



Data Management

KeepCoding

Fechas: 12 y 13 de diciembre de 2019.

Horas: 8h

Instructora: Nerea Sevilla

Nombre Material: Ejercicios_Modelado.

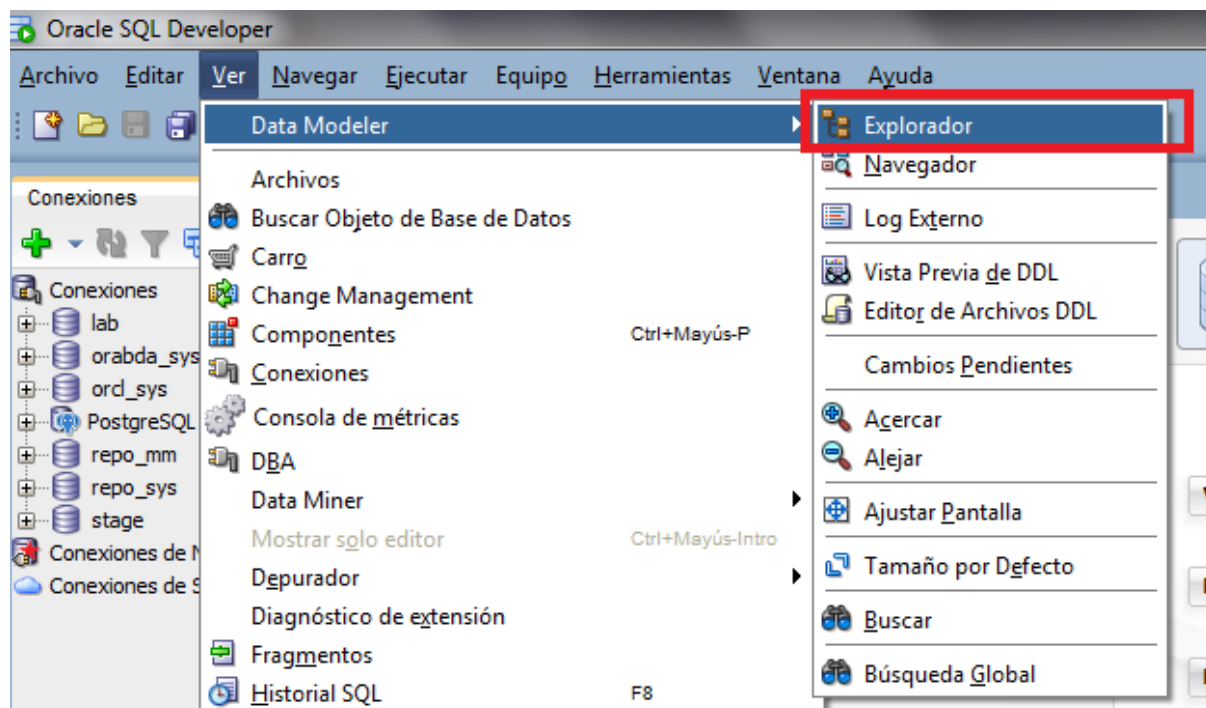
Ejercicio 1.- Creación Modelo Lógico con SQL Developer.

Los modelos lógicos o modelos conceptuales, son las herramientas que nos van a permitir comunicarnos con los usuarios de negocio y nos van a permitir definir las estructuras donde se almacenarán los datos de los sistemas.

Además, los modelos lógicos nos van a ayudar a documentar el ciclo de vida de los desarrollos y justificar los requerimientos y especificaciones que el negocio nos plantean. Ya que nos van a permitir definir las entidades principales y sus atributos que intervendrán en el desarrollo y que permitirán construir posteriormente el modelo E/R que sustentará la base de datos de los desarrollos.

Como analistas de proyectos, deberemos siempre, definir un modelo lógico o conceptual, generar el modelo físico que se implementarán. DataModeler nos puede ayudar a hacerlo.

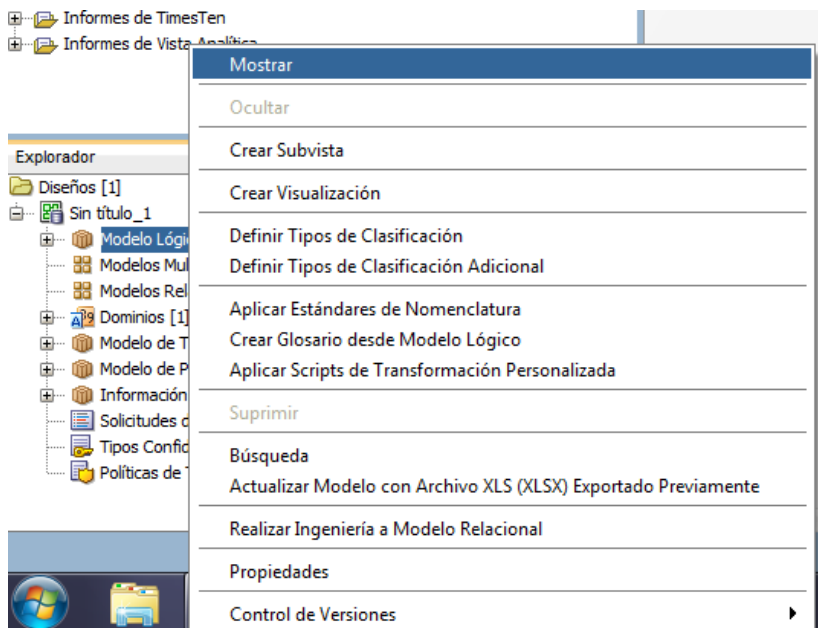
Desde el SQL Developer, abriremos el Data Modeler.
Haciendo click en el menú principal; Ver; Data Modeler; Explorador.



Vamos a crear un modelo de datos para la base de datos de un sistema de gestión de roles y privilegios.




Desde el Explorador del Data Modeler y estando en el menú lateral izquierdo, hacer click en Diseños; Sin Título; “Modelo Lógico”. Botón derecho del menú contextual y luego en la opción “Mostrar, para abrir el editor de Modelos Lógicos.




Podemos ver la barra de herramientas del Data Modeler:



Estando el área de trabajo, hacemos clic en el icono Nueva Entidad de la barra de herramientas , y el cursor cambia a un “+” y formamos un cuadrado y al soltar nos aparece la siguiente pantalla:



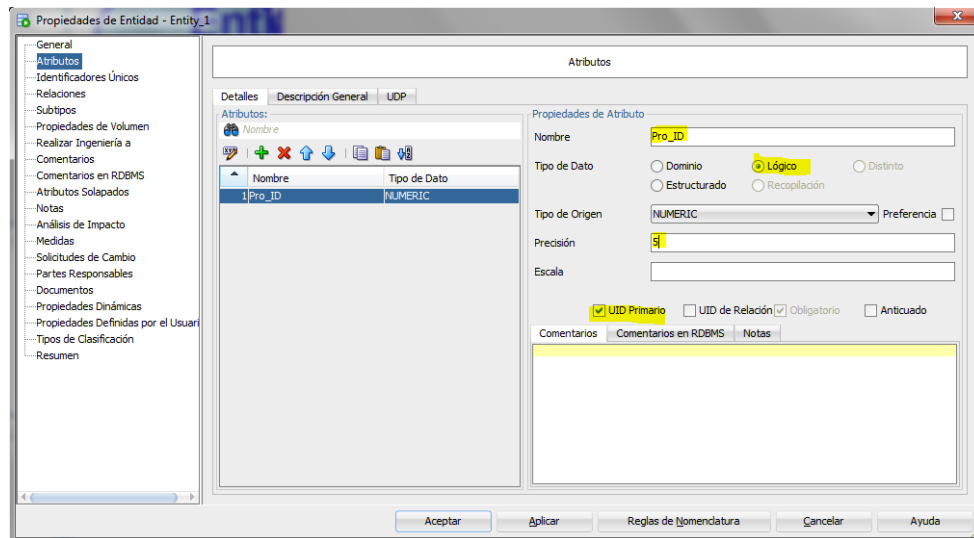
Introducimos el nombre de la entidad, en nuestro caso Usuarios:

Luego vamos a la ficha de Atributos (panel izquierdo) y añadimos los atributos de la tabla, hacemos clic en el signo  para ir añadiendo atributos a la entidad que estamos creando.

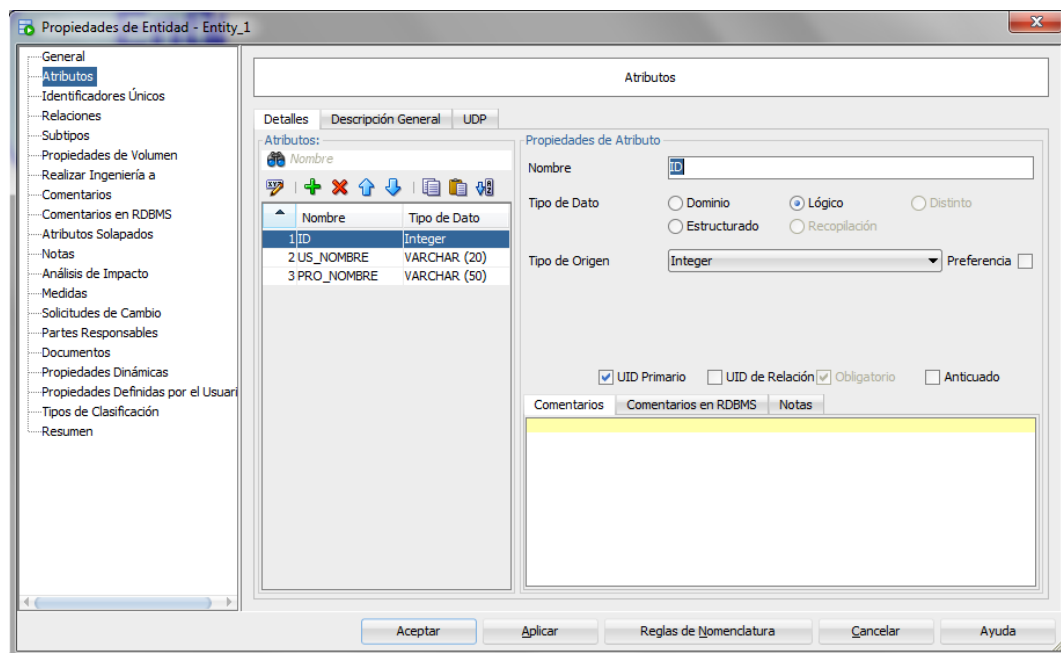
Definimos los siguientes atributos lógicos para la entidad **Usuarios**,

- US_ID (NUMERIC,6) Identificador del usuario. Clave primaria
- US_NOMBRE (VARCHAR 20) Nombre del usuario. Campo obligatorio
- US_PASS (VARCHAR 20) Password. Campo obligatorio

Las definiciones de los campos con la propiedad obligatorio nos permiten cumplir con la calidad de datos para evitar datos incompletos, lo veremos en la función de Calidad de datos.



