National University of Computer & Emerging Sciences, Karachi

School of Computing Department

Fall 2023, Lab Manual - 06

Course Code: CL-2005	Course : Database Systems Lab
Instructor(s) :	Abeer Gauher

Relational Modeling

SQL Data Modeler

Oracle SQL Developer Data Modelers a standalone, independent product, available for download from the Oracle Technology Network (OTN).SQL Developer Data Modeler runs on Windows, Linux and MacOS X.

Logical Models

The logical model in SQL Developer Data Modeler includes standard logical modeling facilities, such as drawing entities and relationships etc.

Relational Models

The SQL Developer Data Modeler relational model is an intermediate model between the logical model and the physical models. It supports relational design decisions independent of the constraints of the target physical platform(s).All many-to-many relationships and all super type/sub-types entity hierarchies are resolved during forward engineering (transformation) of the logical model, or part of it, to a relational model.

Physical Model

A physical data model defines all of the logical database components and services that are required to build a database or can be the layout of an existing database. A physical data model consists of the table's structure, column names and values, foreign and primary keys and the relationships among the tables.

Data Modeling for a Small Database

We will use SQL Developer Data Modeler to create models for a simplified library database Entities Included will be:

- Books
- Patrons (people who have library cards)
- Transactions (checking a book out, returning a book, and so on).

We are using only a subset of the possible steps for theTop-Down Modeling approach.

1. Develop the Logical Model.
2. Develop the Relational Model.
3. Generate DDL.
4. Save the Design.

1. Develop the logical model

The logical model for the database includes three entities:

Books (describes each book in the library),

Patrons (describes each person who has a library card), and

Transactions (describes each transaction involving a patron and a book).

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

Adding Domains

In planning for our data needs, we have determined that several kinds of fields will occur in multiple kinds of records, and many fields can share a definition. For example, we have decided that:

- The first and last names of persons can be up to25 characters each.
- Street address lines can be up to 40 characters.
- City names can be up to25 characters.
- State codes (United States) are2-character standard abbreviations.
- Zip codes (United States postal codes) can be up to 10 characters (nnnnn-nnnn).
- Book identifiers can be up to 20 characters.
- Other identifiers are numeric, with up to 7 digits (no decimal places).Titles (books, articles, and so on) can be up to 50 characters.
- These added domains will also be available after we exit Data Modeler and restart it later.

Steps to add domains:

1. Click Tools, then Domains Administration.
2. In the Domains Administration dialog box, add domains with the following definitions.
   Click Add to start each definition, and click Apply after each definition.

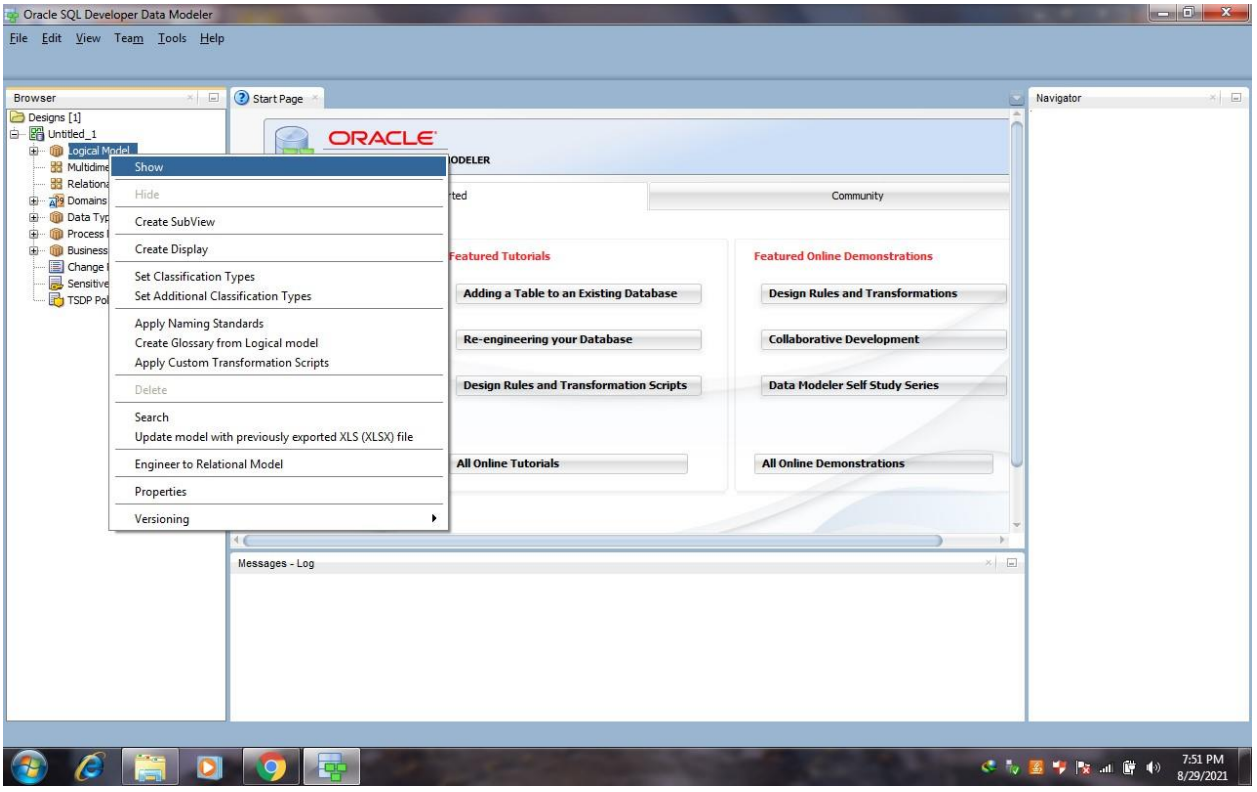
3. When we have finished defining these domains, click Save. This creates a file named default domains.xmlin the domains directory (folder) under the location where we installed Data Modeler.
4. Optionally, copy the defaultdomains.xmlfile to a new location (not under the Data Modeler installation directory), and give it an appropriate name, such as library_domains.xml. We can then import domains from that file when we create other designs.
5. Click Close to close the dialog box.

