**Lab 3: Data Modeling with Oracle SQL Developer**

Oracle SQL Developer Data Modeler is a free graphical tool that enhances productivity and simplifies data modeling tasks. Using Oracle SQL Developer Data Modeler users can create, browse and edit, logical, relational, physical, multi-dimensional, and data type models. The Data Modeler provides forward and reverse engineering capabilities and supports collaborative development through integrated source code control. The Data Modeler can be used in both traditional and in Cloud environments.

Download Modeller from the below link:

<https://www.oracle.com/database/technologies/appdev/datamodeler.html>

Unzip and run datamodeler.exe to create ER Models:

# **Data Modeler Tutorial: Modeling for a Small Database**

<https://docs.oracle.com/cd/E39885_01/doc.40/e48205/tut_data_modeling.htm#DMDUG36166>

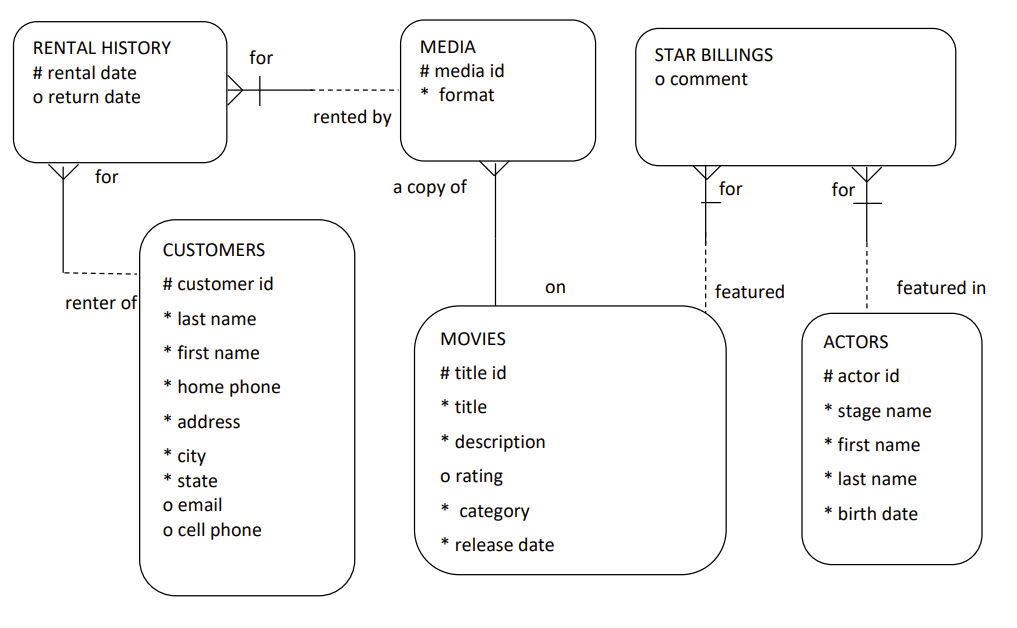
**Project ERD** - **OracleFlix Online Media Rentals**

You will perform the following major steps:

1. [Develop the Logical Model](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/tut_data_modeling.htm#CBAHGHGF).
2. [Develop the Relational Model](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/tut_data_modeling.htm#BABDHGDH).
3. [Generate DDL](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/tut_data_modeling.htm#BABFJJBC).
4. [Save the Design](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/tut_data_modeling.htm#BABEBICH).

# [Develop the Logical Model](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/tut_data_modeling.htm#CBAHGHGF)

To create ER model we will use the following ERD:

The logical model for the database includes following entities:

The model has the following entities:

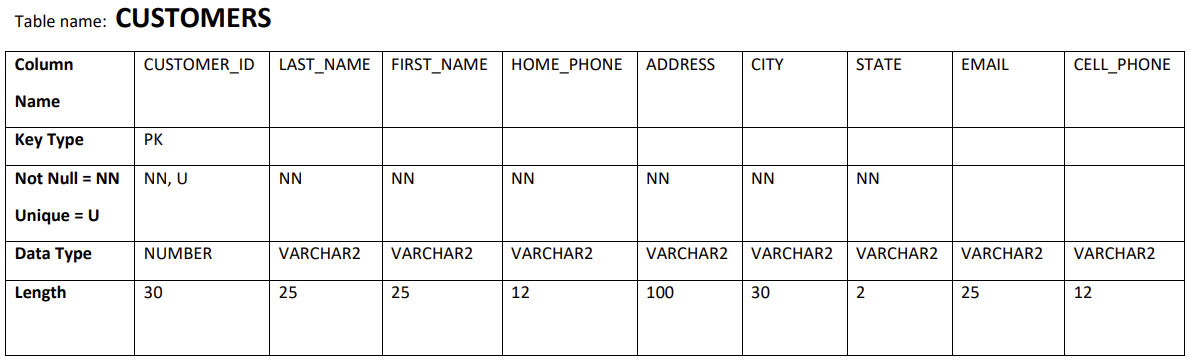
CUSTOMERS, MOVIES, MEDIA, RENTAL\_HOSTORY, ACTORS, STAR\_BILLINGS

However, before you create the entities, create some domains that will make the entity creation (and later DDL generation) more meaningful and specific.