Desde la misma ventana, añadimos todos los atributos tal y como los hemos definido. La entidad completa quedaría así.



DataModeler permite definir dominios de tipo de datos, es muy útil para mantener la consistencia sobre las tablas que utilizan el mismo tipo de datos.

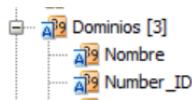
En nuestro ejemplo, vamos a definir 2 dominios nuevos, uno para el identificador de tablas y otro para el nombre.

La definición de dominios se hace desde el menú Herramientas; DataModeler; Administrador de dominios.



Los dominios se aplicarán a todos los proyectos, y se puede importar (.xml) dominios de otros proyectos.

Una vez definidos los dominios, los tendremos disponibles en el Explorador del DataModeler:



Ahora creamos las entidades **Roles y Privilegios**, para ello igual que antes, desde la barra de menú, Nueva Entidad y marcamos en el editor de modelo lógico.

Los atributos lógicos de la entidad Roles son:

- ROL_ID (NUMBER,6) Identificador del rol. Clave primaria
- ROL_NOMBRE (VARCHAR 25) Nombre del rol. Campo obligatorio

Y los de la entidad Privilegios:

- PRIV_ID (NUMBER,6) Identificador del privilegio. Clave primaria
- PRIV_NOMBRE (VARCHAR 25) Nombre del privilegio. Campo obligatorio

Las crearemos en el modelo lógico.



Propiedades de Entidad - Rol

General
Atributos
Identificadores Únicos
Relaciones
Subtipos
Propiedades de Volumen
Realizar Ingeniería a
Comentarios
Comentarios en RDBMS
Atributos Solapados
Notas
Análisis de Impacto
Medidas
Solicitudes de Cambio
Partes Responsables
Documentos
Propiedades Dinámicas
Propiedades Definidas por el Usuario
Tipos de Clasificación
Resumen

Atributos

Detalles Descripción General UDP

Atributos:

Nombre	Tipo de Dato
1 ROL_ID	Number_ID
2 ROL_NOM	Nombre

Propiedades de Atributo

Nombre: ROL_ID

Tipo de Dato: ☒ Dominio ☐ Lógico ☐ Distinto
☐ Estructurado ☐ Recopilación

Tipo de Origen: Number_ID Preferencia ☐

☒ UID Primario ☐ UID de Relación ☒ Obligatorio ☐ Anticuoado

Comentarios Comentarios en RDBMS Notas

Aceptar Aplicar Reglas de Nomenclatura Cancelar Ayuda

Propiedades de Entidad - privilegio

General
Atributos
Identificadores Únicos
Relaciones
Subtipos
Propiedades de Volumen
Realizar Ingeniería a
Comentarios
Comentarios en RDBMS
Atributos Solapados
Notas
Análisis de Impacto
Medidas
Solicitudes de Cambio
Partes Responsables
Documentos
Propiedades Dinámicas
Propiedades Definidas por el Usuario
Tipos de Clasificación
Resumen

Atributos

Detalles Descripción General UDP

Atributos:

Nombre	Tipo de Dato
1 PRIV_ID	Number_ID
2 priv_nom	Nombre

Propiedades de Atributo

Nombre: PRIV_NOM

Tipo de Dato: ☒ Dominio ☐ Lógico ☐ Distinto
☐ Estructurado ☐ Recopilación

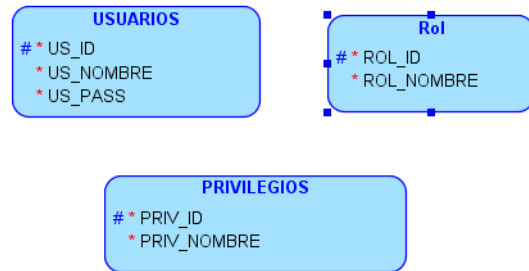
Tipo de Origen: Nombre Preferencia ☐

☐ UID Primario ☐ UID de Relación ☒ Obligatorio ☐ Anticuoado

Comentarios Comentarios en RDBMS Notas

Aceptar Aplicar Reglas de Nomenclatura Cancelar Ayuda

En el editor del modelo lógico podemos ver las entidades que hemos creado.



Ahora nos quedaría crear las relaciones entre las entidades. Un usuario tiene asignado un Rol y sin embargo un Rol se le puede asignar a varios usuarios, es por esto que la relación entre las entidades Roles y Usuarios es de 1:N. Además del nombre de la relación, indicaremos las claves primarias de cada entidad.

Propiedades de Relación - Relation_Rol_Usuario

General

Nombre: Relation_Rol_Usuario

Usar Claves de Sustitución: ☐

Cardinalidad de Origen

Origen: ROLES

Clave de Origen: ROLES.Rol PK

Nombre en Origen:

Sinónimo de Entidad de Origen: ROLES

Cardinalidad de Origen a Destino: 1 *

Origen Opcional: ☒

Transferible: ☒

Rol Dominante: Ninguno

Con Identificación: ☐

Suprimir Regla: NO ACTION

Cardinalidad de Destino

Destino: USUARIOS

Clave de Destino: USUARIOS.USUARIO...

Nombre en Destino:

Sinónimo de Entidad de Destino: USUARIOS

Cardinalidad de Destino a Origen: 1

Destino Opcional: ☐

Transferible: ☒

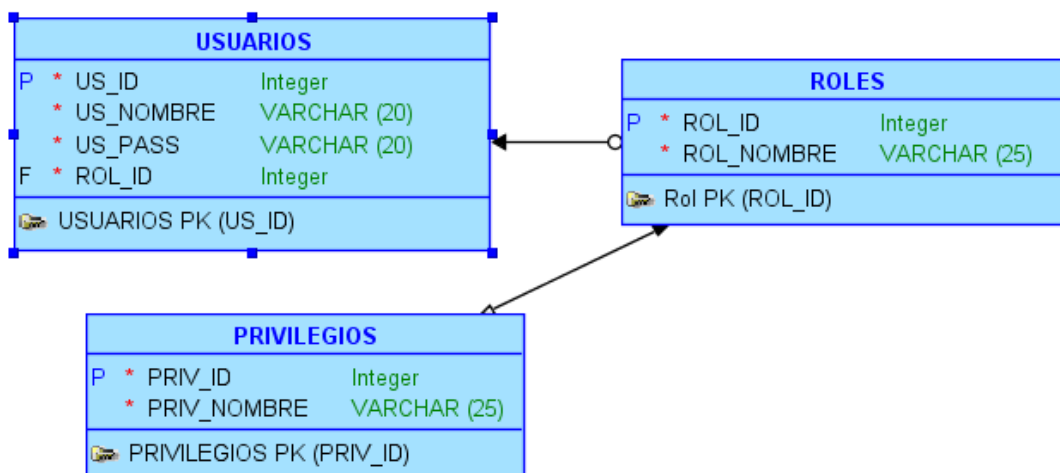
En Arco: ☐

Aceptar Aplicar Cancelar Ayuda

La relación entre Roles y Privilegios es de M:N porque un rol puede tener muchos privilegios y un privilegio puede ser asignado a muchos roles. Además del nombre de la relación, indicaremos las claves primarias de cada entidad.



En el modelo lógico las relaciones M:N quedan representadas, veremos que al pasar al modelo físico estas relaciones se transforman automáticamente en una entidad especial.



Nota: Para cambiar de Notación y ver en el modelo lógico los atributos y las relaciones, en el editor del modelo lógico, pulsar botón derecho; Notación; y cambiar a Notación Bachman.

Ejercicio 2.- Generación del Modelo Físico o E/R .

Una vez que tenemos el modelo lógico con las entidades, sus atributos y las relaciones entre ellas, generaremos el modelo relacional o Entidad-Relación (E/R) a partir de la definición del modelo lógico.