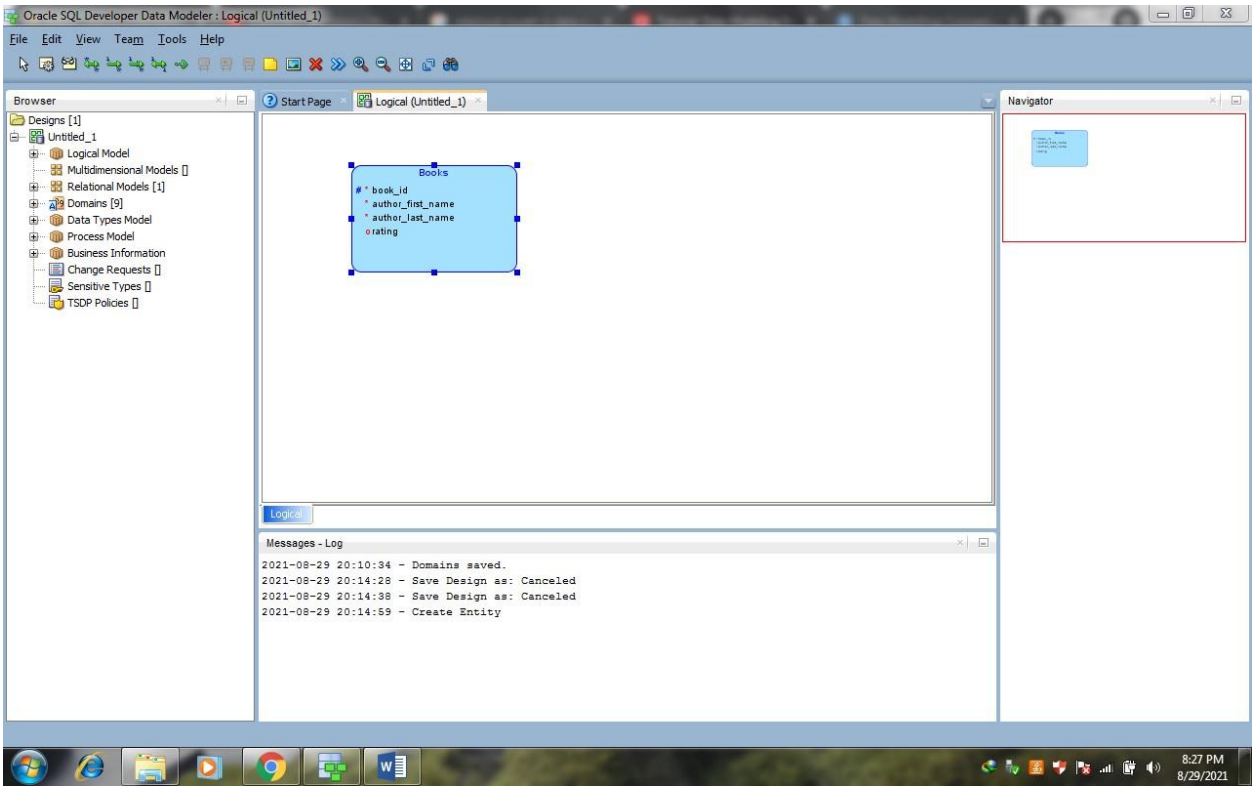
Creating Entities

Creating the book Entity:

- Create the Books entity as follows:
- In the main are a (right side) of the SQL Developer Data Modeler window, click the Logical tab.
- Click the new Entity icon.
- 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 dialog box is displayed.
- Click General on the left, and specify as follows:
- Name: Books
- Click Attributes on the left, and use the Add (+) icon to add the following attributes, one at a time. (For data types, select from the Domain types except or Rating, which is a Logical type.)

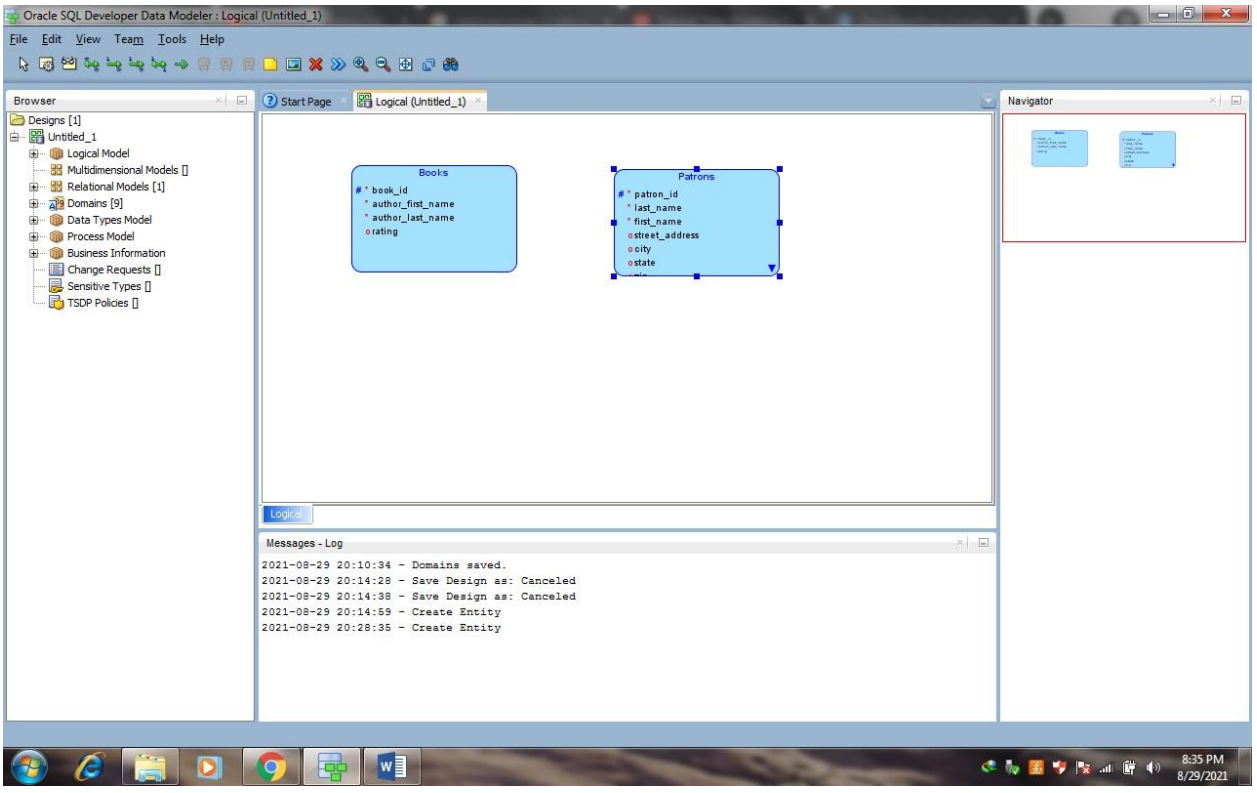
Name	Datatype	Other Information and Notes
book_id	Domain: Book Id	Primary UID (unique identifier). (The Dewey code or other book identifier.)
title	Domain: Title	M (mandatory, that is, must not be null).
author_last_name	Domain: Person Name	M (mandatory, that is, must not be null).
author_first_name	Domain: Person Name	25 characters maximum.
rating	Logical type: NUMERIC (Precision=2, Scale= 0)	(Librarian's personal rating of the book, from 1 (poor) to 10 (great).)




