Assignment 2: DATABASES AND MYSQL

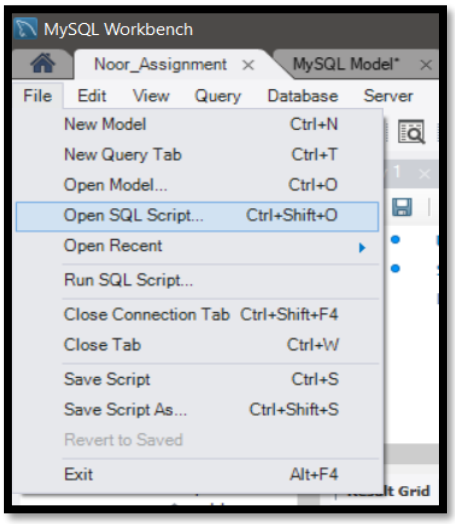
QUERYING THE DATABASE AND CREATING EER DIAGRAM

NOORULAIN FAHAD

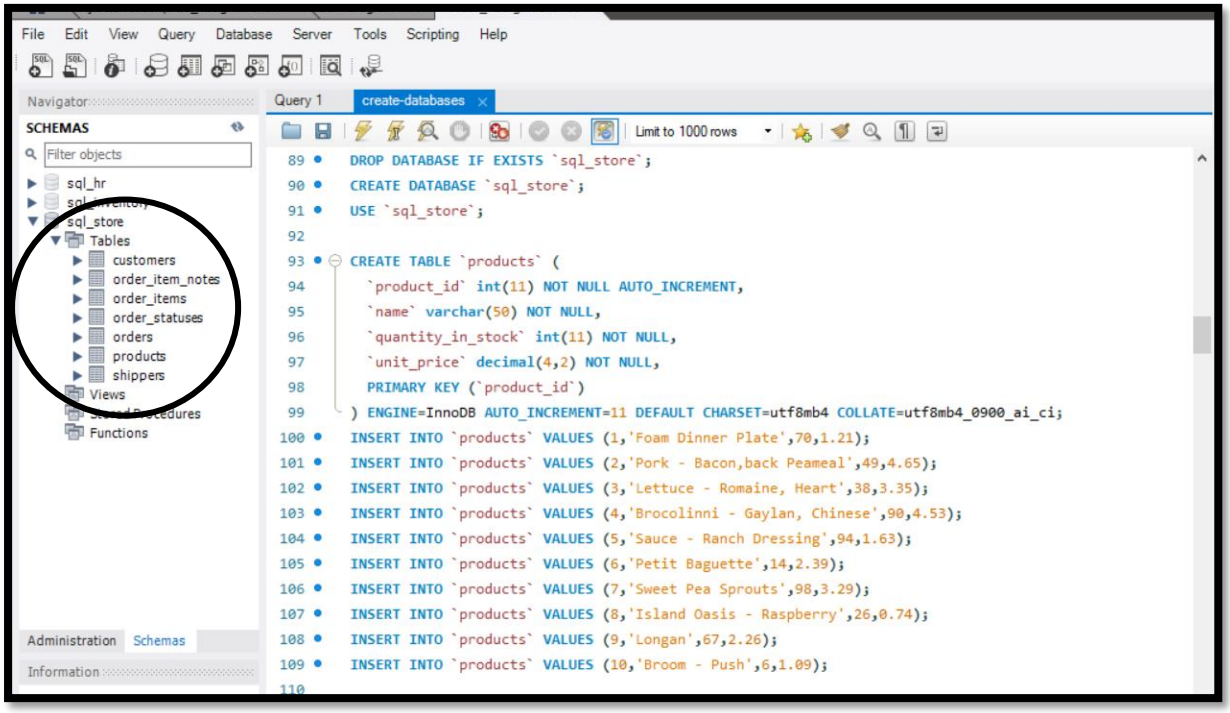
Table of Contents

QUERYING A DATABASE IN MYSQL WORKBENCH

For this project, sql script file named **create_databases.sql** is used. To open an existing SQL script file in MYSQL workbench, go to **File > Open SQL Script** or press **Ctrl + Shift + O** or click on the Open SQL Script icon on the top pane.

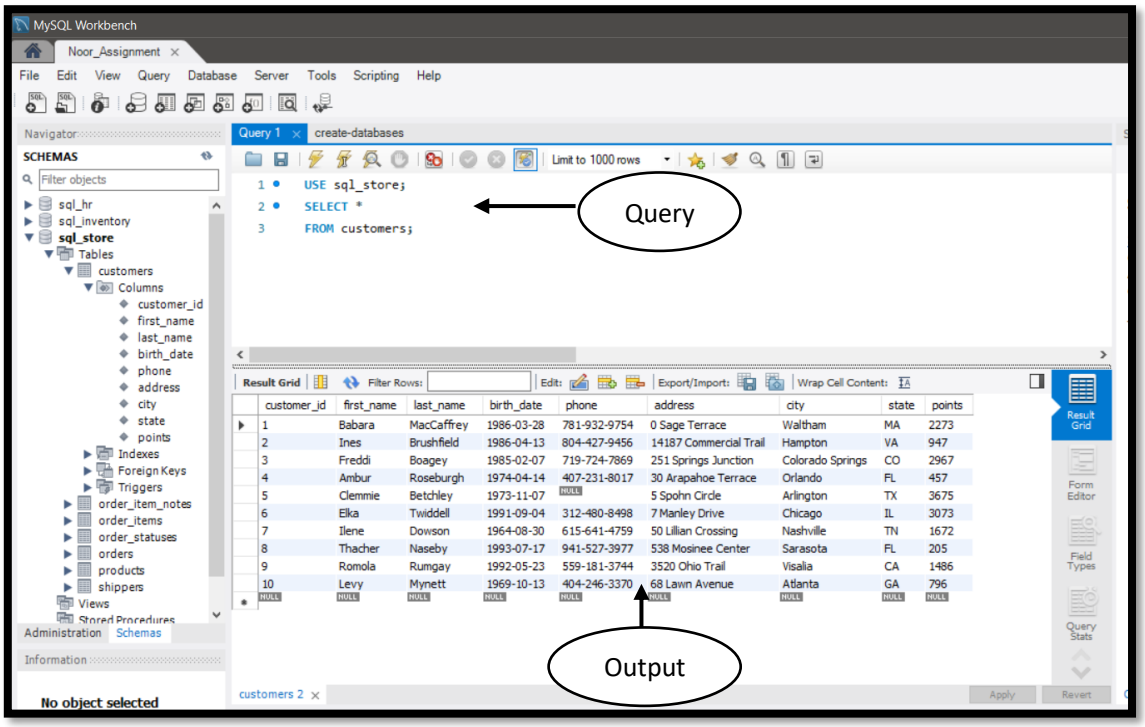


This file was executed to create the database and load the tables into it. The database sql_store has seven tables in it named customers, order_item_notes, order_items, order_statuses, orders, products and shippers.

