Diseño e implementación de un sistema de control del consumo energético

Alfonso Javier Collado Castro
Grado de Ingeniería Informática

Jordi Ferrer Duran
Consultor TFG

15 de junio de 2014

B) GNU Free Documentation License (GNU FDL)

C) Copyright

FICHA DEL TRABAJO FINAL

Título del trabajo:	Diseño e implementación de un sistema de control del consumo energético
Nombre del autor:	Alfonso Javier Collado Castro
Nombre del consultor:	Jordi Ferrer Duran
Fecha de entrega (mm/aaaa):	06/2014
Área del Trabajo Final:	Base de datos
Titulación:	*Grado de Ingeniería Informática*

Resumen del Trabajo (máximo 250 palabras):

El proyecto de Trabajo de Fin de Grado (TFG) del área de Base de datos tiene una doble vertiente que permite desarrollar ampliamente las competencias adquiridas a lo largo del Grado sobre Base de datos.

La primera de las partes desarrolla la implementación de una base de datos transaccional para el control del consumo eléctrico mediante contadores inteligentes. La BBDD propuesta permite desarrollar los diferentes compromisos de gestión de las compañías suministradoras, consumidores, contadores, consumos, y núcleos energéticos (*Smart Cities*), proporcionando los procedimientos almacenados necesarios para su gestión.

La segunda parte desarrolla la implementación de una base de datos analítica, *Data Warehouse*, para la explotación de los datos almacenados en la primera parte del proyecto. El objetivo es la implementación de las entidades necesarias para la explotación analítica de los datos mediante consultas sencillas.

Abstract (in English, 250 words or less):

The TFG database area project has a double side it allows to develop widely the skills acquired during the course about databases.

The first part develops the implementation of a transactional database to manage the power consumption by smart meters. The proposed database deals with the management commitments of companies, consumers, smart meters, consumption and smart cities, providing the store procedures needed for management.

The second part develops the implementation of an analytical database, a data warehouse, for the operation of the data stored in the first part of the project, the transactional database. The goal is to implement the tools needed for an analytical use of data by simple entity queries.

Palabras clave (entre 4 y 8):

Database
OnLine Transaction Processing
DataWarehouse

Índice

Lista de figuras

Lista de tablas

1. Introducción

1.1. Contexto y justificación del Trabajo

La situación energética está sometida, por su naturaleza, a una dinámica compleja, llena de interrelaciones, resultando extremadamente sensible a factores externos, lo cual impone nuevos retos que se traducen en escenarios cada vez más tensionados.

Las administraciones públicas, desde su responsabilidad, están respondiendo ante estos escenarios legislando para promover y desarrollar una gestión más eficiente de la energía e introducir mayores cuotas de libertad en el mercado que la gestiona. La traducción de todas estas preocupaciones se materializó en la Directiva Europea 2009/72/EC sobre normas comunes para el mercado interior de la electricidad (y derogación de la Directiva 2003/54/CE).

A través de la Directiva Europea 2009/72/EC, la Unión Europea anima a los estados miembros para que el año 2020 el 80% de los consumidores cuente con sistemas de contador inteligentes. En este contexto, la utilización de contadores de electricidad inteligentes o electrónicos permite una serie de ventajas alineadas con los objetivos anteriores, siendo la principal, la capacidad de telegestión de estos dispositivos.

La implantación de los contadores inteligentes eliminará los problemas de las distribuidoras para emitir facturas en base a los consumos reales de los hogares, eliminando las comprobaciones manuales de los contadores y la facturación estimada de los servicios. Además, disponer de la información del consumo real de los consumidores también facilita la gestión de la producción y distribución de la energía.

No obstante, las capacidades de telegestión de estos contadores, permite a través del seguimiento del consumo y la recogida masiva de datos, identificar patrones vitales relativos a las costumbres de los consumidores, poniendo en riesgo la intimidad de los consumidores. Al mismo tiempo, la cantidad de información que puede manejar un solo sistema (*Smart grid*) también supone una vulnerabilidad ante la manipulación o intromisión en los sistemas de gestión.

La necesidad a la que responde este trabajo es el diseño de un sistema de base de datos operacional que permita, como plataforma base, la gestión de las lecturas de contadores de electricidad inteligentes (o electrónicos), y la explotación analítica de los datos relacionados, garantizando durante todo el proceso la intimidad de los usuarios y la seguridad de las operaciones mediante su registro y trazabilidad.

1.2. Objetivos del Trabajo

Deberíamos distinguir entre los objetivos del trabajo y los objetivos pedagógicos asociados a la realización del TFG.

El trabajo cuenta con dos objetivos fundamentales:

- ✓ El diseño y desarrollo de una base de datos relacional para la gestión operacional (OLTP, *On Line Transaction Processing*) de los contadores inteligentes y demás actores relacionados.
- ✓ El diseño y desarrollo de una base de datos para la gestión analítica (OLAP, *On Line Analytic Processing*) con un almacén de datos (*Data Warehouse*) de los contadores inteligentes y resto de actores relacionados.

Los objetivos pedagógicos que se persiguen con la realización del trabajo son:

- ✓ Poner en práctica los conocimientos adquiridos en las asignaturas de Base de Datos.
- ✓ Utilizar el lenguaje SQL.
- ✓ Ampliar los conocimientos mediante la utilización de nuevas herramientas.
- ✓ Detectar las necesidades básicas del sistema objeto de análisis.
- ✓ Proponer un diseño que se ajuste a los requisitos proporcionados.
- ✓ Implementar un sistema que encapsule las funciones de acceso a los datos.

1.3. Enfoque y método seguido

El conjunto de fases, ciclo de vida, por las que pasará el sistema a desarrollar se ajusta a un modelo lineal-secuencial del tipo *top-down* clásico o cascada. En cada fase del proyecto se obtendrán resultados que serán la base de partida de la fase siguiente. La generación de resultados se alineará con las diferentes metas de los entregables del proyecto en forma de PEC's.

En cualquier caso, si distinguiremos 2 metodologías para cada una de las partes en las que se estructurará el proyecto, homogéneas con la visión general del ciclo de vida de una base de datos [1] establecida anteriormente.

La primera parte del trabajo consiste en el diseño y desarrollo de una base de datos relacional para la gestión operacional de los contadores inteligentes. Para el desarrollo de este apartado se utilizará el enfoque ANSI/SPARC, también conocido por arquitectura en tres esquemas [2], y la metodología de modelado UML. La selección de esta metodología en detrimento de un enfoque a través del método de integración de vistas (método *bottom-up*) se produce por compatibilidad con el proceso general de ciclo de vida descendente escogido para el desarrollo del trabajo.

La segunda parte del trabajo consiste en el diseño y desarrollo de una base de datos para la gestión analítica (OLAP) de los contadores inteligentes y otros actores relacionados. Para el desarrollo de este apartado se utilizará la metodología propuesta por Ralph Kimball [3]. La elección de esta tecnología en detrimento de otras (SEMMA, KDD, P3TQ, DMAMC…) se basa en su alta implantación, y su mayor detalle sobre las tareas y actividades a realizar en cada fase del proyecto.

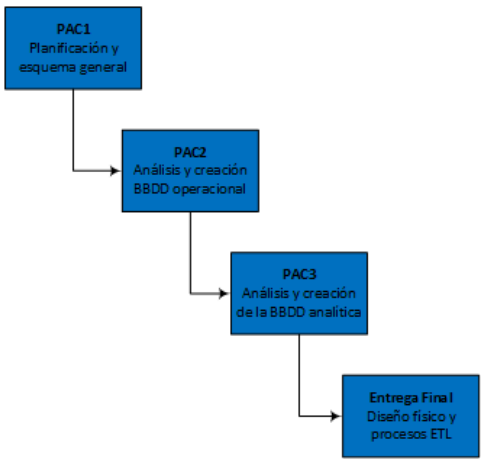


Ilustración 1. Esquema del proyecto y entregas

1.4. Planificación del Trabajo

La elaboración de la planificación ha requerido contar con los siguientes puntos:

- ✓ Identificar los entregables del proyecto
- ✓ Identificar las actividades de primer nivel
- ✓ Descomponer las actividades de primer nivel en tareas y subtareas, creando el EDT
- ✓ Compatibilizar el desarrollo de los trabajos con las fechas de las entregas parciales (PEC) del proyecto

Las actividades identificadas son:

Título	Inicio	Fin	Tiempo	% tiempo
Elaboración del Plan de trabajo (PAC1)	01/03/2014	16/03/2014	12 d	15%
Análisis y creación de la BBDD del sistema operacional (PAC2)	17/03/2014	13/04/2014	21 d	26,25%
Análisis y creación del Data Warehouse del sistema analítico (PAC3)	14/04/2014	11/05/2014	21 d	26,25%
Entrega final	12/05/2014	15/06/2014	26 d	32,50%
Total	01/03/2014	15/06/2014	80 d	

Tabla 1. Planificación del proyecto

La descomposición en tareas y subtareas para cada hito o entrega se muestra en el siguiente esquema:

1. **Elaboración del Plan de trabajo (PAC1)**
   - 1.1. Estudio del proyecto y búsqueda información
   - 1.2. Elección del software
   - 1.3. Definición alcance del proyecto
   - 1.4. Elaboración plan de trabajo
   - 1.5. Redacción memoria del proyecto
   - 1.6. Revisión y entrega
2. **Análisis y creación de la BBDD del sistema operacional (PAC2)**
   - 2.1. Modelado conceptual
     - 2.1.1. Análisis de requisitos
     - 2.1.2. Generación del esquema conceptual
     - 2.1.3. Restricciones de integridad
   - 2.2. Diseño lógico estándar y específico
     - 2.2.1. Descripción del diseño lógico estándar
     - 2.2.2. Descripción del diseño lógico específico en Oracle.
     - 2.2.3. Creación de la base de datos y las tablas relacionadas
     - 2.2.4. Pruebas de las tablas de la base de datos
     - 2.2.5. Diseño de los procedimientos ABM (Alta, Baja, Modificación) mediante procedimientos almacenados
     - 2.2.6. Pruebas y validación de los procedimientos CRUD
     - 2.2.7. Creación disparadores (*triggers*)
     - 2.2.8. Pruebas disparadores
   - 2.3. Diseño físico
     - 2.3.1. Índices
     - 2.3.2. Particiones
     - 2.3.3. Seguridad
     - 2.3.4.Validación y justificación del diseño físico
   - 2.4. Redacción memoria del proyecto
   - 2.5. Revisión y entrega
3. **Análisis y creación del *Data Warehouse* del sistema analítico (PAC3)**
   - 3.1. Estudio de los conceptos de DW y Metodología de diseño de modelos OLAP a partir de modelos OLTP. Breve introducción a los conceptos de DW y metodología de diseño del DW
   - 3.2. Diseño conceptual del sistema
     - 3.2.1. Proceso de negocio
     - 3.2.2. Análisis de requisitos

3.2.3. Definición del gránulo
3.2.4. Identificación de las dimensiones. Atributos descriptores y jerarquías de agregación
3.2.5. Identificación de hechos
3.2.6. Esquema conceptual del sistema
3.2.7. Restricciones de integridad
3.3. Diseño lógico
3.3.1. Descripción del diseño lógico estándar
3.3.2. Descripción del diseño lógico específico en Oracle
3.3.3. Pruebas de las tablas de la BBDD
3.4. Redacción memoria del proyecto
3.5. Revisión y entrega

4. Entrega final

4.1.1.Diseño e implementación del ETL
4.1.2. Pruebas del ETL
4.1.3. Carga del almacén de datos. Datos de prueba
4.2. Diseño físico
4.2.1. Creación de la BBDD del DW
4.2.2. Creación del esquema y almacenamiento para la BBDD
4.2.3. Índices
4.2.4. Particiones/fragmentación
4.2.5. Seguridad
4.2.6. Definición del esquema multidimensional
4.3. Diseño e implementación de las consultas
4.4. Redacción final de la memoria del trabajo
4.5. Confección de la presentación del trabajo
4.6. Redacción del autoinforme de evaluación de las competencias transversales
4.7. Preparación de los entregables del trabajo (BBDD y DW)
4.8. Revisión y entrega final

1.4.1. Cronograma de las tareas (diagramas de Gantt)

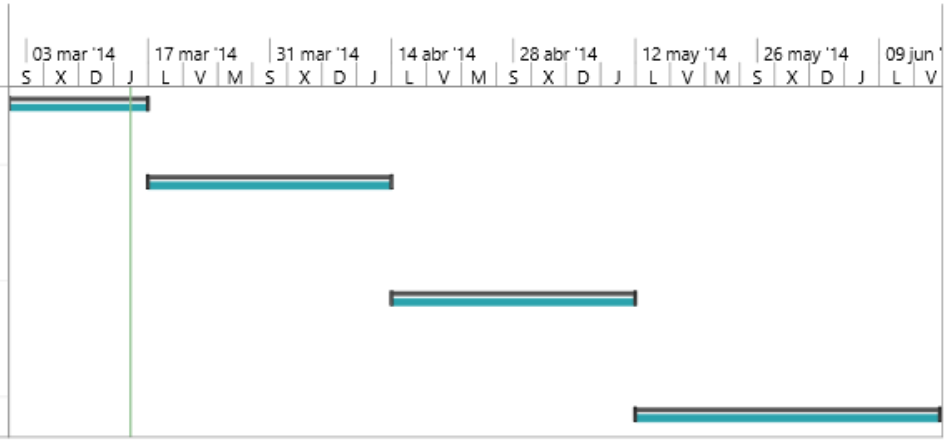
Los cronogramas definidos para cada entregable son:

Cronograma general del proyecto

Inicio 01/03/2014

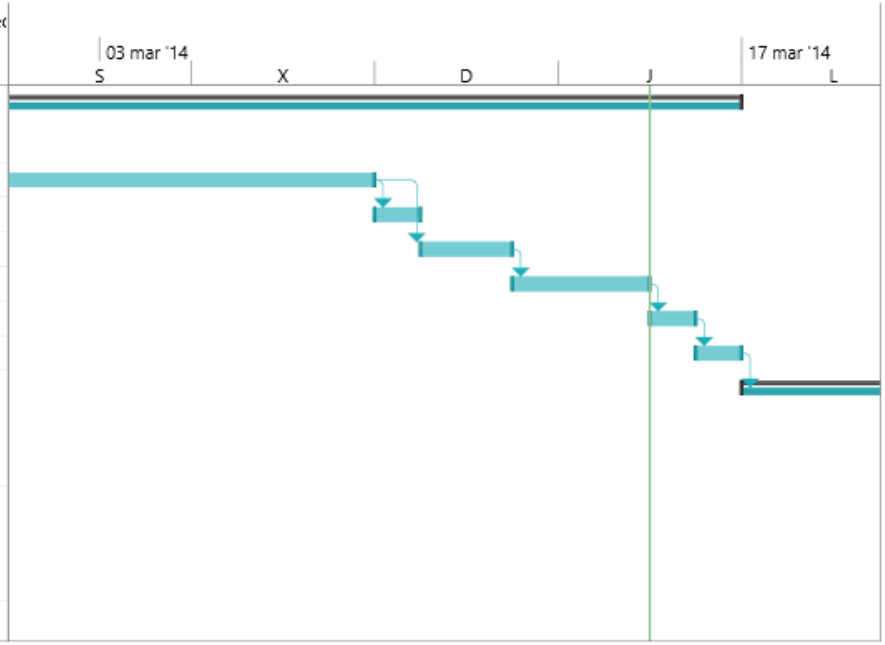
Fin: 15/06/2014

Id	Nombre de tarea	Duración	Comienzo	Fin
1	1 Elaboración del Plan de trabajo (PAC1)	12 días	sáb 01/03/14	dom 16/03/14
8	2 Análisis y creación de la BBDD del sistema operacional (PAC2)	21 días	lun 17/03/14	dom 13/04/14
29	3 Análisis y creación del Data Warehouse del sistema analítico (PAC3)	21 días	lun 14/04/14	dom 11/05/14
42	4 Entrega final	26 días	lun 12/05/14	dom 15/06/14



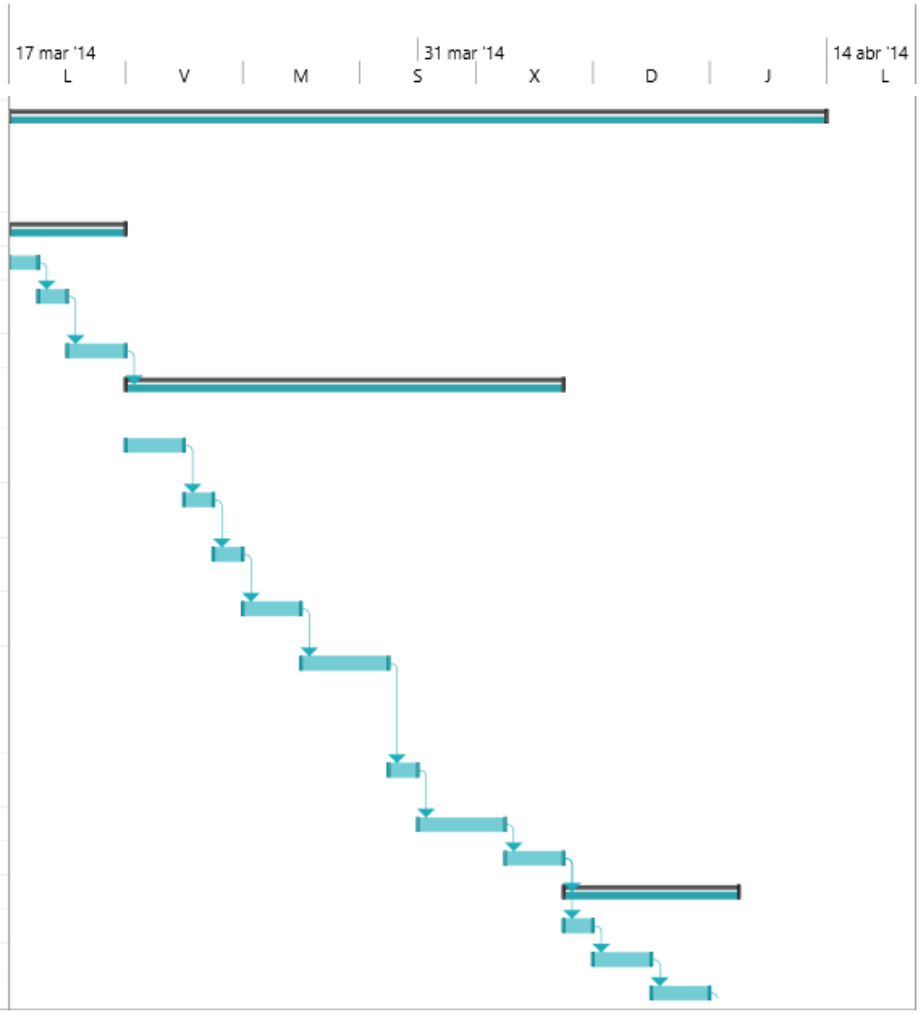
Cronograma del entregable nº 1 (PAC1, 16/03/2014)

Id	Nombre de tarea	Duració	Comienzo	Fin	Pred
1	**1 Elaboración del Plan de trabajo (PAC1)**	**12 días**	**sáb 01/03/14**	**dom 16/03/14**	
2	**1.1 Estudio proyecto y búsqueda i**	7 días	sáb 01/03/14	sáb 08/03/14	
3	**1.2 Elección del software**	1 día	dom 09/03/1	dom 09/03/14	2
4	**1.3 Definición alcance del proyect**	2 días	lun 10/03/14	mar 11/03/14	2
5	**1.4 Elaboración plan de trabajo**	3 días	mié 12/03/1	vie 14/03/14	4
6	**1.5 Redacción memoria del proyec**	1 día	sáb 15/03/14	sáb 15/03/14	5
7	**1.6 Revisión y entrega**	1 día	dom 16/03/1	**dom 16/03/1**	6
8	**2 Análisis y creación de la BBDD del sistema operacional (PAC2)**	**21 días**	**lun 17/03/14**	**dom 13/04/14**	**7**
29	**3 Análisis y creación del Data Warehouse del sistema analítico (PAC3)**	**21 días**	**lun 14/04/14**	**dom 11/05/14**	**28**
42	**4 Entrega final**	**26 días**	**lun 12/05/14**	**dom 15/06/14**	**41**

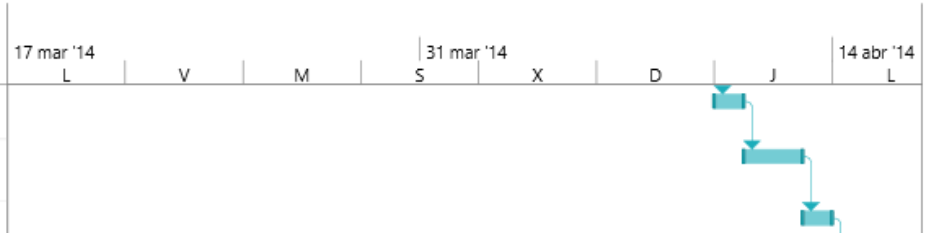


Cronograma del entregable 2 (PAC2, 13/04/2014)

Id	Nombre de tarea	Duración	Comienzo	Fin
8	**2 Análisis y creación de la BBDD del sistema operacional (PAC2)**	**21 días**	**lun 17/03/14**	**dom 13/04/14**
9	**2.1 Modelado conceptual**	**4 días**	**lun 17/03/14**	**jue 20/03/14**
10	2.1.1 Análisis de requisitos	1 día	lun 17/03/14	lun 17/03/14
11	2.1.2 Generación del esquema conceptual	1 día	mar 18/03/14	mar 18/03/14
12	2.1.3 Restricciones de integridad	2 días	mié 19/03/14	jue 20/03/14
13	**2.2 Diseño lógico estándar y específico**	**11 días**	**vie 21/03/14**	**vie 04/04/14**
14	2.2.1 Descripción del diseño lógico estándar	2 días	vie 21/03/14	sáb 22/03/14
15	2.2.2 Descripción del diseño lógico específico en Oracle.	1 día	dom 23/03/14	dom 23/03/14
16	2.2.3 Creación de la base de datos y las tablas relacionadas	1 día	lun 24/03/14	lun 24/03/14
17	2.2.4 Pruebas de las tablas de la base de datos	2 días	mar 25/03/14	mié 26/03/14
18	2.2.5 Diseño de los procedimientos CRUD (Create, Read, Update, Delete) mediante procedimientos almacenados	3 días	jue 27/03/14	sáb 29/03/14
19	2.2.6 Pruebas y validación de los procedimientos CRUD	1 día	dom 30/03/14	dom 30/03/14
20	2.2.7 Creación disparadores (triggers	3 días	lun 31/03/14	mié 02/04/14
21	2.2.8 Pruebas disparadores	2 días	jue 03/04/14	vie 04/04/14
22	**2.3 Diseño físico**	**5 días**	**sáb 05/04/14**	**jue 10/04/14**
23	2.3.1 Índices	1 día	sáb 05/04/14	sáb 05/04/14
24	2.3.2 Transacciones	2 días	dom 06/04/14	lun 07/04/14
25	2.3.3 Seguridad	2 días	mar 08/04/14	mié 09/04/14

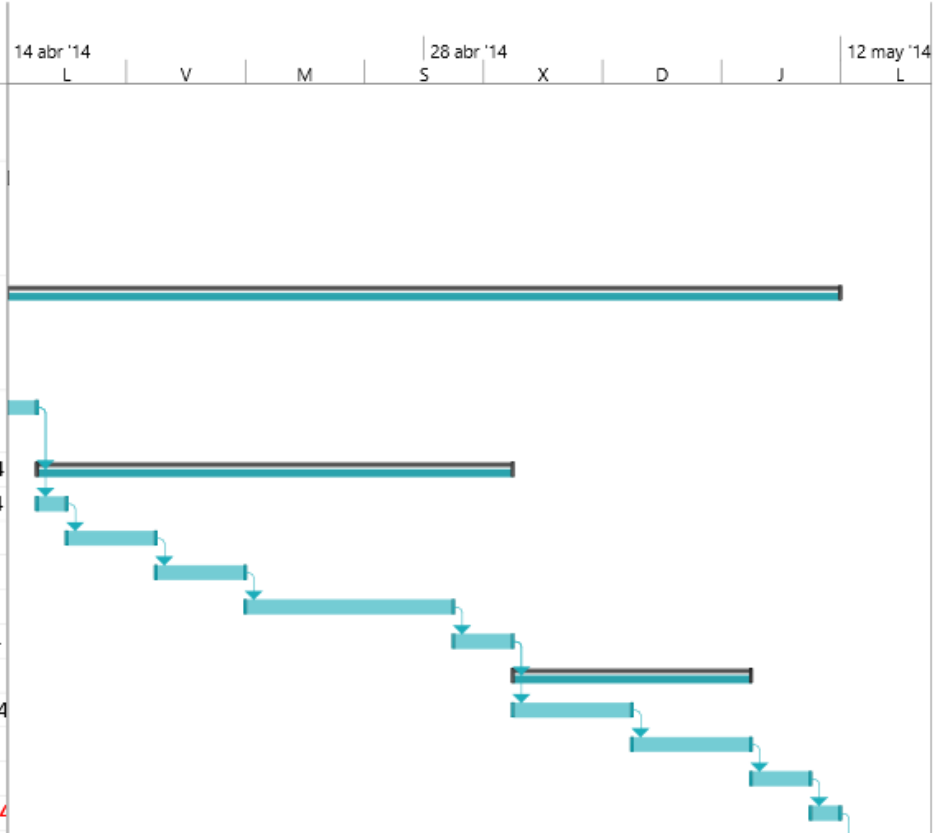


Id	Nombre de tarea	Duració	Comienzo	Fin
26	2.3.4 Validación y justificación del diseño físico	1 día	jue 10/04/14	jue 10/04/14
27	**2.4 Redacción memoria del proyecto**	2 días	vie 11/04/14	sáb 12/04/14
28	**2.5 Revisión y entrega**	1 día	dom 13/04/14	**dom 13/04/14**