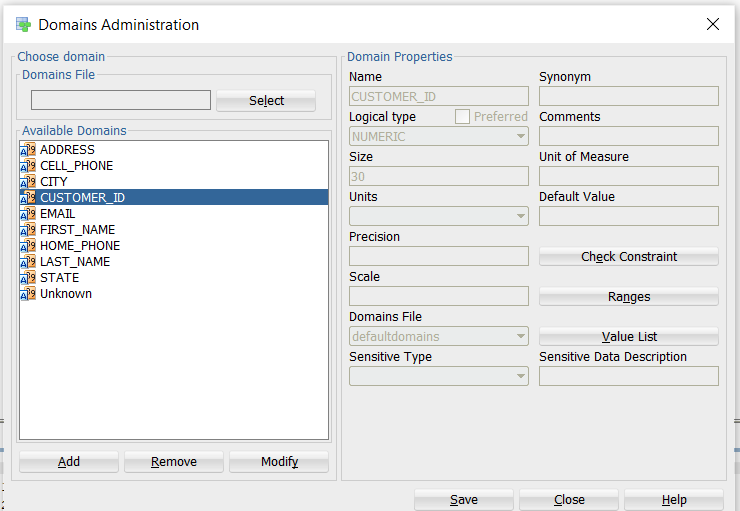
## Adding Domains:

In planning for your data needs, you have determined that several kinds of fields will occur in multiple kinds of records, and many fields can share a definition.

For example, you have decided that CUSTOMER table will have following structure:



1. Click **Tools**, then **Domains Administration**.
2. In the [Domains Administration](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/dialogs_data_modeling.htm#BABBFGAA) dialog box, add domains with the above definitions. Click **Add** to start each definition, and click **Apply** after each definition.



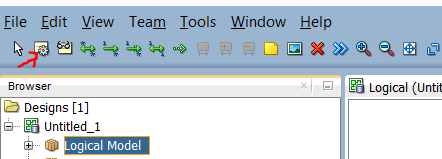
When you have finished defining these domains, click **Save**. This creates a file named defaultdomains.xml in the **datamodeler/domains directory** or **datamodeler\domains** folder under the location where you installed Data Modeler.

1. Click Close to close the dialog box.

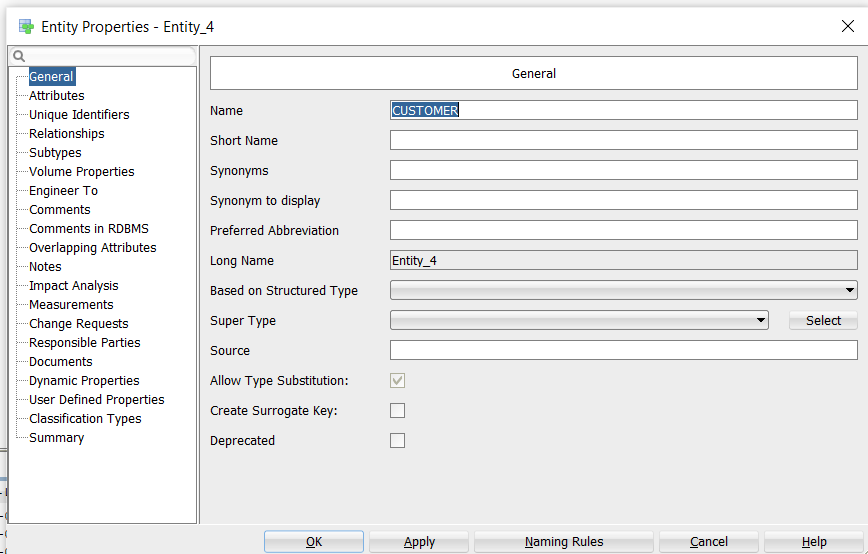
## Creating the CUSTOMER Entity

Create the Books entity as follows:

1. In the main area (right side) of the Data Modeler window, click the **Logical tab.**
2. Click the New Entity icon.

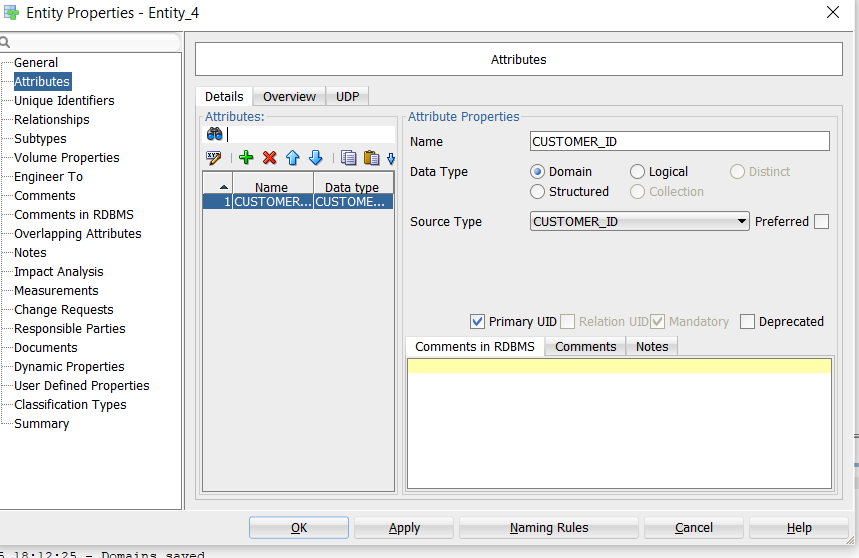


1. Click in the logical model pane in the main area; and in the Logical pane press, diagonally drag, and release the mouse button to draw an entity box. The [Entity Properties](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/dialogs_data_modeling.htm#BABHACHD) dialog box is displayed.
2. Click **General** on the left, and specify as follows:

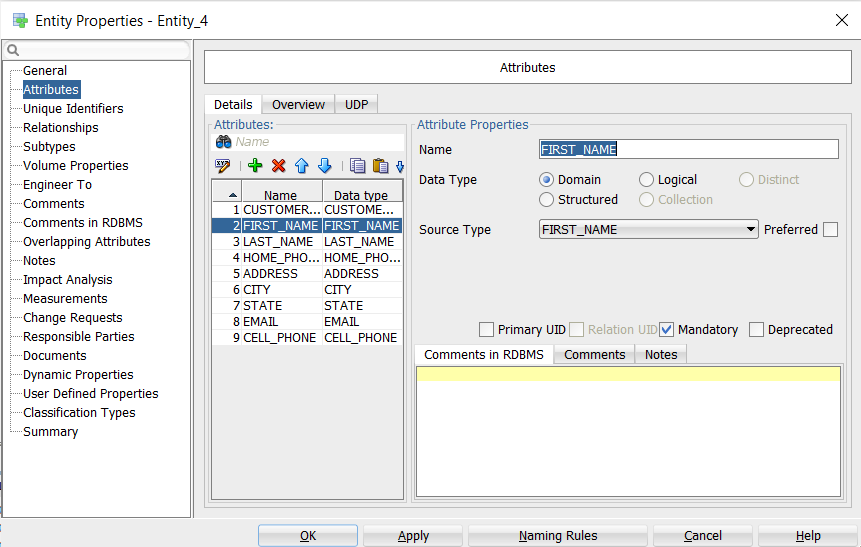


**Name**: CUSTOMER

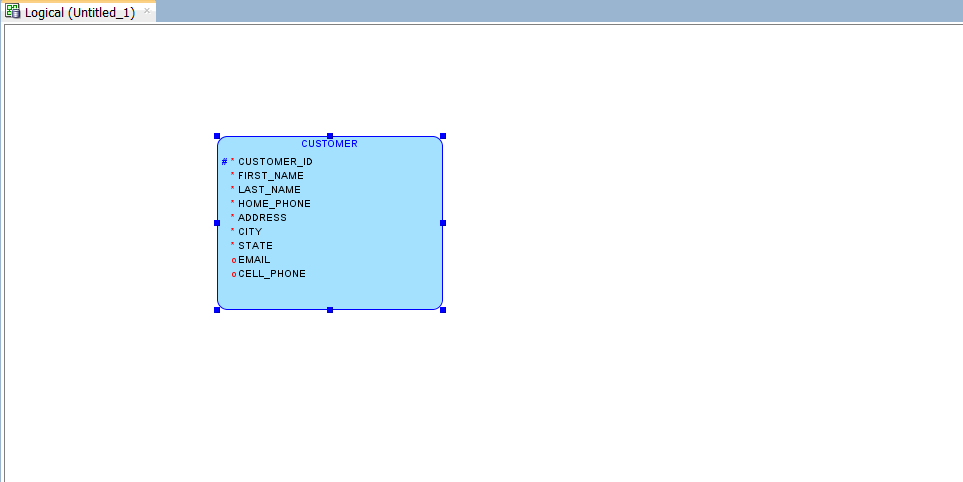
1. Click **Attributes** on the left, and use the **Add (+)** icon to add the following attributes, one at a time. (For datatypes, select from the Domain types) and from Source Type select CUSTOMER\_ID, check Primary UID to make it key



1. Add all attributes and click OK to finish creating CUSTOMER Entity

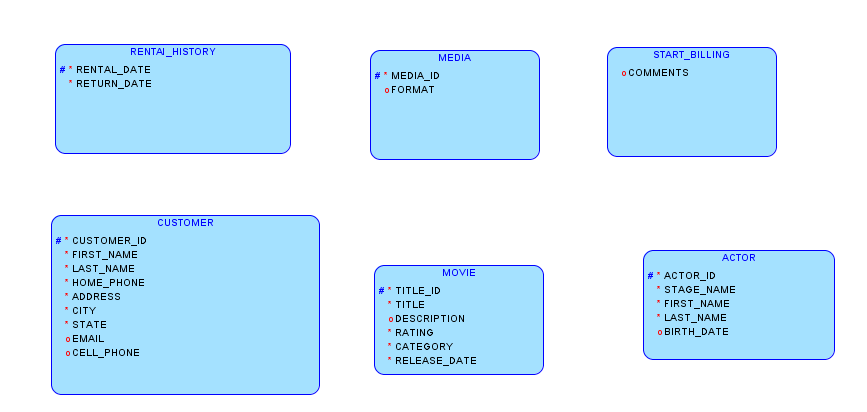


After finishing the customer entity is created:

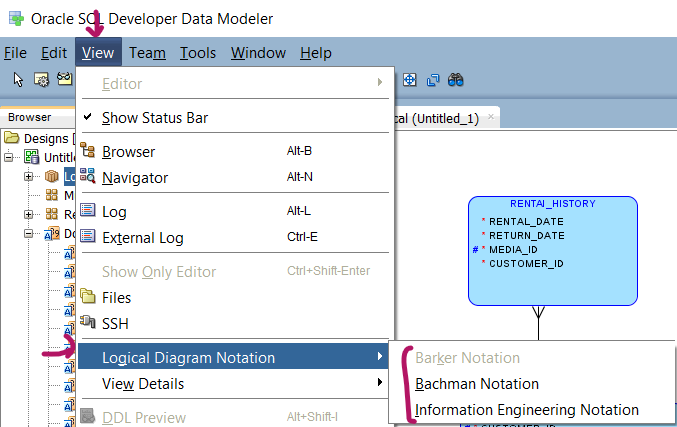


**PRACTICE EXERCISE:** Repeat the above steps for creating domains and entities for the below remaining entities:

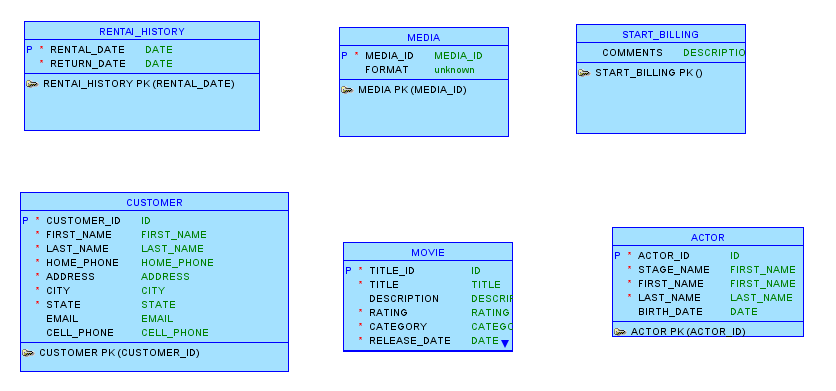
When you are done with all entities, the logical model pane in the main area should look like the following figure.



Note that for this figure, Barker Notation is used (you can change to Bachman notation by clicking View, then Logical Diagram Notation, then Bachman notation).



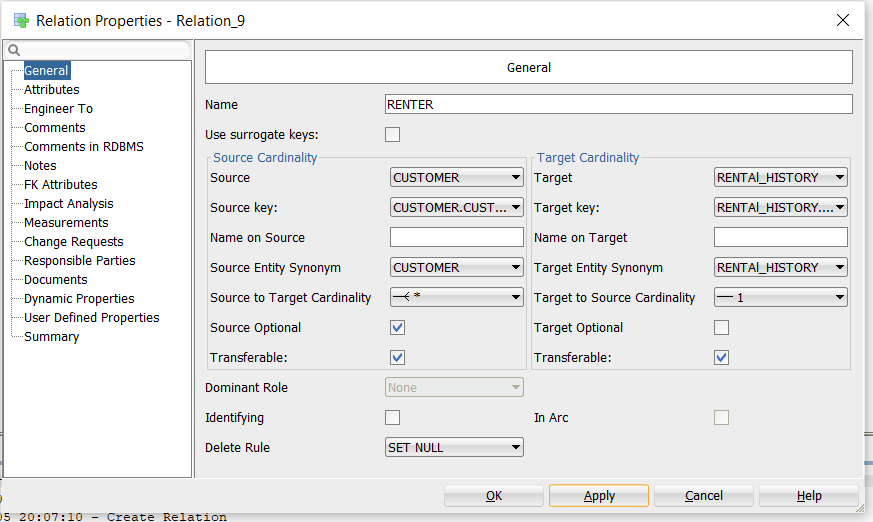
Bachman Notation



# Creating Relations Between Entities

CUSTOMER and RENTAl\_HISTORY has one-to-Many relationships from customer to RENTAL\_HISTORY.

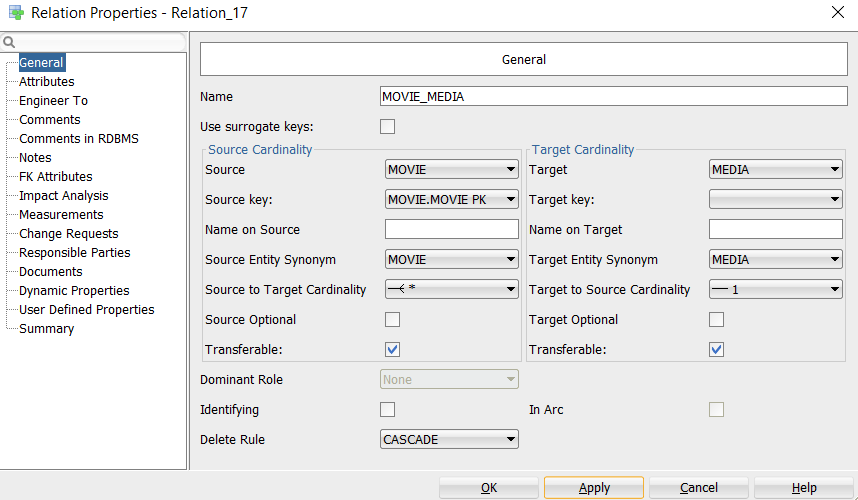
1. Click the 1:N Relation icon. 
2. Click first in the CUSTOMER box, then in the RENTAL\_HISTORY box. A line with an arrowhead is drawn from CUSTOMER to RENTAL\_HISTORY and following property dialogue will appear:



1. Check source optional box, since every customer may not rent a media, however, rental history should be mandatory if some customer rent a media, therefore leave target optional unchecked.

MOVIE and MEDIA has one-to-Many relationships from MOVIE to MEDIA

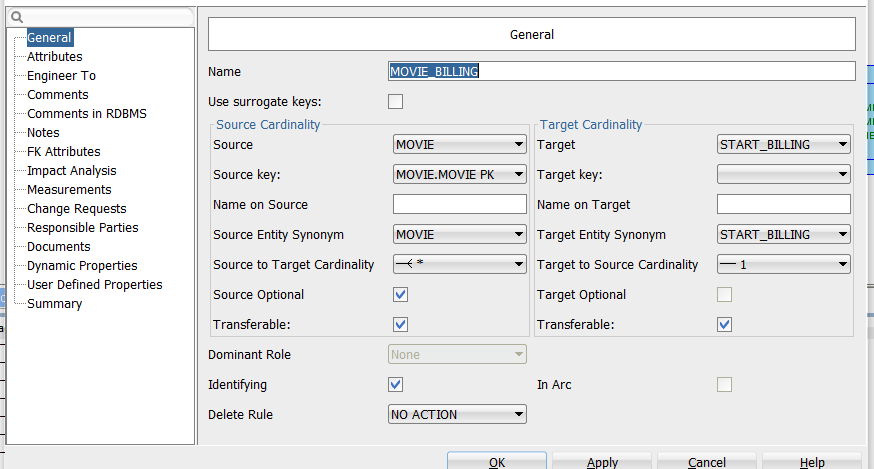
Add one-to-many relationship from MOVIE to MDIA. Note media cannot exist without movie keep source and target unchecked and set delete rule as CASCADE as shown in the below figure.