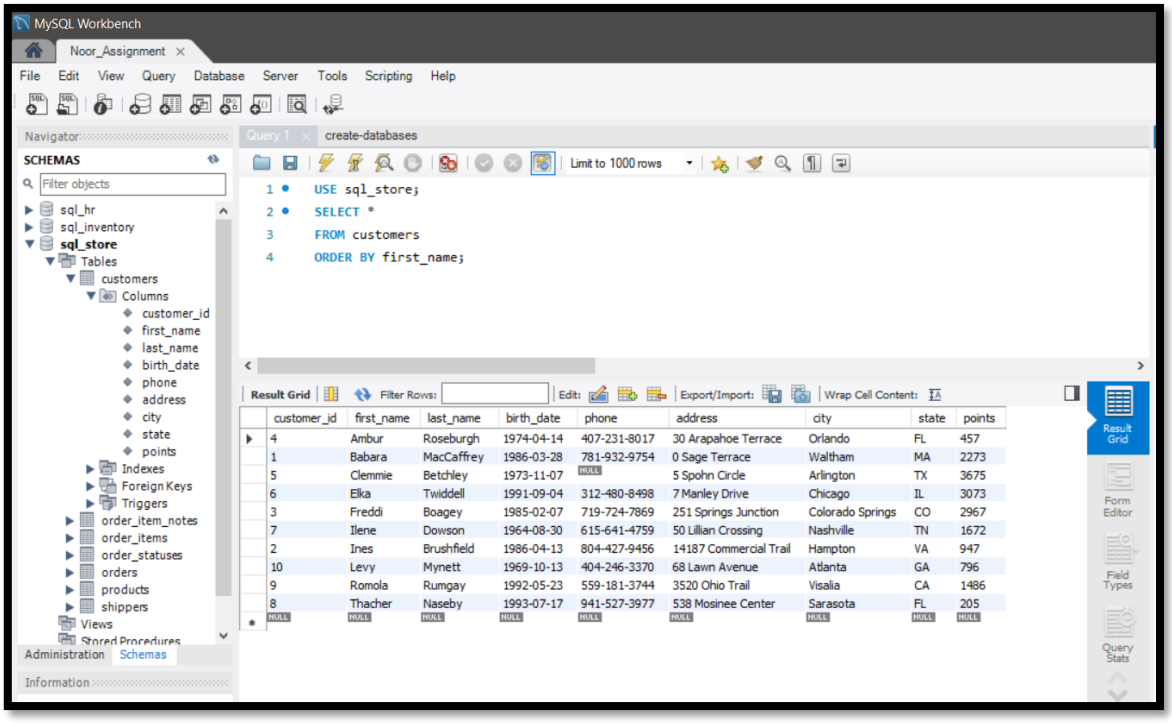


QUERY 1

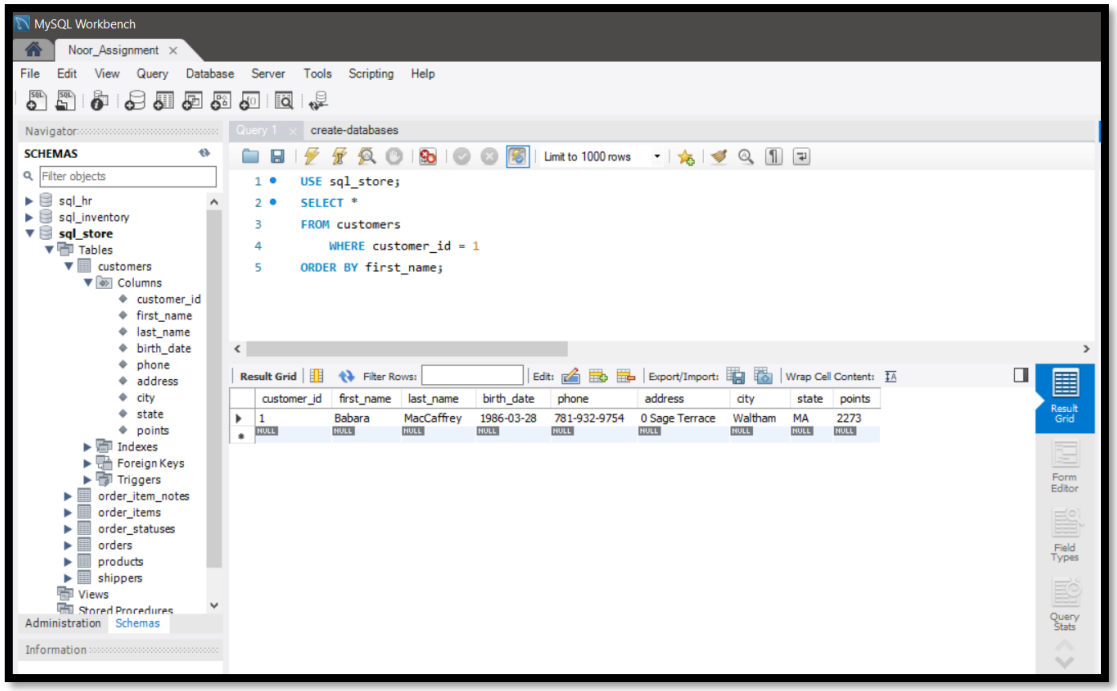
Following output was obtained after executing the Query number 1. The asterisk (*) retrieved all the columns from the customers table.



The **ORDER BY** clause sorted the output by first name of the customer.

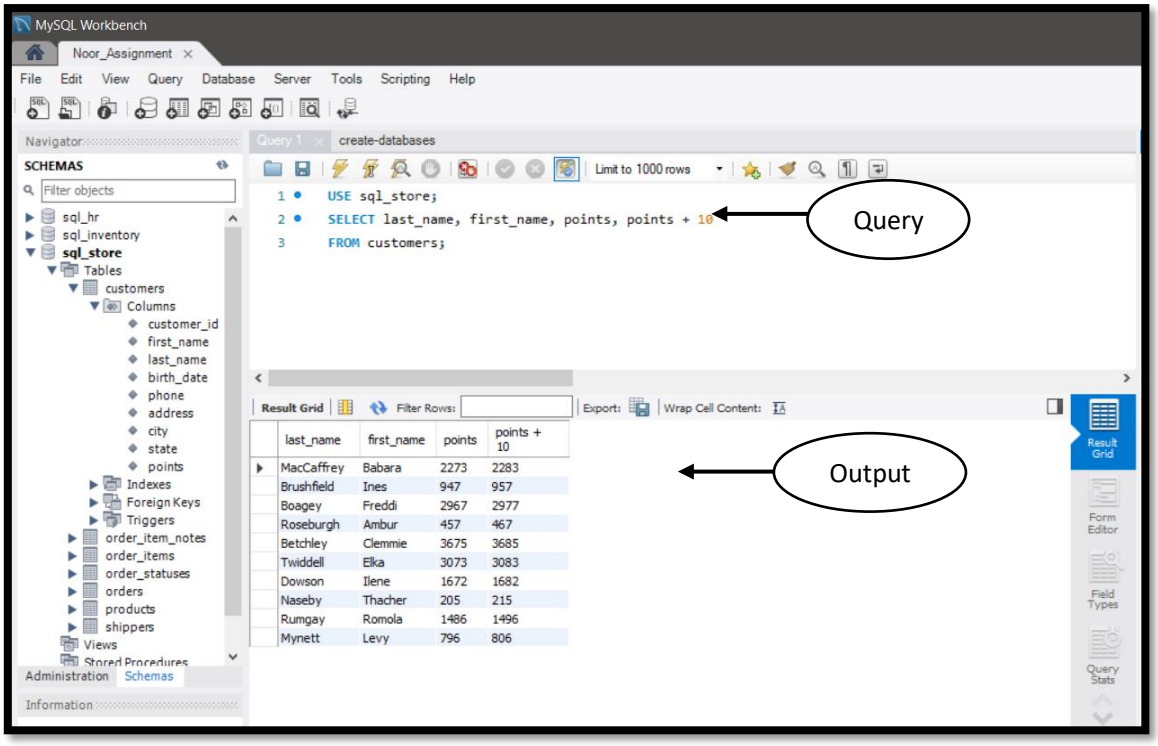


The following query returns only the customer whose customer id is 1. The **WHERE** clause returns the data when a specific condition is met.



QUERY 2

In SQL, you can perform arithmetic operations in a query to calculate values based on data stored in the database. The most common arithmetic operations used in SQL are addition (+), subtraction (-), multiplication (*), and division (/). Following output was obtained after executing the Query number 2.