Oracle SQL Developer Data Modeler : Logical (Untitled_1)
File Edit View Team Tools Help
Browser
Designs [1]
Untitled_1
Logical Model
Multidimensional Models []
Relational Models [1]
Domains [9]
Data Types Model
Process Model
Business Information
Change Requests []
Sensitive Types []
TSDP Policies []
Start Page
Logical (Untitled_1)
Books
* book_id
* author_first_name
* author_last_name
o rating
Logical
Navigator
Messages - Log
2021-08-29 20:10:34 - Domains saved.
2021-08-29 20:14:28 - Save Design as: Canceled
2021-08-29 20:14:38 - Save Design as: Canceled
2021-08-29 20:14:59 - Create Entity
8:27 PM
8/29/2021

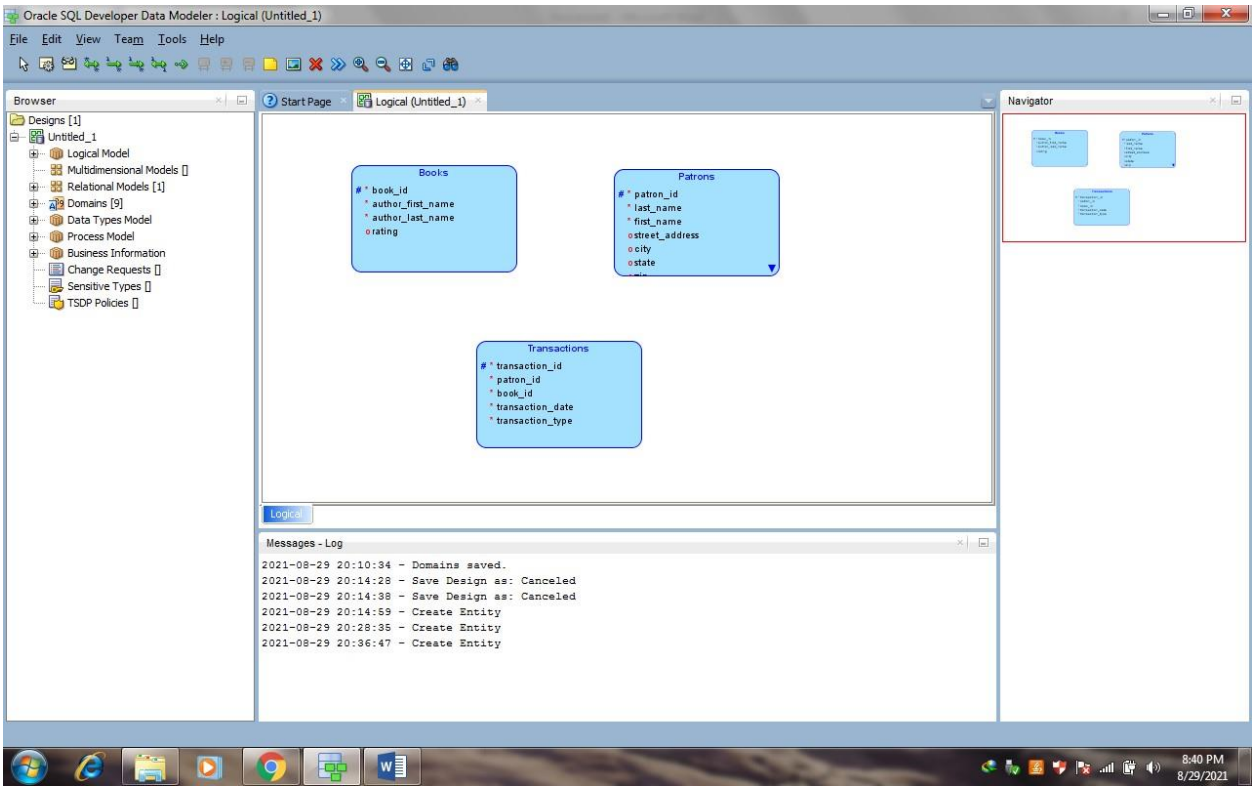
Creating the Patrons Entity

Attribute Name	Type	Other Information and Notes
patron_id	Domain: Numeric Id	Primary UID (unique identifier). (Unique patron ID number, also called the library card number.)
last_name	Domain: Person Name	M (mandatory, that is, must not be null). 25 characters maximum.
first_name	Domain: Person Name	(Patron's first name.)
street_address	Domain: Address Line	(Patron's street address.)
city	Domain: City	(City or town where the patron lives.)
state	Domain: State	(2-letter code for the state where the patron lives.)
zip	Domain: Zip	(Postal code where the patron lives.)
location	Domain:Address	Oracle Spatial geometry object representing the patron's geocoded address.



Creating the Transactions Entity

Attribute Name	Type	Other Information and Notes
tranction_id	Domain: Numeric Id	Primary UID (unique identifier). (Unique transaction ID number)
transaction_date	Logical type: Datetime	M (mandatory, that is, must not be null). Date and time of the transaction.
transaction_type	Domain: Numeric Id	M (mandatory, that is, must not be null). (Numeric code indicating the type of transaction, such as 1 for checking out a book.)