Optionally, double-click a line (or right-click a line and select Properties) and view the [Relation Properties](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/dialogs_data_modeling.htm#BABDJCBE) information.

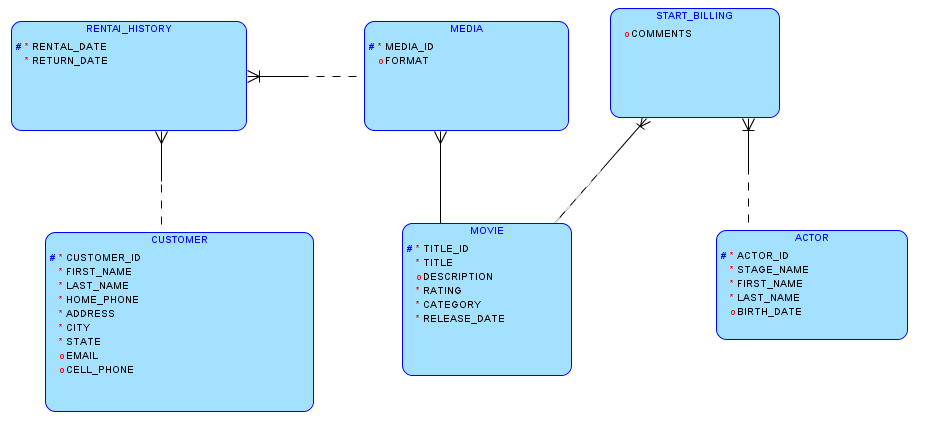
MOVIE and START\_BILLING Relationship:

MOVIE and START\_BILLING has one-to-Many relationships from MOVIE to START\_BILLING as shown in the below figure. Please note that identifying box is checked to make the TITLE\_ID both PK and FK.

