

# Tutorial for Basic Data Modeler

This document is a brief description of how to use this software to make simple designs entity-relationship, convert them to relations (tables) and generate the necessary scripts to create the tables in an Oracle database. This document is not intended to be exhaustive, so your own Oracle manuals can be downloaded from Oracle's website to facilitate the 1st practice session.

## 1. Data Modeler

SQL Developer Data Modeler is a design tool for modelling databases that provides an environment to capture, model, manage and use metadata. Can be downloaded free from the website of Oracle ([SQL Data modeler](#)) .

The Data Modeler allows simple ER designs (logical model), translate them into tables (relational model) and finally generate the SQL script to create the database. Likewise, it allows you to directly introduce relational models of complex ER designs and generate SQL scripts. Fig. 1 shows a screenshot of the home screen models. The window on the left shows the tools (marked in red in the figure) necessary to enter ER designs and relational models.

The following example describes the necessary steps to get the code to generate a SQL database from an entity-relationship diagram defined by the Data Modeler.

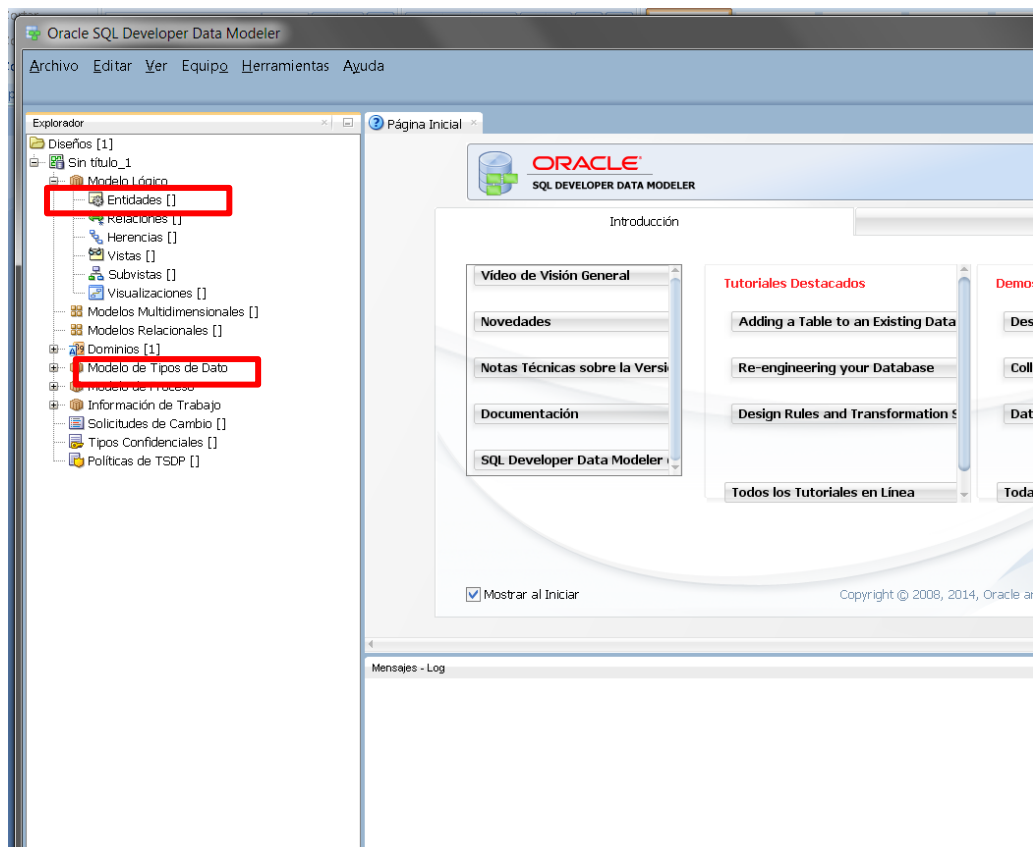
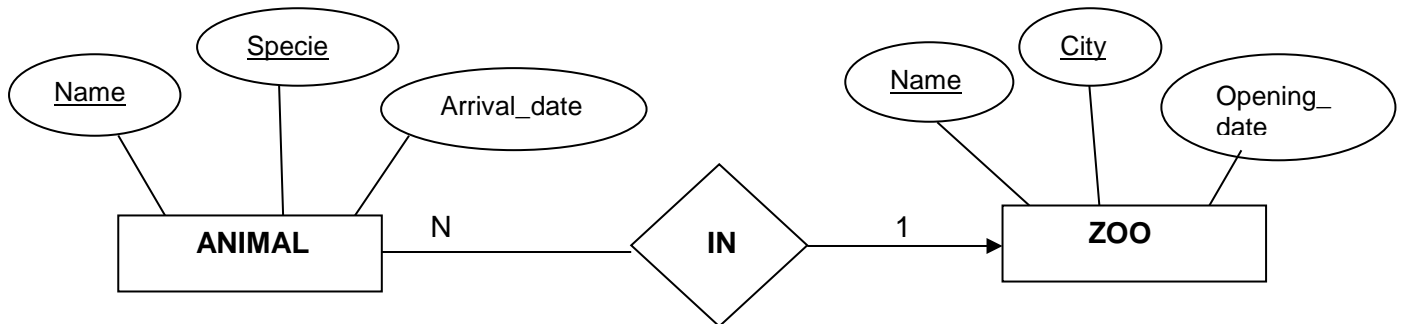


