ESCUELA TÉCNICA SUPERIOR EN INGENIERÍA INFORMÁTICA

INGENIERÍA INFORMÁTICA

CURSO 2009/2010

Proyecto de Fin de Carrera

Una propuesta para el desarrollo de Almacenes de Datos XML

Autor: Emilio José Sánchez Gutiérrez

Tutora: Belén Vela Sánchez

RESUMEN

En este Proyecto Fin de Carrera se ha realizado una propuesta metodológica para el desarrollo de Almacenes de Datos XML y posteriormente se ha validado la misma mediante el desarrollo de un caso de estudio.

Para tal propósito se han realizado una serie de estudios previos, como sobre los Almacenes de Datos en general (características, usos, etc), una metodología para el desarrollo dirigido por modelos de un Almacén de Datos Relacional, la tecnología XML (utilidad y construcción de un XML Schema) y las Bases de Datos XML. Además, se han estudiado cuáles son las aproximaciones existentes hoy en día para el desarrollo de Almacenes de Datos XML. Por último, se han estudiado también los productos empleados para el desarrollo del caso de estudio, como son XML Spy, Oracle Workflow, Oracle XML DB y, principalmente, Oracle Warehouse Builder.

Una vez realizados estos estudios previos, se ha desarrollado un Almacén de Datos Relacional siguiendo la metodología estudiada y utilizando la herramienta Oracle Warehouse Builder.

Posteriormente se ha realizado una propuesta para el desarrollo de un Almacén de Datos XML. Esta propuesta está basada en una aproximación metodológica dirigida por modelos, en la que, partiendo de un PIM Multidimensional y aplicando las reglas de transformación definidas en la misma, se puede obtener el PSM XML, que será un XML Schema (el esquema del Almacén de Datos XML).

Seguidamente, para validar esta propuesta se ha desarrollado un caso de estudio. En este caso de estudio, una vez obtenido el XML Schema y haciendo uso de la herramienta Oracle XML DB, se ha procedido a registrarlo para implementar así la Base de Datos XML correspondiente al esquema del Almacén de Datos XML.

1. Introducción 5

1.1 Presentación del problema 6

1.2 Objetivos 7

1.3 Método de trabajo 8

1.4 Medios Hardware y Software 10

1.5 Estructura de la memoria 11

2. Estudios previos 12

2.1 Almacenes de Datos 13

2.2.1 Definición 13

2.2.2 Utilidades 14

2.2.3 Características 15

2.2.4 Estructuras de almacenamiento relacionales 16

2.2.5 Aproximaciones de desarrollo 20

2.2.6 Productos que soportan Almacenes de Datos 21

2.2 Metodología para desarrollar un Almacén de Datos Relacional 23

2.3 Tecnología XML y Bases de Datos XML 29

2.4 Aproximaciones existentes para desarrollar Almacenes de Datos XML 32

2.5 Producto Oracle Database 11g Release 1 35

2.5.1 Oracle Warehouse Builder 36

2.5.2 Oracle XML DB 38

3. Estudio y uso de una metodología para el desarrollo de Almacenes de Datos Relacionales 44

3.1 Caso de estudio 45

4. Aproximación metodológica para desarrollo de Almacenes de Datos XML 69

4.1 Descripción del proceso metodológico 70

4.2 Reglas de transformación de PIM a PSM 71

4.3 Caso de estudio 72

4.3.1 PIM del Almacén de Datos 72

4.3.2 PSM del Almacén de Datos XML 73

4.3.3 Implementación del Almacén de Datos XML 85

4.3.4 Propuesta de implementación para un Almacén de Datos XML en OWB .. 87

5. Conclusiones y futuros trabajos 94

5.1 Conclusiones 95

5.2 Futuros trabajos 96

6. Bibliografía y lugares visitados en Internet 97

Apéndice A: Instalación y configuración de Oracle Warehouse Builder 100

1. Instalación 101

2. Configuración 102

1. Introducción

1.1 Presentación del problema

Hoy en día las tecnologías de la información han provocado un gran impacto sobre la forma en que los productos y servicios se ofertan, distribuyen y venden. Se han convertido en una herramienta clave para el porvenir de los negocios.

La situación actual, en la que cada vez es más alto el nivel competitivo entre las empresas, provoca que las organizaciones recurran con más frecuencia a las nuevas tecnologías para conseguir estrategias de gestión que les reporten la información que necesitan.

Entre las nuevas tecnologías de la información que se están imponiendo en la actualidad, destacan los Almacenes de Datos (Data Warehouses, DW).

Un Almacén de Datos contiene una gran cantidad de información (datos históricos) especialmente diseñada para realizar análisis estratégicos sobre la información que contienen. Esta información suele ser muy útil para la toma de decisiones de la empresa.

En la actualidad cada vez es más común el uso de Internet para el porvenir de los negocios. Esto afecta directamente a los almacenes de datos, ya que sus fuentes de datos provienen en muchos casos de la Web.

El principal problema que presentan los datos de la Web para los Almacenes de Datos es que estos datos, frecuentemente son de tipos heterogéneos y complejos. Por este motivo, en algunos casos los diseñadores de almacenes de datos utilizan la tecnología XML para estructurar y unificar todos estos datos antes de ser almacenados. Además, XML se emplea en los Almacenes de Datos para implementar el modelo multidimensional, para definir los correspondientes artefactos de diseño (Hechos, Dimensiones, Medidas, Jerarquías, etc) y para facilitar el intercambio de datos y metadatos entre las fuentes de datos heterogéneas y los almacenes de datos.

En este PFC se va a realizar un estudio sobre los Almacenes de Datos y, concretamente, sobre los Almacenes de Datos XML. Además se va a estudiar una de las metodologías dirigidas por modelos para el desarrollo de Almacenes de Datos Relacionales (Mazón y Trujillo, 2007).

El propósito final de este PFC es proponer una aproximación metodológica para el desarrollo de un Almacén de Datos XML y validar la propuesta mediante el desarrollo de un caso de estudio.

También se verán las soluciones tecnológicas existentes para el desarrollo de Almacenes de Datos XML y se seleccionará la que mejor se adapte a este tipo de sistemas.

1.2 Objetivos

El objetivo principal de este PFC es desarrollar un Almacén de Datos XML proponiendo para ello una aproximación metodológica y validando la propuesta mediante un caso de estudio.

Con este fin, se han definido una serie de objetivos parciales:

- **Realización de estudios previos sobre:**
  - ***Almacenes de Datos***

    En primer lugar, será necesario hacer un estudio sobre qué son los Almacenes de Datos, para qué sirven, sus características, etc.
  - ***Metodología para desarrollo de Almacenes de Datos Relacionales***

    Se estudiará una metodología dirigida por modelos para la construcción de Almacenes de Datos Relacionales usando una extensión UML.
  - ***Tecnología XML y Bases de Datos XML***

    Se estudiarán las características principales de XML, su importancia en la actualidad y sus aportaciones en el entorno de los Almacenes de Datos. Además se estudiarán las particularidades de las Bases de Datos XML.
  - ***Aproximaciones existentes para desarrollar Almacenes de Datos XML***

    Se estudiarán las distintas aproximaciones que existen a día de hoy para desarrollar Almacenes de Datos XML.

- ***Productos***

  Se realizarán unos estudios previos sobre el producto Oracle 11g y algunos de sus productos que serán empleados para el desarrollo del caso de estudio, como Oracle Warehouse Builder y Oracle XML DB.

- **Propuesta metodológica para el desarrollo de Almacenes de Datos XML**

  Para poder crear de forma sistemática un Almacén de Datos XML, se planteará una aproximación metodológica definiendo unas reglas de transformación genéricas para pasar del PIM al PSM del Almacén de Datos XML.

- **Desarrollo de un Almacén de Datos XML para un caso de estudio.**

  Con el propósito de validar la propuesta metodológica del punto anterior, se realizará un caso de estudio para desarrollar un Almacén de Datos XML.

1.3 Método de trabajo

El método de trabajo seguido en la realización de este PFC está constituido por diferentes fases, como se puede ver en la figura 1. Paralelamente a las mismas se ha llevado a cabo la elaboración de la memoria de este trabajo.
A continuación, se van a describir las tareas a realizar en cada una de las fases:

FASE 1. Estudios previos

En esta fase se van a realizar una serie de estudios previos necesarios para alcanzar los objetivos marcados.

FASE 2. Estudio y uso de una metodología para el desarrollo de Almacenes de Datos Relacionales

Se estudiará una metodología para el desarrollo de Almacenes de Datos Relacionales (Mazón y Trujillo, 2007) y también se desarrollará un Almacén de Datos Relacional para un caso de estudio (el mismo que posteriormente se usará para validar nuestra propuesta).

FASE 3. Aproximación metodológica para Almacenes de Datos XML

En esta fase se va a proponer una aproximación metodológica definiendo unas reglas de transformación para conseguir transformar un PIM Multidimensional en un PSM, que en nuestro caso será el XML Schema del Almacén de Datos XML.

FASE 4. Validación de metodología para Almacenes de Datos XML

Con el fin de validar la propuesta metodológica realizada para Almacenes de Datos XML se desarrollará el mismo caso de estudio que en la fase 2.
En función de las mejoras o correcciones requeridas, se irá modificando y refinando la propuesta metodológica a medida que se avance con el caso de estudio.

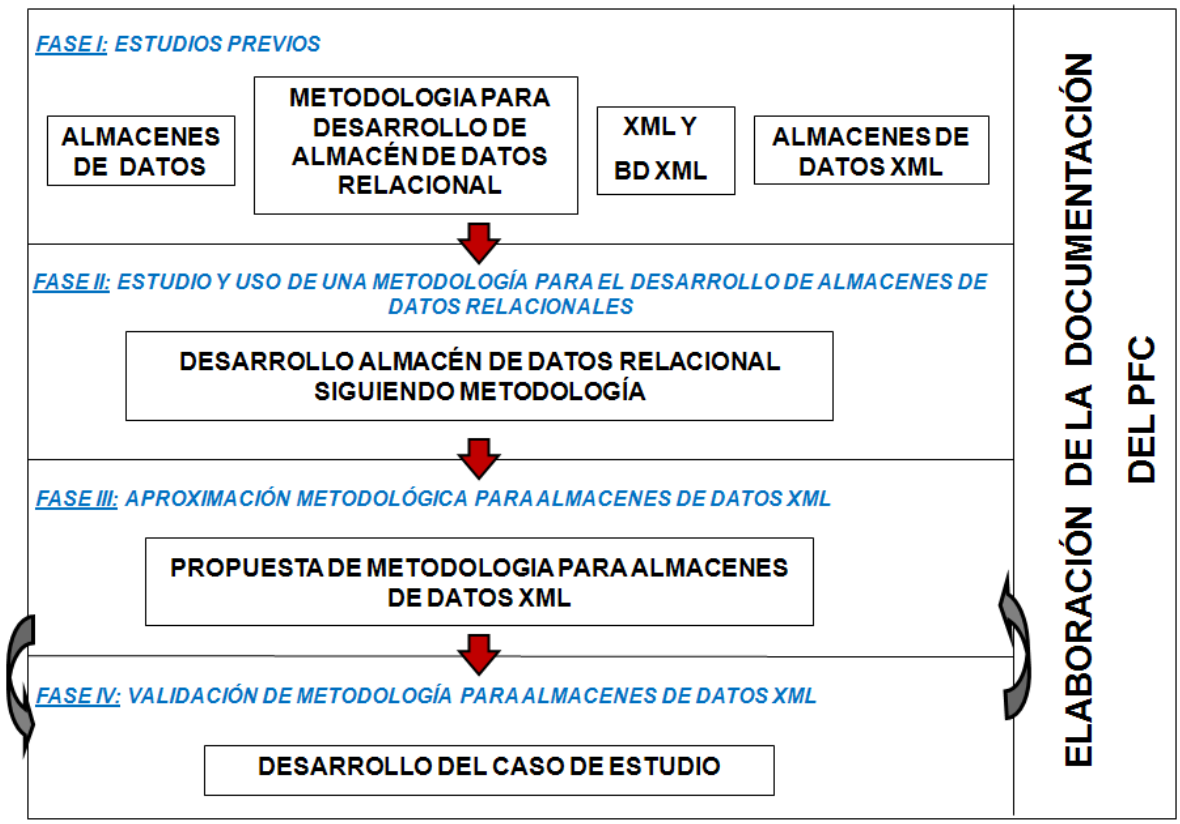

Figura 1: *Método de trabajo*

1.4 Medios Hardware y Software

Para la realización de este PFC se han requerido los siguientes medios HW y SW:

Medios Hardware

- Ordenador Intel® Core™2 CPU 6320, 1.86GHz, 1GB Memoria RAM.
- Ordenador Portátil ASUS Intel Core Duo, 2048 Mb, 160GB HDD.
- Impresora HP PSC 1510 All-In-One.

Medios Software

- ***Microsoft Windows XP – Profesional (PC):*** Sistema Operativo empleado como plataforma de trabajo.
- ***Microsoft Word 2007:*** Procesador de textos utilizado para la realización de la documentación de este proyecto.
- ***Microsoft Excel 2007:*** Hoja de cálculo utilizado para la creación de ficheros fuente en los Almacenes de Datos.
- ***Microsoft Office PowerPoint 2007:*** Utilizado para la realización de figuras y presentaciones.
- ***Jude Community 5.5.2 (August 28, 2009):*** Utilizado para la realización del PIM y el PSM de datos.
- ***XML Spy 1.3 (2001):*** Empleado para la realización del XML Schema sobre el contexto del caso de estudio a desarrollar.
- ***Oracle Database 11g Release 1:*** Sistema Gestor de Bases de Datos usado para la implementación del caso de estudio.
- ***Oracle Workflow:*** Herramienta de Oracle empleada para la creación del flujo de proceso de datos en el Almacén de Datos.
- ***Oracle Warehouse Builder:*** Herramienta de Oracle empleada para la construcción del Almacén de Datos.
- ***SQL Plus:*** Herramienta para interactuar con la BD en línea de comandos.
- ***Oracle XML DB:*** Utilizado para registrar un XML Schema y hacer pruebas de validación con instancias de documentos XML.
- ***Mozilla Firefox 3.6:*** Navegador usado para buscar información en Internet.
- ***Adobe Acrobat Reader 9.0:*** Visualizador de documentos PDF para la lectura de documentación técnica y artículos.

1.5 Estructura de la memoria

La presente memoria está estructurada en seis capítulos de la siguiente forma:

Este **primer capítulo** es una introducción al PFC que incluye una presentación del problema, los objetivos que se pretenden alcanzar, el método de trabajo a seguir y la descripción de los medios hardware y software usados.

En el **segundo capítulo** se hace una descripción de los estudios previos necesarios para la realización de este trabajo:

- Almacenes de Datos
- Una metodología para desarrollar un Almacén de Datos Relacional
- La tecnología XML y las Bases de Datos XML
- Las aproximaciones existentes para desarrollar Almacenes de Datos XML
- Productos

El **tercer capítulo** corresponde al estudio y uso de una metodología para el desarrollo de Almacenes de Datos Relacionales, comentando cuáles son las reglas de transformación propuestas para pasar de un PIM Multidimensional a un PSM Relacional. Se explicará cómo se ha llevado a cabo la aplicación de estas reglas para desarrollar el caso de estudio para la construcción de un Almacén de Datos Relacional.

El **cuarto capítulo** se centrará en la propuesta metodológica diseñada para el desarrollo de Almacenes de Datos XML detallando todos los pasos a seguir y su posterior validación mediante un caso de estudio.

En el **quinto capítulo** se exponen las conclusiones finales a las que se ha llegado y los futuros trabajos.

El **sexto capítulo** contiene la bibliografía y los lugares de Internet visitados.

Por último, se incluye al final de esta documentación un apéndice sobre la instalación y configuración de Oracle Warehouse Builder.

2. Estudios previos

Para la realización de este PFC ha sido necesario llevar a cabo una serie de estudios previos sobre los Almacenes de Datos en general, una metodología para el desarrollo de Almacenes de Datos Relacionales, la tecnología XML y las Bases de Datos XML. Además se han estudiado las aproximaciones existentes en la actualidad para desarrollar Almacenes de Datos XML y los productos Oracle Warehouse Builder y Oracle XML DB.

2.1 Almacenes de Datos

Al comienzo de este PFC, fue necesario realizar un estudio teórico de las características y el concepto de un Almacén de Datos, para así poder crear uno, posteriormente.

Este estudio respondió a preguntas sobre el concepto y explicó los diferentes términos que se detallan a continuación en los siguientes subapartados.

2.2.1 Definición

Un Almacén de Datos es una Base de Datos corporativa que permite realizar el proceso comúnmente conocido como “ETL” (*Extraction, Transformation and Load*). Este proceso consiste en extraer información de una o varias fuentes (otras BD, ficheros, etc), transformarla (depurarla, eliminando inconsistencias entre los sistemas operacionales) y cargarla en el esquema de almacenamiento destino del Almacén, para ofrecer así funcionalidad útil para el usuario y con un menor tiempo de respuesta (la información se encuentra ahora agrupada por temas organizados mediante Dimensiones).

La estructura de almacenamiento de los Almacenes de Datos sigue el paradigma Multidimensional, organizando la información en base a Hechos y Dimensiones. Un **Hecho** contiene las medidas o atributos que son relevantes para un proceso de negocio (ventas, repartos…), mientras que una **Dimensión** representa el contexto en el que un Hecho es analizado (dependiente, producto, cliente…) estructurando sus atributos de manera jerarquizada por niveles (categoría, subcategoría…).

Un apunte interesante que conviene mencionar es que, en el ámbito de la implementación de los Almacenes de Datos, la información de interés de los mismos (tabla de Hechos), se recoge en una estructura denominada **Cubo**

(On-Line Analytical Processing, OLAP)**,** de donde se extraerá la información para realizar las consultas para los análisis.

Las diferencia fundamental entre un Almacén de Datos y una Base de Datos (BD) operacional es que, no trata de almacenar únicamente las operaciones transaccionales de la empresa en el día a día, sino que contiene un histórico con todos los datos de la empresa, de manera organizada y preparada para realizar análisis avanzados, incluso de tipo estratégico, lo que se antojaría imposible con las BD tradicionales.

2.2.2 Utilidades

Dentro de las ventajas que ofrece el uso de Almacenes de Datos, destaca por encima de todas la ayuda que proporciona a sus usuarios a la toma de decisiones gerenciales (en función de lo que pasó en el pasado, podemos estimar qué sería lo más conveniente en el futuro) o la visualización del histórico de una variable en concreto (evolución de los datos).

Viene a resolver problemas de manejo y uso adecuado de grandes fuentes de datos y de diversos tipos. Por ello, uno de los principales objetivos es el de convertir los datos operacionales en información relacionada y estructurada, homogénea y de mayor calidad, identificada convenientemente mediante los Metadatos asociados a los datos (cada dato estará identificado por una descripción, un origen, historial, etc).

En resumen, el fin de un Almacén de Datos (principalmente en el ámbito empresarial) es reunir y consolidar las diferentes BD que se tienen en una organización (diferentes departamentos o áreas funcionales) como subsistemas de información independientes, en una gran BD, recogiendo datos muy dispares y ofreciendo así una mejor organización de cara a la administración de la empresa y la fuente fiable de información necesaria para la toma de decisiones de la misma. Esto es indispensable para el entendimiento del pasado y para contar con los elementos necesarios para la planeación del futuro de corto, mediano y largo plazo.

Los ejecutivos y administradores buscan respuestas a preguntas como:

- *¿Qué productos están comprando nuestros clientes?*

- *¿Qué productos no están comprando?*

- *¿Qué está haciendo la competencia?*

- *¿Cómo están los costos por cada línea de producto, comparados con los últimos tres años?*

- *¿Qué factores causan incrementos en los costos?*

2.2.3 Características

Las características principales de un Almacén de Datos son las que se describen a continuación:

- **Integrado**

  Los datos almacenados en el Almacén deben estar integrados en estructuras consistentes, es decir, no puede haber inconsistencias entre los diversos sistemas operacionales.

- **Orientado al tema**

  Sólo se integran los datos necesarios para el proceso de negocio desde el entorno operacional. Los datos se clasifican por temas para facilitar su acceso y entendimiento. Esto es posible gracias a la jerarquía que existe en el esquema destino entre los Hechos y las Dimensiones. Cada Dimensión corresponde a un tema determinado pudiendo contener diferentes niveles en cada Dimensión (para detallar categorías y subcategorías).

  Un ejemplo claro es el de las ventas, en el que la tabla de Hechos “Ventas” contiene las Dimensiones “Dependiente”, “Lugar”, “Producto” y “Fecha” de la venta (podrían ser otros temas los empleados para las Dimensiones, pero estos son los comunes).

- **Variante en el tiempo**

  A diferencia de los sistemas operacionales, en los que los datos siempre reflejan el estado de la actividad del negocio en el presente, los Almacenes de Datos mantienen un histórico con los valores que ha tomado una variable a lo largo del tiempo. Esto es útil para, entre otras cosas, permitir comparaciones y observar evoluciones.

- **No volátil**

  La información del Almacén de Datos es permanente, no sufre modificaciones. Esto es así porque la información existe para ser leída, y si se desea realizar alguna actualización, se añadirá un nuevo valor.

- **Uso de Metadatos**

  Los Metadatos son de utilidad para conocer la procedencia de la información, su periocidad de refresco, su fiabilidad, etc.

2.2.4 Estructuras de almacenamiento relacionales

Siguiendo la metodología desarrollada por Kimball y su enfoque dimensional para la construcción de un Almacén de Datos, este modelo (no normalizado) incluye las Dimensiones de análisis y sus atributos, su organización jerárquica y los diferentes Hechos de negocio que se quieren analizar**.**

En este enfoque, existen tres modelos de esquemas para un Almacén de Datos, en cuanto a su estructura de almacenamiento:

1. **Estrella**

   Se compone de una tabla de Hechos central que contiene los datos para el análisis, rodeada de las tablas de Dimensiones. Las tablas dimensionales tienen una clave primaria simple, mientras que en la tabla de Hechos, la clave principal está compuesta por varias claves que referencian a cada una de las claves principales de las dimensionales, como se puede observar en la figura 2, donde la clave primaria de la

tabla de Hechos “Hechos_Ventas” está formada por “idProducto”, “idAlmacen”, “idPromocion”, “idCliente” e “idTiempo”.

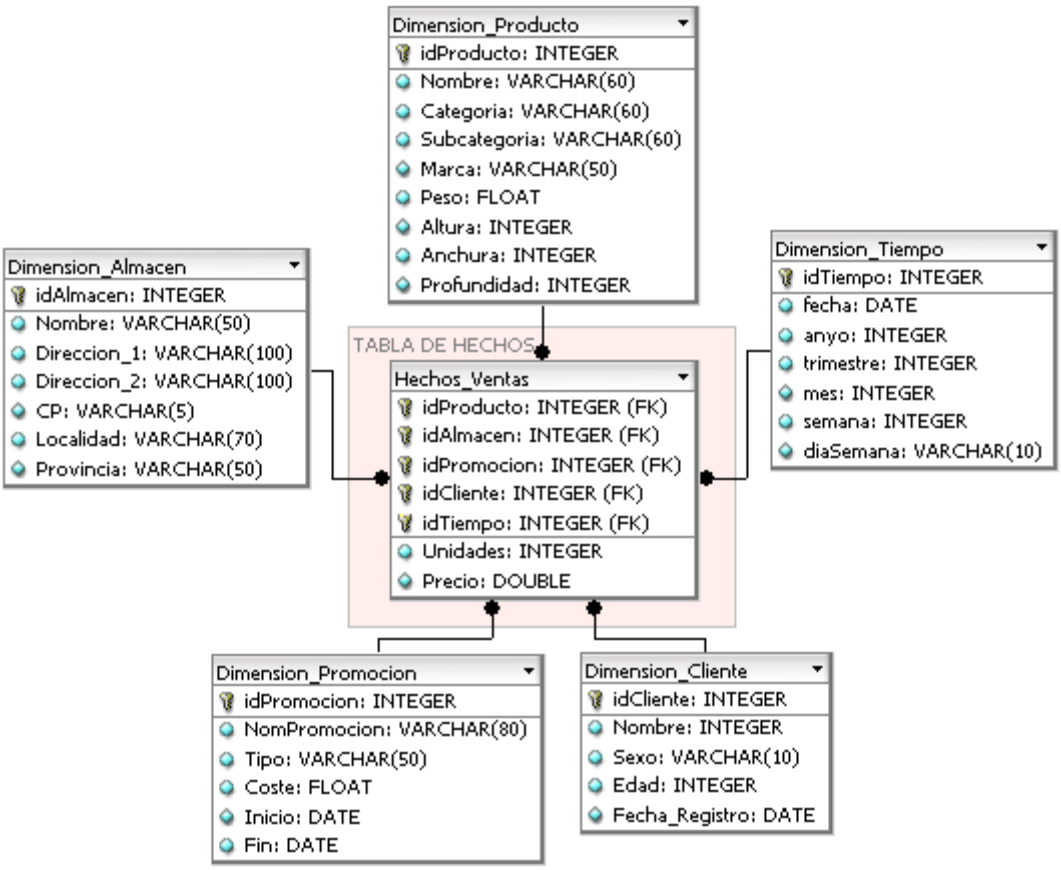

Figura 2: *Esquema en estrella de un Almacén de Datos*

2. Copo de nieve

Esta estructura de almacenamiento es una variación del esquema en “Estrella”.

Se da cuando alguna de las Dimensiones se implementa con más de una tabla de datos, es decir, que estén compuestas por más de un nivel en su jerarquía.

La finalidad es normalizar las tablas y así reducir el espacio de almacenamiento al eliminar la redundancia de datos.

Tiene la contrapartida de generar peores rendimientos al tener que crear más tablas de Dimensiones y más relaciones entre las tablas (*JOINS*) lo que tiene un impacto directo sobre el rendimiento.

Por ejemplo, en la figura 3, se puede observar una variación del modelo anterior donde las Dimensiones "Dimension_Producto" y "Dimension_Cliente" contienen 2 o 3 niveles en sus jerarquías para representar las categorías y subcategorías (marca de un producto, sexo o franja de edad de un cliente, etc).