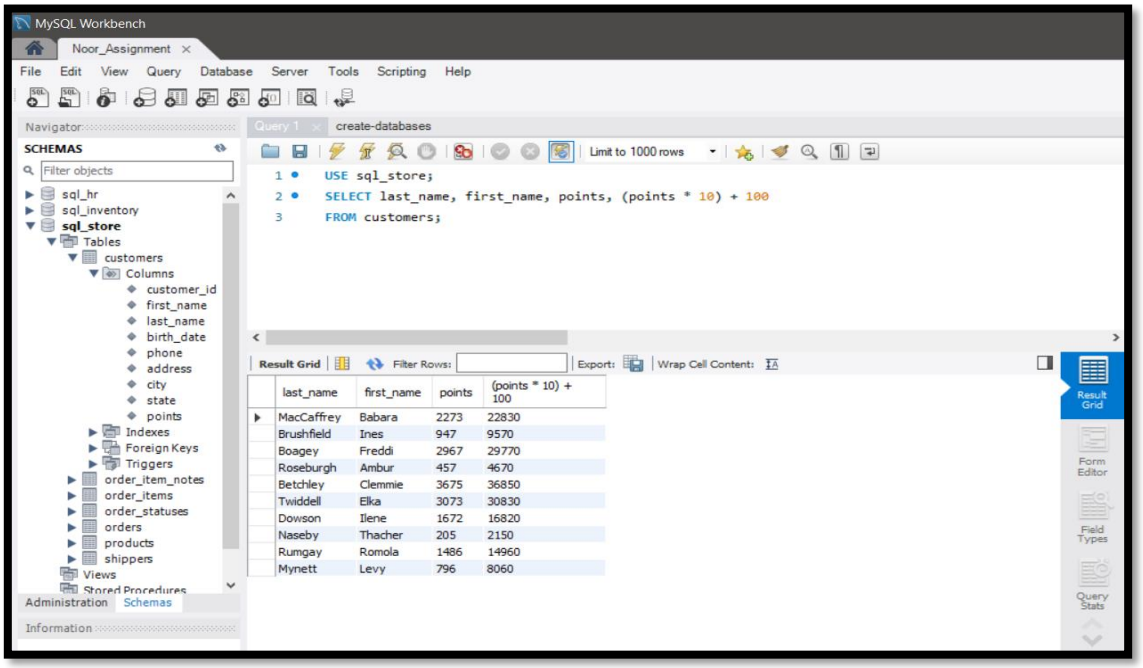


TASK - 1

Using the Query 2 you created, change the points to read times by 10 and plus 100. Record your results in your word document.

SOLUTION – 1

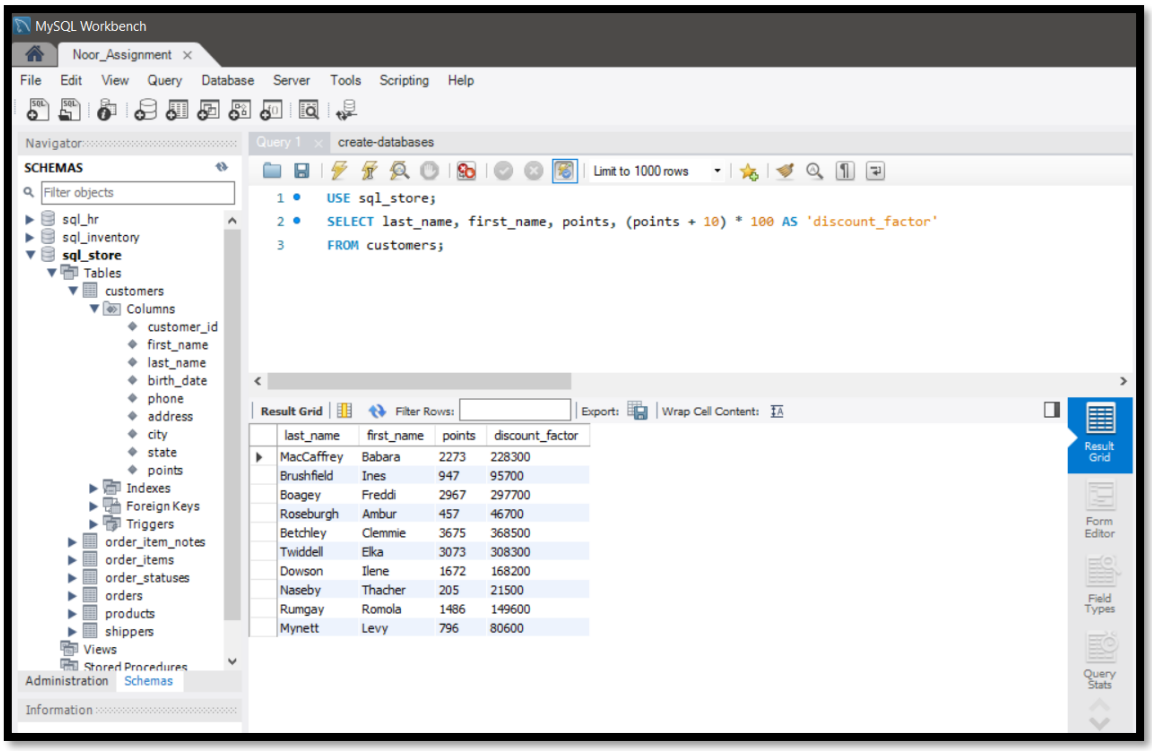
Query and Output are shown below.



Change the Query 2 code to create a discount factor, so the table now shows a discount header and changing the (point + 10) *100.

SOLUTION – 2

To create discount factor column, **AS** keyword was used. In SQL, the AS keyword is used to assign an alias, or a temporary name, to a column or a table in a query. The alias can be used in place of the original name to make the query easier to read and understand. Query and Output are shown below.



TASK – 2

Write a SQL query to return all the products in our database in the result set. I want to make three new columns, name, unit price, and new column called new price, which is based on this expression, (unit price * 1.1). So, what you are doing is increasing the product price of each by 10%. So, with the query we want all the products the original price and the new price.

SOLUTION

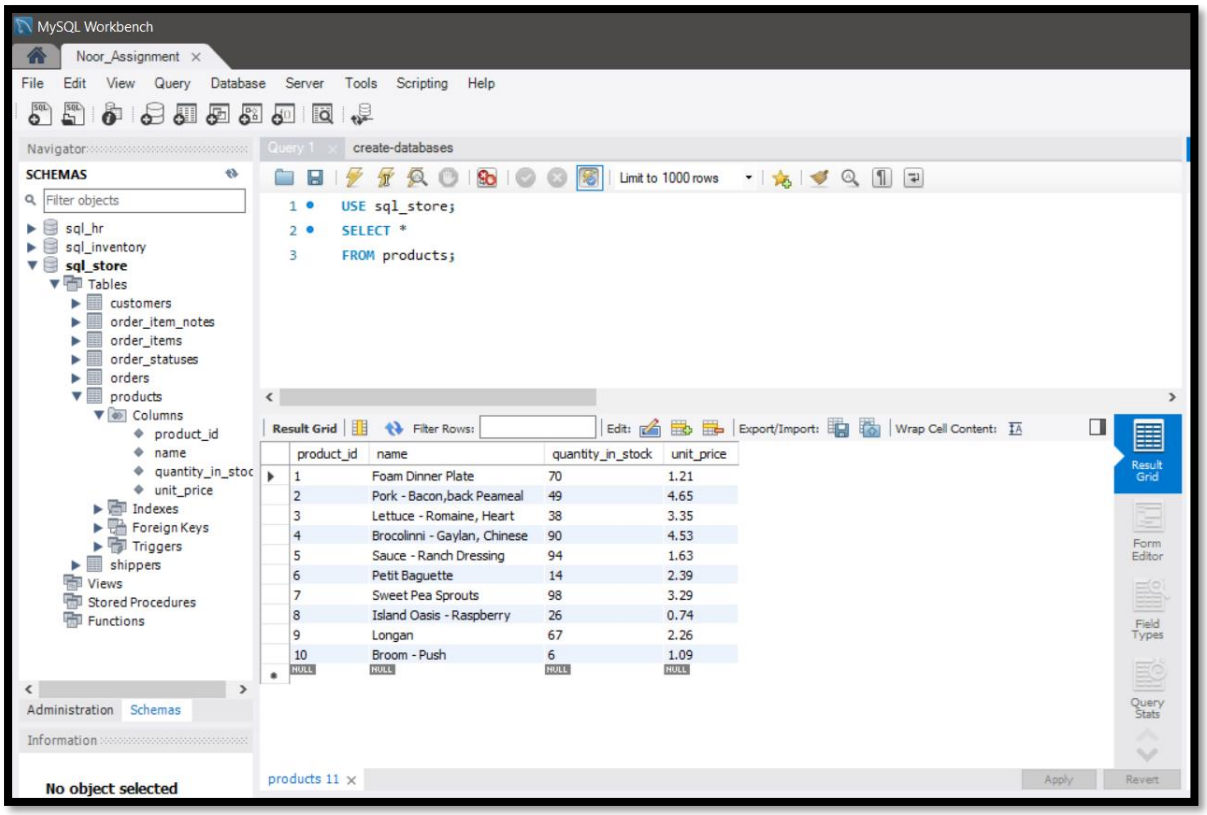
Firstly, following query was executed to check all column names in the products table.