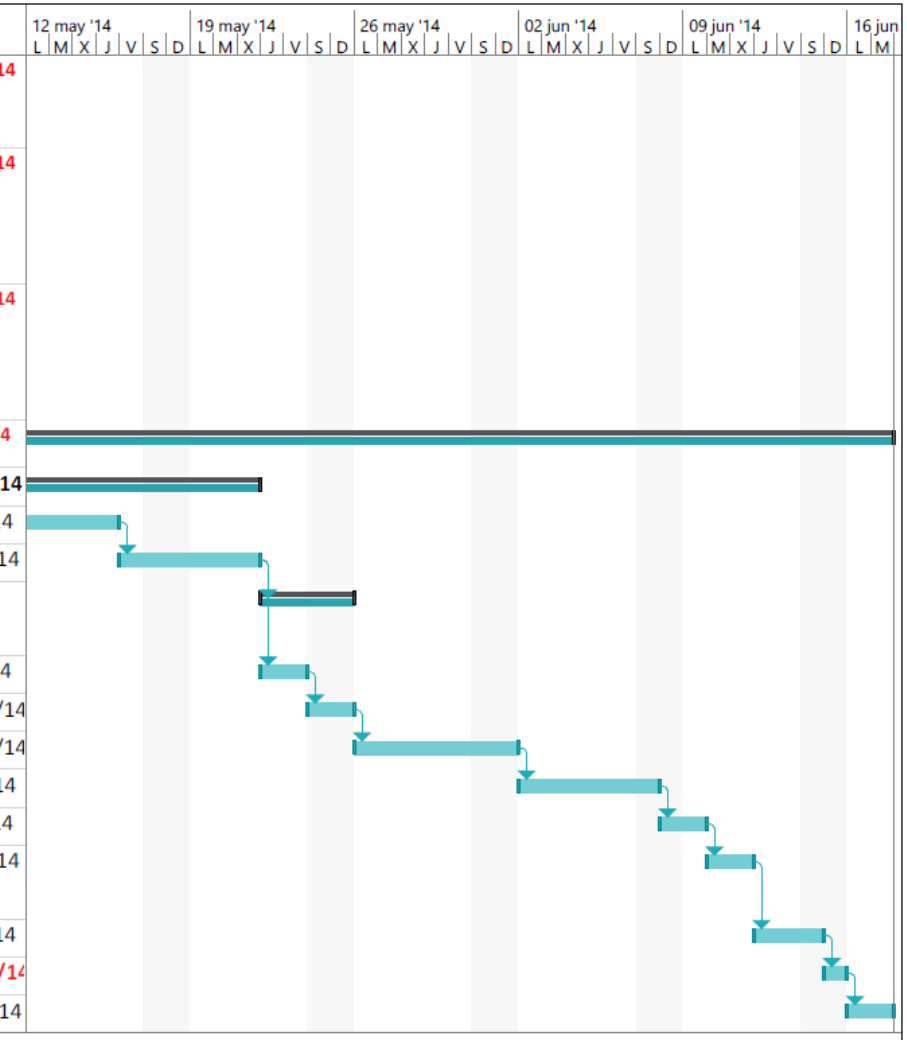
Cronograma del entregable 3 (PAC3, 11/05/2014)

Id	Nombre de tarea	Duració	Comienzo	Fin
1	**1 Elaboración del Plan de trabajo (PAC1)**	**12 días**	**sáb 01/03/14**	**dom 16/03/14**
8	**2 Análisis y creación de la BBDD del sistema operacional (PAC2)**	**21 días**	**lun 17/03/14**	**dom 13/04/14**
29	**3 Análisis y creación del Data Warehouse del sistema analítico (PAC3)**	**21 días**	**lun 14/04/14**	**dom 11/05/14**
30	**3.1 Estudio de los conceptos de DW y Metodología de diseño de**	1 día	lun 14/04/14	lun 14/04/14
31	**3.2 Diseño conceptual del sistema**	**12 día**	**mar 15/04/1**	**mié 30/04/14**
32	3.2.1 Análisis de requisitos	1 día	mar 15/04/1	mar 15/04/14
33	3.2.2 Identificación de hechos	3 días	mié 16/04/1	vie 18/04/14
34	3.2.3 Gránulos	2 días	sáb 19/04/14	lun 21/04/14
35	3.2.4 Dimensiones	5 días	mar 22/04/1	lun 28/04/14
36	3.2.5 Restricciones de integridad	2 días	mar 29/04/1	mié 30/04/14
37	**3.3 Diseño lógico**	**6 días**	**jue 01/05/14**	**jue 08/05/14**
38	3.3.1 Descripción del diseño lógico	3 días	jue 01/05/14	dom 04/05/14
39	3.3.2 Esquemas	4 días	lun 05/05/14	jue 08/05/14
40	**3.4 Redacción memoria del proyect**	2 días	vie 09/05/14	sáb 10/05/14
41	**3.5 Revisión y entrega**	1 día	dom 11/05/1	**dom 11/05/14**



Cronograma de la entrega final (15/06/2014)

Id	Nombre de tarea	Duración	Comienzo	Fin
1	**1 Elaboración del Plan de trabajo (PAC1)**	**12 días**	**sáb 01/03/14**	**dom 16/03/14**
8	**2 Análisis y creación de la BBDD del sistema operacional (PAC2)**	**21 días**	**lun 17/03/14**	**dom 13/04/14**
29	**3 Análisis y creación del Data Warehouse del sistema analítico (PAC3)**	**21 días**	**lun 14/04/14**	**dom 11/05/14**
42	**4 Entrega final**	**27 días**	**lun 12/05/14**	**mar 17/06/14**
43	**4.1 Diseño físico**	**8 días**	**lun 12/05/14**	**mié 21/05/14**
44	4.1.1 Definición del esquema multidi	4 días	lun 12/05/14	jue 15/05/14
45	4.1.2 Diseño e implementación del E	4 días	vie 16/05/14	mié 21/05/14
46	**4.2 Implementación del Almacén de datos (Data Warehouse)**	**3 días**	**jue 22/05/14**	**dom 25/05/14**
47	4.2.1 Implementación almacén de da	2 días	jue 22/05/14	vie 23/05/14
48	4.2.2 Carga del Almacén de datos	2 días	sáb 24/05/14	dom 25/05/14
49	**4.3 Diseño e implementación de las**	6 días	lun 26/05/14	dom 01/06/14
50	**4.4 Redacción final de la memoria c**	6 días	lun 02/06/14	sáb 07/06/14
51	**4.5 Confección de la presentación d**	2 días	dom 08/06/1	lun 09/06/14
52	**4.6 Redacción del autoinforme de evaluación de las competencias transversales**	2 días	mar 10/06/14	mié 11/06/14
53	**4.7 Preparación de los entregables**	3 días	jue 12/06/14	sáb 14/06/14
54	**4.8 Revisión y entrega final**	1 día	dom 15/06/1	**dom 15/06/14**
55	**4.9 Debate final**	2 días	lun 16/06/14	mar 17/06/14



1.4.2. Jornada de trabajo

En el cronograma planteado no se ha distinguido horario en función del tipo de día. No obstante si se ha tenido en cuenta la distribución de las diferentes tareas en función de la carga de trabajo asumible por día que ahora detallaremos:

Tipo día	Tiempo total
Lunes	2 h
Martes	3 h
Miércoles	2 h
Jueves	3 h
Viernes	4 h
Sábado	6 h
Domingo	4 h
TOTAL	34 h

Tabla 2. Tiempo de dedicación por día

La dedicación semanal es exclusiva para el proyecto y no interfiere con otras asignaturas y actividades que se desarrollarán de forma paralela

La aplicación de la jornada de trabajo al proyecto nos reporta el siguiente detalle:

Nombre de tarea	Duración	L	M	X	J	V	S	D	TOTAL horas
Elaboración del Plan de trabajo (PAC1)	**12 días**								**52**
Estudio proyecto y búsqueda información	7 días	1	1	1	1	1	2	1	**26**
Elección del software	1 día							1	**4**
Definición alcance del proyecto	2 días	1	1						**5**
Elaboración plan de trabajo	3 días			1	1	1			**9**
Redacción memoria del proyecto	1 día						1		**4**
Revisión y entrega	1 día							1	**4**

Tabla 3. Planificación detallada 1er. entregable

Nombre de tarea	Duración	L	M	X	J	V	S	D	TOTAL horas
Análisis y creación de la BBDD del sistema operacional (PAC2)	**21 días**								**88**
Modelado conceptual	**4 días**								**10**
Análisis de requisitos	1 día	1							2
Generación del esquema conceptual	1 día		1	1					5
Restricciones de integridad	2 días				1				3
Diseño lógico estándar y específico	**11 días**								**48**
Descripción del diseño lógico estándar	2 días					1	1		8
Descripción del diseño lógico específico en Oracle.	1 día							1	4
Creación de la base de datos y las tablas relacionadas	1 día	1							2
Pruebas de las tablas de la base de datos	2 días		1	1					5
Diseño de los procedimientos CRUD (*Create, Read, Update, Delete*) mediante procedimientos almacenados	3 días				1	1	1		11
Pruebas y validación de los procedimientos CRUD	1 día							1	4
Creación disparadores (*triggers*)	3 días	1	1	1					7
Pruebas disparadores	2 días				1	1			7
Diseño físico	**5 días**								**18**
Índices	1 día						1		4
Particiones/fragmentación	2 días	1						1	6
Seguridad	2 días		1	1					5
Validación y justificación del diseño físico	1 día				1				3
Redacción memoria del proyecto	2 días					1	1		**8**
Revisión y entrega	1 día							1	**4**

Tabla 4. Planificación detallada 2º entregable

Nombre de tarea	Duración	L	M	X	J	V	S	D	TOTAL horas
Análisis y creación del Data Warehouse del sistema analítico (PAC3)	**21 días**								**88**
Estudio de los conceptos de DW y Metodología de diseño de modelos OLAP a partir de modelos OLTP	1 día	1							2
Diseño conceptual del sistema	**12 días**								**49**
Proceso de negocio	1 día		1						3
Análisis de requisitos y def. gránulo	3 días			1	1	1			9
Dimensiones	2 días	1					1	1	10
Hechos	5 días	1	1	1	1	1	1	1	22
Esquema conceptual	2 días		1	1					5
Diseño lógico	**6 días**								**25**
Descripción del diseño lógico estándar	3 días				1	1	1	1	15
Descripción del diseño lógico específico	4 días	1	1	1	1				10
Redacción memoria del proyecto	2 días					1	1		**8**
Revisión y entrega	1 día							1	**4**

Tabla 5. Planificación detallada 3er entregable

Nombre de tarea	Duración	L	M	X	J	V	S	D	TOTAL horas
Entrega final	**26 días**								**110**
Diseño físico	**8 días**								**29**
Creación de la BBDD. Esquema, índices, particiones y seguridad	4 días	1	1	1	1				10
Diseño e implementación del ETL	4 días	1	1	1		1	1	1	19
Implementación del Almacén de datos (Data Warehouse)	**3 días**								**15**
Implementación almacén de datos	2 días				1	1			7
Carga del Almacén de datos	2 días						1	1	8
Diseño e implementación de las consultas	6 días	1	1	1	1	1	1	1	**22**
Redacción final de la memoria del trabajo	6 días	1	1	1	1	1	1		**18**
Confección de la presentación del trabajo	2 días	1						1	**6**
Redacción del autoinforme de evaluación de las competencias transversales	2 días		1	1					**5**
Preparación de los entregables del trabajo (BBDD y DW)	3 días				1	1	1		**11**
Revisión y entrega final	1 día							1	**4**
Debate Final	2 días	1	1						**5**

Tabla 6. Planificación detallada entrega final

El total de horas que se ha estimado para la realización de este proyecto es de 343 horas (52+88+88+115)

1.4.3. Recursos utilizados

Para la realización de este proyecto se han utilizado los siguientes recursos:

- ✓ Hardware
  - Ordenador de sobremesa (Windows 8.1)
  - Tableta IPAD (consulta información, correos, internet…)
- ✓ Software
  - Virtualización: Vmware Workstation v.10.0.2
  - Sistema gestor de bases de datos: Oracle Database 11g Enterprise Edition sobre Centos 6.5 x64
  - Ofimática: Microsoft Office 2013
  - Diagramas de Gantt: Microsoft Project 2013
  - Diagramas UML, E/R: Microsoft Visio 2013
  - Copias de seguridad: Sistema Operativo Windows 8.1

1.4.4. Riesgos y plan de contingencia

Los riesgos que se han valorado para la realización del proyecto son:

1. Avería de los sistemas de SI/TI para la realización del proyecto
2. Pérdida de tiempo ocasionada por contingencias comunes como: aumento jornada laboral, viajes laborales, enfermedades, compromisos familiares, etc…
3. Dificultad para el desarrollo de ciertos temas que haga que excedan el tiempo asignado en el cronograma

El plan de contingencias considerado contiene las siguientes medidas:

1. Averías sistemas SI/TI
   - ✓ Mantenimiento preventivo de los equipos de SI/TI
   - ✓ Medidas de seguridad de las TI:
     - Sistema de discos en Raid
     - Sistema de alimentación ininterrumpida (SAI)
     - Equipo alternativo de trabajo
     - Conexión alternativa (móvil) a internet
   - ✓ Medidas de protección de los SI:
     - Antivirus registrado y actualizado
     - Sistema de máquinas virtuales para el desarrollo del trabajo

- Copias de seguridad periódicas

2. Pérdidas de tiempo
   - ✓ Recuperación del tiempo del proyecto aumentando la dedicación en los días:
     - Lunes: Eliminación de otras tareas y aumento de la jornada efectiva a 4 horas
     - Miércoles: Ídem que el Lunes
     - Sábado: Eliminación de otras tareas y aumento de la jornada efectiva a 8 horas
     - Domingo: ídem que el sábado
   - ✓ Solicitud de vacaciones y permisos retribuidos en el trabajo para aumentar la carga lectiva a una jornada completa (8 horas)
3. Dificultad en el desarrollo de los temas
   - ✓ Consultas al tutor de la asignatura
   - ✓ Búsquedas bibliográficas
   - ✓ Consulta en internet
   - ✓ Investigación operativa

1.5. Breve sumario de productos obtenidos

Los productos que se han obtenido en este proyecto son:

- ✓ BBDD del sistema operacional compuesta por:
  - Tablas de datos
  - Funciones CRUD en forma de *Store Procedures*
  - Desencadenadores (*triggers*) para el control de los diferentes procesos
  - Definición de seguridad: Usuarios y permisos
- ✓ BBDD del Almacén de datos (*Data Warehouse*)
  - Tablas de datos
  - Procedimientos ETL en forma de *Store Procedures*
  - Cubos
  - Consultas OLAP de acuerdo con los requisitos requeridos
- ✓ Memoria del TFG
- ✓ Presentación virtual
- ✓ Autoinforme de competencias transversales

1.6. Breve descripción de los otros capítulos de la memoria

Los capítulos que componen la memoria son:

- ✓ Capítulo 1. Introducción: Se dispone la planificación del proyecto, la metodología a emplear y los recursos que se utilizarán.

- ✓ Capítulo 2. Análisis y creación de la BBDD del sistema operacional: Donde se abordará el diseño conceptual de la BBDD, el diseño lógico, el diseño físico, los procedimientos CRUD y las pruebas del sistema.
- ✓ Capítulo 3. Análisis y creación del almacén de datos (Data Warehouse) del sistema analítico: Donde se abordará el diseño conceptual del sistema analítico, el diseño lógico, el diseño físico, los procedimientos ETL y la creación de las consultas requeridas.
- ✓ Capítulo 4. Conclusiones: Valoración del TFG y conclusiones al finalizar el trabajo.
- ✓ Capítulo 5. Glosario: Relación y descripción de términos específicos vinculados al TFG.
- ✓ Capítulo 6. Bibliografía: Listado de referencias bibliográficas utilizadas en el TFG.
- ✓ Capítulo 7. Anexos: Información adicional (código, pruebas, etc...)

2. Análisis y creación de la BBDD del sistema operacional

2.1. Modelado conceptual

El modelado conceptual es la etapa posterior al análisis de requisitos. El objetivo de esta fase es obtener una representación conceptual de alto nivel (independiente de la tecnología con la que desarrolle la BBDD) en forma de diagrama en Lenguaje Unificado de Modelización (UML).

El modelado UML se ha realizado con la herramienta Microsoft Visio.
La nomenclatura seguida en los diagramas UML responde a las siguientes reglas:

- ✓ Todas las entidades, relaciones y atributos se expresan en inglés.
- ✓ El etiquetado de los tipos de entidad se realiza utilizando nombres en singular siguiendo la grafía Pascal.
- ✓ En el etiquetado de los atributos se utiliza nombres simples en singular evitando que aparezca el nombre de la entidad. Se utiliza la grafía Camel.
- ✓ En los atributos booleanos se utiliza un nombre singular precedido de la forma *is* (es).
- ✓ Los tipos de entidad se etiquetan, preferiblemente, con verbos en tercera persona del singular. Se utiliza la grafía Pascal.

2.1.1. Análisis de requisitos

A continuación redactaremos, de forma sintética, diferentes enunciados que establecemos como requisitos una vez realizada la lectura de los materiales del proyecto:

- ✓ Los países están compuestos por un conjunto de ciudades
- ✓ Las compañías están relacionadas con los países a través de la entidad ciudades.
- ✓ Diversas compañías puedes suministrar electricidad en una ciudad
- ✓ Diversas compañías pueden suministrar electricidad a un país
- ✓ Las compañías poseen contadores
- ✓ Un contador solo puede pertenecer a una compañía
- ✓ Un consumidor puede tener varios contadores (y por lo tanto varios contratos de servicio)
- ✓ Un consumidor puede ser atendido por diferentes compañías
- ✓ Los contadores están relacionados con las ciudades a través de la dirección donde está ubicado el contador
- ✓ Un consumidor está identificado por un documento que puede ser de varios tipos (pasaporte, DNI, visado, NIE…)
- ✓ Un consumidor solo tiene una cuenta bancaria para realizar los pagos del suministro eléctrico

- ✓ Los cambios de precio afectan a todas las ciudades donde la compañía presta servicio
- ✓ Los consumos de los contadores solo pueden ser leídos en intervalos de una hora
- ✓ Todos las acciones ABM (Alta, Baja , Modificación) se realizarán mediante procedimientos almacenados
- ✓ Una tabla LOG deber registrar para cada llamada a un procedimiento almacenado: el nombre del procedimiento ejecutado, los parámetros de entrada, los parámetros de salida, y el resultado global de la ejecución del procedimiento.

2.1.2. Generación del esquema conceptual

Diagrama UML de casos de uso de la BBDD operacional

El diagrama UML de los casos de uso de la BBDD consiste en:

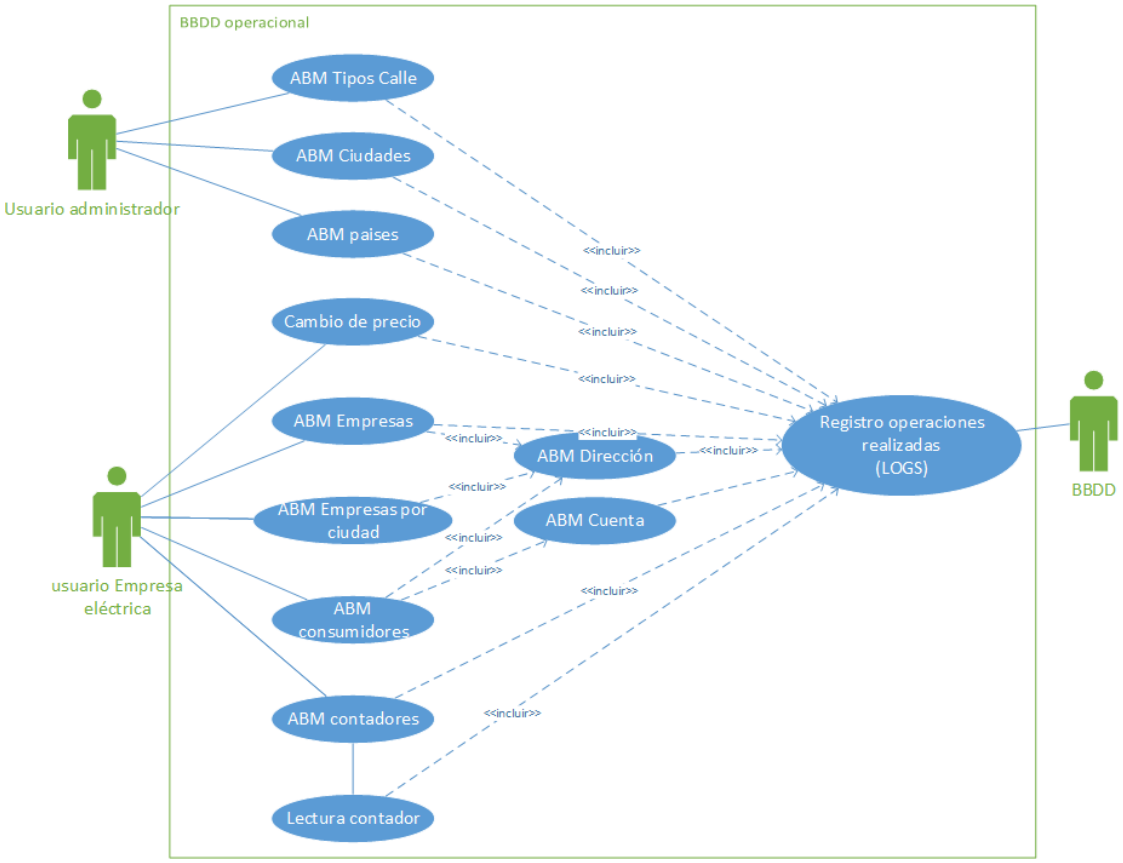


Ilustración 2. Casos de uso BBDD operacional

Diagrama UML de la BBDD operacional

El diseño UML de la base de datos operacional se muestra en la página siguiente:

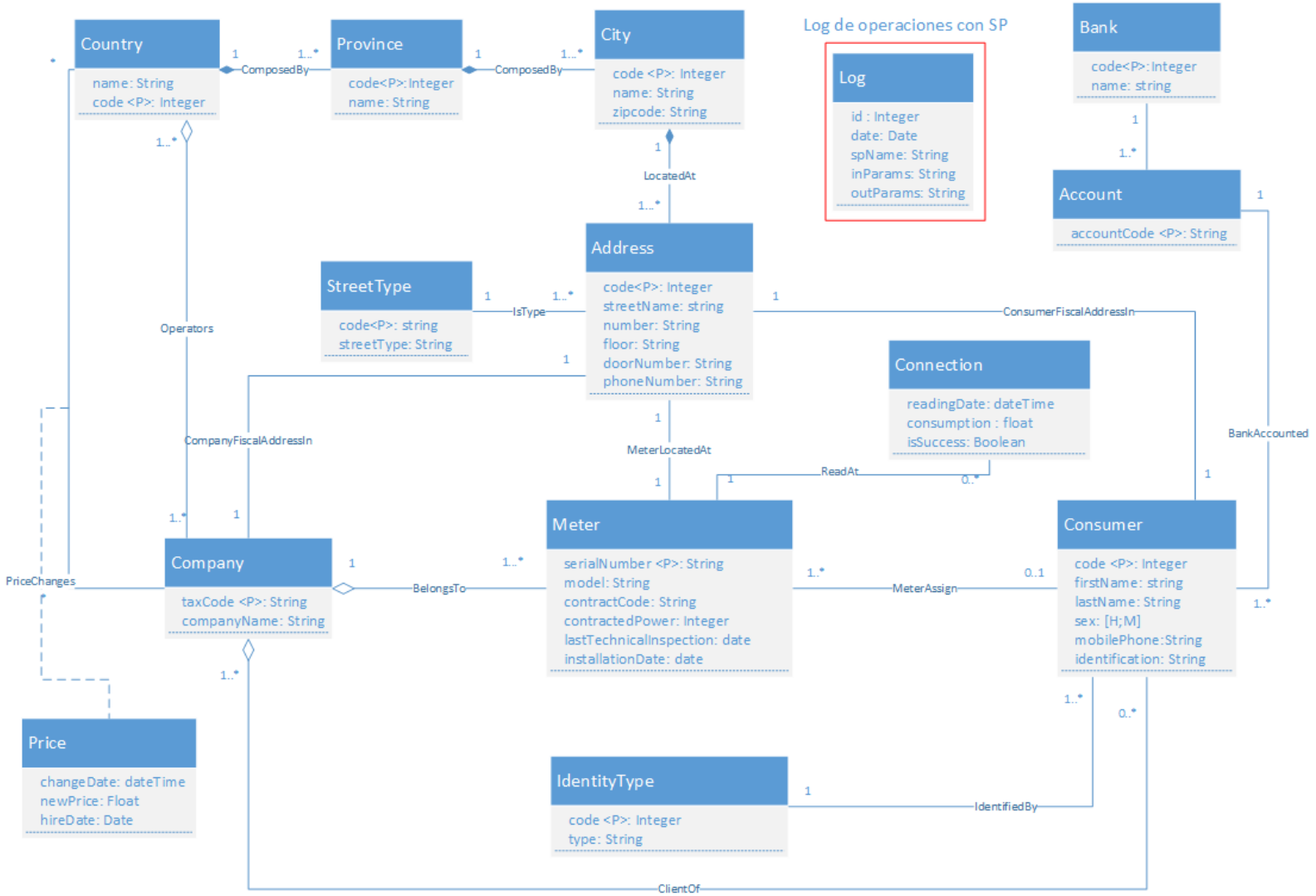


Ilustración 3. Diagrama UML

2.1.3. Restricciones de integridad

Las restricciones de integridad comprenden:

- ✓ Restricciones en los tipos de entidad
- ✓ Restricciones en los atributos ya sea por el dominio o por atributos derivados
- ✓ Restricciones en los tipos de relaciones o cardinalidad de los tipos

En relación con los tipos de entidad los aspectos más relevantes tienen que ver con las entidades que se han modelado con cardinalidad unitaria.

De esta forma, sólo se permite una entidad *Account* para cada *Consumer* porque se entiende que sólo es relevante almacenar la cuenta del cliente con la que se van a realizar los pagos. No obstante, 2 clientes distintos podrían tener la misma cuenta de pago (cuenta con dos titulares).

Igualmente un cliente sólo puede tener un tipo de identificación *Identity*, aunque se dé el caso, por ejemplo de que un cliente disponga tanto de DNI como de pasaporte. De igual forma se comporta la entidad *Address*. Un contador (*Meter*), una empresa (*Company*), y un cliente (*Consumer*) sólo pueden tener una dirección (*Address*).

Un caso aparte es la restricción sobre la lectura de los contadores (sólo se permite la lectura de los contadores en intervalos de una hora), y cada lectura comporta un registro de tipo log (entidad *Connection*). Este tipo de registro se aplicará mediante un desencadenador o *trigger*.