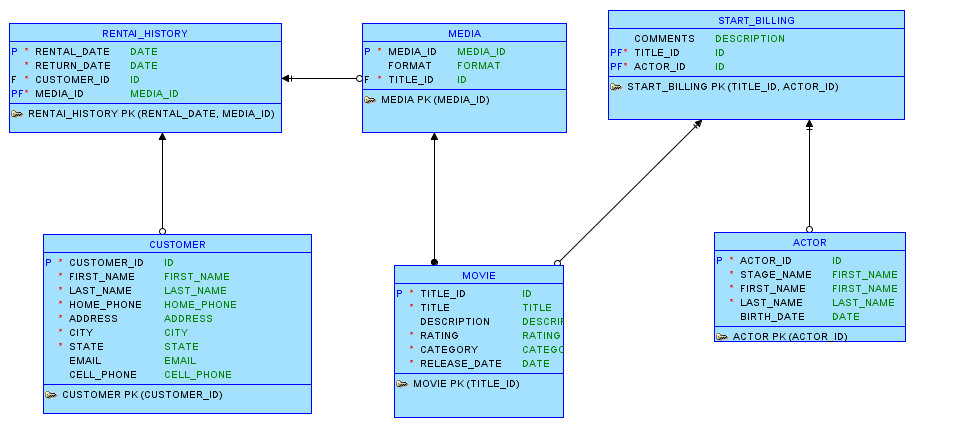


**Practice exercise: Complete all the remaining relationship.**

The final ERD should look like the following if using **Barker Notation.**



**Bachman Notation**

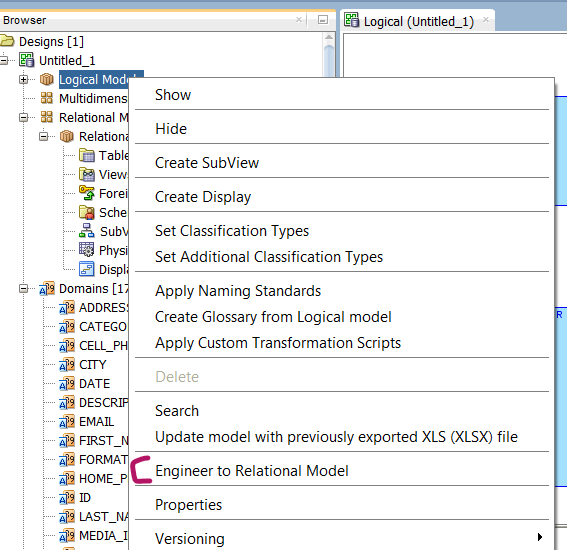


# Develop the Relational Model

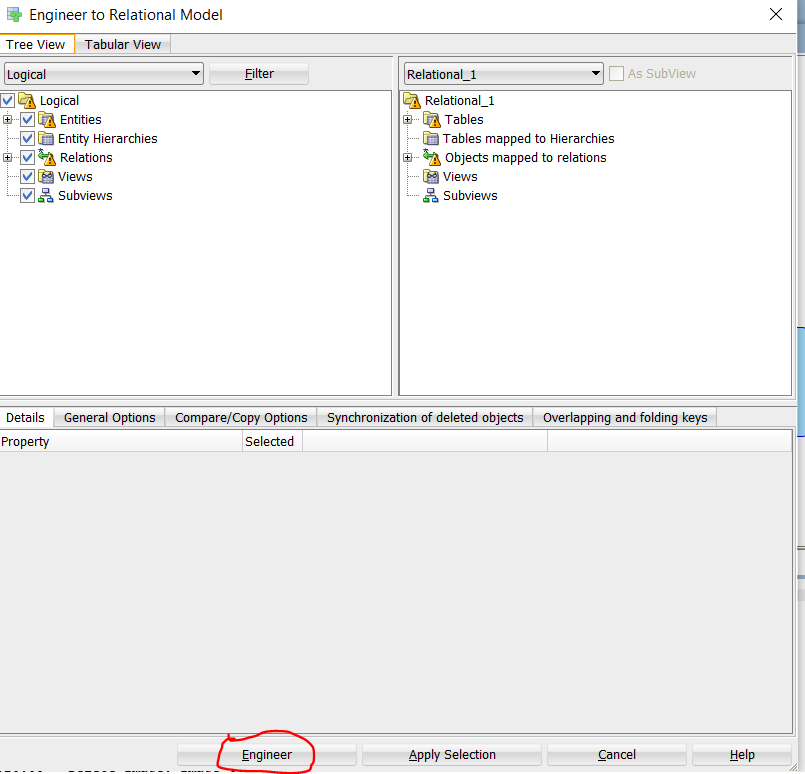
The relational model for the NetFlix project consists of tables that reflect the entities of the logical model and all attributes of each entity. In the simplified data model for this tutorial, a single relational model reflects the entire logical model; however, for other data models you can create one or more relational models, each reflecting all or a subset of the logical model. (To have a relational model reflect a subset of the logical model, use the "filter" feature in the dialog box for engineering a relational model.)

Develop the relational model as follows:

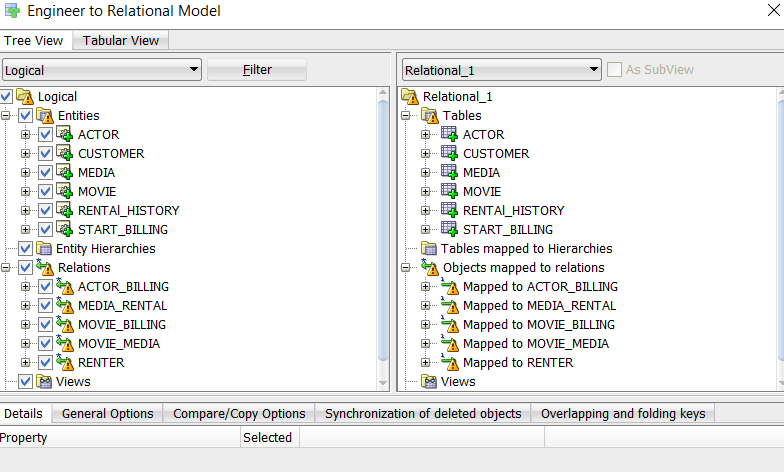
With the logical model selected, click the Engineer to Relational Model icon, or right-click the logical model in the navigator, then select **Engineer to Relational Model**. The [Engineering](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/dialogs_data_modeling.htm#BABGEAGD) dialog box is displayed.



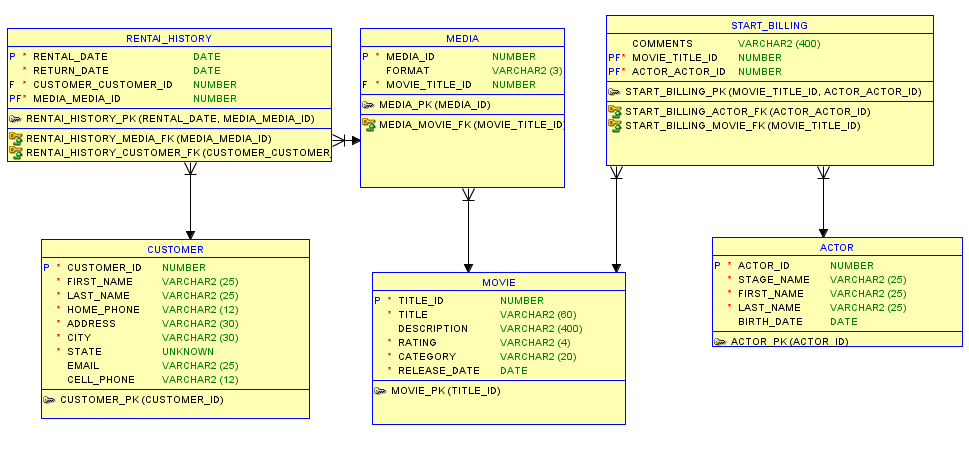
Accept all defaults (do not filter), and click **Engineer**. This causes the Relational\_1 model to be populated with tables and other objects that reflect the logical model.



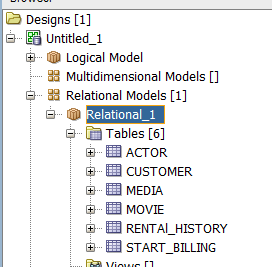
Expand each unit to see how logical model is mapped to relational model:



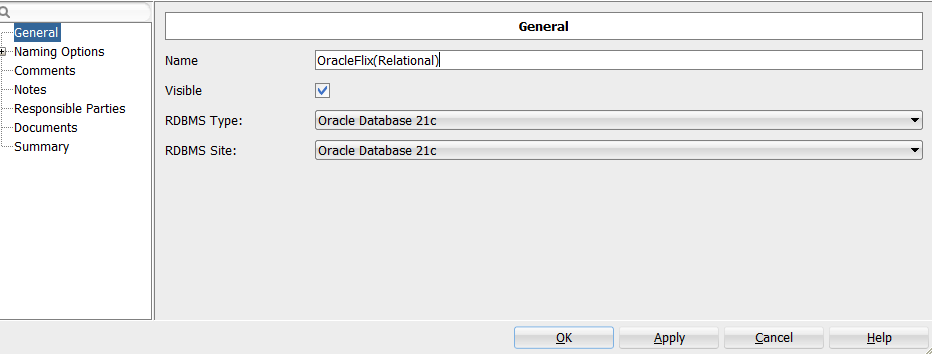
Relational model created like the below figure. Note the model is not perfect, please inspect and remove if some unnecessary columns appears.