USE sql_store;

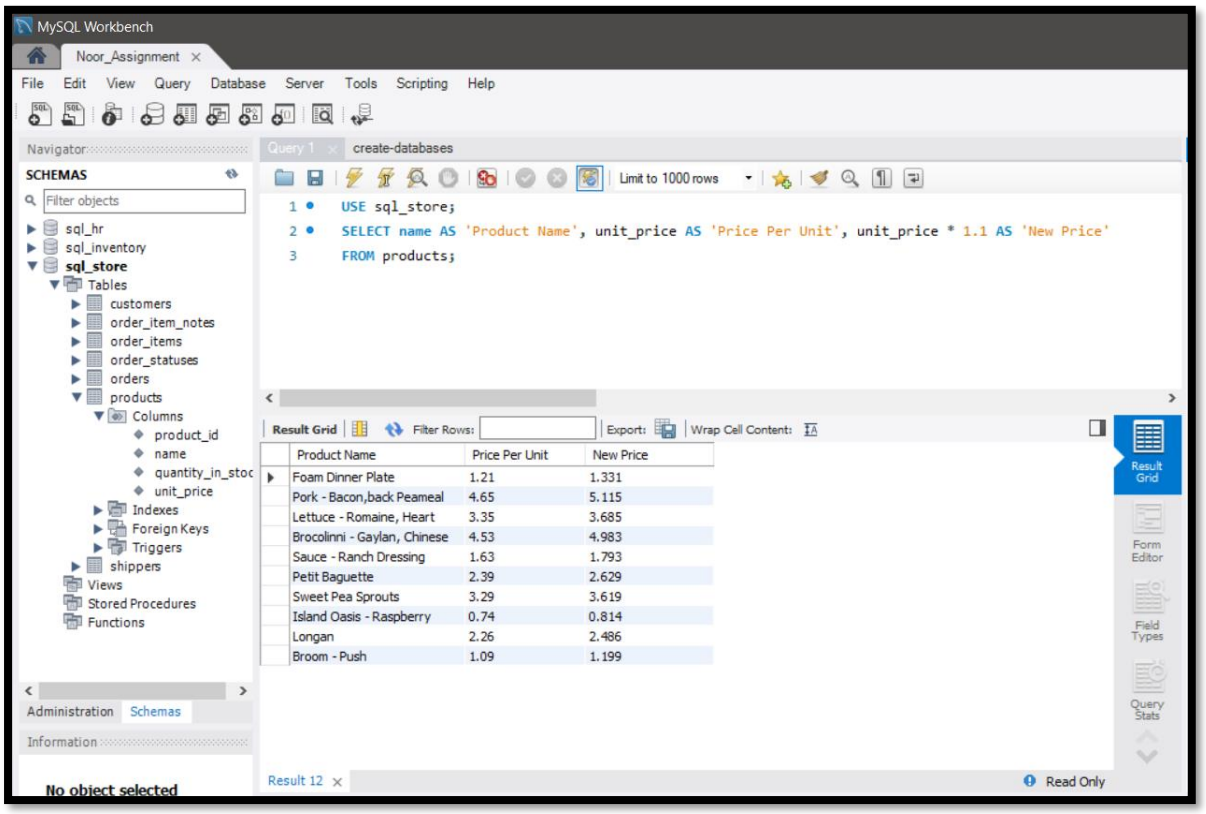
SELECT *

FROM products;

Query and Output are shown below.



Only three columns (name, unit price and new price) are required for the solution. Query and output are shown as follows.

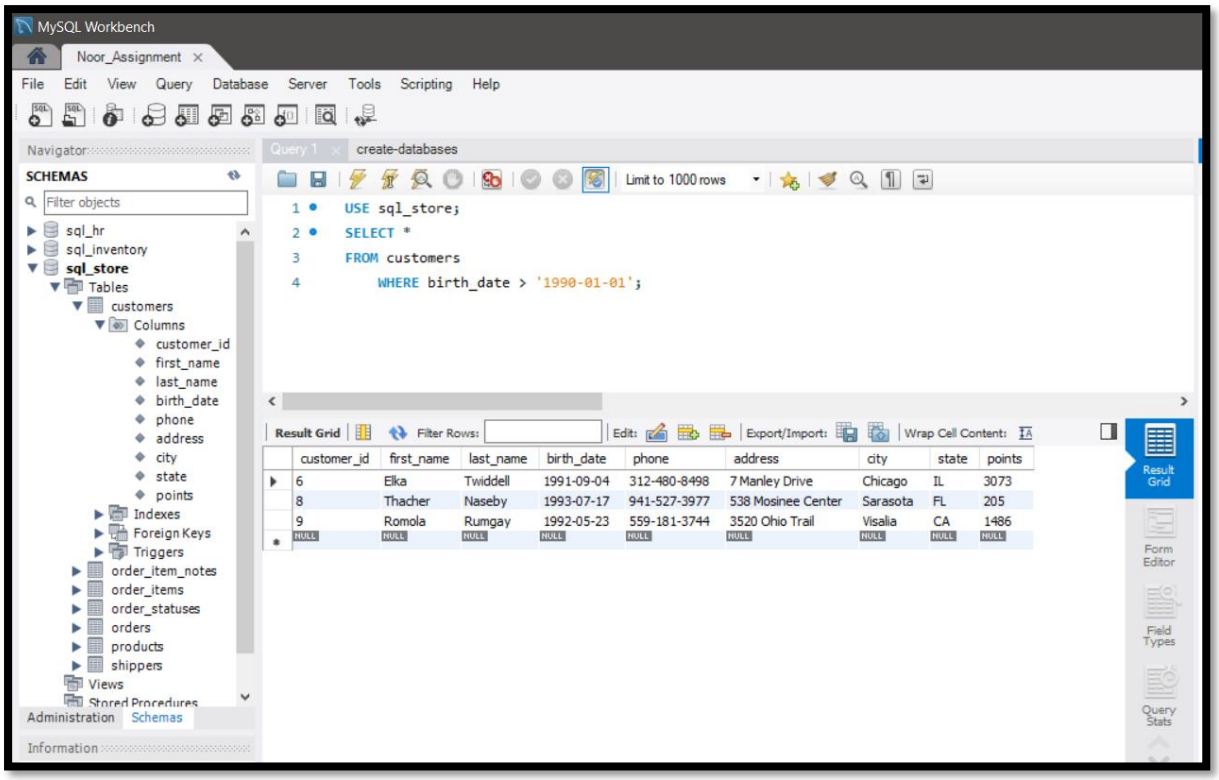


TASK-3

In this task, create a new query to find all the customers with a birth date of > '1990-01-01'.

SOLUTION

To return only the customers born after 1990, WHERE clause was used. In SQL, the **WHERE** clause is used to filter data from a table based on specific conditions. The WHERE clause can use comparison operators such as < (less than), > (greater than), <= (less than or equal to), >= (greater than or equal to), = (equal to), and <> (not equal to) to compare values and retrieve data based on specific conditions.