Fig.1 Screenshot of the main screen of your SQL Data Modeler

### Example: a Zoo

The pupils in a school want to make a database of all zoos in Catalonia, in order to have a list of the different animals in each of them. The database should record every animal's name, species and date of arrival to the zoo, the name of each zoo, the city where it is located and the date of inauguration.



Make the ER diagram.

**1.1 ER Diagram Creation:** The first step is to define all the entities involved in the database. In this case you should create two entities, "Animal" and "Zoo". The relationship that links these organizations could be called "Belongs\_to", as animals belong to the zoo. How could we incorporate this information into Data Modeler? The first step is to enable the visualization of the logic model (ER). To do this we tick on the square labelled "Visible" and activate the logical model as shown in Fig.2.

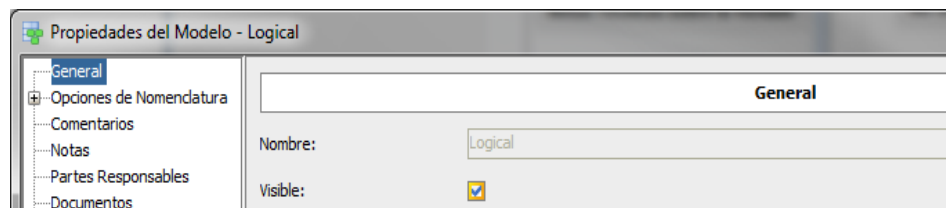


Fig.2 Activation of the display of logic models

**1.1.1 Creating entities:** To start creating the entities using Data Modeler's main menu, we press the icon shown in Fig. 3 (in the red box):

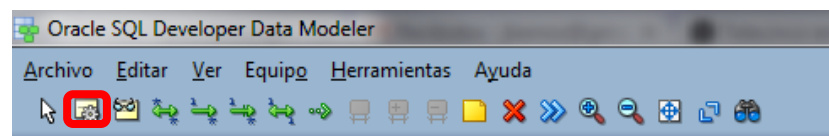


Fig.3 Creation of new entities

Using your mouse, draw a rectangle to represent the entity. You will see a dialog like that of Fig. 4 where you can enter the name of the entity and describe its attributes:

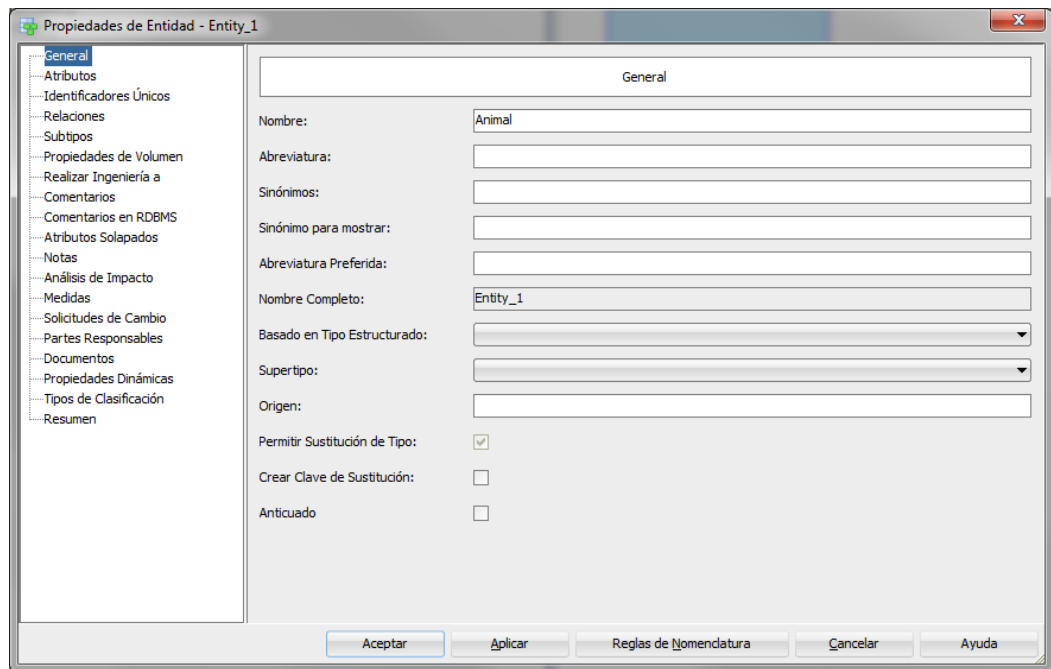


Fig. 4 Dropdown menu of creating entity

Draw another rectangle to create the second entity and describe its attributes

In our case, both entities would have two attributes: Animal would include the name of the animal and the date of arrival at the zoo, and Zoo should include the name and date of inauguration. To specify the domain (type) of each attribute you should open the attribute's dialog and set the attributes type (**Tipo de Dato**) as **lógico**. You must specify the particular type of data in the **Tipo** drop-down menu. Attributes that belong to the primary key (which can be many), are specified by checking the **UID primario** field. Fig. 5 shows an example of how the attribute "number" and animal's "species" are declared as **char** and CP (**UID Primario**) of the entity.