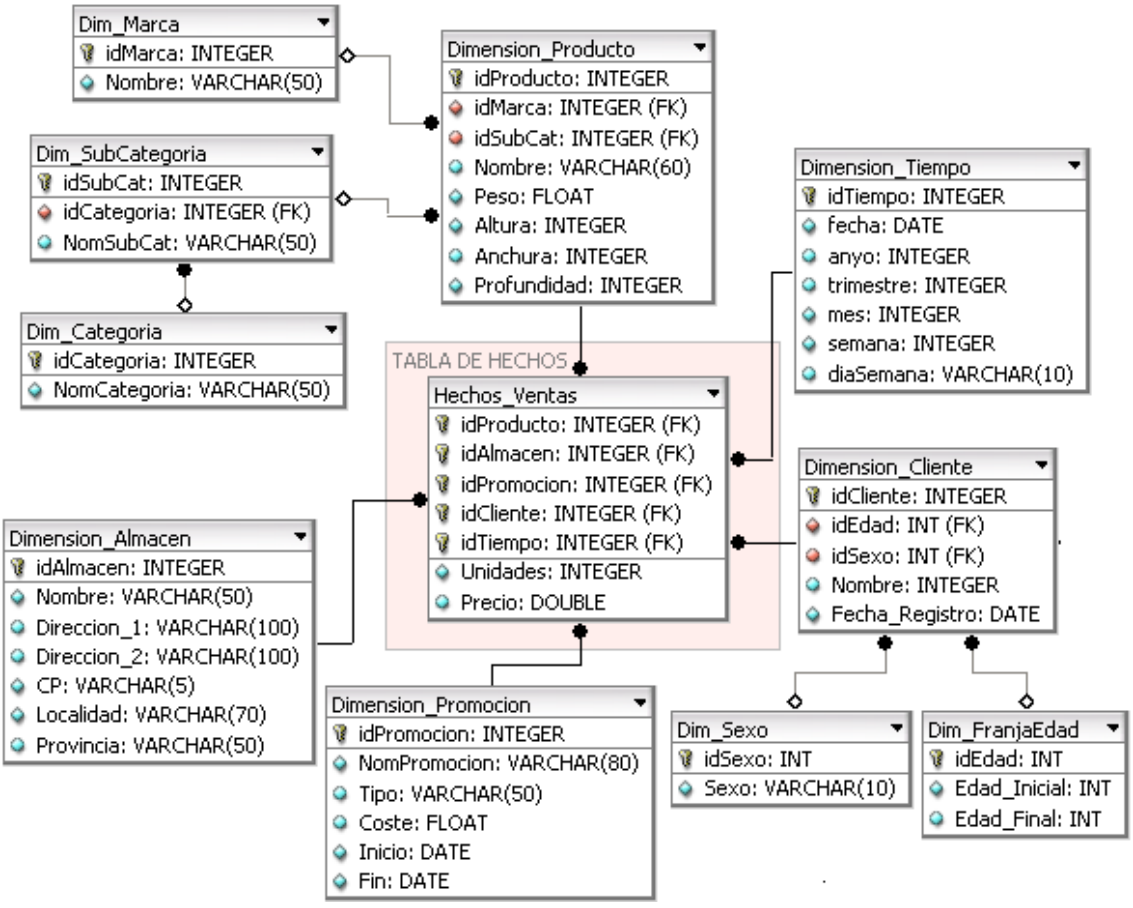

Figura 3: *Esquema en copo de nieve de un Almacén de Datos*

3. Constelación

Esta estructura de almacenamiento es una variación del esquema en estrella, que proporciona un mejor uso del espacio de almacenamiento.

Los esquemas en estrella y copo de nieve pueden generalizarse con la inclusión de distintas tablas de Hechos que comparten todas o algunas de las Dimensiones, tal y como se puede observar en la figura 4.

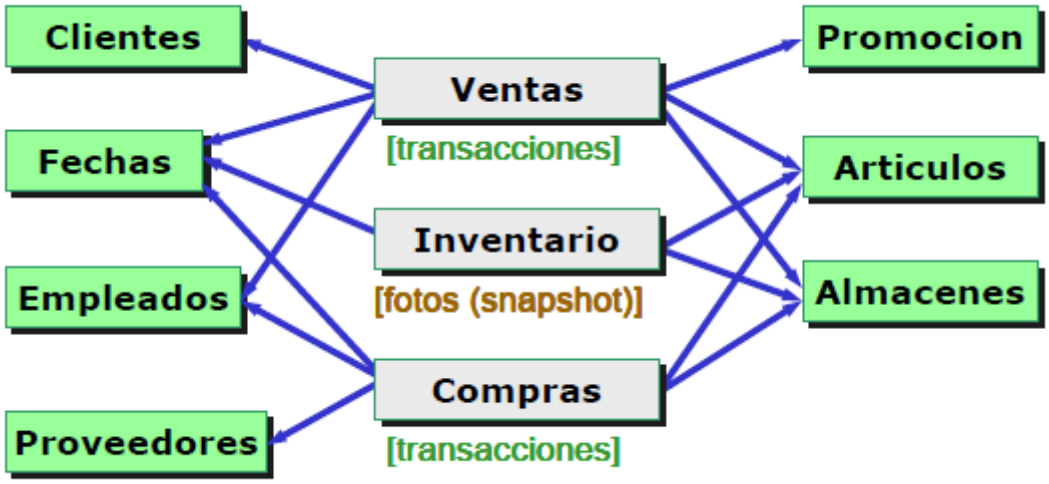

Figura 4: *Esquema en constelación de un Almacén de Datos*

En el ejemplo anterior, puede observarse cómo las tablas de Hechos "Ventas", "Inventario" y "Compras" referencian a las Dimensiones "Fechas", "Artículos" y "Almacenes" todas ellas. De la misma manera, ocurre con la Dimensión "Empleados", que es referenciada por las tablas de Hechos "Compras" y "Ventas".

Y en cuanto al mejor uso del espacio de almacenamiento, se debe al hecho de que una misma Dimensión pueda ser referenciada por más de una tabla de Hechos.

2.2.5 Aproximaciones de desarrollo

Existen dos metodologías o enfoques diferentes para la construcción de un Almacén de Datos encabezadas por dos de las figuras más grandes de la Business Intelligence, la metodología desarrollada por **Ralph Kimball** (y su enfoque dimensional) y la definida por **Will Inmon** (y su enfoque *Enterprise Warehouse*).

La propuesta de ambos autores puede sintetizarse de la siguiente manera:

- Bill Inmon: según este autor, el Almacén de Datos es una parte del todo que conforma a un sistema de inteligencia. Una empresa tiene un Almacén de Datos, y los Data Marts tienen como fuente de información ese Almacén de Datos. Ésta aproximación también es conocida como "Top-Down".
- Ralph Kimball: bajo el paradigma de este autor, el Almacén de Datos se compone por el conglomerado de todos los Data Marts generados en una empresa. La información siempre se almacena en un modelo dimensional. Otra forma de denominar esta aproximación es como "Bottom-up".

A partir de estos dos conceptos es como se implementaron los Almacenes de Datos en un principio y hasta el día de hoy, pues han sido los más influyentes y mejor conceptualizados por sus desarrolladores. Sin embargo, eso no quiere decir que hayan surgido otros paradigmas e incluso, con la aparición de las nuevas tecnologías, los ya establecidos han evolucionado.

2.2.6 Productos que soportan Almacenes de Datos

Algunos de los productos que soportan la tecnología Almacén de Datos son los siguientes:

- **Oracle**

Es actualmente el líder del mercado global en Almacén de Datos gracias a su herramienta Oracle Warehouse Builder. Crear un Almacén de Datos en Oracle es la mejor manera de consolidar y organizar todos sus datos para que se puedan administrar, ver y analizar fácilmente. Con una sola interfaz de su administración, características de autodiagnóstico y autoajuste, Oracle Database 11*g* simplifica el mantenimiento de su Almacén de Datos en constante expansión.

Este producto mejora el rendimiento, la disponibilidad y la facilidad de gestión de los Almacenes de Datos mediante la partición de las tablas grandes.

Además, proporciona una herramienta especializada para el desarrollo de Almacenes de Datos a partir de su Release1, llamado “Oracle Warehouse Builder”. Esta herramienta de integración empresarial, administra todo el ciclo de vida de los datos y metadatos para Oracle Database 11*g.*

- **MySQL**

MySQL posee un motor de alto rendimiento para las consultas, proporcionando además una gran rapidez en la inserción de los datos.

El uso de este motor permite aprovechar mejor los motores específicos para mejorar el rendimiento y reducir los costes de almacenamiento. Aporta también algunas características, como tablas de memoria principal o índices hash, que reducen el tiempo de espera hasta en un 80%. Estas características hacen de MySQL un fuerte candidato para webs y aplicaciones de Negocio Inteligente (mediante Almacenes de Datos).

- **SQL Server**

Proporciona la herramienta SQL *Server Integration Services* (SSIS) para aplicar las operaciones de ETL que se requieren para construir un Almacén de Datos. SSIS proporciona entre otras ventajas:

- Gran rapidez como herramienta de ETL.
- Conectividad a la no-SQL Server, por ejemplo, las fuentes de datos Oracle, Teradata, mainframe de DB2, y sistemas de ERP.
- Una arquitectura escalable que proporciona una Plataforma multihilo de 64 bits.

La herramienta que finalmente fue seleccionada para el desarrollo de este caso de estudio fue **Oracle Database 11g Release1**, principalmente por su extensión **Oracle Warehouse Builder**.

Esta decisión fue tomada debido a que, además de tratarse de una herramienta muy útil y rápida para la construcción de Almacenes de Datos, incorpora un amplio soporte de almacenamiento y la gestión de datos XML, aportando una documentación muy amplia detallada. Además, permite realizar todo el desarrollo completo del Almacén mediante una interfaz gráfica, sin ser necesario emplear la ejecución de órdenes por consola (salvo para desbloquear cuentas de usuario y asignarles diferentes privilegios).

2.2 Metodología para desarrollar un Almacén de Datos Relacional

En este apartado se va a resumir una propuesta metodológica para el desarrollo de Almacenes de Datos Relacionales, definida por Mazón y Trujillo en 2007, cuyo título es "*An MDA approach for the development of Data Warehouses*".

Los Almacenes de Datos son comúnmente representados como una arquitectura multicapa, en el cual los datos entrantes en una capa corresponden a los salientes de la capa anterior. Se han propuesto diferentes métodos y aproximaciones para el diseño de las capas de un Almacén de Datos, sin embargo muchos de los problemas que surgen cuando se trabaja con este tipo de herramientas, procede de la interoperabilidad e integración de las diferentes capas.

Para sobrellevar este problema, la primera propuesta que realizan los autores ha sido la definición de un marco para desarrollar Almacenes de Datos basado en UML y en el Proceso Unificado (*Unified Process*, UP). Este marco dirige el diseño de todo el Almacén de Datos usando diferentes perfiles UML.

Estos perfiles han sido desarrollados para adaptar UML a ciertos aspectos del diseño de Almacenes de Datos, como el modelo multidimensional del repositorio del Almacén de Datos y el modelo del proceso ETL, describiendo el trasiego de datos desde las fuentes hasta el almacén destino.

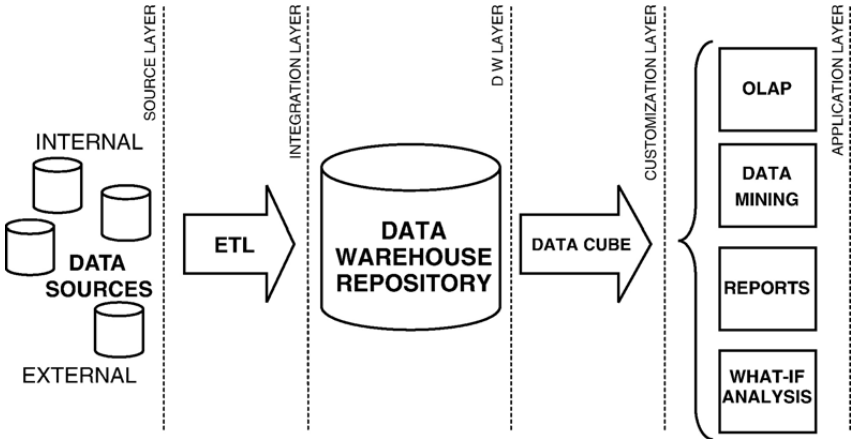

Figura 5: *Arquitectura multicapa de los Almacenes de Datos*

Para la implementación final del Almacén de Datos existe un proceso descrito siguiendo la propuesta MDA en el que se definen las reglas de transformación entre diferentes modelos conceptuales. Este marco separa la funcionalidad del sistema en un modelo independiente de la plataforma (*Platform Independent Model,* PIM) de la especificación de la implementación en una tecnología determinada, la cual se recoge en un modelo específico de la plataforma (*Platform Specific Model,* PSM). Por lo tanto, cada PIM puede ser transformado automáticamente a diferentes PSMs, dependiendo de la plataforma requerida.

Finalmente, cada PSM derivado, proporcionará el código acorde a la plataforma establecida para la implementación del Almacén de Datos.

Las principales ventajas de este marco son:

1. La complicada tarea del diseño de un Almacén de Datos es abordada de manera sistemática y bien estructurada, gracias a la utilización de MDA. Además, al realizarse desde una perspectiva conceptual es totalmente independiente de los detalles de implementación.
2. El diseño sigue un sistema de modelado integrado, evitando de este modo la interoperabilidad entre capas y problemas de integración.
3. La implementación del Almacén de Datos puede ser derivada del desarrollo previo de un PIM y la posterior aplicación de las correspondientes reglas de transformación para la obtención del PSM.
4. Utiliza UML, un lenguaje de modelado orientado a objetos comúnmente aceptado y usado, lo cual evita el aprendizaje de nuevos modelos a los desarrolladores y sus notaciones correspondientes para el modelado Multidimensional (MD). Además, esta ventaja de utilizar UML para abordar el diseño de cada componente del Almacén de Datos de manera integrada, proporciona también una fácil interoperabilidad entre las diferentes capas.

- **PIM Multidimensional**

Un PIM es una vista de un sistema independiente de la plataforma. Esto significa que describe el comportamiento del sistema evitando mencionar los detalles específicos de una plataforma. Este punto de vista corresponde a un nivel conceptual que representa las principales propiedades multidimensionales sin tener en cuenta los detalles específicos de la tecnología empleada para la Base de Datos. Además, la especificación del Almacén de Datos es independiente de la plataforma donde será implementado.

El PIM será desarrollado siguiendo el perfil UML propuesto para el modelado multidimensional. Este perfil contiene los estereotipos necesarios para representar elegantemente las principales propiedades multidimensionales a nivel conceptual.

Específicamente, las propiedades estructurales del modelado multidimensional son representadas mediante un diagrama de clases UML, en el que se muestra la información claramente organizada en Hechos y Dimensiones. Estos Hechos y Dimensiones son representados como "**clases Hecho**" y "**clases Dimensión**", respectivamente.

Las "clases Hecho" están compuestas por los atributos que describen la información de interés para los Hechos y por relaciones de agregación con las n "clases Dimensión". La cardinalidad mínima en estas relaciones para las clases Dimensión es 1, para indicar que cada Hecho debe estar siempre relacionado con todas y cada una de las Dimensiones que contiene.

Por otro lado, debido a que las Dimensiones pueden contener uno o más niveles en su jerarquía (para representar categorías y subcategorías, por ejemplo), las "clases Dimensión" también deben incorporar un mecanismo en UML para representarlo. Se trata de incorporar para cada uno de los niveles de la jerarquía de la Dimensión una "**clase Base**". Cada clase Base puede contener diferentes atributos de Dimensión (un atributo OID, por ejemplo), pero lo que sí debe contener es un atributo descriptor. Este atributo descriptor es necesario porque será el que ejerza de representante del nivel a la hora de asociarse con el resto de niveles de la jerarquía (relaciones entre clases Base en el PIM) para la carga de los datos.

En la figura 6 se muestra un ejemplo de un PIM de Datos Multidimensional.

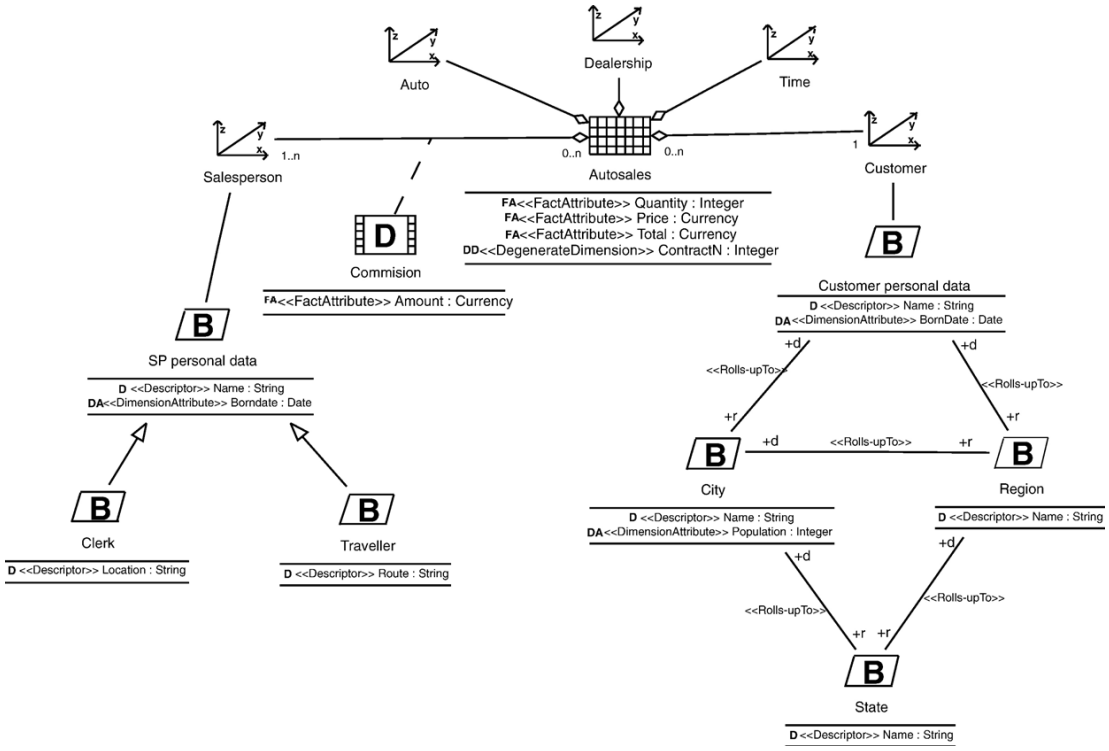

Figura 6: *PIM Multidimensional*

En este PIM, se puede observar una tabla de Hechos central “Ventas” (*Autosales*) compuesta de 5 Dimensiones. Los atributos de un Hecho “Venta” son las “unidades”, el “precio” de la unidad y la “cantidad total a pagar”.

En cuanto a las Dimensiones, las recogidas en este modelo son “Fecha” (*Time*), “Distribuidor” (*Dealership*), “Autoservicio” (*Auto*), “Dependiente” (*Salesperson*) y “Cliente” (*Customer*).

La Dimensión “Dependiente” tiene una jerarquía compuesta por dos niveles. El primer nivel está constituido por una tabla “Base” referente a los “datos del Dependiente” (*SP Personal Data*). El segundo nivel está constituido por dos tablas Base como subcategorías que heredan del nivel anterior y sirven para recoger los datos referentes a “Empleado” (*Clerk*) si se trata de un empleado normal, conocer su localización, y “Viajante” (*Traveller*) en caso de que deba realizar viajes, recoger sus rutas.

Además, también se observa una atributo “Comisión” (*Commision*) entre la tabla de Hechos “Ventas” y la Dimensión “Dependiente” para representar la cantidad percibida por éste en cada venta que realiza.

Por otro lado, la Dimensión “Cliente” posee una jerarquía de tres niveles. El primer nivel se constituye de una tabla “Base” que recoge los “datos del cliente” (*Customer personal data*). El segundo nivel lo constituyen dos tablas Base, las cuales recogen tanto la “Ciudad” (*City*) como la “Región” (*Region*) del cliente. Finalmente, el tercer nivel de la jerarquía, lo compone una tabla Base referente al “Estado” (*State*) para recoger el Estado al que pertenece dicho cliente.

El resto de Dimensiones (*Auto, Dealership y Time*) están compuestas por un único nivel en su jerarquía.

- **PSM**

Un PSM es una vista de un sistema específico para una plataforma. Representa el modelo del mismo sistema especificado por el PIM, pero también especifica cómo este sistema hace uso de la plataforma seleccionada.

En este marco, un PSM se corresponde con una representación lógica del modelo multidimensional orientado a un específico tipo de tecnología de Base de Datos.

La representación lógica más común para los modelos multidimensionales es el esquema relacional en “estrella”. Este esquema consiste en una tabla de Hechos central que contiene una clave compuesta que referencia a cada una de las Dimensiones, cada una de ellas con una única clave primaria.

Este modelo relacional nos permite representar tablas, atributos, claves primarias, claves ajenas, etc.

- **Transformación de PIM a PSM**

Definir las transformaciones formales que permitan derivar un PSM de un PIM de manera automática es la tarea más compleja de MDA. Este marco proporciona una notación gráfica que permite especificar formalmente transformaciones entre modelos que son fácilmente leíbles, entendibles, adaptables y mantenibles.

Han sido desarrollados todos los pasos necesarios para transformar un PIM a un PSM para una plataforma relacional (de acuerdo a una representación del esquema en estrella), como se puede observar en la tabla 1.

PIM	PSM
Transformaciones para las clases	
Hechos	Tabla de Hechos que incluirá, además de sus atributos de Hecho, un atributo ID que actuará como identificador.
Dimensiones	Tabla de Dimensión que incluirá, además de sus atributos de Dimensión, un atributo ID que actuará como identificador.
Bases	Tabla de Base que incluirá, además de sus atributos de Base, un atributo ID que actuará como identificador.
Transformaciones para las relaciones	
Hecho - Dimension	La tabla de Hechos dispondrá de una clave ajena hacia cada una de las Dimensiones que la compone, representando su anterior relación de agregación con una relación de 1 a N.
Dimension - Base	Cada Dimensión contendrá una clave ajena hacia su correspondiente tabla Base, relacionándose con ella mediante una relación de 1 a N.
Base - Base	Las relaciones entre Bases se llevarán a cabo mediante una relación de 1 a 1, independiente del nivel de la jerarquía a la que pertenezca.

Tabla 1: *Reglas de transformación de PIM a PSM para Almacén de Datos Relacional*

El modelo fuente es la parte del metamodelo PIM que interfiere con la parte del metamodelo PSM que representa el modelo destino (esquema de almacenamiento). En este caso, una colección de elementos de nuestro perfil UML que representa una clase Dimensión y el nivel terminal (clase Base asociada con clase Dimensión) juega con un conjunto de elementos del metamodelo relacional CWM (Common Warehouse Metamodel) que representa una "clase Tabla" con una clave primaria. Esta relación determina la transformación en el siguiente orden:

Una vez que el PIM es validado, se realiza la transformación de sus elementos (y sus correspondientes asociaciones), que serán creados de acuerdo al metamodelo PSM: una nueva clase Tabla con el mismo nombre que la clase Dimensión, una columna y una clave primaria.

2.3 Tecnología XML y Bases de Datos XML

2.3.1 Tecnología XML

XML es una tecnología en realidad muy sencilla que tiene a su alrededor otras tecnologías que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores.

Independientemente de la relevancia que XML tiene sobre el desarrollo de Almacenes de Datos, también tiene una enorme importancia en la tecnología web actual, ya que es la base de numerosos procesos y técnicas.

El objetivo fundamental de XML es intercambiar datos estructurados entre sistemas de información, fundamentalmente a través de Internet. Se trata de un formato de texto plano, lo que facilita enormemente la transferencia de información, logrando independencia con respecto a las diferentes plataformas.

A la hora de crear documentos XML, existen multitud de maneras en cuanto a sintaxis de recoger la misma información, por lo que siempre se recomienda para evitar problemas la utilización de estándares a los que ceñirse. XML ofrece dos posibles estándares:

- **XML Schema**

  XML Schema es un estándar definido por el W3C para poder especificar una estructura válida para documentos XML.
  Un XML Schema es un documento que define el contenido y la estructura de un tipo de documentos XML, es decir, describe los elementos y atributos que pueden estar contenidos en un documento XML y la forma en que los elementos están organizados dentro de la estructura jerárquica del documento.
  Su estructura consiste en:

  - Los elementos que pueden aparecer en el documento.
  - Los atributos que puede tener cada elemento.
  - El modo en que se anidan los elementos (padres e hijos).
  - El orden en que deben aparecer los elementos hijos de un mismo padre.
  - El número máximo de elementos hijos para cada padre.
  - Si un elemento puede estar vacío o no.
  - Los tipos de datos para elementos y atributos.
  - Los valores por defecto y fijos para elementos y atributos.

- **DTD**

  DTD es una definición que especifica restricciones en la estructura y sintaxis de un documento XML.

  En una DTD podemos hacer cuatro tipos de declaraciones:

  - Declaraciones de tipo de elemento, establecen qué elementos pueden formar parte del documento y cuales pueden formar parte de su interior (los elementos se anidan unos dentro de otros).

  - Declaraciones de listas de atributos de cada uno de los elementos (si es que tienen).

  - Declaraciones de entidades, tanto internas (sólo se puede referenciar a su contenido desde el mismo fichero), como externas (se las puede referenciar desde otros ficheros).

- Declaraciones de notación, para indicar el tipo de los ficheros binarios referenciados mediante las referencias a entidades externas.

- **Diferencias entre DTD y XML Schema**

  La principal diferencia entre ambos estándares es que el XML Schema, además de definir las restricciones de estructura y sintaxis del documento XML, permite una serie de ventajas adicionales tales como:

  - una especificación más robusta de los tipos base que se pueden emplear dentro del esquema,
  - permitir tipos definidos por el usuario,
  - permitir agrupaciones de atributos,
  - permitir validaciones de documentos con varios *namespaces* y
  - la posibilidad de extender tipos de un modo similar a la herencia en objetos.

2.3.2 Bases de Datos XML

Este tipo de Bases de Datos son completamente distintas a las relacionales, las cuales en algunos casos, tienen en la actualidad soporte para XML, pero aún siguen almacenando toda la información de manera relacional, o en el caso contrario, almacenan todo el documento en formato Binary Large Object (BLOB). La principal característica que brindan estas Bases de Datos XML es la capacidad de obtener los resultados de las consultas en formato XML.

Una Base de Datos XML no posee campos, ni almacena datos atómicos, lo que ella almacena son documentos XML en base a la estructura definida mediante el correspondiente XML Schema.

En este PFC se va a trabajar con la Base de Datos XML que proporciona Oracle, denominada **Oracle XML DB**. Con esta herramienta que posteriormente explicaremos de manera detallada, se pretende desarrollar un caso de estudio para validar la propuesta realizada. Para ello, implementaremos una Base de Datos XML para nuestro Almacén de Datos XML mediante el registro de nuestro XML Schema.

2.4 Aproximaciones existentes para desarrollar Almacenes de Datos XML

Según (Ravat, et al. 2009) en su trabajo titulado *"Finding an Application-Appropriate Model for XML Data Warehouses",* en el proceso de integración en un Almacén de Datos cuya información contenida en documentos XML, existen dos posibles alternativas:

1. Integración lógica (federación), donde los documentos XML son almacenados tal cual (repositorio de ficheros) de manera independiente al Almacén de Datos.
2. Integración física, donde los documentos XML son directamente almacenados en un Almacén de Datos.

A continuación, se van a mostrar estas dos tendencias con más detalle.

1) Integración lógica de documentos XML con un Almacén de Datos

En algunos casos puede no ser posible integrar los datos XML dentro de un Almacén de Datos. Esto puede ser debido a restricciones legales, tales como la imposibilidad del uso de determinados datos privatizados por los derechos de autor.