Creating Relations between Entities

Relations show the relationships between entities: one-to-many, many-to-one, or many-to-many. The following relationships exist between the entities:

Books and Transactions: one-to-many. Each book can be involved in multiple sequential transactions. Each book can have zero or one active checkout

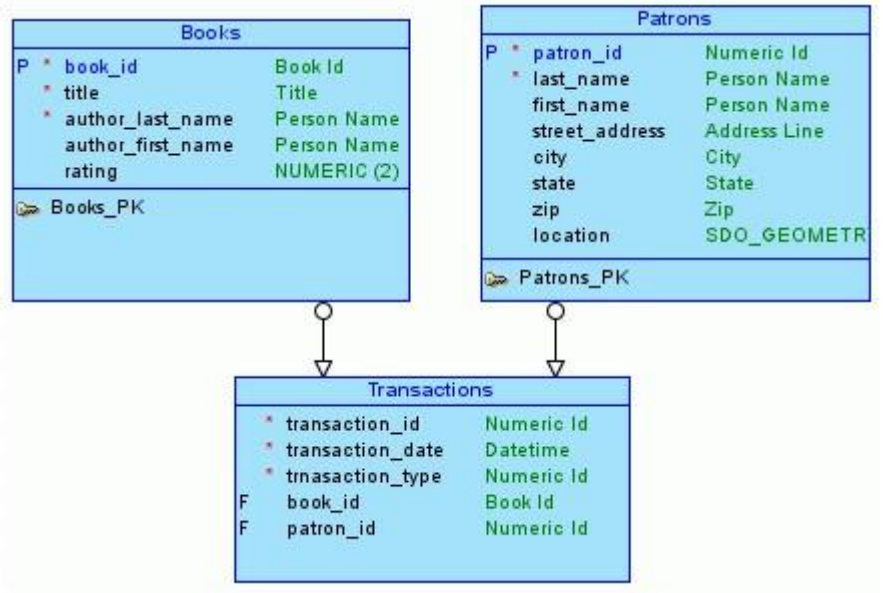
transactions; a book that is checked out cannot be checked out again until after it has been returned.

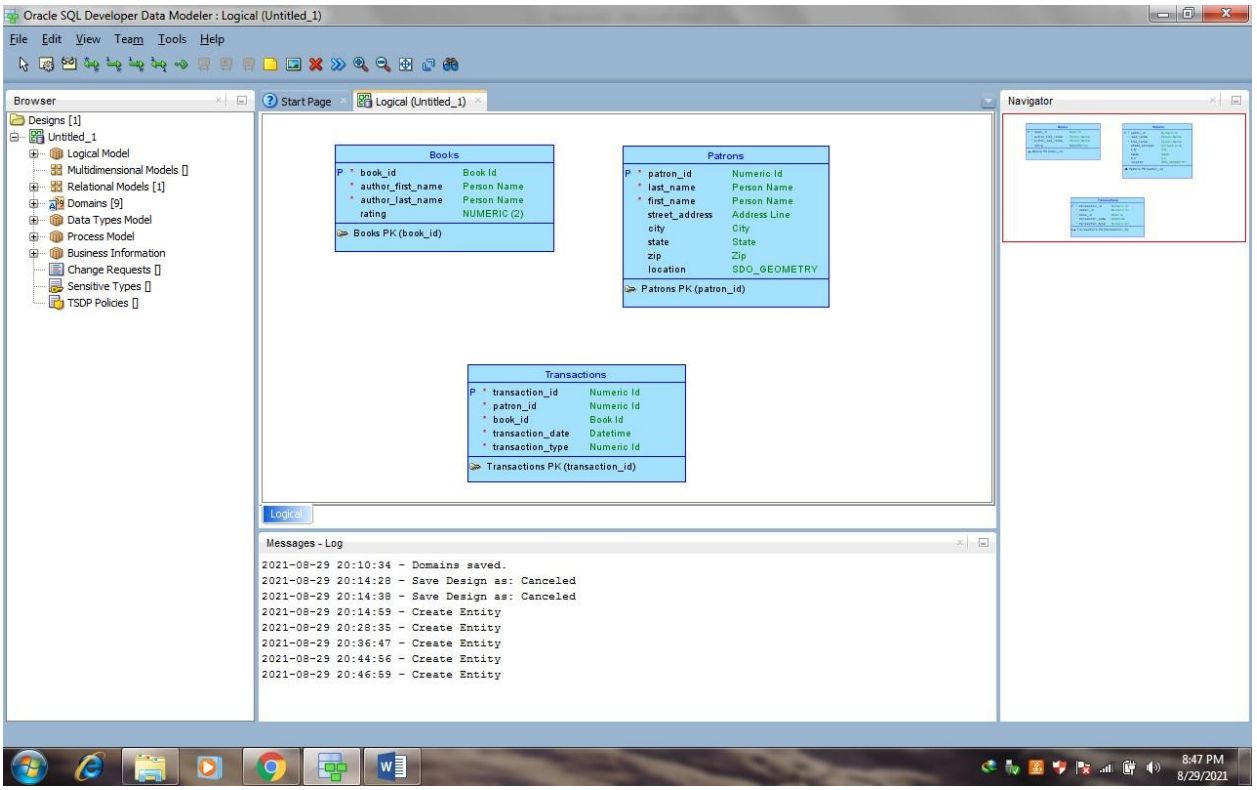
Patrons and Transactions: one-to-many. Each patron can be involved in multiple sequential and simultaneous transactions. Each patron can check out one or many books in a visit to the library, and can have multiple active check out transactions reflecting several visits; each patron can also return checked out books at any time**.**