There are only three customers born after 1990 as seen from the output above.

REFLECTIVE

SQL stands for **Structured Query Language**. It is a standard programming language used to manage and manipulate relational databases. SQL queries are the statements used to retrieve information from a database, update existing data, insert new data, and delete data.

A SQL query typically consists of several clauses including SELECT, FROM, WHERE, GROUP BY, HAVING, and ORDER BY. The **SELECT** clause is used to specify the columns in the database that you want to retrieve. The **FROM** clause is used to specify the table or tables from which you want to retrieve the data. The **WHERE** clause is used to specify conditions that must be

met for the data to be returned. The **GROUP BY** clause is used to group data based on one or more columns. The **HAVING** clause is used to specify conditions for the groups of data that are returned. The **ORDER BY** clause is used to sort the data that is returned.

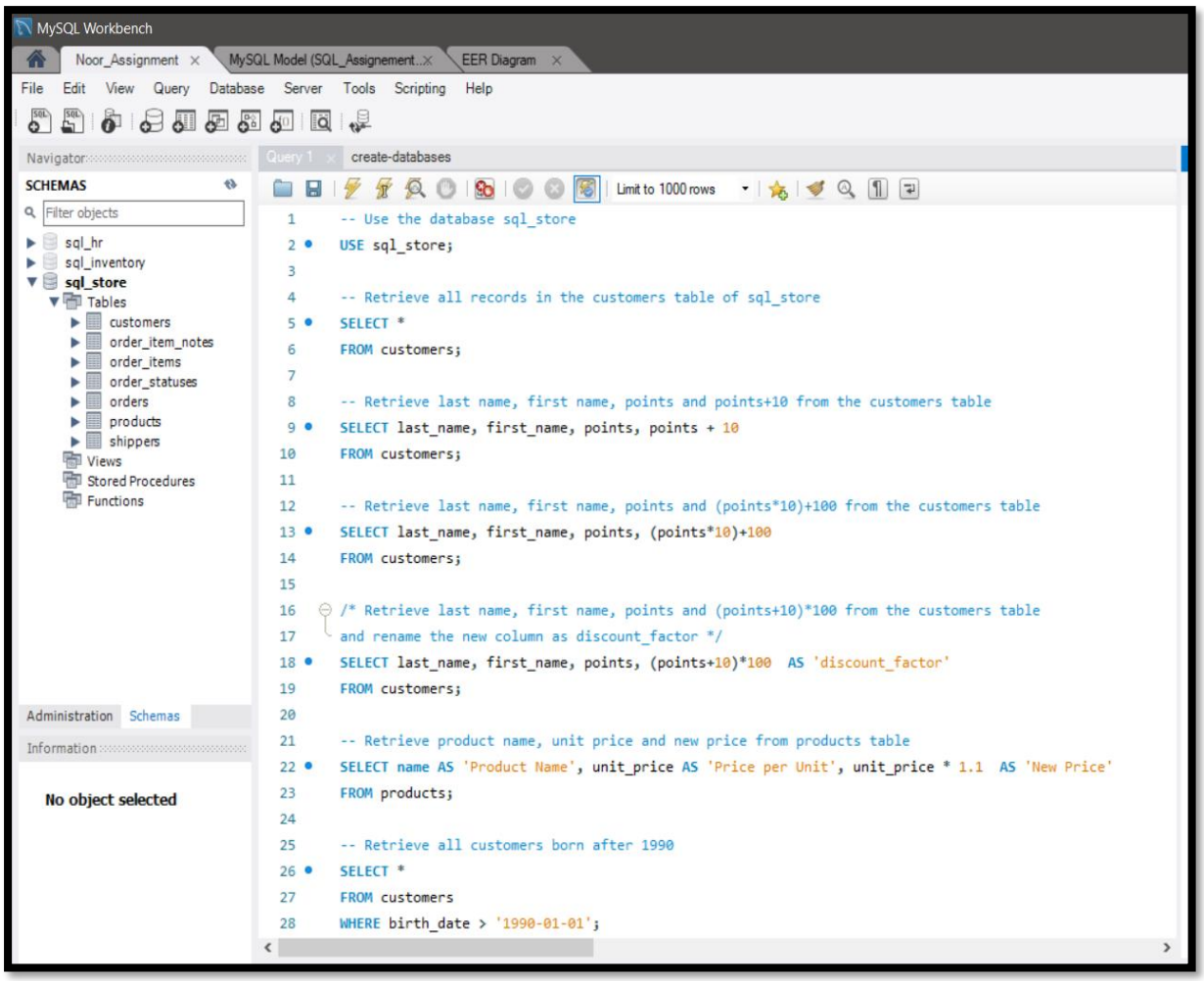
Following is an example of a simple SQL query that retrieves all the data from a table:

SELECT *

FROM [table];

This query will return all columns and all rows from the table. The **asterisk (*)** is used to specify that you want to retrieve all columns. This project gave me insight into SQL syntax, creating and viewing a database. I was also able to query a database and retrieve information. I have practiced and performed various other queries including joins on different databases as well.

The screenshot of all the queries used is shown below. To insert single line comments, - - is used. Multi-line comments are written inside **/* */**. Adding comments to a script make it more understandable.