En cuanto a las restricciones de los atributos el apartado más relevante podría ser los aspectos derivados de la restricción de los atributos de identificación fiscal de las entidades (por ejemplo incluyendo alguna validación, si es posible, del tipo letra del DNI. Además los nombres identificadores de los clientes y de las empresas no pueden ser nulos. Adicionalmente se han establecido identificadores sintéticos para las entidades *Consumer, Address, Connection, Price, Bank* y *StreetType*.

Se ha incluido el modelado de la entidad Log, que si bien no interviene en el desarrollo del resto de las entidades sí que opera a nivel de registro de todas las operaciones que se realizan en la BBDD con procedimientos almacenados (Store Procedures).

2.2. Diseño lógico estándar y específico

EL objetivo de esta fase es obtener un esquema lógico de la base de datos, expresado en modelos lógicos relacionales, a partir del esquema conceptual desarrollado en la anterior fase, teniendo en cuenta las teorías sobre normalización de los modelos lógicos relacionales.

2.2.1. Descripción del diseño lógico estándar

Para la definición del diseño lógico utilizaremos las siguientes convenciones [4]:

- ✓ Los atributos que son clave principal se marcan con un subrayado.
- ✓ Los atributos que tienen valores no nulos se marcan en negrita
- ✓ Las claves candidatas se marcarán con un subrayado con líneas discontinuas.

Respecto a las conversiones realizadas tenemos:

- ✓ Las relaciones binarias con cardinalidad 1 a 1..* se traducen con una clave externa en la relación débil [1].
- ✓ Las relaciones binarias con cardinalidad 1 a 1 se traducen, o bien incorporando la relación débil como atributos a la relación fuerte, o manteniendo ambas relaciones [1].
- ✓ Las relaciones binarias con cardinalidad 1..* a 1..* requieren la creación de una nueva entidad que tiene como claves primarias, claves foráneas de ambas relaciones [1].

De esta manera se obtiene la siguiente conversión del esquema conceptual:

Country (countryCode, **countryName**)

Province (provinceId, **provinceCode, provinceName**, **countryCode**)

{CountryCode} es clave externa de *Country*

En la entidad *Province* hemos generado una clave sintética, *Surrogate Key* [5] *provinceId* (que mecanizaremos con la secuencia y el *trigger* correspondiente), y fijaremos como UNIQUE el binomio *provinceCode*, *CountryCode*, todo ello para facilitar la relación de integridad referencial con la entidad *City*.

City (cityCode, **cityName**, **zipcode**, **provinceCode)**

{ProvinceCode} es clave externa de Province

Address (addressCode, **streetCode**, **streetName**, addressNumber, floor, doorNumber, phoneNumber, **cityCode**)

{cityCode} es clave externa de *City*

{StreetCode} es clave externa de *StreetType*

En la entidad *Address* hemos generado una clave sintética *addressCode* (que mecanizaremos con la secuencia y el trigger correspondiente).

StreetType (streetTypeCode, **streetTypeName**)

Bank (bankCode, **bankName**)

IdentityType (identityCode, **identityType**)

Consumer (consumerCode, **name**, **surname**, **sex**, mobilePhone, **addressCode**, **identityCode**, **identificationNumber**, **bankCode**, **accountCode**)

{addressCode} es clave externa de *Address*

{BankCode} es clave externa de *Bank*

{IdentityCode} es clave externa de *IdentityType*

En la entidad Consumer hemos generado una clave sintética *consumerCode* (que mecanizaremos con la secuencia y el *trigger* correspondiente)

Company(companyTaxCode, **companyName**, **adressCode**)

{AddressCode} es clave externa de Address

Meter (serialNumber, **model**, contractCode, **contractedPower**, lastTechnicalInspection, **installationDate**, **companyCode**, consumerCode, **addressCode**)

{CompanyCode} es clave externa de *Company*

{ConsumerCode} es clave externa de *Consumer*

{AddressCode} es clave externa de *Address*

Connection (readingDate, meterSerialName, instantConsumption, **isSuccess**)

{MeterSerialName} es clave externa de *Meter*

Price (changeDate, countryCode, companyCode, **newPrice**)

{CountryCode} es clave externa de *Country*

{CompanyCode} es clave externa de *Company*

Operators (countryCode, companyCode)

{CountryCode} es clave externa de *Country*

{CompanyCode} es clave externa de *Company*

Clients (companyCode, consumerCode, hireDate)

{companyCode} es clave externa de *Company*

{consumerCode} es clave externa de *Consumer*

LuzLog(**idLog**, execDate, **spName**, inParams, outParams, **user**)

En la entidad *LuzLog* hemos generado una clave sintética *idlog* (que mecanizaremos con la secuencia y el *trigger* correspondiente)

2.2.2. Descripción del diseño lógico específico en Oracle

El objetivo de esta fase es realizar la traducción del modelo lógico estándar anterior en la nomenclatura específica de la solución adoptada para realizar la implementación de la BBDD, en nuestro caso en el sistema ORACLE para lo cual hemos contado con la inestimable ayuda de uno de los mejores manuales de Oracle PL/SQL [6].

Se han clasificado las entidades en:

- ✓ Principales: Son las entidades más importantes del diseño y reflejan los aspectos más importantes
- ✓ Intermedias: Son entidades de apoyo
- ✓ Auxiliares: Son entidades que proporcionan valores definidos a algún atributo de una entidad principal o intermedia

El código completo de la creación de las tablas se acompaña en el anexo I.

Nombre de la tabla	Tipo entidad	Descripción	Observaciones
Country	Intermedia	Contiene la representación de los países donde se presta servicio	Códigos según ISO 3166
Province	Intermedia	Contiene las regiones (provincias) de las que se componen los países	Códigos INE o similar Clave sintética
City	Intermedia	Almacena las ciudades de los diferentes países	Códigos INE o similar
StreetType	Auxiliar	Permite calificar el tipo de vía de la entidad dirección	Códigos INE
Address	Principal	Almacena las direcciones fiscales de las compañías suministradoras, las direcciones fiscales de los consumidores, y las direcciones donde se encuentran instalados los contadores	Clave sintética
Bank	Intermedia	Almacena las entidades bancarias para calificar la cuenta bancaria de los consumidores	Códigos según ISO 9362 SWIFT/BIC
IdentityType	Auxiliar	Permite calificar el tipo de identificación del consumidor (dni, cif, pasaporte…)	
Consumer	Principal	Contiene los datos básicos de los consumidores	Clave sintética
Company	Principal	Contiene los datos básicos de las compañías suministradoras	Código NIF IVA Europeo
Meter	Principal	Contiene los datos básicos de los contadores inteligentes	
Connection	Intermedia	Almacena los datos de las lecturas realizadas en los contadores inteligentes	Las lecturas de los contadores son números enteros
Price	Intermedia	Contiene los precios de la energía para cada país	Los precios de la energía se expresan con 6 decimales
Clients	Principal	Almacena los clientes de cada compañía suministradora	
Operators	Intermedia	Almacena las compañías suministradoras de cada país	
LuzLog	Intermedia	Es una tabla auxiliar que recoge la información de la ejecución de los procedimientos ABM	Clave sintética

Tabla 7. Tablas de la BBDD operacional

2.2.3. Pruebas de las tablas de la BBDD

Para la realización de las pruebas se han introducido valores de prueba en las diferentes tablas.
El código completo de las pruebas se acompaña como Anexo II.

Las validaciones realizadas en las tablas mediante la introducción de datos comprenden:

(1) La entidad es capaz de representar los datos

(2) Las reglas de integridad no permiten la introducción de entidades con valores duplicados (UNIQUE, o PRIMARY KEY)

(3) No se pueden introducir valores nulos (restricciones NOT NULL)

(4) Claves externas (FOREIGN KEY)

(5) Otras reglas de negocio (a especificar)

Tabla	Valores de las entidades (1)	Repetición de campos de clave (2)	Valores nulos (3)	Claves externas (4)	Otras reglas de negocio (5)	Observaciones
COUNTRY	OK	OK	OK	--	--	
PROVINCE	OK	OK	OK	OK	--	Secuencia OK Trigger OK
CITY	OK	OK	OK	OK	--	Zipcode puede requerir una nueva entidad o pasarse a la entidad ADDRESS
STREETTYPE	OK	OK	OK	--	--	
ADDRESS	OK	OK	OK	OK	--	Secuencia OK Trigger OK
BANK	OK	OK	OK	--	--	
IDENTITYTYPE	OK	OK	OK	--	--	El valor de IdentityType puede ser corto
CONSUMER	OK	OK	OK	OK	OK	Secuencia OK Trigger OK
COMPANY	OK	OK	OK	OK	--	Quizás el TAXCODE no sea una buena clave. Estudiar una clave sintética
METER	OK	OK	OK	OK	--	Quizás el SN no sea una buena clave. Estudiar una clave sintética
CONNECTION	OK	OK	OK	OK	Y o N en isSuccess	Trigger al grabar datos para eliminar la lectura
PRICE	OK	OK	OK	OK		
CLIENTS	OK	OK	OK	OK	--	Es una tabla redundante Estudiar su eliminación
OPERATORS	OK	OK	OK	OK	--	
LUZLOG	OK	OK	OK	--	--	

Tabla 8. Validaciones realizadas en las tablas

2.2.4. Diseño de los procedimientos ABM (Alta, Baja, Modificación) mediante procedimientos almacenados

Los procedimientos almacenados diseñados incorporan las siguientes características;

- ✓ Control de errores ante la introducción de parámetros con valor nulo.
- ✓ Control de errores ante la introducción de parámetros con valores manifiestamente incorrectos (campos de caracteres con un largo mayor del requerido)
- ✓ Control de errores ante la introducción de valores con clave primaria duplicada.

- ✓ Control de errores ante registros que incumplan las normas de integridad referencial (claves externas)
- ✓ Control de errores en algunas fechas de acuerdo con la lógica de negocio.
- ✓ Devolución de un parámetro que indica si el procedimiento se ha ejecutado sin errores (valor OK si el procedimiento se ha ejecutado correctamente, y en caso contrario el código del error correspondiente)
- ✓ Devolución de parámetros necesarios para la gestión de la identidad en los casos de creación de claves sintéticas.
- ✓ Registro de las operaciones realizadas en la tabla de LOG.

No se han desarrollado procedimientos almacenados para las operaciones de lectura (READ).

El código de los procedimientos almacenados de la BBDD se acompaña como anexo III.

Los procedimientos creados han sido:

#	Nombre procedimiento	Tablas Afectadas	Propósito	Observaciones
1	INS_LUZLOG	LUZLOG	Insertar datos	Introduce la fecha y hora de la ejecución del SP y el usuario de forma automática. Se llama en la ejecución de todos los SP
2	INS_COUNTRY	COUNTRY	Insertar datos	
3	INS_PROVINCE	PROVINCE	Insertar datos	Devuelve ProvinceId
4	INS_CITY	CITY	Insertar datos	Devuelve CityCode
5	INS_STREETTYPE	STREETTYPE	Insertar datos	
6	INS_ADDRESS	ADDRESS	Insertar datos	Devuelve AddressCode
7	INS_BANK	BANK	Insertar datos	
8	INS_IDENTITYTYPE	IDENTITYTYPE	Insertar datos	
9	INS_CONSUMER	CONSUMER	Insertar datos	Devuelve ConsumerCode
10	INS_COMPANY	COMPANY	Insertar datos	
11	INS_METER	METER	Insertar datos	Se añade lógica de control para evitar poder introducir una fecha de revisión del contador anterior a la fecha de instalación del mismo
12	INS_CONNECTION	CONNECTION	Insertar datos	
13	INS_PRICE	PRICE	Insertar datos	Se añade lógica de control para evitar realizar un cambio de precio con efecto retroactivo en el tiempo
14	INS_CLIENTS	CLIENTS	Insertar datos	
15	INS_OPERATORS	OPERATORS	Insertar datos	

Tabla 9. Procedimientos Almacenados para la inserción de datos en las entidades (continuación)

#	Nombre procedimiento	Tablas Afectadas	Propósito	Observaciones
1	DEL_LUZLOG	LUZLOG	Borrar datos	Se implementa aunque no tiene sentido
2	DEL_COUNTRY	COUNTRY	Borrar datos	
3	DEL_PROVINCE	PROVINCE	Borrar datos	
4	DEL_CITY	CITY	Borrar datos	
5	DEL_STREETTYPE	STREETTYPE	Borrar datos	
6	DEL_ADDRESS	ADDRESS	Borrar datos	
7	DEL_BANK	BANK	Borrar datos	
8	DEL_IDENTITYTYPE	IDENTITYTYPE	Borrar datos	
9	DEL_CONSUMER	CONSUMER	Borrar datos	
10	DEL_COMPANY	COMPANY	Borrar datos	

Tabla 10. Procedimientos almacenados para el borrado de datos

#	Nombre procedimiento	Tablas Afectadas	Propósito	Observaciones
11	DEL_METER	METER	Borrar datos	
12	DEL_CONNECTION	CONNECTION	Borrar datos	
13	DEL_PRICE	PRICE	Borrar datos	
14	DEL_CLIENTS	CLIENTS	Borrar datos	
15	DEL_OPERATORS	OPERATORS	Borrar datos	

Tabla 11. Procedimientos almacenados para el borrado de datos (continuación)

#	Nombre procedimiento	Tablas Afectadas	Propósito	Observaciones
1	UPD_COUNTRY	COUNTRY	Actualizar datos	
2	UPD_PROVINCE	PROVINCE	Actualizar datos	
3	UPD_CITY	CITY	Actualizar datos	
4	UPD_STREETTYPE	STREETTYPE	Actualizar datos	
5	UPD_ADDRESS	ADDRESS	Actualizar datos	
6	UPD_BANK	BANK	Actualizar datos	
7	UPD_IDENTITYTYPE	IDENTITYTYPE	Actualizar datos	
8	UPD_CONSUMER	CONSUMER	Actualizar datos	

Tabla 12. Procedimientos almacenados para la actualización de datos

#	Nombre procedimiento	Tablas Afectadas	Propósito	Observaciones
9	UPD_COMPANY	COMPANY	Actualizar datos	
10	UPD_METER	METER	Actualizar datos	Se añade lógica de control para evitar poder introducir una fecha de revisión del contador anterior a la fecha de instalación del mismo
11	UPD_CONNECTION	CONNECTION	Actualizar datos	
12	UPD_PRICE	PRICE	Actualizar datos	Se añade lógica de control para evitar realizar un cambio de precio con efecto retroactivo en el tiempo
13	UPD_CLIENTS	CLIENTS	Actualizar datos	

Tabla 13. Procedimientos almacenados para la actualización de datos (continuación)

2.2.5. Pruebas y validación de los procedimientos ABM

La validación de los SP CRUD se ha realizado ejecutando diversos juegos de pruebas:

- Juego de pruebas 1:
  - ✓ Borrar datos con parámetros correctos: Revisar error y registro en tabla Log
  - ✓ Borrar datos con parámetros incorrectos: Revisar error y registro en tabla Log
- Juego de Pruebas 2
  - ✓ Insertar datos con todos los parámetros correctos: Revisar error y registro en tabla Log
  - ✓ Insertar datos con parámetros incorrectos: Revisar error y registro en tabla Log
- Juego de Pruebas 3
  - ✓ Actualizar datos con todos los parámetros correctos: Revisar error y registro en tabla Log
  - ✓ Actualizar datos con parámetros incorrectos: Revisar error y registro en tabla Log

Los parámetros que se han comprobado son:

(1) Compilación correcta de la SP

(2) Actuación correcta sobre la tabla de destino

(3) Control de errores en el propio procedimiento

(4) Control de errores derivados de las restricciones propias de la tabla destino

(5) Registro de los datos en la tabla LOG

(6) Devolución de los parámetros adecuados (solo en aquellos procedimientos que lo requieren)

El conjunto completo de los scripts de pruebas de los procedimientos almacenados se incorporará como Anexo IV.

#	Procedimiento almacenado	Compilación correcta (1)	Tabla destino (2)	Control errores SP (3)	Control errores restricciones (4)	Registro en la tabla LOG (5)	Observaciones
1	INS_LUZLOG	OK	OK	OK	OK	OK	
2	INS_COUNTRY	OK	OK	OK	OK	OK	
3	INS_PROVINCE	OK	OK	OK	OK	OK	
4	INS_CITY	OK	OK	OK	OK	OK	
5	INS_STREETTYPE	OK	OK	OK	OK	OK	
6	INS_ADDRESS	OK	OK	OK	OK	OK	
7	INS_BANK	OK	OK	OK	OK	OK	
8	INS_IDENTITYTYPE	OK	OK	OK	OK	OK	
9	INS_CONSUMER	OK	OK	OK	OK	OK	
10	INS_COMPANY	OK	OK	OK	OK	OK	
11	INS_METER	OK	OK	OK	OK	OK	
12	INS_CONNECTION	OK	OK	OK	OK	OK	
13	INS_PRICE	OK	OK	OK	OK	OK	
14	INS_CLIENTS	OK	OK	OK	OK	OK	
15	INS_OPERATORS	OK	OK	OK	OK	OK	
16	DEL_LUZLOG	OK	OK	OK	OK	OK	
17	DEL_COUNTRY	OK	OK	OK	OK	OK	
18	DEL_PROVINCE	OK	OK	OK	OK	OK	
19	DEL_CITY	OK	OK	OK	OK	OK	
20	DEL_STREETTYPE	OK	OK	OK	OK	OK	
21	DEL_ADDRESS	OK	OK	OK	OK	OK	
22	DEL_BANK	OK	OK	OK	OK	OK	
23	DEL_IDENTITYTYPE	OK	OK	OK	OK	OK	
24	DEL_CONSUMER	OK	OK	OK	OK	OK	
25	DEL_COMPANY	OK	OK	OK	OK	OK	
26	DEL_METER	OK	OK	OK	OK	OK	
27	DEL_CONNECTION	OK	OK	OK	OK	OK	
28	DEL_PRICE	OK	OK	OK	OK	OK	
29	DEL_CLIENTS	OK	OK	OK	OK	OK	
30	DEL_OPERATORS	OK	OK	OK	OK	OK	
31	UPD_COUNTRY	OK	OK	OK	OK	OK	
32	UPD_PROVINCE	OK	OK	OK	OK	OK	
33	UPD_CITY	OK	OK	OK	OK	OK	
34	UPD_STREETTYPE	OK	OK	OK	OK	OK	
35	UPD_ADDRESS	OK	OK	OK	OK	OK	
36	UPD_BANK	OK	OK	OK	OK	OK	
37	UPD_IDENTITYTYPE	OK	OK	OK	OK	OK	
38	UPD_CONSUMER	OK	OK	OK	OK	OK	
39	UPD_COMPANY	OK	OK	OK	OK	OK	
40	UPD_METER	OK	OK	OK	OK	OK	
41	UPD_CONNECTION	OK	OK	OK	OK	OK	
42	UPD_PRICE	OK	OK	OK	OK	OK	
43	UPD_CLIENTS	OK	OK	OK	OK	OK	

Tabla 14. Pruebas Procedimientos ABM

2.2.6. Creación disparadores (triggers)

La lista de disparadores de la BBDD es:

#	Nombre	Tabla afectada	Objeto	Observaciones
1	Insert_provinceId_Province	Province	Introducir la clave sintética de la entidad	
2	Insert_AddressCode_Address	Address	Introducir la clave sintética de la entidad	
3	Insert_consumerCode_Consumer	Consumer	Introducir la clave sintética de la entidad	
4	Insert_idLog_LuzLog	LuzLog	Introducir la clave sintética de la entidad	
5	Ins_Consum_Connection	Connection	Grabar la lectura del contador en función de las normas de privacidad establecidas	Salvo los datos en las lecturas erróneas de los contadores.

Tabla 15. Disparadores

El listado completo de los scripts de los disparadores se encuentra como anexo I a este documento, junto con los scripts de creación de las tablas correspondientes.

2.2.7. Pruebas y validaciones de los disparadores

Se han realizado las siguientes pruebas de control de los disparadores:

- ✓ Pruebas durante la inserción de nuevos valores en la tabla
- ✓ Pruebas durante la actualización de los valores de la tabla

#	Nombre	Tabla afectada	Inserción	Actualización	Observaciones
1	Insert_provinceId_Province	Province	OK	--	
2	Insert_AddressCode_Address	Address	OK	--	
3	Insert_consumerCode_Consumer	Consumer	OK	--	
4	Insert_idLog_LuzLog	LuzLog	OK	--	
5	Ins_Consum_Connection	Connection	OK	OK	(1)

Tabla 16. Pruebas disparadores

La mayoría de los *triggers* que se han programado tienen que ver con la inserción de una clave sintética (valor de una secuencia) como clave principal de la entidad por lo que la prueba al actualizar este valor no tiene sentido (los disparadores actúan solo al insertar un valor nuevo).

(1) El disparador *Ins_consum_connection* tiene un comportamiento diferente. Este disparador se utiliza para modelar la restricción de privacidad de lectura de los contadores. Se entiende que cada vez que se realiza una lectura de un contador se produce la inserción de un registro en esta tabla. La lógica del disparador actúa discriminando en función de la hora de la última lectura si realiza una escritura correcta del consumo del contador, o por el contrario produce una lectura ciega (con valor de consumo NULL y éxito de la operación igual a "N"). De esta forma solo se registran los valores de consumo si no se han producido lecturas exitosas de ese mismo contador en un plazo superior a una hora.
Evidentemente, esta restricción también afecta al registro de ejecución de la tabla LuzLog, de manera que, en este registro se omiten los datos de valor de consumo.

2.3. Diseño físico

La fase de diseño físico es la última etapa en la construcción de una BBDD. El objetivo de esta fase es dotar a nuestro diseño de la estructura física que le permita almacenar la información en un soporte físico no volátil, optimizar las políticas de acceso a la base de datos (creación de índices) y establecer las condiciones de seguridad para el acceso de los datos.

Los pasos que se seguirán son:

- ✓ La preparación de un servidor para la instalación del sistema gestor de base de datos ORACLE.
- ✓ La creación de la BBDD, los metadatos necesarios en el sistema ORACLE para soportar el diseño lógico realizado
- ✓ La aplicación del diseño lógico según los apartados anteriores.
- ✓ La optimización de las políticas de acceso a los datos (índices y fragmentación)
- ✓ La estructura de seguridad para el acceso a los datos.

2.3.1. Preparación del servidor, instalación del sistema Oracle y creación de la BBDD

Para la realización de este apartado hemos realizado las siguientes tareas previas:

- ✓ Instalar una distribución Linux Centos v.6.5 x64 en una máquina virtual de VMware con las siguientes características:
  - 2 procesadores
  - 4096 Mb RAM
  - 40 GB HDD (originariamente eran 20 GB, pero ha sido necesaria una extensión con las herramientas LVM)

  La instalación de Oracle en Linux requiere (para obtener el soporte oficial de Oracle) la instalación en un Sistema Red Hat Linux, Suse Linux, u Oracle Linux (basado a su vez en Red Hat, pero con ciertos parches propios de Oracle). Se ha seleccionado Centos porque se trata de una distribución de Linux basada en los paquetes de Red Hat especialmente dedicada a la comunidad Open Source. De manera que podemos asimilar Centos como un clon de una instalación Red Hat, plenamente compatible con Oracle.
- ✓ Instalar y configurar Oracle Database 11g Release 2 (11.2.0.1.0) for Linux x86_64 en Centos :
  - Instalar prerrequisitos y paquetes necesarios con yum install
  - Configurar grupo de usuarios y usuario de Oracle
  - Configurar entorno (variables path, export, ….)
  - Instalar producto
  - Establecer las rutinas de encendido y apagado… (en /inet.d/)
  - Pruebas y problemas diversos
- ✓ Instalar y configurar un cliente de Oracle Database 11g Release 2 (11.2.0.1.0) Linux x86_64 en Centos [7]

- Prerrequisitos y paquetes necesarios
- Instalación
- Configuración

✓ Instalar y configurar un Java JDK (en nuestro caso una versión 8) para Linux x64 (paquete rpm)

✓ Instalar SQL Developer v.4.0.1 para Linux x64:

- Prerrequisitos: Java JDK y DISPLAY
- Configuración y conexión producto

✓ Crear una nueva BBDD con bdca (DBCA en inglés de Oracle *Data Base Creation Assistant*) [7]

Hemos decidido crear la base de datos con esta herramienta porque nos proporciona las siguientes ventajas:

✓ Optimización del rendimiento

✓ Verificar la correcta configuración de los discos compartidos para cada *tablespace*

✓ Configuración de los servicios de red Oracle

✓ Arrancado de la instancia de la base de datos y de los *listener* correspondientes.

Los pasos de la creación de la BBDD con la herramienta DBCA han consistido en:

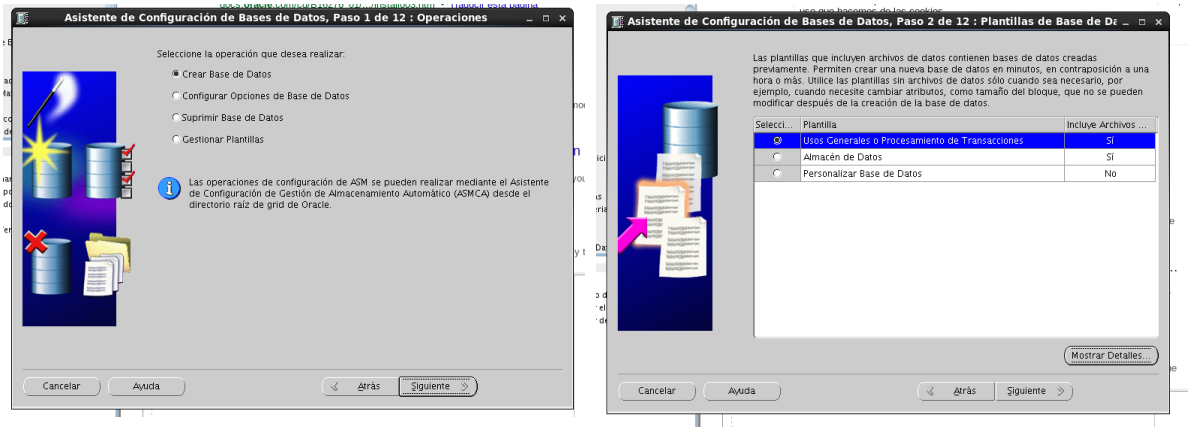


Ilustración 4. Creación BBDD con dbca. 1

La base de datos se va a crear para propósitos operacionales (OLTP) por lo que seleccionamos crear una base de datos para usos generales o procesamiento de transacciones.

El nombre de la base de datos será “luz”.