Un modelo E/R representa como el sistema va a ser implementado dentro de una base de datos relacional

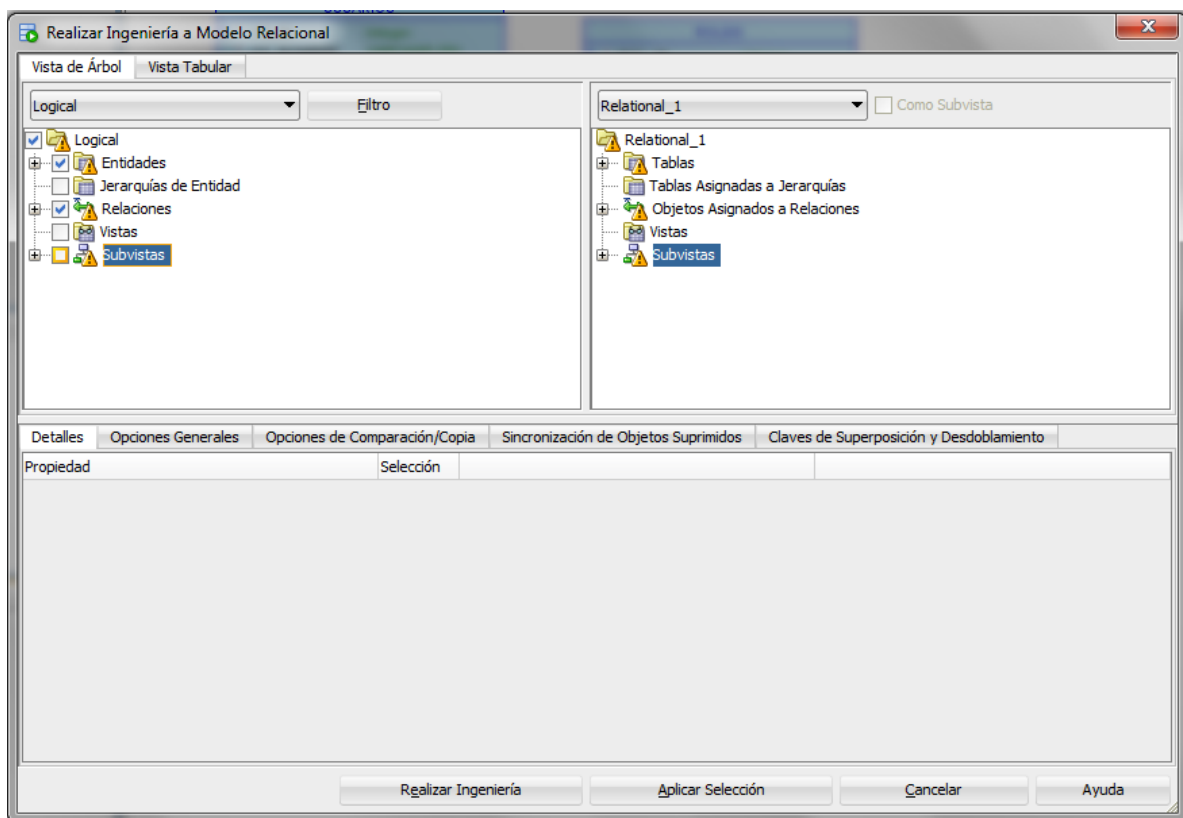


Respecto al modelado, dos de las normas que se deben cumplir y que deberán estar incluidas en cualquier definición de la política de gestión de datos de una organización son:

1. Toda tabla tiene que tener clave primaria.
2. No es aceptable ningún modelo físico que no esté, al menos, en tercera forma normal. Esto significa que no podrán existir relaciones N:N en el modelo relacional.

La generación automática del modelo físico basado en un modelo lógico, nos facilita el cumplimiento de estas normas indispensables.

Para esto existe una utilidad en Data Modeler, que es “Realizar Ingeniería a Modelo Relacional”, este sería el icono  en la barra de herramientas.

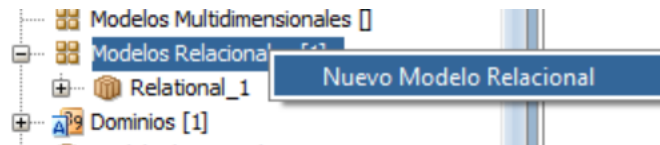


Seleccionamos realizar solo la ingeniería de las Entidades y las Relaciones del modelo lógico al modelo E/R, y pulsamos Realizar Ingeniería.

Como resultado se muestra el modelo relacional que ha se generado automáticamente.

En un diseño sólo puede haber modelo lógico, pero pueden generarse varios modelos físicos. Vamos a crear un nuevo modelo físico, pero aplicando un glosario de términos que nos permita estandarizar nombres en nuestros sistemas.

Para ello, en el Explorador; Modelos Relacionales; Nuevo Modelo Relacional



Ejercicio 3.- Creando un –Glosario IT

Un modelo de glosario es un modelo que describe los nombres y las abreviaturas para los objetos de datos. Un modelo de glosario también define palabras principales (como Empleado o Empresa), palabras de clase (como ID de empleado o nombre) y modificadores (como Primero o Anual).

En ocasiones, al trabajar con diversas personas en el mismo proyecto, la denominación de los objetos de datos varía. Es posible que un arquitecto de datos hable de las entidades de ventas como SALES_x y que otro modelador de datos se refiera a las mismas como SLS_x. Con el paso de tiempo, la comprensión se acaba dificultando y la incoherencia puede afectar a diversos sistemas, lo que provoca confusión, prolonga los tiempos de desarrollo o, incluso, impide establecer conexiones entre dos campos en dos sistemas distintos que significan lo mismo.

Para evitar este problema, es necesario definir un conjunto de normas al principio del proceso de diseño de datos que ayudarán a los miembros de una organización a buscar datos de forma coherente y a identificar términos con el mismo significado.

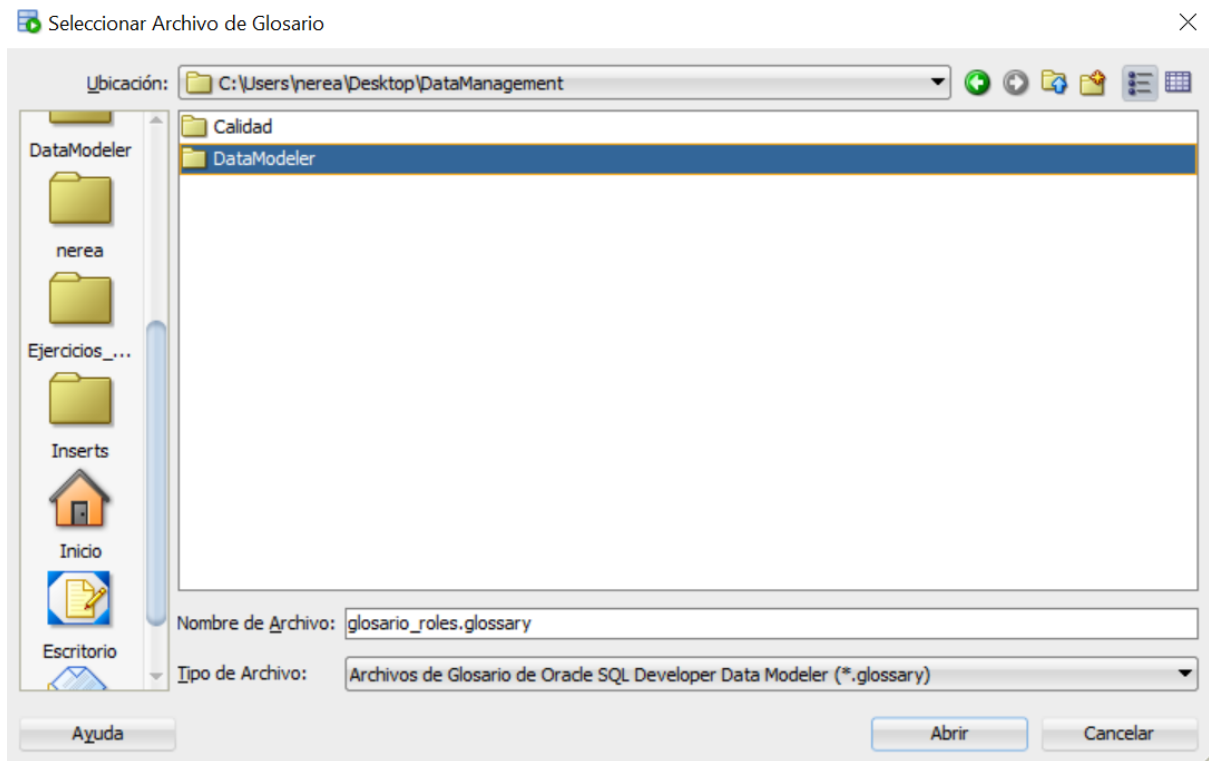
Las normas de denominación o de nomenclatura en los objetos de datos aumentan la comprensión común de los datos, ya que se utilizan las mismas convenciones de nombres en toda la organización. Se puede utilizar modelos de glosario para habilitar la puesta en común de los datos entre las distintas secciones y reducir la redundancia de datos mediante la consolidación de los elementos de datos sinónimos y coincidentes.

Data Modeler y en relación con los glosarios se puede:

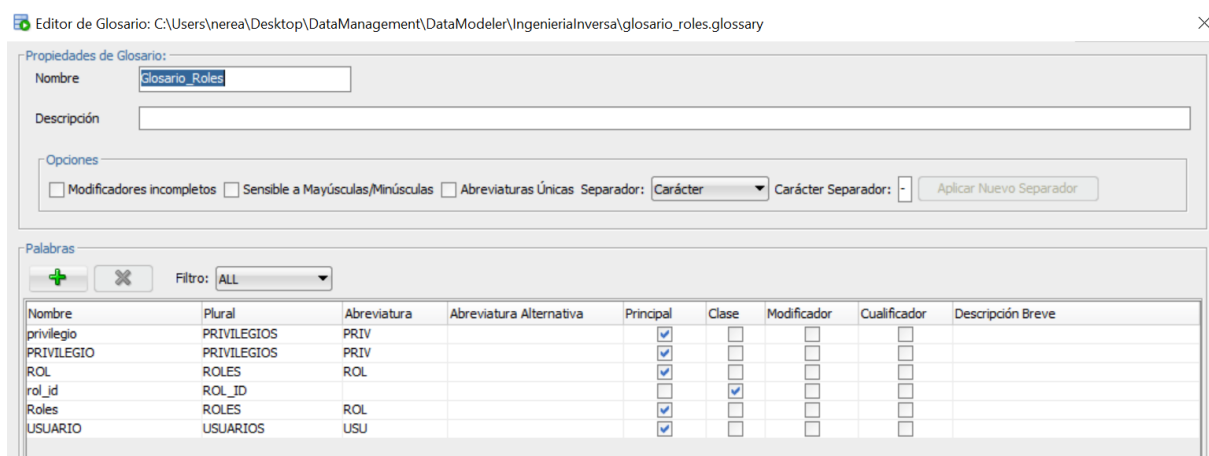
- Crear un modelo de glosario desde cero, se guardarán con extensión. glossary
- Importar glosario desde ficheros planos (txt, csv) o Microsoft Excel
- Exportar el glosario a formato csv o xls.