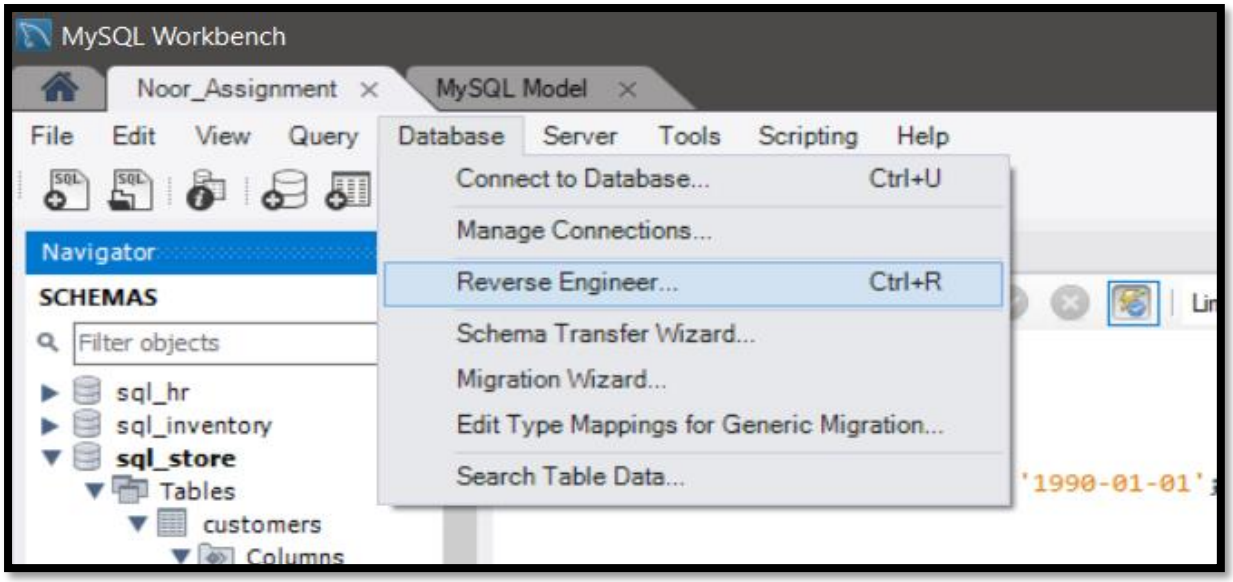
EER DIAGRAM

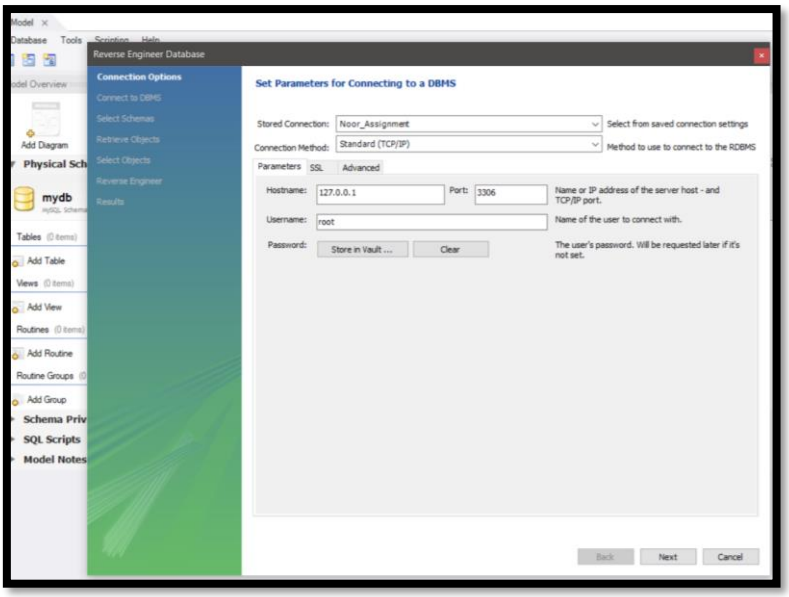
DEFINITION

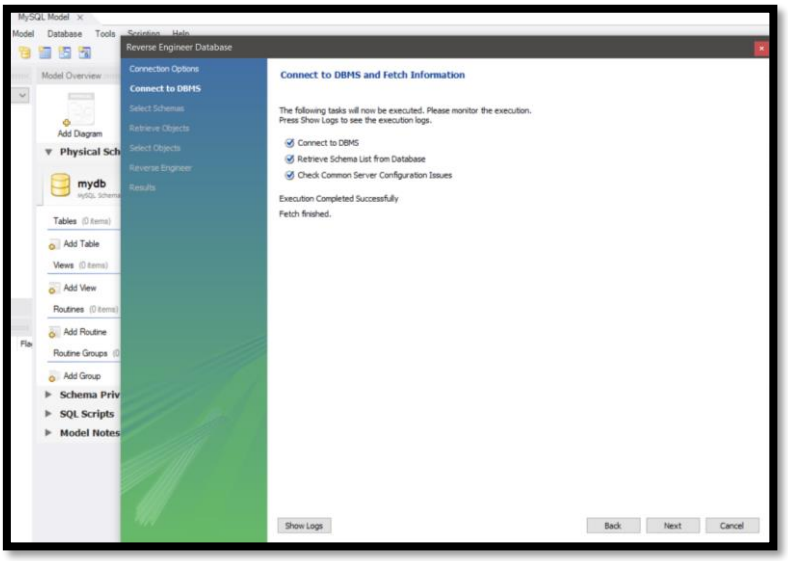
EER stands for **Enhanced Entity-Relationship**. EER diagrams are used in Data Base Management Systems (DBMS) to model the relationships between entities and attributes in a database. EER diagrams provide a visual representation of the entities, their attributes, and the relationships between them, which helps in understanding the structure of the database and the relationships between the data stored in it. These diagrams are easy to understand, both for database designers and for stakeholders who may not have a technical background.

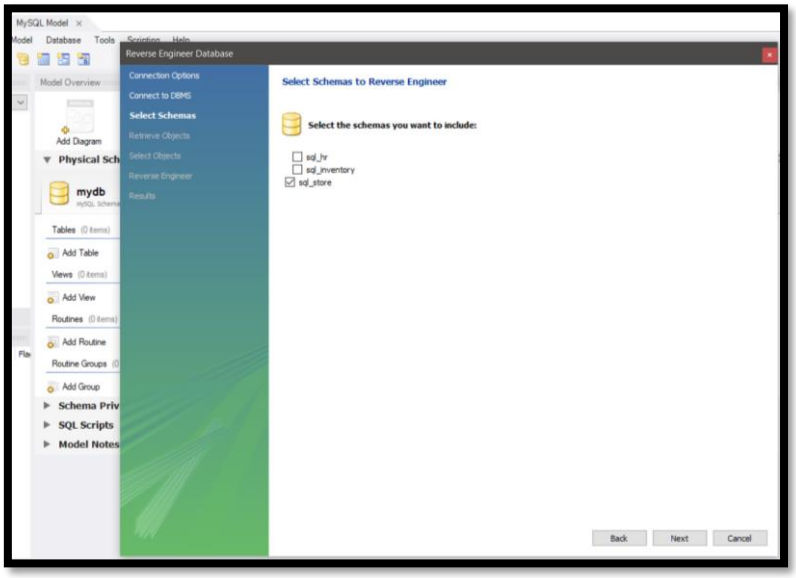
CREATING EER DIAGRAM IN MYSQL WORKBENCH

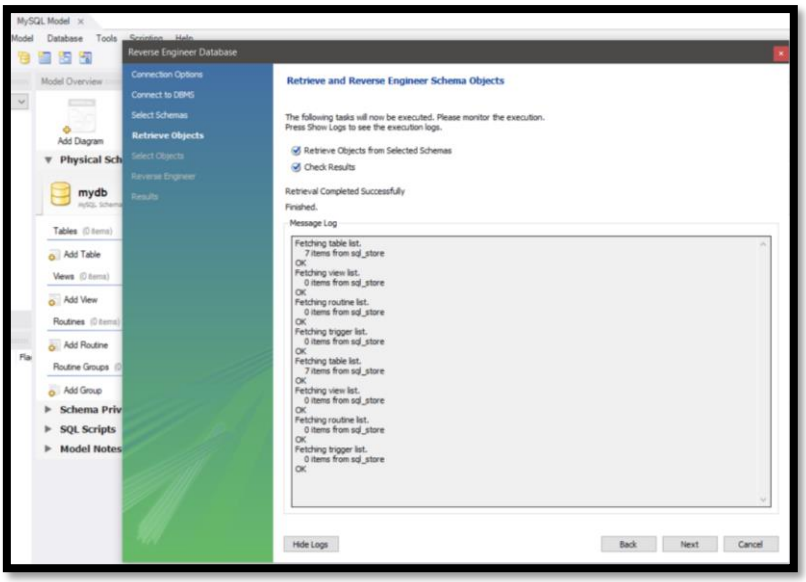
Databases are modelled using entity-relationship diagrams and this practice is called **Forward engineering**. But the process of creating entity-relationship diagram from the existing database schema is called **Reverse engineering**. Following is the step-by-step procedure to create an EER diagram in MYSQL workbench.