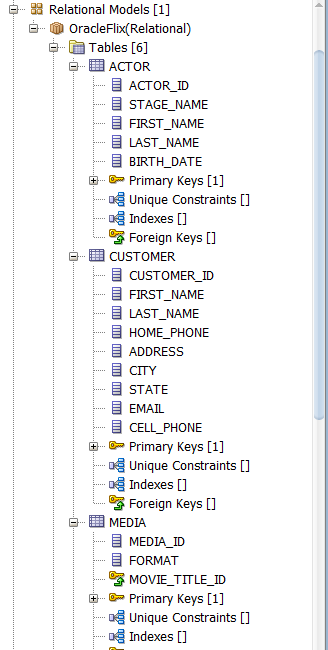


Expand the Relational Models node in the object browser on the left side of the window, and expand Relational\_1 and optionally nodes under it that contain any entries (such as Tables and Columns), to view the objects created.



Change the name of the relational model from Relational\_1 to something more meaningful for diagram displays, such as OracleFlix (relational). Specifically, right-click Relational\_1 in the hierarchy display, select **Properties**, in the General pane of the [Model Properties - <name> (Relational)](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/dialogs_data_modeling.htm#BABIHFFE) dialog box specify **Name** as **OracleFlix(relational)**, and click **OK**.



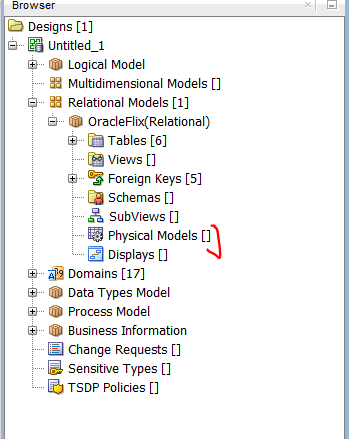


# Generate DDL

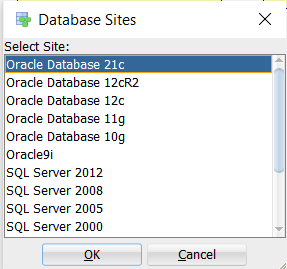
Generate Data Definition Language (DDL) statements that you can use to create database objects that reflect the models that you have designed. The DDL statements will implement the physical model (type of database, such as Oracle Database 21C) that you specify.

Develop the physical model as follows:

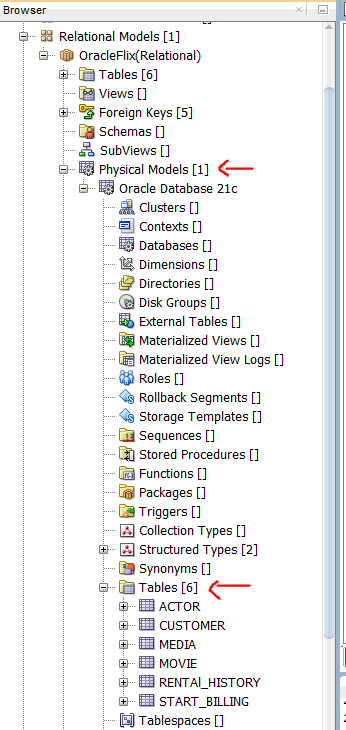
1. Optionally, view the physical model before you generate DDL statements:
   1. With the relational model selected and expanded, right-click the Physical Models node and select **New**. A dialog box is displayed for selecting the type of database for which to create the physical model.



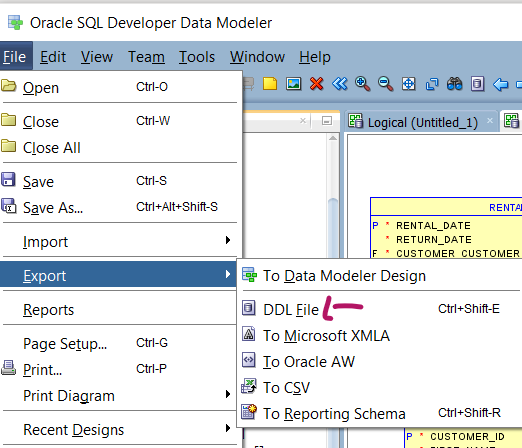
* 1. Specify the type of database (for example, Oracle Database 21), and click **OK**. A physical model reflecting the type of database is created under the Physical Models node.



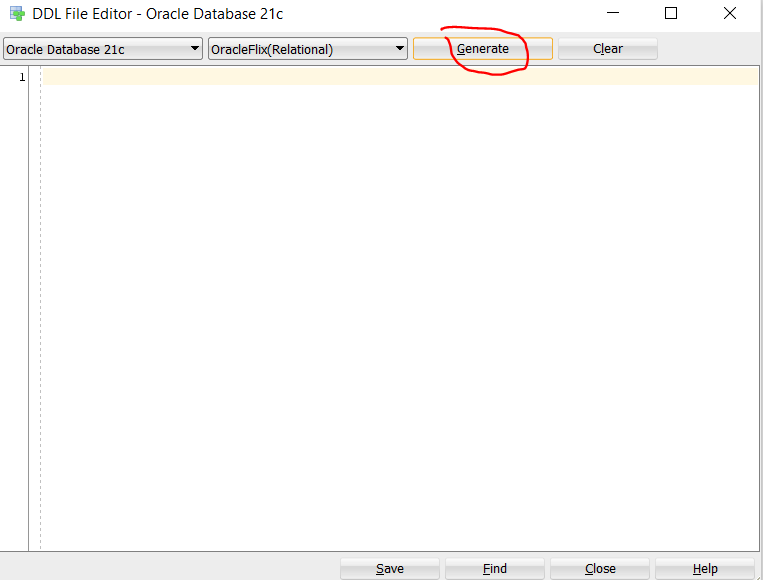
* 1. Expand the Physical Models node under the OracleFlix relational model, and expand the newly created physical model and the Tables node under it, to view the table objects that were created.