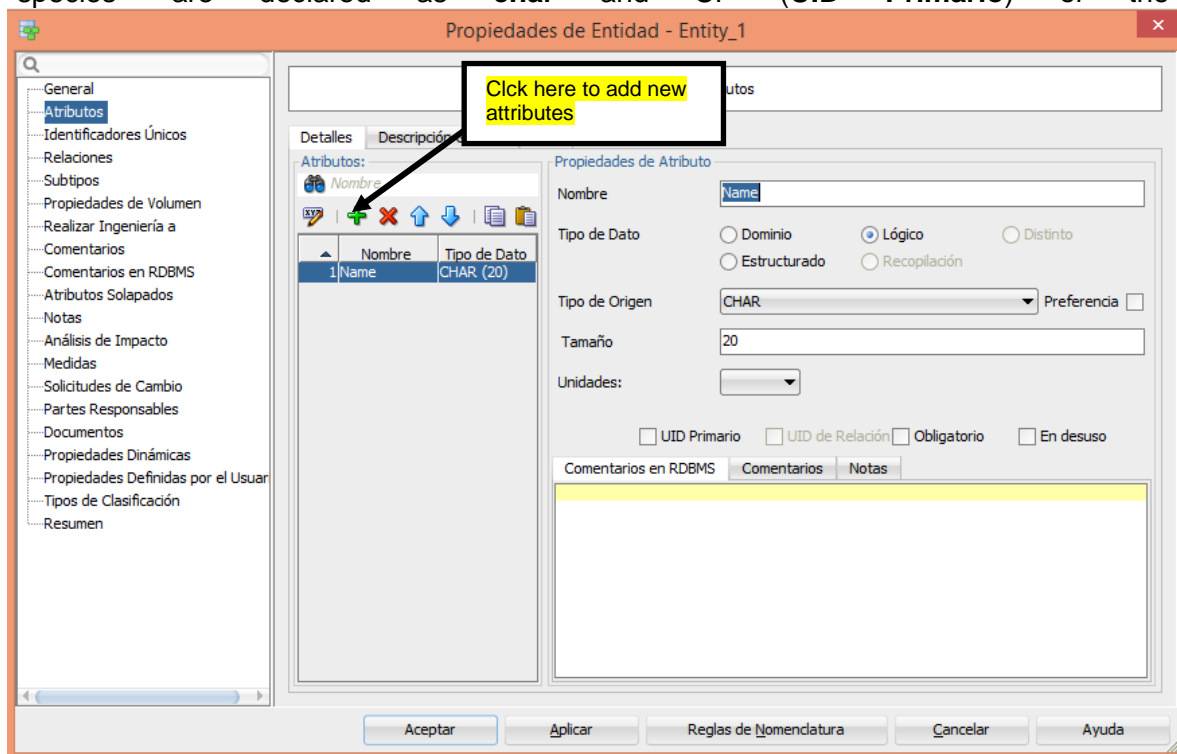


Fig.5 Introduction of an attribute that belongs to the primary key

Once the two entities are defined in the ER diagram, they will be shown as boxes (see Fig. 6).

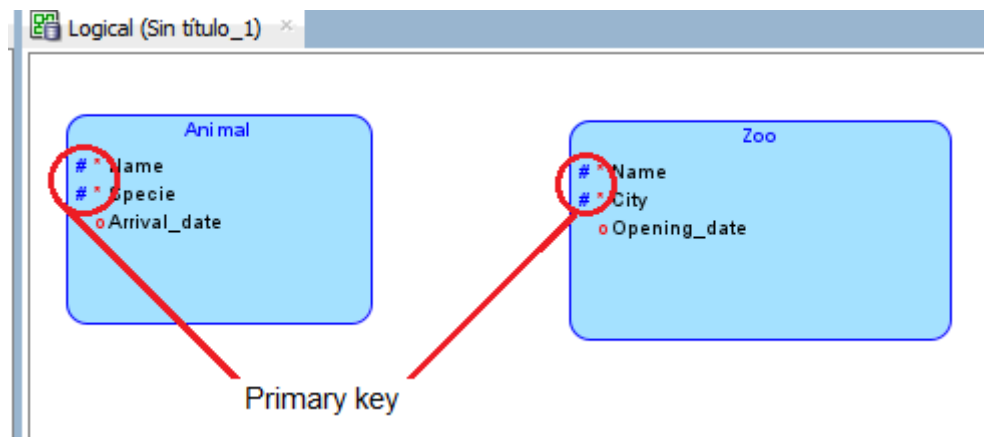


Fig. 6 Entities defined as primary key

1.1.2 *Relationships creation*: We should now decide what should be the relationship between these two entities. At this point we must consider that Data Modeler supports only binary relationships without attributes. If these are 1-N or N-1 remember their attributes can be passed to the entity N. If they are N-N, they should be incorporated as a new table in the final diagram created by Data Modeler.