Pero, por otro lado, esto también puede ser debido a restricciones físicas, tales como datos que, al estar cambiando de manera muy rápida, sea imposible sincronizar estos cambios con los procesos de refresco de la información del Almacén de Datos (por ejemplo, a la hora de almacenar los productos del stock de un supermercado).

La solución consiste en tener dos sistemas por separado: por un lado, el Almacén de Datos, y por otro, un repositorio para documentos XML. Y a la hora de efectuar las consultas oportunas sobre ellos, un motor cualificado de consultas se encargará de distribuir las extracciones de información entre ambos sistemas y los resultados serán fusionados en un análisis multidimensional, como se puede ver en la figura 7.

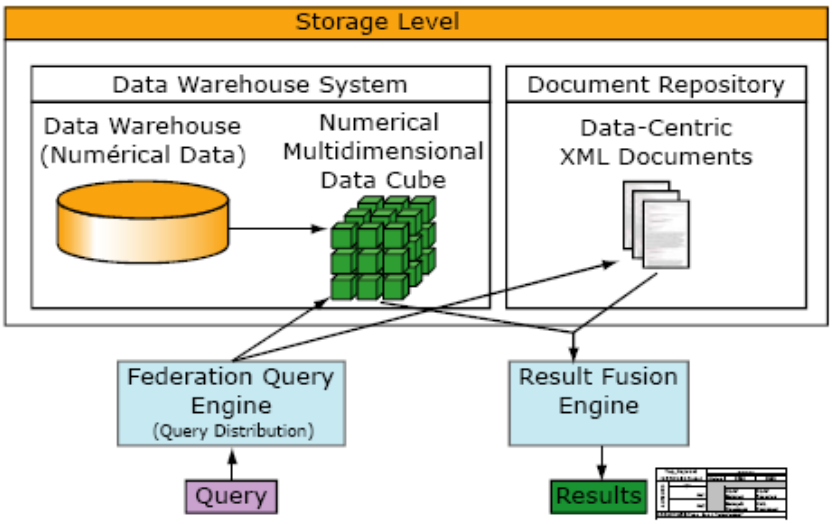

Figura 7: *Arquitectura del modelo lógico de integración de XML en un Almacén de Datos*

Las principales dificultades que se plantean en esta aproximación son:

- La fusión de XML con los datos almacenados en el Almacén de Datos para mostrar el resultado de las consultas.
- Creación de una única vista unificando ambos sistemas.

2) Integración física de datos XML en un Almacén de Datos

El objetivo de este tipo de Almacenes de Datos es operar sobre una Base de Datos XML nativa y permitir consultas de estilo OLAP sobre dicho almacén XML (ver figura 8). Por lo tanto, por un lado, se extraerán los datos contenidos en los ficheros XML fuente (primera etapa del proceso ETL), y por otro, se almacenarán en el esquema destino del Almacén de Datos para así poder ser analizados posteriormente.

Para cumplir con tal propósito, será necesario diseñar un modelo que permita almacenar todas las características del documento XML original, es decir, recoger en forma de metadatos la información relevante de la estructura del documento, además por supuesto de los datos contenidos en él.

Los datos XML extraídos del Almacén de Datos son estructurados en un Cubo multidimensional y las futuras consultas OLAP serán lanzadas sobre este Cubo.

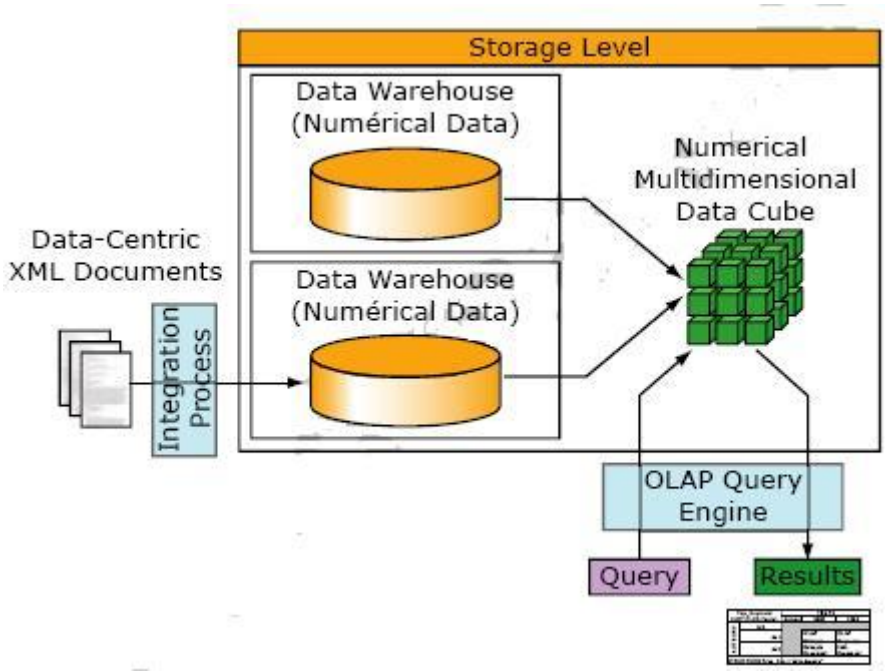

Figura 8: *Arquitectura de almacenamiento físico de documentos XML en Almacénd de Datos*

Los principales problemas de esta aproximación son:

- Diseñar el modelo lógico multidimensional empleado en el Cubo OLAP.
- Bajo rendimiento, debido a que la tecnología de las Bases de Datos XML son menos maduras que las relacionales.

Una vez conocidas y estudiadas ambas aproximaciones, se trabajará en este PFC para desarrollar un Almacén de Datos que integre físicamente los datos de los documentos XML en su almacén.

2.5 Producto Oracle Database 11g Release 1

Para el desarrollo de este PFC, se decidió utilizar la herramienta Oracle Database 11g Release1**.** Se trata de una herramienta comúnmente conocida por su capacidad para ofrecer seguridad en la información y un gran sistema de almacenaje, capaz de soportar grandes cantidades de datos.

Además se trata de una herramienta empleada en algunas asignaturas de la carrera, lo cual la colocó con cierta ventaja por delante de otras plataformas como MySQL o SQL Server, también muy bien preparadas.

Pero el motivo fundamental de su elección no fue otro que la posibilidad que ofrece Oracle de crear Almacenes de Datos.

Oracle incluye en su paquete de Base de Datos 11g una herramienta llamada **"Oracle Warehouse Builder"** (OWB) especializada en la creación de Almacenes de Datos y su posterior manejo de información de manera eficiente.

En cuanto a elegir la versión 11g en lugar de la 10g u otras anteriores fue debido a que, además de las ventajas que ofrece tener la última versión de un Software, esta última proporciona en el mismo paquete la Base de Datos tradicional de Oracle y la herramienta Oracle Warehouse Builder, mientras que en las demás versiones se debía descargar de forma independiente.

Recientemente se ha encontrado información sobre un nuevo paquete que incorpora Oracle 11g Release 2, capaz de tratar directamente con documentos XML a través de un tipo de dato llamado "XML Type". Gracias a esto, se da la posibilidad de extraer el contenido de los documentos XML fuente y crear unas tablas relacionales donde almacenar su información de manera automática.

El motivo por el que no se utilizó el Release 2 en este PFC fue que en el momento en el que se descargó la versión anterior (Release 1), ésta aún era una versión Beta.

Dentro de los productos de Oracle empleados para el desarrollo de este caso de estudio, cabe destacar Oracle Warehouse Builder y Oracle XML DB, que veremos a continuación.

2.5.1 Oracle Warehouse Builder

Es la herramienta para la creación de Almacenes de Datos empleada en este Proyecto de Fin de Carrera.

Se trata de un entorno bastante intuitivo e idóneo para comprender cuáles son los pasos necesarios a realizar en la construcción de un Almacén de Datos y para entender cuál es su funcionamiento una vez creado.

En cuanto a su modelo de persistencia, OWB crea un repositorio de metadatos en la BD, donde el usuario elige durante el período de instalación.

En este repositorio, OWB crea tablas, procedimientos, vistas, triggers, etc. para que la herramienta pueda funcionar correctamente.

Una vez creada la estructura del Almacenes de Dato, cuando llega la hora de desplegar cada una de sus partes (desde el entorno gráfico generalmente), se generan objetos de Oracle, principalmente procedimientos almacenados, que serán los elementos a los que se llame en realidad cuando se ejecuta un flujo de proceso.

El control de todo el proceso reside en los objetos de la Base de Datos, pero la información que se ofrece en el entorno visual corresponde a la documentación, es decir, a los **metadatos**.

Esta información, además de poder ser accedida mediante el entorno visual, también se puede extraer directamente de los objetos del repositorio, pero estos objetos están pensados básicamente para servir a la herramienta, no para que se les consulte directamente. Además, buscar información directamente en el repositorio puede ser bastante complicado. Esto suele hacerse únicamente cuando la interfaz gráfica no es suficiente.

La herramienta de OWB, con la que vamos a trabajar durante prácticamente todo el proyecto es el **Centro de Diseño**. Para acceder a él, como se explicó anteriormente, se debe haber creado un espacio de trabajo y un usuario de Base de Datos como propietario.

Una vez que hemos conseguido acceder a la aplicación, se podrán observar 3 paneles:

1) Explorador de Proyectos

Muestra la estructura del árbol de objetos que contiene cada uno de los proyectos que hemos creado.

Inicialmente crea un proyecto vacío llamado “MY_PROJECT”.

Dentro de este árbol de objetos, cabe destacar algunos como los siguientes:

- **Tablas externas,** encargadas de extraer los metadatos de los ficheros fuente y almacenarlos en una estructura que posteriormente permita trabajar con ellos de diferentes maneras. En consecuencia, tendremos una por cada fichero.
- **Tablas (relacionales),** generadas automáticamente tras la creación de un objeto. Se encargan de almacenar en tablas “reales” de Base de Datos toda la información que se va elaborando en la construcción del Almacén de Datos.
- **Dimensiones,** contendrá las Dimensiones del proyecto, que almacenarán la información relativa a cada uno de los temas a tratar en un Hecho.
- **Cubo,** contendrá el o los Cubos creados en el proyecto, que se encargarán de almacenar los Hechos del Almacén de Datos.
- **Correspondencias,** serán las encargadas de definir el proceso a seguir a la hora de ejecutar el proceso ETL. Indicará las tablas externas de donde tienen que extraer la información las Dimensiones y el Cubo, etc.

2) Explorador de Localizaciones

Muestra la estructura del árbol de objetos que contiene cada una de las localizaciones que hemos creado para dar soporte a los módulos de cada proyecto.

Las localizaciones actúan como almacenes para los metadatos generados en diferentes partes de un proyecto.

La creación de estos almacenes es necesaria a la hora de crear un flujo de trabajo, un esquema de usuario nuevo o un nuevo proyecto entre otros.

3) Explorador Global

Muestra la estructura del árbol de objetos global del sistema, pero para el propósito de este proyecto, únicamente se deberá conocer el funcionamiento de los esquemas de usuarios y sus roles en el nodo “Seguridad”.

A la hora de crear un usuario, o más bien añadir un esquema de usuario al proyecto, se puede realizar de dos maneras, seleccionando un usuario de los disponibles en la lista o creando un nuevo usuario, debiendo en tal caso tener privilegios de Administrador de Base de Datos.

Una vez que hemos creado la estructura del Almacén de Datos, debemos realizar el despliegue de los objetos para crear sus correspondientes instancias, necesarias para la futura ejecución. Estas operaciones de despliegue y ejecución las haremos desde una herramienta diferente llamada **“Centro de Control”**, la cual también será lanzada desde el Centro de Diseño (“Herramientas” > “Administrador del Centro de Control”).

2.5.2 Oracle XML DB

Oracle XML DB permite almacenar los datos XML en una columna de una tabla a través del tipo *XMLType*, crear tablas en función de la estructura del documento o bien guardar el propio documento en forma binaria en una columna de tipo CLOB.
Los aspectos más importantes a destacar de Oracle XML DB de cara a este PFC son:

- **XML Type**

Para proporcionar flexibilidad y versatilidad, Oracle XML DB está construido sobre la abstracción XML Type para almacenar, consultar, acceder, transformar y manipular datos XML.

XML Type es un tipo de dato nativo para XML. Proporciona métodos que permiten realizar operaciones como validaciones mediante un XML Schema o transformaciones XSL de contenido XML. Pero también es posible utilizar XML Type con cualquier otro tipo de dato. Por ejemplo, es posible utilizarlo cuando se desea hacer alguna de las siguientes operaciones:

- Crear una columna en una tabla relacional.
- Declarar variables PL/SQL.
- Definir y llamar a procedimientos y funciones PL/SQL.

➢ **Repositorio de Oracle XML DB**

El modelo relacional (tablas) es aceptado como un mecanismo efectivo para almacenar datos estructurados, pero no lo es tanto para los datos semiestructurados o no estructurados, como los documentos XML.

Por ejemplo, un libro no es fácilmente representable como un conjunto de filas en una tabla, sino que sería más natural representar un libro como una jerarquía *libro:capítulo:sección:párrafo*, y para representar tal jerarquía utilizar un conjunto de carpetas y subcarpetas.

El repositorio de Oracle XML DB es un componente de la BD de Oracle que ha sido optimizado para manejar datos XML como ficheros dentro de un sistema de ficheros. Este repositorio contiene ciertos recursos, que pueden ser tanto carpetas (directorios o contenedores) como ficheros, que contienen las siguientes propiedades:

- Está identificado por ruta y nombre.
- En su contenido puede haber datos XML, pero no necesariamente siempre.
- Contiene información (metadatos o propiedades) sobre los datos almacenados. Oracle XML DB utiliza esta información para manejar cada recurso.

- Puede también contener metadatos sobre el usuario, que a pesar de no formar parte del contenido, esta información puede quedar asociada.
- Contiene una lista asociada para controlar quién accede a cada recurso y para qué.

➢ **Registrar XML Schema en Oracle XML DB**

Los documentos creados externamente pueden ser introducidos en la base de datos de dos formas diferentes: bien mediante transferencia FTP al repositorio de Oracle o bien a través de la creación de una instancia de XML con una sentencia SQL. En ambos casos el documento XML debe llevar en la cabecera (en el atributo del espacio de nombres *xmlns*) la ruta del XML Schema contra el que se valida. Si el archivo XML no lleva ninguna referencia al esquema que sigue, el archivo es almacenado pero no incluido dentro de las estructuras internas del gestor. El repositorio funcionaría, en ese caso, como un depósito de ficheros.
El primer paso para la integración del contenido XML en la base de datos consiste en registrar un XML Schema que defina la estructura de los documentos XML que se quieren almacenar. Cuando se registra un XML Schema, se generan automáticamente los tipos (que heredan de XMLType) y las tablas necesarias para la gestión de los datos incluidos en las instancias del esquema registrado.

El proceso de registrar un XML Schema en la Base de Datos se realizará a través de la siguiente sentencia PL/SQL:

```
Begin

dbms_xmlschema.registerSchema('/home/OE/xsd/Fichero_Info.xsd',
xdbURIType('/home/OE/xsd/Fichero_Info.xsd'), TRUE, TRUE, FALSE, TRUE);

 end;
```

El procedimiento *registerSchema* pertenece al paquete *dbms_xmlschema,* utilizado para el manejo de XML Schemas. Los parámetros que admite son:

- la URL que identificará unívocamente el XML Schema a registrar,
- el contenido del XML Schema en sí

- y una serie de variables booleanas con diferentes opciones sobre la generación de objetos y de tablas.

Para indicarle al SGBD que tiene que obtener el contenido del XML Schema de una dirección concreta se utiliza una URI con la sentencia *xdbURIType*.

Si queremos borrar el esquema previamente registrado (así como las estructuras de tipos y tablas creadas internamente) deberemos utilizar una sentencia similar a la siguiente:

```
Begin
dbms_xmlschema.deleteSchema('/home/OE/xsd/Fichero_Info.xsd',
dbms_xmlschema.DELETE_CASCADE_FORCE);

end;
```

➢ **Gestión de documentos XML con XML DB**

Tras el registro de un XML Schema en XML DB de Oracle, el sistema está preparado para almacenar y gestionar documentos XML definidos bajo el XML Schema registrado. A continuación, se detallarán las operaciones básicas en la gestión de información (inserción, modificación, consulta y borrado) desde la perspectiva del repositorio como sistema de ficheros y mediante el uso de sentencias SQL.

▪ **Añadir un documento**

Como ya se ha comentado anteriormente, es posible añadir un nuevo documento XML al repositorio solamente copiándolo físicamente al directorio XML creado (vía FTP, por ejemplo). Dado que el documento XML tiene una referencia en su cabecera al esquema que sigue, el gestor de XML DB reconoce automáticamente que ese fichero es una instancia del XML Schema registrado, descomponiendo su contenido en las correspondientes tablas objeto-relacionales.
También es posible insertar información en las tablas creadas durante el registro del esquema con una tradicional sentencia *INSERT* de SQL. Si además se pretende obtener un documento XML físico almacenado en el

repositorio, se puede hacer mediante un procedimiento PL/SQL que trabaje con la instrucción *dbms_xdb.createResource*.

La estrategia seguida en este proyecto para la inserción en la BD XML ha sido a través de sentencias *INSERT* de SQL.

- **Editar y/o actualizar un documento XML**

Una vez que se ha añadido un documento XML, es posible manipular esta información. Existen dos métodos posibles: El primero emplea el protocolo FTP para llegar al fichero en cuestión y a continuación, editarlo con una herramienta que admita el procesamiento del lenguaje XML. Por otro lado, también es posible la modificación de los documentos XML mediante la utilización de la sentencia SQL "*update*" sobre las filas de las tablas que guardan la información de dichos documentos. En el siguiente ejemplo se puede observar una actualización en la que se modifica el precio de los productos (de 9.95€ a 10€).

```
UPDATE tabla_ventas t
SET value(t) = UPDATEXML (value(t),
 '/starpackage/dimensions/producto/precio()', '10')
WHERE EXTRACTVALUE(value(t),
            /starpackage/dimensions/producto/precio') = '9.95';
```

En este trabajo, el método elegido para la actualización de los documentos XML ha sido la utilización de la sentencia *SQL UPDATE*.

- **Borrado de documentos XML**

Al igual que en las anteriores operaciones, la operación de borrado puede llevarse a cabo sobre una instancia XML completa o desde sentencias SQL.
En el primero de los casos de borrado de un documento se puede realizar mismamente desde el Explorador de Windows. Accediendo al sitio de red que representa el repositorio se seleccionaría el archivo y se procedería al borrado del mismo como si de un archivo cualquiera se tratara. Es asimismo posible mediante un procedimiento PL/SQL que trabaje con la instrucción *dbms_xdb.deleteResource*.

Mediante sentencias SQL bastaría con lanzar una sentencia "*delete*" sobre la fila de la tabla deseada borrando de este modo el documento XML. En el siguiente ejemplo se eliminan los dependientes cuya edad es 18.

```
DELETE FROM tabla_ventas t

WHERE EXTRACTVALUE (value(t),
'/starpackage/dimensions/dependiente/edad')='18';
```

Nuevamente se ha elegido el uso de sentencias SQL para realizar el borrado de documentos XML en la BD XML implementada en este trabajo.

- **Consulta de datos XML**

El acceso a la información almacenada en XML se realiza a través de consultas *SELECT*, combinadas con métodos proporcionados por el tipo XMLType que utilizan XPath como forma de navegar por el contenido XML. Cuando se emplea una expresión XPath, lo que hace el motor de XML DB es reescribir la consulta, traduciendo la expresión XPath del operador SQL XML a una consulta XML convencional sobre las estructuras de datos objeto-relacionales que hay debajo del contenido XML. En el siguiente ejemplo se utiliza el método *extractValue()* para obtener el valor de un nodo tipo texto, la sentencia extrae todos los nombres de los dependientes almacenados en la tabla ventas.

```
SELECT EXTRACTVALUE(value(t),

'/starpackage/dimensions/dependiente/nombre') as nombre FROM tabla_ventas t;
```

3. Estudio y uso de una metodología para el desarrollo de Almacenes de Datos Relacionales

En este capítulo se expondrá un breve resumen sobre la metodología descrita anteriormente para desarrollar un Almacén de Datos Relacional y se explicará detalladamente cómo fue aplicada para este caso de estudio.

Según se ha visto en el capítulo anterior en la metodología para desarrollar un Almacén de Datos Relacional, propuesta por Mazón y Trujillo (2007), partimos de un modelo multidimensional (PIM) que representaremos con UML extendido. A partir de este PIM, aplicando las reglas de transformación que se proponen en la metodología, se obtendrá el PSM Relacional.

3.1 Caso de estudio

Siguiendo esta metodología se comenzó realizando el PIM del problema a resolver sobre “Ventas”. Este modelo, al ser independiente de la plataforma o tecnología, servirá para todo el desarrollo del caso de estudio, es decir, tanto para el Almacén de Datos Relacional como para el Almacén de Datos XML.

Una vez que se haya realizado el PIM, aplicando las correspondientes reglas de transformación marcadas en esta metodología, se diseñará el PSM Relacional, como veremos más adelante.

- **PIM Multidimensional**

En primer lugar se ha realizado el PIM para el problema a resolver en este caso de estudio, como se puede apreciar en la figura 9.

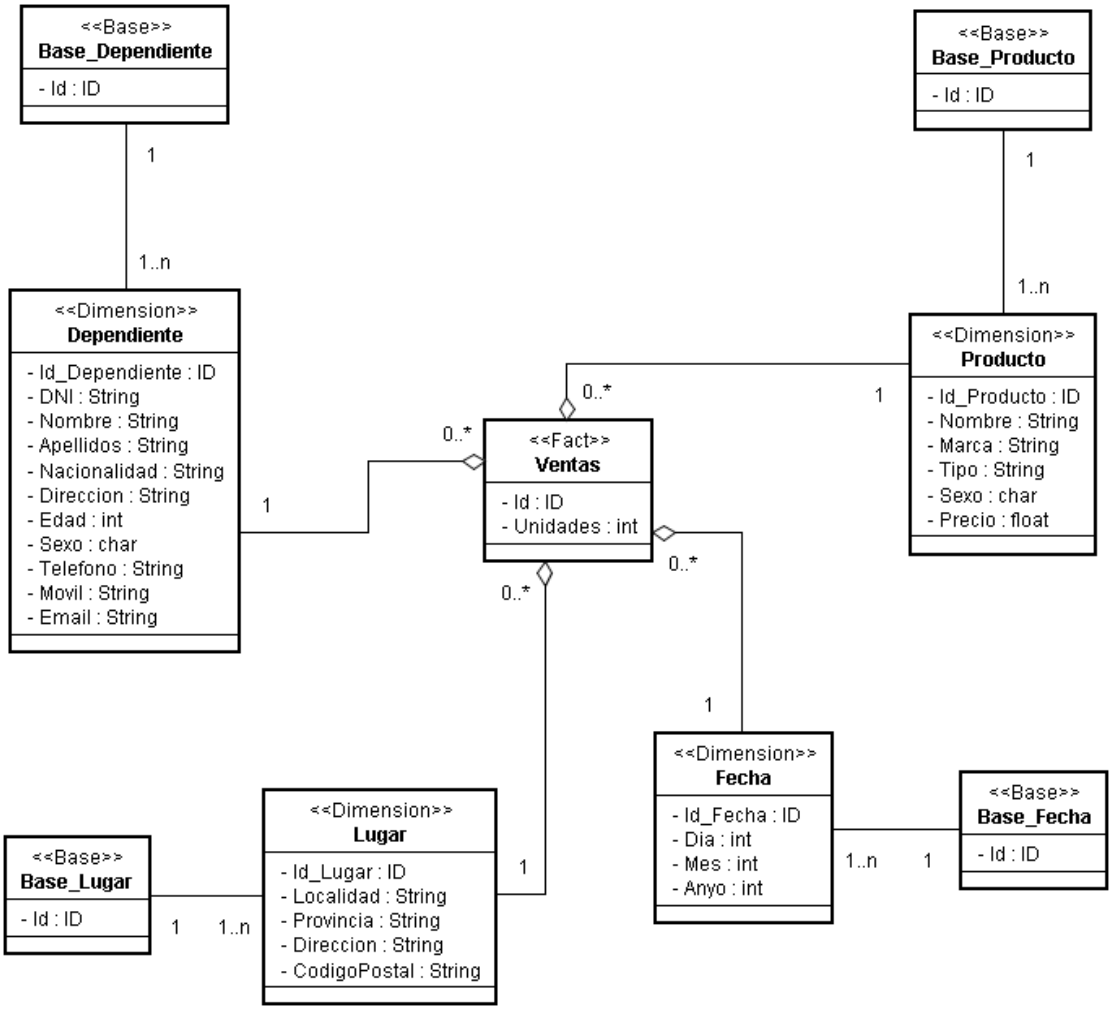

Figura 9: *PIM del caso de estudio*

Como se puede observar en el modelo, en este caso de estudio se pretende recoger la información relevante de una venta en función de la fecha en que tuvo lugar, el dependiente que la realizó, el lugar donde se vendió y el propio producto que fue vendido.

Para ello se han diseñado una serie de tablas relacionales en notación UML representando los Hechos, las Dimensiones y las bases necesarias.

De esta manera, se dispone de:

- Un Hecho (Fact) denominado **Ventas**, que contiene los siguientes elementos:
  - Un atributo “ID”, que actuará como identificador de cada Hecho.
  - Un atributo “Unidades”, que servirá para recoger el número de unidades vendidas del producto en cuestión.
  - Una relación de agregación con la Dimensión “Dependiente”, donde almacenará la información correspondiente al dependiente que realizó la venta.
  - Una relación de agregación con la Dimensión “Producto”, donde almacenará la información correspondiente al producto que fue vendido.
  - Una relación de agregación con la Dimensión “Fecha”, donde almacenará la información correspondiente a la fecha en que se vendió.
  - Una relación de agregación con la Dimensión “Lugar”, donde almacenará la información correspondiente al lugar donde se vendió.
- Una Dimensión denominada **Dependiente**, que contiene los siguientes elementos:
  - Un atributo “Id_Dependiente”, que actuará como identificador de cada dependiente.
  - Una serie de atributos para recoger DNI, nombre, apellidos, nacionalidad, dirección, edad, sexo, teléfono fijo, teléfono móvil y e-mail del dependiente.
  - Una relación de agregación con la tabla de Hechos “Ventas”.
  - Una relación de 1 a N con la tabla “Base_Dependiente”, donde la Dimensión Dependiente únicamente se puede relacionar con una Base_Dependiente.

