de relaciones que expresan cardinalidad:

Cada esposo (entidad) está casado (relación) con una única esposa (entidad) y viceversa. Es una relación 1:1.

Una factura (entidad) se emite (relación) a una persona (entidad) y sólo una, pero una persona puede tener varias facturas emitidas a su nombre. Todas las facturas se emiten a nombre de alguien. Es una relación 1:N.

Un cliente (entidad) puede comprar (relación) varios artículos (entidad) y un artículo puede ser comprado por varios clientes distintos. Es una relación N:M.

También es usual utilizar para varios el *

- **Clave candidata**

En una tabla puede que tengamos más de una columna que puede ser clave primaria por sí misma. En ese caso se puede escoger una para ser la clave primaria y las demás claves serán claves candidatas.

- **Clave Forana**

Una clave ajena o Forana (foreign key o clave foránea) es aquella columna que existiendo como dependiente en una tabla, es a su vez clave primaria en otra tabla.

- **Clave Primaria**

Una clave primaria es aquella columna (o conjunto de columnas) que identifica únicamente a una fila. La clave primaria es un identificador que va a ser siempre único para cada fila. Se acostumbra a poner la clave primaria como la primera columna de la tabla pero es más una conveniencia que una obligación. Muchas veces la clave primaria es numérica auto-incrementada, es decir, generada mediante una secuencia numérica incrementada automáticamente cada vez que se inserta una fila.

En una tabla puede que tengamos más de una columna que puede ser clave primaria por sí misma. En ese caso se puede escoger una para ser la clave primaria y las demás claves serán claves candidatas.

La clave primaria puede especificarse directamente en el momento de la creación de la tabla también. En el estándar SQL, las claves primarias pueden estar compuestas por una o más columnas. Cada columna que forme parte de la clave primaria queda implícitamente definida como NOT NULL. Nótese que algunos sistemas de bases de datos requieren que se marque explícitamente a las columnas de clave primaria como NOT NULL.

- **Dominios**

Un dominio describe un conjunto de posibles valores para cierto atributo también denominados tipo. Como un dominio restringe los valores del atributo, puede ser considerado como una restricción. Matemáticamente, atribuir un dominio a un atributo significa "todos los valores de este atributo deben de ser elementos del conjunto especificado".

Distintos tipos de dominios son: enteros, cadenas de texto, fecha, no procedurales etc.

- **Entidad**

Representa una “cosa” u "objeto" del mundo real con existencia independiente, es decir, se diferencia unívocamente de otro objeto o cosa, incluso siendo del mismo tipo, o una misma entidad.

- **Entidades fuertes y débiles**

Cuando una entidad participa en una relación puede adquirir un papel fuerte o débil. Una entidad débil es aquella que no puede existir sin participar en la relación, es decir, aquella que no puede ser unívocamente identificada solamente por sus atributos.

Una entidad fuerte (también conocida como entidad regular) es aquella que sí puede ser identificada unívocamente. En los casos en que se requiera, se puede dar que una entidad fuerte "preste" algunos de sus atributos a una entidad débil para que, esta última, se pueda identificar.

- **Feedback**

En la teoría de sistemas, en cibernética en la teoría de control, entre otras disciplinas, la retroalimentación (en inglés feedback) es un mecanismo de control de los sistemas dinámicos por el cual una cierta proporción de la señal de salida se redirige de nuevo a la entrada, regulando su comportamiento. El feedback también está presente en numerosos espacios tecnológicos. En este sentido, gran parte de los aparatos y máquinas que utilizamos en nuestra vida cotidiana funcionan a través del sistema de feedback ya que suponen el intercambio y traspaso permanente de datos (de cualquier tipo).

- **FK**

Sigla de Clave Forana.

- **Insert**

Instrucción de SQL para insertar datos en una tabla.

- **Integridad Referencial**

La integridad referencial es una propiedad deseable en las bases de datos. Gracias a la integridad referencial se garantiza que una entidad (fila o registro) siempre se relaciona con otras entidades válidas, es decir, que existen en la base de datos. Implica que en todo momento dichos datos sean correctos, sin repeticiones innecesarias, datos perdidos y relaciones mal resueltas.

- **MagicDraw UML12.0**

Programa comercial para diseño de UML.

- **Normalización SGB**

El proceso de normalización de bases de datos consiste en aplicar una serie de reglas a las relaciones obtenidas tras el paso del modelo entidad-relación al modelo relacional.

Las bases de datos relacionales se normalizan para:

- ✓ Evitar la redundancia de los datos.
- ✓ Evitar problemas de actualización de los datos en las tablas.
- ✓ Proteger la integridad de los datos.

En el modelo relacional es frecuente llamar tabla a una relación, aunque para que una tabla sea considerada como una relación tiene que cumplir con algunas restricciones:

- ✓ Cada tabla debe tener su nombre único.
- ✓ No puede haber dos filas iguales. No se permiten los duplicados.
- ✓ Todos los datos en una columna deben ser del mismo tipo.

Las formas normales son aplicadas a las tablas de una base de datos. Decir que una base de datos está en la forma normal N es decir que todas sus tablas están en la forma normal N.