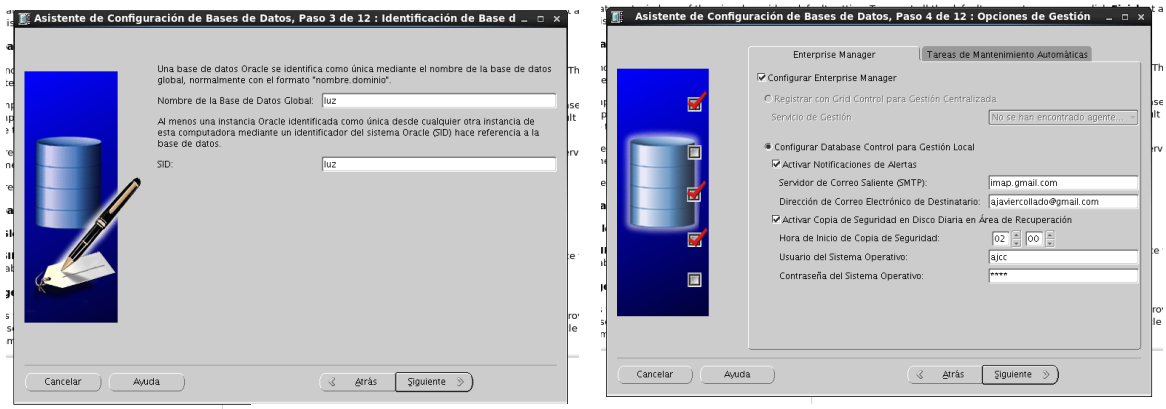


Ilustración 5. Creación BBDD operacional con dbca. 2.

Se activan las opciones de notificación por correo electrónico de incidencias, y las copias de seguridad

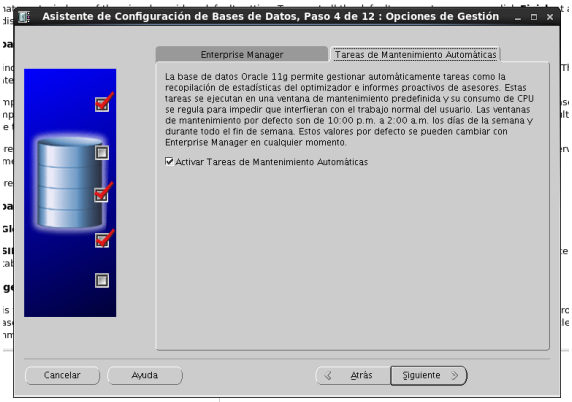


Ilustración 6. Creación BBDD operacional con dbca. 4

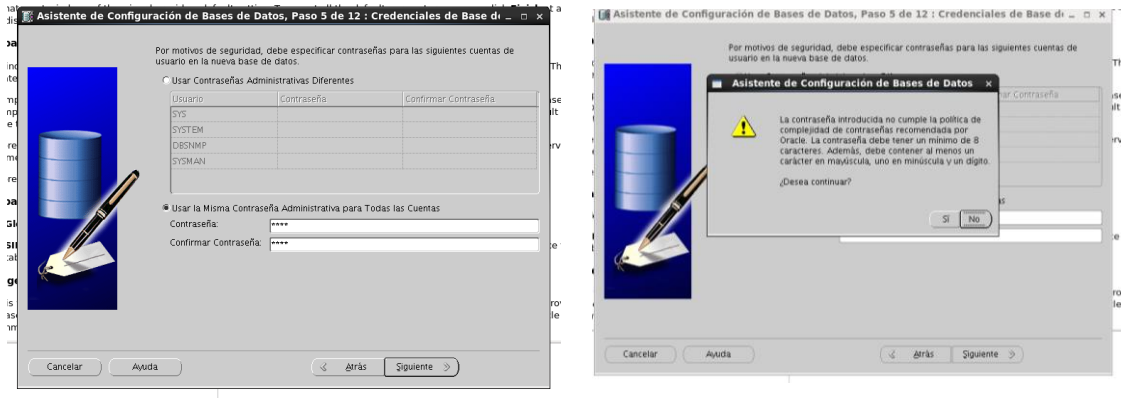


Ilustración 7. Creación BBDD operacional con dbca. 3

A pesar de ser más seguro el establecimiento de usuarios y contraseñas diferentes para cada cuenta de la BBDD, escogemos, por simplicidad, ejecutar todos los procesos con una misma contraseña (“asus”). La contraseña escogida debe cumplir unos requisitos de complejidad (que en este caso no se dan) de ahí el mensaje de advertencia de la segunda imagen de la ilustración 7.

El siguiente paso es determinar la ubicación de los ficheros de la BBDD

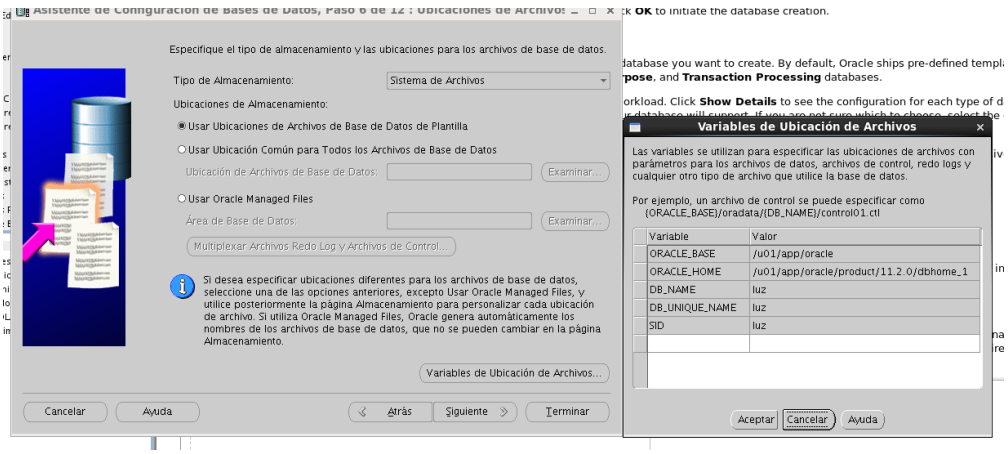


Ilustración 8. Creación BBDD operacional con dbca. 5

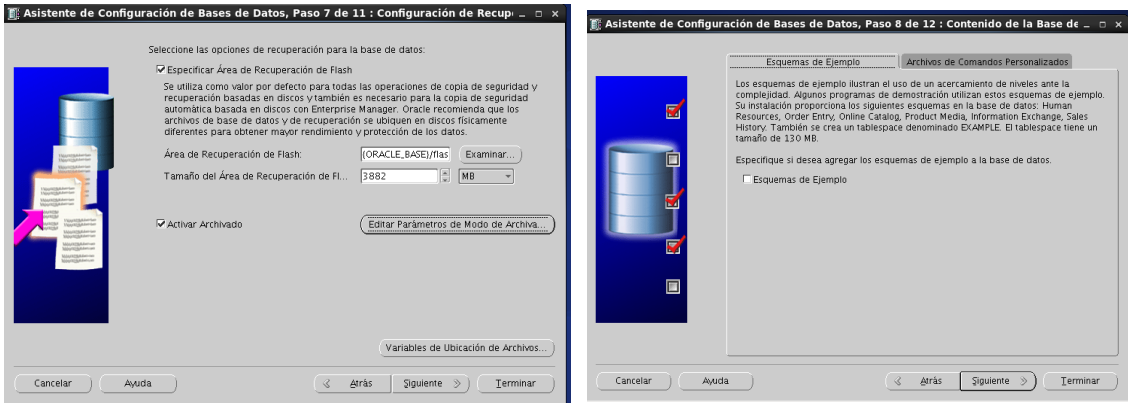


Ilustración 9. Creación BBDD operacional con dbca. 6

Se ha seleccionado archivado (log de transacciones) para un modelo de recuperación completo de la BBDD.

En un principio, no solicitamos la creación de los esquemas de ejemplo porque no tienen sentido en esta base de datos.

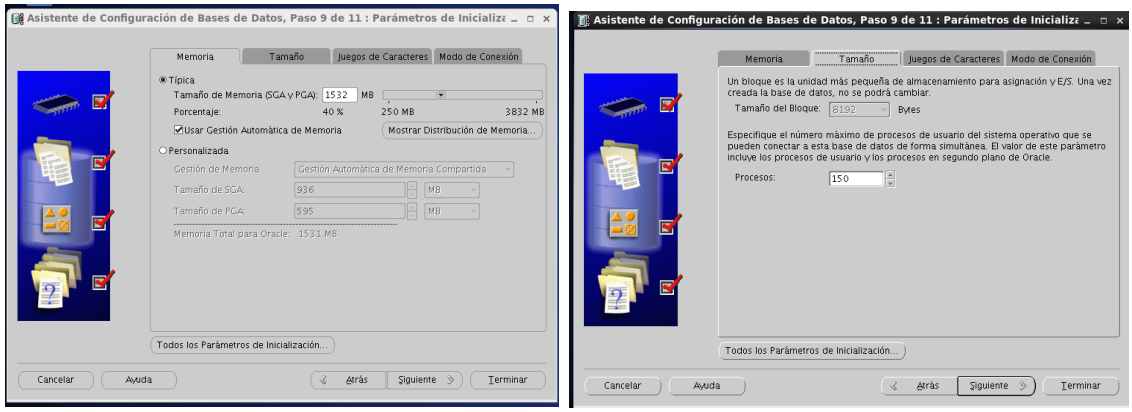


Ilustración 10. Creación BBDD operacional con dbca. 7

No se han modificado los aspectos de gestión de la memoria, dejando todos los valores por defecto.

Se selecciona el modo compartido para el Servidor bajo la premisa que esta BBDD se trata de un recurso al que accederán las diferentes compañías para la carga de datos de los contadores, los clientes, etc…

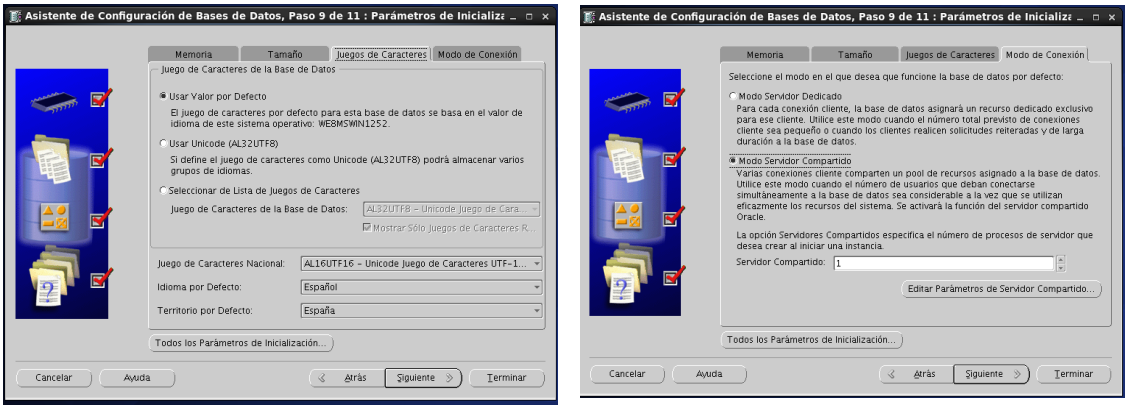


Ilustración 11. Creación BBDD operacional con dbca. 8

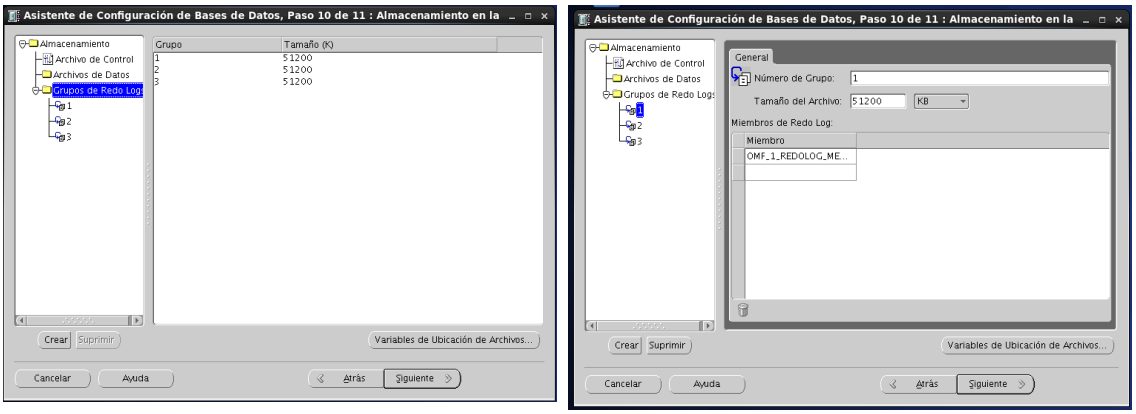


Ilustración 12. Creación BBDD operacional con dbca. 9

Los parámetros quedan de esta forma

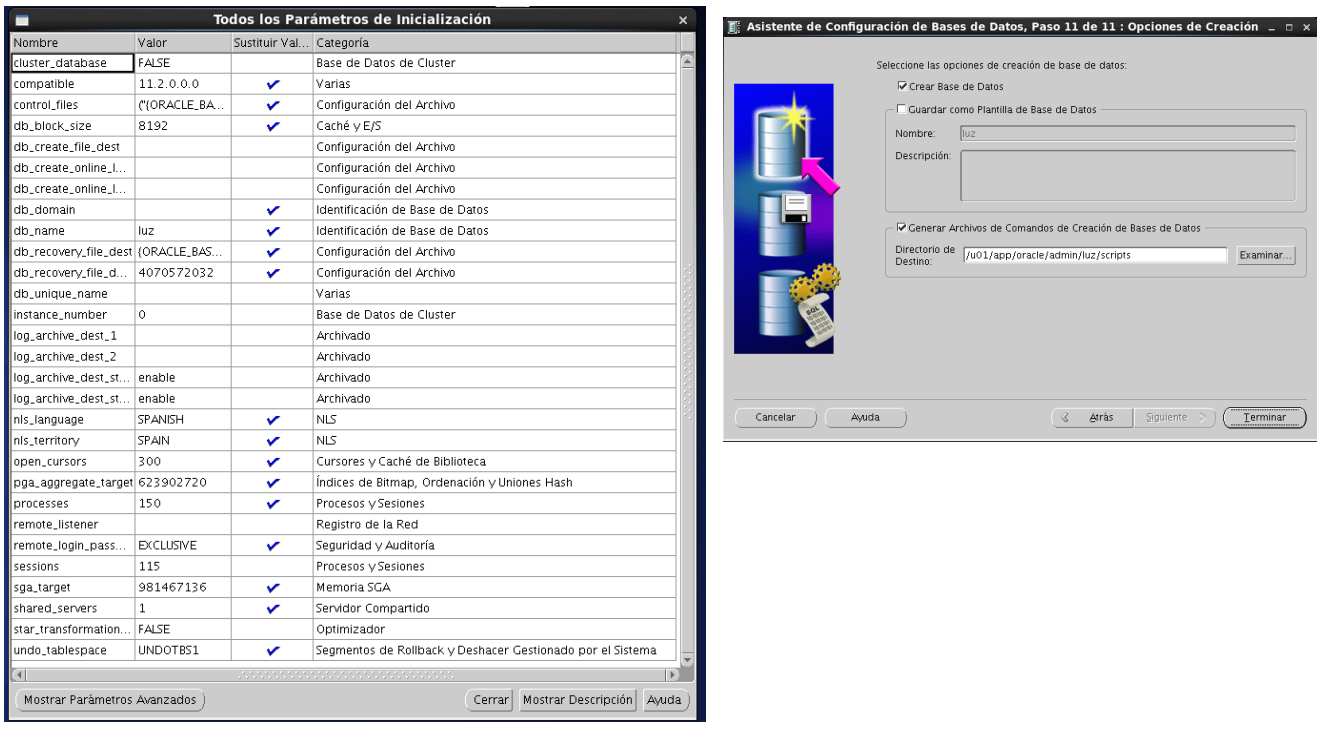


Ilustración 13. Creación BBDD operacional con dbca. 10

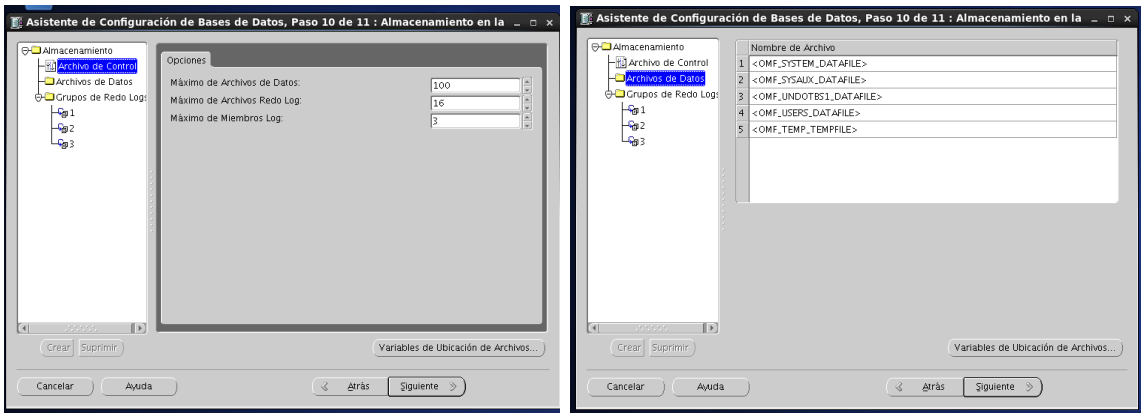


Ilustración 14. Creación BBDD operacional con dbca.11

La generación de los scrips se produce de forma correcta

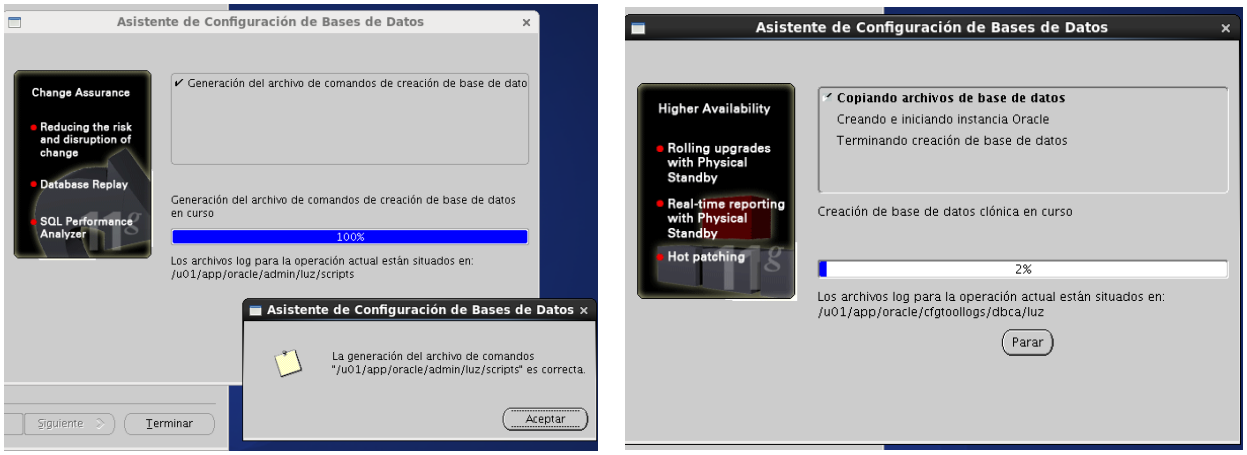


Ilustración 15. Creación BBDD operacional con dbca. 12

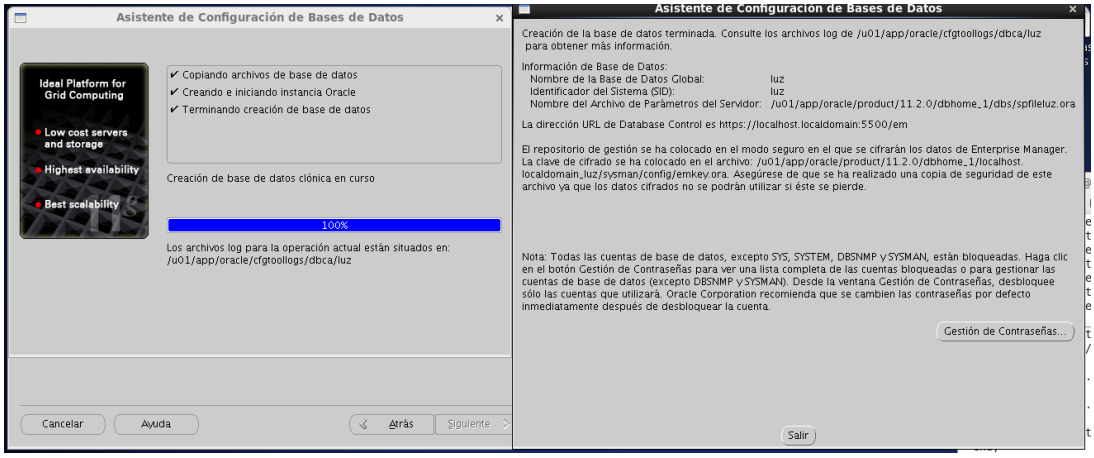


Ilustración 16. Creación BBDD operacional con dbca. 13

La BBDD ha sido creada correctamente. Podemos iniciar la consola de la BBDD para comprobar su funcionamiento con la siguiente orden:

```
./emctl start dbconsole
```

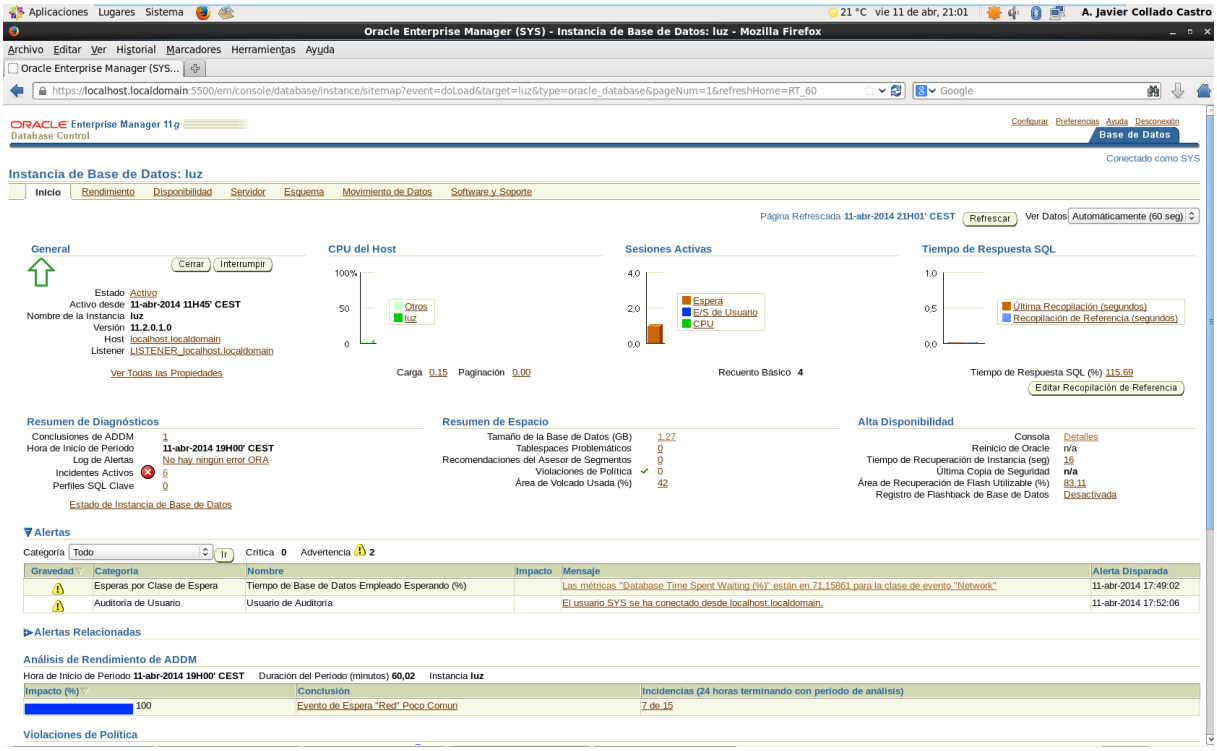


Ilustración 17. Creación BBDD. Consola general

A partir de aquí configuraremos una conexión con la BBDD luz con SQLDeveloper para lanzar todo el proceso

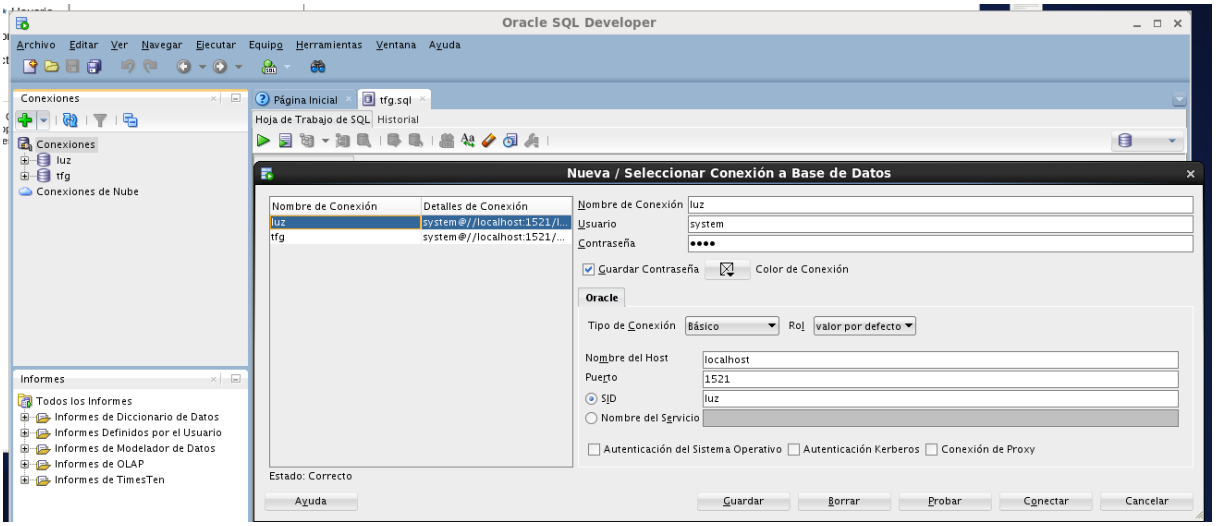


Ilustración 18. Creación BBDD. Configuración conexión BBDD

2.3.2. Índices

Durante la creación de las tablas se han establecido los siguientes índices de acuerdo con el modelo lógico:

Nombre de índice	TABLA	Atributo	Tipo
PK_countryCode_Country	Country	CountryCode	INTEGER
PK_Province	Province	provinceId	INTEGER (sintético)
pk_cityCode_City	City	cityCode	INTEGER
PK_streetTypeCode_StreetType	StreetType	streetTypeCode	INTEGER
PK_addressCode_Address	Address	addressCode	INTEGER (sintético)
PK_bankCode_Bank	Bank	bankCode	VARCHAR2(11 CHAR)
PK_identityCode_IdentityType	IdentityType	identityCode	INTEGER
PK_consumerCode_Consumer	Consumer	consumerCode	INTEGER (sintético)
PK_CompanyCode_Company	Company	Company	VARCHAR2 (15 CHAR)
PK_serialNumber_Meter	serialNumber	Meter	VARCHAR2 (20 CHAR)
PK_Connection	readingDate, meterSerialNumber	Connection	TIMESTAMP VARCHAR2 (20 CHAR)
PK_Price	ChangeData, countryCode, companyCode	Price	DATE INTEGER VARCHAR2 (15 CHAR)
PK_Clients	companyCode, consumerCode	Clients	VARCHAR2 (15 CHAR) INTEGER
PK_Operators	countryCode, companyCode	Operators	INTEGER VARCHAR2 (15 CHAR)
Pk_LuzLog	IdLog	LuzLog	INTEGER (sintético)

Tabla 17. Índices

En esta fase del proyecto intentaremos proponer nuevos índices derivados de las necesidades operativas de la BBDD, y no en base al diseño lógico establecido.