- Una Dimensión denominada **Producto**, que contiene los siguientes elementos:
  - Un atributo “Id_Producto”, que actuará como identificador de cada producto.
  - Una serie de atributos para recoger nombre, marca, tipo, sexo y precio del producto.
  - Una relación de agregación con la tabla de Hechos “Ventas”.
  - Una relación de 1 a N con la tabla “Base_Producto”, donde la Dimensión Producto únicamente se puede relacionar con una Base_Producto.
- Una Dimensión denominada **Lugar**, que contiene los siguientes elementos:
  - Un atributo “Id_Lugar”, que actuará como identificador de cada punto de venta.
  - Una serie de atributos para recoger la dirección, localidad, provincia y código postal del lugar.
  - Una relación de agregación con la tabla de Hechos “Ventas”.
  - Una relación de 1 a N con la tabla “Base_Lugar”, donde la Dimensión Lugar únicamente se puede relacionar con una Base_Lugar.
- Una Dimensión denominada **Fecha**, que contiene los siguientes elementos:
  - Un atributo “Id_Fecha”, que actuará como identificador de cada fecha.
  - Una serie de atributos para recoger día, mes y año de la fecha indicada.
  - Una relación de agregación con la tabla de Hechos “Ventas”.
  - Una relación de 1 a N con la tabla “Base_Fecha”, donde la Dimensión Fecha únicamente se puede relacionar con una Base_Fecha.

- Una Base denominada **Base_Dependiente**, que contiene los siguientes elementos:
  - Un atributo “Id”, que actuará como identificador de la base.
  - Una relación de 1 a N con la Dimensión “Dependiente”, donde la tabla Base_Dependiente se puede relacionar con N dependientes.
- Una Base denominada **Base_Producto**, que contiene los siguientes elementos:
  - Un atributo “Id”, que actuará como identificador de la base.
  - Una relación de 1 a N con la Dimensión “Producto”, donde la tabla Base_Producto se puede relacionar con N productos.
- Una Base denominada **Base_Lugar**, que contiene los siguientes elementos:
  - Un atributo “Id”, que actuará como identificador de la base.
  - Una relación de 1 a N con la Dimensión “Lugar”, donde la tabla Base_Lugar se puede relacionar con N lugares.
- Una Base denominada **Base_Fecha**, que contiene los siguientes elementos:
  - Un atributo “Id”, que actuará como identificador de la base.
  - Una relación de 1 a N con la Dimensión “Fecha”, donde la tabla Base_Fecha se puede relacionar con N fechas.

- **PSM Relacional**

Tras haber realizado el PIM, se va a desarrollar el PSM Relacional para el modelo de almacenamiento del Almacén de Datos Relacional del caso de estudio, como se observa en la figura 10. Este modelo ha sido diseñado mediante una arquitectura en estrella en la que las Dimensiones están compuestas por un único nivel.

El proceso que se ha llevado a cabo para desarrollar el PSM Relacional en base al PIM anterior ha seguido una serie de reglas de transformación indicadas en la metodología que se está siguiendo.

Dentro de las reglas de transformación ofrecidas en la tabla de la tabla 1 y, teniendo en cuenta que en este PIM todas las Dimensiones sólo poseen un único nivel en su jerarquía, únicamente han sido empleadas las siguientes:

1. La tabla de Hechos tendrá como clave primaria un atributo ID.
2. La tabla de Hechos dispondrá de una clave ajena hacia cada una de las Dimensiones que la compone, representando su anterior relación de agregación con una relación de 1 a N.
3. Cada tabla de Dimensión dispondrá de un atributo ID como clave primaria.
4. Cada tabla de Dimensión contendrá una clave ajena hacia su correspondiente tabla Base, relacionándose con ella mediante una relación de 1 a N.

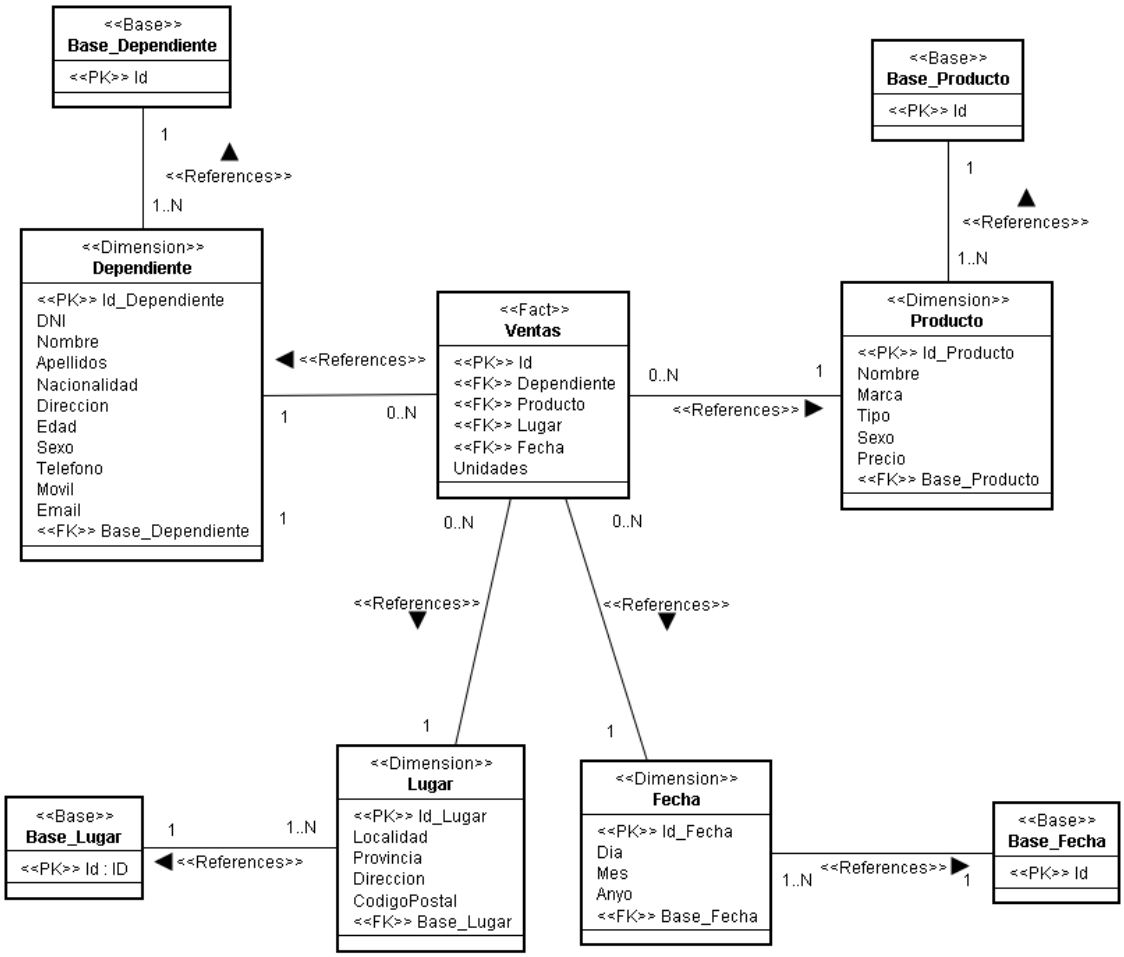

Figura 10: *PSM Relacional en estrella del caso de estudio*

- **Implementación en OWB**

A continuación, se va a explicar de manera detallada en qué consisten cada uno de los pasos a realizar para implementar un Almacén de Datos Relacional en la herramienta OWB.

1. Creación de ficheros planos como fuente de información

En primer lugar, será necesario contar con los ficheros que contienen los datos a extraer localizados en alguna carpeta accesible. Una opción para dichos ficheros puede ser en formato .csv (excell) ya que hace una distinción entre los campos tomando como elemento separador la coma. Se aconseja que la primera línea del fichero contenga el nombre de los atributos y las siguientes por los datos.

Una vez que tenemos creados estos ficheros, desde el OWB debemos indicar que tenemos un archivo con datos pendientes de ser extraídos. Para ello, tendremos que crear un nuevo archivo en OWB seleccionando el directorio donde se encuentran nuestros ficheros fuente e indicando cuáles de estos se van a tener en cuenta para la extracción.

A continuación, OWB nos pide que creemos una localización origen para alojar los metadatos resultantes de la extracción de los ficheros fuente y poder trabajar con ellos en el futuro.

Antes de finalizar, debemos asegurarnos de tener seleccionada la opción de "Importar después de finalizar", esto abrirá automáticamente el Asistente de Importación de Metadatos, que nos indicará los pasos necesarios para completar el proceso de manera adecuada.

Realización:

1. En el panel "Explorador de Proyectos" expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos "Bases de Datos".
3. Click derecho sobre "Archivos".
4. Seleccionamos "Nuevo".

2. Importación de los metadatos del fichero plano

Una vez que se ha abierto el Asistente de Importación de Metadatos, nos mostrará que, aquellos ficheros que seleccionamos como contenedores de los datos que necesitamos no tienen creado el enlace requerido (aspa roja), por lo que, de uno en uno, tendremos que ir realizando los pasos necesarios para la correcta importación de los metadatos.

Por lo tanto, seleccionamos el primer fichero y pulsamos en el botón "Ejemplo" para comenzar. En este proceso, ayudaremos a OWB a saber cómo realizar la importación mediante indicaciones como cuáles son los delimitadores entre los atributos (coma) y las tuplas (retorno de carro). Además, podremos modificar el nombre y tipo de los atributos que contiene, etc.

Como apunte, conviene remarcar que, si en el fichero fuente hemos introducido (como se recomendó) en la primera línea el nombre de los campos, en este punto debemos marcar la opción “Utilizar la primera línea como nombre de campo”.

Repetiremos este proceso con el resto de los ficheros fuente.

Realización: En este caso no habrá nada que hacer, puesto que el Asistente de Importación de Metadatos se abre automáticamente.

3. Definición de un almacén destino

Para poder diseñar nuestro módulo almacén destino, debemos seguir los siguientes pasos:

a. Crear un esquema de usuario destino

En primer lugar crearemos un nuevo esquema de usuario de BD con el que trabajaremos en este módulo. Para ello deberemos introducir el nombre de usuario y contraseña del administrador de la BD y los datos del nuevo usuario.

Realización:

1. En el panel “Explorador Global” expandimos “Seguridad”
2. Click derecho sobre “Usuarios”.
3. En el cuadro de diálogo que aparece seleccionamos “Crear nuevo usuario”.
4. Introducimos los datos del nuevo usuario y del Administrador de la Base de Datos.

b. Crear una localización para el esquema destino

Crearemos a continuación una localización o ubicación donde se almacenarán los objetos creados en el módulo.

Para la creación de la nueva localización indicaremos el nombre de la misma, los datos del usuario de BD que queremos asignar como propietario de la localización (los introducidos anteriormente), los datos de la conexión (localhost:1521:orcl) y el esquema destino, en el que introduciremos de nuevo el nombre del usuario propietario, para que no haya confusiones.

Antes de aceptar, debemos probar la conexión.

Realización:

1. En el panel “Explorador de Conexiones” expandimos “Bases de Datos”.
2. Click derecho sobre “Oracle”.
3. Seleccionamos “Nuevo”.

c. Crear un módulo destino

En este punto, lo que vamos a hacer va a ser crear un módulo destino vacío (sin importar ningún objeto) asociado a un esquema destino vacío (no contiene ningún objeto).

Para la creación de un módulo tendremos que introducir, en primer lugar, el nombre que le queremos dar y seleccionar que va a ser un almacén destino y no una fuente de datos.

Posteriormente debemos especificar cuál va a ser nuestra localización destino, es decir, la ubicación donde queremos que se almacenen los objetos que se creen en el módulo.

Al seleccionar la localización deseada (que debería ser la que acabamos de crear, puesto que ésta era su finalidad), OWB extraerá automáticamente la información relativa al usuario que tiene asignado y todos los datos referentes a la conexión que hemos definido en la creación de dicha localización.

Finalizamos con la opción “Importar después de finalizar” desactivado.

Realización:

1. En el panel “Explorador de Proyectos” expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos “Bases de Datos”.
3. Click derecho sobre “Oracle”.
4. Seleccionamos “Nuevo”.

4. Creación de tablas externas

A continuación crearemos y configuraremos las tablas externas, que se encargarán de extraer y almacenar los metadatos de los ficheros planos en una estructura que nos permita posteriormente trabajar con ellos. En consecuencia, tendremos una por cada fichero.

a. Creación de una tabla externa

Para la creación de una tabla externa por lo tanto, una vez que le asignamos un nombre, tendremos que indicar cuál va a ser el fichero que contiene los metadatos requeridos y cuál la localización donde éstos están almacenados, que será la que creamos anteriormente como fuente de datos (esta localización debería ser la única que aparezca en el menú).

Realización:

1. En el panel “Explorador de Proyectos” expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos “Bases de Datos”.
3. Expandimos “Oracle”.
4. Expandimos nuestro módulo.
5. Click derecho sobre “Tablas externas”.
6. Seleccionamos “Nuevo”.

b. Configuración de una tabla externa

Una vez creadas las tablas externas, ahora tenemos que configurarlas para crear el enlace que las una al fichero correspondiente, es decir, aquel del que deben extraer los metadatos. Para ello también tendremos que indicar la localización donde están alojados (localización origen).

Nota: Es importante truncar todas las tablas externas que creemos para evitar warnings y errores futuros por los tipos de datos. Para ello, una vez que estamos en la ventana de configuración, como edición de campo asignamos en Recorte “Ambos”, esto lo que hará será eliminar los blancos de la izquierda y derecha que OWB genera automáticamente cuando no se rellena al completo un tipo de dato.

Realización:

1. En el panel “Explorador de Proyectos” expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos “Bases de Datos”.
3. Expandimos “Oracle”.
4. Expandimos nuestro módulo.
5. Expandimos “Tablas externas”.
6. Click derecho sobre la tabla externa.
7. Seleccionamos “Configurar”.
8. En el cuadro de diálogo desplegado, click derecho sobre “Ficheros de datos” y “Crear”.
9. Aceptamos el nombre por defecto para el fichero de datos.
10. Introducir el nombre del fichero plano al que va a hacer referencia.
11. Seleccionamos la localización (será la misma que le asignamos en su creación).

5. Diseño y creación de Dimensiones

El motivo por el que crearemos las Dimensiones antes que el Cubo (almacenamiento de Hechos) es porque este último, para su creación, necesita indicar las referencias a sus correspondientes Dimensiones, y obviamente deben existir.

Suele ser habitual a la hora de crear un Almacén de Datos crear una Dimensión de tiempo, para poder recoger así la fecha de cada uno de los Hechos almacenados (ejemplo: ¿cuál ha sido el gasto total en este trimestre en comparación con el año pasado?). Debido a esto, Oracle Warehouse Builder facilita un Asistente específico para la creación de Dimensiones temporales.

Nosotros crearemos una Dimensión para almacenar las fechas en las que se han realizado las ventas, pero no la crearemos usando el Asistente para Creación de Dimensiones Temporales, sino el Asistente común. El motivo de hacerlo de esta

manera no es otro que el de la simplicidad, dado que éste es un ejemplo sencillo con el que introducirnos en los entornos de ALMACÉN DE DATOS, almacenaremos las fechas en una única Dimensión de un solo nivel mediante una cadena de caracteres (Ej: 11-Dic-1985).

Como apunte para los siguientes pasos, conviene aclarar que una Dimensión, desde el punto de vista relacional, puede estar constituida por un único nivel o varios, distribuidos en este último caso jerárquicamente. Para este ejemplo, como se comentó, por simplicidad únicamente haremos Dimensiones de un único nivel, por lo que no tendremos jerarquías.

Nota: En cada nivel de la Dimensión, para poder desarrollar correctamente el flujo de procesos, siempre hay que tener al menos un atributo de "Negocio" que represente a dicho nivel y un atributo "Sustituto" de tipo numérico. Además, si únicamente se desea tener un nivel en la Dimensión, hay que eliminar la Jerarquía por defecto que te asigna OWB automáticamente.

a. Creación de una Dimensión general

En primer lugar tendremos que darle un nombre a la Dimensión.

Una vez asignado el nombre, tendremos que decidir si nuestra Dimensión será Relacional (ROLAP), pensada para grandes almacenamientos de datos unido a una gran velocidad de consulta, o Multidimensional (MOLAP), pensada principalmente como almacenaje de un conjunto de datos para su análisis. En nuestro caso elegiremos ROLAP.

Seguidamente, debemos especificar cuáles serán los atributos de nuestra Dimensión y sus propiedades, cumpliendo con el requisito anteriormente mencionado de tener al menos un atributo de Negocio y uno como Sustituto.

Ahora debemos indicar cuáles van a ser los niveles que contiene nuestra Dimensión (en caso de que así sea) y la jerarquía que se desea para ellos. Además tendremos que decidir qué atributos de Dimensión queremos aplicar a cada nivel.

En este caso pondremos un único nivel y, será muy importante que, para no tener problemas futuros, eliminemos la jerarquía como se ha comentado anteriormente.

A continuación debemos indicar si queremos que nuestra Dimensión almacene el histórico de los cambios realizados sobre los datos que contiene o no. Por ejemplo, si se trata de una Dimensión lentamente cambiante, probablemente sí que interese guardar este tipo de información. En nuestro caso elegiremos no guardar histórico.

Por último, una vez que finalizamos la operación, debemos observar un dato importante, y es que de forma automática, OWB nos ha creado 3 objetos:

- El objeto *Dimensión* que pretendíamos.
- Un objeto *secuencia*, que se encargará durante el flujo de proceso de rellenar los datos correspondientes al identificador sustituto de la Dimensión.
- Un objeto *tabla* para soportar la implementación relacional de la Dimensión temporal almacenando de forma física los datos.

Realización:

1. En el panel “Explorador de Proyectos” expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos “Bases de Datos”.
3. Expandimos “Oracle”.
4. Expandimos nuestro módulo.
5. Click derecho sobre “Dimensiones”.
6. Seleccionamos “Nuevo + Usando Asistente”.
7. Para ver el resto de los objetos creados tendremos que expandir dentro del mismo módulo “Tablas” y “Secuencias”.

b. Visualización gráfica mediante Editor de cartografías

Si se desea en cualquier momento ver de forma gráfica de qué está constituida una Dimensión siempre se puede abrir el Editor de Cartografías y observarlo.

Esta herramienta está disponible también para el resto de los objetos.

Realización:

1. En el panel “Explorador de Proyectos” expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos “Bases de Datos”.
3. Expandimos “Oracle”.
4. Expandimos nuestro módulo.
5. Expandimos “Dimensiones”.
6. Doble click sobre la Dimensión que queremos visualizar.
7. En el Editor de Cartografía seleccionamos en el menú “Ver” la opción “Diseño Automático” para poder visualizar el mapa de la Dimensión de forma completa y ampliarla.

6. Creación y diseño de un Cubo

Para la creación de un Cubo, debemos introducir en primer lugar el nombre que deseamos que tenga y, al igual que en las Dimensiones, si va a ser Relacional (ROLAP) o Multidimensional (MOLAP). En nuestro caso será ROLAP.

En la siguiente ventana, el sistema nos pedirá que seleccionemos cuáles serán las Dimensiones a las que nuestro Cubo referenciará. Para ayudarnos, nos mostrará todas las Dimensiones que en ese momento estén creadas, independientemente de si son o no del proyecto actual.

Una vez que hemos seleccionado las Dimensiones, OWB nos pide que introduzcamos los valores del campo “Medidas” de nuestro Cubo. Introducimos el nombre (en nuestro ejemplo, únicamente mediremos las unidades vendidas) y dejamos el tipo por defecto, que será NUMBER.

Por último, si queremos ver cómo ha quedado nuestro Cubo, podemos abrir el Editor de Cartografía de la misma manera que lo hicimos para las Dimensiones.

Realización:

1. En el panel “Explorador de Proyectos” expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos “Bases de Datos”.
3. Expandimos “Oracle”.
4. Expandimos nuestro módulo.
5. Click derecho sobre “Cubos”.
6. Seleccionamos “Nuevo + Usando Asistente”.

7. Diseño de una correspondencia entre una tabla externa y una Dimensión

Como primer paso, tenemos que asignar un nombre a la correspondencia. Cuando aceptamos, se desplegará automáticamente el Editor de Cartografía. En él, tenemos que hacer una serie de pasos, en primer lugar, introducir el objeto “Dimensión” referente a la Dimensión en cuestión. Para ello, desde la ventana “Explorador” (si no está visible se puede mostrar desde el menú “Ventana”). Una vez seleccionada aparecerá en el lienzo.

Seguidamente y repitiendo el mismo proceso, abrimos la tabla relacional que contiene los datos necesarios para la Dimensión.

Nota: Si pinchamos en  nos mostrará la correspondencia a su tamaño predeterminado. Para expandir un objeto pincharemos en  .

Posteriormente tenemos que realizar las conexiones entre la fuente origen (tabla externa) y el destino (tabla relacional). Estas conexiones, las realizaremos pinchando y arrastrando desde los campos de la tabla externa hacia los correspondientes de la tabla relacional. De esta forma, especificamos gráficamente el flujo de datos entre ambas.

En último lugar, tendremos que generar el código de la correspondencia, que en definitiva, es el objetivo de todo este desarrollo. Para generarlo tenemos que irnos al menú “Correspondencias” y seleccionar “Generar”. Una vez generado, se mostrará por pantalla el código obtenido.

Se repetirá el mismo proceso para cada una de las 3 Dimensiones restantes.

Realización:

1. En el panel “Explorador de Proyectos” expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos “Bases de Datos”.
3. Expandimos “Oracle”.
4. Expandimos nuestro módulo.
5. Click derecho sobre “Correspondencias”.
6. Seleccionamos “Nuevo”.
7. Nota: El proceso de realización del resto de operaciones necesarias ya han sido descritas anteriormente.

8. Diseño de una correspondencia entre una tabla externa y un Cubo

En este caso, vamos a crear la correspondencia entre una tabla externa y un Cubo (almacenará los Hechos del almacén).

El proceso de creación de esta correspondencia es muy parecido al anterior, se introduce el nombre que va a tener como primer paso y se despliega de forma automática el Editor de Cartografía.

A continuación debemos poner la tabla externa en cuestión y el Cubo en el lienzo. El siguiente paso es definir las conexiones entre ambos objetos, que también se realizará de manera semejante al caso de las Dimensiones, como se puede apreciar en la figura 11.

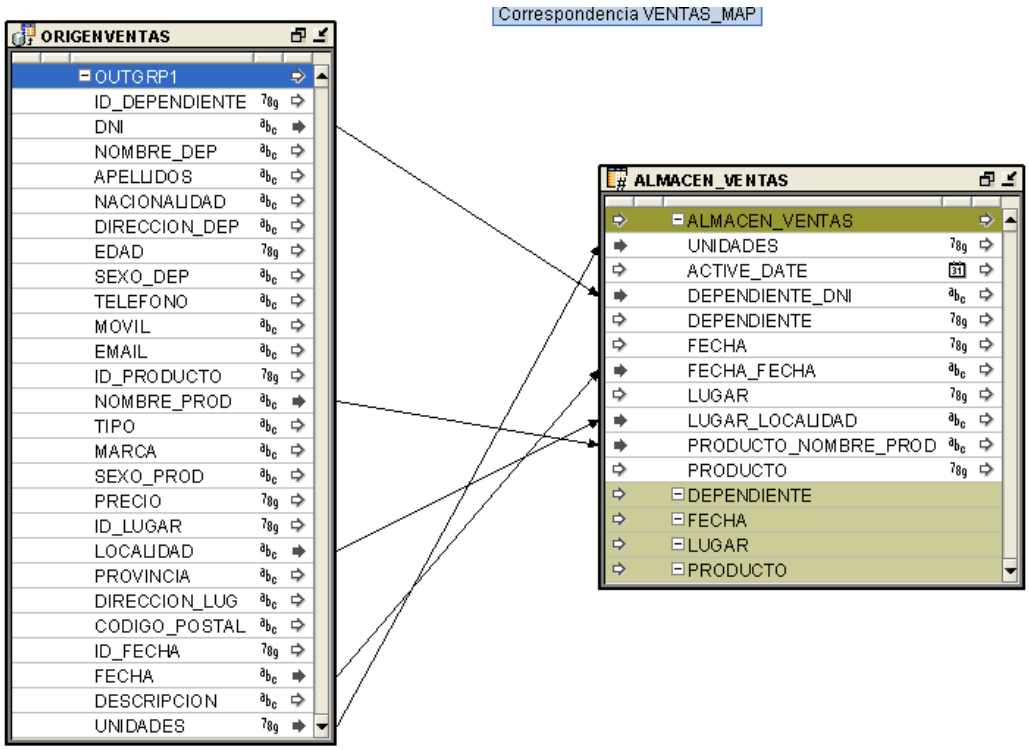

Figura 11: *Representación de las conexiones en la correspondencia entre una tabla externa y una Dimensión.*

En último lugar, tendremos que generar el código de la correspondencia, que en definitiva, es el objetivo de todo este desarrollo. Para generarlo tenemos que irnos al menú “Correspondencias” y seleccionar “Generar”. Una vez generado, se mostrará por pantalla el código obtenido.

Realización:

1. En el panel “Explorador de Proyectos” expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos “Bases de Datos”.
3. Expandimos “Oracle”.
4. Expandimos nuestro módulo.
5. Click derecho sobre “Correspondencias”.
6. Seleccionamos “Nuevo”.
7. Nota: El proceso de realización del resto de operaciones necesarias ya han sido descritas anteriormente.

9. Diseño del flujo de proceso ETL

a. Registrar el usuario del flujo de trabajo (Oracle Workflow)

En primer lugar, para poder diseñar un flujo de proceso, tenemos que conceder funciones específicas del flujo de trabajo de Oracle Workflow (OWF) a un usuario, en nuestro caso, el que creamos en el período de configuración de OWB, es decir, “owf_mgr” para asignarle privilegios y que así pueda ejecutar un flujo de proceso en el Centro de Control.

Si no fue añadido en su momento, se puede añadir expandiendo el nodo “Seguridad” del panel “Explorador Global” y haciendo click derecho sobre “Usuarios”, seleccionando a continuación “Nuevo”.

Cuando el sistema ofrezca la lista de usuarios disponibles, seleccionar “OWF_MGR”.

b. Diseño de un flujo de proceso ETL

Antes de nada, debemos crearnos un módulo de flujo de proceso, que contendrá en su interior un paquete de flujo de proceso y el propio objeto del flujo de proceso.

Para la creación del módulo de flujo de proceso tendremos que, por un lado asignarle un nombre, y por otro, crearemos una localización para el flujo de proceso de Oracle Workflow para especificar dónde se desea desplegar dicho flujo. En la nueva localización debemos especificar como password “owf_mgr” y como esquema “OWF_MGR”.

Inmediatamente después de crear el módulo, se desplegarán de manera automática dos cuadros de diálogo, uno para que le demos un nombre al paquete del flujo de proceso y otro para el propio flujo de proceso.

A continuación se abrirá el Editor de Procesos con sólo dos actividades iniciales en el lienzo, la de “Comienzo” (Start) y la de “Final sin errores”.

Nosotros vamos a empezar a añadir las siguientes actividades al lienzo para conseguir construir el flujo de proceso:

- "Fork" (Tenedor): Lo arrastraremos desde el cuadro "Paleta" y nos ayudará para ofrecer la posibilidad de lanzar varias actividades de manera concurrente.
- "Correspondencias de las Dimensiones": Las arrastraremos desde el cuadro de "Explorador", asegurándonos de que estamos en la pestaña "Objetos Disponibles", expandiendo nuestro módulo, las correspondencias que queremos que intervengan en nuestro flujo de proceso.
- **Nota:** Si nos da el error "No se puede determinar si es una correspondencia PL/SQL o SQLLDR", tendremos que irnos al Centro de Diseño, hacer click derecho sobre la correspondencia, darle a "Configurar" y seleccionar "PL/SQL".
- "And": Lo arrastraremos desde el cuadro "Paleta".
- "Correspondencia del Cubo": La arrastraremos del mismo modo que lo hicimos en las correspondencias de Dimensión.
- "Final con errores": Lo arrastraremos desde el cuadro "Paleta".
- "Final con advertencias": Lo arrastraremos desde el cuadro "Paleta".