In the Zoo database we will have a belonging ratio of 1 to N, i.e. a zoo can have many animals but an animal can only belong to one zoo. How can you do this with Data Modeler? In the toolbar to press the button corresponding to relations 1 to N as shown in Fig. 7:

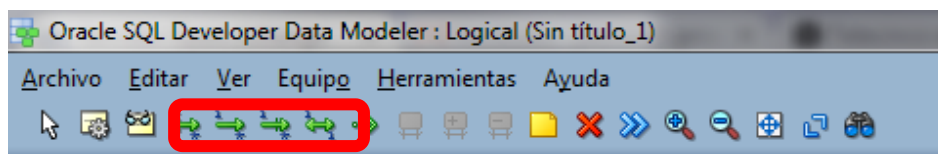


Fig. 7 Establishing relationships between entities

The entity-relationship diagram would end as shown in Fig. 8

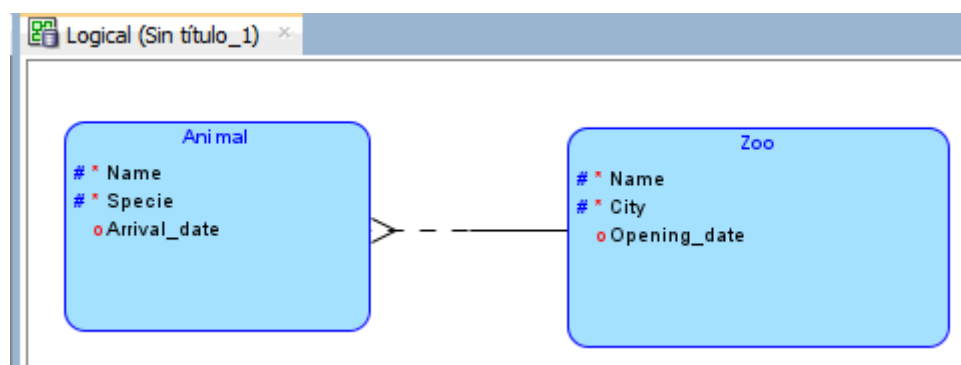


Fig.8 Final Entity Relationship Diagram

1.2 **Creation of the relational model**: at this point we will create a table structure.

1.2.1 Select the **logical model** of the project tree, Fig. 9:

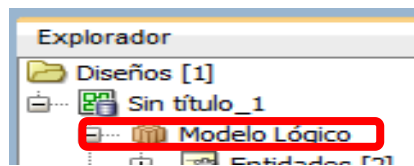


Fig.9 Selecting the ER diagram you want turn into tables

1.2.2 Right-click the mouse on the logical model to be converted, and select the option **Realizar Ingeniería a Modelo Relacional**, as shown in Fig.10.