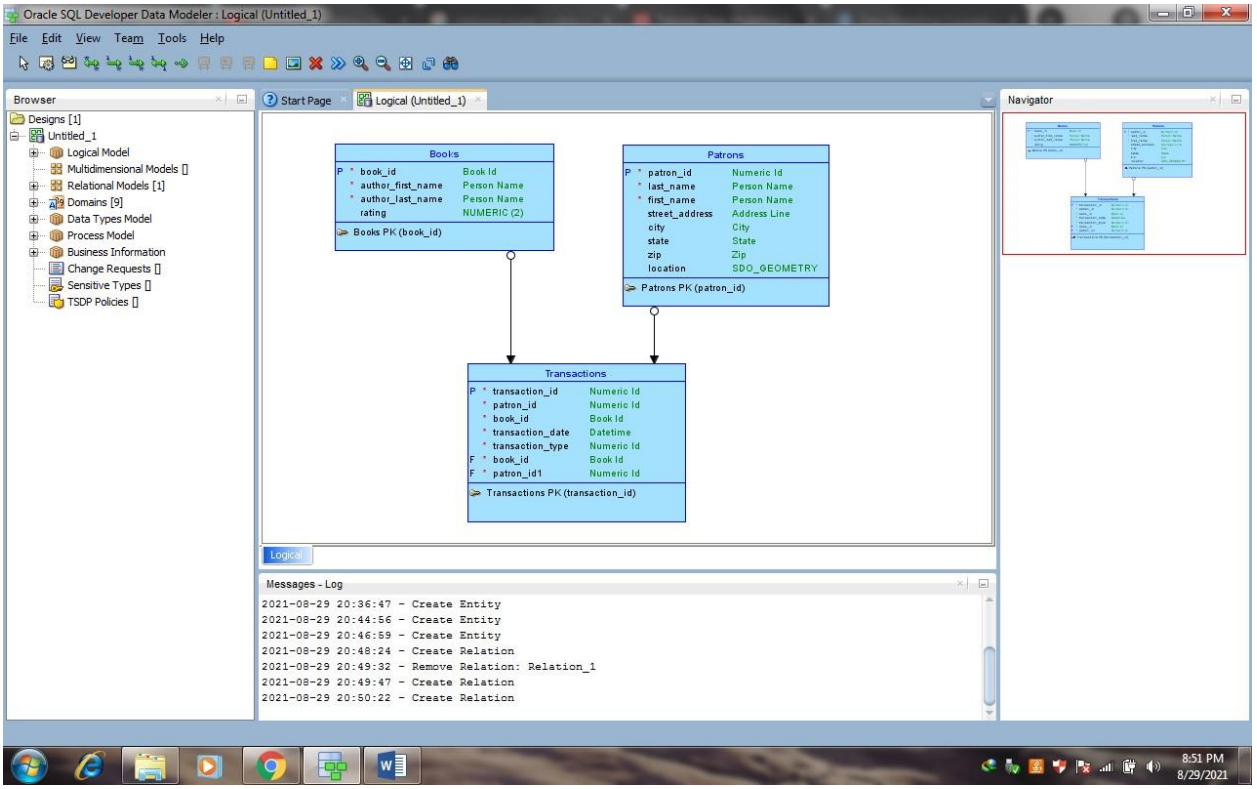
Steps to Create Relationships:

1. In the logical model pane in the main area, arrange the entity boxes as follows: Books on the left, Patrons on the right, and Transactions either between Books and Patrons or under them and in the middle. (If the pointer is still cross-hairs, click the Select icon at the top left to change the pointer to an arrow.)

2. Click the New 1: N Relation icon.
3. Click first in the Books box, then in the Transactions box. A line with an arrowhead is drawn from Books to Transactions.
4. Click the New 1: N Relation icon.
5. Click first in the Patrons box, then in the Transactions box. A line with an arrowhead is drawn from Patrons to Transactions.
6. Optionally, double-click a line (or right-click a line and select Properties) and view the Relation Properties information.







2. Develop the Relational Model

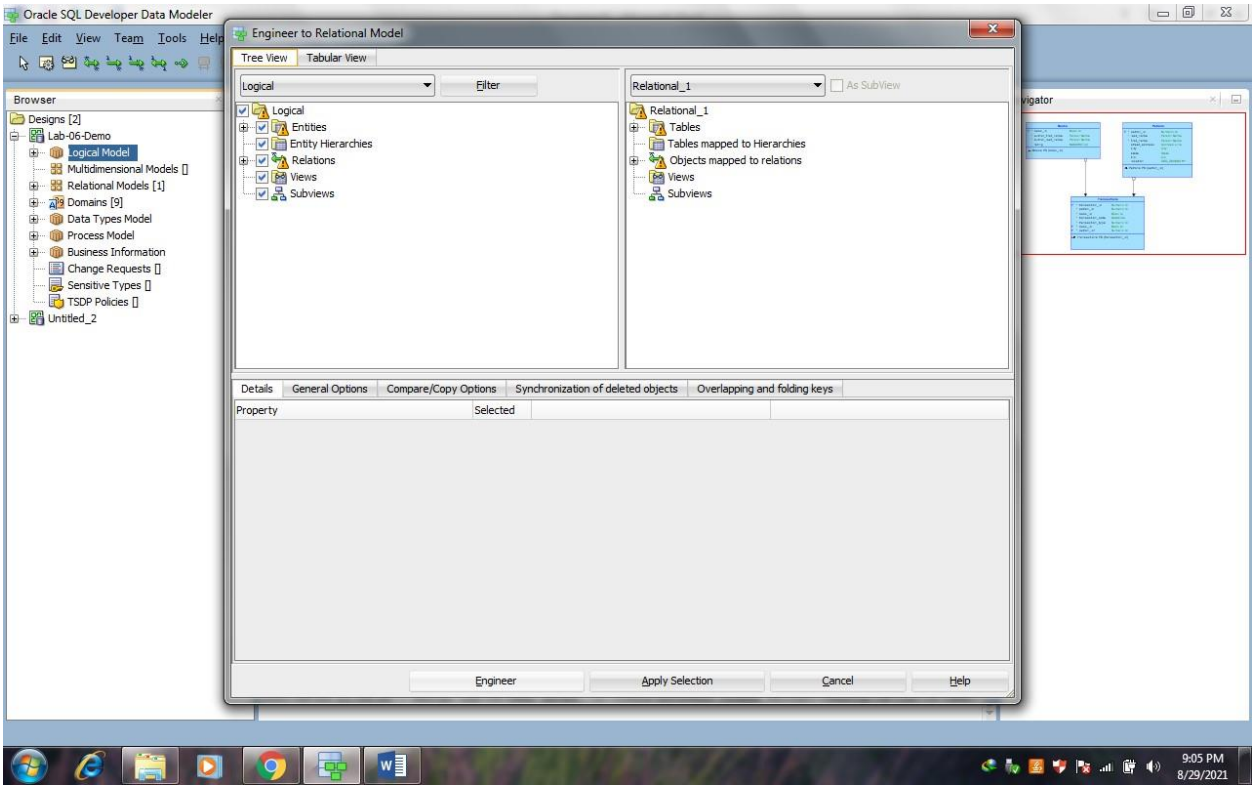
The relational model for the library tutorial data base consists of tables that reflect the entities of the logical model (Books, Patrons, and Transactions) and