Lo que haremos ahora en el lienzo, será describir las transiciones para indicar la secuencia y las condiciones en las que queremos que las actividades se desarrollen en el flujo de proceso.

Estas transiciones las llevaremos a cabo pinchando y arrastrando desde una actividad a otra, quedando el flujo de proceso como se puede observar en la figura 12.

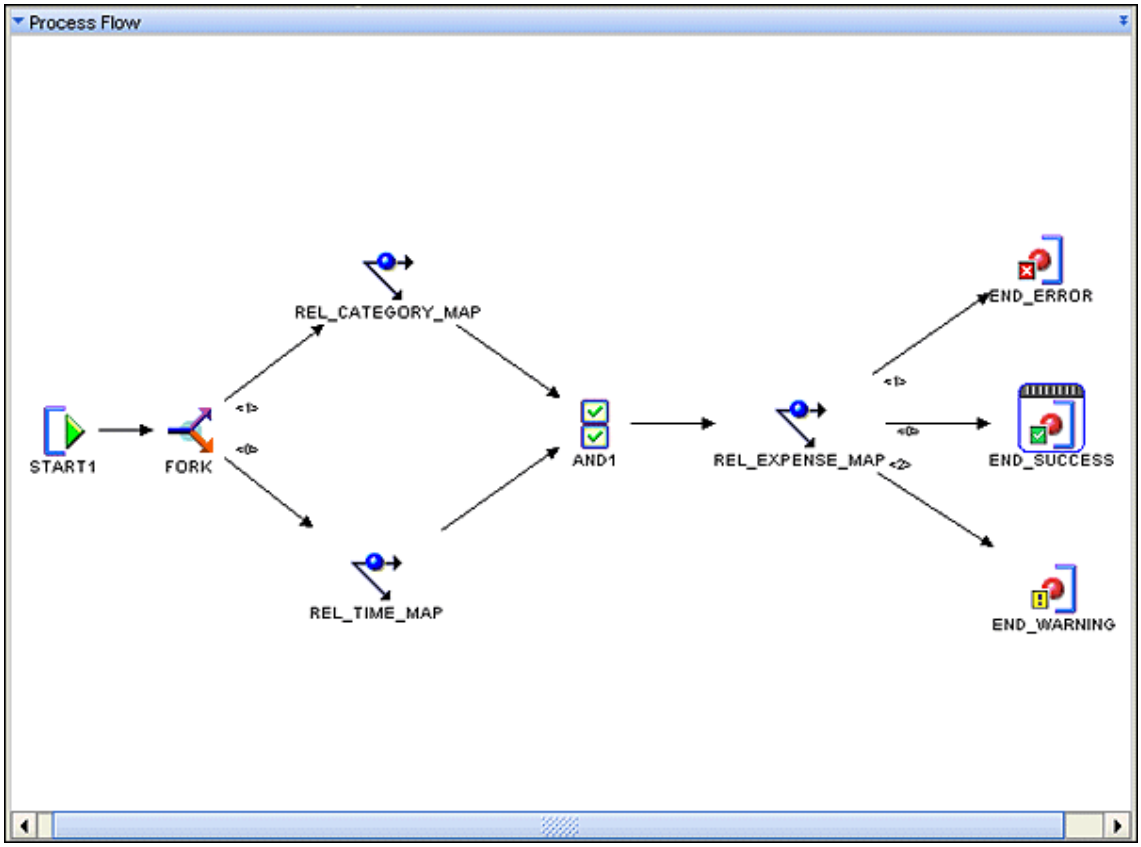

Figura 12: *Flujo de proceso ETL*

Un detalle importante que no debemos olvidar una vez que tenemos construido nuestro flujo de proceso es que, debemos configurar las transiciones entre la correspondencia del Cubo y el conjunto de actividades finales.

Para configurarlas, debemos hacer click sobre ellas e irnos al menú “Detalles del objeto” y hacer click sobre . En el cuadro de diálogo que aparece pinchamos en la opción “Condiciones Enumeradas” y seleccionamos la correspondiente para cada tipo de transición (correcto, con errores o con advertencias).

En último lugar, debemos generar el código correspondiente al flujo de proceso que acabamos de diseñar. Para ello nos vamos al menú “Flujo de Proceso” y seleccionamos “Generar”.

Realización:

1. En el panel “Explorador de Proyectos” expandimos el nodo del proyecto Oracle en el que estamos trabajando.
2. Expandimos “Flujos de Procesos”.

3. Click derecho sobre “Módulos de Flujos de Proceso”.
4. Seleccionamos “Nuevo”.
5. Nota: El proceso de realización del resto de operaciones necesarias ya han sido descritas anteriormente.

10. Despliegue de una tabla externa

Conviene mencionar antes de nada que, en todas las operaciones de despliegue de objetos, el procedimiento es exactamente el mismo, y por lo tanto, en todas ellas se utiliza la misma herramienta de trabajo, que será el “Administrador del Centro de Control”. Para lanzarlo lo haremos desde el menú “Herramientas”.

Este Administrador, se divide en 3 paneles:

- **Objetos:** Desde donde se puede acceder a todos los objetos de diseño a través de su localización.
- **Detalles de objetos:** donde se muestra el estado de cada objeto y se le permite al usuario modificarlo.
- **Trabajos del Centro de Control:** donde se muestra el resultado de aplicar las operaciones oportunas a los objetos seleccionados (resultados de despliegue), o también llamado trabajo.

Para la implementación o despliegue de una tabla externa, una vez que estamos en el Administrador del Centro de Control, dentro del panel “Objetos” debemos expandir la localización donde se encuentren y hacer un click sobre el nodo “Tablas Externas” para seleccionar todas. Tras hacer esto, en el panel “Detalles de Objetos” debemos pinchar sobre el botón “Acciones por defecto” y observamos cómo han cambiado su acción a “Crear”. Inmediatamente después, para llevar a cabo su despliegue por fin, hacemos click sobre el icono  . Los resultados obtenidos los podemos observar en el panel “Trabajos del Centro de Control”.

Como éste será el primer despliegue que se haga en esta localización destino, el sistema le pedirá que la registre. En el cuadro de diálogo que aparece, tras

comprobar todos los parámetros, antes de aceptar comprobamos la conexión, si todo está correcto, finalizamos la operación.

Nota: Este apunte es importante para todas las implementaciones que realicemos, no sólo para las tablas externas. Si queremos ver el código que se ha generado tras el despliegue, debemos hacer doble click en el panel “Trabajos del Centro de Control” sobre el trabajo deseado.

Esta acción abrirá el cuadro de diálogo “Detalles de Trabajo”, y en él, debemos asegurarnos de tener seleccionado el nodo “Tablas Externas” (o el correspondiente en cada caso) y hacer click sobre la pestaña “Archivo de comandos” (Script). A continuación seleccionamos el trabajo deseado y pinchamos en “Ver Código”.

11. Despliegue de un Cubo y sus Dimensiones

Antes de poder implementar el Cubo y sus Dimensiones, es necesario implementar los objetos “Secuencia” y “Tablas” que éstos usan.

Otro paso importante que tendremos que dar en este punto es configurar desde el Centro de Diseño todas las Dimensiones y el cubo para que desplieguen sus objetos como “Sólo Catálogo”.

Como expliqué anteriormente, el proceso de despliegue de un objeto es el mismo que para otro, independientemente de lo que represente.

12. Despliegue de correspondencias

Realización del proceso de despliegue anteriormente descrito para las correspondencias de nuestro ejemplo.

Nota: Si nos aparece el error “No se pueden desplegar las correspondencias PL/SQL en el esquema destino”, es porque en la instancia actual del Centro de Control no se encuentra registrado nuestro módulo. Para solucionarlo, hacer click derecho sobre él y “Registrar”.

13. Despliegue del flujo de proceso ETL

Antes de poder ejecutar el flujo de proceso para probar el Almacén de Datos, es necesario desplegarlo.

Para desplegar un flujo de proceso, una vez que estamos en el Centro de Control, como en los demás casos, debemos expandir en el árbol de objetos nuestro módulo donde se encuentra el flujo de proceso que hemos creado.

Una vez expandido, veremos nuestro paquete de flujo de proceso (por defecto llamado “PK”). Seleccionamos dicho paquete y, en el panel “Detalles de Objetos” debemos pinchar sobre el botón “Acciones por defecto” y observamos cómo han cambiado su acción a “Crear”. Inmediatamente después, para llevar a cabo su despliegue por fin, pinchamos sobre el icono  .

Los resultados obtenidos los podemos observar en el panel “Trabajos del Centro de Control”.

14. Cargar el almacén destino mediante flujos de proceso ETL

Una vez que todos los objetos han sido desplegados, incluido el flujo de proceso, llega el momento de ejecutar dicho flujo con el fin de que los datos sean almacenados en el almacén destino.

Para ello, expandimos de la misma manera que en el paso anterior hasta llegar al paquete de nuestro flujo de proceso (PK).

Bajo este paquete, se encuentra ya nuestro flujo de proceso, lo seleccionamos y pinchamos en el menú “Archivo” y “Inicio”.

Una vez iniciado, se puede ver en el panel de “Trabajos del Centro de Control” cómo ha pasado a estado “Ejecutando”. Para ver el resultado de la ejecución basta con hacer doble clic sobre el resultado obtenido y se observará un resumen de la ejecución (datos cargados en almacén, etc).

Finalmente, para comprobar que todos los datos han sido cargados correctamente en las Dimensiones y el Cubo, desde el Centro de Diseño hacemos clic derecho sobre cada una de ellas y pulsamos en “Datos”.

4. Aproximación metodológica para desarrollo de Almacenes de Datos XML

Tras haber desarrollado un Almacén de Datos Relacional, se va a exponer una propuesta sobre una aproximación metodológica para el desarrollo de un Almacén de Datos XML.

4.1 Descripción del proceso metodológico

Para desarrollar un Almacén de Datos XML partiremos de un PIM Multidimensional y, aplicando las reglas de transformación que se definirán en el siguiente apartado, se obtendrá el PSM XML, que será el esquema del Almacén de Datos XML. En la figura 13 se puede ver el proceso.

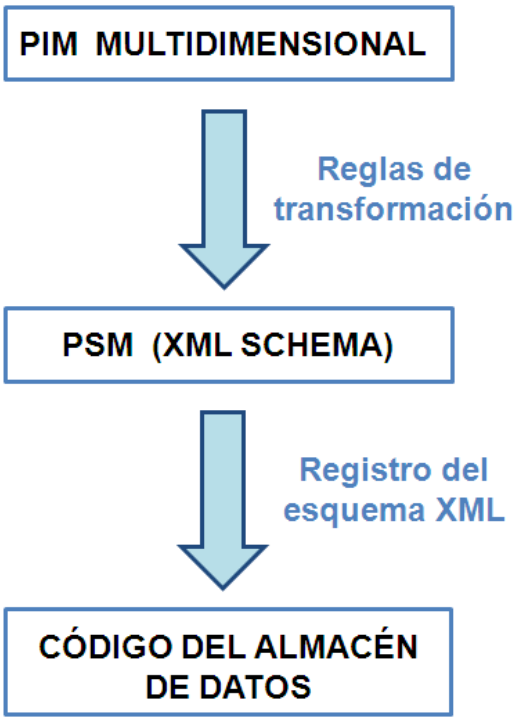

Figura 13: *Proceso metodológico de la propuesta realizada*

Posteriormente, el esquema XML obtenido se registrará en un producto, que en nuestro caso será el XMLDB de Oracle, para obtener así el esquema del Almacén de Datos XML.

4.2 Reglas de transformación de PIM a PSM

Para que esta metodología propuesta pueda ser empleada de forma sistemática en las transformaciones de un PIM Multidimensional a un PSM XML, se han definido las reglas de transformación (mostradas en la tabla 2) entre los modelos a nivel conceptual.

PIM	PSM XML
Transformaciones para las clases	
Star Packages	Cada Star Package incluirá como subelemento una secuencia de tipo complexType con los Hechos, las Dimensiones y las Bases como subelementos XML con sus correspondientes atributos.
Hechos	Se creará un subelemento por cada Hecho que incluirá como subelemento una secuencia de tipo complexType con un atributo ID, los atributos del Hecho como subelementos XML y un elemento un elemento IDREFS para referenciar a cada una de las Dimensiones.
Dimensiones	Se creará un subelemento por cada Dimensión que incluirá como subelemento una secuencia de tipo complexType con un subelemento XML por cada atributo de la Dimensión, un elemento IDREFS para referenciar los Hechos (Ventas) y un elemento IDREF para representar la asociación con el elemento Base.
Bases	Se creará un subelemento por cada Base que incluirá como subelemento una secuencia de tipo complexType con un atributo ID, los atributos de la Base como subelementos XML y un elemento IDREF para referenciar a su correspondiente Dimensión.
Transformaciones para las relaciones entre clases	
Hecho - Dimensión	El elemento correspondiente al Hecho debe incluir un elemento IDREFS para referenciar las 0..n Dimensiones. La Dimensión debe contener un elemento IDREFS para referenciar los 1..n Hechos.
Dimensión - Base	El elemento correspondiente a la Dimensión debe incluir un elemento IDREF que referencie la Base correspondiente. La Base debe contener un elemento IDREF a la Dimensión.
Base - Base	El elemento correspondiente a la Base debe incluir un elemento IDREFS que referencie al resto de las Bases

Tabla 2: *Reglas de transformación de PIM Multidimensional a PSM XML*

4.3 Caso de estudio

Con el propósito de validar esta propuesta, se ha desarrollado el caso de estudio que se describe a continuación.

4.3.1 PIM del Almacén de Datos

En este punto se muestra de nuevo el PIM Multidimensional desarrollado para este caso de estudio, puesto que se trata del modelo a partir del cual se desarrollará el PSM XML.

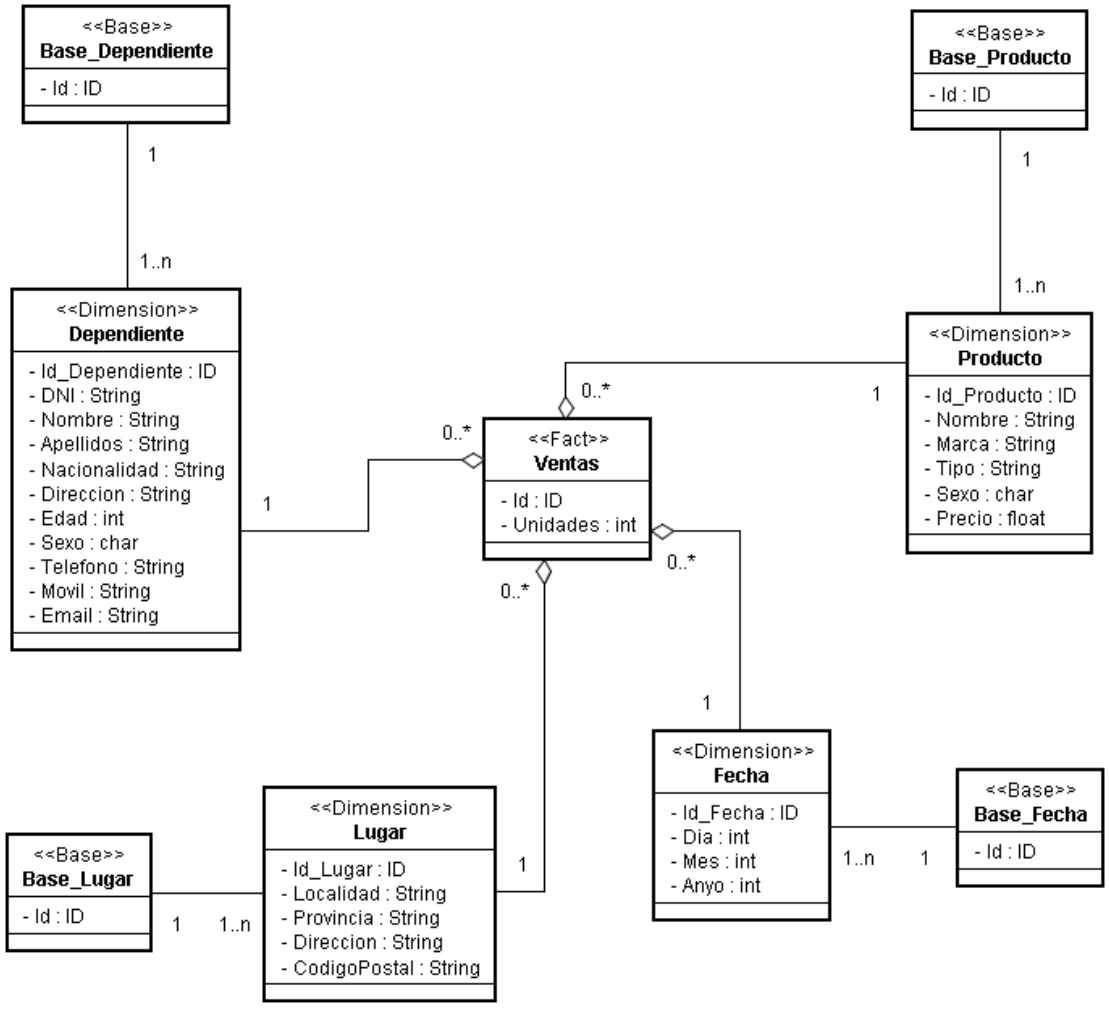

Figura 14: *PIM del caso de estudio*

El análisis realizado sobre el PIM Multidimensional en el capítulo 3 de esta memoria sería igualmente válido en este punto puesto que se trata del mismo modelo.

4.3.2 PSM del Almacén de Datos XML

Partiendo del PIM Multidimensional, y aplicando las reglas de transformación indicadas en la figura 15, se ha desarrollado el PSM para este caso de estudio, es decir, un XML Schema.

Este esquema, ha sido desarrollado utilizando la herramienta **XML Spy**, que ofrece una interfaz gráfica para la creación de dicho esquema generando de manera automática el código del esquema posteriormente.

En primer lugar, se han creado 4 tipos complejos, uno para cada Dimensión del problema: “TIPO_DEPENDIENTE”, “TIPO_PRODUCTO”, “TIPO_LUGAR” y “TIPO_FECHA”.

En cuanto a los elementos del esquema, se parte de un elemento raíz llamado “STARPACKAGE” del que derivan tres subelementos: “FACTS”, “DIMENSIONS” y “BASES” (este último opcional).

Dentro del elemento “FACTS”, únicamente trabajaremos con una tabla de Hechos, que será llamada “VENTAS”.

Por otro lado, el elemento “DIMENSIONS” contiene en su interior las cuatro Dimensiones del problema: “DEPENDIENTE”, “PRODUCTO”, “LUGAR” y “FECHA”. Cada una de ellas, será de su tipo complejo correspondiente creado anteriormente (“FECHA” de tipo “TIPO_FECHA”…).

Finalmente, el elemento “BASES” contendrá como subelementos las bases correspondientes a cada una de las Dimensiones: “BASE_DEPENDIENTE”, “BASE_PRODUCTO”, “BASE_LUGAR” y “BASE_FECHA”.

A continuación se realizará un análisis en profundidad sobre cada Hecho, Dimensión y Base del esquema.

- **Tabla de Hechos "Ventas"**

La tabla de Hechos Ventas, estará comprendida por un subelemento UNIDADES, que hará alusión a las unidades vendidas. Además contendrá un subelemento por cada una de las Dimensiones que referencie de tipo IDREF: "DIM_DEPENDIENTE", "DIM_PRODUCTO", "DIM_LUGAR" y "DIM_FECHA". También dispondrá de un atributo ID que servirá como identificador para cada uno de los Hechos.

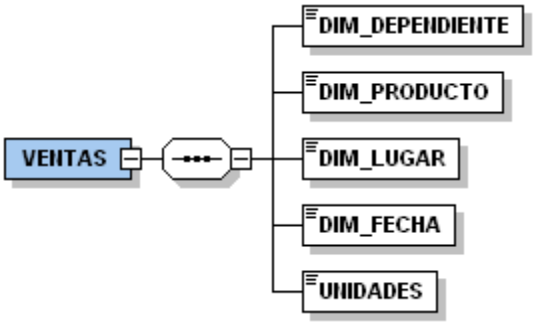

Figura 15: *Elementos de la tabla de hechos "Ventas"*

El código generado al implementar el elemento Ventas es el siguiente:

```
<xs:element name="VENTAS">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="DIM_DEPENDIENTE" type="xs:IDREF"/>
      <xs:element name="DIM_PRODUCTO" type="xs:IDREF"/>
      <xs:element name="DIM_LUGAR" type="xs:IDREF"/>
      <xs:element name="DIM_FECHA" type="xs:IDREF"/>
      <xs:element name="UNIDADES" type="xs:integer"/>
    </xs:sequence>
    <xs:attribute name="ID" type="xs:ID" use="required"/>
  </xs:complexType>
</xs:element>
```

- **Dimensión "Dependiente"**

La Dimensión Dependiente, tendrá asignada un tipo complejo creado para el almacenamiento de los datos relevantes de cada "Dependiente". La información que contendrá será el DNI, nombre, apellidos, nacionalidad, dirección, edad, sexo, teléfono, móvil y e-mail.

Además contendrá un atributo identificador "ID" de tipo ID, un elemento "VENTAS" de tipo IDREFS que referencia a todas las ventas en las que participa y un elemento "BASE" de tipo IDREF para referenciar a la Base correspondiente.

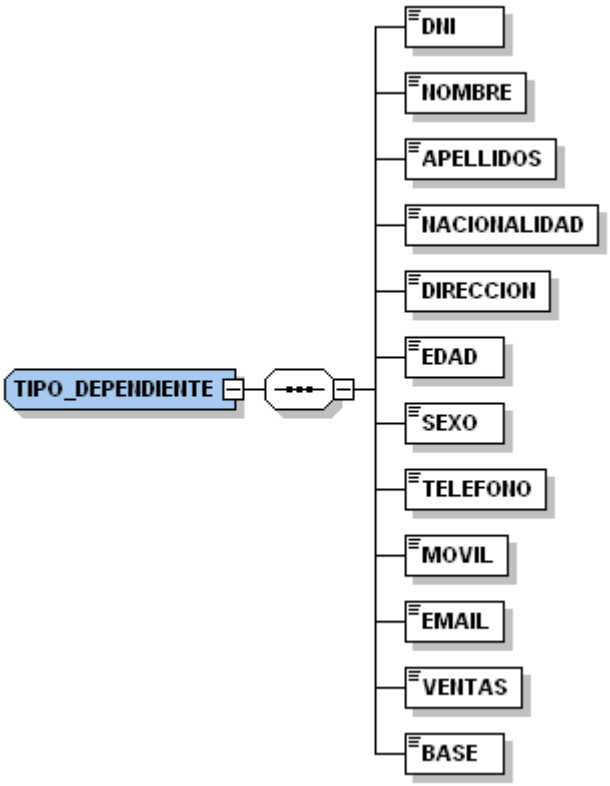

Figura 16: *Creación de tipo complejo TIPO_DEPENDIENTE*

El código generado al implementar este tipo de dato Dependiente es el siguiente:

```
<xs:complexType name="TIPO_DEPENDIENTE">
  <xs:sequence>
    <xs:element name="DNI" type="xs:string"/>
    <xs:element name="NOMBRE" type="xs:string"/>
    <xs:element name="APELLIDOS" type="xs:string"/>
    <xs:element name="NACIONALIDAD" type="xs:string"/>
    <xs:element name="DIRECCION" type="xs:string"/>
    <xs:element name="EDAD" type="xs:integer"/>
    <xs:element name="SEXO" type="xs:string"/>
    <xs:element name="TELEFONO" type="xs:string"/>
    <xs:element name="MOVIL" type="xs:string"/>
    <xs:element name="EMAIL" type="xs:string"/>
    <xs:element name="VENTAS" type="xs:IDREFS"/>
    <xs:element name="BASE" type="xs:IDREF"/>
  </xs:sequence>
</xs:complexType>
```

Una vez creado el tipo de dato “TIPO_DEPENDIENTE” queda asignárselo a la Dimensión Dependiente.

El código resultante sería el siguiente:

```
<xs:element name="DEPENDIENTE">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="TIPO_DEPENDIENTE">
        <xs:attribute name="ID" type="xs:ID" use="required"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
```

- **Dimensión “Producto”**

En cuanto a la Dimensión Producto, tendrá asignada un tipo complejo creado para el almacenamiento de los datos de tipo “Producto”. Estos datos serán el nombre, la marca, el tipo, el sexo y el precio del producto.

Además contendrá un atributo identificador “ID” de tipo ID, un elemento “VENTAS” de tipo IDREFS que referencia a todas las ventas en las que participa y un elemento “BASE” de tipo IDREF para referenciar a la Base correspondiente.

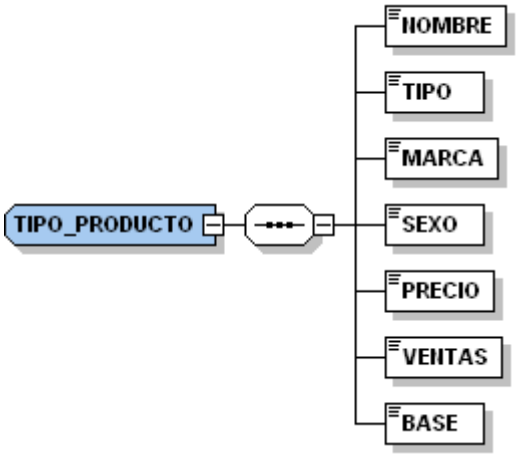

Figura 17: *Creación de tipo complejo TIPO_PRODUCTO*

El código generado al implementar este tipo de dato Producto es el siguiente:

```
<xs:complexType name="TIPO_PRODUCTO">
  <xs:sequence>
    <xs:element name="NOMBRE" type="xs:string"/>
    <xs:element name="TIPO" type="xs:string"/>
    <xs:element name="MARCA" type="xs:string"/>
    <xs:element name="SEXO" type="xs:string"/>
    <xs:element name="PRECIO" type="xs:float"/>
    <xs:element name="VENTAS" type="xs:IDREFS"/>
    <xs:element name="BASE" type="xs:IDREF"/>
  </xs:sequence>
</xs:complexType>
```

Una vez creado el tipo de dato “TIPO_PRODUCTO” queda asignárselo a la Dimensión Producto.

El código resultante sería el siguiente:

```
<xs:element name="PRODUCTO">
   <xs:complexType>
    <xs:complexContent>
      <xs:extension base="TIPO_PRODUCTO">
        <xs:attribute name="ID" type="xs:ID" use="required"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
 </xs:element>
```

- **Dimensión “Lugar”**

La Dimensión Lugar, tendrá asignada un tipo complejo creado para el almacenamiento de los datos de tipo “Lugar”. Esta información incluye la dirección, la localidad, la provincia y el código postal del punto de venta.