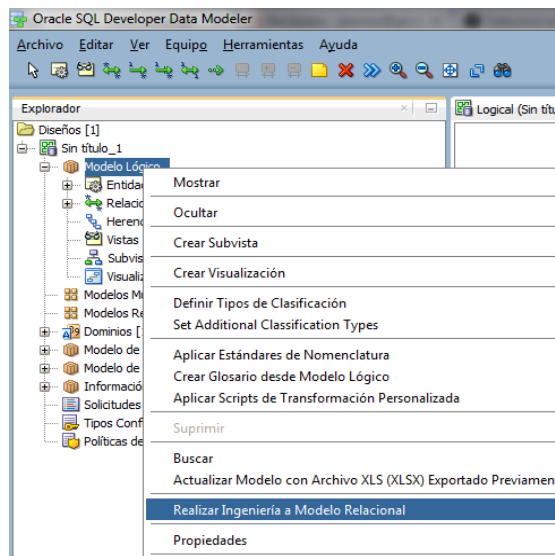


Fig.10 Diagram Conversion into tables

1.2.3 Mark the options shown in Fig. 11 and press the button **Realizar Ingeniería**

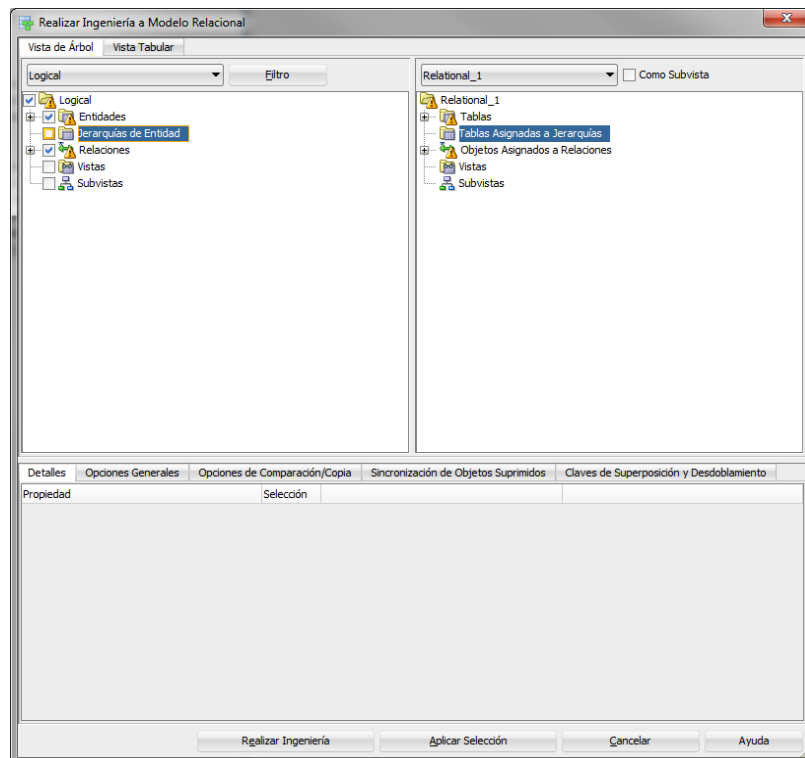


Fig.11 Dropdown menu related to the conversion of diagram to tables

1.2.4 Fig. 12 showing the obtained relational model:

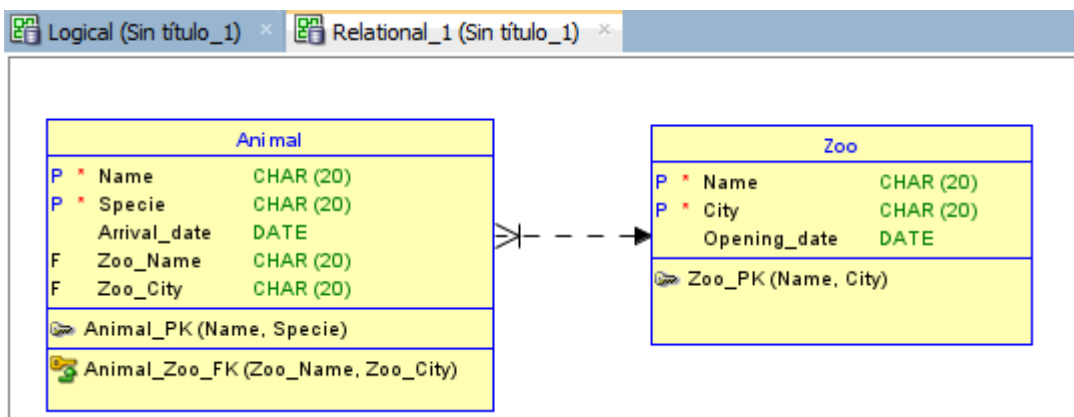


Fig.12 Final Relational Model

1.2.5 **Another option is to create a Relational Model directly**, using tools that Data Modeler gives us, as shown in Fig.13.

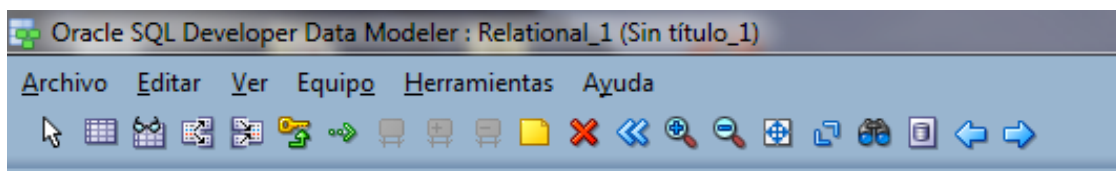


Fig.13 Direct creation of tables using Data Modeler

**1.3. Step of relational model to SQL code:** To do this you must press the appropriate button in the toolbar, as shown in Fig.14:

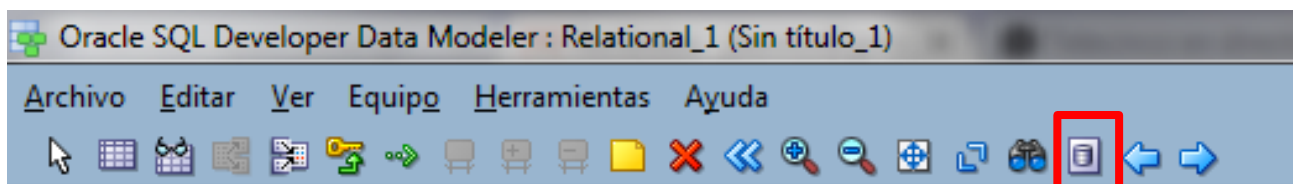


Fig.14 Direct generation of SQL code Button

On the next screen should press the Generate button, as shown in Fig.15:

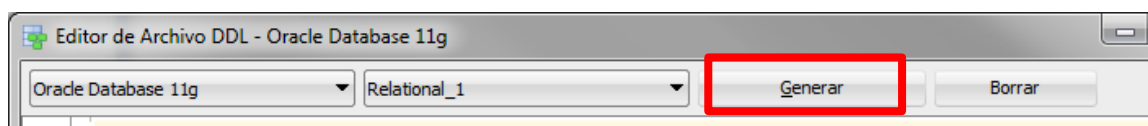


Fig.15 Initial of SQL code generation

Then Tables it will be necessary select the submenu select 'DROP' (see Fig.16). The DROP used to remove so be careful not to destroy necessary information.

