TFG – MEMORIA 2014

Sistema BBDD – Gestión del Consumo energético

El trabajo propuesto consiste en implementar un sistema de BD para dar respuesta a una necesidad planteada para la comisión europea de la energía que necesita tener controlada la gestión del consumo energético que hacen las diferentes compañías implicadas en el suministro y control de este servicio.

Yago Blanco Presas – TFG

MEMORIA 2014

Contenido MEMORIA

1. Objetivos del trabajo 3
2. Planificación del proyecto 3
3. Entregas 5
4. Materiales técnicos y Recursos 6
5. Plan de Contingencias 7
6. Análisis y diseño 7
   6.1. Análisis de requisitos 7
      6.1.1. Requisitos funcionales. 7
      6.1.2. Requisitos no funcionales. 10
7. Diseño 10
   7.1. Diseño conceptual 10
   7.2. Identificación de las entidades. 11
   7.3. Identificación de las relaciones 12
   7.4. Atributos de las entidades 13
   7.5. Diseño lógico 14
8. Implementación del proyecto 15
   8.1. Instalación SGBD Oracle 15
   8.2. Configuración Oracle y herramientas 17
   8.3. Implementación diseño E/R 19
      8.3.1. Diseño tablas 20
      8.3.2. Diseño Constraint Primary Key / Foreign Key / UNIQUE 25
   8.4. Implementación de funcionalidades 26
      8.4.1. Implementación procedimientos ABM 26
      8.4.2. Implementación procedimientos de consulta y estadísticas 29
   8.5. Testing de pruebas 33
      8.5.1. Diseño del testing de pruebas 33
      8.5.2. Script del testing de pruebas 33
      8.5.3. Ejecución y depuración del testing de pruebas 34
   8.6. Valoración económica y Recurso 34
   8.7. Glosario 38
   8.8. Bibliografía 38
   8.9. Anexos 38

1. Objetivos del trabajo

El objetivo general del TFC de la UOC es el de proporcionar, el diseño e implementación de una BBDD que recoja todas las necesidades para la gestión del consumo energético que hacen las diferentes compañías implicadas en el suministro y control de este servicio.

Uno de los componentes que se está poniendo en marcha en la actualidad son los contadores de electricidad electrónicos. Estos dispositivos están substituyendo los tradicionales ya que, entre otras ventajas, permiten una lectura en remoto y no hace falta tener lectores que pasen a leer mensualmente el consumo, permiten adaptar la tarifa al precio variante de electricidad, permiten reducir las pérdidas de energía.

2. Planificación del proyecto

La planificación que voy a utilizar ***será en base a una metodología en cascada***, que nos definirá el ciclo de vida de todo el proyecto. Pasaremos a realizar una tarea siempre y cuando se ha realizado la tarea previa. Ajustaremos lo máximo posible las fechas indicadas por la UOC para el término de las siguientes entregas:

PAC1 (Planificación del TFC): Diseño del plan de trabajo. Para realizar el diseño del plan de trabajo realizaremos lo más detallado posible.

PAC2 (Análisis de requisitos): Recopilación de la documentación y producto que hayamos generado hasta el momento, si puede ser en un formato pensado ya cara para la entrega final.

PAC3 (Implementacion del proyecto): En esta fase del proyecto realizaremos el desarrollo y la programación de lo definido anteriormente en la PAC2. Por todo ello, habremos de diseñar y crear los scripts de tablas, índices, procedimientos, disparadores… Habremos de implantarlo en un sistema de Gestor de BBDD (ORACLE)

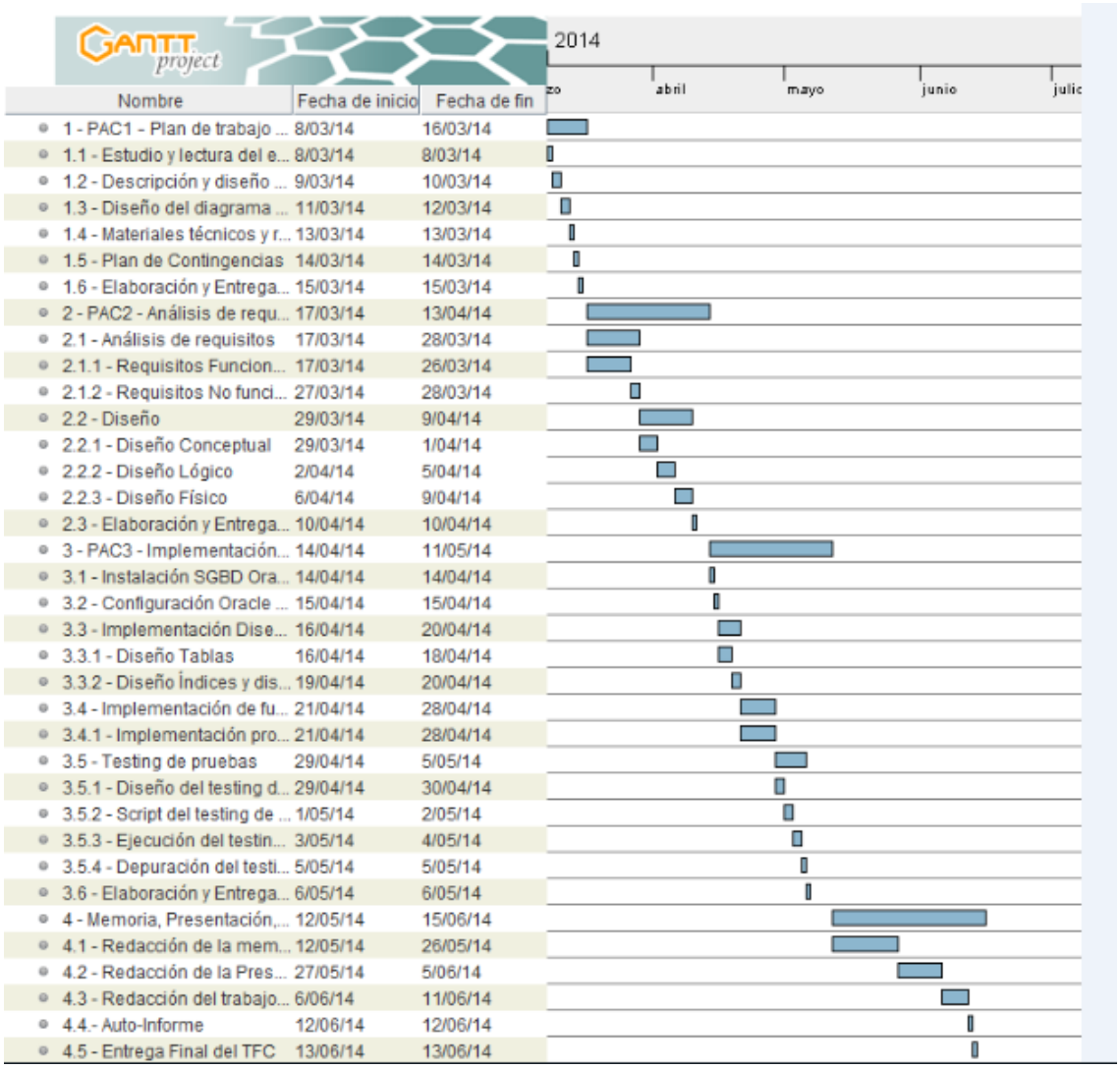
ENTREGA FINAL: En esta entrega se incluirá lo siguiente:

- Memoria
- Presentación
- Trabajo práctico
- Auto-informe de las competencias transversales

Detallamos el diagrama de Gantt, según las especificaciones comentadas anteriormente:

TAREA	DIAS	FECHA INICIO	FECHA FINAL
1 - PAC1 – Plan de trabajo (PT)	**8**	**08/03/2014**	**16/03/2014**
1.1 - Estudio y lectura del enunciado	1	08/03/2014	08/03/2014
1.2 - Descripción y diseño de las tareas a realizar (metodología en cascada)	2	09/03/2014	10/03/2014
1.3 - Diseño del diagrama de Gantt	2	11/03/2014	12/03/2014
1.4 - Materiales técnicos y recursos	1	13/03/2014	13/03/2014
1.5 - Plan de Contingencias	1	14/03/2014	14/03/2014
1.6 - Elaboración y Entrega documentación PAC1	1	15/03/2014	15/03/2014
2 - PAC2 - Análisis de requisitos	**25**	**17/03/2014**	**13/04/2014**
2.1 - Análisis de requisitos	12	17/03/2014	28/03/2014
2.1.1 - Requisitos Funcionales	10	17/03/2014	26/03/2014
2.1.2 - Requisitos No funcionales	2	27/03/2014	28/03/2014
2.2 - Diseño	12	29/03/2014	09/04/2014
2.2.1 - Diseño Conceptual	4	29/03/2014	01/04/2014
2.2.2 - Diseño Lógico	4	02/04/2014	05/04/2014
2.2.3 - Diseño Físico	4	06/04/2014	09/04/2014
2.3 - Elaboración y Entrega documentación PAC2	1	10/04/2014	10/04/2014
3 - PAC3 - Implementación del proyecto	**23**	**14/04/2014**	**11/05/2014**
3.1 - Instalación SGBD Oracle	1	14/04/2014	14/04/2014
3.2 - Configuración Oracle y Herramientas	1	15/04/2014	15/04/2014
3.3 - Implementación Diseño E/R	5	16/04/2014	20/04/2014
3.3.1 - Diseño Tablas	3	16/04/2014	18/04/2014
3.3.2 - Diseño Índices y disparadores	2	19/04/2014	20/04/2014
3.4 - Implementación de funcionalidades	8	21/04/2014	28/04/2014
3.4.1 - Implementación procedimientos ABM	8	21/04/2014	28/04/2014
3.5 - Testing de pruebas	7	29/04/2014	05/05/2014
3.5.1 - Diseño del testing de pruebas	2	29/04/2014	30/04/2014
3.5.2 - Script del testing de pruebas	2	01/05/2014	02/05/2014
3.5.3 - Ejecución del testing de pruebas	2	03/05/2014	04/05/2014
3.5.4 - Depuración del testing de pruebas	1	05/05/2014	05/05/2014
3.6 - Elaboración y Entrega documentación PAC3	1	06/05/2014	06/05/2014
4 - Memoria, Presentación, Trabajo práctico o Producto, Auto-informe	**32**	**12/05/2014**	**15/06/2014**
4.1 - Redacción de la memória	15	12/05/2014	26/05/2014
4.2 - Redacción de la Presentación	10	27/05/2014	05/06/2014
4.3 - Redacción del trabajo práctico	6	06/06/2014	11/06/2014
4.4.- Auto-Informe	1	12/06/2014	12/06/2014
4.4 - Entrega Final del TFC	1	13/06/2014	13/06/2014

A continuación, mostramos el diagrama de Gantt:



Nombre	Fecha de inicio	Fecha de fin
1 - PAC1 - Plan de trabajo ...	8/03/14	16/03/14
1.1 - Estudio y lectura del e...	8/03/14	8/03/14
1.2 - Descripción y diseño ...	9/03/14	10/03/14
1.3 - Diseño del diagrama ...	11/03/14	12/03/14
1.4 - Materiales técnicos y r...	13/03/14	13/03/14
1.5 - Plan de Contingencias	14/03/14	14/03/14
1.6 - Elaboración y Entrega...	15/03/14	15/03/14
2 - PAC2 - Análisis de requ...	17/03/14	13/04/14
2.1 - Análisis de requisitos	17/03/14	28/03/14
2.1.1 - Requisitos Funcion...	17/03/14	26/03/14
2.1.2 - Requisitos No funci...	27/03/14	28/03/14
2.2 - Diseño	29/03/14	9/04/14
2.2.1 - Diseño Conceptual	29/03/14	1/04/14
2.2.2 - Diseño Lógico	2/04/14	5/04/14
2.2.3 - Diseño Físico	6/04/14	9/04/14
2.3 - Elaboración y Entrega...	10/04/14	10/04/14
3 - PAC3 - Implementación...	14/04/14	11/05/14
3.1 - Instalación SGBD Ora...	14/04/14	14/04/14
3.2 - Configuración Oracle ...	15/04/14	15/04/14
3.3 - Implementación Dise...	16/04/14	20/04/14
3.3.1 - Diseño Tablas	16/04/14	18/04/14
3.3.2 - Diseño Índices y dis...	19/04/14	20/04/14
3.4 - Implementación de fu...	21/04/14	28/04/14
3.4.1 - Implementación pro...	21/04/14	28/04/14
3.5 - Testing de pruebas	29/04/14	5/05/14
3.5.1 - Diseño del testing d...	29/04/14	30/04/14
3.5.2 - Script del testing de ...	1/05/14	2/05/14
3.5.3 - Ejecución del testin...	3/05/14	4/05/14
3.5.4 - Depuración del testi...	5/05/14	5/05/14
3.6 - Elaboración y Entrega...	6/05/14	6/05/14
4 - Memoria, Presentación,...	12/05/14	15/06/14
4.1 - Redacción de la mem...	12/05/14	26/05/14
4.2 - Redacción de la Pres...	27/05/14	5/06/14
4.3 - Redacción del trabajo...	6/06/14	11/06/14
4.4.- Auto-Informe	12/06/14	12/06/14
4.5 - Entrega Final del TFC	13/06/14	13/06/14

3. Entregas

Las diferentes entregas se realizarán con unas fechas marcadas por la UOC para realizar un correcto seguimiento de la evaluación continuada del TFC que se evaluará mediante el RAC.

PAC 1 – Planificación del TFC

En esta primera fase se definirá el plan de trabajo a desarrollar. Para ello utilizaré las metodologías de planificación aprendidas para alcanzar el objetivo pretendido. Junto con el plan de trabajo incluiré los recursos humanos, técnicos y materiales necesarios para el desarrollo del proyecto. Definiré que riesgos supondría dicha implantación y que controles debería aplicar para reducirlo.

PAC 2 – Análisis de requisitos

- Análisis de los requerimientos proporcionados en el enunciado.
- Elaboración del diseño conceptual de la Base de Datos. Para ello utilizaré los diagramas de E/R
- Se transforma el esquema conceptual al lógico que utilizará las estructuras de datos del modelo de base de datos en el que se basará el SGBD.
- Se realizará el diseño físico en base al lógico. En este sistema se tendrá en cuenta que el SGBD a la que se va a aplicar será Oracle.

PAC 3 – Implementación

- Se instalará y configurará el SGBD Oracle.
- Elaboración del diseño físico a partir del diagrama E/R realizado en la etapa del diseño físico.
- Este punto incluye el diseño de las tablas, índices, procedimientos y disparadores.
- Se implementará la funcionalidad para los procedimientos de ABM (alta, baja y modificaciones).
- Análisis y diseño de un juego de pruebas. Creación del script y ejecución

ENTREGA FINAL: En esta entrega se incluirá lo siguiente:

- Memoria
- Presentación
- Trabajo práctico
- Auto-informe de las competencias transversales

4. Materiales técnicos y Recursos

Para elaborar todo el proyecto necesitaremos de los siguientes recursos:

- Un jefe de proyecto (que supervisará todo el proyecto)
- Un analista programador que realizará los diseños técnicos
- Un programador para codificar el código SQL

A nivel de materiales técnicos podemos nombrar los siguientes:

- Un ordenador portátil

Y por último, las siguientes aplicaciones para poder implementar todo el proyecto:

- Microsoft Word: para realizar la documentación oportuna

- Adobe Acrobat: Para la transformación de cualquier documento a un documento con extensión pdf.
- Microsoft Excel: Para la realización de tablas.
- Gantt Project: Para hacer el diagrama de Gantt.

- Oracle: El sistema gestor de base de datos.
- PostGresSQL: Para realizar los scripts y ejecutarlos.
- UltraEdit: Para gestionar ficheros planos.
- Magic Drawn: Para hacer diagramas

5. Plan de Contingencias

Existen diversos riesgos o factores que pueden afectar de alguna forma el desarrollo del proyecto y así no tener tiempo de reaccionar para el cumplimiento de las fechas acordadas. Para prevenir estos contratiempos tendré en cuenta estos puntos:

- Para cuando haya un problema de error de datos, error del sistema, problemas del hardware o errores en el SGBD, tendré una copia diaria de información tanto en local como en la nube. En este caso utilizaré dos tipos la de dropbox y la de google drive.

- Cuando por motivos personales (compromisos o enfermedades) o laborales (ampliación del horario laboral) no se pueda mantener con el plan de trabajo, se podrá utilizar los días de fines de semana para ponerse al día.

6. Análisis y diseño

6.1. Análisis de requisitos

En este apartado se incluye todos los requisitos y funcionalidades que tengan que cumplir el diseño de la base de datos. Además de las nuevas propuestas que mejorarán y amplían su funcionalidad.

6.1.1. Requisitos funcionales.

Descripción trabajo

El trabajo propuesto consiste al implementar un sistema de BBDD para dar re puesta a una necesidad planteada por la comisión europea de la energía que necesita tener controlada la gestión del consumo energético que hacen las diferentes compañías implicadas en el suministro y control de este servicio.

Cómo se muy sabido, los gobiernos están poniendo mucho de énfasis en una gestión eficiente de la energía por, entre otros razones, motivos económicos y ecológicos.

Cada vez más, las necesidades energéticas se disparan y es muy importante reducir los costes de este servicio para poder llegar a cubrir todas las necesidades a un coste razonable y sin malograr los recursos naturales de que disponemos.

Entre otras iniciativas, está cogiendo mucha fuerza las relacionadas con el que se denomina “Smart Cities” (ciudades inteligentes) donde, mediante la instalación de varios componentes inteligentes, se intenta optimizar los diferentes recursos energéticos de las ciudades.

Uno de los componentes que se está poniendo en marcha en la actualidad son los contadores de electricidad electrónicos. Estos dispositivos están sustituyendo los tradicionales puesto que, entre otras ventajas, permiten una lectura en remoto y no hay que tener lectores que pasen a leer mensualmente el consumo, permiten adaptar la tarifa al precio variando de la electricidad, permiten reducir las pérdidas de energía,....

Pero estos dispositivos también permiten que las compañías gestoras del servicio tengan acceso a mucha información de los consumidores que podría vulnerar la intimidad de las personas. Por ejemplo, se podría llegar a saber las horas o días en que los consumidores no están en casa. Por este motivo, la comisión europea de la energía quiere regular qué se puede analizar de manera remota, y de qué manera, y nos pide que definimos la estructura de BBDD que sirva de plataforma baso por la gestión de estos contadores electrónicos.

Los requisitos funcionales establecidos por el cliente y las mejoras propuestas para el modelo de datos a diseñar son los siguientes:

R1. El modelo tiene que permitir guardar todos los datos asociados a los contadores electrónicos, a las empresas que los gestionan, a los consumidores a los que dan servicio en los diferentes países europeos y definir como se puede analizar la información recogida.

-Los datos mínimos a guardar, para cada elemento de la BBDD, serían:
- -Asociados a los contadores electrónicos
  - - Contadores: número de serie, modelo, código de contrato asociado, potencia contratada por el cliente (en Kilowatios), fecha de la última inspección técnica, ...
- -Asociados a las empresas
  - -Empresas: NIF, nombre, domicilio fiscal, países donde dan servicio,…
- -Asociados a los consumidores

-Consumidores: número de identificación (DNI, pasaporte, permiso de residencia,..), dirección, código de contrato de contador, datos bancarios, empresa suministradora,...

R2. Recogida de datos e información

-Sólo se permitirá la conexión remota a los contadores en intervalos de una hora.
-Cada conexión se tiene que registrar en una mesa de log donde se indique: fecha y hora de conexión, número de serie de contador leído, consumo del contador en aquel preciso momento y resultado de la lectura (correcta/incorrecta).

R3. Realizar la facturación adecuada a las variaciones del precio de la energía

-Se tendrán que registrar los diferentes cambios de precio. Los datos a guardar serán la fecha y hora del cambio de precio, el país afectado y el precio a partir de aquel momento.

R4. Se realizarán las siguientes funcionalidades en base a los requisitos antes descritos:

-Se implementarán los procedimientos de Alta – Baja – Modificación (ABM) de todas las entidades que se consideren relevantes.

-No se implementarán los procedimientos de Alta – Baja – Modificación (ABM) de las tablas generadas y que no tenga que ver con la funcionalidad anterior

-Se implementará y describirá los procedimientos para la consulta de información.

-Estos procedimientos serán los siguientes:

- El consumo eléctrico mensual de cada contador medido en KW/h.
- Dada una ciudad y un mes como parámetros, el listado de todos los contadores donde el consumo mensual ha superado el 80% del consumo mediano de todos los contadores de la ciudad en aquel mismo periodo de tiempo. Todo esto ordenado de forma ascendente por el tanto por ciento de consumo eléctrico consumido.
- Dados un mes y un país concretos, consumo eléctrico de cada ciudad como suma de todos los contadores instalados.
- Dada una empresa de suministro y un mes concreto, porcentaje de lecturas de contadores efectuadas de forma correcta.

- Se consideran como tal las que la conexión se ha efectuado de manera correcta y se ha podido leer la lectura.

- Listado de contadores que tengan un determinado número de año de antigüedad.

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

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

- Top 10 de contadores que históricamente han tenido más consumo a cada una de las ciudades

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

6.1.2. Requisitos no funcionales.

En este apartado se desarrollara que requisitos no son funcionales:

1. El sistema de gestión de base de datos será Oracle.
2. En el apartado del sistema operativo no se desarrollará en ninguno en concreto ya que Oracle puede ser instalado en cualquier sistema operativo.
3. Política de backups: Este apartado no se contemplará en este desarrollo, ya que solo se nos solicita para desarrollar un sistema para el almacenamiento de dicha información. Este tema se debería de tratar una vez realizada la implementación.
4. Los procedimientos dispondrán como mínimo de un parámetro de salida, llamado RSP de tipo String. Este indicará si la ejecución ha finalizado con éxito 'OK' o si ha fracasado 'ERROR + Tipo de Error'. Todas las llamadas a estos procedimientos se guardarán en una tabla de logs, junto con todos sus datos, parámetros de entrada y salida.
5. Carga inicial del sistema: Se tiene que tener en cuenta que en un inicio se tendrá que realizar la carga de datos de tablas auxiliares.