Crearemos un glosario para aplicar unas normas de denominaciones sobre los objetos en los modelos lógicos. Ir a Herramientas; Data Modeler; Editor Glosario.

Lo primero que nos pedirá es un nombre para el archivo de glosario, lo llamaremos glosario_rols.glossary



Lo siguiente será crear la relación de palabras de nuestro glosario.



Cuando se asigna un nombre a un objeto de datos, hay que tener en cuenta dos elementos: las reglas semánticas y el formato. Desde el punto de vista semántico, los objetos de modelo de glosario incluyen palabras principales, palabras de clase y modificadores:

Nombres

Los nombres se utilizan para describir los objetos. Se utiliza el lenguaje común, como Nombre de cuenta o Número de ID de empleado, para describir estos tipos de datos.

Abreviaturas

Las abreviaturas son nombres cortos que se quieren utilizar para los objetos del modelo. Por ejemplo, se puede utilizar la abreviatura y la abreviatura alternativa



EMP para el objeto EMPLEADO que modelan la información relacionada con los datos de los empleados.

Palabras principales

Las palabras principales son palabras que representan el concepto empresarial para el que se están recopilando datos. Las palabras principales son nombres que describen el área temática de los datos. Por ejemplo, la palabra principal PRÉSTAMO especifica que todos los objetos de datos que utilizan esta palabra principal están relacionados con el préstamo.

Palabras de clase

Las palabras de clase son palabras que identifican una propiedad, categoría o clasificación de datos diferenciada. Por ejemplo: TARIFA, NOMBRE

Modificadores

Los modificadores son palabras que identifican o distinguen con mayor detalle las palabras principales o de clase. Permiten que los nombres de objeto sean transparentes y exclusivos, y pueden ayudar a restringir el significado de las palabras principales o secundarias. Por ejemplo: SIGUIENTE, PRIMERO

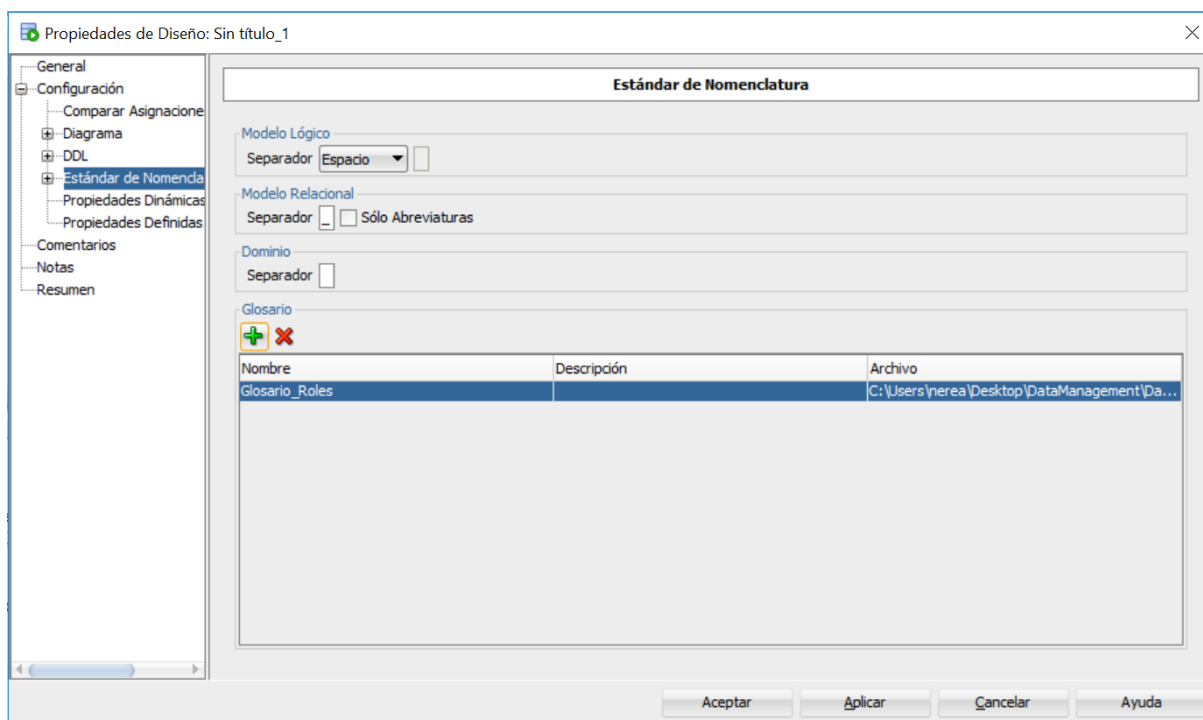
Con la opción Guardar, y Cerrar

Ejercicio 7.- Asociación del-Glosario con el modelo

El siguiente paso será relacionar el Glosario con el modelo. Para ello seleccionaremos el diseño que se llamará "Sin titulo_1" si no se ha guardado todavía, en el árbol de la ventana izquierda y botón derecho para visualizar el menú contextual, elegiremos Propiedades.

La configuración del Separador es importante: determina si los nombres se tratarán como una o más palabras.

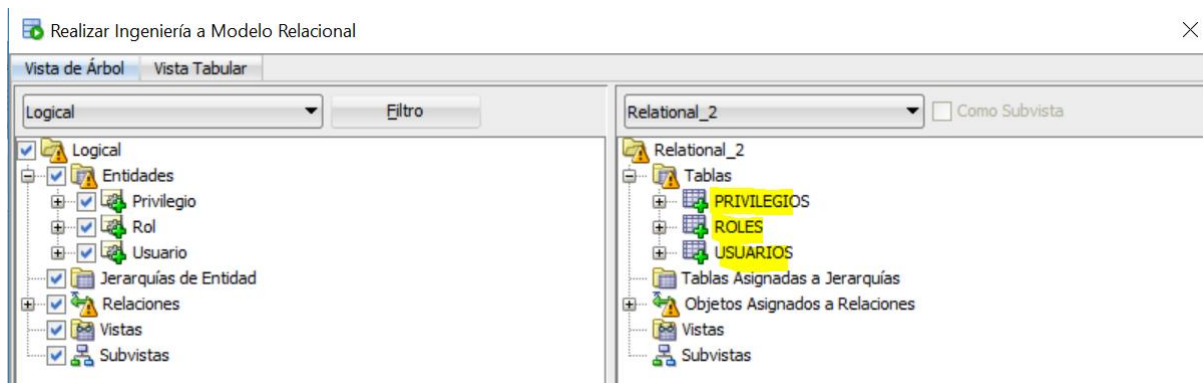
Haremos click en  para elegir el glosario guardado en el ejercicio anterior.

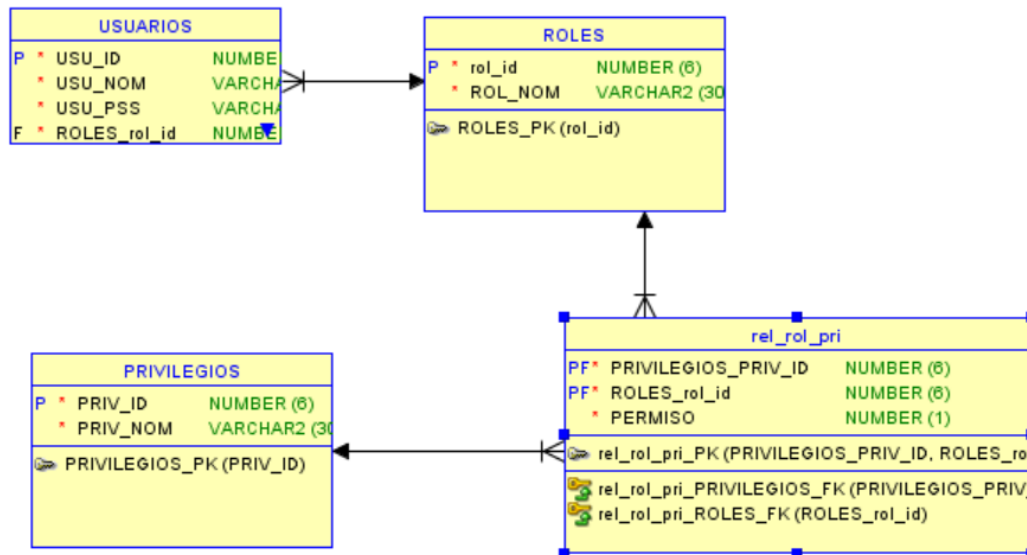


Pulsaremos el botón Aplicar.

Aunque tengamos el glosario de palabras creado y asociado por defecto cuando se crea una nueva entidad o tabla en el diseño de los modelos, no me va a dar un error si los nombres no cumplen con las reglas de nombres, sólo se aplican si se las reglas de diseño relacionadas con la nomenclatura.