Además contendrá un atributo identificador “ID” de tipo ID, un elemento “VENTAS” de tipo IDREFS que referencia a todas las ventas en las que participa y un elemento “BASE” de tipo IDREF para referenciar a la Base correspondiente.

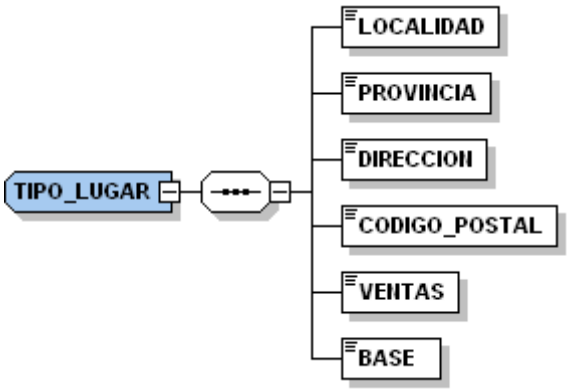

Figura 18: *Creación de tipo complejo TIPO_LUGAR*

El código generado al implementar este tipo de dato Lugar es el siguiente:

```
<xs:complexType name="TIPO_LUGAR">
  <xs:sequence>
    <xs:element name="LOCALIDAD" type="xs:string"/>
    <xs:element name="PROVINCIA" type="xs:string"/>
    <xs:element name="DIRECCION" type="xs:string"/>
    <xs:element name="CODIGO_POSTAL" type="xs:string"/>
    <xs:element name="VENTAS" type="xs:IDREFS"/>
    <xs:element name="BASE" type="xs:IDREF"/>
  </xs:sequence>
</xs:complexType>
```

Una vez creado el tipo de dato “TIPO_LUGAR” queda asignárselo a la Dimensión Lugar.

El código resultante sería el siguiente:

```
<xs:element name="LUGAR">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="TIPO_LUGAR">
        <xs:attribute name="ID" type="xs:ID" use="required"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
```

- **Dimensión "Fecha"**

En cuanto a la Dimensión Fecha, se trataría de un tipo complejo creado para el almacenamiento de los datos de tipo "Fecha", que contendrá la información del día (2 dígitos), el mes (2 dígitos) y el año (4 dígitos) y una descripción opcional, como se puede apreciar en la figura 19. Además se tiene una descripción opcional, por si se tratase de alguna fecha importante.

Además contendrá un atributo identificador "ID" de tipo ID, un elemento "VENTAS" de tipo IDREFS que referencia a todas las ventas en las que participa y un elemento "BASE" de tipo IDREF para referenciar a la Base correspondiente.

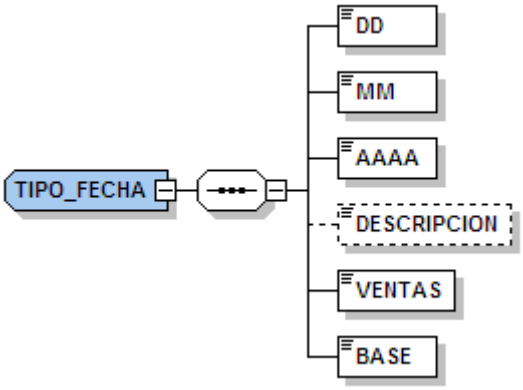

Figura 19: *Creación de tipo complejo TIPO_FECHA*

El código generado al implementar este tipo de dato Fecha es el siguiente:

```
<xs:complexType name="TIPO_FECHA">
  <xs:sequence>
    <xs:element name="DD" type="xs:string"/>
    <xs:element name="MM" type="xs:string"/>
    <xs:element name="AAAA" type="xs:string"/>
    <xs:element name="DESCRIPCION" type="xs:string" minOccurs="0"/>
    <xs:element name="VENTAS" type="xs:IDREFS"/>
    <xs:element name="BASE" type="xs:IDREF"/>
  </xs:sequence>
</xs:complexType>
```

Una vez creado el tipo de dato “TIPO_FECHA” queda asignárselo a la Dimensión Fecha.

El código resultante sería el siguiente:

```
<xs:element name="FECHA">
   <xs:complexType>
    <xs:complexContent>
      <xs:extension base="TIPO_FECHA">
        <xs:attribute name="ID" type="xs:ID" use="required"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
 </xs:element>
```

- **Base “Dependiente”**

Esta Base está compuesta por un atributo identificador “ID” de tipo ID y un subelemento que referencie a la Dimensión Dependiente de tipo IDREF.

Figura 20: *Elemento de la Base Dependiente*

El código resultante es el siguiente:

```
 <xs:element name="BASE_DEPENDIENTE">
   <xs:complexType>
    <xs:sequence>
      <xs:element name="DIM_DEPENDIENTE" type="xs:IDREF"/>
    </xs:sequence>
    <xs:attribute name="ID" type="xs:ID" use="required"/>
  </xs:complexType>
</xs:element>
```

- **Base "Producto"**

Esta Base está compuesta por un atributo identificador "ID" de tipo ID y un subelemento que referencie a la Dimensión Producto de tipo IDREF.

Figura 21: *Elemento de la Base Producto*

El código resultante es el siguiente:

```
<xs:element name="BASE_PRODUCTO">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="DIM_PRODUCTO" type="xs:IDREF"/>
      </xs:sequence>
      <xs:attribute name="ID" type="xs:ID" use="required"/>
    </xs:complexType>
  </xs:element>
```

- **Base "Lugar"**

Esta Base está compuesta por un atributo identificador "ID" de tipo ID y un subelemento que referencie a la Dimensión Lugar de tipo IDREF.

Figura 22: *Elemento de la Base Lugar*

El código resultante es el siguiente:

```
<xs:element name="BASE_LUGAR">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="DIM_LUGAR" type="xs:IDREF"/>
      </xs:sequence>
       <xs:attribute name="ID" type="xs:ID" use="required"/>
    </xs:complexType>
  </xs:element>
```

- **Base “Fecha”**

Esta Base está compuesta por un atributo identificador “ID” de tipo ID y un subelemento que referencie a la Dimensión Fecha de tipo IDREF.

Figura 23: *Elemento de la Base Fecha*

El código resultante es el siguiente:

```
<xs:element name="BASE_FECHA">
   <xs:complexType>
     <xs:sequence>
       <xs:element name="DIM_FECHA" type="xs:IDREF"/>
     </xs:sequence>
     <xs:attribute name="ID" type="xs:ID" use="required"/>
    </xs:complexType>
  </xs:element>
```

Finalmente, el esquema global del caso de estudio quedaría tal y como se muestra en la figura 24. Únicamente no se muestran los subelementos de las Dimensiones, pero ya han sido definidos anteriormente.

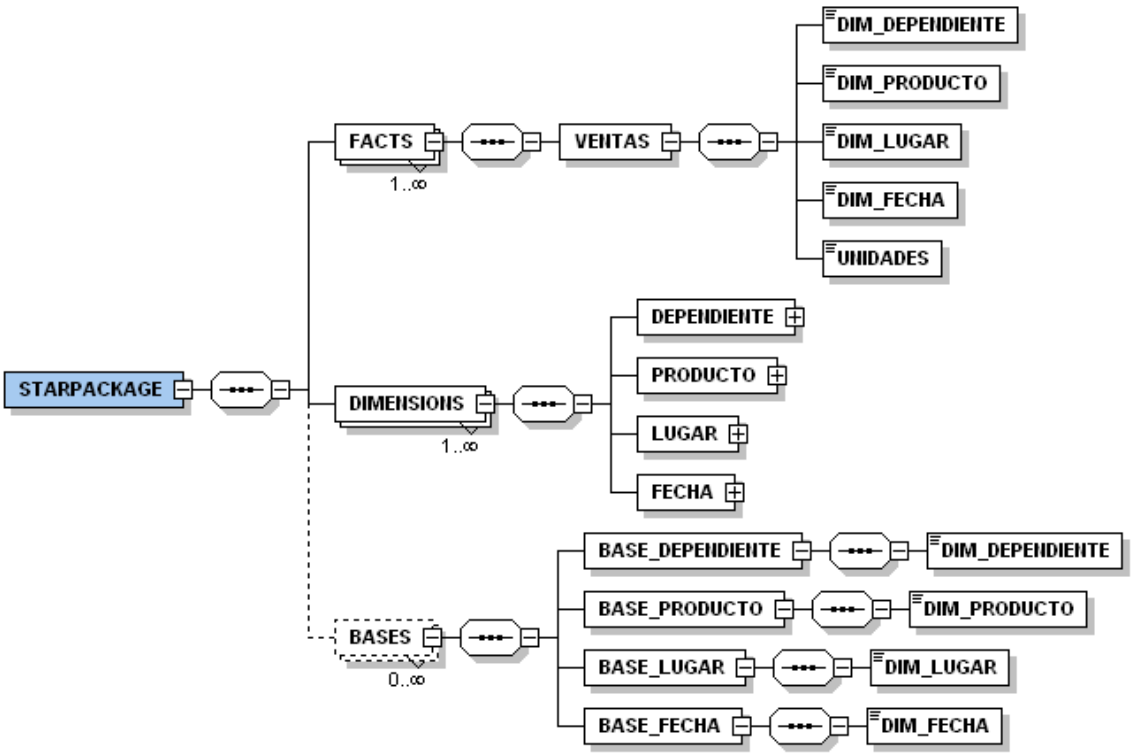

Figura 24: *XML Schema completo*

4.3.3 Implementación del Almacén de Datos XML

El esquema XML obtenido en la sección anterior es implementado en Oracle Database 11g. El uso del estándar W3C XML Schema como modelo de definición de datos permite realizar la implementación del esquema XML obtenido en cualquier SGBD que soporte el estándar.

Se selecciona Oracle Database 11g para realizar la implementación del Almacén de Datos XML por ser un SGBD robusto y maduro, con una amplia aceptación en el mercado, también porque su última versión incorpora un amplio soporte al almacenamiento y la gestión de datos XML, y por la amplia documentación disponible.

Para que el esquema XML pueda ser usado para validar documentos XML en Oracle XML DB primero se debe registrar. Cuando se registra el esquema, se genera una estructura objeto-relacional para almacenar los documentos XML y mejorar el rendimiento dentro de la BD, a partir de ese momento se pueden insertar archivos XML.

Oracle proporciona un esquema XML (*http://xmlns.oracle.com/xdb*) especialmente diseñado para permitir controlar la forma en que XML DB crea la estructura objeto-relacional. Con el fin de mejorar el rendimiento y controlar la forma en que Oracle XML BD crea la estructura objeto-relacional, se realizó una iteración adicional en el desarrollo del esquema XML para incluir el esquema proporcionado por Oracle.

El proceso para registrar el XML Schema ha sido realizado en todo momento mediante interfaz gráfica, salvo la configuración de las variables de entorno.

Este proceso es el siguiente:

1. Crear una variable de entorno llamada ORACLE_HOME con el directorio correspondiente.
2. Crear una variable de entorno llamada LD_LIBRARY_PATH y añadirla el directorio ORACLE_HOME/LIB.
3. Crear una variable de entorno llamada JAVA_HOME y asignarla el directorio donde esté instalado Java.

4. Añadir a la variable de entorno CLASSPATH lo siguiente:

   %ORACLE_HOME%/jdbc/lib/ojdbc5.jar; %ORACLE_HOME%/jlib/orail8n.jar

5. Abrir un navegador e ir a la dirección: ***https://<hostname>:1158/em*** e introducir como usuario ***system*** y clave ***oracle***, siempre y cuando no se haya asignado previamente una clave al usuario System.

6. Ir a: *Schema > Configuration*

   Y configurar los puertos para el protocolo FTP, que será **2100** y para HTTP, que será **8080**.

7. Ir a: *Schema > Resources > Create*

   Indicamos que deseamos obtener el esquema XML desde el sistema de archivos local e indicamos la ruta donde está alojado. Además, solicita que se introduzca la url donde está alojado este esquema. En este caso, cabría la posibilidad de haberlo cargado en algún repositorio o, simplemente alojarlo en algún sitio de Internet. La opción escogida fue ésta última, se alojó el XML Schema en un servidor y se indicó la url correspondiente.

8. Una vez que ha sido registrado, ya se podrá visualizar en la jerarquía, como se puede observar en la figura 25.

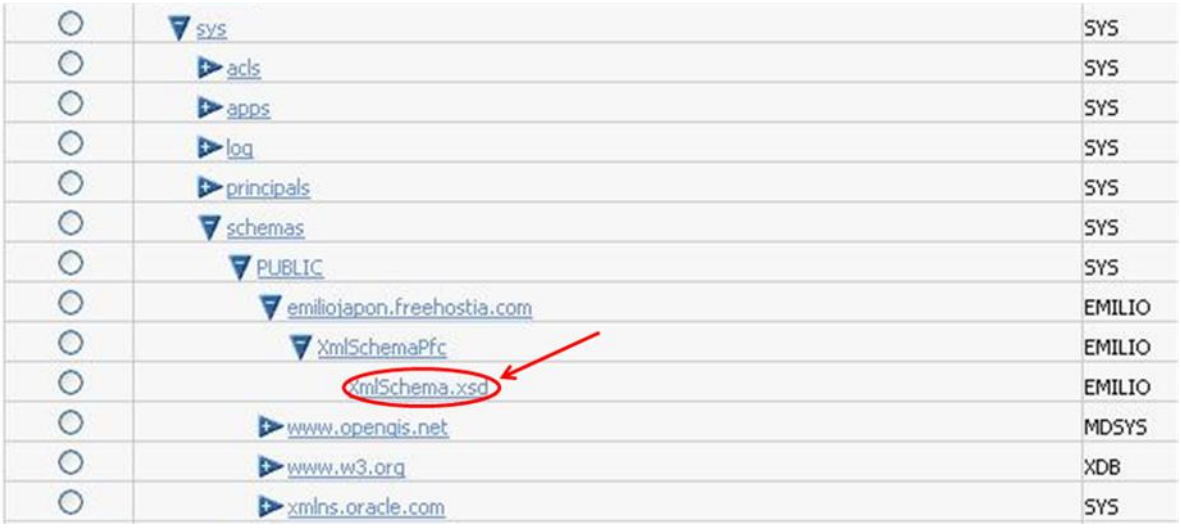

Figura 25: *Visualización del esquema XML registrado dentro de la jerarquía*

Después de registrarlo se puede usar para validar documentos XML, crear tablas y columnas XMLType definidas por el esquema XML. El esquema XML se registra con un identificador único, que es usado por las instancias de los documentos XML que se van a validar con este esquema. El identificador suele tener forma de URL (*https://localhost:1158/.../database/xsd/XmlSchema.xsd*).

El resultado de la validación de la BD XML implementada en Oracle Database 11g fue completamente satisfactorio.

El amplio soporte que Oracle Database 11g proporciona a XML, permite mejorar aún más la implementación realizada, se pueden definir índices XML para optimizar las consultas, crear vistas sobre los documentos XML almacenados para proporcionar acceso relacional a los datos, también permite definir restricciones (Constraints) entre los documentos XML y las tablas relacionales de la Base de Datos para mantener la integridad referencial.

4.3.4 Propuesta de implementación para un Almacén de Datos XML en OWB

En primer lugar, se debe aclarar que la secuencia de pasos que se deberían dar para construir un Almacén de Datos XML en Oracle Warehouse Builder es el mismo que para un Almacén de Datos Relacional (creación de tablas externas, correspondencias, etc).

Una diferencia importante entre el Almacén Relacional y el Almacén para XML está en la extracción de los datos. En el Almacén Relacional debíamos crearnos un fichero con los datos y a partir de ahí extraerlos, pero para el Almacén XML, como las fuentes de datos serán documentos XML, su extracción será más compleja, pues al no contar con el tipo de dato XML Type (proporcionado el Release2 de Oracle 11g) que lo realiza de manera automática, se ha convertido en un problema sin resolver, es por ello no ha sido posible implementarlo.

De cualquier manera, en este caso de estudio se incluye una propuesta sobre cómo debería ser la estructura de almacenamiento de un Almacén de Datos XML para el XML Schema correspondiente. En base a este diseño y siguiendo los pasos para la construcción de un Almacén de Datos Relacional en OWB

detallados en el capítulo anterior, sería posible implementar un Almacén de Datos XML en OWB, una vez solucionado el problema de la extracción de datos.

Debido a que el esquema desarrollado para la propuesta del Almacén de Datos XML es muy extenso, se mostrará por partes.

En primer lugar, se muestran en la figura 26 dos de las tres tablas de Hechos del modelo, que son “Bases” y “Facts”, ambas compuestas por un atributo “ID” de tipo “ID”, además de los subelementos XML correspondientes a sus Dimensiones.

La tabla de Hechos “Bases” se compone de cuatro Dimensiones de un único nivel. Estas Dimensiones son “Base_Fecha”, “Base_Lugar”, “Base_Producto” y “Base_Dependiente”, todas ellas constituidas por un atributo “ID” de tipo “ID” y un subelemento XML de tipo “IDREF” para referenciar a la Dimensión correspondiente (Base_Lugar a la Dimensión Lugar, etc.).

Por otro lado, la tabla de Hechos “Facts” se compone únicamente de una Dimensión de un solo nivel, la Dimensión “Ventas”. Esta Dimensión está constituida por un atributo “ID” de tipo “ID” y una serie de subelementos XML, que son “Unidades” de tipo entero para determinar las unidades vendidas de un mismo producto en una compra, “Dependiente”, “Producto”, “Lugar” y “Fecha”, todas ellas de tipo “IDREF”, para referenciar a cada una de las Dimensiones respectivamente.

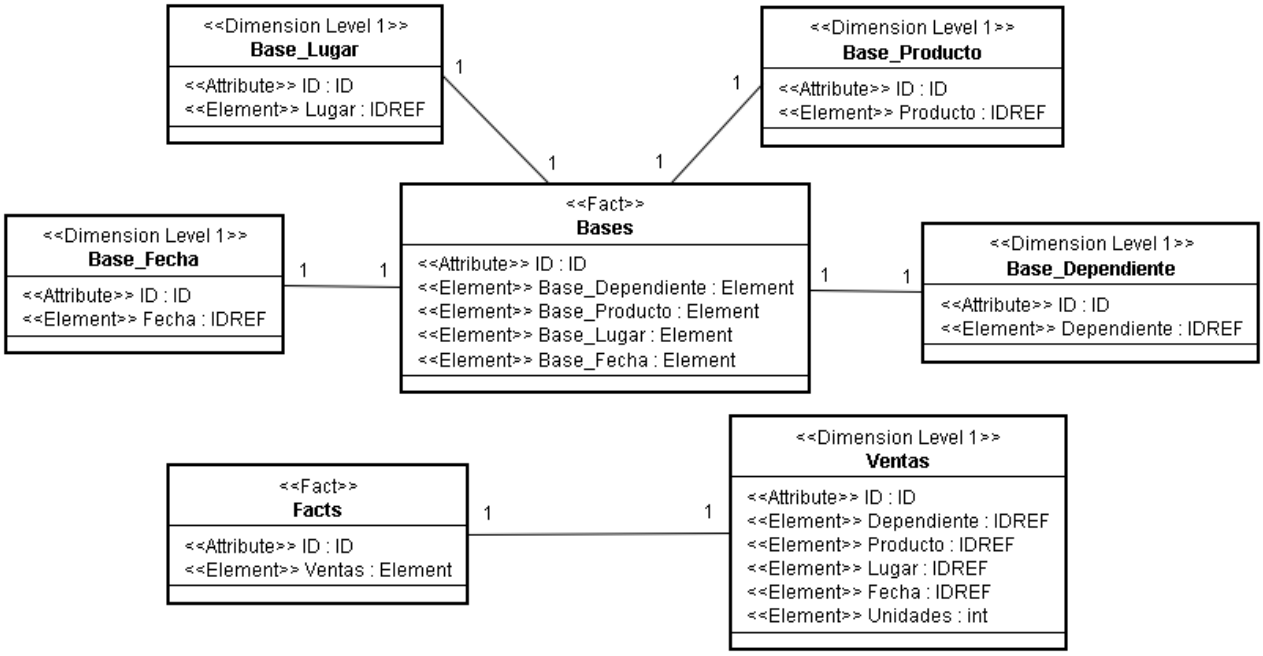

Figura 26: *Almacén de Datos XML*

A continuación, se muestra en las figuras 27 y 28 la parte correspondiente a la tabla de Hechos “Dimensions”.

En la figura 27, en primer lugar se observa la tabla de Hechos “Dimensions” con un atributo “ID” de tipo “ID” y con cuatro subelementos XML que componen a cada una de las Dimensiones del Almacén de Datos. Estos subelementos son de tipos complejos definidos por el usuario (“TIPO_DEPENDIENTE”, “TIPO_LUGAR”, “TIPO_FECHA” y “TIPO_PRODUCTO”).

En cuanto a la Dimensión de nivel 1 “Dependiente” está compuesta por un atributo “ID” de tipo “ID” y una serie de subelementos XML. Dentro de estos subelementos están “Ventas” de tipo “IDREFS” para referenciar a cada uno de los Hechos en los que interviene esta Dimensión y “Base_Dependiente” de tipo “IDREF” para referenciar a la tabla Base correspondiente. Además, el resto de subelementos XML sirven para referenciar a cada una de las Dimensiones de nivel 2 en su misma jerarquía. Estos subelementos son “DNI”, “Nombre”, “Apellidos”, “Nacionalidad”, “Dirección”, “Edad”, “Sexo”, “Teléfono”, “Móvil” y “Email”.

Por otro lado, en las tablas correspondientes al nivel 2 de la Dimensión Dependiente, únicamente se citan cuatro subelementos XML, que son “Valor”, “Tipo”, “Mínimo” y “Máximo” de ocurrencias. En realidad sería necesario almacenar más características que resultan importantes dentro de un elemento XML, pero por simplicidad se han incluido únicamente las citadas.

Las relaciones entre las Dimensiones de nivel 2 y la Dimensión Dependiente pueden ser de 1 a 1 o de 1 a N, en función de si varios dependientes pueden tener o no el mismo valor para un determinado campo, es decir, para el caso del DNI o el teléfono móvil, no podrá ser repetido, por lo que será una relación de 1 a 1, mientras que para el caso del nombre o la edad sí, por lo tanto será una relación de 1 a N.

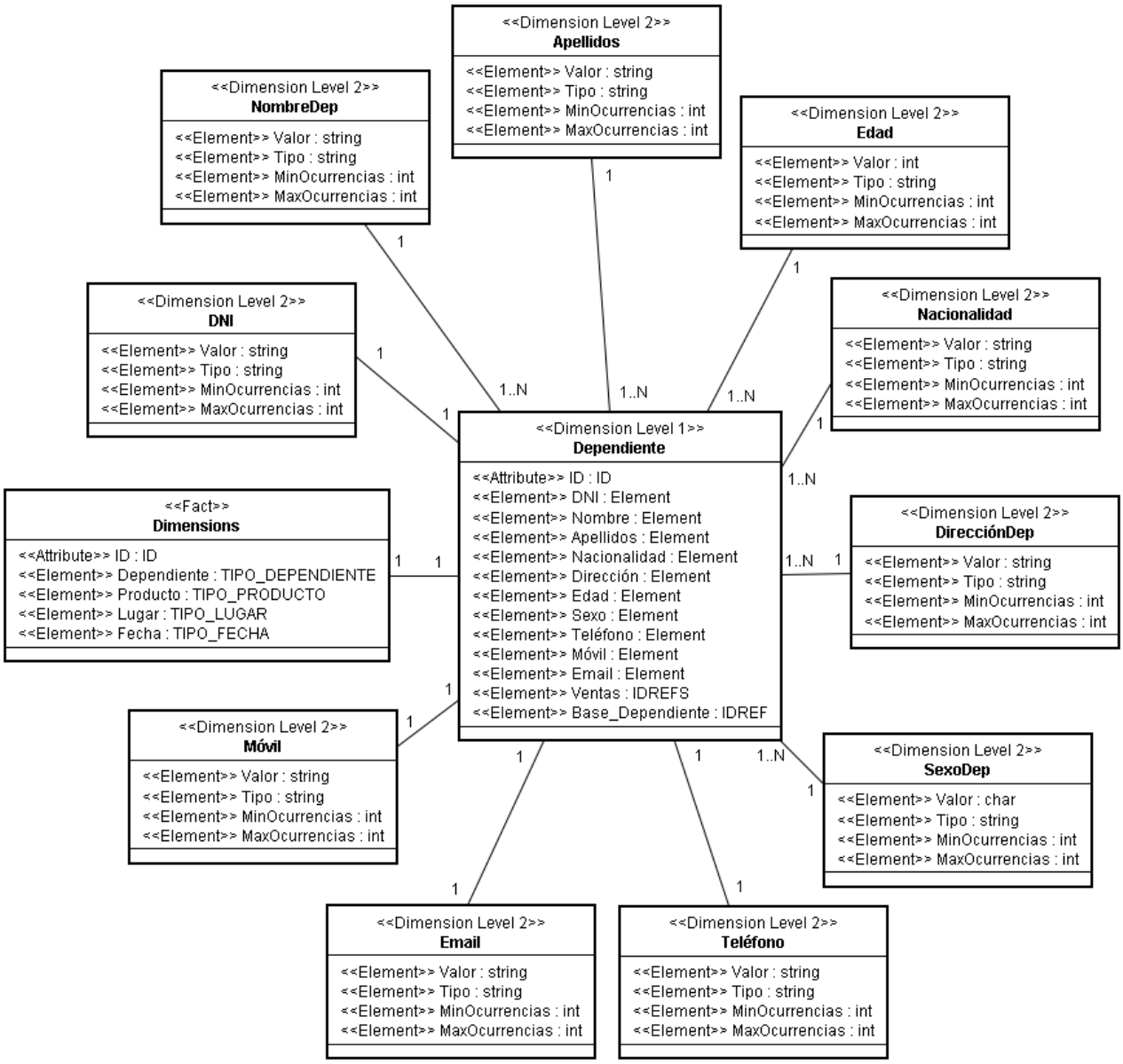

Figura 27: *Almacén de Datos XML*

Finalmente, en la figura 28 se muestra de nuevo la tabla de Hechos "Dimensions" y las Dimensiones "Fecha", "Producto" y "Lugar", además de las Dimensiones de nivel 2 que constituyen cada una de las anteriores.

En primer lugar, la Dimensión "Fecha" contiene un atributo "ID" de tipo "ID" y una serie de subelementos XML. Entre estos subelementos se encuentra "Base_Fecha" de tipo "IDREF" para referenciar a la tabla Base correspondiente y "Ventas" de tipo "IDREFS" para referenciar a cada uno de los Hechos en los que interviene esta Dimensión. Además, contiene también los subelementos

"Día", "Mes", "Año" y "Descripción", cuyo contenido estará contenido en las Dimensiones de nivel 2 correspondientes.

En esta segunda parte del diagrama se mantiene el apunte mencionado anteriormente sobre la información recogida de los elementos XML en las Dimensiones de nivel 2.