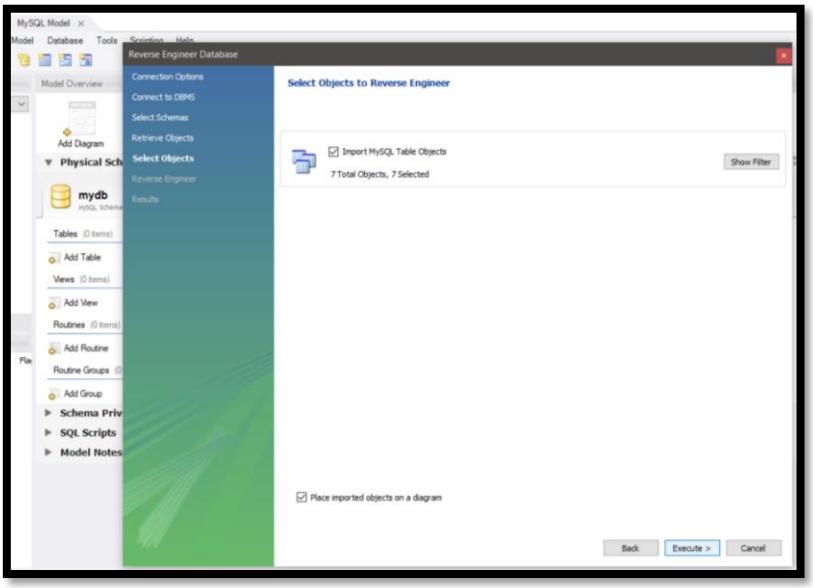


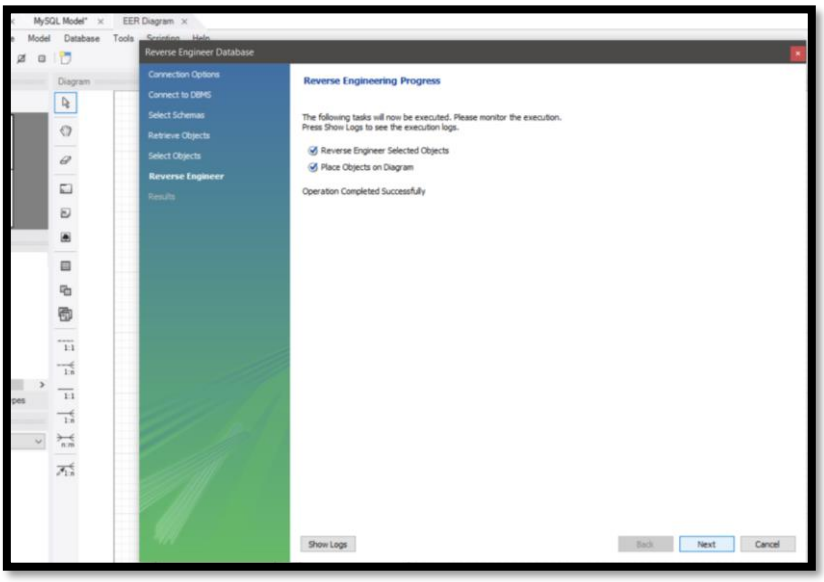

Reverse Engineer Database
Connection Options
Connect to DBMS
Select Schemas
Retrieve Objects
Select Objects
Reverse Engineer
Results
Set Parameters for Connecting to a DBMS
Stored Connection:
Noor_Assignment
Select from saved connection settings
Connection Method:
Standard (TCP/IP)
Method to use to connect to the RDBMS
Parameters
SSL
Advanced
Hostname:
127.0.0.1
Port:
3306
Name or IP address of the server host - and TCP/IP port.
Username:
root
Name of the user to connect with.
Password:
Store in Vault ...
Clear
The user's password. Will be requested later if it's not set.
Back
Next
Cancel

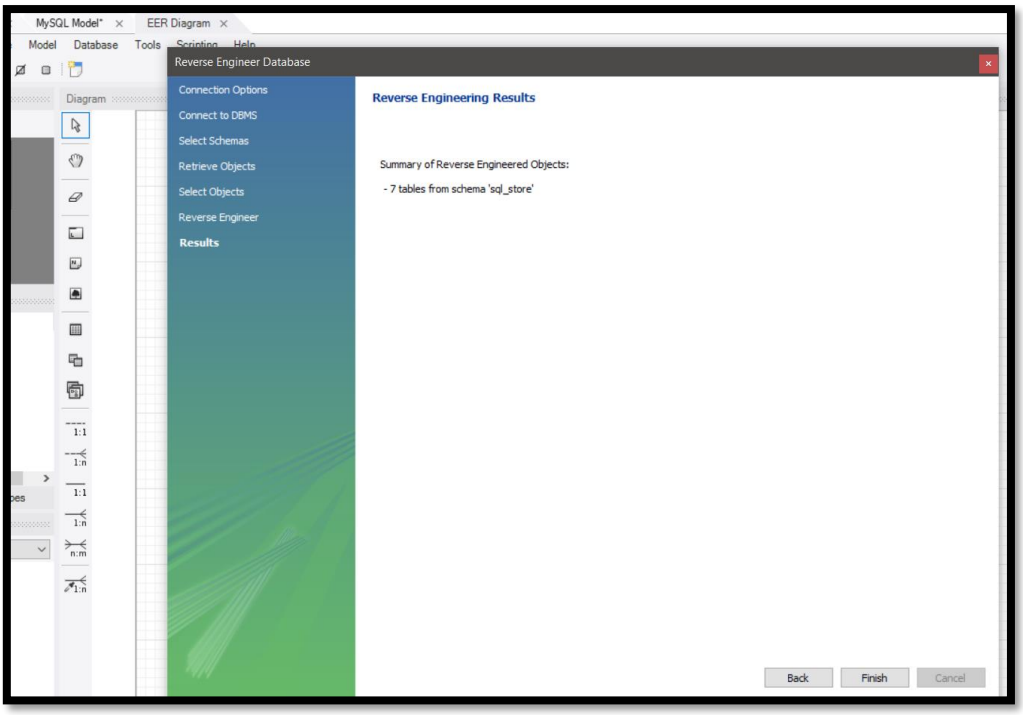

Reverse Engineer Database
Connection Options
Connect to DBMS
Select Schemas
Retrieve Objects
Select Objects
Reverse Engineer
Results
Connect to DBMS and Fetch Information
The following tasks will now be executed. Please monitor the execution.
Press Show Logs to see the execution logs.
Connect to DBMS
Retrieve Schema List from Database
Check Common Server Configuration Issues
Execution Completed Successfully
Fetch finished.
Show Logs
Back
Next
Cancel


Reverse Engineer Database
Connection Options
Connect to DBMS
Select Schemas
Retrieve Objects
Select Objects
Reverse Engineer
Results
Select Schemas to Reverse Engineer
Select the schemas you want to include:
sql_hr
sql_inventory
sql_store
Back
Next
Cancel


Reverse Engineer Database
Connection Options
Connect to DBMS
Select Schemas
Retrieve Objects
Select Objects
Reverse Engineer
Results
Retrieve and Reverse Engineer Schema Objects
The following tasks will now be executed. Please monitor the execution.
Press Show Logs to see the execution logs.
Retrieve Objects from Selected Schemas
Check Results
Retrieval Completed Successfully
Finished.
Message Log
Hide Logs
Back
Next
Cancel