7. Diseño

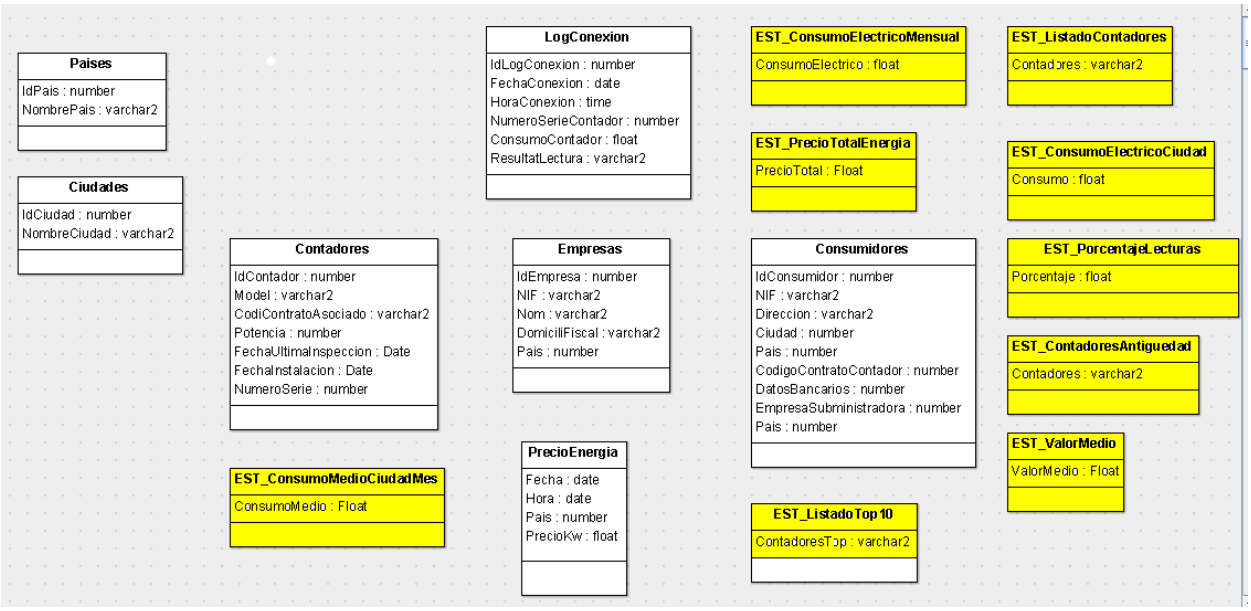
7.1. Diseño conceptual

En este apartado mostraré un diagrama UML como resultado de esta etapa. El diagrama nos muestra información de la base de datos sin

condicionarnos por el sistema tecnológico a utilizar. Este sistema después lo podemos migrar a otro sistema pero el que nos interesará será el relacional (SGBD), para su posterior implantación.

Para ello debemos identificar todas las entidades que el sistema necesita y definir sus relaciones.

Después para cada entidad definir sus atributos. Los objetos que están en color amarillo formarán parte del módulo de estadísticas.



7.2. Identificación de las entidades.

Identificaremos las entidades, entendidas como objetos del mundo real sobre el cual queremos almacenar cierta información, las clasificaré de la siguiente manera:

-Principales: son las de mayor importancia para el sistema. Contemplan la información básica del sistema y el resto de tipos de entidades se estructura a su alrededor.

-Intermedias: aparecerán como resultado de relaciones N-M entre entidades principales. Aquí obtendremos una nueva tabla para reflejar esta entidad intermedia y su clave primaria será de tipo compuesta y a partir de las claves primarias pertenecientes a las entidades principales que forman esta relación N-M. En algunos casos nos interesará que la clave primaria no sea la agrupación de las claves foráneas y definirnos una primaria de tipo secuencia.

-Auxiliares: son necesarias por los datos que iremos almacenando en ellos y su existencia responde a algún tipo de requerimiento del sistema.También incluiremos aquí la tabla de Logs.

Principales	Intermedias	Auxiliares
Contadores	LogConexion	EST_ConsumoElectricoMensual
Empresas	PrecioEnergia	EST_ListadoContadores
Consumidores		EST_ConsumoElectricoCiudad
		EST_PorcentajesLecturas
		EST_ContadoresAntiguedad
		EST_ValorMedio
		EST_ListadoTop10
		EST_PrecioTotalEnergia
		EST_ConsumoMedioCiudadMes
		LOGS

7.3. Identificación de las relaciones

Entidad Origen	Entidad Destino	Descripción
Contadores	LogConexion	Los contadores tiene asignado un log de conexión
Consumidores	Empresas	Los consumidores tienen asignado una empresas subministradora
Consumidores	Contadores	Los consumidores tienen asignado un contador
Empresas	Paises	Las empresas tienen asignado un país/es donde subministran
Consumidores	Paises	Los consumidores tienen asignado un país
Consumidores	Ciudad	Los consumidores tienen asignada una ciudad

7.4. Atributos de las entidades

A continuación se muestra una tabla con todas las entidades y para cada entidad los atributos que tiene. También se define que atributos formarán la primary key estos están definidos entre paréntesis después de la palabra PK

CONTADORES	PK(IdContador, NumeroSerie), Model, CodigoContratoAsociado, Potencia, FechaUltimaInspeccion, FechaInstalacion
EMPRESAS	PK(IdEmpresa, NIF), Nombre, DomicilioFiscal, Pais
CONSUMIDORES	PK(IdConsumidor, NIF), Direccion, Ciudad, CodigoContratoContador, DatosBancarios, EmpresaSubministradora, Pais
LOGCONEXION	PK(IdLogConexion, FechaConexion, HoraConexion), NumeroSerie, ConsumoContador, ResultadoContador
PRECIOENERGIA	PK(Fecha, Hora, Pais), PrecioKw
PAISES	PK(IdPais), NombrePais
CIUDADES	PK(IdCiudad), NombreCiudad
EST_ConsumoElectricoMensual	PK(NumeroSerie), Consumo
EST_ListadoContadores	PK(NombreCiudad, Mes), NumeroSerie,Model, CodigoContratoAsociado, Potencia, FechaUltimaInspeccion, FechaInstalacion
EST_ConsumoElectricoCiudad	PK(Mes, NombrePais), Consumo
EST_PorcentajeLecturas	PK(NombreEmpresa, Mes), Porcentaje
EST_ContadoresAntiguedad	PK(IdContador, NumeroSerie), Model, CodigoContratoAsociado, Potencia, FechaUltimaInspeccion, FechaInstalacion
EST_ValorMedio	PK(NombreCiudad, Any), ValorMedio
EST_ListadoTop10	PK(IdContador, NumeroSerie), Model, CodigoContratoAsociado, Potencia, FechaUltimaInspeccion, FechaInstalacion
EST_ConsumoMedioCiudadMes	PK (NombreCiudad, Mes), Consumo
EST_PrecioTotalEnergia	PK(IdContador, NumeroSerie), PrecioKw

7.5. Diseño lógico

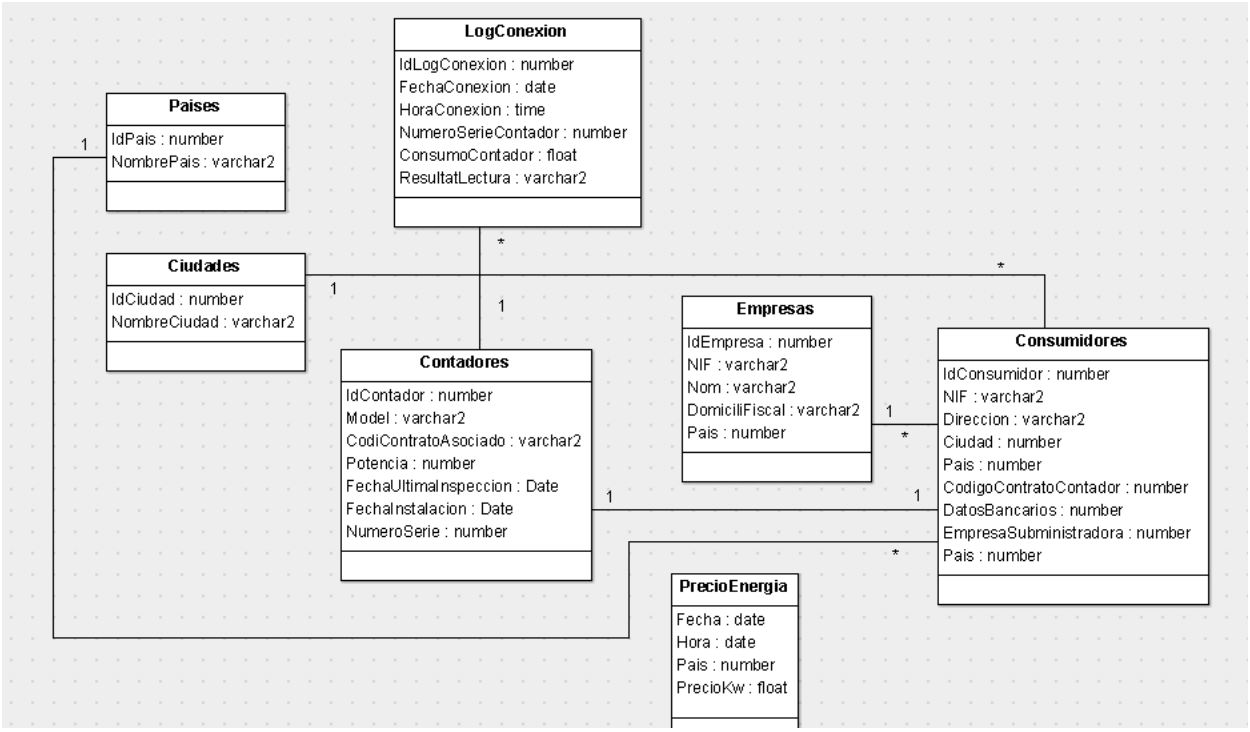
Una vez realizado el paso anterior se procede a transformarlo en el modelo lógico relacional, siguiendo las siguientes reglas:

- Las entidades generan relaciones.
- Las interrelaciones binarias 1:1 y 1:N originan claves foráneas.
- Las interrelaciones binarias M:N originan nuevas relaciones.

Las entidades pueden deben contener las siguientes características:

- Se debe indicar las siguientes restricciones:
  - PK: Primary Key, clave primaria.
  - FK: Foreign Key, clave foránea.
  - NOT NULL: no vacio.
  - CK: Check, chequea valores validos de un campo.
  - UK: Unique key, clave única.