Por otro lado, la Dimensión "Producto" posee un atributo "ID" de tipo "ID" y una serie de subelementos XML. Dentro de estos subelementos se encuentra "Ventas" de tipo "IDREFS" para referenciar a cada uno de los Hechos en los que interviene esta Dimensión y "Base_Producto" de tipo "IDREF" para referenciar a la tabla Base correspondiente. Además, contiene los subelementos "Nombre", "Tipo", "Marca", "Sexo" y "Precio", cuyo contenido estará contenido en las Dimensiones de nivel 2 correspondientes.

Finalmente, la Dimensión "Lugar" posee un atributo "ID" de tipo "ID" y una serie de subelementos XML. Dentro de estos subelementos se encuentra "Ventas" de tipo "IDREFS" para referenciar a cada uno de los Hechos en los que interviene esta Dimensión y "Base_Lugar" de tipo "IDREF" para referenciar a la tabla Base correspondiente. Además, contiene los subelementos "Localidad", "Provincia", "Dirección" y "CódigoPostal", cuyo contenido estará contenido en las Dimensiones de nivel 2 correspondientes.

Las relaciones entre las Dimensiones de nivel 2 y las Dimensiones de nivel 1 pueden ser de 1 a 1 o de 1 a N, en función de si, por ejemplo, varios productos tienen el mismo precio, en cuyo caso sería de 1 a N, o si por el contrario, la dirección de un lugar no pueda ser repetida, en tal caso sería de 1 a 1.

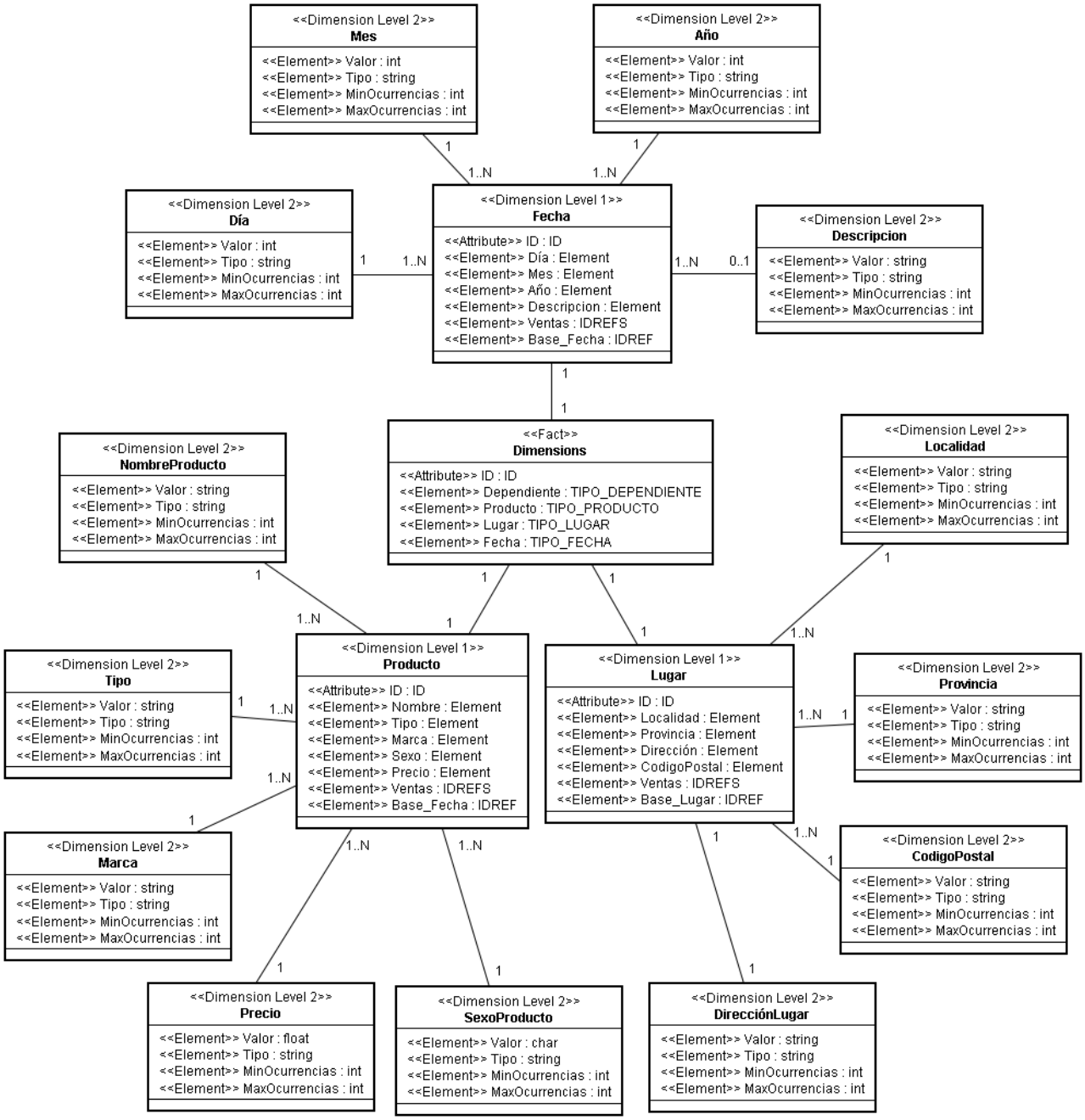

Figura 28: *Almacén de Datos XML*

5. Conclusiones y futuros trabajos

5.1 Conclusiones

Al comenzar este proyecto fin de carrera se fijó como objetivo el desarrollo de un Almacén de Datos XML, proponiendo para ello una aproximación metodológica y validando la propuesta mediante un caso de estudio. Para alcanzar el mismo, se fijaron una serie de objetivos parciales, que a continuación vamos a analizar para ver en qué medida se han satisfecho.

Inicialmente se han realizado una serie de estudios previos, necesarios para llevar a cabo el proyecto, como son los Almacenes de Datos en general, una metodología para el desarrollo dirigido por modelos de Almacenes de Datos Relacionales, la tecnología XML y las BD XML. Además, se han estudiado tanto las diferentes aproximaciones que existen para el desarrollo de Almacenes de Datos XML, como los productos que a posteriori se han empleado para la realización del caso de estudio, es decir, XML Spy, Oracle Workflow, Oracle XML DB y, principalmente, Oracle Warehouse Builder.

Tras la realización de estos estudios se seleccionó un caso de estudio basado en el tradicional ejemplo de “Ventas” para el desarrollo de un Almacén de Datos Relacional siguiendo la propuesta metodológica estudiada. Para ello, en primer lugar se desarrolló un PIM Multidimensional para el caso de estudio seleccionado y, seguidamente, se desarrolló el PSM Relacional aplicando las reglas de transformación indicadas en la metodología estudiada.
Una vez que se diseñó el PSM Relacional, se procedió a la implementación del Almacén de Datos Relacional en la herramienta Oracle Warehouse Builder. Al tratarse del primero, también sirvió para familiarizarse con esta herramienta.

Posteriormente, se propuso una aproximación metodológica definiendo unas reglas de transformación genéricas para pasar del PIM al PSM, es decir, al XML Schema del Almacén de Datos XML.
Seguidamente, para validar esta propuesta, se procedió a realizar el desarrollo del Almacén de Datos XML para el mismo caso de estudio, aplicando las reglas de transformación definidas en este PFC para obtener el XML Schema del Almacén de Datos XML. Este XML Schema se registró posteriormente en el producto XMLDB de Oracle.

Para terminar, también se hizo una propuesta de cómo podría implementarse el Almacén de Datos XML, usando el producto OWB de Oracle 11g Release 2.
Por lo tanto, podemos concluir afirmando que se ha conseguido de forma satisfactoria alcanzar el objetivo fijado al inicio de este trabajo.

A nivel personal este proyecto me ha aportado varias cosas. Principalmente la oportunidad de profundizar en conceptos que en algunos casos no había visto durante la carrera Ingeniería Informática, como los Almacenes de Datos y adquirir conocimientos sobre las herramientas anteriormente mencionadas.

Me ha resultado un proyecto interesante, ya que me ha permitido conocer la gran importancia que puede tener el disponer de este tipo de herramientas para un negocio a la hora de tomar decisiones gerenciales o estratégicas.

5.2 Futuros trabajos

Una vez finalizado el presente proyecto fin de carrera, se pueden considerar una serie de mejoras y ampliaciones sobre el trabajo desarrollado.

Debido a que no han sido empleados los productos específicos para desarrollar Almacenes de Datos XML que hay actualmente, como el Release 2 de Oracle 11g, que incorpora el tipo de dato XML Type, sería bueno tratar de construir el Almacén correspondiente a este caso de estudio en esta herramienta, ya que facilitaría mucho las cosas de cara a su construcción.

Otra posible mejora podría ser ampliar el marco, en el que está basado el diseño de este Almacén. En este caso de estudio únicamente se tienen en cuenta las Dimensiones: Dependiente, Producto, Lugar y Fecha, quizás resulte interesante incluir alguna más.

Por otro lado, ya que este caso de estudio se ha realizado sobre un ejemplo conocido, como es el caso de las ventas, también se podría probar a desarrollar otro tipo de Almacén de Datos que pueda resultar útil para un determinado problema, empleando para ello la metodología propuesta en este PFC.

6. Bibliografía y lugares visitados en Internet

Bibliografía

- **"An MDA approach for the development of data warehouses"**

  José-Norberto Mazón y Juan Trujillo, Decision Support Systems, Vol. 45, Issue 1, Pages 41-58, 2008

- **"Finding an Application-Appropriate Model for XML Data Warehouses"**

  Franck Ravat, et al., Information Systems, Vol.35, Issue 6, Pages 662-687, 2010

- **"The Data Warehouse Toolkit (2ND Edition)"**

  Ralph Kimball, Wiley, The complete guide to Dimensional Modeling, 2008

Lugares de Internet sobre documentación de Almacenes de Datos

- http://www.oracle.com/technology/obe/11gr1_owb/index.htm
- http://www.oracle.com/technology/obe/11gr1_owb/owb11g_update_extend_knowledge/less1_setting_up/less1_setting_up.html
- http://www.oracle.com/technology/obe/admin/owb_main.html
- http://www.oracle.com/technology/sample_code/products/warehouse/index.html
- http://www.oracle.com/technology/tech/xml/xmldb/index.html
- http://www.oracle.com/global/lad/solutions/business_intelligence/dw_home.html
- http://www.oracle.com/technology/tech/xml/xmldb/index.html
- http://www.monografias.com/trabajos57/data-warehouse-sql/data-warehouse-sql.shtml

Lugares de Internet sobre herramientas que dan soporte para Almacenes de Datos

- http://www.mysql.com/why-mysql/application-scenarios/data-warehouse.html
- http://www.microsoft.com/sqlserver/2008/en/us/data-warehousing.aspx
- http://www.upv.es/entidades/ASIC/software/IOA_Productos.pdf

Lugares de Internet sobre instalación y configuración de Oracle 11g

- http://www.oracle.com/lang/es/database/index.html
- http://www.oracle.com/pls/db111/portal.portal_db?selected=11&frame=

- **Lugares de Internet sobre instalación, configuración y primeros comienzos de Oracle Warehouse Builder**

  - http://www.oracle.com/technology/obe/obe_bi/index.html
  - http://www.oracle.com/technology/obe/11gr1_owb/index.htm
  - http://blogs.oracle.com/warehousebuilder/

- **Lugares de Internet sobre foros de Oracle Warehouse Builder**

  - http://forums.oracle.com/forums/forum.jspa?forumID=57
  - http://www.dataprix.com/forums/herramientas/oracle-warehouse-builder-owb
  - http://oraclebi.adiante.es/dudas-owb-f3/

- **Lugares de Internet sobre Oracle Workflow**

  - http://www.oracle.com/technology/products/integration/workflow/workflow_fov.html
  - http://www.oracle.com/technology/products/ias/workflow/index.html
  - http://www.dataprix.com/instalacion-oracle-workflow

- **Lugares de Internet sobre información de XML**

  - http://en.wikipedia.org/wiki/XML
  - http://www.w3.org/XML/Schema
  - http://www.w3schools.com/schema/schema_intro.asp
  - http://www.w3schools.com/dtd/dtd_intro.asp
  - http://mat21.etsii.upm.es/mbs/MechXML/que_es_xml_schema.htm
  - http://www.spanish-translator-services.com/espanol/t/infoset.htm

- **Lugares de Internet sobre Oracle XML DB**

  - http://www.oracle.com/technology/obe/11gr1_db/datamgmt/xmldb/xmldb.htm
  - http://www.oracle.com/technology/global/lad-es/pub/articles/jain-xmldb.html
  - http://download.oracle.com/docs/cd/E11882_01/appdev.112/e10492/xdb02rep.htm#i1034512
  - http://www.filibeto.org/sun/lib/nonsun/oracle/11.1.0.6.0/B28359_01/appdev.111/b28369/xdb03usg.htm#BABDADEB
  - http://dis.unal.edu.co/profesores/eleon/cursos/arquitecturaBD/presentaciones/Expo_XML_en_Oracle.pdf
  - http://dieguz2.blogspot.com/2009/03/oracle-registrar-esquema-xsd.html

Apéndice A: Instalación y configuración de Oracle Warehouse Builder

1. Instalación

Se recomienda encarecidamente seguir estas pautas durante la instalación de Oracle, ya que a lo largo de todo el manual se va a hacer referencia a los datos aquí mencionados:

- Nombre de Base de Datos: "**orcl**".
- Puerto: "**1521**".

A pesar de que se puede instalar en el directorio que prefiera el usuario, cuando nos queramos referir a algunos en concreto, en este manual se asume que la instalación de las diferentes herramientas se encontrará alojadas en estos directorios:

- Directorio home de Oracle: **C:**
- Directorio home de OWB: **C:\product\11.1.0\db_1**

2. Configuración

2.1 Requisitos

En este manual se va a suponer que el servidor de BD y el cliente OWB se encontrarán en la misma máquina.

2.2 Instalación de Oracle Workflow (flujo de trabajo)

Dado que estamos trabajando con la versión 11g de Oracle, en el paquete de instalación ya viene incluido el software para la instalación de Oracle Workflow, así que por lo tanto, realizaremos los siguientes pasos:

2.2.1 Abrimos línea de comandos de Windows:
(Inicio -> Ejecutar -> cmd)

2.2.2 Entramos dentro del directorio home de OWB en la carpeta contenedora de los ficheros de instalación de Workflow:
(cd C:\product\11.1.0\db_1\owb\wf\install)

2.2.3 Ejecutamos el fichero instalable de Workflow **(*wfinstall.bat*)**

2.2.4 Aparecerá un cuadro de diálogo como el de la figura 29, donde tendremos que seleccionar la opción de sólo instalar Workflow en el Servidor e introducir los siguientes datos:

- Un nombre de usuario para la cuenta Workflow, por ejemplo “owf_mgr”.
- Una contraseña para dicha cuenta, por ejemplo, “owf_mgr”.
- La contraseña de la Base de Datos instalada en la máquina.
- Para el descriptor de conexión TNS, hay que poner la descripción tal cual viene en el fichero de configuración tnsnames.ora (c:\ product\11.1.0\db_3\NETWORK\ADMIN), que será algo como:

```
(DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = LOCALHOST)(PORT =
1521))(CONNECT_DATA =(SERVER = DEDICATED)(SERVICE_NAME = orcl)))
```

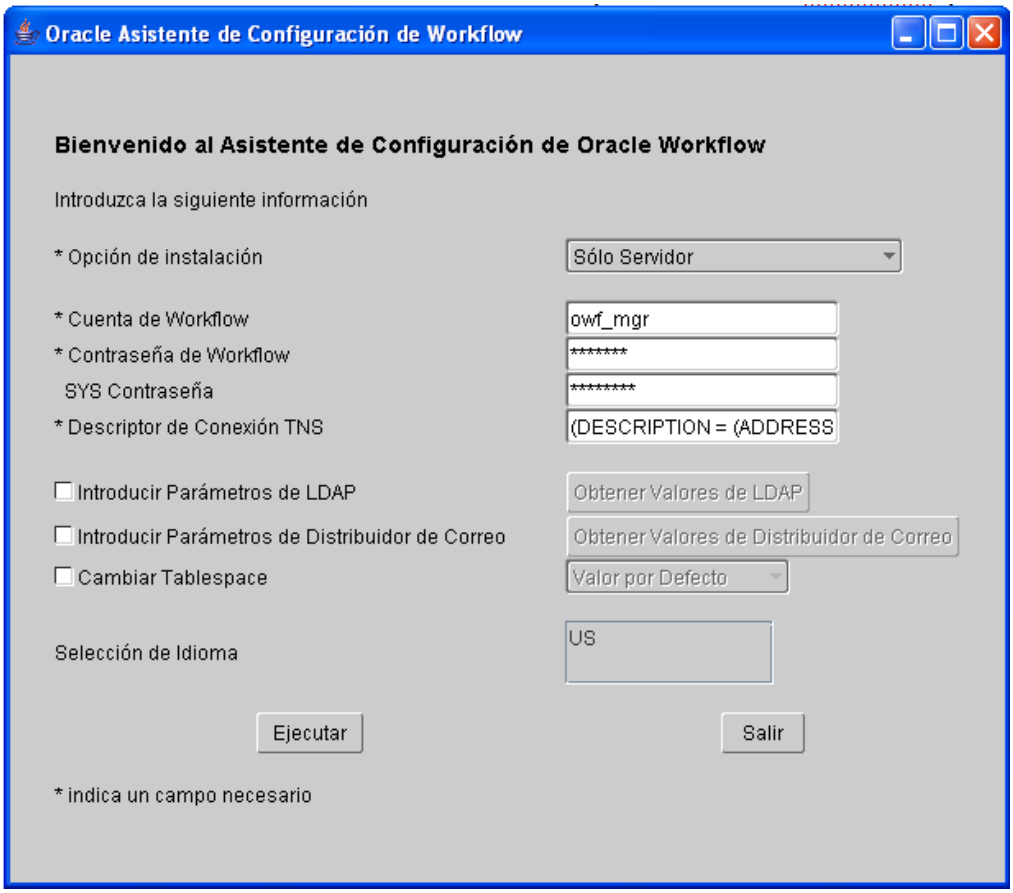

Figura 29: *Cuadro de diálogo para la configuración de Oracle Workflow*

Notas:

1. Es importante no introducir saltos de línea en la descripción.

2. Aunque en la documentación oficial de Oracle se diga que en el descriptor hay que poner “host:puerto:nombre_servicio”, puede dar problemas, debido a que el nombre de servicio que aparece en el fichero tnsnames.ora no es al que se llama.

2.2.5 Hacemos click en “Ejecutar” y comenzará el proceso de instalación, que durará algunos minutos.

2.2.6 Paralelamente, se puede seguir el progreso de configuración mediante los mensajes que van apareciendo en “c:\product\11.1.0\db_3\owb\wf\install\wf.log.”

2.2.7 Finalmente, una vez que nos ha comunicado que todo se ha realizado satisfactoriamente, procederemos a dar privilegios para poder ejecutar cualquier procedimiento al usuario owf_mgr, tal como se muestra en la figura 30.

Para ello hacemos lo siguiente:

- Lanzamos el SQL Plus:
  Inicio > Programas > Oracle OraDb11g_home1 > Desarrollo de Aplicaciones > SQL Plus.
- Nos conectamos como “sys as sysdba”.
- Introducimos la contraseña de la BD.
- Ejecutamos la siguiente orden:
  “grant execute any procedure to owf_mgr;”

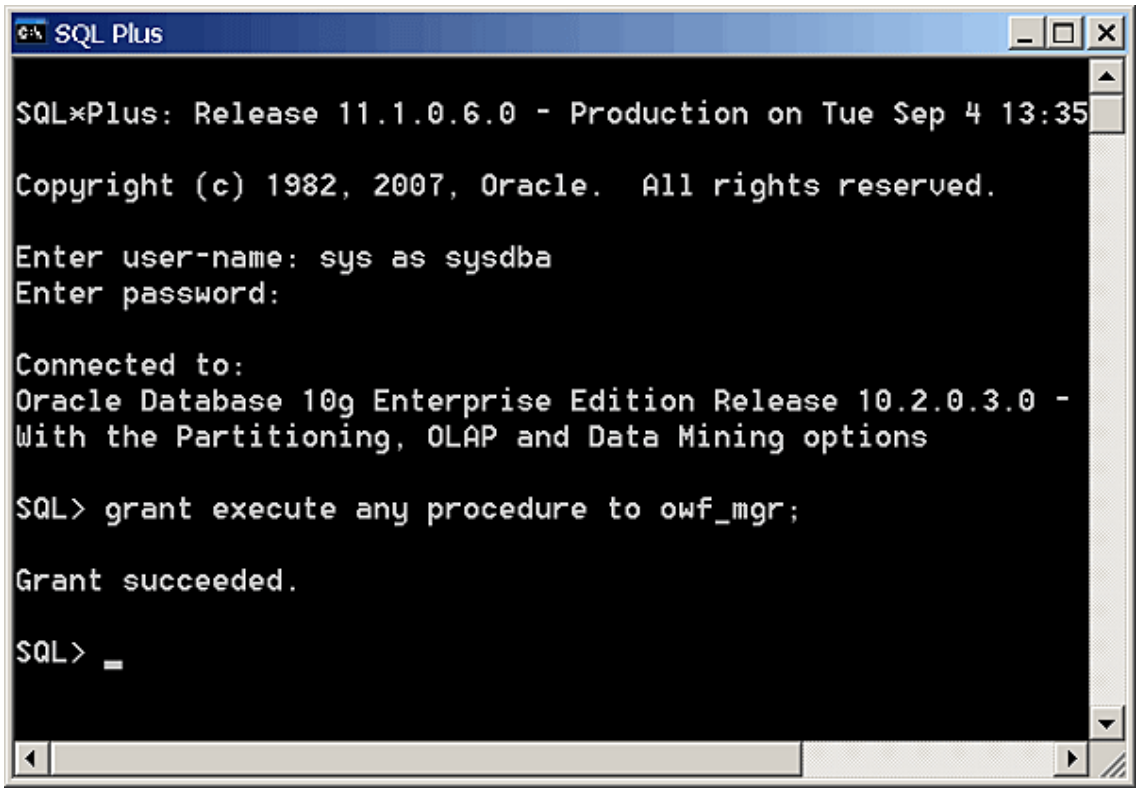

Figura 30: *Proceso de asignación de privilegios a un usuario desde SQL Plus.*

2.3 Configuración del entorno del proyecto

Para instalar y configurar el entorno de Oracle Warehouse Builder (OWB) es necesario realizar los siguientes pasos:

2.3.1 Crear un repositorio usando el Asistente de OWB

Al entrar en el Centro de Diseño de OWB por primera vez, se da la posibilidad de crear un nuevo usuario Warehouse Builder con el que acceder en el futuro. Además se puede crear un nuevo espacio de trabajo, un usuario para dicho espacio e instalar el repositorio Warehouse Builder.

Para realizar esto se deberán seguir los siguientes pasos:

1) Abrir el Asistente de Repositorios de OWB *(Inicio > Programas > Oracle OraDb11g_home1 > Warehouse Builder > Administration > Repository Assistant)*

2) En primer lugar se tendrá que introducir el nombre de máquina (localhost), el puerto (1521) y el nombre de servicio (orcl), tal y como se muestra en la figura 31:

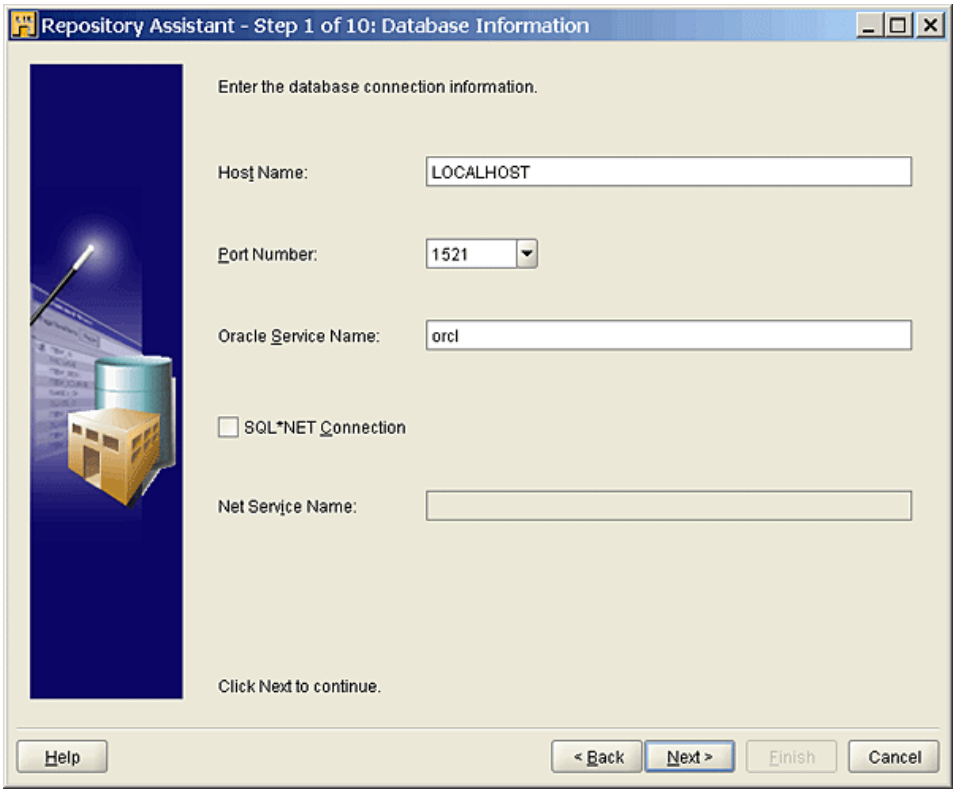

Figura 31: *Creación de un repositorio en OWB (paso 1)*

3) A continuación seleccionamos la opción de Administrar espacios de trabajo en Warehouse Builder, como se muestra en la figura 32.