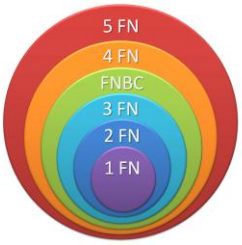


Ilustración 97 Reglas de Normalización de SGBD

Regla 0: el sistema debe ser relacional, base de datos y administrador de sistema. Ese sistema debe utilizar sus facilidades relacionales (exclusivamente) para manejar la base de datos.

Regla 1: la regla de la información, toda la información en la base de datos es representada unidireccionalmente, por valores en posiciones de las columnas dentro de filas de tablas. Toda la información en una base de datos relacional se representa explícitamente en el nivel lógico exactamente de una manera: con valores en tablas.

Regla 2: la regla del acceso garantizado, todos los datos deben ser accesibles sin ambigüedad. Esta regla es esencialmente una nueva exposición del requisito fundamental para las llaves primarias. Dice que cada valor escalar individual en la base de datos debe ser lógicamente direccionable especificando el nombre de la tabla, la columna que lo contiene y la llave primaria.

Regla 3: tratamiento sistemático de valores nulos, el sistema de gestión de base de datos debe permitir que haya campos nulos. Debe tener una representación de la "información que falta y de la información inaplicable" que es sistemática, distinto de todos los valores regulares.

Regla 4: catálogo dinámico en línea basado en el modelo relacional, el sistema debe soportar un catálogo en línea, el catálogo relacional debe ser accesible a los usuarios autorizados. Es decir, los usuarios deben poder tener acceso a la estructura de la base de datos (catálogo).

Regla 5: la regla comprensiva del sublenguaje de los datos, el sistema debe soportar por lo menos un lenguaje relacional que;

Tenga una sintaxis lineal.

Puede ser utilizado recíprocamente y dentro de programas de uso.

Soporte operaciones de definición de datos, operaciones de manipulación de datos (actualización así como la recuperación), seguridad e integridad y operaciones de administración de transacciones.

Regla 6: regla de actualización, todas las vistas que son teóricamente actualizables deben ser actualizables por el sistema.

Regla 7: alto nivel de inserción, actualización, y cancelación, el sistema debe soportar suministrar datos en el mismo tiempo que se inserte, actualiza o esté borrando. Esto significa que los datos se pueden recuperar de una base de datos relacional en los sistemas construidos de datos de filas múltiples y/o de tablas múltiples.

Regla 8: independencia física de los datos, los programas de aplicación y actividades del terminal permanecen inalterados a nivel lógico cuandoquiera que se realicen cambios en las representaciones de almacenamiento o métodos de acceso.

Regla 9: independencia lógica de los datos, los cambios al nivel lógico (tablas, columnas, filas, etc.) no deben requerir un cambio a una solicitud basada en la estructura. La independencia de datos lógica es más difícil de lograr que la independencia física de datos.

Regla 10: independencia de la integridad, las limitaciones de la integridad se deben especificar por separado de los programas de la aplicación y se almacenan en la base de datos. Debe ser posible cambiar esas limitaciones sin afectar innecesariamente las aplicaciones existentes.

Regla 11: independencia de la distribución, la distribución de las porciones de la base de datos a las varias localizaciones debe ser invisible a los usuarios de la base de datos. Los usos existentes deben continuar funcionando con éxito:

cuando una versión distribuida del SGBD se introdujo por primera vez

cuando se distribuyen los datos existentes se redistribuyen en todo el sistema.

Regla 12: la regla de la no subversión, si el sistema proporciona una interfaz de bajo nivel de registro, a parte de una interfaz relacional, que esa interfaz de bajo nivel no se pueda utilizar para subvertir el sistema, por ejemplo: sin pasar por seguridad relacional o limitación de integridad. Esto es debido a que existen sistemas anteriormente no relacionales que añadieron una interfaz relacional, pero con la interfaz nativa existe la posibilidad de trabajar no relacionalmente.

- **OR**

Una puerta lógica, o compuerta lógica, es un dispositivo electrónico el cual es la expresión física de un operador booleano en la lógica de conmutación. Su valor {a O b}.

- **Oracle**

Producto comercial de Base de Datos.

- **PK**

Siglas de Clave Primaria

- **PL/SQL**

PL/SQL (Procedural Language/Structured Query Language) es un lenguaje de programación incrustado en Oracle.

PL/SQL soportará todas las consultas, ya que la manipulación de datos que se usa es la misma que en SQL, incluyendo nuevas características:

El lenguaje PL/SQL está incorporado en:

En un entorno de base de datos los programadores pueden construir bloques PL/SQL para utilizarlos como procedimientos o funciones, o bien pueden escribir estos bloques como parte de scripts SQL*Plus.

Los programas o paquetes de PL/SQL se pueden almacenar en la base de datos como otro objeto, y todos los usuarios que estén autorizados tienen acceso a estos paquetes. Los programas se ejecutan en el servidor para ahorrar recursos a los clientes.