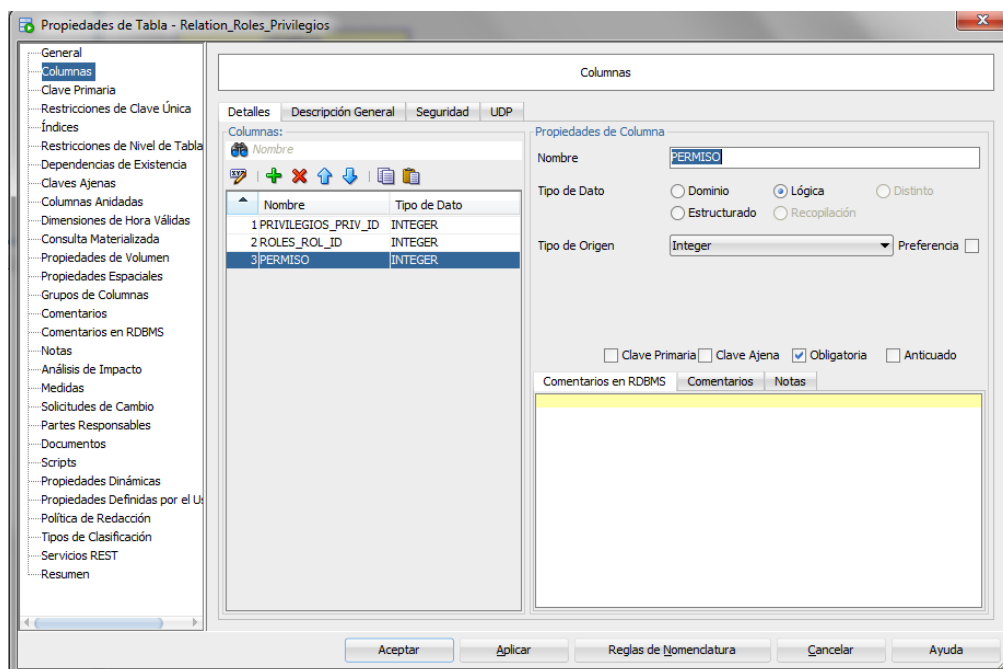
Si ahora realizamos una ingeniería inversa desde el modelo lógico veremos que en la transformación se aplican las definiciones realizadas en el glosario, ya que los nombres de las tablas ahora aparecen en mayúsculas.



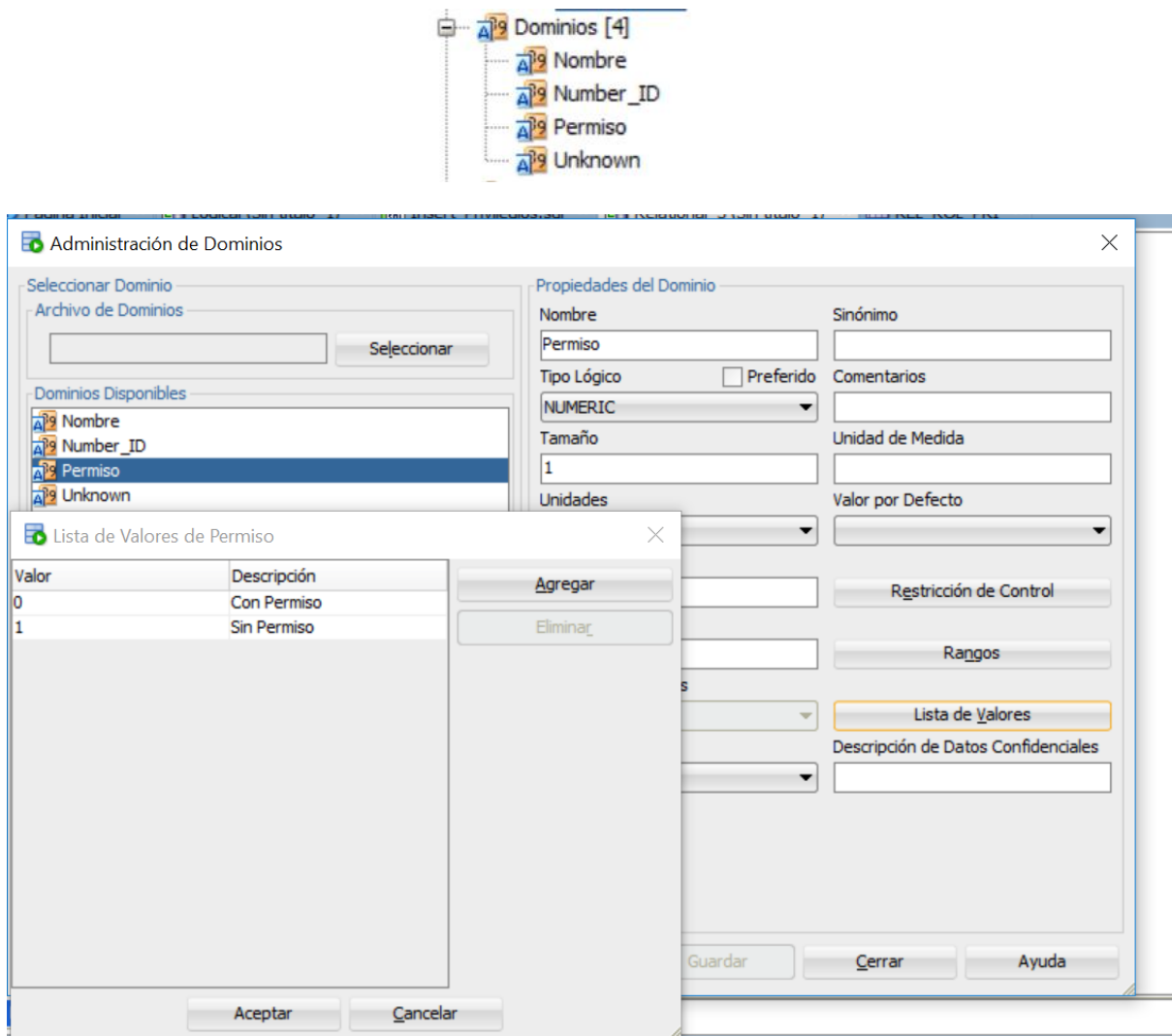


Podemos comprobar que la relación M:N entre ROLES y PRIVILEGIOS del modelo lógico, en el modelo relacional ha generado una nueva entidad Relation_Roles_Privilegios, con las claves de ambas entidades y la relación entre esta nueva entidad y las entidades ROLES y PRIVILEGIOS es de 1:N.

Vamos añadir un nuevo atributo a esta nueva entidad, que es PERMISO, para indicar el tipo de permiso.



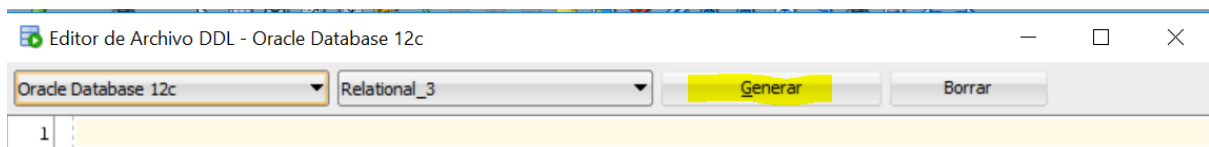
Podemos también crear un nuevo dominio para el campo Permiso y especificar una lista de valores, para garantizar la integridad de los y así aplicar una restricción en la inserción de datos en el campo y evitar que se introduzcan datos erróneos.



Ejercicio 3.- Generación de script.

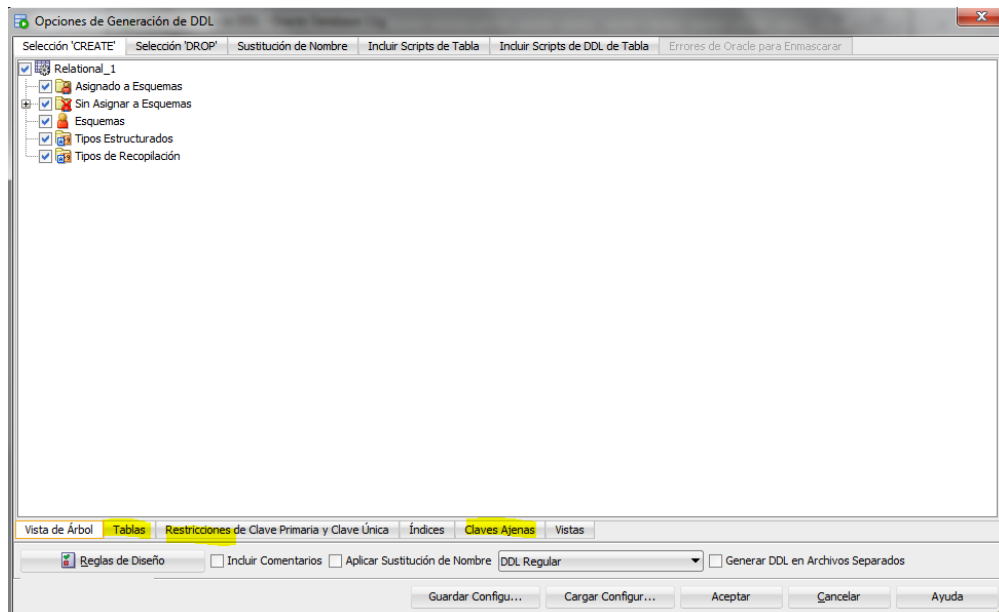
Para terminar lo que haremos es generar el script con las sentencias DDL (Data Definition Language) que nos permitan crear el modelo físico relacional que acabamos de crear.