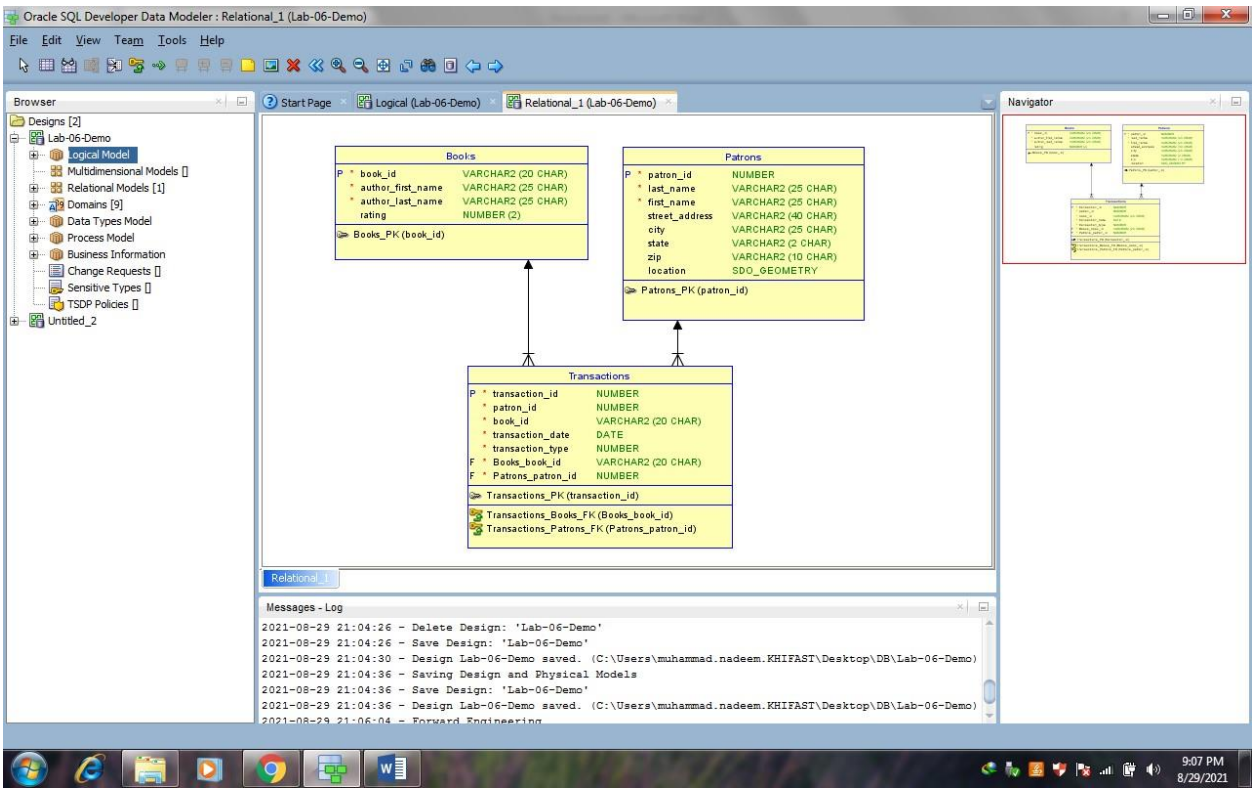
all attributes of each entity. In the simplified data model for this tutorial, a single relational model reflects the entire logical model; however, for other data models we can create one or more relational models, each reflecting all or a sub set of the logical model. (To have a relational model reflect a subset of the logical model, use the "filter" feature in the dialog box for engineering a relational model.).

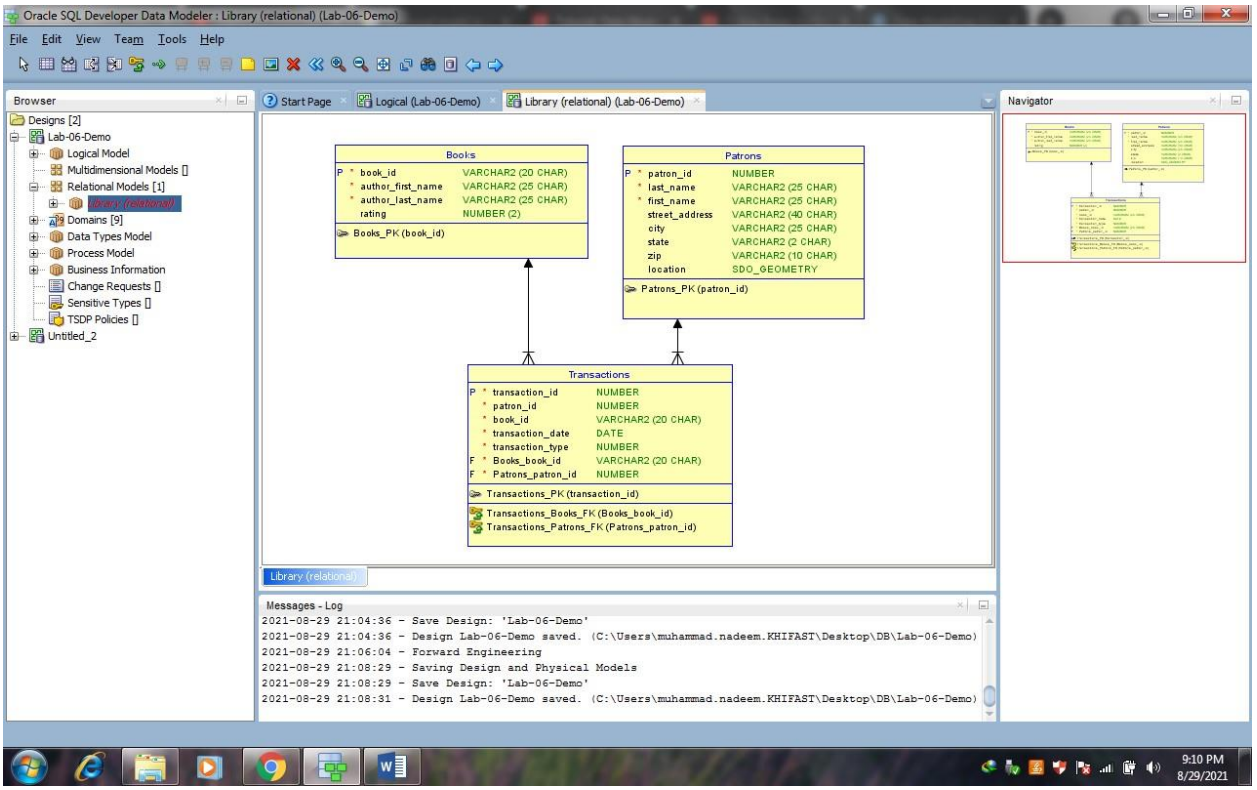
Develop the relational model as follows:

1. With the logical model selected, click **Design**, then **Engineer to Relational Model**. The Engineering dialog box is displayed.
2. 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 he logical model.
3. Optionally, expand the Relational Models node in the object browser on the left side of the window, and expand Relational_1and nodes under it that contain any entries (such as Tables and Columns), to view the objects created.