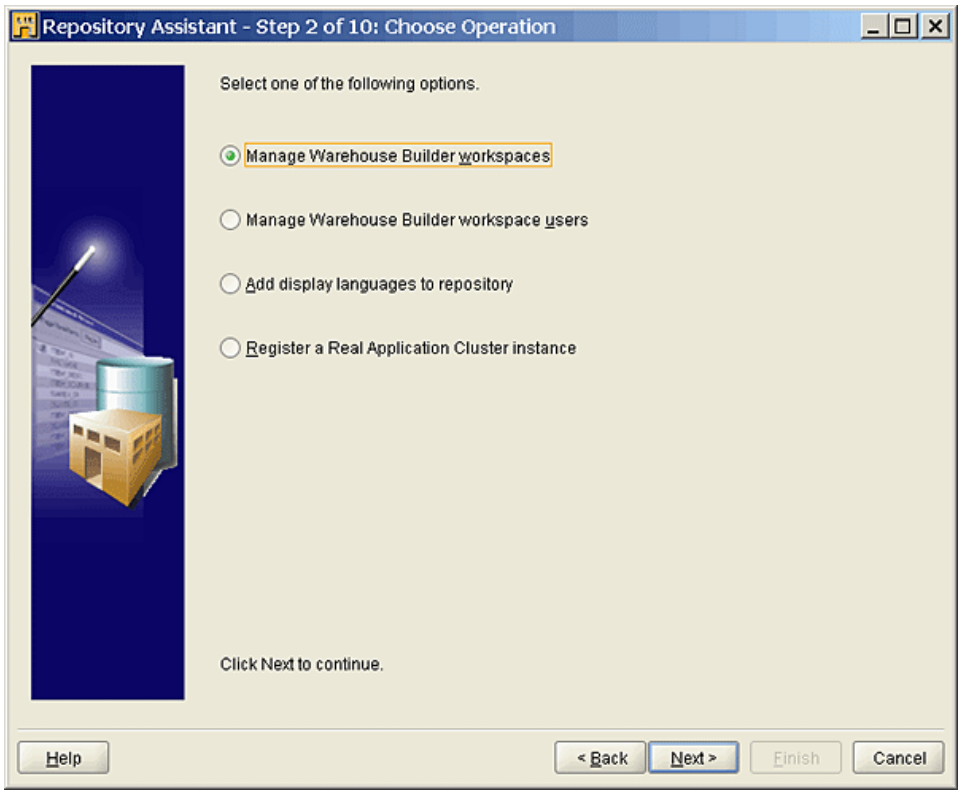

Figura 32: *Creación de un repositorio en OWB (paso 2)*

4) Seguidamente indicamos que queremos crear un nuevo espacio de trabajo, como podemos ver en la figura 33:

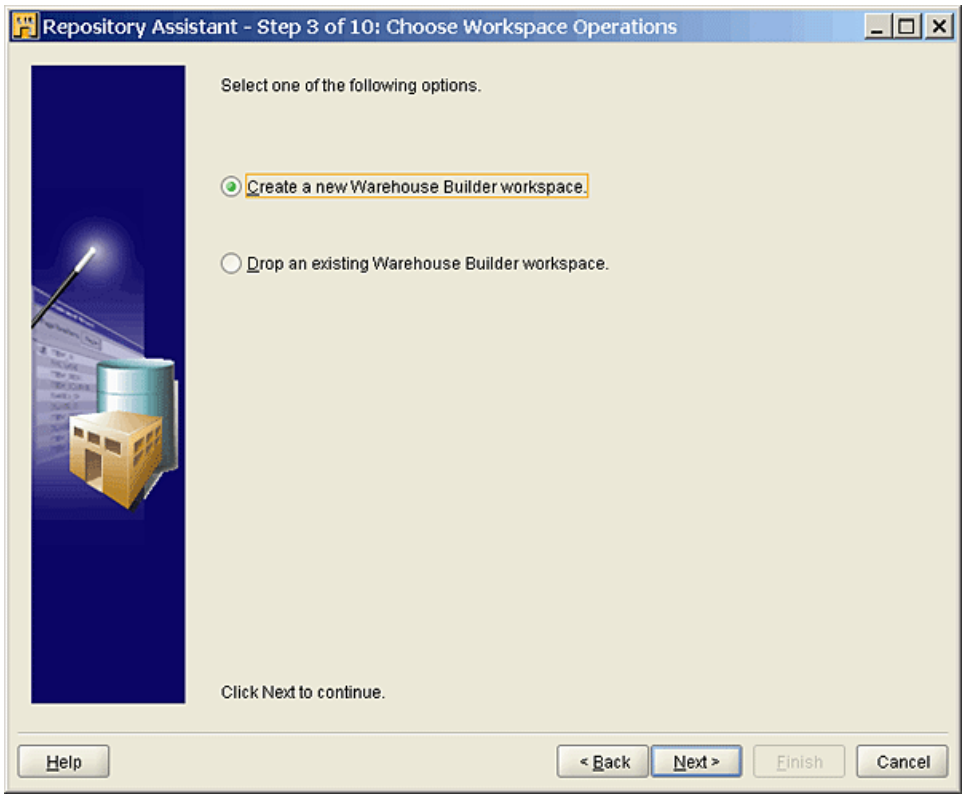

Figura 33: *Creación de un repositorio en OWB (paso 3)*

5) En la siguiente ventana indicamos que deseamos crear un espacio de trabajo con un usuario nuevo como propietario, como se muestra en la figura 34:

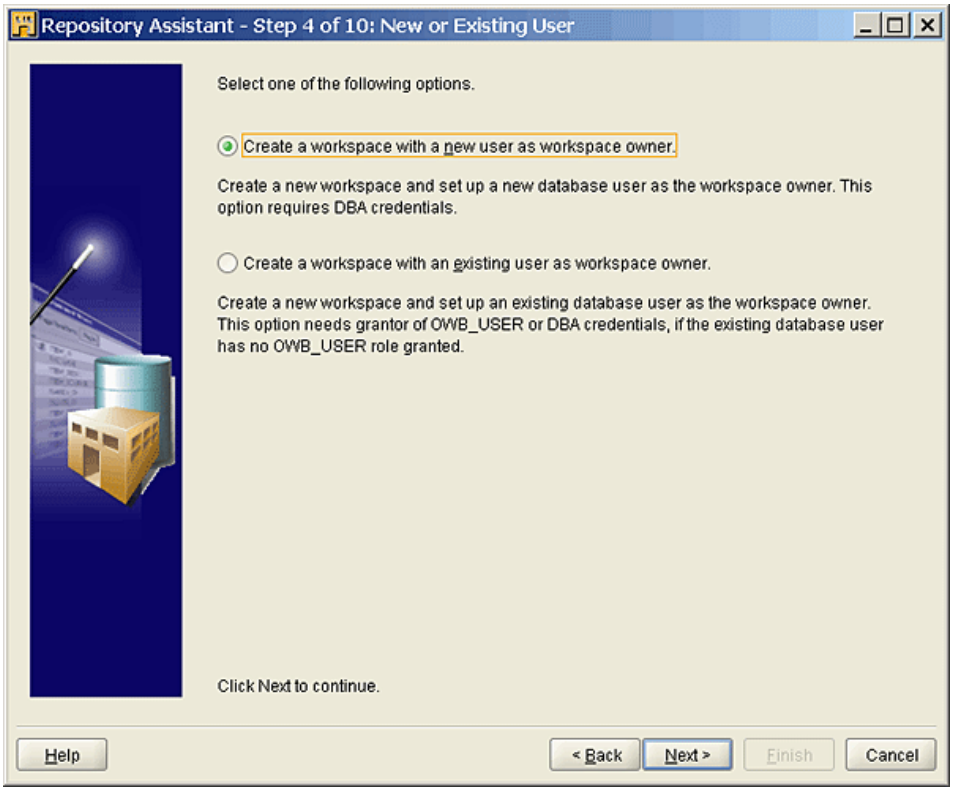

Figura 34: *Creación de un repositorio en OWB (paso 4)*

6) En el siguiente paso debemos introducir los datos correspondientes del Administrador de la Base de Datos (por defecto el usuario SYSTEM con la clave que se le asignó en el periodo de instalación). Ver la figura 35:

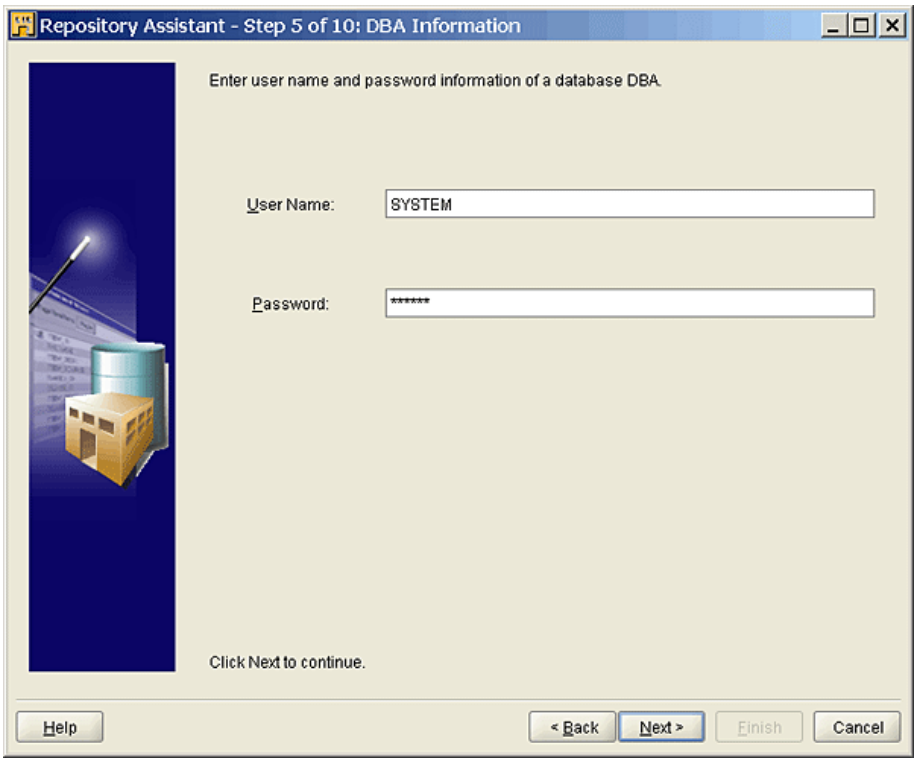

Figura 35: *Creación de un repositorio en OWB (paso 5)*

7) A continuación, debemos introducir los datos del nuevo usuario que será el propietario del espacio de trabajo y el nombre que le queramos dar a dicho espacio. Por ejemplo, como nombre de usuario y contraseña ponemos “owb” y como nombre de espacio de trabajo “my_workspace”, como se puede ver en la figura 36:

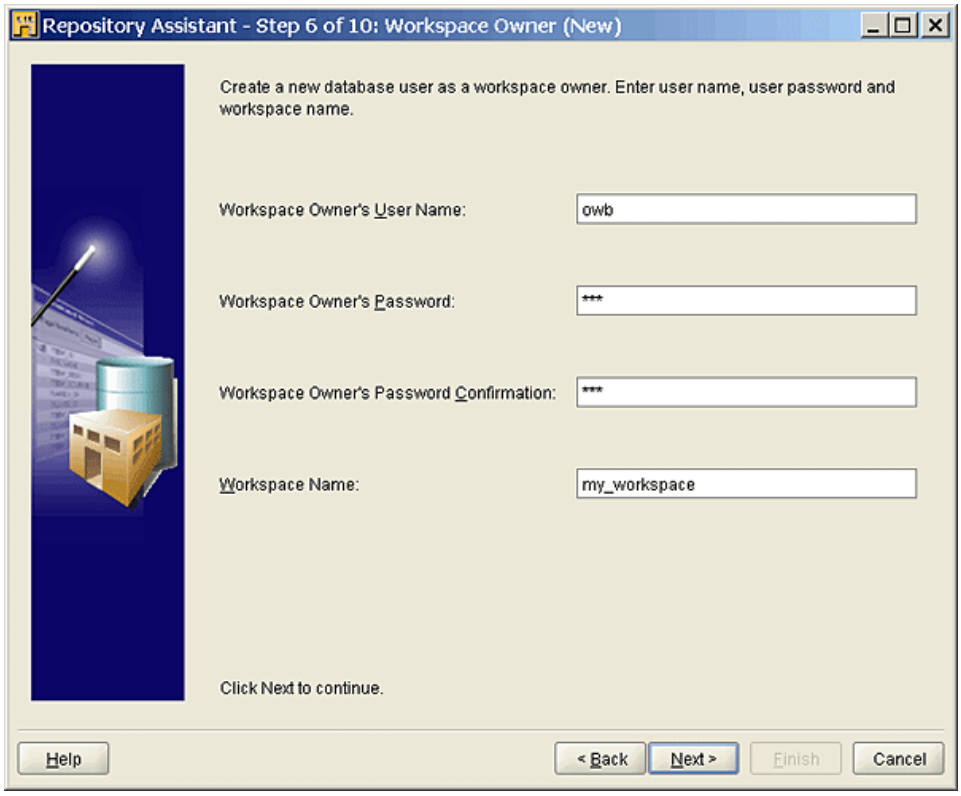

Figura 36: *Creación de un repositorio en OWB (paso 6)*

8) En el siguiente paso nos pide que introduzcamos la contraseña del usuario del sistema OWBSYS, que por defecto su clave será igual. (Ver la figura 37):

Nota: Si el sistema nos muestra un mensaje de error en este punto diciéndonos que la cuenta del usuario OWBSYS se encuentra desactivada, tendremos que activarla mediante código SQL. Para ello abriremos el SQL Plus (como se indicó anteriormente) y ejecutaremos la siguiente orden:

alter user OWBSYS identified by OWBSYS account unlock;

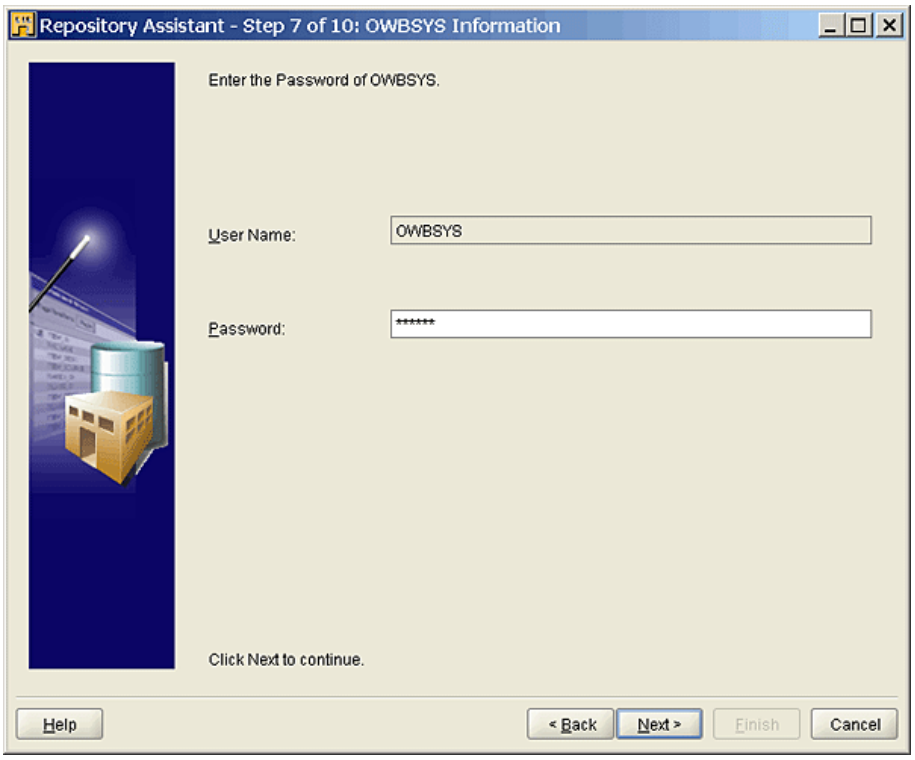

Figura 37: *Creación de un repositorio en OWB (paso 7)*

9) En este paso dejaremos los valores por defecto, que son referentes al espacio de tablas, tal y como se ve en la figura 38:

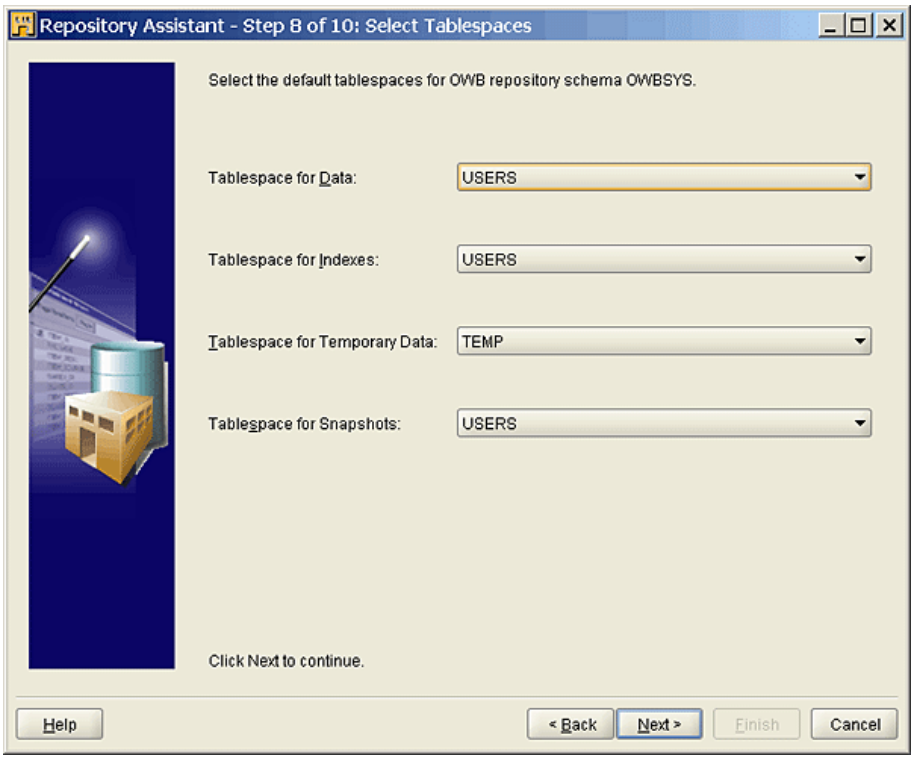

Figura 38: *Creación de un repositorio en OWB (paso 8)*

10) En este paso seleccionaremos como idioma para el repositorio el español o inglés, dependiendo de cada preferencia. Ver figura 39:

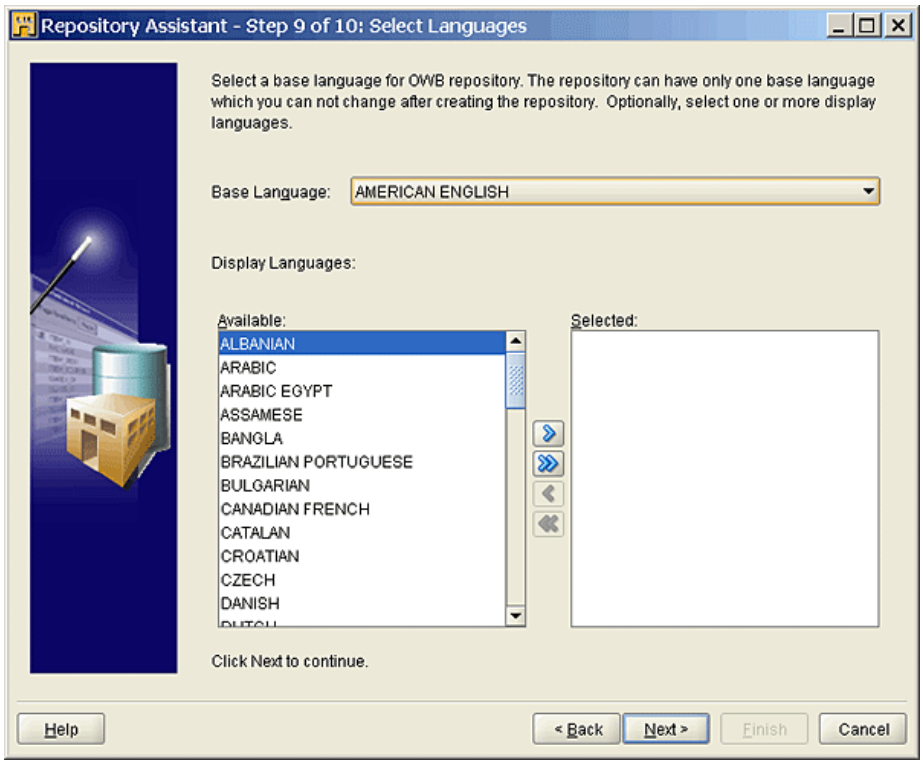

Figura 39: *Creación de un repositorio en OWB (paso 9)*

11) En la siguiente ventana nos pide que indiquemos los usuarios para el espacio de trabajo, pero como por el momento no se va a hacer uso de este espacio no seleccionamos ninguno, tal y como se aprecia en la figura 40:

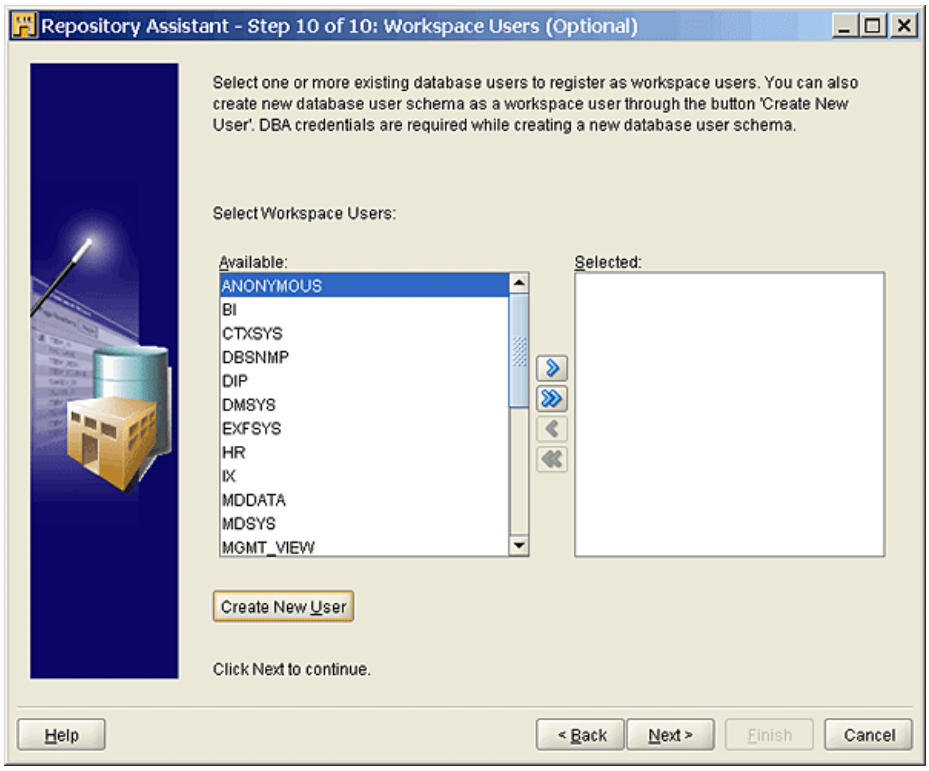

Figura 40: *Creación de un repositorio en OWB (paso 10)*

12) Finalmente se iniciará el proceso de instalación del nuevo repositorio.

2.3.2 Abrir Centro de Diseño de OWB

El Centro de Diseño será la herramienta con la que se trabaje desde ahora en adelante en este proyecto.

Para abrir el Centro de Diseño de OWB se hará de la siguiente manera: *(Inicio > Programas > Oracle OraDb11g_home1 > Warehouse Builder > Design Center)*

Una vez que aparece el cuadro de diálogo, introducimos como nombre de usuario y contraseña “owb”, seleccionamos la casilla “Connection Details” y especificamos como máquina “localhost”, como puerto “1521” y como nombre de servicio “orcl”, tal y como se muestra en la figura 41:

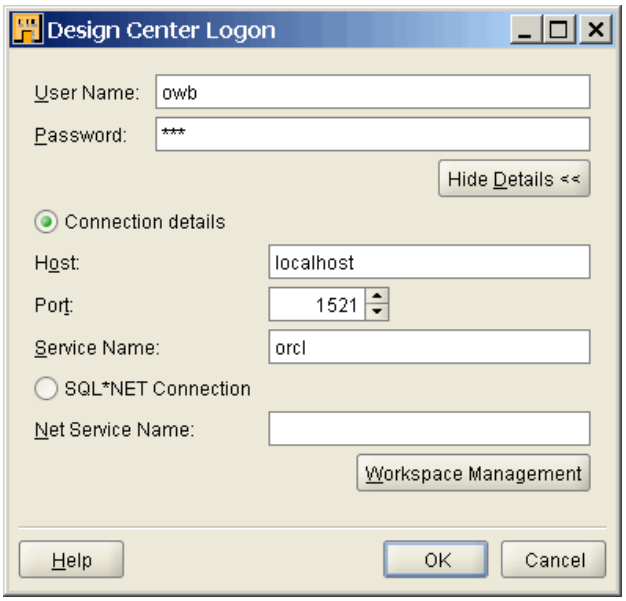

Figura 41: *Cuadro de diálogo para la autentificación del Centro de Diseño*

En este punto aparece por fin el Centro de Diseño de OWB, la principal aplicación cliente de Warehouse Builder.

El Centro de Diseño se divide en tres paneles: el Explorador Global, el Explorador de la conexión y el explorador de proyectos, que contendrá un proyecto inicial vacío llamado MY_PROJECT que se crea cuando se instala Warehouse Builder. (Ver figura 42).

En el punto 2.5.2 del capítulo 2 se explica detalladamente en qué consisten cada uno de los paneles del Centro de Diseño.

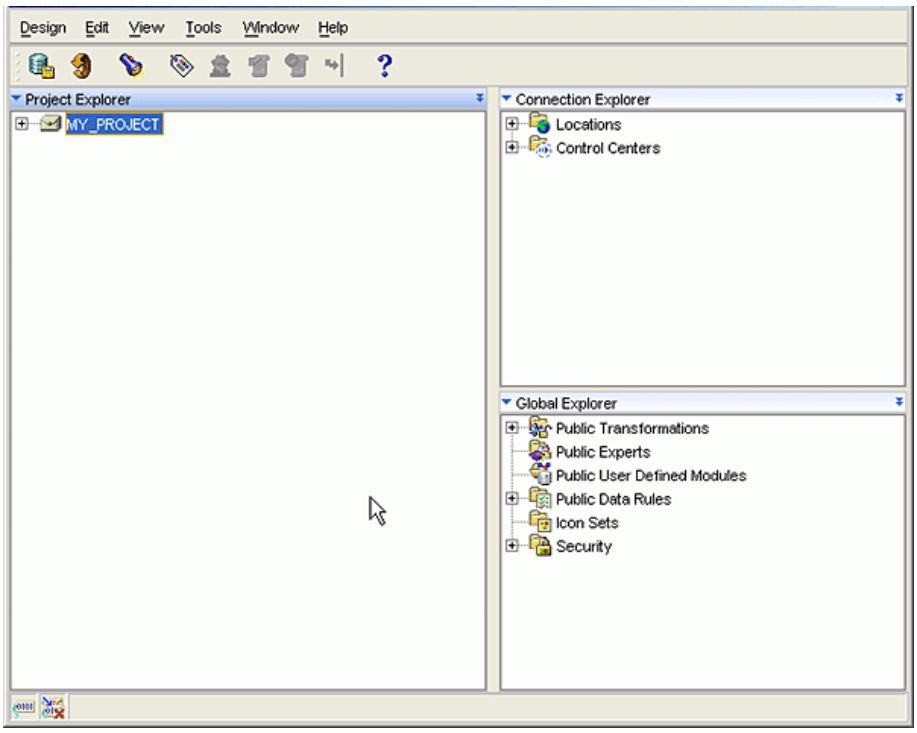

Figura 42: *Cuadro de diálogo para la autentificación del Centro de Diseño*