Se utilizará el modelo físico generado anteriormente.

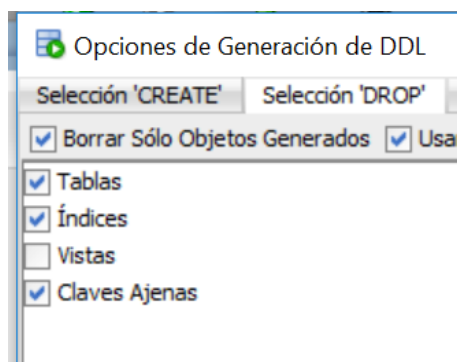


Pulsaremos al botón Generar, para mostrar los objetos del modelo físico que se utilizarán para crear las sentencias SQL para su creación en la base de datos.

En la pantalla se muestran los objetos del modelo físico en forma de árbol en la primera ficha. También se pueden ver las Tablas, Claves Primarias (PK) o Restricciones y Claves Ajenas o Foráneas (FK).



Incluimos también las sentencias de borrado de tablas, por si tenemos que ejecutar el script varias veces.



Pulsaremos en Aceptar para generar el script automáticamente.

El script generado se nos muestra en pantalla.




```
1  -- Generado por Oracle SQL Developer Data Modeler 17.4.0.355.2131
2  -- en:      2019-07-01 11:34:48 CEST
3  -- sitio:   Oracle Database 12c
4  -- tipo:    Oracle Database 12c
5
6
7
8  DROP TABLE privilegios CASCADE CONSTRAINTS;
9
10 DROP TABLE rel_rol_pri CASCADE CONSTRAINTS;
11
12 DROP TABLE roles CASCADE CONSTRAINTS;
13
14 DROP TABLE usuarios CASCADE CONSTRAINTS;
15
16 CREATE TABLE privilegios (
17     priv_id    NUMBER(6) NOT NULL,
18     priv_nom   VARCHAR2(30) NOT NULL
19 );
20
21 ALTER TABLE privilegios ADD CONSTRAINT privilegios_pk PRIMARY KEY ( priv_id );
22
23 CREATE TABLE rel_rol_pri (
24     privilegios_priv_id  NUMBER(6) NOT NULL,
25     roles_rol_id         NUMBER(6) NOT NULL,
26     permiso              NUMBER(1) NOT NULL
27 );
28
29 ALTER TABLE rel_rol_pri
```

Guardaremos el script generado, pulsando en Guardar. Lo guardaremos con extensión sql para poder ejecutar, lo llamaremos “Roles_Script.sql”.

Ejercicio 4.- Creación de objetos desde script.

Nos conectaremos a la base de datos stage y ejecutaremos el script “Roles_Script.sql”. Con el menú principal Archivo; Abrir.

Para ejecutar, pulsaremos al botón Ejecutar Script  nos solicitará la base de datos donde se creará el modelo físico. El resultado será la creación de todas las entidades, con sus atributos, claves primarias y foráneas, así como las relaciones, tal y como se han definido.



```
-- Generado por Oracle SQL Developer Data Modeler 17.4.0.355.2131
-- en: 2019-07-01 11:18:47 CEST
-- sitio: Oracle Database 12c
-- tipo: Oracle Database 12c

DROP TABLE privilegios CASCADE CONSTRAINTS;

DROP TABLE rel_rol_pri CASCADE CONSTRAINTS;

DROP TABLE roles CASCADE CONSTRAINTS;

DROP TABLE usuarios CASCADE CONSTRAINTS;

CREATE TABLE privilegios (
  priv_id NUMBER(6) NOT NULL,
  priv_nom VARCHAR2(30) NOT NULL
);

ALTER TABLE privilegios ADD CONSTRAINT privilegios_pk PRIMARY KEY ( priv_id );

CREATE TABLE rel_rol_pri (
  privilegios_priv_id NUMBER(6) NOT NULL,
  roles_rol_id NUMBER(6) NOT NULL,
  permiso NUMBER(1) NOT NULL
```

Con estos ejercicios hemos aprendido a:

- Diseñar el modelo lógico o conceptual
- Generar modelos entidad relación (E/R)
- Crear el modelo físico mediante ejecución de script generado automáticamente a partir del modelo físico.
- Definición de dominios de tipos de datos, con listas de valores

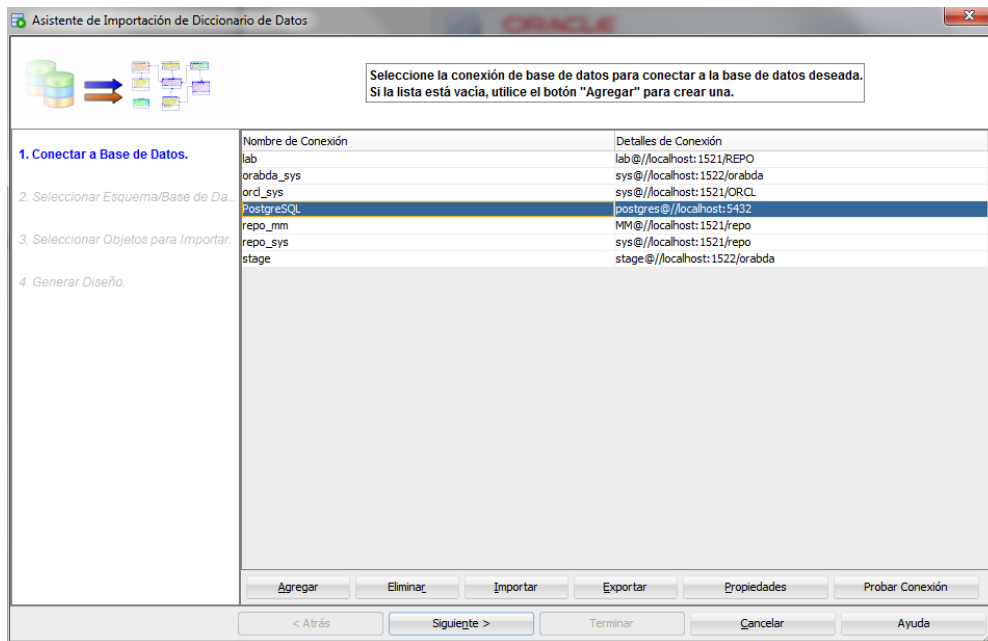
Como tenemos el modelo físico podemos insertar datos a las tablas que hemos creado, para ellos ejecutamos el script SQL Insertar_datos.sql aportado como parte del material.

Ejercicio 5.- Ingeniería Inversa de PostgreSQL.

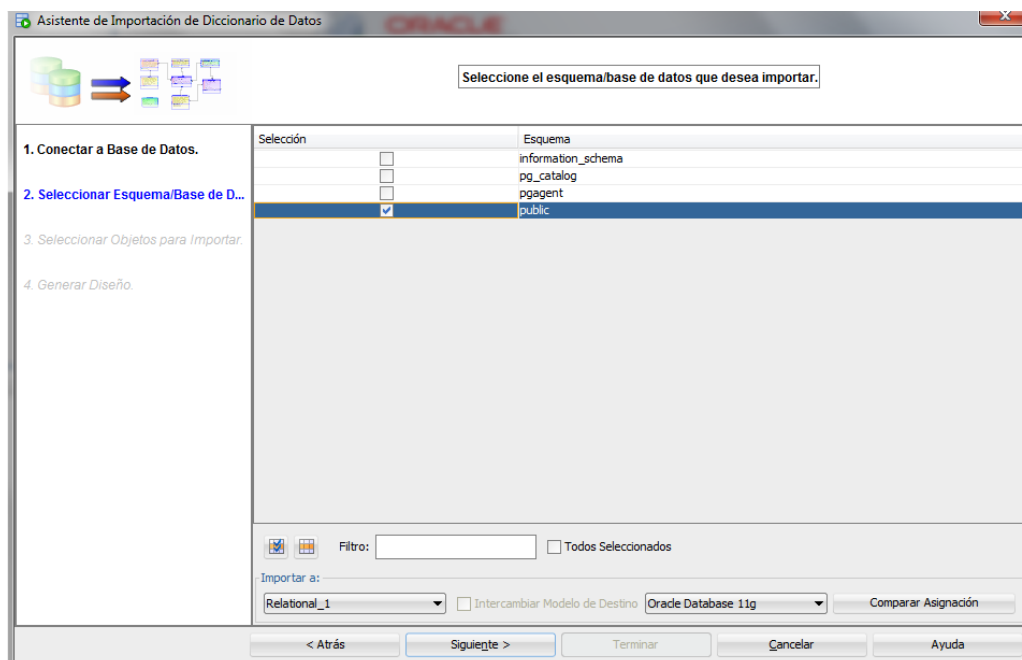
En muchas ocasiones tendremos que trabajar con bases de datos de las que no dispongamos los modelos lógico y físico, imprescindibles para el gobierno TI. En estos casos, no veremos obligados a utilizar los diccionarios de datos para descubrir las tablas y campos, esta técnica se denomina ingeniería inversa.

Cerraremos el modelo de datos utilizado en los ejercicios anteriores.