4. Change the name of the relational model from Relational_1 to something more meaningful for diagram displays, such as Library(relational).Specifically, right-clickRelational_1 in the hierarchy display, select **Properties**, in the General pane of the Model Properties-<name>(Relational) dialog box specify **Name** as Library(relational),and click **OK**.







3. Generate DDL

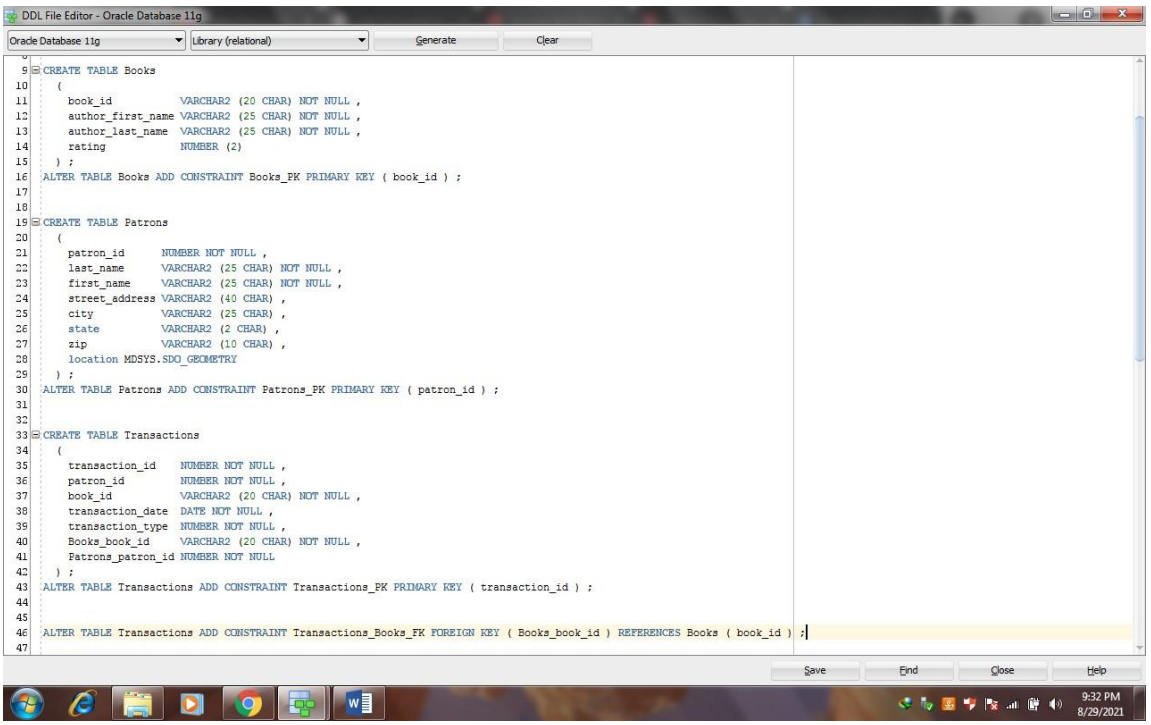
Develop the physical model as follows:

1. Optionally, view the physical model before we generate DDL statements:

a. With the Library logical model selected, click Physical, then Open Physical Model. A dialog boxis displayed for selecting the type of database for which to create the physical model.
b. Specify the type of database(for example, Oracle Database11g),and click OK.In the hierarchy display on the left side of the window, a Physical Models node is added under the Library relational model node, and a physical model reflecting the type of databaseis created under the Physical Models node.
c. Expand the Physical Models node under Library (the relational model), and expand the newly created physical model and nodes under it that contain any entries (such as Tables and Columns), to view the objects created.

2. Click File, then Export, then DDL File.
3. Select the database type (for example, Oracle Database11g) and click Generate. The DDL Generation Options dialog box is displayed.
4. Accept all defaults, and click OK. A DDL file editor is displayed, with SQL statements to create the tables and add constraints. (Although we can edit statements in this window, do not edit any statements for this tutorial exercise.)