A continuación se muestra el diagrama lógico relacional



LogConexion

```
CONSTRAINT IdLogConexion_pk PRIMARY KEY (IdLogConexion),
CONSTRAINT fkNumeroserie FOREIGN KEY (NumeroSerieContador)
   REFERENCES contadores (NumeroSerie)
```

Contadores

CONSTRAINT IdContador_pk PRIMARY KEY (IdContador),
CONSTRAINT fkCodigoContrato FOREIGN KEY (CodigoContratoAsociado)
REFERENCES consumidores (CodigoContratoContador)

Empresas

CONSTRAINT IdEmpresa_pk PRIMARY KEY (IdEmpresa, NIF),

Consumidores

CONSTRAINT IdConsumidor_pk PRIMARY KEY (IdConsumidor, NIF),
CONSTRAINT fkEmpresaSubministradora FOREIGN KEY (EmpresaSubministradora)
REFERENCES empresas (Nombre),
CONSTRAINT fkCodigoContrato FOREIGN KEY (CodigoContratoContador)
REFERENCES contadores (CodigoContratoAsociado),
CONSTRAINT fkCiudad FOREIGN KEY (Ciudad)
REFERENCES ciudades (IdCiudad),
CONSTRAINT fkPais FOREIGN KEY (Pais)
REFERENCES paises (IdPais),

Paises
CONSTRAINT IdPais_pk PRIMARY KEY (IdPais),

Ciudades
CONSTRAINT IdCiudad_pk PRIMARY KEY (IdCiudad),

PrecioEnergia
CONSTRAINT Idprecio_pk PRIMARY KEY (Fecha, Hora, Pais),

8. Implementación del proyecto

8.1. Instalación SGBD Oracle

Desde la página web http://www.postgresql.org/download/windows/ nos descargaremos el Sistema Gestor de Base de Datos (SGBSD) que vamos a utilizar para la implementación del proyecto. Una vez descargado el fichero obtendremos el árbol siguiente de directorios.

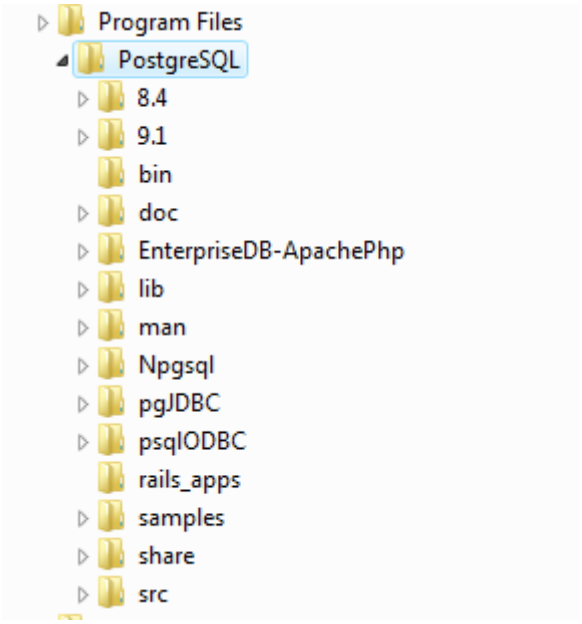


Figura 1. Árbol de directorios

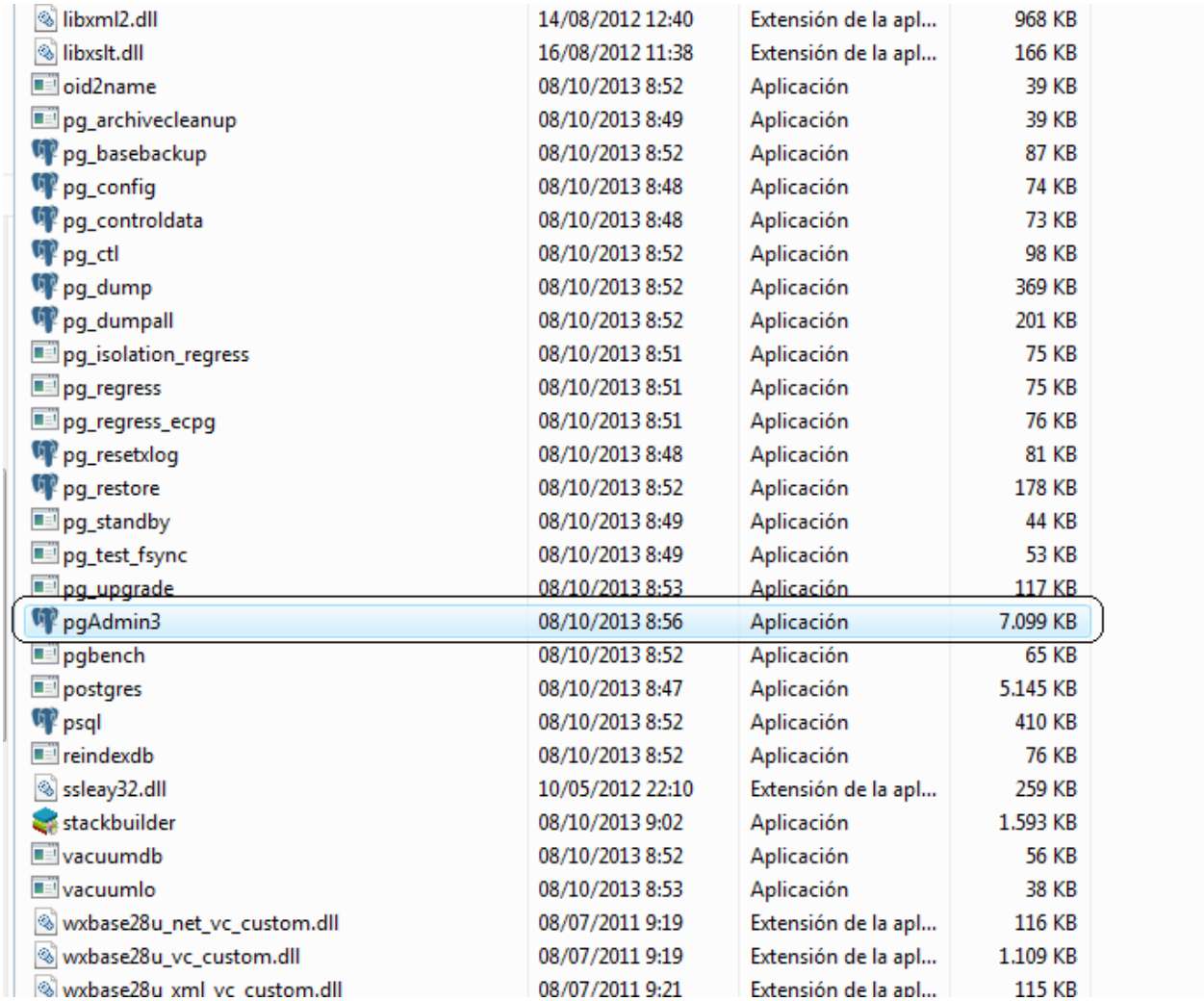


Figura 2. pgAdmin3

Mediante el archivo pgAdmin3 ejecutaremos el Sistema de Gestor de Base de Datos.

8.2. Configuración Oracle y herramientas

Una vez hemos ejecutado el Sistema de Gestor de Base de Datos mostraremos la siguiente pantalla:

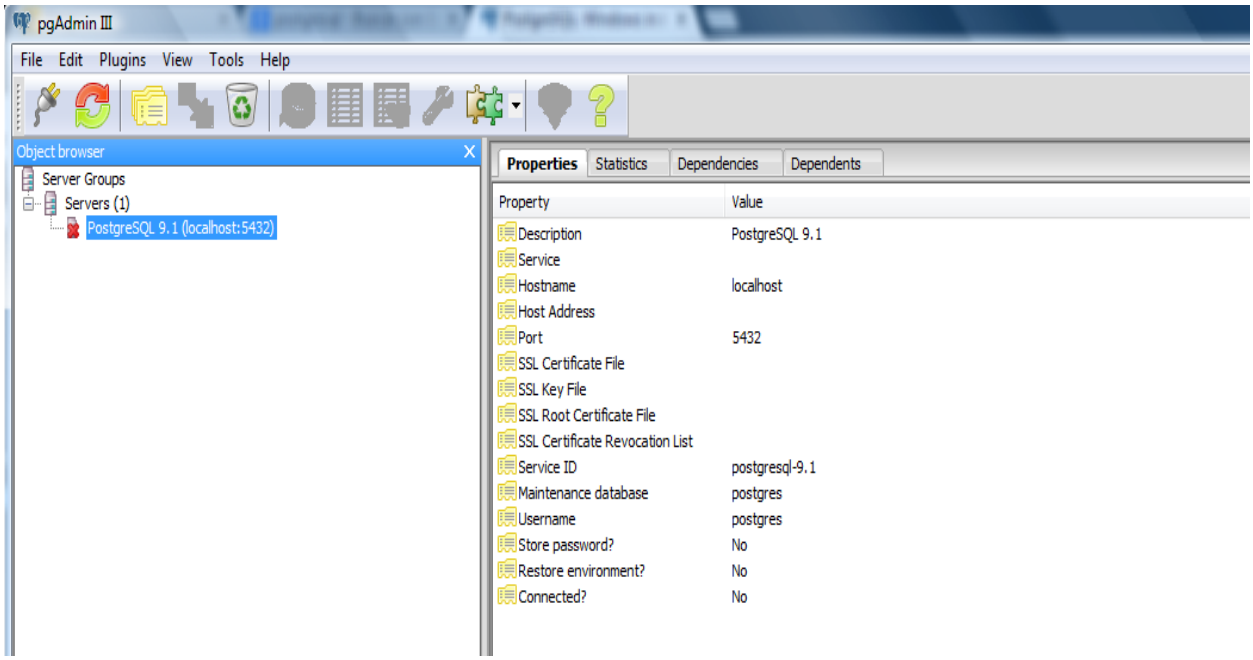


Figura 3. Pantalla principal pgAdminIII

Para conectarnos con el servidor deberemos introducir la clave que previamente en la instalación del SGBD hemos definido.

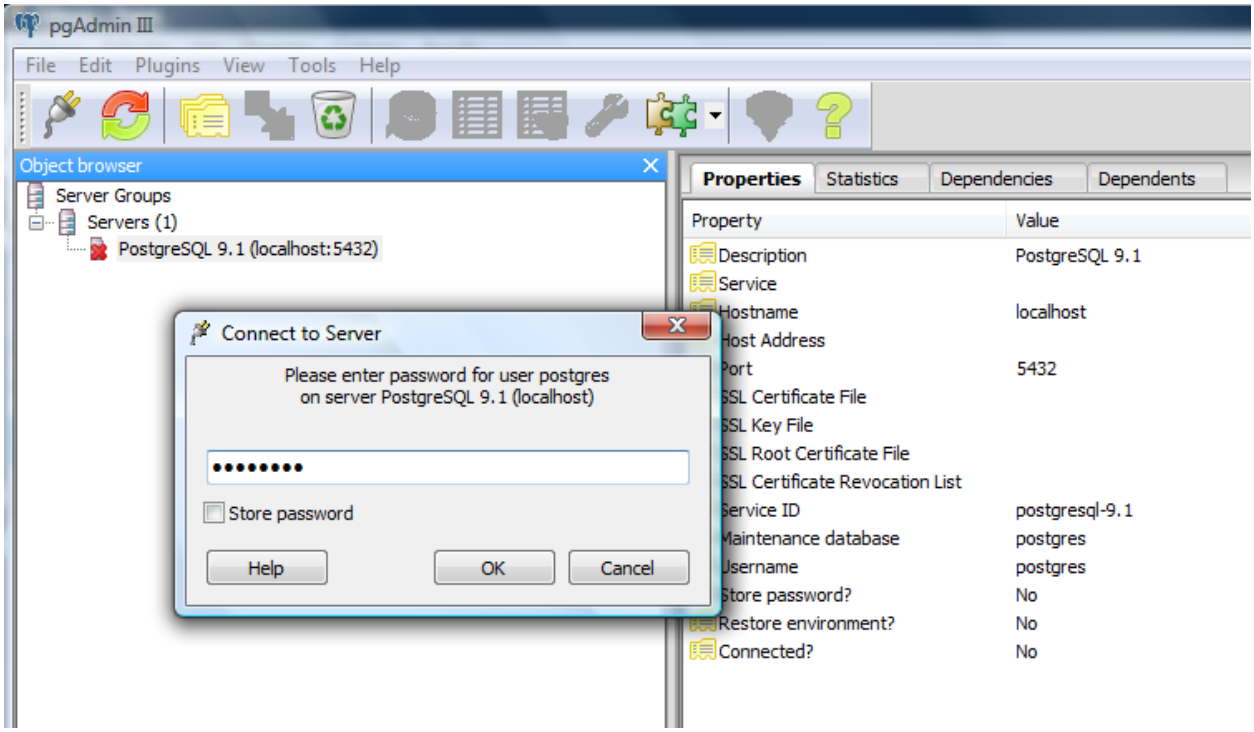


Figura 4. Conexión con el servidor

Una vez conectados se nos mostrará por defecto la base de datos “postgres” que nos facilita el aplicativo. En ella podremos definir los esquemas, funciones, tablas, índices, vistas, disparadores (triggers), tablespaces, procedures y roles que necesitemos para la implementación del proyecto. También podremos dar de alta nuestra bbdd.