Una correcta y acertada valoración de los índices necesarios debe realizarse en base a las consultas realizadas más frecuentes y el análisis de los tiempos de acceso. El modelo desarrollado hasta este punto no consta de ningún tipo de consulta, tan solo sirve de base para el modelo transaccional, por lo tanto no es posible determinar fehacientemente la necesidad de índices adicionales. No obstante, podemos basándonos en las recomendaciones sobre las creaciones de índices, proponer nuevas estructuras:

- ✓ La creación de índices únicos (claves primarias) para determinar la especificidad de las relaciones tal y como se han expresado en la tabla anterior
- ✓ La creación de índices en las claves externas (FOREIGN KEYS) para evitar el mapeado completo de la tabla externa y los bloqueos a nivel de tabla [7].

De esta forma se propondría la creación de los siguientes índices:

#	TABLA	Atributo	Nombre del índice	Observaciones
1	Province	countryCode	ix_Province_CountryCode	
2	City	provinceCode	ix_City_ProvinceCode	
3	Address	streetCode	--	Se desaconseja la creación de este índice porque la tabla StreetType será, en cualquiera de los casos de un tamaño muy reducido
4	Address	cityCode	ix_Address_CityCode	
5	Consumer	addressCode	ix_Consumer_AddressCode	
6	Consumer	identityCode	--	Se desaconseja la creación de este índice porque la tabla IdentityType será, en cualquiera de los casos de un tamaño muy reducido

Tabla 18. Creación de índices

#	TABLA	Atributo	Nombre del índice	Observaciones
7	Consumer	bankCode	ix_Consumer_BankCode	
8	Company	addressCode	ix_Company_AddressCode	
9	MeterMeter	companyCode	ix_Meter_CompanyCode	
10	Meter	consumerCode	ix_Meter_ConsumerCode	
11	Meter	addressCode	ix_Meter_AddressCode	
12	Connection	meterSerialNumber	ix_Connection_MeterSN	
13	Price	countryCode	ix_Price_CountryCode	
14	Price	companyCode	ix_Price_CompanyCode	
15	Clients	companyCode	ix_Clients_CompanyCode	
16	Clients	consumerCode	ix_Clients_ConsumerCode	
17	Operators	countryCode	ix_Operators_CountryCode	
18	Operators	companyCode	ix_Operators_CompanyCode	

Tabla 19. Creación de índices (continuación)

En el Anexo V se encontrará la codificación de los índices propuestos de la tabla anterior.

Para la creación de los anteriores índices ha sido necesaria la creación de un TABLESPACE, se incluye el código en el anexo V.

2.3.3. Particiones

Una práctica recomendable para optimizar el rendimiento de la BBDD es aplicar técnicas de partición a las tablas. El particionado, entre otras ventajas [7] y [8], permite:

- ✓ Reducir el número de bloques que se solicitan al sistema de almacenamiento.
- ✓ Acceso simultaneo a los datos de las tablas implicadas en una proyección (*join*) que comparten una misma política de particionado (paralelismo)
- ✓ Incremento de la velocidad de borrado de registros (si fuera necesario)
- ✓ Facilitar las tareas de copia de seguridad de los datos (los datos menos frecuentes se pueden ubicar en una partición de solo lectura y así evitar realizar copias de alta frecuencia de esos fragmentos)
- ✓ Reubicación de las particiones con datos menos críticos (antiguos o con menor latencia)
- ✓ Realizar trabajos de mantenimiento que solo afecten a particiones concretas

Las particiones que se podrían plantear en la BBDD responderían a diversas estrategias:

- ✓ Particionado de los datos por País asumiendo una división geográfica de la BBDD
- ✓ Particionado de los datos por Compañía asumiendo una división empresarial de los datos.
- ✓ Particionado de los datos en función de su antigüedad (en aquellas tablas que contengan un atributo temporal)

En cualquiera de los casos la aplicación de una estrategia de particionado supone la modificación del diseño lógico realizado. Una posible propuesta para aplicar un particionado basado en una división geográfica de la BBDD supondría:

Tabla	Criterio	Estrategia	Observaciones
COUNTRY	--	--	--
PROVINCE	COUNTRY	REFERENCE PARTITIONING	El diseño ya incluye el atributo CountryCode
CITY	COUNTRY	REFERENCE PARTITIONING	Es necesario modificar el diseño para incluir el atributo CountryCode en la entidad
STREETTYPE	--	--	Es una tabla general y de un tamaño muy reducido
ADDRESS	COUNTRY	REFERENCE PARTITIONING	Es necesario modificar el diseño para incluir el atributo CountryCode en la entidad
BANK	--	--	La entidad bancos es transversal en cuanto al país
IDENTITYTYPE	--	--	Es una tabla general y de un tamaño muy reducido
CONSUMER	COUNTRY	REFERENCE PARTITIONING	La relación 1 a 1 con la entidad Address permite generar un atributo CountryCode en la entidad
COMPANY	-	--	La relación con Address solo tiene por objeto modelar la dirección fiscal de la entidad porque una compañía puede dar servicio en varios países.
METER	COUNTRY	REFERENCE PARTITIONING	La relación 1 a 1 con la entidad Address permite generar un atributo CountryCode en la entidad
CONNECTION	COUNTRY	REFERENCE PARTITIONING	La relación 1 a * con Meter soporta la creación de un atributo CountryCode para aplicar un particionado por referencia
PRICE	COUNTRY	REFERENCE PARTITIONING	La entidad ya contiene una referencia a CountryCode
CLIENTS	--	--	Esta entidad no soporta particionado por CountryCode
OPERATORS	COUNTRY	REFERENCE PARTITIONING	La entidad ya contiene una referencia a CountryCode
LUZLOG	COUNTRY	REFERENCE PARTITIONING	La entidad soportaría la inclusión de un atributo CountryCode para aplicar un particionado por referencia

Tabla 20. Criterios de particionado

2.3.4. Seguridad

Distinguiremos dos aspectos diferentes sobre la seguridad de la base de datos:

- ✓ Los relacionados con la creación de un sistema de usuarios con permisos de acceso sobre unas determinadas entidades
- ✓ La protección de los datos personales incluidos en las entidades de la BBDD.

Para el primero de estos aspectos distinguiremos 3 tipos de roles de usuarios de acuerdo con el diagrama de casos de uso relacionado en el punto 2.1.2. Cada uno de los usuarios relacionados debe disponer de una serie de permisos sobre las tablas del sistema. Así tendremos la siguiente distribución de permisos de lectura (R de Read), escritura (W de Write) , y lectura/escritura (RW):

Tabla	Rol Usuario Administrador BBDD	Rol Usuario Empresa eléctrica	Rol Usuario BBDD (system)	Observaciones
COUNTRY	RW	R	RW	
PROVINCE	RW	R	RW	
CITY	RW	R	RW	
STREETTYPE	RW	R	RW	
ADDRESS	R	RW	RW	
BANK	RW	R	RW	
IDENTITYTYPE	RW	R	RW	
CONSUMER	R	RW	RW	
COMPANY	RW	R	RW	
METER	R	RW	RW	
CONNECTION	R	Acceso especial (1)	RW	
PRICE	R	RW	RW	
CLIENTS	R	RW	RW	
OPERATORS	R	RW	RW	
LUZLOG	R	W	RW	

Tabla 21. Permisos por entidad

La justificación a este diseño responde a:

- ✓ El usuario administrador dispone de acceso total a las entidades generales de la BBDD. Es el encargado de proveer los "diccionarios" necesarios para el resto de las entidades.
- ✓ El usuario empresa eléctrica (o usuarios) dispondría de acceso total a todas las entidades relacionadas con el objeto de su actividad, si bien este acceso total, podría estar restringido a la parte de las entidades afectadas por su compañía. Es decir acceso a los clientes, contadores y direcciones relacionados con su compañía.
- ✓ El usuario Base de datos tiene acceso total a todos los objetos de la aplicación y administra especialmente el contenido de la tabla Log.

El acceso a la tabla Connection es especialmente sensible porque contiene los datos de consumo, por lo que además de disponer de un acceso parcial (solo al consumo de los contadores de su compañía) tambien tiene las restricciones impuestas por el trigger de inserción para preservar la privacidad de los consumidores (lecturas a intervalos de 1 hora del consumo de los contadores).

En lo que respecta a la protección de datos personales las iniciativas para dotar de la seguridad necesaria a la base de datos pasarían por:

- ✓ Administrar los accesos de la base de datos (la anterior clasificación de tipos de usuario es el primer paso)
- ✓ Definir las tablas con datos personales (básicamente la tabla consumer)
- ✓ Establecer un sistema de control y registro de acceso a la información (la tabla de Log permite este tipo de control)
- ✓ Estudiar la posible encriptación de los datos más sensibles de la tabla Consumer (identificación del consumidor, cuenta bancaria, etc…)

El anexo V contiene los scripts de creación de roles y permisos sobre las difentes entidades.

2.3.5. Validación y justificación del diseño físico

Como resumen podemos enunciar las siguientes conclusiones sobre el modelo físico:

- ✓ Apartado de preparación del Servidor, instalación del sistema y creación de la BBDD: Se ha implementado completamente
- ✓ Índices: Se ha implementado completamente suministrando los scripts de creación de los índices
- ✓ Particiones: Se apunta una posible implementación en base a una clasificación geográfica
- ✓ Seguridad: Se ha implementado parcialmente incluyendo los scripts de creación de los roles y la asignación de los permisos sobre las tablas (requiere la creación de los usuarios).

3. Análisis y creación del almacén de datos (Data Warehouse) del sistema analítico

3.1. Breve introducción de los conceptos DW y metodología de diseño de Data Warehouse a partir de modelos OLTP

Un almacén de datos o *Datawarehouse* (DW) no es más que una colección de datos orientada a un determinado fin o ámbito concreto, integrado, no volátil, variable en el tiempo, y cuyo fin es ayudar a la toma de decisiones.

Un entorno de DW puede ser construido mediante diversas aproximaciones entre las que podemos distinguir los esquemas estrella y los cubos OLAP (*On Line Analytical processing*). Los esquemas estrella son generalmente utilizados en entornos de bases de datos relacionales y se gestionan mediante consultas SQL convencionales. Los cubos OLAP responden a una funcionalidad superior y cuentan con estrategias específicas tanto para el almacenamiento de la información (capa física) como para la formulación de consultas a los datos (excediendo las capacidades de SQL como lenguaje) [3].

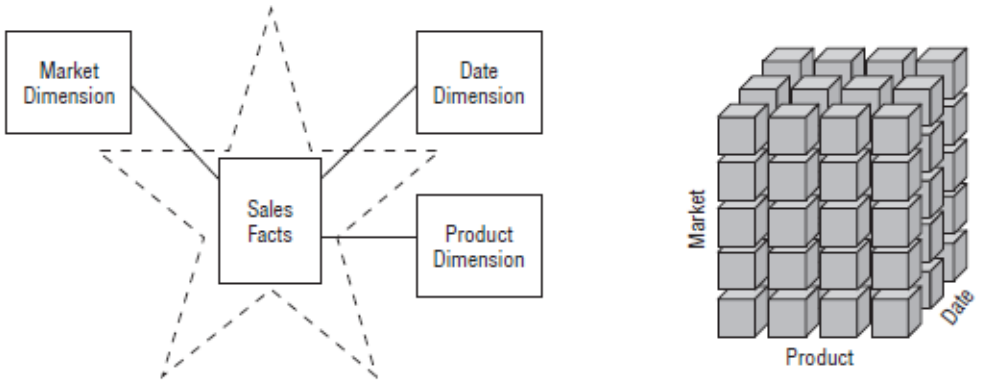


Ilustración 19. Esquemas estrella vs. cubos OLAP . (Kimball R., 2013)

Una de las principales características de un DW es la multidimensionalidad mediante la que se consigue representar la información en forma de hechos y dimensiones en lugar de tablas y atributos.

La aproximación que se realiza en este trabajo es mediante esquemas estrella, y la metodología empleada responde a la metodología establecida por Kimball [3]:

- ✓ Paso 1: Seleccionar el proceso del negocio
- ✓ Paso 2: Declarar el gránulo
- ✓ Paso 3: Identificar las dimensiones
- ✓ Paso 4: Identificar los hechos

Posteriormente será necesario implementar el resto de los elementos necesarios para completar el sistema: El proceso de carga y transformación de los datos en el DW (procesos ETL, *Extract, Transformation, and Load*), y la generación de las consultas requeridas.

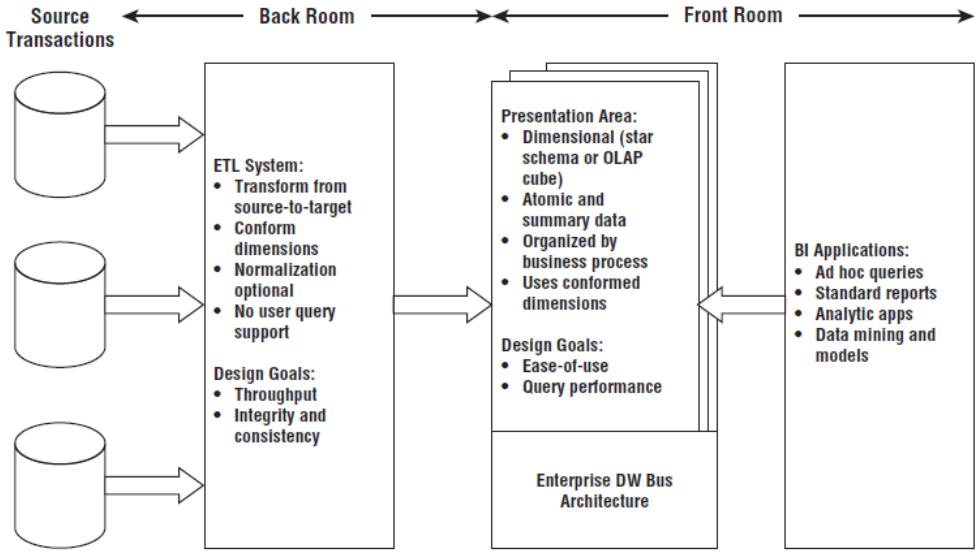


Ilustración 20. Elementos principales de la arquitectura DW de Kimball [3]

3.2. Diseño conceptual del sistema

3.2.1. Proceso de negocio

El proceso de negocio seleccionado es la medición de los consumos energéticos derivados de la lectura de los contadores inteligentes.

3.2.2. Análisis de Requisitos

Los principales requisitos que debe cumplir nuestro DW son:

- ✓ El almacén de datos tiene que ofrecer los resultados en tiempo constante 1. Es decir, mediante una orden SELECT sobre un registro de una tabla (que no sea una vista calculada o materializada, ni se utilicen funciones de agregado con la cláusula GROUP BY).
- ✓ Las principales consultas que debe proporcionar el sistema son:
  - Consumo eléctrico mensual de cada contador
  - Contadores con consumo mensual superior al 80% por ciudad y mes
  - Consumo eléctrico por ciudad/País/mes
  - Lecturas correctas por Empresa/mes
  - Listado contadores con x años de antigüedad
  - Valor medio de la energía por Ciudad/año
  - Precio total de la energía mensual consumida por contador

- Top 10 de los contadores con un consumo superior en cada ciudad
- Consumo medio de todos los consumidores por ciudad/mes

3.2.3. Definición del Gránulo

El granulo es la especificación de la información que muestra cada fila de la tabla de hechos, y debe responder, según Kimball [3] a la pregunta ¿Cómo describirías una fila de la tabla de hechos? De esta forma, el gránulo se definirá como el consumo energético en una fecha y hora determinada de un contador que pertenece a un cliente, una compañía, una ciudad, un país concreto, y que se genera a un precio determinado.

El gránulo así definido es lo suficientemente atómico, y único, como para representar una entidad con una gran cantidad de dimensiones lo cual favorecerá la capacidad de análisis del modelo.

Cuando se definen los gránulos y los hechos resulta recomendable hacer una valoración de la cantidad de hechos que acumulará la tabla de hechos. Una tabla así definida puede almacenar, por la restricciones en la medición de los contadores, hasta 24 hechos por día (una medición cada hora). Según la CNMC (Comisión Nacional de Mercados y Competencia de España) el número estimado de contadores de energía eléctrica que deberán ser convertidos en inteligentes asciende a 27,7 millones de unidades (fuente http://www.cnmc.es/es-es/energ%C3%ADa/sobreenerg%C3%ADa.aspx).

De esta forma, el número posible de registros de la tabla de hechos de un día, en España, podría ser de 27,7millones de contadores*24 horas (máximo número de consultas válidas a un contador en un día) lo cual nos da 664,8 millones de registros diarios. El número de registros anual asciende pues a 664,8 millones registros diarios*365 días lo cual nos da la nada despreciable cifra de 242.652 millones de registros anuales para un país como España. Quizás fuera necesario eliminar la vertiente temporal de la tabla de hechos (eliminando la posibilidad de realizar análisis sobre las horas de mayor consumo) dejando la tabla de hechos en 27,7*365=10.110,5 millones de registros. Otra posibilidad es decidir realizar los análisis horarios en modelos de carácter mensual.

3.2.4. Identificación de las dimensiones. Atributos, descriptores y jerarquías de agregación

Siguiendo la metodología descrita, las dimensiones deben responder a la pregunta ¿Cómo se describen los datos resultado de los eventos de medición del proceso de negocio? Habitualmente las dimensiones también responden a las preguntas de quien, qué, dónde, cuándo, por qué y cómo. Las dimensiones están siempre íntimamente ligadas con los datos descriptivos.

Las dimensiones básicas y sus atributos, que además permitirán responder adecuadamente a las consultas requeridas, serán:

- ✓ La dimensión fecha (Date) que contará con las agregaciones día de la semana, día del mes, mes y año
- ✓ La dimensión Tiempo (Time) que contará con agregaciones para cada una de las horas en formato 24 horas.
- ✓ La dimensión Contador (Meter) que contará con agregaciones por el modelo de contador y la fecha de instalación.
- ✓ La dimensión dirección (Address) que contará con agregaciones por ciudad y país.
- ✓ La dimensión Compañía que solo contará con una agregación por nombre de la compañía
- ✓ La dimensión Consumidor que solo contará con una agregación por nombre de consumidor.

Las jerarquías de agregación son los atributos que permiten agrupar registros mientas que los descriptores son atributos que permiten seleccionar registros. De esta forma tendremos que:

- ✓ La dimensión fecha (Date) tiene que permitir tanto seleccionar como agrupar por fecha, día de la semana, día del mes, mes y año
- ✓ La dimensión Tiempo (Time) funciona de igual forma. Tiene que permitir tanto seleccionar como agrupar por cada una de las horas en formato 24 horas. Adicionalmente se podrían establecer jerarquías de tipo horario nocturno/ horario diurno para agrupar una franja de horas.
- ✓ La dimensión Contador (Meter) tiene que permitir seleccionar y agrupar por contador y modelo.
- ✓ La dimensión dirección (Address) debe permitir seleccionar y agrupar por ciudad y país.
- ✓ La dimensión Compañía debe permitir seleccionar y agregar por nombre de la compañía
- ✓ La dimensión Consumidor, de igual forma, debe permitir seleccionar y agregar por nombre de consumidor.

3.2.5. Identificación de los hechos (*facts*)

Los hechos representan el tema objeto del análisis y responden a la pregunta ¿Qué es lo que se mide en el proceso de negocio seleccionado? En este caso la métrica que se pretende modelar es el consumo energético.

De esta forma expresaremos el consumo energético en un momento dado, como consumo instantáneo basado en la diferencia entre dos valores:

- ✓ El dato de la lectura del contador inmediatamente anterior (*previousReading*)
- ✓ El dato de la lectura instantánea (*actualReading*) del contador

La diferencia entre estos dos valores nos proporciona el consumo en el momento concreto que el gránulo establece. El gránulo así definido conforma un hecho de tipo derivado en tanto en cuanto se obtiene de la operación de 2 atributos, y de carácter aditivo (la suma de los consumos de todos los gránulos es consistente).

Resulta necesario definir, también, como se va a proceder cuando se evalúe una lectura errónea de un contador cuyo valor es cero. La regla que se define asignará un valor de 0 al consumo cuando cualquiera de los valores de lectura del contador (*previousReading* o *actualReading*) sea cero. De esta forma nos encontraremos en la tabla de hechos, filas que tendrán un consumo 0 que indican una lectura errónea. El valor de la métrica Consumo (*consumption*), de tipo aditivo, no se verá afectada por estos valores (no son valores nulos, son valores 0).

Otro hecho que debería registrarse es el coste de esta energía, por lo que sería necesario incluir en la tabla de hechos el atributo del precio (Price) de la energía en ese momento concreto, para ese contador concreto. El atributo derivado Coste (*Cost*) sería el resultado de aplicar la operación de consumo por el precio de la energía. De esta forma cada fila de la tabla de hechos consta de los atributos consumo, precio, y coste.