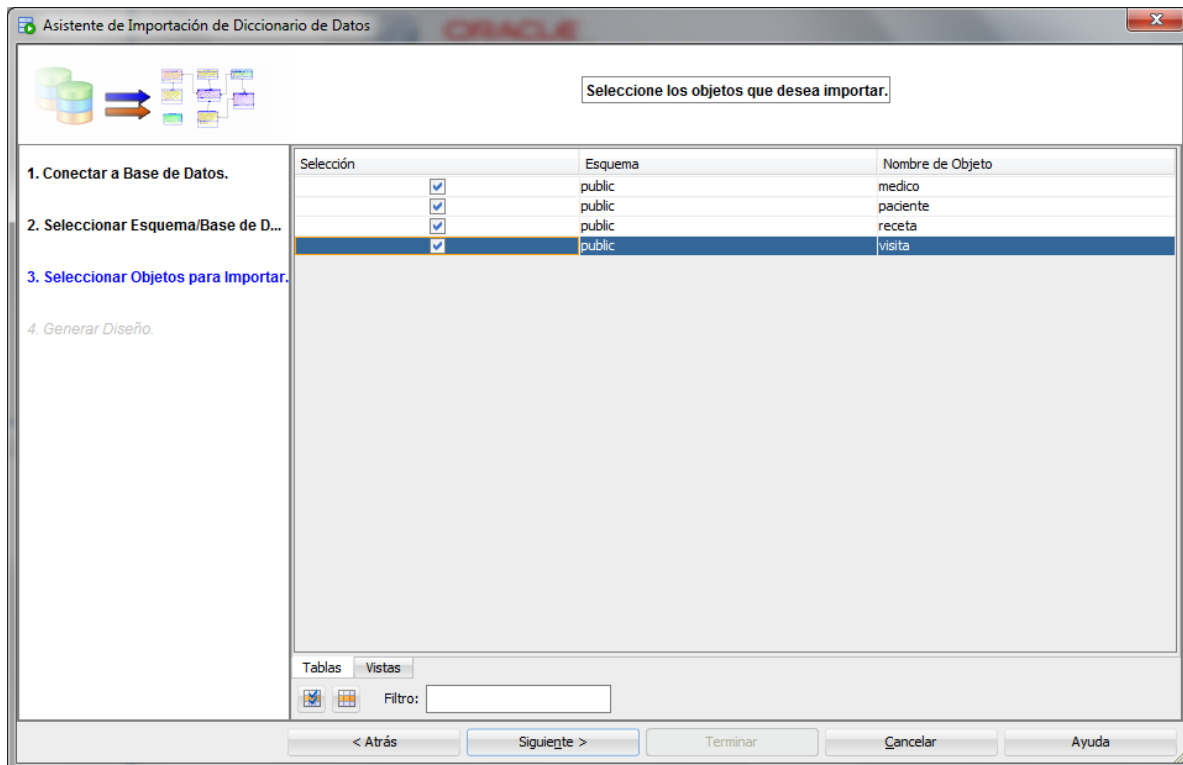
Vamos a realizar la ingeniería inversa de la base de datos PostgreSQL, para ello vamos a Archivo; Data Modeler; Importar; Diccionario de Datos, y elegimos la base de datos de la que queremos realizar la ingeniería inversa. En nuestro caso progresql y pulsamos Siguiente.



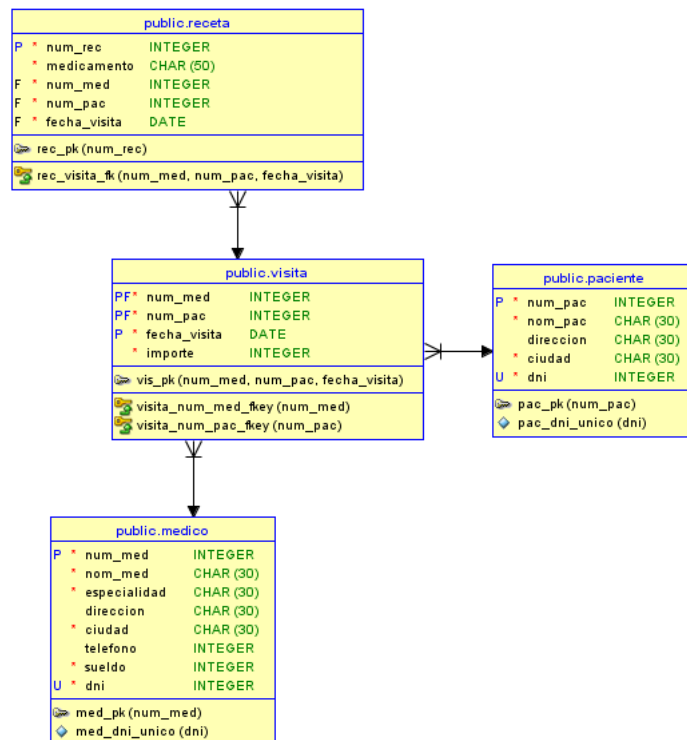
Ahora debemos elegir el esquema de la base de datos y pulsamos Siguiente



El siguiente paso es seleccionar las tablas que queremos que aparezcan en el Diagrama E/R . Una vez seleccionados, pulsamos "Siguiente" y a continuación "Terminar".



El resultado es el modelo relacional de la base de datos utilizando el diccionario de datos.

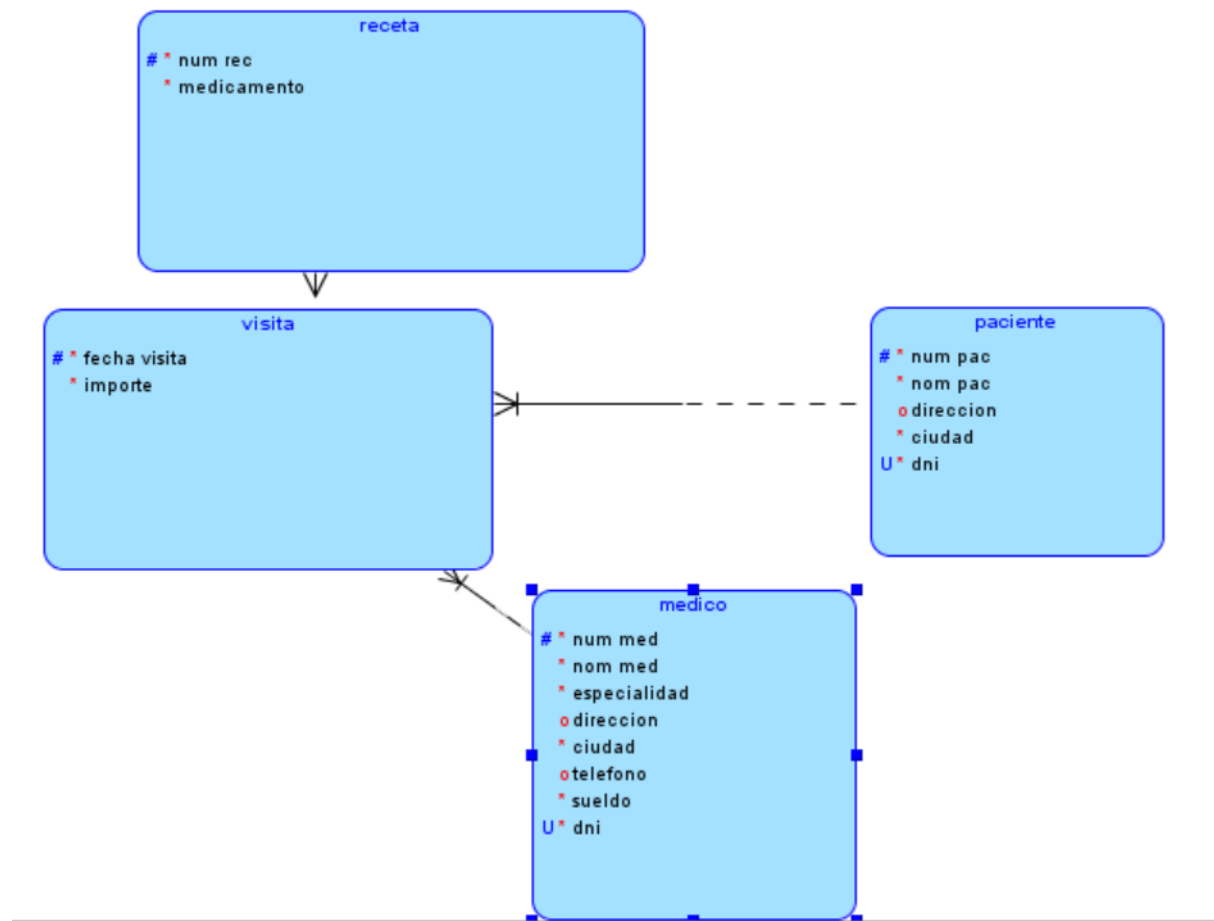


Una vez disponemos del Diagrama Entidad/Relación, podemos exportarlo o imprimirlo. Para ello, "Archivo"-> "Data Modeler" -> "Imprimir Diagrama" y seleccionamos el formato de salida que queremos.



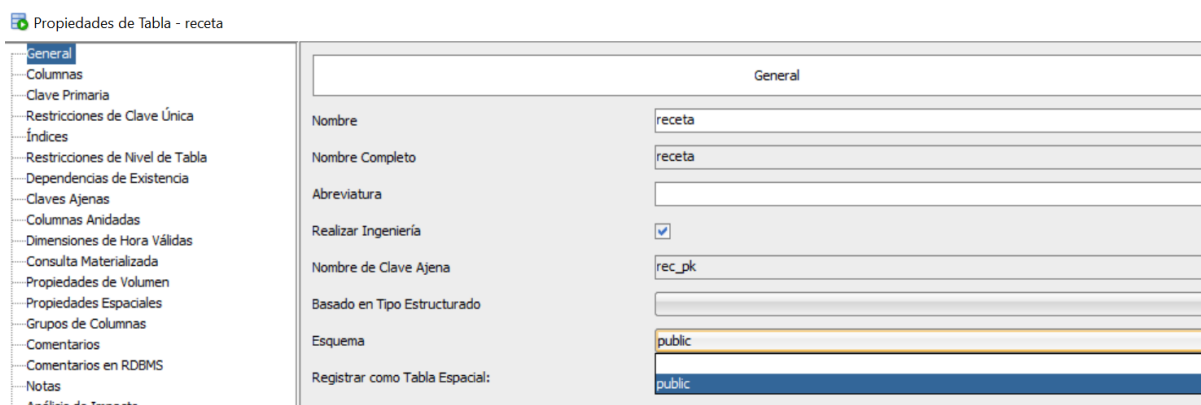
Para generar los informes con la definición de las estructuras de las tablas y/o vistas del modelo físico o entidades del modelo lógico se hará mediante la opción Archivo; Data Modeler; Informes.