- **Procedure**

Un procedimiento almacenado es código ejecutable que se asocia y se almacena con la base de datos. Los procedimientos almacenados usualmente recogen y personalizan operaciones comunes, como insertar un registro dentro de una tabla, recopilar información estadística, o encapsular cálculos complejos. Son frecuentemente usados por un API por seguridad o simplicidad.

Los procedimientos almacenados no son parte del modelo relacional, pero todas las implementaciones comerciales los incluyen.

- **Relación**

En una base de datos relacional, todos los datos se almacenan y se accede a ellos por medio de relaciones. Las relaciones que almacenan datos son llamadas "relaciones base" y su implementación es llamada "tabla". Otras relaciones no almacenan datos, pero son calculadas al aplicar operaciones relacionales. Estas relaciones son llamadas "relaciones derivadas" y su implementación es llamada "vista" o "consulta". Las relaciones derivadas son convenientes ya que expresan información de varias relaciones actuando como si fuera una sola.

- **Restricciones**

Una restricción es una condición que obliga el cumplimiento de ciertas condiciones en la base de datos. Algunas no son determinadas por los usuarios, sino que son inherentemente definidas por el simple hecho de que la base de datos sea relacional. Algunas otras restricciones las puede definir el usuario, por ejemplo, usar un campo con valores enteros entre 1 y 10.

Las restricciones proveen un método de implementar reglas en la base de datos. Las restricciones restringen los datos que pueden ser almacenados en las tablas. Usualmente se definen usando expresiones que dan como resultado un valor booleano, indicando si los datos satisfacen la restricción o no.

Las restricciones no son parte formal del modelo relacional, pero son incluidas porque juegan el rol de organizar mejor los datos. Las restricciones son muy discutidas junto con los conceptos relacionales.

- **Rol**

Participación en un escenario de casos de Uso.

- **Script**

Código susceptible de ser interpretado por un compilador o cualquier tipo de componente capaz de efectuar una serie secuencial de órdenes establecidas.

- **SGBD**

Siglas de Sistema Gestos de Base de Datos.

- **SO**

Siglas de Sistema Operativo.

- **SQL**

Lenguaje Declarativo para construcción de Base de Datos Relacionales.

- **Staff**

Parte directiva de una organización.

- **Tabla**

Entidad definida.

- **Tables**

Tabla en SQL

- **Triggers**

Procedimientos predefinidos por los administradores de Base de Datos que incluyen restricciones de procesos o conjunto de ellos.

- **Tupla**

Una tupla se define como una función finita que asocia unívocamente los nombres de los atributos de una relación con los valores de una instanciación de la misma. En términos simplistas, es una fila de una tabla relacional.

- **UML**

Lenguaje Unificado de Modelado (LUM o UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables.

- **Update**

Instrucción en lenguaje SQL para proceder a modificar información dentro de una entidad afectando el contenido de las tuplas.

- **View**

Una vista de base de datos es un resultado de una consulta SQL de una o varias tablas; también se le puede considerar una tabla virtual.

Las vistas tienen la misma estructura que una tabla: filas y columnas. La única diferencia es que sólo se almacena de ellas la definición, no los datos. Los datos que se recuperan mediante una consulta a una vista se presentarán igual que los de una tabla. De hecho, si no se sabe

que se está trabajando con una vista, nada hace suponer que es así. Al igual que sucede con una tabla, se pueden insertar, actualizar, borrar y seleccionar datos en una vista. Aunque siempre es posible seleccionar datos de una vista, en algunas condiciones existen restricciones para realizar el resto de las operaciones sobre vistas.

Una vista se especifica a través de una expresión de consulta (una sentencia SELECT) que la calcula y que puede realizarse sobre una o más tablas. Sobre un conjunto de tablas relacionales se puede trabajar con un número cualquiera de vistas.

- **VISIO 2007**

Producto de Microsoft para diseño de UML.

16 BIBLIOGRAFÍA.

- **GUIA PRÁCTICA SQL** Editorial Anaya. Autor Francisco Charte Ojeda.
- **ORACLE 11g** Editorial Ra-Ma. Autor Teaching Soft Group.
- **APRENDIENDO UML** Editorial Prentice Hill. Autor Joseph Achmuller.
- **Base de datos Modelo Entidad Relación** Autores Guillermo Storti, Gladys Ríos y Gabriel Campodónico.
- **Bases de Datos / O.E.I../ U.P.M.** Transparencias de modelos Conceptuales de BD.
- **BASES DE DATOS** Editorial UOC formación de Postgrado. Autores Rafael Camps Paré, Oscar Pérez Mora, Carme Martín Escofet, Marc Gibert Ginestà, Dolors Costal Costa, Luis Alberto Casillas Santillán. Primera edición: mayo 2005.
- **Components lògics d'una base de dades.** Universitat Oberta de Catalunya Autores M. Elena Rodríguez González, Toni Urpí Tubella.
- **Components d'emmagatzematge d'una base de dades.** Universitat Oberta de Catalunya. Autor Ramon Segret i Sala.
- **El llenguatge SQL** Universitat Oberta de Catalunya. Autor Carme Martín Escofet , Carme Martín Escofet.
- **Programació amb SQL** Universitat Oberta de Catalunya. Autor M. Elena Rodríguez González Josep Maria Marco Simó.