Una aproximación a la tabla de hechos de forma gráfica sería:

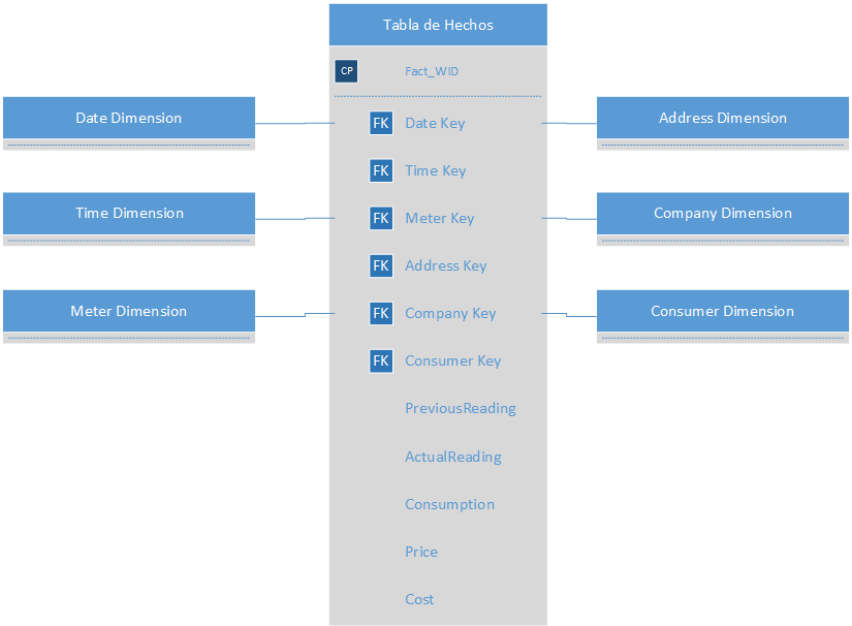


Ilustración 21. Tabla de hechos

3.2.6. Esquema conceptual del sistema

Esquema conceptual

El esquema de la estrella propuesta con los atributos de las dimensiones podría ser:

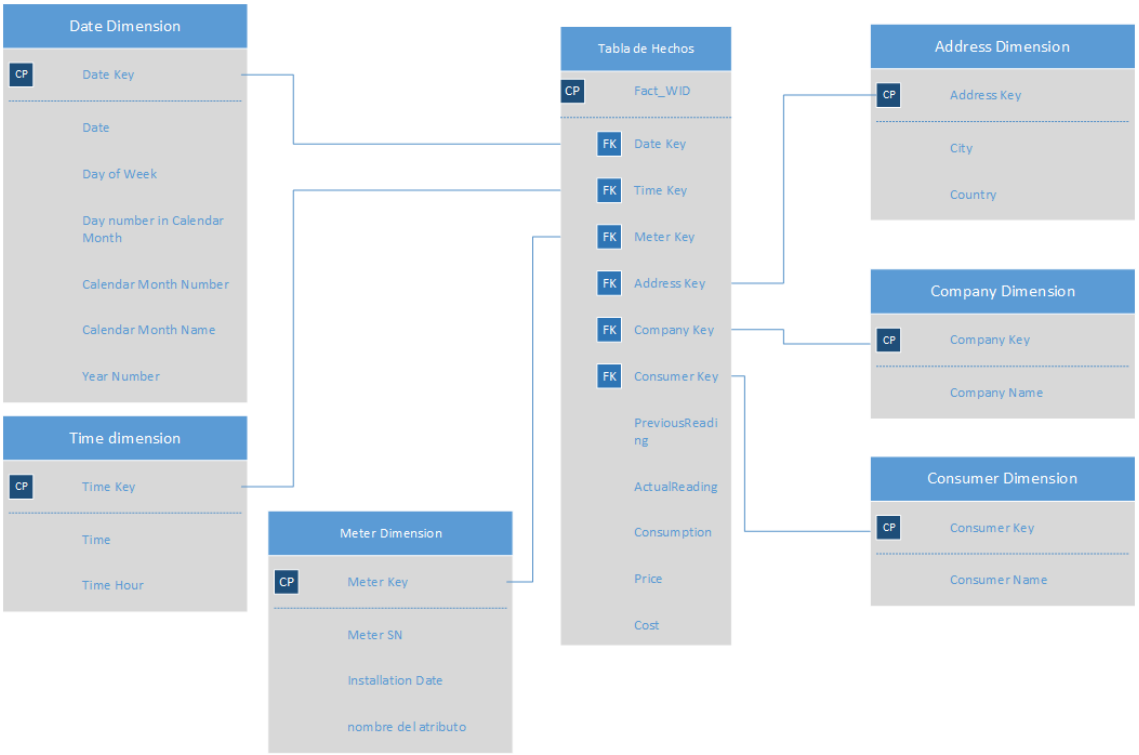


Ilustración 22. Esquema del DW Luz

Diagrama UML de casos de uso de la BBDD estratégica

El diagrama de uso del almacén de datos contiene las siguientes tareas:

- ✓ Proceso de carga del AD:
  - Extracción
  - Transformación
  - Carga
- ✓ Consultas:
  - Consumo eléctrico mensual de cada contador
  - Contadores con consumo mensual superior al 80% por ciudad y mes
  - Consumo eléctrico por ciudad/País/mes
  - Lecturas correctas por Empresa/mes
  - Listado contadores con x años de antigüedad
  - Valor medio de la energía por Ciudad/año
  - Precio total de la energía mensual consumida por contador
  - Top 10 de los contadores con un consumo superior en cada ciudad
  - Consumo medio de todos los consumidores por ciudad/mes

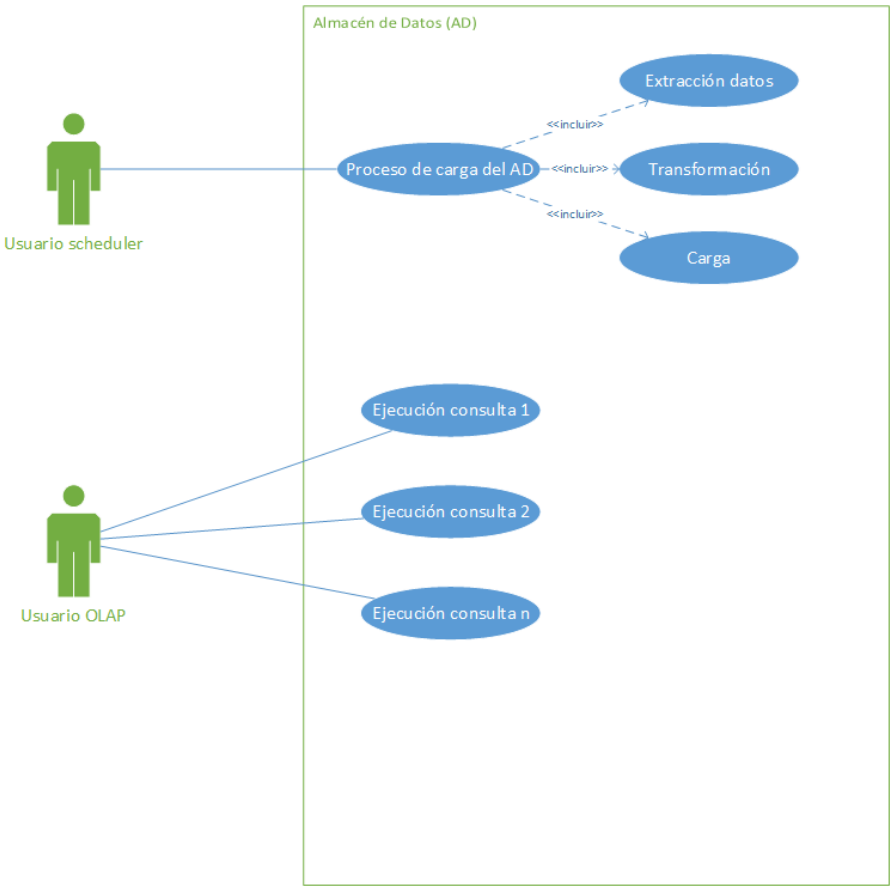


Ilustración 23. Casos de uso del DW

3.2.7. Restricciones de integridad

Las restricciones de integridad que requiere este modelo se resumen en:

- ✓ La tabla de hechos muestra los valores de consumo de un contador en un momento dado.
- ✓ Las lecturas de los contadores no se producen en intervalos homogéneos para todos los contadores. Es decir, pueden existir registros que comprendan el consumo en 1 hora (diferencia entre la lectura anterior y la actual) y registros que comprendan el consumo de 1 día (de nuevo la diferencia temporal entre la lectura anterior y la lectura actual).
- ✓ Los casos en los que una de las lecturas sea errónea (valor 0) se grabarán con consumo 0, identificando de esta forma los casos con lectura errónea.
- ✓ Los casos no pueden tener consumo nulo. O vale 0 o tiene valor.

3.3. Diseño lógico del sistema

El objetivo de esta fase es obtener un esquema lógico de la base de datos del DW, expresado en modelos lógicos relacionales, a partir del esquema conceptual desarrollado en la anterior fase.

3.3.1. Descripción del diseño lógico estándar

Para la definición del diseño lógico utilizaremos las siguientes convenciones [4]:

- ✓ Los atributos que son clave principal se marcan con un subrayado.
- ✓ Los atributos que tienen valores no nulos se marcan en negrita
- ✓ Las claves candidatas se marcarán con un subrayado con líneas discontinuas.

De esta manera se obtiene la siguiente conversión del esquema conceptual de la estrella:

- ✓ Para las tablas de dimensiones:

  W_Date_D (date_WID, **dayOfWeek, dayNumberMonth, monthNumber, monthName, yearNumber**)

  En la entidad W_Date_D hemos generado una clave sintética, *Surrogate Key* [5] date_WID. Si bien teniendo en cuenta las observaciones sobre las clave de la dimensión fecha [3] página 79, finalmente date_WID será un campo de fecha.

  W_Time_D(time_WID, **timeHourNumber**)

  W_Meter_D(meter_WID, **meterInstallationDate, modelName, meterCode**)

  meter_WID es una clave sintética

  W_Address_D(address_WID, **addressCode**, **cityMeterAddres, countryMeterAddress**)

  address_WID es una clave sintética

  W_Company_D(company_WID, **companyCode**, **companyName**)

  company_WID es una clave sintética

  W_Consumer_D(consumer_WID, **consumerCode**, **consumerName**)

  consumer_Wid es una clave sintética

- ✓ Para la tabla de hechos:

  W_Consumption_F(consumption_WID, **date_WID, time_WID, meter_WID, address_WID, company_WID, consumer_WID**, **previousReading, actualReading, consumption, price, cost**)

  {date_WID} es clave externa de W_Date_D

  {time_WID} es clave externa de W_Time_D

  {meter_WID} es clave externa de W_Meter_D

  {address_WID} es clave externa de W_Date_D

  {company_WID} es clave externa de W_Company_D

  {consumer_WID} es clave externa de W_Consumer_D

3.3.2. Descripción del diseño lógico específico en Oracle

El objetivo de esta fase es realizar la traducción del modelo lógico estándar anterior en la nomenclatura específica de la solución adoptada para realizar la implementación de la BBDD, en nuestro caso en el sistema ORACLE para lo cual seguimos contado con la inestimable ayuda de uno de los mejores manuales de Oracle PL/SQL [6].

Las tablas de dimensiones se generarán con claves primarias sintéticas independientes de las claves establecidas en las tablas originales. Este diseño permite, primero una fácil actualización de las tablas y, en segundo lugar, la independencia de las claves de las tablas origen (que pueden cambiar).

La sintaxis para la creación de las tablas de un *datawarehouse* no difiere de la sintaxis utilizada para la creación de las tablas de una base de datos operacional. En este caso cobra vital importancia el ajuste en el tamaño de los campos, teniendo en cuenta que, la tabla de hechos va a almacenar millones de registros. También es importante definir las restricciones estrictamente necesarias (ya se validaran los datos durante el proceso ETL) para evitar demoras de comprobación de restricciones en las operaciones de inserción de datos en el DW (se van a insertar millones de registros). Adicionalmente es necesario determinar la ubicación física de cada tabla a efectos de gestión del almacenamiento. Estas y más consideraciones se definirán en el apartado de diseño físico del Datawarehouse.

El código completo de la creación de las tablas se acompaña en el anexo VII.

Nombre de la tabla	Tipo entidad	Descripción	Observaciones
W_Date_D	Dimensiones	Tabla de las dimensiones de fecha	
W_Time_D	Dimensiones	Tabla de la dimensión tiempo	
W_Meter_D	Dimensiones	Tabla de la dimensión de los contadores	
W_Address_D	Dimensiones	Tabla de la dimensión de las direcciones	
W_Company_D	Dimensiones	Tabla de la dimensión de las empresas suministradoras	
W_Consumer_D	Dimensiones	Tabla de la dimensión de los consumidores	
W_Consumption_F	Hechos	Tabla de hechos	

Tabla 22. Tablas de la BBDD analítica

3.3.3. Pruebas de las tablas de la BBDD

Para la realización de las pruebas se han introducido valores de prueba en las diferentes tablas. El código completo de los valores introducidos se acompaña como Anexo VIII.

Las validaciones realizadas en las tablas mediante la introducción de datos comprenden:

(6) La entidad es capaz de representar los datos
(7) Las reglas de integridad no permiten la introducción de entidades con valores duplicados (UNIQUE, o PRIMARY KEY)
(8) No se pueden introducir valores nulos (restricciones NOT NULL)
(9) Claves externas (FOREIGN KEY)
(10) Otras reglas de negocio (a especificar)

Nombre de la tabla	Valores de las entidades (1)	Repetición de campos de clave (2)	Valores nulos (3)	Claves externas (4)	Otras reglas de negocio (5)	Observaciones
W_Date_D	OK	OK	OK	--	--	Tabla dimensiones
W_Time_D	OK	OK	OK	--	--	Tabla dimensiones
W_Meter_D	OK	OK	OK	--	--	Tabla dimensiones
W_Address_D	OK	OK	OK	--	--	Tabla dimensiones
W_Company_D	OK	OK	OK	--	--	Tabla dimensiones
W_Consumer_D	OK	OK	OK	--	--	Tabla dimensiones
W_Consumption_F	OK	OK	OK	OK	OK	Tabla de hechos

Tabla 23. Validaciones realizadas en las tablas

3.3.4. Creación disparadores (triggers)

La lista de disparadores de la BBDD del DW es:

#	Nombre	Tabla afectada	Objeto	Observaciones
1	Insert_meter_WID	W_METER_D	Introducir la clave sintética de la entidad	
2	Insert_address_WID	W_ADDRESS_D	Introducir la clave sintética de la entidad	
3	Insert_company_WID	W_COMPANY_D	Introducir la clave sintética de la entidad	
4	Insert_consumer_WID	W_CONSUMER_D	Introducir la clave sintética de la entidad	
5	Insert_consumption_Wid	W_CONSUMPTION_F	Introducir la clave sintética de la entidad	
6	consumption_Calcs	W_CONSUMPTION_F	Introducir los valores de los atributos derivados consumption y energyCost	Solamente si ambos valores de lecturas son diferentes de 0 Coste energía si los valores de las lecturas son diferentes de 0 y hay precio

Tabla 24. Disparadores del DW

El listado completo de los scripts de los disparadores se encuentra como anexo VII a este documento, junto con los scripts de creación de las tablas correspondientes.

3.3.5. Pruebas y validaciones de los disparadores

Se han realizado las siguientes pruebas de control de los disparadores:

- ✓ Pruebas durante la inserción de nuevos valores en la tabla
- ✓ Pruebas durante la actualización de los valores de la tabla

#	Nombre	Tabla afectada	Inserción	Actualización	Observaciones
1	Insert_meter_WID	W_METER_D	OK	--	Sólo insert
2	Insert_address_WID	W_ADDRESS_D	OK	--	Sólo insert
3	Insert_company_WID	W_COMPANY_D	OK	--	Sólo insert
4	Insert_consumer_WID	W_CONSUMER_D	OK	--	Sólo insert
5	Insert_consumption_Wid	W_CONSUMPTION_F	OK	--	Sólo insert
6	consumption_Calcs	W_CONSUMPTION_F	OK	OK	Insert, Update

Tabla 25. Pruebas disparadores

Salvo el *trigger* número 6, el resto responden a la inserción de la clave sintética de la entidad (valor de una secuencia) como clave principal de la entidad por lo que la prueba al actualizar este valor no tiene sentido (los disparadores actúan solo al insertar un valor nuevo).

El disparador *consumption_Calcs* se utiliza para actualizar los valores de los atributos derivados de la tabla W_CONSUMPTION_F. La condición para actualizar los datos del atributo *consumption* es que los campos *previousReading* y *actualReading* sean diferentes de 0. En caso contrario se trata de una lectura errónea y no existe consumo (consumo=0). Al mismo tiempo, también se produce la actualización del atributo energyCost siempre y cuando exista consumo y el precio sea diferente de 0.

3.3.6. Diseño e implementación del ETL

El proceso ETL está formado por los procesos de Extracción, Transformación y Carga (Load) en la BBDD definitiva.

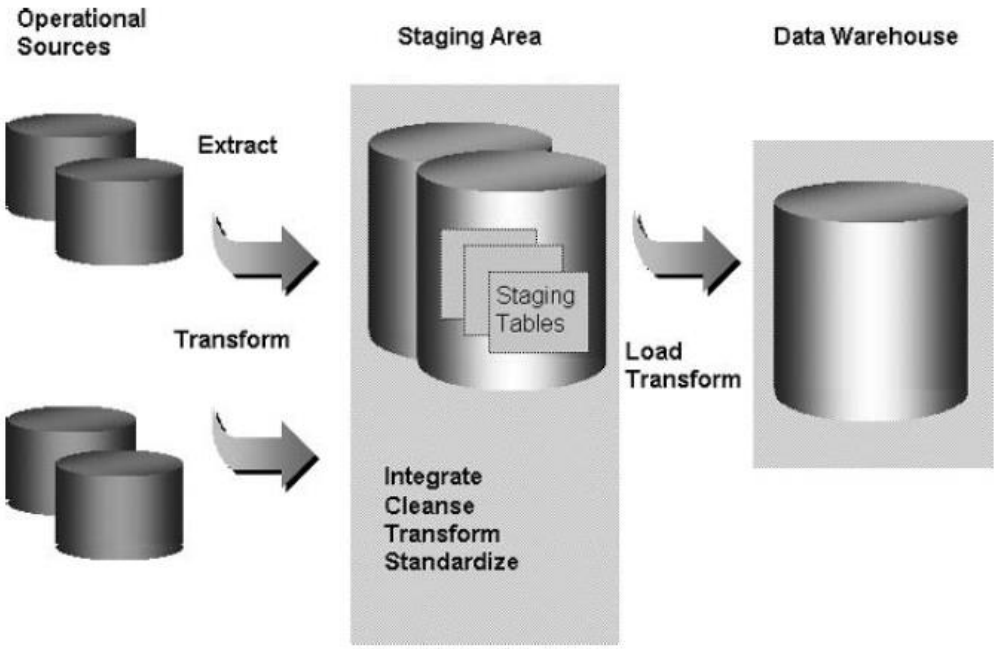


Ilustración 24. El proceso ETL. Fuente [9] pág. 155

El proceso ETL contará con los siguientes pasos:

1. Generación de los datos de prueba y mapeado de los datos origen
2. Selección de técnica para el proceso ETL
3. Determinación de estrategia para datos históricos
4. Procesos de Extracción
5. Procesos de transformación
6. Procesos de carga
7. Pruebas y comprobaciones

3.3.7. Generación de los datos de prueba y mapeado de los datos origen

Se ha generado la siguiente plantilla de datos para las pruebas del ETL

Tabla	Número de datos	**Observaciones**
COUNTRY	30 registros	
PROVINCE	87 registros	
CITY	1.348 registros	
STREETTYPE	6 registros	Es un diccionario de los tipos de calle. Interviene en ADDRESS
ADDRESS	1.100 registros	100 registros de compañía, 500 registros de consumer y 500 registros de meter
BANK	3 registros	No interviene en los procesos ETL ni en las consultas del DW
IDENTITYTYPE	3 registros	Es un diccionario de los tipos de identificación. Interviene en CONSUMER
CONSUMER	503 registros	De ciudades aleatorias entre las 1000 existentes
COMPANY	53 registros	
METER	481 registros	
CONNECTION	111.971 registros	Lecturas de los contadores para todos los meses del año de forma aleatoria
PRICE	2.371 registros	10 cambios de precio al año por compañía y país=10*100*30
CLIENTS	--	No hemos rellenado esta tabla
OPERATORS	278 registros	Las 100 compañías operan en los 30 paises
LUZLOG	336.771	En algunos casos se han utilizado los procedimientos de Alta diseñados para el proceso ETL (dejando por lo tanto registro de LOG)

Tabla 26. Referencia de los datos de prueba generados

3.3.8. Selección de técnica para el proceso ETL

La implementación del ETL en una base de datos ORACLE tiene varias aproximaciones [9] y [10] de las cuales proporcionaremos una sucinta explicación.

Existen dos tipos lógicos de extracción:

- ✓ Extracción total (*full extraction*): Se extraen todos los datos del sistema origen de una vez. No es necesario controlar los cambios de datos en el origen. En nuestro caso esta opción no es la idónea porque se van generando datos con las lecturas de los contadores, nuevos clientes, nuevos contadores… y es necesario prever la incorporación de datos adicionales al DW. Además el volumen de datos tratados hace que no sea operacional proceder a una carga completa del DW cada vez que se desea consultarlo.
- ✓ Extracción incremental (*incremental extraction*): Se extraen los datos añadidos o modificados a partir de un punto concreto en el pasado. Es necesario identificar la información que ha cambiado o se añadido. Esto se puede realizar a través de ciertas columnas del sistema origen, *timestamps*, o mecanismos como las "*change tables*" de Oracle.

En nuestro caso el mecanismo de extracción que se utilizará será el de extracción incremental.

Existen dos formas físicas para la extracción de los datos:

- ✓ Extracción online: Los datos se extraen del sistema origen directamente. Lo cual implica el acceso, de alguna manera, o a las tablas del sistema origen o a un sistema intermedio

donde se guarden los datos de forma temporal (llámese *snapshots logs* o tablas de cambios –*change tables*).

- ✓ Extracción offline: Los datos no se extraen directamente del sistema origen, sino que se genera algún tipo de repositorio donde extraer los datos fuera del sistema origen. Algunos de estos tipos de repositorio podrían ser Ficheros planos, Ficheros de volcado (dump files), redo y archivos de log, y tablespaces transportables.

En nuestro caso, y de acuerdo con el gran volumen de datos que genera el sistema operacional utilizaremos una extracción física online. De esta forma no es necesario generar ningún mecanismo para la generación de datos ahorrando tiempo de procesamiento y espacio físico (las connotaciones de espacio son especialmente importantes por la gran cantidad de datos del sistema operacional).

Las técnicas y herramientas para la extracción podrían clasificarse en:

- ✓ *Embedded ETL in the Oracle Database*: Oracle incluye extensiones al lenguaje SQL (multi–table insert y *MERGE* SQL) que permiten construir *scripts* ETL directamente.
- ✓ *SQL*Loader*: Se trata de una utilidad de Oracle para la lectura de datos en ficheros planos (principalmente) aunque puede atacar otras bases de datos Oracle, bases de datos *legacy* (no relacionales incluso) y diversos formatos de hojas de cálculo.
- ✓ *Change Data Capture*: Además de una estrategia para implementar el proceso ETL, *Change Data Capture* da respuesta a la carga incremental de datos en el DW. Mediante la descripción de una serie de tablas de cambios (*change tables*) Oracle actualiza los cambios realizados en la base de datos (a través del acceso a los registros de transacciones), mediante *triggers* que forman parte de las transacciones comunes, hacia las tablas de cambios.
- ✓ *Data Pump*: Se trata de un mecanismo optimizado (permite paralelismo) de importación de datos de alta velocidad.
- ✓ *Transportable Tablespaces*: Es una característica de Oracle que permite copiar los datos contenidos en un *Tablespace* de una base origen a una base de datos de destino.
- ✓ *Oracle Warehouse Builder*: Se trata de una herramienta que provee de un *framework* gráfico para la definición y ejecución de procesos ETL.

Se ha implementado la extracción de datos directamente en SQL mediante la opción *Embedded ETL*, a través de "*database links*" aprovechando que el sistema origen es también una base de datos de Oracle. Se podría utilizar un esquema basado en *Change Tables* pero se considera que la actualización de las *Change tables* podría incidir en el rendimiento del sistema operacional, además de requerir, de nuevo, de un almacenamiento adicional. La estructura incremental de los datos facilita el proceso de extracción directa.

3.3.9. Determinación de estrategia para datos históricos

Es necesario determinar cómo se van a tratar los datos históricos, es decir, ¿cómo recargamos la BBDD? ¿Qué ocurre si tenemos dimensiones que han cambiado? ¿Qué ocurre con la tabla de hechos?

En primer lugar, se hace necesario distinguir entre la estrategia de actualización de las tablas de dimensiones y la estrategia de actualización de las tablas de hechos.

Las tablas de dimensiones definidas en el DW se consideran a todos los efectos dimensiones de tipo 1 (*Slowly Changing Dimension, SCD, Type* 1). Recogen los valores de fecha, tiempo, contadores (Metter), direcciones (Address), compañías (Company), y consumidores (Consumer). La estrategia de actualizaciones de estas dimensiones puede realizarse mediante la lógica de la instrucción SQL MERGE de Oracle. La lógica de la instrucción MERGE de Oracle habilita un UPDATE si el dato ya existe en la tabla o un INSERT si el dato no existe en la tabla. Realiza ambas operaciones en un solo paso.

No parece necesario realizar un seguimiento de los cambios en las dimensiones dada la orientación estadística de los datos del DW. Además, cualquier seguimiento que se desee realizar sobre las dimensiones podría realizarse a través de la creación de nuevas jerarquías en las tablas de dimensiones. Por lo tanto las tablas de dimensiones no recogen información histórica sobre los cambios en las dimensiones.

La tabla de hechos se basa, fundamentalmente, en la tabla de lecturas de los contadores (CONNECTION). La tabla CONNECTION es una tabla incremental porque las lecturas de los contadores se van añadiendo conforme se van produciendo. La lógica generada a través de los procedimientos almacenados del sistema operacional no permite la modificación de lecturas realizadas. Tan solo permite añadir nuevas lecturas. Por lo tanto el tratamiento de los datos históricos de esta tabla consiste únicamente en tener en cuenta la fecha de volcado de datos. La lectura de un contador siempre avanza hacia adelante.

3.3.10. Procesos de Extracción

El acceso a la base de datos operacional se realiza mediante la creación de un database link
El código del enlace es:

```
/*********************************************/
/* DATABASE LINK TO THE OLTP DATABASE LUZ *****/
CREATE DATABASE LINK
  LUZ_OLTP
CONNECT TO system
IDENTIFIED BY asus
USING 'localhost:1521/luz';
```

Es importante remarcar que la creación de un *database link* requiere un usuario con permisos elevados. Tanto permisos elevados para ejecutar la orden de creación de un *database link* en la base de datos del DW, como permisos elevados del usuario de la base de datos destino cuyas credenciales se explicitan en el script del *database link*. A partir de la creación del databaselink podemos acceder a las tablas y consultas de la base de datos operacional desde la base de datos del DW.

Distinguiremos 2 procesos de extracción: el proceso de extracción de los datos de las tablas de dimensiones y el proceso de extracción de la tabla de hechos. Ambos procesos se han implementado a través de sendas consultas que operan como *Stagging Area* para los datos transaccionales.

La extracción de datos para las tablas de dimensiones se realiza mediante la consulta (VIEW) `STA_DIMENSIONS_QUERY`. Esta consulta reúne todos los datos necesarios, a través del *Database Link* mencionado anteriormente, y sucesivas operaciones de *JOIN* para reunir los datos necesarios.

La extracción de los datos para la tabla de hechos se realiza mediante la consulta `STA_FACT_QUERY` que a su vez se base en la anterior consulta `STA_DIMENSIONS_QUERY` .

3.3.11. Procesos de transformación

Distinguiremos las transformaciones realizadas en las tablas de dimensiones y las transformaciones realizadas en la tabla de hechos.

En las tablas de dimensiones no se produce ningún tipo de transformación más allá de las operaciones necesarias para generar las diferentes jerarquías en las tablas de dimensiones.

En la tabla de hechos se producen 3 transformaciones relevantes:

- ✓ En primer lugar se calculan los diferentes ID (WID, de Warehouse ID) de las dimensiones.
- ✓ En segundo lugar se produce el cálculo del valor de la lectura anterior (véase punto 3.2.3 Definición del gránulo.
- ✓ En tercer y último lugar se determina el precio de la energía en el momento de producirse las lecturas.

El valor del consumo, y el coste del mismo se calculan mediante *triggers* en el proceso de carga de datos.

3.3.12. Procesos de carga del almacén de datos

Se ha implementado una tabla de LOG que registra los diferentes procesos de carga del DW, distinguiendo entre carga/actualización de las dimensiones y carga de la tabla de hechos.

El proceso de carga de las tablas de dimensiones se ha realizado mediante el procedimiento almacenado `ETL_LOAD_DIMENSIONS`. Este procedimiento no requiere parámetro ninguno. El proceso de actualización/inserción de las dimensiones se realiza mediante la instrucción MERGE de PLSQL que nos permite definir de una sola pasada tanto la actualización de las dimensiones como la inserción de los nuevos valores.

El proceso de carga de la tabla de hechos se ha implementado mediante el procedimiento almacenado `ETL_LOAD_FACTS`. Este procedimiento almacenado requiere 2 parámetros, la fecha inicial y la fecha final de los datos que serán cargados en la tabla de hechos. Fechas que hacen referencia al período de registro de los datos de consumo de los contadores y a toda la información relacionada con esta fecha (cambios de precio, compañías con suministro, clientes con suministro, etc…). El procedimiento almacenado para la carga de la tabla de hechos incorpora una lógica de control para evitar la carga de periodos repetidos.

3.3.13. Pruebas del ETL

Se ha generado un completo juego de datos de prueba para comprobar los procesos ETL del DW.