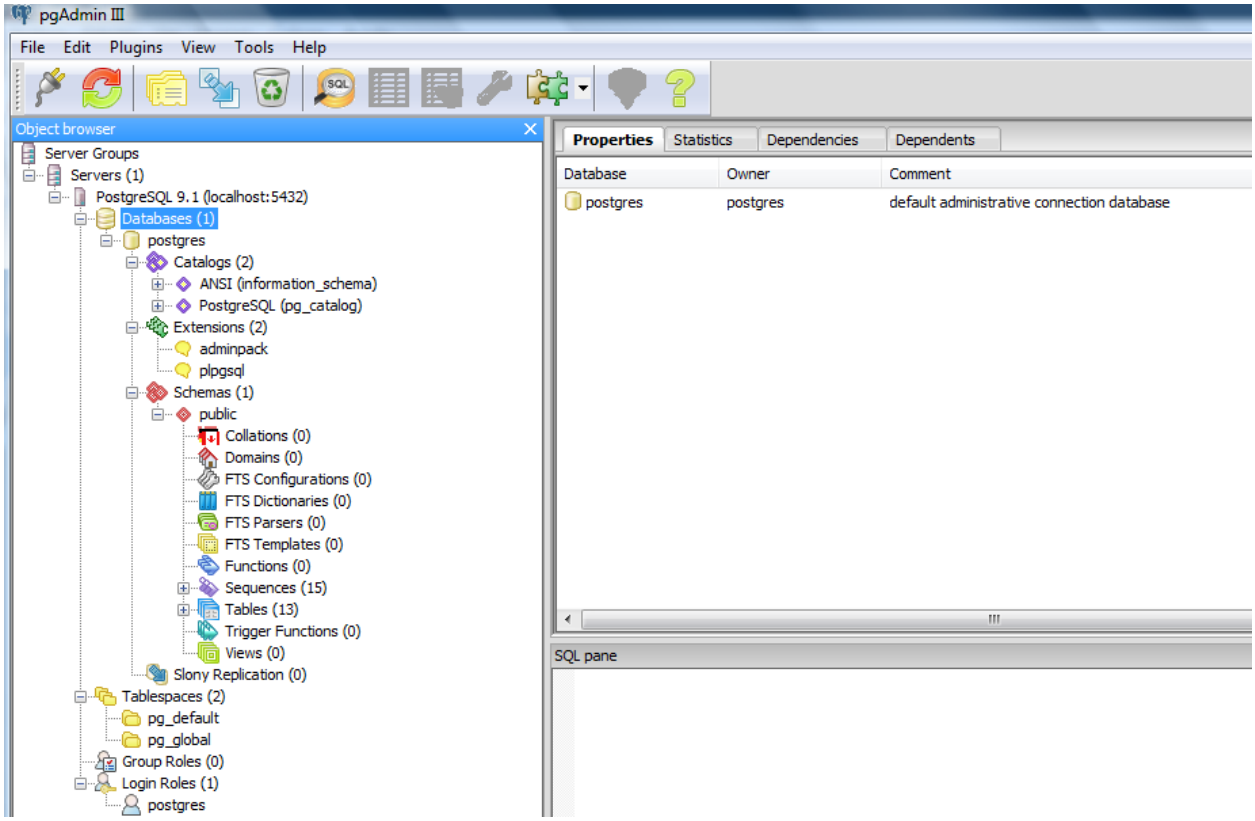


Figura 5. Database

Una vez hemos llegado a este punto ya seremos capaces de empezar a desarrollar toda la BBDD necesaria para cumplir con todos los requerimientos del TFG.

8.3. Implementación diseño E/R

Para realizar la implementación en el SGBD tendremos que crear los “tablespaces” donde almacenaremos por un lado un espacio de trabajo para los índices y otro espacio de trabajo para los datos (tablas, secuencias, triggers, funciones, …)

Para ello generaremos 2 “tablespaces”, uno para los datos y otro para los índices. En el siguiente script SQL se generarán los “tablespaces” correspondientes.

TABLESPACES
(TFGBBDD_tablespace.sql)

```
-- Tablespace: TFG_BBDD_IDX

-- DROP TABLESPACE "TFG_BBDD_IDX"

CREATE TABLESPACE "TFG_BBDD_IDX"
  OWNER postgres
  LOCATION 'C:/';
COMMENT ON TABLESPACE "TFG_BBDD_IDX"
  IS 'Tablespace índices';

-- Tablespace: TFG_BBDD_DATOS

-- DROP TABLESPACE "TFG_BBDD_DATOS"

CREATE TABLESPACE "TFG_BBDD_DATOS"
  OWNER postgres
  LOCATION 'd:/';
COMMENT ON TABLESPACE "TFG_BBDD_DATOS"
  IS 'Tablespace Datos';
```