Podemos generar el modelo lógico de datos.



Debemos cambiar las propiedades en el modelo físico generado antes de crear el script DDL de construcción de modelo E/R

Quitaremos el nombre del esquema 'public' en todas las tablas, dado que no existe. Para ello doble clic en cada tabla del modelo, dejándolo en blanco.





Para adaptar los tipos y tamaños de PostgreSQL a Oracle, cambiaremos una propiedad de la tabla visita y receta, cambiando el campo de Date a Varchar(10) , ya que en PostgreSQL el tratamiento de fechas es diferente que en Oracle.

Nombre	Tipo de Dato
1 num_med	INTEGER
2 num_pac	INTEGER
3 fecha_visita	VARCHAR2 (10)
4 importe	INTEGER

Tipo de Dato: ☐ Dominio ☒ Lógica ☐ Distinto
☐ Estructurado ☐ Recopilación

Tipo de Origen: VARCHAR Preferencia ☐

Tamaño: 10

Unidades:

Columnas:

Nombre	Tipo de Dato
1 num_med	NUMBER (10)
2 num_pac	NUMBER (10)
3 fecha_visita	VARCHAR2 (10)
4 importe	NUMBER (10)

Propiedades de Columna

Nombre: fecha_visita

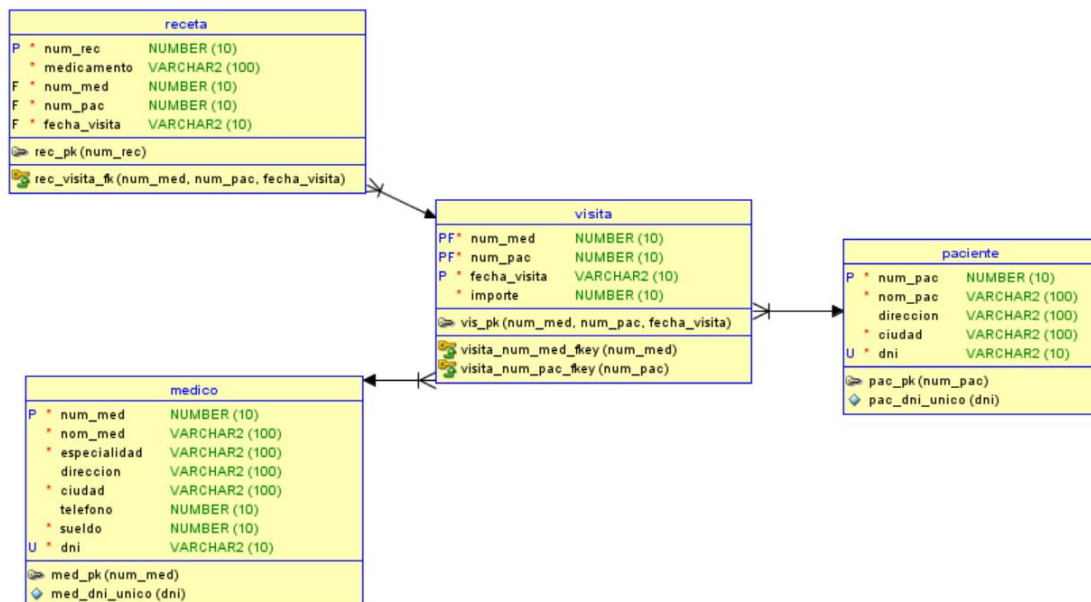
Tipo de Dato: ☐ Dominio ☒ Lógica ☐ Distinto
☐ Estructurado ☐ Recopilación

Tipo de Origen: VARCHAR Preferencia ☐

Tamaño: 10

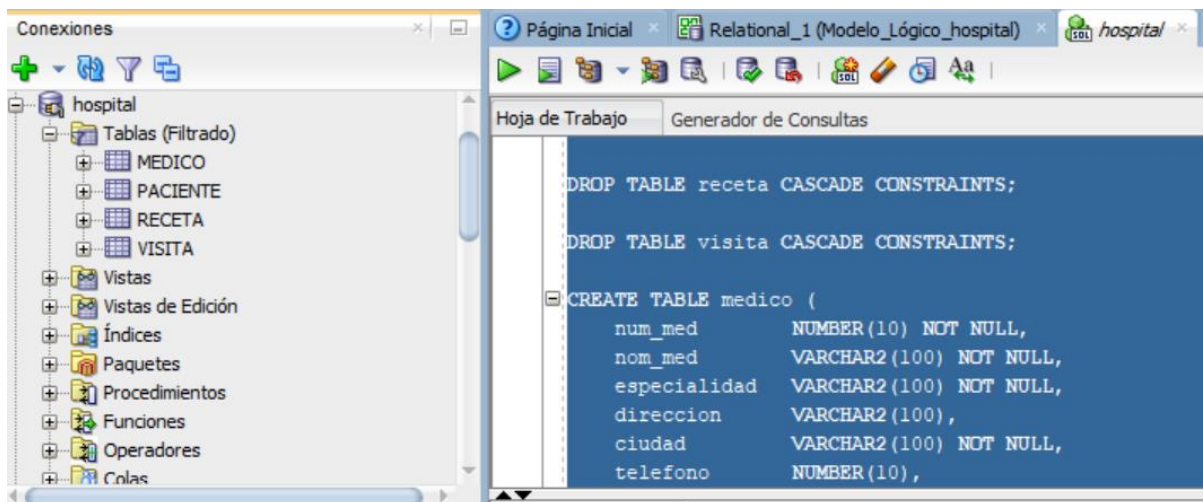
Unidades:

Y los tipos integer por numeric(10) y los char(30) por varchar2(100), el resultado del modelo físico E/R es:

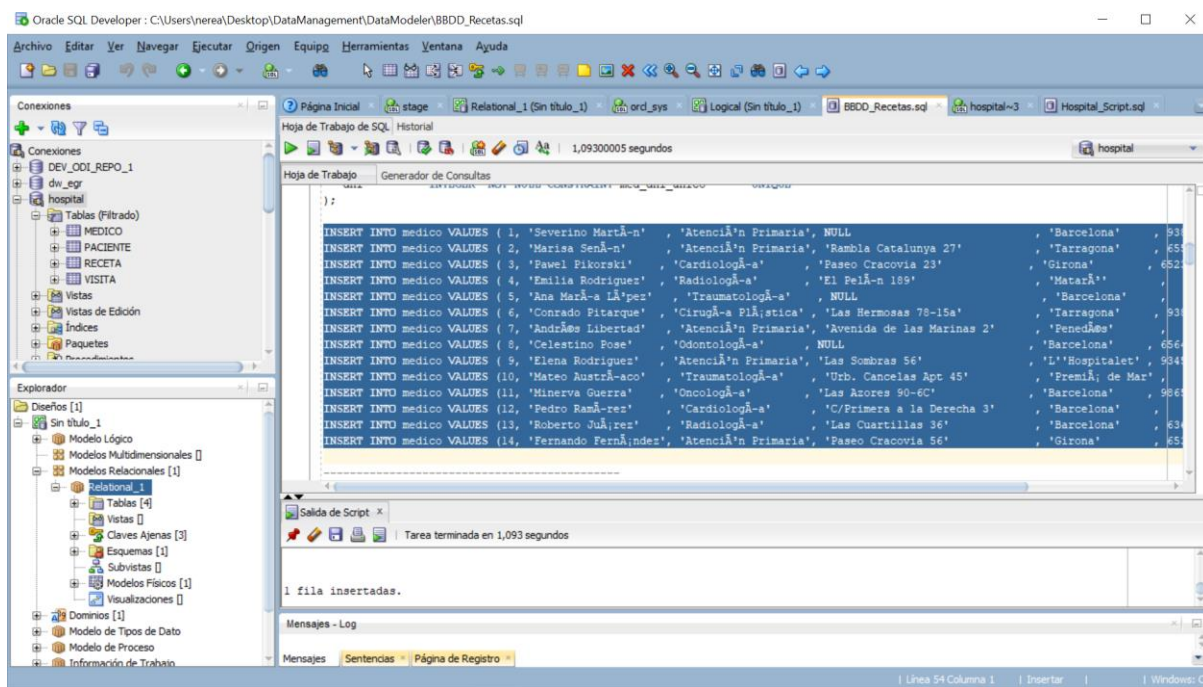


Ahora, podríamos repetir los ejercicios 3 y 4, para a partir del modelo relacional generar el script DDL que nos permita crear los objetos en el esquema "hospital".

Guardaremos el script con el nombre Hospital_Script.sql y lo ejecutaremos en el esquema hospital (las sentencias DROP, darán error porque todavía las tablas no existen)



Ahora podemos ejecutar las sentencias insert del script BBDD_Recetas.sql para insertar los datos.



Con este ejercicio mediante la técnica de ingeniería inversa de DataModeler hemos creado una base de datos Oracle con la estructura y datos de una base de datos PostgreSQL.

Vamos a ejecutar el script Hospital_script.slq también en esquema stage, para realizar los ejercicios de ODI.