Las pruebas del ETL han finalizado con el siguiente resultado

Nombre de la tabla	Datos cargados	Datos desechados	Comprobaciones realizadas	Observaciones
W_City_D	OK	OK	Verificación del número de registros cargados Verificación de las ciudades cargadas	
W_Company_D	OK	OK	Verificación del número de registros cargados Verificación de las compañias cargadas	
W_Consumer_D	OK	OK	Verificación del número de registros cargados Verificación de los consumidores cargados	
W_Country_D	OK	OK	Verificación del número de registros cargados Verificación de los paises cargados	
W_Date_D	OK	OK	Verificación del número de registros cargados Verificación de las fechas cargadas	
W_Meter_D	OK	OK	Verificación del número de registros cargados Verificación de los contadores cargados	

Tabla 27. Comprobaciones procesos ETL

Nombre de la tabla	Datos cargados	Datos desechados	Comprobaciones realizadas	Observaciones
W_Time_D	OK	OK	Verificación del número de registros cargados y comprobación de las horas	Realmente solo es necesario cargar esta tabla una vez con las 24 horas el día.
W_Consumption_F	OK	OK	Verificación del número de registros Verificación de los datos de "lectura anterior" Verificación de los datos de precio de la energía.	El proceso de carga de los datos de hechos es bastante lento.
Fact_Control_DATE	OK	OK	Verificación del LOG de los procesos ETL (dimensiones y hechos)	

Tabla 28. Comprobaciones procesos ETL (Continuación)

3.4. Diseño físico

La fase de diseño físico es la última etapa en la construcción del DW. El objetivo de esta fase es dotar a nuestro diseño de la estructura física que le permita almacenar la información en un soporte físico no volátil, optimizar las políticas de acceso a la base de datos (almacenamiento, fragmentación, creación de índices) y establecer las condiciones de seguridad para el acceso de los datos.

Los pasos que se seguirán son:

- ✓ La creación de una BBDD del tipo datawarehouse, incluyendo los metadatos necesarios en el sistema ORACLE para soportar el diseño lógico realizado
- ✓ La aplicación del diseño lógico según los apartados anteriores.
- ✓ La optimización de las políticas de acceso a los datos (índices y fragmentación)
- ✓ La estructura de seguridad para el acceso a los datos.

3.4.1. Creación de la BBDD del DW

El servidor en el que se ubicará el DW es el mismo que almacena la BBDD operacional. Esta situación no es especialmente favorable en términos de rendimiento, pero para testear el entorno de prueba del DW será suficiente. Los pasos realizados son:

- ✓ Crear una nueva BBDD con bdca (DBCA en inglés de Oracle Data Base Creation Assistant) [9] que se encuentra en `/u01/app/Oracle/product/11.2.0/dbhome_1/bin`
  Hemos decidido crear la base de datos con esta herramienta porque nos proporciona las siguientes ventajas:
  - ✓ Optimización del rendimiento (al seleccionar la opción almacén de datos)
  - ✓ Verificar la correcta configuración de los discos compartidos para cada *tablespace*
  - ✓ Configuración de los servicios de red Oracle
  - ✓ Arrancado de la instancia de la base de datos y de los listener correspondientes.

Los pasos de la creación de la BBDD con la herramienta DBCA no difieren en exceso de los realizados en el apartado 2.3, salvo que en el paso representado por la ilustración 4 se ha seleccionado la opción de almacén de datos (se mostrarán sólo las pantallas que muestran diferencias):

Ilustración 25. Ejecución de dbca para el DW LUZ

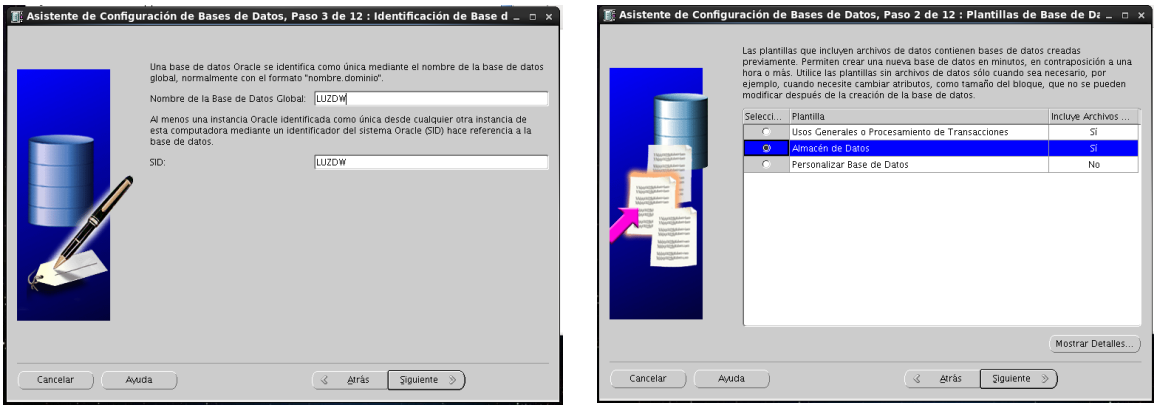


Ilustración 26. Parámetros de la BBDD LUZDW

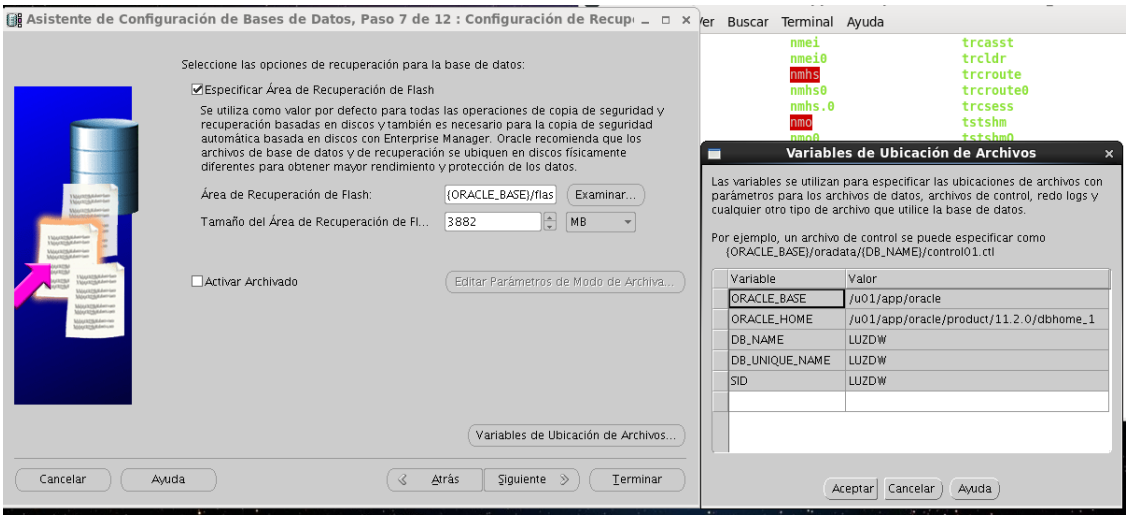


Ilustración 27. Selección del tipo de datos DW para LUZDW

En el apartado tamaño (*sizing*) define el tamaño del bloque (página) de la base de datos. Es recomendable utilizar bloques de tamaño grande (16.384 bytes) [9] para optimizar los procesos del DW (un mayor bloque supone más registros almacenados en la misma página, lo cual reduce las operaciones de I/O). En el caso que se presenta en la ilustración 27, las limitaciones del entorno de prueba impiden este tipo de optimización.

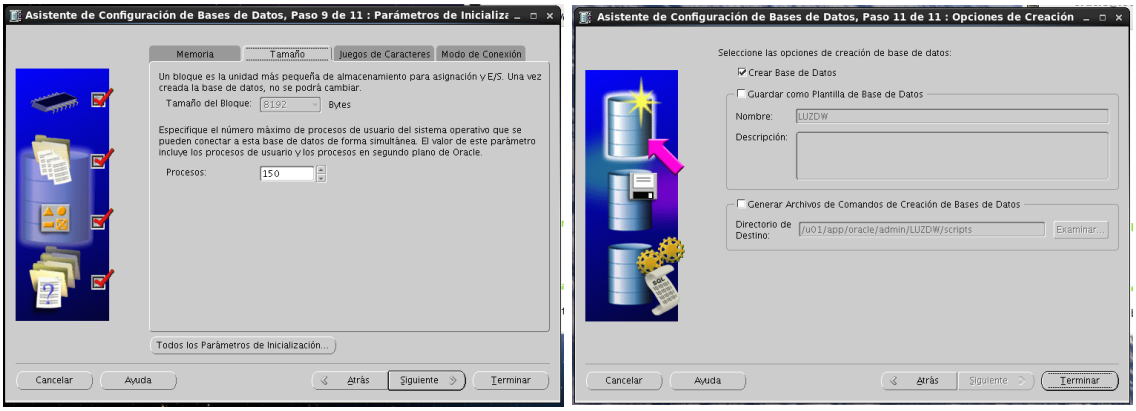


Ilustración 28. Tamaño de la página física de la BBDD del DW y datos finales de configuración

La ausencia de ningún tipo de información sobre el tamaño de los archivos de la BBDD tiene su explicación en el formato de instalación. Cuando se usa una base de datos preconfigurada no es posible controlar el tamaño de los archivos sólo la ubicación de los mismos.

Una vez finalizado el asistente, la BBDD ha sido creada correctamente. Podemos iniciar la consola de la BBDD para comprobar su funcionamiento con la siguiente orden:

```
./emctl start dbconsole
```

Es posible obtener el error "*environment variable ORACLE_UNQNAME not defined*" ya que disponemos de varias bases de datos en el mismo servidor (obviamente se trata de un entorno de prueba, en un entorno de producción sería más indicado disponer de un servidor para cada BBDD).

Ilustración 29. Error de variable UNQNAME al iniciar la consola

Realizamos manualmente un export de la variable con:

```
export ORACLE_UNQNAME=LUZDW
```

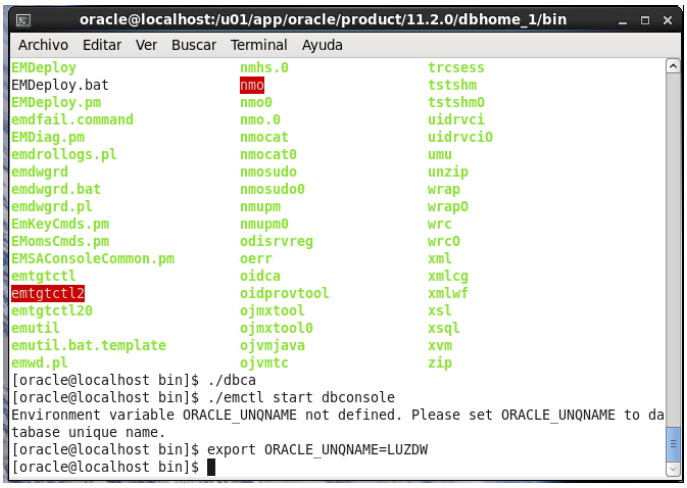


Ilustración 30. Configurando la variable ORACLE_UNQNAME para el inicio de la consola de Oracle

Y ya es posible iniciar la consola general de la BBDD con la orden `./emctl start dbconsole`

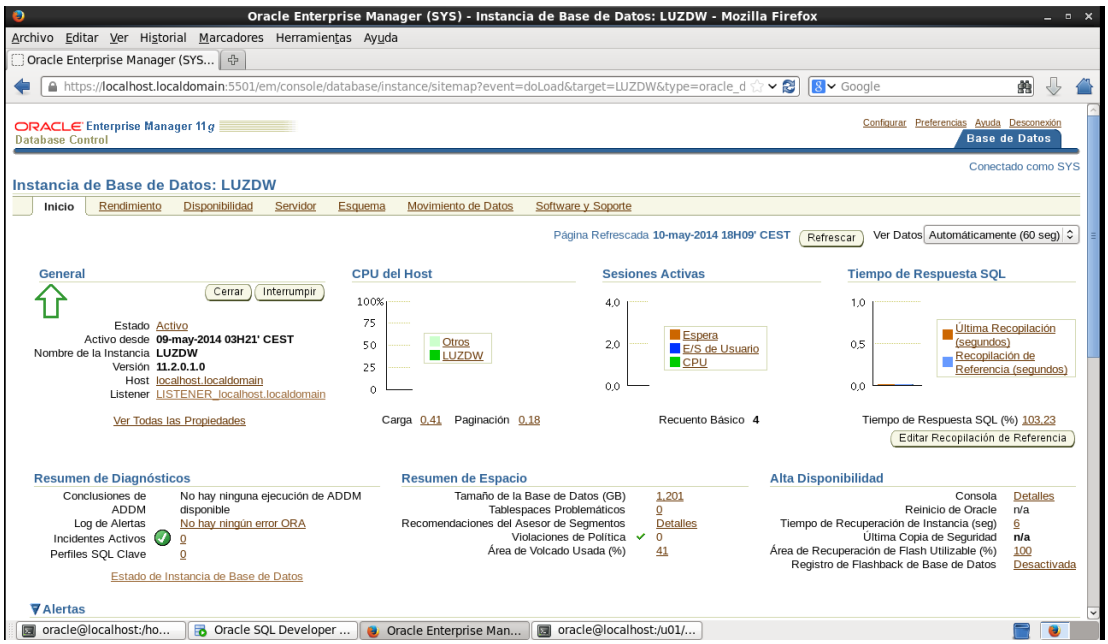


Ilustración 31. Enterprise Manager en LUZDW

Todos los objetos de la base de datos necesitan ubicarse en un esquema por lo que el siguiente paso será la creación de un esquema en la base de datos.

3.4.2. Creación del esquema y almacenamiento para la BBDD

Los diferentes objetos de la BBDD del DW se estructuraran en el esquema lógico *dw*. La creación del esquema requiere primero la creación de un usuario y la asignación de permisos a ese usuario. Los detalles de implementación se encuentran en el anexo VI.

De acuerdo con la previsión del ingente número de registros que va almacenar la BBDD del DW cobra especial importancia planificar adecuadamente como se va a desarrollar el almacenamiento de las diferentes entidades de la BBDD.

El almacenamiento en ORACLE se organiza mediante lo que se denomina TABLESPACES. El TABLESPACE es el nombre lógico usado en el ámbito de un SCHEMA para especificar la ubicación donde deben residir los objetos de la base de datos. La definición de un TABLESPACE incluye la ubicación física de los ficheros, el tamaño, el tipo de crecimiento, etc…

La definición de los TABLESPACES en el DW quedará:

#	Nombre TABLESPACE	Descripción	Observaciones
1	DIMENSIONS	Para almacenar todos los datos de las entidades de dimensiones	
2	DEFAULT	El TABLESPACE por defecto para todos los usuarios	
3	SUMMARY	Para las consultas materializadas que se creen	
4	FACTS_01 …FACTS12	Un TABLESPACE por cada mes para almacenar los datos de la entidad de hechos	El número de registros por mes se ha evaluado en un máximo de 19.944 millones, podría ser recomendable el uso de *tablespaces* del tipo *bigFile* o la creación de ficheros adicionales para cada *TABLESPACE*. Será importante añadir la cláusula *AUTOEXTEND ON* para habilitar el crecimiento automático de los ficheros.
5	FACTS_INDEX	Para alojar los índices de la tabla de hechos	

Tabla 29. Lista de TABLESPACES del DW

El espacio inicialmente asignado a cada *tablespace* y el factor de crecimiento también son datos que pueden influir negativamente en el rendimiento de la solución. En base a los criterios de dimensionamiento establecidos es necesario evaluar el tamaño de la tabla de hechos para determinar el espacio inicial ideal:

De esta forma el espacio de la tabla se define como: $V_T = N_B * T_B$ donde N_B es el número de bloques y T_B es el tamaño del bloque.

El tamaño de un bloque T_B viene definido por el tamaño de la cabecera (C_B) y el tamaño de los datos (D_B). La fórmula para calcularlo es $T_{bloque} = C_{bloque} + D_{bloque}$.

No obstante para determinar el espacio de los datos (D_B) es necesario determinar el valor del espacio libre distribuido (EL, o lo que es lo mismo, la relación PCTFREE/PCTUSED) que limitan el espacio útil (EU). Así tendremos que $D_{bloque} = EL_{bloque} + EU_{bloque}$. Que podemos expresar como $EL_{bloque} = \frac{PCTFREE * D_{bloque}}{100}$ y $EU_{bloque} = \frac{(100 - PCTFREE) * D_{bloque}}{100}$

Para estimar el valor del número de bloques tendremos que $N_B = \overline{FB} * N_{filas}$, donde FB es el valor del factor de bloqueo medio $\overline{FB} = \frac{EU_{bloque}}{\overline{T_{fila}}}$ teniendo que EU_{bloque} es el espacio útil dentro del bloque.

El tamaño de las filas de la tabla es:

	Campo	Tipo dato	Espacio	Total
1	consumption_WID	INTEGER	4 bytes	
2	date_WID	INTEGER	4 bytes	
3	time_WID	INTEGER	4 bytes	
4	meter_WID	INTEGER	4 bytes	
5	address_WID	INTEGER	4 bytes	
6	company_WID	INTEGER	4 bytes	
7	consumer_WID	INTEGER	4 bytes	
8	previousReading	INTEGER	4 bytes	
9	actualReading	INTEGER	4 bytes	
10	consumption	INTEGER	4 bytes	
11	price	NUMBER	22 bytes	Un tipo *number* puede ocupar entre 0 y 22 bytes. Asumimos el peor de los casos (22)
12	cost	NUMBER	22 bytes	
TOTAL			**84 bytes**	

Tabla 30. Cálculo del tamaño de una fila de la tabla de hechos

Así pues tenemos que T_{fila}=84 bytes

El tamaño de bloque por defecto es de 8Kb, aunque ya hemos citado que el valor idóneo en una BBDD utilizada para un DW podría ser 16Kb. Tomando el valor de 16 Kb tenemos que el espacio útil sería: EU=16*1024-84=16.300 bytes.

Todos los bloques tienen una cabecera de 86 bytes. La cabecera de fila usa 3 bytes. Las filas de longitud inferior a 250 bytes usan 1 byte de marca de longitud (las filas de longitud superior a 250 bytes utilizan 3 bytes) [8].

El parámetro PCTFREE se fija inicialmente a 0, es decir, evaluaremos el espacio necesario en base a la premisa de que se ocupa todo el espacio disponible en el bloque para el almacenamiento de los datos. Para el almacenamiento de los índices consideraremos un valor de PCTFREE=33.

El factor de bloqueo medio quedaría como $\overline{FB} = \frac{EU_{bloque}}{\overline{T_{fila}}} = \frac{16.300\ bytes}{84\ bytes} = 194\ filas\ por\ bloque$

El número de bloques para 19.944 millones de filas sería $N_{bloque} = \frac{19.944.000.000\ filas}{194 \frac{filas}{bloque}} = 102.804.124\ bloques$

Por lo tanto el tamaño de la tabla sería de $V_T = N_B * T_B = 102.804.124\ bloques * 16\ \frac{Kb}{bloque} = 1.644.865.979\ KB \equiv 1.606.314\ MB \equiv 1.569\ GB$

Para un conjunto de contadores similar al de España, la explotación del DW de un año completo podría suponer un almacenamiento de 12*1.600 GB= 19.200GB≈ 18,75 TB aproximadamente.

Una vez que hemos determinado es espacio de los datos es necesario evaluar también el espacio que ocuparán los índices de las tablas. Y el principal índice será el de la tabla de hechos. Oracle genera índices de tipo B+ automáticamente para las claves primarias. Las entradas de un índice en un árbol B+ están representadas siempre en las hojas. La entrada consiste en el valor que se indiza más un puntero al resto de la fila (ROWID). Supondremos que ROWID sólo se compone de FICHERO, BLOQUE y FILA y por lo tanto tiene 12 bytes (4 bytes para cada descriptor).

Así pues tendremos que el tamaño de cada entrada en una hoja vendrá definido por el tamaño del campo clave (consumption_WID) más el tamaño del ROWID: $T_{entrada\ hoja} = T_{clave} + T_{ROWID\ ext} = 4\ B + 12\ B = 16B$

Para los punteros internos al árbol vamos a suponer que FILA no es un valor necesario y por lo tanto el puntero ocupará tan solo 9 bytes, ORACLE almacena solo un prefijo separador en los nodos no hoja. El valor del prefijo separador depende de los valores y de su distribución así que utilizaremos para su estimación el tamaño de la entrada. Se obtendrá así, una cota superior para el número de bloques en los niveles internos y altura del árbol.

$$T_{entrada\ nº\ hoja\ max} = T_{clave} + T_{ROWID\ int} = 4B + 9B = 13B$$

Como ya hemos comentado anteriormente, el valor del espacio libre de los bloques de los índices se estimará en 1/3 del total (PCTFREE=33). Así tendremos que,

$$EU_{bloque} = \frac{(100 - PCTFREE) * D_{bloque}}{100} = \frac{(100 - 33) * 16.300}{100} = 10.921$$

Para calcular el número de bloques hoja, restaremos en primer lugar el espacio ocupado por los punteros internos que enlazan estos para el recorrido secuencial. El número de bloques se calcula de forma análoga que en el cálculo para las tablas de datos

$$EU_{bloque\ nodo\ hoja} = 10.921 - 2 * T_{ROWID\ int} = 10.921 - 2 * 9 = 10.903B$$

El factor de bloqueo medio quedaría como $\overline{FB} = \frac{EU_{bloque\ nodo\ hoja}}{T_{entrada\ hoja}} = \frac{10.903\ ytes}{16\ bytes} = 681\ filas\ por\ bloque$

El número de bloques para 19.944 millones de filas sería $N_{nodo\ hoja} = \frac{19.944.000.000\ filas}{681 \frac{filas}{bloque}} = 29.286.344\ bloques$

Para el segundo nivel del tendremos que N_{bloque} -1=29.286.344 bloques -1=29.286.343 entradas separadoras para almacenar. No existen punteros internos para el recorrido secuencial

$$\overline{FB_{nodo\ nº\ hoja}} = \frac{EU_{bloque\ nodo\ hoja}}{\overline{T_{entrada\ hoja}}} = \frac{10.903B}{13B} = 839B$$

El número de bloques para almacenar el segundo nivel del árbol sería de $N^2_{nodo\ hoja} = \frac{29.286.344\ entradas}{839\frac{entradas}{bloque}} = 34.906\ bloques$

Para el tercer nivel del índice tendríamos de forma análoga que N_{bloque} -1=34.906 bloques -1=34.905 entradas separadoras para almacenar. No existen punteros internos para el recorrido secuencial

$$\overline{FB_{nodo\ nº\ hoja}} = \frac{EU_{bloque\ nodo\ hoja}}{\overline{T_{entrada\ hoja}}} = \frac{34.905B}{13B} = 2685B$$

El número de bloques para almacenar el tercer nivel del árbol sería de $N^3_{nodo\ hoja} = \frac{34.905\ entradas}{2685\frac{entradas}{bloque}} = 13\ bloques$

El cuarto nivel del árbol se podría almacenar en un solo bloque. De esta forma tendríamos el tamaño total del índice como $T_{índice} = N_{bloques} * T_{bloques} = (29.286.344 + 34.906 + 13 + 1) * 16KB = 469.140.224KB \equiv 458.145\ MB \equiv 45\ GB \equiv\ 0.04\ TB$ como espacio necesario mínimo para el tablespace mensual que almacene los índices de la tabla de hechos

Es necesario contar con el nivel de permisos adecuado para poder ejecutar las órdenes de creación de TABLESPACES (usuario system). La implementación de los tablespaces en ORACLE se encuentra detallada en el anexo VI.

3.4.3. Índices

Durante la creación de las tablas se han establecido los siguientes índices de acuerdo con el modelo lógico:

Nombre de índice	TABLA	Atributo	Tipo
PK_W_Time_D	W_TIME_D	time_WID	INTEGER
PK_W_Meter_D	W_METER_D	meter_WID	INTEGER (sintético)
PK_W_Address_D	W_ADDRESS_D	address_WID	INTEGER (sintético)
PK_W_COMPANY_D	W_COMPANY_D	company_WID	INTEGER (sintético)
PK_W_CONSUMER_D	W_CONSUMER_D	consumer_WID	INTEGER (sintético)
PK_W_Date_D	W_DATE_D	date_WID	DATE
PK_W_CONSUMPTION_F	W_CONSUMPTION_F	consumption_Wid	INTEGER (sintético)

Tabla 31. Índices del DW

En esta fase del proyecto intentaremos proponer nuevos índices derivados de las necesidades operativas del DW, y no en base al diseño lógico establecido.

Una correcta y acertada valoración de los índices necesarios debe realizarse en base a las consultas realizadas más frecuentes y el análisis de los tiempos de acceso. Además es necesario valorar la ganancia de tiempo en el acceso a los datos gracias a los índices contra la pérdida de tiempo en el mantenimiento de los índices y la cantidad de espacio necesario para el

almacenamiento de los mismos. En cualquier caso, y basándonos en los consejos de [7] y [9] se propondrían la creación de los siguientes índices:

#	TABLA	Atributo	Nombre del índice	Observaciones
1	W_CONSUMPTION_F	date_WID	Ix_ date_WID	Almacenamiento en el tablespace FACTS_INDEX
2	W_CONSUMPTION_F	time_WID	Ix_ time_WID	Almacenamiento en el tablespace FACTS_INDEX
3	W_CONSUMPTION_F	meter_WID	Ix_ meter_WID	Almacenamiento en el tablespace FACTS_INDEX
4	W_CONSUMPTION_F	address_WID	Ix_address_WID	Almacenamiento en el tablespace FACTS_INDEX
5	W_CONSUMPTION_F	company_WID	Ix_ company_WID	Almacenamiento en el tablespace FACTS_INDEX
6	W_CONSUMPTION_F	consumer_WID	Ix_ consumer_WID	Almacenamiento en el tablespace FACTS_INDEX

Tabla 32. Propuesta de índices

En el Anexo VI se encontrará la codificación de los índices propuestos de la tabla anterior.

3.4.4. Particiones/Fragmentación

Como ya hemos visto las dimensiones de la tabla de hechos (W_CONSUMPTION_F) hace necesario aplicar patrones de fragmentación para optimizar el rendimiento del almacenamiento. Queda por determinar la estrategia de fragmentación.

Por la disposición de los TABLESPACES parece que la mejor estrategia es una fragmentación de tipo horizontal (subconjuntos de tuplas) derivada (en base a predicados definidos sobre atributos de otras relaciones, en este caso la tabla de dimensiones de tiempo). En este caso el esquema de asignación para la fragmentación vendría dado por el atributo mes (*monthNumber*) de la tabla de dimensiones de tiempo W_Date_D.