8.3.1. Diseño tablas

Para la creación de todas las tablas utilizaremos el siguiente script SQL

TFGBBDD_Esquema.sql

```
-- Table: "Paises"

-- DROP TABLE "Paises";

CREATE TABLE "Paises"
(
  "IdPais" integer NOT NULL, -- Identificador del pais
  "NombrePais" character(50), -- Nombre del pais
  CONSTRAINT "PK_ID_PAIS" PRIMARY KEY ("IdPais" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX"
)
WITH (
  OIDS=FALSE
)
TABLESPACE "TFG_BBDD_DATOS";
ALTER TABLE "Paises"
  OWNER TO postgres;
COMMENT ON TABLE "Paises"
  IS 'Tabla de paises';
COMMENT ON COLUMN "Paises"."IdPais" IS 'Identificador del pais';
COMMENT ON COLUMN "Paises"."NombrePais" IS 'Nombre del pais';
```

```
-- Table: "Ciudades"

-- DROP TABLE "Ciudades";

CREATE TABLE "Ciudades"
(
  "IdCiudad" integer NOT NULL, -- Identificador de la ciudad
  "NombreCiudad" character(50), -- Nombre de la ciudad
  CONSTRAINT "PK_IdCiudad" PRIMARY KEY ("IdCiudad" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX"
)
WITH (
  OIDS=FALSE
)
TABLESPACE "TFG_BBDD_DATOS";
ALTER TABLE "Ciudades"
  OWNER TO postgres;
COMMENT ON TABLE "Ciudades"
  IS 'Tabla de ciudades';
COMMENT ON COLUMN "Ciudades"."IdCiudad" IS 'Identificador de la ciudad';
COMMENT ON COLUMN "Ciudades"."NombreCiudad" IS 'Nombre de la ciudad';
```

```
-- DROP TABLE "Contadores";

CREATE TABLE "Contadores"
(
  "IdContador" integer NOT NULL, -- Identificador del contador
  "Modelo" character(50), -- Modelo del contador
  "CodigoContratoAsociado" integer NOT NULL, -- Codigo del contrato asociado
  "Potencia" integer, -- Potencia en Kilowatios
  "FechaUltimaInspeccion" date NOT NULL, -- Fecha última inspección
  "FechaInstalacion" date NOT NULL, -- Fecha de instalación
  "NumeroSerie" integer NOT NULL, -- Número de Serie del contador
  CONSTRAINT "PK_IdContador" PRIMARY KEY ("IdContador" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX",
  CONSTRAINT "UN_CodigoContratoAsociado" UNIQUE ("CodigoContratoAsociado" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX",
  CONSTRAINT "UN_NumeroSerie" UNIQUE ("NumeroSerie" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX"
)
WITH (
  OIDS=FALSE
)
TABLESPACE "TFG_BBDD_DATOS";
ALTER TABLE "Contadores"
  OWNER TO postgres;
COMMENT ON TABLE "Contadores"
  IS 'Tabla de contadores';
COMMENT ON COLUMN "Contadores"."IdContador" IS 'Identificador del contador';
COMMENT ON COLUMN "Contadores"."Modelo" IS 'Modelo del contador';
COMMENT ON COLUMN "Contadores"."CodigoContratoAsociado" IS 'Codigo del contrato asociado';
COMMENT ON COLUMN "Contadores"."Potencia" IS 'Potencia en Kilowatios';
COMMENT ON COLUMN "Contadores"."FechaUltimaInspeccion" IS 'Fecha última inspección';
COMMENT ON COLUMN "Contadores"."FechaInstalacion" IS 'Fecha de instalación';
COMMENT ON COLUMN "Contadores"."NumeroSerie" IS 'Número de Serie del contador';
```

```
-- Table: "PrecioEnergia"

-- DROP TABLE "PrecioEnergia";

CREATE TABLE "PrecioEnergia"
(
  "Fecha" date NOT NULL, -- Fecha Precio Energia
  "Hora" time(6) with time zone NOT NULL, -- Hora Precio Energia
  "Pais" character(30) NOT NULL, -- Pais Precio Energia
  "PrecioKw" real, -- Precio en Kilowatios del precio de la energia
  CONSTRAINT "PK_PrecioEnergia" PRIMARY KEY ("Fecha" , "Hora" , "Pais" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX"
)
WITH (
  OIDS=FALSE
)
TABLESPACE "TFG_BBDD_DATOS";
ALTER TABLE "PrecioEnergia"
  OWNER TO postgres;
COMMENT ON TABLE "PrecioEnergia"
  IS 'Tabla Precios Energia';
COMMENT ON COLUMN "PrecioEnergia"."Fecha" IS 'Fecha Precio Energia';
COMMENT ON COLUMN "PrecioEnergia"."Hora" IS 'Hora Precio Energia';
COMMENT ON COLUMN "PrecioEnergia"."Pais" IS 'Pais Precio Energia';
COMMENT ON COLUMN "PrecioEnergia"."PrecioKw" IS 'Precio en Kilowatios del precio de la energia';
```

```
-- Table: "LogConexion"

-- DROP TABLE "LogConexion";

CREATE TABLE "LogConexion"
(
  "IdLogConexion" integer NOT NULL, -- Identificador del log de conexion
  "FechaConexion" date NOT NULL, -- Fecha de conexion del log
  "HoraConexion" time(6) with time zone NOT NULL, -- Hora del log de conexion
  "NumeroSerieContador" integer NOT NULL, -- Numero de Serie del contador
  "ConsumoContador" real, -- Consumo del contador
  "ResultatLectura" character(2) NOT NULL, -- Resultado de la lectura ('OK','KO')
  CONSTRAINT "PK_IdLogConexion" PRIMARY KEY ("IdLogConexion" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX",
  CONSTRAINT "FK_NumeroSerieContador" FOREIGN KEY ("NumeroSerieContador")
      REFERENCES "Contadores" ("NumeroSerie") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
)
WITH (
  OIDS=FALSE
)
TABLESPACE "TFG_BBDD_DATOS";
ALTER TABLE "LogConexion"
  OWNER TO postgres;
COMMENT ON TABLE "LogConexion"
  IS 'Tabla Log Conexion';
COMMENT ON COLUMN "LogConexion"."IdLogConexion" IS 'Identificador del log de conexion';
COMMENT ON COLUMN "LogConexion"."FechaConexion" IS 'Fecha de conexion del log';
COMMENT ON COLUMN "LogConexion"."HoraConexion" IS 'Hora del log de conexion';
COMMENT ON COLUMN "LogConexion"."NumeroSerieContador" IS 'Numero de Serie del contador';
COMMENT ON COLUMN "LogConexion"."ConsumoContador" IS 'Consumo del contador';
COMMENT ON COLUMN "LogConexion"."ResultatLectura" IS 'Resultado de la lectura (''OK'',''KO'')';
```

```
-- Table: "Empresas"

-- DROP TABLE "Empresas";

CREATE TABLE "Empresas"
(
  "IdEmpresa" integer NOT NULL, -- Identificador de la empresa
  "NIF" character(12) NOT NULL, -- Identificador fiscal de la empresa
  "Nombre" character(50), -- Nombre de la empresa
  "DomicilioFiscal" character(50), -- Domicilio fiscal de la empresa
  "Pais" character(50), -- Pais
  CONSTRAINT "PK_Empresa" PRIMARY KEY ("IdEmpresa" , "NIF" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX",
  CONSTRAINT "UN_IdEmpresa" UNIQUE ("IdEmpresa" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX"
)
WITH (
  OIDS=FALSE
)
TABLESPACE "TFG_BBDD_DATOS";
ALTER TABLE "Empresas"
  OWNER TO postgres;
COMMENT ON TABLE "Empresas"
  IS 'Tabla de empresas';
COMMENT ON COLUMN "Empresas"."IdEmpresa" IS 'Identificador de la empresa';
COMMENT ON COLUMN "Empresas"."NIF" IS 'Identificador fiscal de la empresa';
COMMENT ON COLUMN "Empresas"."Nombre" IS 'Nombre de la empresa';
COMMENT ON COLUMN "Empresas"."DomicilioFiscal" IS 'Domicilio fiscal de la empresa';
COMMENT ON COLUMN "Empresas"."Pais" IS 'Pais ';
```

```
-- Table: "Consumidores"

-- DROP TABLE "Consumidores";

CREATE TABLE "Consumidores"
(
  "IdConsumidor" integer NOT NULL, -- Identificador del consumidor
  "NIF" character(12) NOT NULL, -- Identificador del consumidor
  "Direccion" character(50), -- Dirección del consumidor
  "Ciudad" character(50), -- Ciudad del consumidor
  "Pais" character(50), -- Pais del consumidor
  "CodigoContratoContador" integer NOT NULL, -- Codigo del contrato del contador
  "DatosBancarios" integer, -- Datos bancarios del consumidor
  "CodigoEmpresaSubministradora" integer NOT NULL, -- Codigo de la empresa subministradora
  "CodigoPais" integer NOT NULL, -- Codigo del pais
  CONSTRAINT "PK_Consumidores" PRIMARY KEY ("IdConsumidor" )
  USING INDEX TABLESPACE "TFG_BBDD_IDX",
  CONSTRAINT "FK_Consumidores" FOREIGN KEY ("CodigoContratoContador")
      REFERENCES "Contadores" ("CodigoContratoAsociado") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT "FK_EmpresaSubministradora" FOREIGN KEY ("CodigoEmpresaSubministradora")
      REFERENCES "Empresas" ("IdEmpresa") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT "FK_Pais" FOREIGN KEY ("CodigoPais")
      REFERENCES "Paises" ("IdPais") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
)
WITH (
  OIDS=FALSE
)
```

```
TABLESPACE "TFG_BBDD_DATOS";
ALTER TABLE "Consumidores"
  OWNER TO postgres;
COMMENT ON TABLE "Consumidores"
  IS 'Tabla de consumidores';
COMMENT ON COLUMN "Consumidores"."IdConsumidor" IS 'Identificador del consumidor';
COMMENT ON COLUMN "Consumidores"."NIF" IS 'Identificador del consumidor';
COMMENT ON COLUMN "Consumidores"."Direccion" IS 'Dirección del consumidor';
COMMENT ON COLUMN "Consumidores"."Ciudad" IS 'Ciudad del consumidor';
COMMENT ON COLUMN "Consumidores"."Pais" IS 'Pais del consumidor';
COMMENT ON COLUMN "Consumidores"."CodigoContratoContador" IS 'Codigo del contrato del contador';
COMMENT ON COLUMN "Consumidores"."DatosBancarios" IS 'Datos bancarios del consumidor';
COMMENT ON COLUMN "Consumidores"."CodigoEmpresaSubministradora" IS 'Codigo de la empresa subministradora';
COMMENT ON COLUMN "Consumidores"."CodigoPais" IS 'Codigo del pais';
```

8.3.2. Diseño Constraint Primary Key / Foreign Key / UNIQUE

TABLA	NOMBRE ÍNDICE	DESCRIPCIÓN
PAISES	CONSTRAINT "PK_ID_PAIS" PRIMARY KEY ("IdPais")	El identificador del país ha de ser único. No puede existir varios países con el mismo identificador
CIUDADES	CONSTRAINT "PK_IdCiudad" PRIMARY KEY ("IdCiudad")	El identificador de la ciudad ha de ser único. No pueden existir varias ciudades con el mismo identificador
LOGCONEXION	CONSTRAINT "PK_IdLogConexion" PRIMARY KEY ("IdLogConexion")	
LOGCONEXION	CONSTRAINT "FK_NumeroSerieContador" FOREIGN KEY ("NumeroSerieContador") REFERENCES "Contadores" ("NumeroSerie")	
CONTADORES	CONSTRAINT "PK_IdContador" PRIMARY KEY ("IdContador")	
CONTADORES	CONSTRAINT "UN_CodigoContratoAsociado" UNIQUE ("CodigoContratoAsociado")	
CONTADORES	CONSTRAINT "UN_NumeroSerie" UNIQUE ("NumeroSerie")	
PRECIOENERGIA	CONSTRAINT "PK_PrecioEnergia" PRIMARY KEY ("Fecha" , "Hora" , "Pais")	
EMPRESAS	CONSTRAINT "PK_Empresa" PRIMARY KEY ("IdEmpresa" , "NIF")	
EMPRESAS	CONSTRAINT "UN_IdEmpresa" UNIQUE ("IdEmpresa")	
CONSUMIDORES	CONSTRAINT "FK_Consumidores" FOREIGN KEY ("CodigoContratoContador") REFERENCES "Contadores" ("CodigoContratoAsociado")	
CONSUMIDORES	CONSTRAINT "FK_EmpresaSubministradora" FOREIGN KEY ("CodigoEmpresaSubministradora") REFERENCES "Empresas" ("IdEmpresa")	

CONSUMIDORES	CONSTRAINT "FK_Pais" FOREIGN KEY ("CodigoPais") REFERENCES "Paises" ("IdPais")	

8.4. Implementación de funcionalidades

8.4.1. Implementación procedimientos ABM

Todos los procedimientos estarán dentro de la carpeta asociada. Tendremos dos carpetas: la de los procedimientos de ABM que se llamara “procedimientosABM” y la de consultas y estadísticas que se llamara “procedimientos_consulta”. Cada procedimiento estará asociado a un fichero sql.

Procedimientos ABM

Alta país: p_alta_pais
Da de alta un país. En caso de que el país ya existiera mostraríamos un error “País existente”. En caso de que el país no exista se verificarán todos los campos que no admitan nulos
Parámetros de entrada
• IdPais: integer, Identificador del país (Deberemos crear previamente su secuencia correspondiente) • NombrePaís: character de 50 posiciones, Nombre del país
Parámetros de salida
• Rsp: character
Pre-condición
Todos los campos character o integer deben de tener menos del tamaño máximo definido en la base de datos
Post-condición
Se ha generado un nuevo país en la tabla de paises
Retorna
• “OK”: el procedimiento ha sido un éxito. • “ERROR: PAIS YA EXISTE“: Dicho país ya está dado de alta • “ERROR: [SQLERRM]”: El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Alta ciudad: p_alta_ciudad
Da de alta una ciudad. En caso de que la ciudad ya existiera mostraríamos un error “Ciudad existente”. En caso de que la ciudad no exista se verificarán todos los campos que no admitan nulos
Parámetros de entrada
• IdCiudad: integer, Identificador de la ciudad (Deberemos crear previamente su secuencia correspondiente) • NombreCiudad: character de 50 posiciones, Nombre de la ciudad
Parámetros de salida
• Rsp: character
Pre-condición
Todos los campos character o integer deben de tener menos del tamaño máximo definido en la base de datos
Post-condición
Se ha generado una nueva ciudad en la tabla de ciudades
Retorna
• “OK”: el procedimiento ha sido un éxito. • “ERROR: CIUDAD YA EXISTE“: Dicha ciudad ya está dada de alta