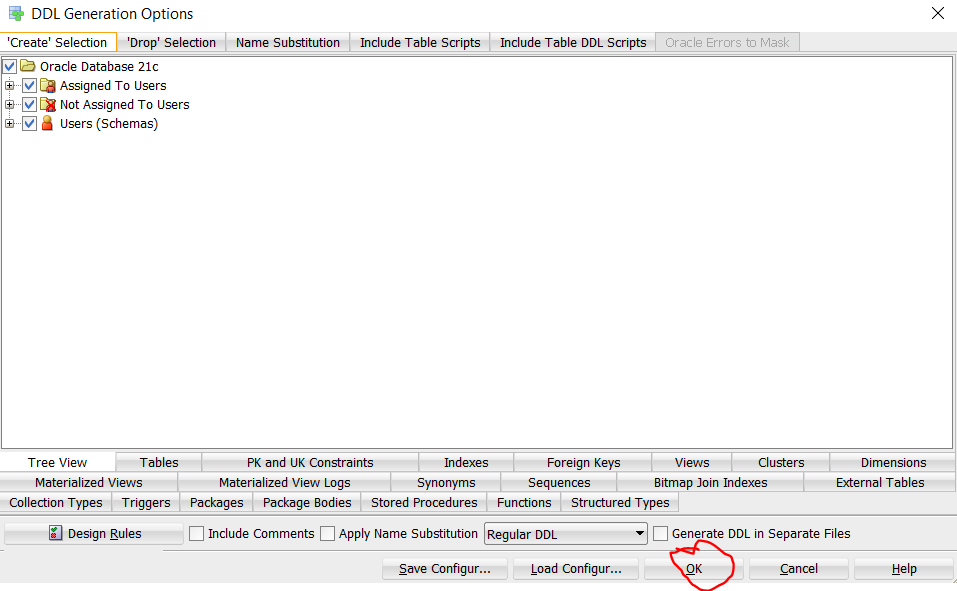
1. Click **File**, then **Export**, then **DDL File**.



1. Select the database type (for example, Oracle Database 21C) and click **Generate**. The [DDL Generation Options](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/dialogs_data_modeling.htm#BABDHGHC) dialog box is displayed.



1. Accept all defaults, and click **OK**. A DDL file editor is displayed, with SQL statements to create the tables and add constraints. (Although you can edit statements in this window, do not edit any statements for this tutorial exercise.)



1. Click **Save** to save the statements to a .sql script file (for example, **create\_OracleFlix\_objects.sql**) on your local system.

Later, run the script (for example, using a database connection and SQL Worksheet in SQL Developer) to create the objects in the desired database.

1. Click **Close** to close the DDL file editor.

# Save the Design

Save the design by clicking **File**, then **Save**. Specify the location and name for the XML file to contain the basic structural information (for example, **OracleFlix\_design.xml**). A directory or folder structure will also be created automatically to hold the detailed information about the design.

Continue creating and modifying design objects, if you wish. When you are finished, save the design again if you have made any changes, then exit Data Modeler by clicking **File**, then **Exit**.

You can later open the saved design and continue working on it, as explained in [Section 1.6, "Saving, Opening, Exporting, and Importing Designs"](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/data_modeling.htm#BABFAFEG).

# Saving, Opening, Exporting, and Importing Designs

To store a design (or parts of a design) that you are working on, you can save or export it.

* **Saving** a design enables you to save all elements of the design: the logical model, relational models, physical models, process model, and business information. An XML file and a directory structure (described in [Section 1.3.1, "Database Design"](https://docs.oracle.com/cd/E39885_01/doc.40/e48205/data_modeling.htm#BABCFHIH)) are created for a new design or updated for the existing design, which is stored in Data Modeler format.

To save a design, click **File**, then **Save**. If the design was not previously saved, specify the location and XML file name. To save a design in a different file and directory structure. click **File**, then **Save As**.

* **Exporting** a design enables you to save parts of the design (logical model, relational models but no physical models, and data types model) to a file. You can export in a variety of formats, both non-Oracle and Oracle. Thus, exporting provides flexibility in output formats, but saving enables you to save more design objects if you only need Data Modeler output.

To export a design, click **File**, then **Export**, then the output format.

To use a design that had been saved, you can **open** it by clicking **File**, then **Open**. Opening a design makes all models and objects in the saved design available for you to work on. Any saved physical models are not initially visible in the object browser; however, you can make a physical model visible by right-clicking Physical Models under the desired relational model, selecting Open, and then specifying the database type (such as Oracle 11*g*).

To use a design that had been saved by Data Modeler, or exported or saved by another data modeling tool, you can **import** it by clicking **File**, then **Import**, then the type of design to be imported. Usually, you specify a file, and then use a wizard that enables you to control what is imported.

For more information on Data Modeler please feel free to explore the below link:

<https://docs.oracle.com/cd/E39885_01/doc.40/e48205/tut_data_modeling.htm#DMDUG36166>