En una partición horizontal derivada la tabla hijo hereda la partición de la tabla padre a través de la referencia a la clave externa. Por eso la tabla W_DATE_D, pese a ser una tabla de dimensiones se almacenará en los TABLESPACES determinados para la tabla de hechos. De esta forma no es necesario añadir en la tabla de hechos el atributo *monthNumber* para realizar la partición, y podemos definir la partición por referencia de la clave externa.

VALOR DE monthNumber en W_DATE_D	PARTICION	TABLESPACE	OBSERVACIONES
1	PART1	FACTS_01	Tablespace para ENERO
2	PART2	FACTS_02	Ídem FEBRERO
3	PART3	FACTS_03	Ídem MARZO
4	PART4	FACTS_04	…
5	PART5	FACTS_05	
6	PART6	FACTS_06	
7	PART7	FACTS_07	
8	PART8	FACTS_08	
9	PART9	FACTS_09	
10	PART10	FACTS_10	
11	PART11	FACTS_11	
12	PART12	FACTS_12	Tablespace para DICIEMBRE

Tabla 33. TABLESPACES según particiones definidas

El detalle de la aplicación práctica del diseño de las particiones se encuentra en los scripts de creación de las tablas en el anexo VII.
El resto de entidades (de dimensiones) no requiere la aplicación de patrones de fragmentación por la volumetría de las tablas.

3.4.5. Seguridad

La operativa del DW de acuerdo al diagrama de casos de uso requiere de 2 usuarios bien diferenciados: Un usuario con capacidad para realizar los procesos ETL, y otro usuario para realizar las consultas diseñadas.

El usuario con capacidad para realizar los procesos ETL, la creación de las tablas y demás elementos del DW es dw. En el anexo VI puede encontrarse la definición lógica del mismo.

De igual forma, el usuario que ejecuta el conjunto de consultas del DW se denomina *dwQuery*. En el anexo VI puede encontrarse la definición lógica del mismo.

3.5. Summary Tables

Una eficiente gestión del DW requiere la creación de tablas sumarizadas (*summary tables*) que permitan el acceso rápido a la información solicitada frecuentemente, en nuestro caso, la información requerida en las consultas a desarrollar.

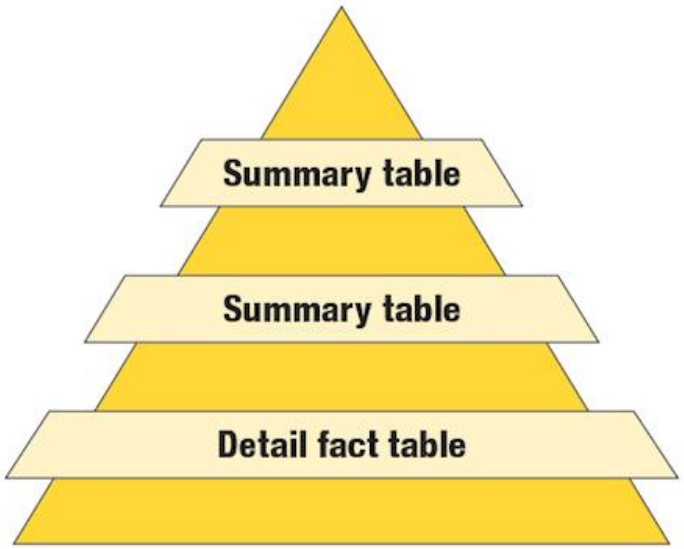


Ilustración 32. Pirámide de tablas resumen. Fuente [11]

La característica fundamental de estas tablas es que son mucho más pequeñas que la tabla de hechos general y, por lo tanto, la consulta de las mismas es inmediata. Conforme vamos descendiendo en la pirámide de tablas resumen las tablas tienen mayor cantidad de datos y las consultas resultan más costosas, aunque disponemos de mayor riqueza en los datos.

La estrategia de creación de las tablas resumen requiere evaluar de qué forma se van a crear y actualizar para evitar que contengan datos no actualizados. Básicamente existen tres formas de construir una tabla resumen [9]:

- ✓ Mediante disparadores en la tabla de Hechos: De manera que al actualizar la tabla de hechos se actualizan también las tablas resumen.
- ✓ Durante el proceso ETL: El mismo proceso que carga de la tabla de hechos, procede a cargar las tablas resumen.
- ✓ Proceso posterior al ETL: Una vez se han desarrollado los procesos ETL se procede a reconstruir (REBUILT) las tablas resumen.

La estrategia implementada ha sido mediante disparadores para respetar los condicionantes expuestos en el enunciado del problema. Mediante el trigger de actualización de datos de la tabla de hechos se rellenan las diferentes tablas resumen.

Las tablas resumen que se han creado son:

#	Tabla resumen	Observaciones
1	W_CITY_SUMMARY_M	Resume los datos de la tabla de hechos de la jerarquía City de Address agrupados por mes y año
2	W_CITY_CONS_SUMMARY_M	Resume los datos de cada consumidor por la jerarquía City de Address agrupados por mes y año
3	W_CITY_METER_SUMMARY_M	Resume los datos de cada contador por la jerarquía City de Address agrupados por mes y año
4	W_COUNTRY_SUMMARY_M	Resume los datos de la tabla de hechos de la jerarquía Country de Address agrupados por mes y año
5	W_COUNTRY_SUMMARY_Y	Resume los datos de la tabla de hechos de la jerarquía Country de Address agrupados por año. La tabla W_COUNTRY_SUMMARY_M alimenta mediante un disparador esta tabla
6	W_COUNTRY_CITY_SUMMARY_M	Resume los datos de consumo de cada ciudad por mes y año, indicando además el país al que pertenece cada ciudad.
7	W_COMPANY_SUMMARY_M	Resume los datos de la dimensión Company agrupados por mes y año
8	W_CONSUMER_SUMMARY_M	Resume los datos de la dimensión Consumer agrupados por mes y año
9	W_METER_SUMMARY_M	Resume los datos de la dimensión Meter agrupados por mes y año
10	W_CITY_METER_SUM_T	Resume los datos de la dimensión Meter agrupados por ciudad

Tabla 34. Summary Tables

Como se puede observar las tablas resumen están muy ligadas a las entidades que aparecen como dimensiones del DW. Los scripts de la creación de las tablas resumen del DW se encuentran en el anexo VII.

3.6. Diseño e implementación de las consultas

En el anexo XI se puede encontrar una breve explicación sobre la construcción de cada consulta y el detalle de las comprobaciones realizadas para asegurar la corrección del resultado.

3.6.1. Consulta 1

La consulta debe devolver el consumo eléctrico mensual de cada contador medido en Kw/h.

Esta consulta se implementa como un simple *SELECT* a la tabla resumen W_METER_SUMMARY_M cuyo código incluimos a continuación:

```
CREATE OR REPLACE VIEW CONSULTA1 AS
SELECT * FROM W_METER_SUMMARY_M ORDER BY YEARNUMBER, MONTHNUMBER, METER_WID;
```

3.6.2. Consulta 2

Dada una ciudad y un mes como parámetros, el listado de todos los contadores cuyo consumo mensual ha superado el 80% del consumo medio de todos los contadores de la ciudad en ese mismo periodo de tiempo. Todo ello ordenado de forma ascendente por el tanto por ciento de consumo eléctrico consumido

La consulta ha resultado ser un poco más compleja. Utiliza una subconsulta sobre la tabla W_CITY_SUMMARY para recopilar el valor del 80% del consumo medio de todos los contadores de una ciudad para el periodo de tiempo buscado. A partir de aquí se utiliza el valor de esa subconsulta en la cláusula *WHERE* de la consulta general sobre la tabla W_CITY_METER_SUMMARY_M. Esta tabla contiene el consumo de todos los contadores de cada ciudad por periodo de año y mes.

```
CREATE OR REPLACE VIEW CONSULTA2 AS
SELECT
  CITY_WID
  , YEARNUMBER
  , MONTHNUMBER
  , METER_WID
  , CONSUMPTION_M
  , (SELECT CONSUMPTION_M FROM W_CITY_SUMMARY_M WHERE CITY_WID=T.CITY_WID AND
YEARNUMBER=T.YEARNUMBER AND MONTHNUMBER=T.MONTHNUMBER)*80/100 AS TOTAL80
FROM W_CITY_METER_SUMMARY_M T
WHERE (T.CONSUMPTION_M>(SELECT CONSUMPTION_M FROM W_CITY_SUMMARY_M WHERE
CITY_WID=T.CITY_WID AND YEARNUMBER=T.YEARNUMBER AND MONTHNUMBER=T.MONTHNUMBER)*80/100);
```

3.6.3. Consulta 3

Dados un mes y un país concretos, consumo eléctrico de cada ciudad como suma de todos los contadores instalados.

La consulta es trivial recogiendo los datos de la tabla resumen W_COUNTRY_CITY_SUMMARY_M que recoge los consumos y el coste de la energía para cada ciudad correspondiente a un país, acumulado por mes y año.

```
CREATE OR REPLACE VIEW CONSULTA3 AS
SELECT YEARNUMBER, MONTHNUMBER, COUNTRY_WID, CITY_WID, CONSUMPTION_M FROM
W_COUNTRY_CITY_SUMMARY_M
ORDER BY YEARNUMBER, MONTHNUMBER, COUNTRY_WID, CITY_WID;
```

3.6.4. Consulta 4

Dada una empresa de suministro y un mes concreto, porcentaje de lecturas de contadores realizadas de forma correcta. Se considera como tales aquellas en las que la conexión se ha efectuado de forma correcta y se ha podido leer la lectura.

Esta consulta se implementa como un *SELECT* sobre la tabla resumen W_COMPANY_SUMMARY_M, donde se utilizan los valores de las columnas CORRECTREADINGS (filas con lecturas correctas, es decir que tienen un consumo mayor que 0), y TOTALREADINGS (total de lecturas) para calcular el porcentaje que requiere la consulta.

En el ejemplo que se acompaña se utilizan los valores de compañía 53 y de mes =3. Aunque la consulta se puede introducir en un procedimiento almacenado que tenga esos parámetros de entrada.

```
CREATE OR REPLACE VIEW CONSULTA4 AS
SELECT      COMPANY_WID,      YEARNUMBER,      MONTHNUMBER,CORRECTREADINGS,TOTALREADINGS,
TO_CHAR((CORRECTREADINGS/TOTALREADINGS)*100, '999.99') AS READINGS_OK
FROM W_COMPANY_SUMMARY_M
ORDER BY YEARNUMBER;
```

3.6.5. Consulta 5

Listado de los contadores que tienen un determinado número de años de antigüedad

Esta consulta no requiere de ninguna tabla resumen. Se puede obtener el dato directamente desde la tabla de dimensiones W_METER_D donde uno de los atributos de esta tabla es la fecha de instalación del contador, o incluso desde la propia tabla de contadores del sistema operacional. La única dificultad recae en la fórmula para calcular la edad. Así pues tendríamos:

```
CREATE OR REPLACE VIEW CONSULTA5 AS
SELECT    METER_WID,TO_CHAR(METERINSTALLATIONDATE,    'DD/MM/YYYY')    AS    METERDATE,
trunc((to_number(to_char(sysdate,'yyyymmdd'))-
to_number(to_char(METERINSTALLATIONDATE,'yyyymmdd')))/10000) AS AGE
FROM W_METER_D
ORDER BY METERINSTALLATIONDATE;
```

3.6.6. Consulta 6

Dada una ciudad y un año concretos, el valor medio de la energía consumida, teniendo en cuenta que el coste de la energía es variable.

Esta consulta resulta trivial calculada sobre la tabla W_CITY_SUMMARY_Y. Esta tabla se carga como consecuencia de un *trigger* establecido en la tabla W_CITY_SUMMARY_M.

```
CREATE OR REPLACE VIEW CONSULTA6 AS
SELECT         YEARNUMBER,         COUNTRY_WID,         CONSUMPTION,         ENERGYCOST,
TO_CHAR((CONSUMPTION/ENERGYCOST),'99.999999') AS AVERAGE_COST FROM W_COUNTRY_SUMMARY_Y
ORDER BY YEARNUMBER, COUNTRY_WID;
```

3.6.7. Consulta 7

Para cada contador instalado, el precio total de la energía consumida mensualmente.

La consulta utiliza la tabla W_METER_SUMMARY_M para listar los consumos de cada contador por mes y año.

```
CREATE VIEW CONSULTA7 AS
SELECT YEARNUMBER, MONTHNUMBER,METER_WID,ENERGYCOST
FROM W_METER_SUMMARY_M
ORDER BY YEARNUMBER, MONTHNUMBER, METER_WID;
```

3.6.8. Consulta 8

Top 10 de los contadores que históricamente han tenido mayor consumo en cada una de las ciudades.

Es la consulta que más me ha costado obtener. La consulta utiliza la tabla resumen W_CITY_METER_SUM_T que contiene el consumo total de cada contador por ciudad. Se utiliza la función DENSE_RANK para generar los valores de rango dentro de cada grupo de contadores de una ciudad, y para filtrar los valores menores que 10.

```
CREATE OR REPLACE VIEW CONSULTA8 AS
SELECT * FROM (
  SELECT
    CITY_WID, METER_WID,CONSUMPTION_T
    , DENSE_RANK() OVER (PARTITION BY CITY_WID ORDER BY CONSUMPTION_T DESC) drnk
  FROM W_CITY_METER_SUM_T)
WHERE drnk<=10
ORDER BY CITY_WID, METER_WID, CONSUMPTION_T DESC;
```

3.6.9. Consulta 9

Consumo medio de todos los consumidores por cada ciudad y mes.

La consulta utiliza la tabla W_CITY_SUMMARY que tiene el consumo mensual de cada ciudad por mes y año, y el número de consumidores por cada ciudad, año y mes. La media se obtiene dividiendo el consumo total por el número de consumidores.

```
CREATE VIEW CONSULTA9 AS
SELECT CITY_WID, CONSUMPTION_m/CONSUMERNUMBER AS AVERAGE_CONSUM FROM W_CITY_SUMMARY_M
ORDER BY YEARNUMBER, MONTHNUMBER, CITY_WID;
```

4. Conclusiones

4.1. Seguimiento de la planificación. Análisis crítico

4.1.1. PAC1

La entrega de la PAC1 se sucede sin problemas. La planificación funciona sin incidencias.

El apartado con mayores dudas es la metodología para el desarrollo de la BBDD analítica, ya que no cuento con experiencia en el tema. Tras la búsqueda de información, inicialmente se selecciona la metodología HEFESTO, pero al final el problema se desarrolla siguiendo la metodología de Ralph Kimball [3].

Las referencias fundamentales para este apartado no son de tipo formal (libros) si no que se han basado en el examen de otros TFG presentados, la mayoría, en la UOC.

4.1.2. PAC2

La entrega de la PAC2 se produce con bastantes problemas. El apartado de diseño conceptual del sistema es el que concentra la mayoría de retraso. Incluso, después de aceptar un diseño como el más óptimo se suceden continuas revisiones del mismo en las etapas de diseño lógico, lo cual ocasiona más retraso en el proyecto. El proyecto todavía no tiene una forma definida.

Se decide realizar una nueva instalación de Oracle, esta vez no de la versión Express en Windows sino de la versión Enterprise en Linux (versión 11g) lo cual resulta bastante complicado y tedioso. Con todo, al final la experiencia es muy positiva y contamos con una buena instalación sobre un sistema Linux Centos X64 v.6.5. A mitad del trabajo de desarrollo de los procedimientos almacenados, el sistema se queda sin espacio en disco (20 GB resultó ser muy poco espacio) y es necesario proceder a una ampliación del disco duro virtual, y una replanificación de las particiones en Linux Centos.

Resulta necesario activar los planes de contingencia y habilitar jornadas adicionales que se obtienen tanto de la ampliación de la jornada en fines de semana, como en el aumento de jornada lectiva derivado de la petición de días de vacaciones en el trabajo. El retraso recuperado asciende a aproximadamente una semana.

Las referencias fundamentales para la creación del trabajo en esta entrega son: *Desarrollo de Bases de Datos. Casos prácticos desde el análisis a la implementación 2ª Ed, Oracle PL/SQL Programming 6th Edition* [8] *y Oracle PL/SQL Programming 6th Edition* [6]. Si bien han resultado de inestimable ayuda las referencias de Morgan Kaufmann, *Database Design for Smarties. Using UML for Data Modeling* [2] y *Database Design: Know it all* [4].

La evaluación de la PAC2 es positiva pero se detectan varios problemas que es necesario solucionar: la inclusión de las referencias bibliográficas, la retirada del código de la memoria del TFG para ubicarlo en los anexos, etc… En total un tiempo de reproceso de 2 jornadas que debiera recuperarse con la reducción de tiempo establecida para la parte final de redacción de la memoria.

4.1.3. PAC3

La entrega de la PAC3 también se inicia con bastantes problemas. El principal escollo es el desconocimiento de la práctica para generar un Datawarehouse. Aunque durante las diferentes asignaturas de Base de Datos se ha tocado el tema, nunca se ha profundizado lo suficiente como para poder desarrollar un proyecto.

La tarea de búsqueda de información para la que se había planificado una jornada se alarga hasta una semana y media con el consecuente incumplimiento de los planes establecidos. Durante ese tiempo se consulta infinidad de bibliografía (libros, TFG's, artículos…). Entre la bibliografía revisada se utilizará como referencia básica la publicación *The Data Warehouse Toolkit. The definitive Guide to Dimensional Modelling 3d Ed.* [3] que resultará fundamental. La claridad y ejemplos que contiene el libro consigue despejar la mayoria de dudas sobre el diseño conceptual de la base de datos analítica. La estructura del trabajo de la parte analítica cambia y se adapta a la nueva metodologia.

De nuevo es necesario recurrir a los planes de contingencia para recuperar el retraso acumulado (ampliación fines de semana, y nuevos días de vacaciones en el trabajo). No obstante, quizás por la clarividencia aportada por las referencias o quizás por una sobreestimación de los tiempos planificados en apenas 1 semana adicional se recupera todo el retraso. La solicitud de un aplazamiento en la entrega de 3 días aporta una mayor tranquilidad para el desarrollo, aunque finalmente no resulta necesario recurrir a esos días gracias a la mejora del proceso indicada en el párrafo anterior.

4.1.4. Entrega final

La entrega final se aborda de forma accidentada. Pese al adelanto realizado en diversas materias en la PAC3 de nuevo se producen retrasos. Especialmente en la generación de los procedimientos ETL.

Las consultas de la base de datos también reportan especiales problemas, tanto por el mecanismo de las tablas resumen como por la complejidad de algunas consultas (o incluso la dificultad en obtener una comprobación utilizando funciones de agregado), especialmente la consulta 2. De nuevo la activación de los planes de contingencia permite realizar la entrega con tranquilidad.

Además, el mecanismo seleccionado para la carga de los datos en el DW resulta bastante lento por lo que los extensos juegos de pruebas generados para probar las consultas deben ser reducidos considerablemente, resultando un problema seleccionar un juego que disponga de la suficiente representatividad en las tablas resumen para la comprobación de las consultas.

4.2. Conclusiones

La primera conclusión que se extrae del desarrollo del trabajo tiene que ver con los aspectos de planificación. Es especialmente importante ser riguroso con los tiempos planificados y disponer siempre planes de contingencia para la recuperación de tiempo. Sin estos mecanismos planificados este proyecto no se habría presentado.

Respecto a las materias objeto del proyecto, las bases de datos, sin duda el trabajo engloba todos los conocimientos adquiridos durante el grado. Me considero satisfecho de los productos y artefactos generados.

En referencia al apartado de desarrollo de la base de datos operacional el aspecto más gratificante ha sido el desarrollo ordenado mediante la aproximación de los 3 esquemas. La experiencia ha resultado valiosa para el tratamiento formal de un proyecto de base de datos. Resulta fundamental realizar un buen estudio de los requisitos y confeccionar un modelo conceptual lo más ajustado posible a los mismos.

Los aspectos del diseño físico cobran especial importancia en proyectos de gran envergadura. La falta de costumbre me ha hecho, en ocasiones, caer en el error de las magnitudes y generar conjuntos de prueba demasiado extensos para las modestas capacidades de mi entorno de desarrollo. Una correcta previsión de las magnitudes de la base de datos resulta imprescindible para el desarrollo de bases de datos de grandes dimensiones y almacenes de datos.

Ya en el área de almacenes de datos, extraería dos lecciones importantes. La primera es que el objetivo de un almacén de datos no es obtener la tabla de hechos. El trabajo no acaba aquí. El objetivo es responder a las preguntas que motivaron el hecho de crear el almacén de datos, y responder a éstas, de la forma más eficaz y en el menor tiempo posible. De aquí la importancia de las tablas resumen.

La segunda lección tiene que ver con las características de los artefactos que se generan alrededor del almacén de datos. Las respuestas a contestar con el almacén de datos condicionan, como no podía ser de otra forma, el número, el contenido y los procedimientos de las tablas resumen.

Desde mi punto de vista, el apartado más satisfactorio ha sido el relacionado con el desarrollo del almacén de datos. Pese haber cursado la asignatura de *Data Mining* no había tenido un contacto directo, ni siquiera en mi profesión, con la creación de un almacén de datos. Los conocimientos adquiridos son de gran valor y espero ponerlos en práctica en breve.

4.3. Líneas de trabajo futuro

Las líneas de trabajo futuro que no se han podido explorar en este trabajo y han quedado pendientes son:

- ✓ Desarrollo de los aspectos de particionado en la BBDD operacional: La magnitud de la BBDD aconseja la partición bajo criterios geográficos. No se ha podido desarrollar este aspecto.
- ✓ Desarrollo de los aspectos de seguridad en Oracle relacionados con la Ley de Protección de Datos: Los requisitos de almacenamiento de datos personales entendiendo las connotaciones de confidencialidad (usos y costumbres de los consumidores) y privacidad (datos personales) aconsejan el desarrollo de una política de encriptación de datos. Para ello es necesario aplicar técnicas de encriptación de columnas, encriptación de bases de datos (*Transparent Data Encryption*, y otras opciones específicas de Oracle)
- ✓ Desarrollo de bases de datos analíticas mediante sistemas OLAP: El desarrollo de la base de datos analítica se ha realizado mediante sistemas en estrella. Las consultas se realizan en SQL. Queda pendiente explorar la capacidad semántica de las consultas OLAP.

5. Glosario

- DATA MINING: Minería de datos
- DIAGRAMA E/R: Diagrama Entidad/Relación
- DISEÑO CONCEPTUAL: Aproximación conceptual al diseño de una base de datos
- DISEÑO LÓGICO: Parte del diseño de las bases de datos que consiste en generar las entidades y las relaciones del modelo.
- DW: *Data Warehouse,* almacén de datos
- ETL: Extract*, Transform and Load*: Extracción, Transformación y Carga
- FOREIGN KEY: Clave externa
- LOG: Registro de modificaciones
- OLTP: *On Line Transaction Processing,* procesado de transacciones en línea
- OLAP: *On Line Analytic Processing* , procesado analítico en línea
- PL/SQL: Dialecto SQL específico de Oracle.
- PRIMARY KEY: Clave primaria
- Procedimientos ABM: Procedimientos de Alta, Baja, y Modificación de datos
- Procedimientos CRUD: Create, Read, Update, Delete
- SCRIPT: Proceso por lotes
- SCD: *Slow Changing Dimensions*: Hace referencia a las dimensiones del DW que sufren cambios y a la forma de consignar estos cambios (tipos 1,2, y 3)
- STORE PROCEDURE: Procedimiento almacenado
- TFG: Trabajo Fin de Grado
- TRIGGER: Disparador
- UML*: Unified Model Language:* Lenguaje de modelado unificado

6. Bibliografía

[1] T. Teorey, S. Lighstone y T. Nadeau, Database Modelling and Design. Logical Design 5th. Edition, Burlington: Morgan Kaufmann, 2011.

[2] R. J. Muller, Database Design for Smarties. Using UML for Data Modeling, San Francisco: Morgan Kaufmann, 1999.

[3] R. M. Kimball R., The Data Warehouse Toolkit. The definitive Guide to Dimensional Modelling 3d Ed., Indianapolis: Wiley, 2013.

[4] T. J. Teorey, S. Buxton y L. Fryman, Database Design: Know it all, Burlington: Morgan Kaufmann, 2009.

[5] S. Conger, Hands-On Database. An Introduction to Database Design and Development, New Jersey: Prentice Hall, 2012.

[6] S. Feuertstein y B. Pribyl, Oracle PL/SQL Programming 6th Edition, Gravenstein: O'Reilly Media Inc., 2014.

[7] I. Fernandez, Beginning Oracle Database 11g Administration. From Novice to Professional, New York: Apress, 2009.

[8] D. Cuadra, E. Castro y A. M. Iglesias, Desarrollo de Bases de Datos. Casos prácticos desde el análisis a la implementación 2ª Ed., Madrid: Ra-Ma, 2013.

[9] L. Hobbs, S. Hillson, S. Lawande y P. Smith, Oracle Database 10g Data Warehousing, Oxford: Elsevier Inc., 2005.

[10] R. Stackowiak, J. Rayman y R. Greenwald, Oracle® Data Warehousing and Business Intelligence Solutions, Indianapolis: Wiley Publishing, Inc., 2007.

[11] L. Knutsen, "Building Fast Data Warehouse Schemas: Part 3. Design speedy summary tables," 15 10 2010. [Online]. Available: http://ibmdatamag.com/2010/10/building-fast-data-warehouse-schemas-part-3/. [Accessed 10 06 2014].

7. Anexos

- ✓ Anexo I: Scripts de creación de las tablas y de creación de los disparadores
- ✓ Anexo II: Scripts de pruebas de carga de las tablas de la BBDD
- ✓ Anexo III: Scripts de creación de los procedimientos almacenados de la BBDD
- ✓ Anexo IV: Scripts de prueba de los procedimientos almacenados
- ✓ Anexo V: Scripts de diseño físico de la BBDD (índices, particiones, definición de usuarios y permisos)
- ✓ Anexo VI: Scripts de diseño físico de la BBDD (índices, particiones, definición de usuarios y permisos)
- ✓ Anexo VII: Scripts de creación de las tablas (con esquemas de particionado) y de creación de los disparadores asociados a las tablas.
- ✓ Anexo VIII: Scripts de pruebas de carga de las tablas del DW y de los *triggers*
- ✓ Anexo IX: Scripts de generación de datos de prueba en la BBDD transaccional para las pruebas del proceso ETL y consultas del DW
- ✓ Anexo X: Procedimientos ETL del DW
- ✓ Anexo XI: Consultas del almacen de datos