- "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Alta PrecioEnergía: p_alta_precioEnergia
Da de alta todos los precios de la energía. En caso de que algún precio ya existiera mostraríamos un error "Error en el precio". Recordemos que los precios pueden variar a cualquier fecha, hora y país por lo que no se puede dar de alta un precio el mismo día, a la misma hora y en el mismo país. En caso de que el precio no exista se verificarán todos los campos que no admitan nulos
Parámetros de entrada
• Fecha: Date, Fecha en que se marca el precio • Hora: Time, Hora en que se marca el precio • País: character de 50 posiciones, País en el cual se marca el precio • PrecioKw: real, Precio de la energía en Kilowatios
Parámetros de salida
• Rsp: character
Pre-condición
Todos los campos alfanuméricos deben de tener menos del tamaño máximo definido en la base de datos. Los campos de tipo fecha/hora deben de tener el formato correcto a lo definido en la base de datos. El precio en kilowatios no puede ser negativo.
Post-condición
Se ha generado un nuevo precio en la tabla de precios de la energía
Retorna
• "OK": el procedimiento ha sido un éxito. • "ERROR: PRECIO YA EXISTE": El precio ya existe • "ERROR: FECHA PRECIO INCORRECTA": La fecha del precio tiene un formato incorrecto • "ERROR: HORA PRECIO ERRÓNEA": La hora del precio tiene un formato incorrecto • "ERROR: PAIS INEXISTENTE": El país no existe en la tabla de países • "ERROR: PRECIO INCORRECTO": El precio tiene un valor incorrecto (Negativo,..) • "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Alta Empresa: p_alta_empresa
Da de alta una empresa. En caso de que la empresa ya existiera mostraríamos un error "Empresa existente". En caso de que la empresa no exista se verificarán todos los campos que no admitan nulos
Parámetros de entrada
• IdEmpresa: integer, Identificador de la empresa (Deberemos crear previamente una secuencia para las empresas) • NIF: character de 12 posiciones, Identificador fiscal de la empresa • Nombre: character de 50 posiciones, Nombre de la empresa • DomicilioFiscal: character de 50 posiciones, Domicilio fiscal de la empresa • País: character de 50 posiciones
Parámetros de salida
• Rsp: character
Pre-condición
-Todos los campos deben de tener menos del tamaño máximo definido en la base de datos
Post-condición
Se ha generado una nueva empresa en la tabla de empresas
Retorna
• "OK": el procedimiento ha sido un éxito. • "ERROR: EMPRESA YA EXISTE": Dicha empresa ya está dada de alta • "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Alta Contador: p_alta_contador
Da de alta un contador. En caso de que el contador ya existiera mostraríamos un error “Contador existente”. En caso de que el contador no exista se verificarán todos los campos que no admitan nulos
Parámetros de entrada
• IdContador: integer, Identificador del contador (Deberemos crear previamente una secuencia para los contadores) • Modelo: character, modelo del contador • CodigoContratoAsociado: integer, Código del contrato asociado al contador • Potencia: integer, potencia en kilowatios del contador • FechaUltimaInspeccion: Date, fecha de la última inspección del contador • FechaInstalación: Date, fecha de la instalación del contador • NumeroSerie: integer, Número de serie del contador
Parámetros de salida
• Rsp: character
Pre-condición
-Todos los campos deben de tener menos del tamaño máximo definido en la base de datos -El código de contrato asociado debe existir en la tabla de CONSUMIDORES
Post-condición
Se ha generado un nuevo contador en la tabla de contadores
Retorna
• “OK”: el procedimiento ha sido un éxito. • “ERROR: CONTADOR YA EXISTE“: El contador ya está dado de alta • “ERROR: CODIGO DE CONTRATO INEXISTENTE EN LA TABLA DE CONSUMIDORES“: El código de contrato asociado del contador debe existir en la tabla de consumidores • “ERROR: POTENCIA EN KW ERRÓNEA“: La potencia en kilowatios no puede contener un valor negativo • “ERROR: FECHA ULTIMA INSPECCION ERRÓNEA”: La fecha última de inspección del contador debe tener un formato correcto • “ERROR: FECHA INSTALACIÓN ERRÓNEA”: La fecha de instalación del contador debe tener un formato correcto • “ERROR: [SQLERRM]”: El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Alta LogConexion: p_alta_logConexion
Da de alta un log de conexión. En caso de que el log de conexión ya existiera mostraríamos un error “Log de conexión existente”. En caso de que el log de conexión no exista se verificarán todos los campos que no admitan nulos
Parámetros de entrada
• IdlogConexion: integer, Identificador del log de conexión (Deberemos crear previamente una secuencia para los logs de conexión) • FechaConexion: Date, fecha de conexión • HoraConexion: Time, hora de conexión • NumeroSerieContador: integer, Número de serie del contador • ConsumoContador: integer, Consumo del contador • ResultadoLectura: character, Resultado de la lectura (‘OK’,’KO’)
Parámetros de salida
• Rsp: character
Pre-condición
-Todos los campos deben de tener menos del tamaño máximo definido en la base de datos -Todos los campos de tipo fecha/hora debe tener un formato correcto predefinido por la base de datos -El número de serie del contador debe existir en la tabla de CONTADORES
Post-condición
Se ha generado un nuevo log de conexión en la tabla de log de conexiones
Retorna
• “OK”: el procedimiento ha sido un éxito. • “ERROR: LOG DE CONEXIÓN YA EXISTE“: El log de conexión ya existe en la tabla de log de conexiones • “ERROR: NÚMERO DE SERIE DEL CONTADOR INEXISTENTE EN LA TABLA DE CONTADORES“: El número de serie del contador ha de existir en la tabla de contadores

- "ERROR: FECHA CONEXIÓN ERRÓNEA": La fecha de conexión debe tener un formato correcto
- "ERROR: HORA CONEXIÓN ERRÓNEA": La hora de conexión debe tener un formato correcto
- "ERROR: CONSUMO DEL CONTADOR ERRÓNEO": El consumo del contador debe tener un valor positivo
- "ERROR: RESULTADO LECTURA": Resultado de la lectura erróneo. El resultado de la lectura sólo puede devolver dos resultados: Correcto (OK); Incorrecto (KO)
- "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Alta consumidor: p_alta_consumidor
Da de alta un consumidor. En caso de que el consumidor ya existiera mostraríamos un error "Consumidor existente". En caso de que el consumidor no exista se verificarán todos los campos que no admitan nulos
Parámetros de entrada
• IdConsumidor: integer, Identificador del consumidor (Deberemos crear previamente su secuencia correspondiente) • NIF: character de 12 posiciones, Identificador fiscal del consumidor • Direccion: character de 50 posiciones, Dirección del consumidor • Ciudad: character de 50 posiciones, Ciudad del consumidor • CodigoContratoContador: integer, Código del contrato asociado al contador • DatosBancarios: character, Datos bancarios del consumidor • CodigoEmpresaSubministradora: integer, Código de la empresa subministradora • CodigoPais: integer, Código de país
Parámetros de salida
• Rsp: character
Pre-condición
-Todos los campos deben de tener menos del tamaño máximo definido en la base de datos -El código del contrato asociado al contador debe existir en la tabla de CONTADORES -El código la empresa subministradora debe existir en la tabla de EMPRESAS -El código de país debe existir en la tabla de PAISES
Post-condición
Se ha generado un nuevo consumidor en la tabla de consumidores
Retorna
• "OK": el procedimiento ha sido un éxito. • "ERROR: CONSUMIDOR YA EXISTE": El consumidor ya está dado de alta • "ERROR: CODIGO CONTRATO CONTADOR INEXISTENTE EN TABLA CONTADORES": El código del contrato asociado al contador debe existir en la tabla de CONTADORES • "ERROR: CODIGO EMPRESA SUBMINISTRADOR INEXISTENTE EN TABLA EMPRESAS": El código la empresa subministradora debe existir en la tabla de EMPRESAS • "ERROR: CODIGO PAIS INEXISTENTE EN TABLA PAISES": El código de país debe existir en la tabla de PAISES • "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

8.4.2. Implementación procedimientos de consulta y estadísticas

Procedimientos de consultas / Estadísticas

Procedimiento 1: p_Consumo_Electrico_Mensual
El consumo eléctrico mensual de cada contador medido en KW/h.
Parámetros de entrada
• No tiene parámetros de entrada
Parámetros de salida

• Rsp: character • Tabla Rsp: TB_CONSUMO_ELECTRICO_MENSUAL
Procedimiento asociado
• Procedimiento asociado p_Consumo_Electrico_Mensual
Post-condición
Se devuelve una tabla y el resultado del procedimiento
Retorna
• "OK": el procedimiento ha sido un éxito. • "ERROR: SIN DATOS": El procedimiento no ha devuelto ningún dato. • "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Procedimiento 2: p_List_Cont_Cons_Mensual
Dada una ciudad y un mes cómo parámetros, el listado de todos los contadores donde el consumo mensual ha superado el 80% del consumo medio de todos los contadores de la ciudad en ese mismo período de tiempo. Todo esto ordenado de forma ascendente por el tanto por ciento de consumo eléctrico consumido
Parámetros de entrada
• pCiudad: character • pMes: character
Parámetros de salida
• Rsp: character • Tabla Rsp: TB_LIST_CONT_CONS_MENSUAL
Procedimiento asociado
• Procedimiento asociado p_List_Cont_Cons_Mensual
Post-condición
Se devuelve una tabla y el resultado del procedimiento
Retorna
• "OK": el procedimiento ha sido un éxito. • "ERROR: SIN DATOS": El procedimiento no ha devuelto ningún dato. • "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Procedimiento 3: p_Cons_Elect_Ciudad
Dado un mes y un país concreto, consumo eléctrico de cada ciudad como suma de todos los contadores instalados.
Parámetros de entrada
• pMes: character • pPais: character
Parámetros de salida
• Rsp: character • Tabla Rsp: TB_CONS_ELECT_CIUDAD
Procedimiento asociado
• Procedimiento asociado p_Cons_Elect_Ciudad
Post-condición
Se devuelve una tabla y el resultado del procedimiento
Retorna
• "OK": el procedimiento ha sido un éxito. • "ERROR: SIN DATOS": El procedimiento no ha devuelto ningún dato. • "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Procedimiento 4: p_Porcentaje_Lecturas

Dada una empresa de suministros y un mes concreto, porcentaje de lecturas de contadores efectuadas de forma correcta. Se consideran como tal las que la conexión se ha efectuado de manera correcta y se ha podido leer la lectura.

Parámetros de entrada

- pEmpresa: character
- pMes: carácter

Parámetros de salida

- Rsp: character
- Tabla Rsp: TB_PORCENTAJE_LECTURAS

Procedimiento asociado

- Procedimiento asociado p_Porcentaje_Lecturas

Post-condición

Se devuelve una tabla y el resultado del procedimiento

Retorna

- "OK": el procedimiento ha sido un éxito.
- "ERROR: SIN DATOS": El procedimiento no ha devuelto ningún dato.
- "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Procedimiento 5: p_List_Cont_Antig

Listado de contadores que tengan un determinado numero de años de antigüedad.

Parámetros de entrada

- pAntiguedad: integer (Número de años de antigüedad)

Parámetros de salida

- Rsp: character
- Tabla Rsp: TB_LIST_CONT_ANTIG

Procedimiento asociado

- Procedimiento asociado p_List_Cont_Antig

Post-condición

Se devuelve una tabla y el resultado del procedimiento

Retorna

- "OK": el procedimiento ha sido un éxito.
- "ERROR: SIN DATOS": El procedimiento no ha devuelto ningún dato.
- "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Procedimiento 6: p_Valor_Medio

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

Parámetros de entrada

- pCiudad: character
- pAny: integer

Parámetros de salida

- Rsp: character
- Tabla Rsp: TB_VALOR_MEDIO

Procedimiento asociado

- Procedimiento asociado p_Valor_Medio

Post-condición

Se devuelve una tabla y el resultado del procedimiento
Retorna
• “OK”: el procedimiento ha sido un éxito. • “ERROR: SIN DATOS”: El procedimiento no ha devuelto ningún dato. • “ERROR: [SQLERRM]”: El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Procedimiento 7: p_Precio_Total_Ener
Para cada contador instalado, precio total de la energía consumida mensualmente.
Parámetros de entrada
• No tiene parámetros de entrada
Parámetros de salida
• Rsp: character • Tabla Rsp: TB_PRECIO_TOTAL_ENER
Procedimiento asociado
• Procedimiento asociado p_Precio_Total_Ener
Post-condición
Se devuelve una tabla y el resultado del procedimiento
Retorna
• “OK”: el procedimiento ha sido un éxito. • “ERROR: SIN DATOS”: El procedimiento no ha devuelto ningún dato. • “ERROR: [SQLERRM]”: El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Procedimiento 8: p_Top10_Contadores
Top-10 de contadores que históricamente han tenido más consumo a cada una de les ciudades
Parámetros de entrada
• pCiudad: character
Parámetros de salida
• Rsp: character • Tabla Rsp: TB_TOP_CONTADORES
Procedimiento asociado
• Procedimiento asociado p_Top10_Contadores
Post-condición
Se devuelve una tabla y el resultado del procedimiento
Retorna
• “OK”: el procedimiento ha sido un éxito. • “ERROR: SIN DATOS”: El procedimiento no ha devuelto ningún dato. • “ERROR: [SQLERRM]”: El sqlerrm tendrá el mensaje del error que no ha sido tratado.