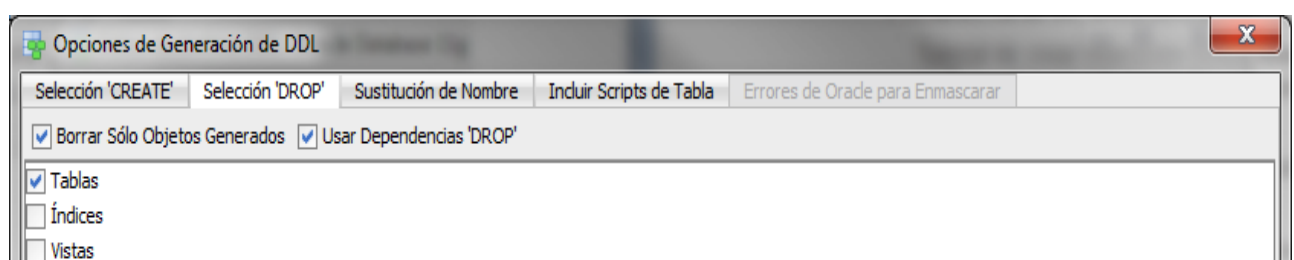
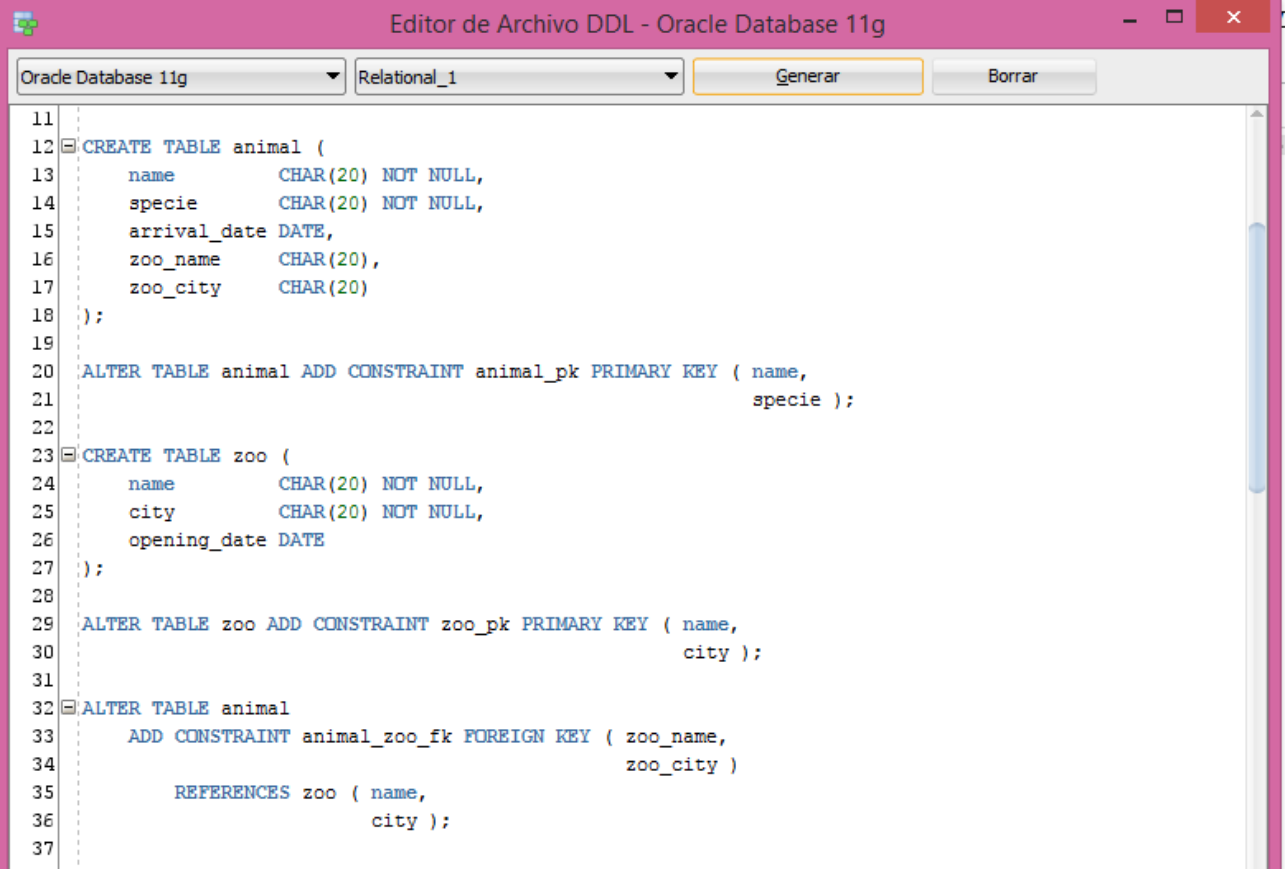


Fig.16 Select the necessary options related to information destruction

And finally you must press the **Aceptar** button. Fig. 17 shows the SQL code obtained:



```
11
12 CREATE TABLE animal (
13     name          CHAR(20) NOT NULL,
14     especie       CHAR(20) NOT NULL,
15     arrival_date  DATE,
16     zoo_name      CHAR(20),
17     zoo_city      CHAR(20)
18 );
19
20 ALTER TABLE animal ADD CONSTRAINT animal_pk PRIMARY KEY ( name,
21                                                         especie );
22
23 CREATE TABLE zoo (
24     name          CHAR(20) NOT NULL,
25     city          CHAR(20) NOT NULL,
26     opening_date  DATE
27 );
28
29 ALTER TABLE zoo ADD CONSTRAINT zoo_pk PRIMARY KEY ( name,
30                                                         city );
31
32 ALTER TABLE animal
33     ADD CONSTRAINT animal_zoo_fk FOREIGN KEY ( zoo_name,
34                                                         zoo_city )
35     REFERENCES zoo ( name,
36                     city );
37
38
```

Fig.17 SQL code

This code can be read or pasted directly in SQL Developer to create the database.