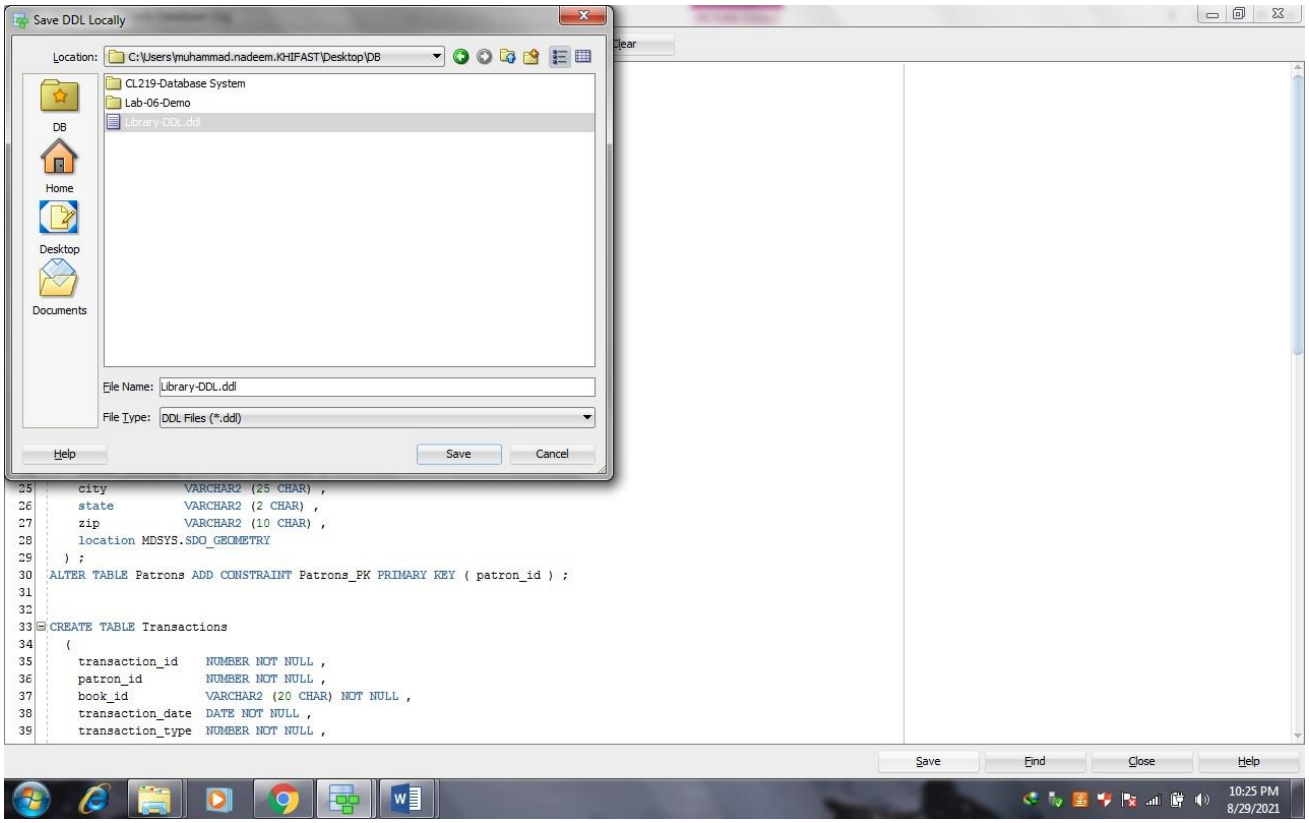
5. Click Save to save the statements to a.Sqlscript file (for example, create_library_objects.sql) on were local system.
   Later, run the script (for example, using a database connection and SQL Worksheet in SQL Developer) to create the objects in the desired database.
6. Close to close the DDL file editor.



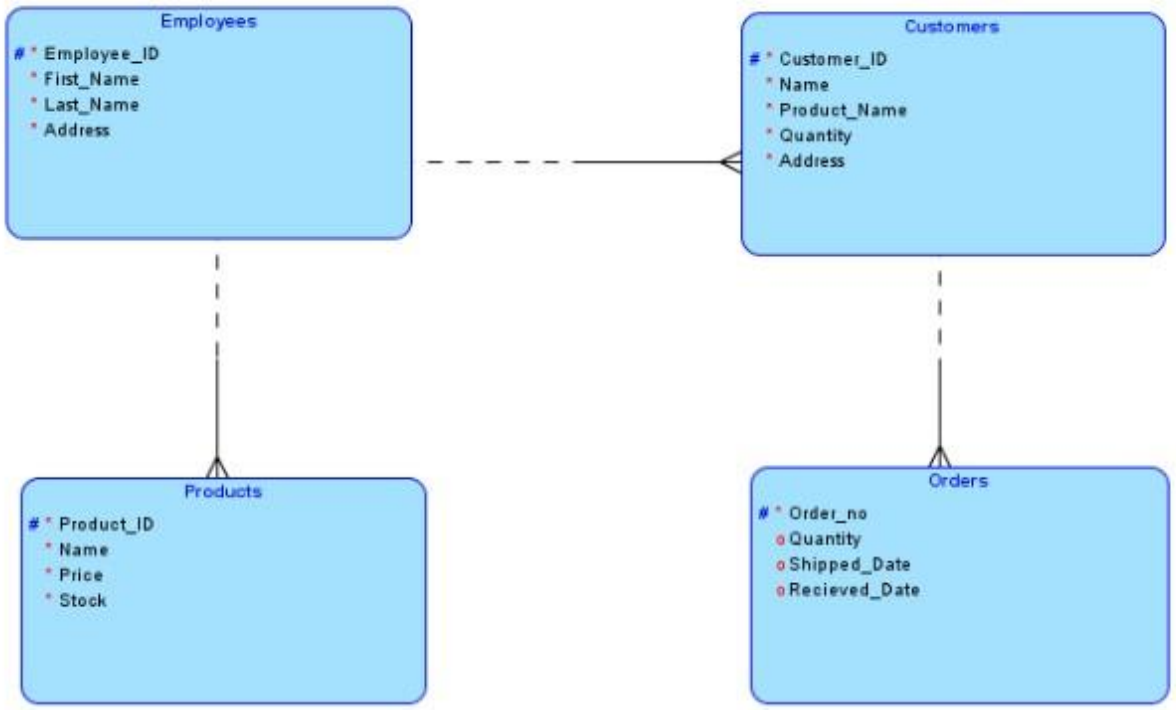
4. Save the Design

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, library_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 we wish. When we are finished, save the design again if we have made any changes, then exit SQL Developer Data Modeler by clicking File, then Exit.



Task - 1



•Create a physical design (DDL) from the above logically designed database.
•Keeping in mind the logical design, create foreign keys in each table where required.

Task -2

In this academic research collaboration network, researchers from various institutions collaborate on multiple research projects funded by diverse funding sources. Research projects result in publications authored by one or more researchers. Each researcher has unique research interests, and each publication has an associated researcher. This scenario involves aspects of managing research projects, researchers, publications, institutions, and funding sources within a highly interconnected academic environment.

Each researcher has ResearcherID that is unique, FirstName, LastName, Email, Affiliation (e.g., university or research institute) and ResearchInterests. Research Projects have ProjectID, ProjectTitle, Description, StartDate,EndDate and FundingSourceID. Publications have distinct PublicationID, Title, PublicationDate, Abstract, Type (e.g., journal article, conference paper) and ResearcherID. Institutions have a unique InstitutionID, InstitutionName, Location, Type (e.g., university, research institute). The funding source has a FundingSourceID, SourceName, SourceType (e.g., government grant, private foundation) and AmountGranted.

The following constraints apply:

1.Researchers collaborate on multiple research projects, and each project involves multiple researchers.

2.Each research project can lead to multiple publications, but each publication is associated with one project.

3.Each researcher is affiliated with one institution, but an institution can have multiple researchers.

4.Each research project is funded by one funding source, but a funding source can fund multiple projects.

a)Develop the Logical Model Diagram for the database and build the design using a data modeling tool data modeler

b)Develop the Relational Model Diagram for the database and build the design using a data modeling tool data modeler

c)Generate DDL