Procedimiento 9: p_Consumo_Medio
Consumo medio de todos los consumidores para cada ciudad y mes.
Parámetros de entrada
• pCiudad: character • pMes: character
Parámetros de salida
• Rsp: character • Tabla Rsp: TB_CONSUMO_MEDIO
Procedimiento asociado
• Procedimiento asociado p_Consumo_Medio
Post-condición
Se devuelve una tabla y el resultado del procedimiento
Retorna
• "OK": el procedimiento ha sido un éxito. • "ERROR: SIN DATOS": El procedimiento no ha devuelto ningún dato. • "ERROR: [SQLERRM]": El sqlerrm tendrá el mensaje del error que no ha sido tratado.

8.5. Testing de pruebas

8.5.1. Diseño del testing de pruebas

Para realizar el juego de pruebas necesitamos la carga inicial de las tablas. Lo efectuaremos mediante el script **Inicializacion.sql.** Previamente a esta carga inicial de tablas deberemos de ejecutar los siguientes scripts:

TFG_BBDD_TableSpace.sql (Crea los espacios de trabajo ("tablespace") necesarios para los datos y para los índices
TFG_BBDD_Esquema.sql (Crea toda la estructura necesaria para el diseño de la base de datos (Tablas, Secuencias,..)

8.5.2. Script del testing de pruebas

• *Juego de pruebas de ABM de Países.* El fichero se llama "**p_alta_pais.sql**". Este juego de pruebas realizará pruebas de Alta, Baja y modificación para el control de la tabla de Países.

• *Juego de pruebas de ABM de Ciudades.* El fichero se llama "**p_alta_ciudad**". Este juego de pruebas realizará pruebas de Alta, Baja y modificación para el control de la tabla de Ciudades.

• *Juego de pruebas de ABM de Contadores.* El fichero se llama "**p_alta_contador.sql**". Este juego de pruebas realizará pruebas de Alta, Baja y modificación para el control de la tabla de Contadores.

• *Juego de pruebas de ABM de Log de Conexiones.* El fichero se llama ”**p_alta_logConexion.sql**”. Este juego de pruebas realizará pruebas de Alta, Baja y modificación para el control de la tabla de Log de Conexiones.

• *Juego de pruebas de ABM de Consumidores.* El fichero se llama “**p_alta_consumidor.sql**”. Este juego de pruebas realizará pruebas de Alta, Baja y modificación para el control de la tabla de Consumidores.

• *Juego de pruebas de ABM de Empresas.* El fichero se llama ” **p_alta_empresa.sql**”. Este juego de pruebas realizará pruebas de Alta, Baja y modificación para el control de la tabla de Empresas.

• *Juego de pruebas de ABM de Precio Energía.* El fichero se llama **p_alta_precioEnergia.sql**”. Este juego de pruebas realizará pruebas de Alta, Baja y modificación para el control de la tabla de Precio de la Energía.

8.5.3. Ejecución y depuración del testing de pruebas

Se ha generado un script para cada tipo de procedimiento de consulta/Estadísticas, estos scripts serán los siguientes:

p_Consumo_Electrico_Mensual.sql
p_List_Cont_Cons_Mensual.sql
p_Cons_Elect_Ciudad.sql
p_Porcentaje_Lecturas.sql
p_List_Cont_Antig.sql
p_Valor_Medio.sql
p_Precio_Total_Ener.sql
p_Top10_Contadores.sql
p_Consumo_Medio.sql

8.6. Valoración económica y Recurso

A continuación se mostrará una tabla con todas las tareas del proyecto y que recursos humanos se utilizarán y el número de horas que realizarán para cada tarea según la planificación antes descrita.

TAREA	RECURSOS	Nº HORAS
01 - PAC1 - Planificación del TFC		**64**
1.1 - Estudio y lectura del enunciado	Jefe de Proyecto	8
1.2 - Descripción y diseño de las tareas a realizar (metodología en cascada)	Jefe de Proyecto	16
1.3 - Diseño del diagrama de Gantt	Jefe de Proyecto	16
1.4 - Materiales técnicos y recursos	Jefe de Proyecto	8

1.5 - Plan de Contingencias	Jefe de Proyecto	8
1.6 - Elaboración y Entrega documentación PAC1	Jefe de Proyecto	8
2 - PAC2 - Análisis de requisitos		**392**
2.1 - Análisis de requisitos	Analista	96
2.1.1 - Requisitos Funcionales	Analista	80
2.1.2 - Requisitos No funcionales	Analista	16
2.2 - Diseño	Analista	96
2.2.1 - Diseño Conceptual	Analista	32
2.2.2 - Diseño Lógico	Analista	32
2.2.3 - Diseño Físico	Analista	32
2.3 - Elaboración y Entrega documentación PAC2	Analista	8
03 - PAC3 – Implementación		**344**
3.1 - Instalación SGBD Oracle	Administrador BD	8
3.2 - Configuración Oracle y Herramientas	Administrador BD	8
3.3 - Implementación Diseño E/R	Analista	40
3.3.1 - Diseño Tablas	Programador	24
3.3.2 - Diseño Índices y disparadores	Programador	16
3.4 - Implementación de funcionalidades	Programador	64
3.4.1 - Implementación procedimientos ABM	Programador	64
3.5 - Testing de pruebas	Programador	56
3.5.1 - Diseño del testing de pruebas	Analista	16
3.5.2 - Script del testing de pruebas	Programador	16
3.5.3 - Ejecución del testing de pruebas	Analista	16
3.5.4 - Depuración del testing de pruebas	Programador	8
3.6 - Elaboración y Entrega documentación PAC3	Analista	8
04 - Producto, Memoria y Presentación		**264**
4.1 - Redacción de la memória	Jefe del Proyecto	120
4.2 - Redacción de la Presentación	Jefe del Proyecto	80
4.3 - Redacción del trabajo práctico	Jefe del Proyecto	48
4.4.- Auto-Informe	Jefe del Proyecto	8
4.4 - Entrega Final del TFC	Jefe del Proyecto	8
TOTAL HORAS PROYECTO		**1.064**

Para realizar el cálculo de los recursos humanos dispongo de la siguiente tabla de tarificación

TARIFA DE PRECIOS	
Recurso	**Precio por Hora**
Jefe de Proyecto	50 €
Analista	45 €
Administrador BD	45 €
Programador	30 €

A continuación se mostrará una tabla de tareas agrupadas por recursos humanos y el coste total que supone dicho proyecto.

JEFE DE PROYECTO		
TAREA	**Nº HORAS**	**COSTE**
1.1 - Estudio y lectura del enunciado	8	400
1.2 - Descripción y diseño de las tareas a realizar (metodología en cascada)	16	800
1.3 - Diseño del diagrama de Gantt	16	800
1.4 - Materiales técnicos y recursos	8	400
1.5 - Plan de Contingencias	8	400
1.6 - Elaboración y Entrega documentación PAC1	8	400
4.1 - Redacción de la memória	120	6000
4.2 - Redacción de la Presentación	80	4000
4.3 - Redacción del trabajo práctico	48	2400
4.4.- Auto-Informe	8	400
4.4 - Entrega Final del TFC	8	400
TOTAL JEFE DE PROYECTO	**328**	**16.400**
ANALISTA		
2.1 - Análisis de requisitos	96	4.320
2.1.1 - Requisitos Funcionales	80	3.600
2.1.2 - Requisitos No funcionales	16	720
2.2 - Diseño	96	4.320
2.2.1 - Diseño Conceptual	32	1.440
2.2.2 - Diseño Lógico	32	1.440
2.2.3 - Diseño Físico	32	1.440
2.3 - Elaboración y Entrega documentación PAC2	8	360
3.3 - Implementación Diseño E/R	40	1.800
3.5.1 - Diseño del testing de pruebas	16	720
3.5.3 - Ejecución del testing de pruebas	16	720
3.6 - Elaboración y Entrega documentación PAC3	8	360
TOTAL ANALISTA	**472**	**21.240**
ADMINISTRADOR BBDD		
03.01 - Instalación SGBD Oracle	8	360
03.02 - Configuración Oracle y Herramientas	8	360
TOTAL ADMINISTRADOR BD	**16**	**720**
PROGRAMADOR		
3.3.1 - Diseño Tablas	24	720
3.3.2 - Diseño Índices y disparadores	16	480

3.4 - Implementación de funcionalidades	64	1.920
3.4.1 - Implementación procedimientos ABM	64	1.920
3.5 - Testing de pruebas	56	1.680
3.5.2 - Script del testing de pruebas	16	480
3.5.4 - Depuración del testing de pruebas	8	240
TOTAL PROGRAMADOR	**248**	**7.440**
TOTAL PROYECTO sin IVA	**1.064**	**45.800**
IVA 21%		**9.618**
TOTAL PROYECTO		**55.418**

8.7. Glosario

- **ABM**: Abreviatura de Alta, Baja y Modificación.
- **SGBD**: Abreviatura de Sistema gestor de Base de Datos.
- **UNIQUE:**
- **Entidad:** Representación de un objeto o concepto del mundo real dentro de un base de datos.
- **Esquema E/R:** Es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información, así como sus interrelaciones y propiedades.
- **Trigger:** Procedimiento que se ejecuta cuando se cumplen una serie de condiciones en la tabla con la que está relacionado.
- **FOREIGN KEY:** El valor o los valores de un registro que hacen referencia a la clave primaria de otra tabla o entidad.
- **PRIMARY KEY:** El valor o los valores de un registro que lo identifican de forma unívoca en una tabla de la BD.
- **Base de datos relacional**: Conjunto de datos organizados y relacionados entre sí.
- **BD**: Abreviatura de Base de Datos
- **Oracle:** Sistema de gestión de Bases de Datos utilizado en el desarrollo de este proyecto.
- **P_:** Prefijo que precede al nombre dado a los procedimientos almacenados.

8.8. Bibliografía

- Documentación asignatura Sistemas de Gestión de Base de Datos (SGBD).
- Documentación asignatura Bases de datos I.
- Documentación asignatura Bases de datos II
- Manual SQL PostGresSQL (PgAdmin III)
- http://www.google.es
- http://www.postgresql.org/docs

8.9. Anexos

En el fichero zip se encontrara los siguientes ficheros:

Dentro de la carpeta SCRIPTS SQL existen 2 carpetas:

INICIALIZACION BDD: Script inicialización.sql (Realiza una carga inicial de datos sobre toda la base de datos.

PROCEDIMIENTOS: Dentro existen 2 subcarpetas:

PROCEDIMIENTOS ABM: Están todos los scripts sql correspondientes a los procedimientos ABM

PROCEDIMIENTOS CONSULTA-ESTADÍSTICA: Scripts sql correspondientes a los procedimientos de consulta y estadística.

TFG_BBDD_Esquema.sql → Genera toda la estructura de la BD

TFG_BBDD_Secuencias.sql → Genera todas las secuencias necesarias
TFG_BBDD_tablespace.sql → Genera los dos espacios de trabajo (Datos e índices)