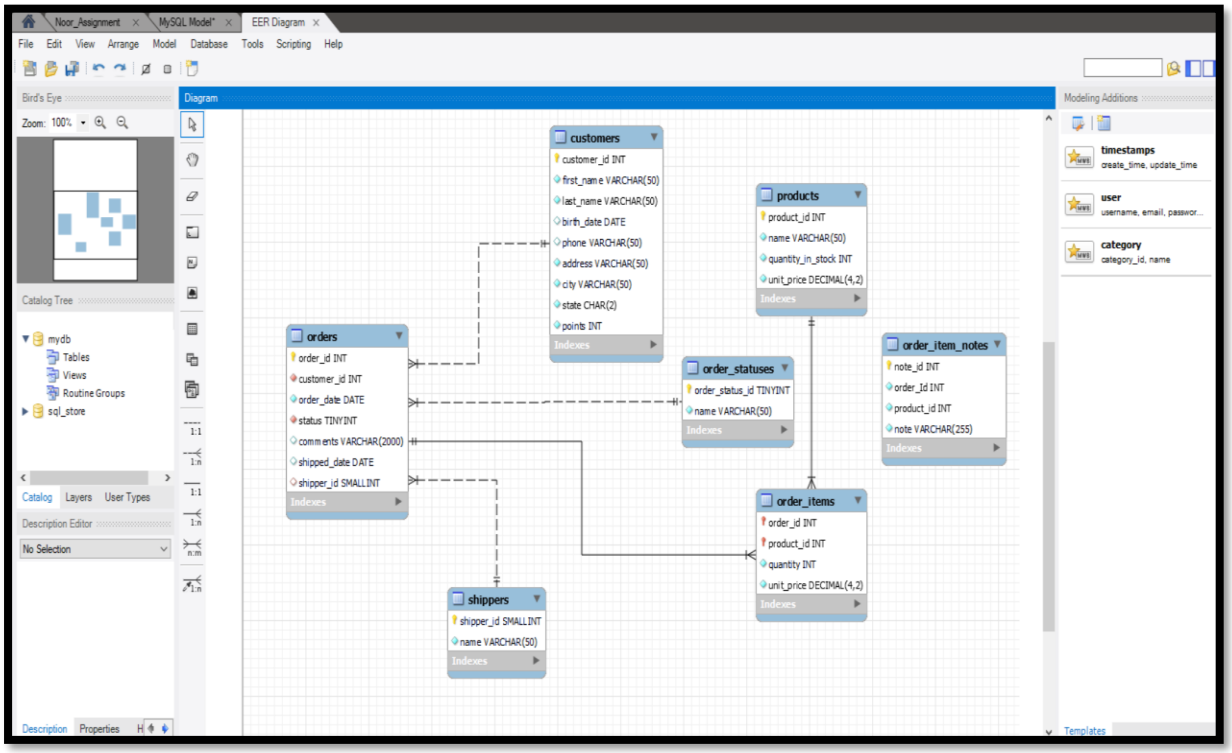

Reverse Engineer Database
Connection Options
Connect to DBMS
Select Schemas
Retrieve Objects
Select Objects
Reverse Engineer
Results
Select Objects to Reverse Engineer
Import MySQL Table Objects
7 Total Objects, 7 Selected
Show Filter
Place imported objects on a diagram
Back
Execute >
Cancel


Reverse Engineer Database
Connection Options
Connect to DBMS
Select Schemas
Retrieve Objects
Select Objects
Reverse Engineer
Results
Reverse Engineering Progress
The following tasks will now be executed. Please monitor the execution.
Press Show Logs to see the execution logs.
Reverse Engineer Selected Objects
Place Objects on Diagram
Operation Completed Successfully
Show Logs
Back
Next
Cancel

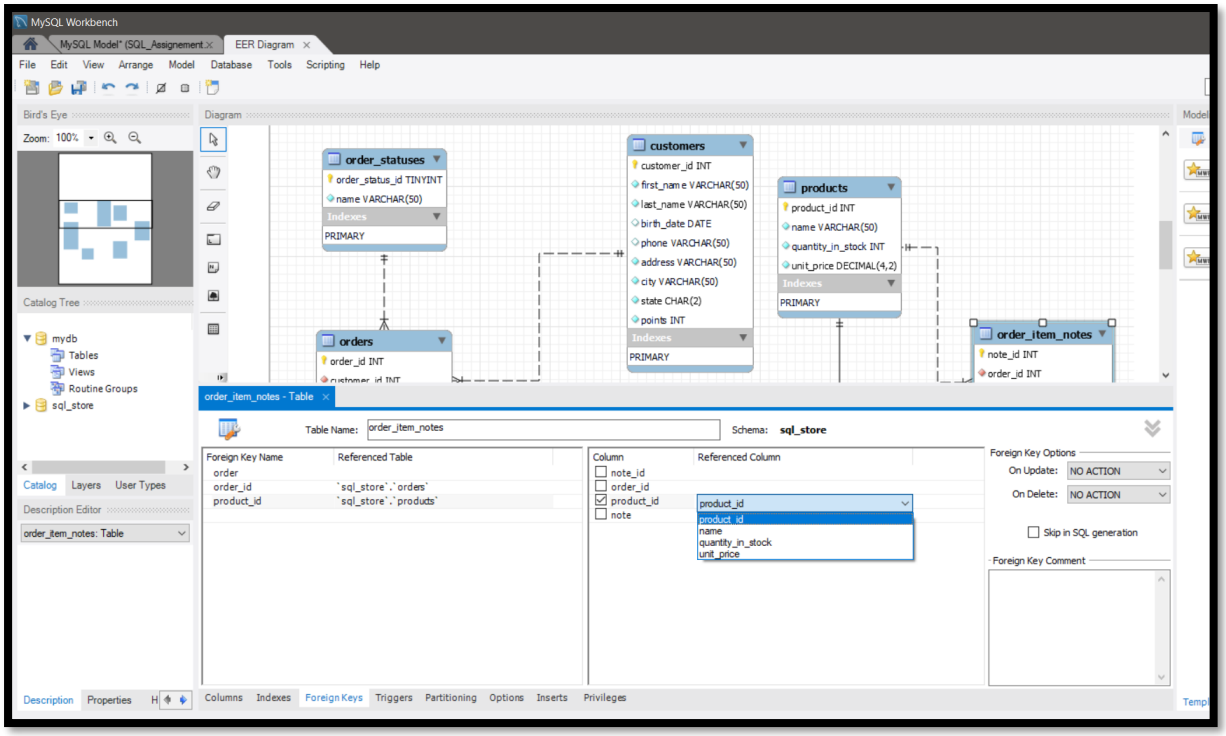


SOLUTION

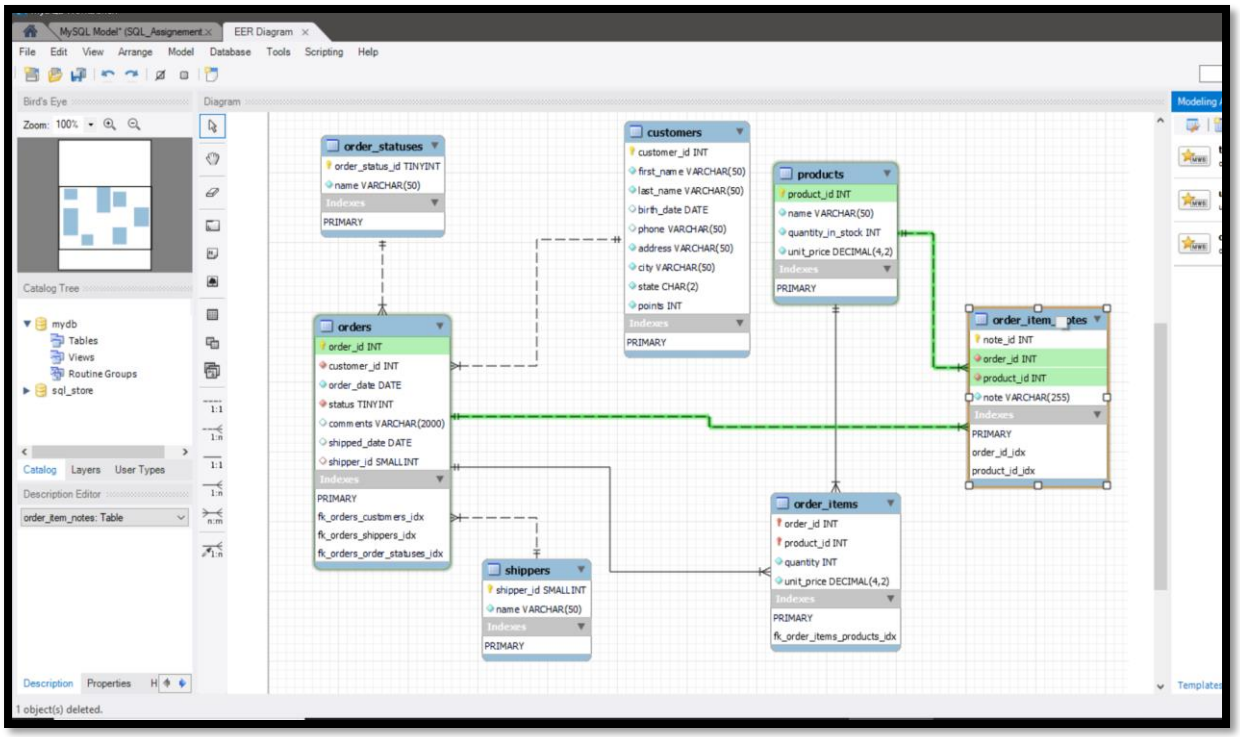
Following EER diagram is created, but the mySQL was unable to identify the relationship between **order_item_notes** table and other tables in the database. By looking at the attributes of order_item_notes table, you can identify the primary and foreign keys and can assign them manually.



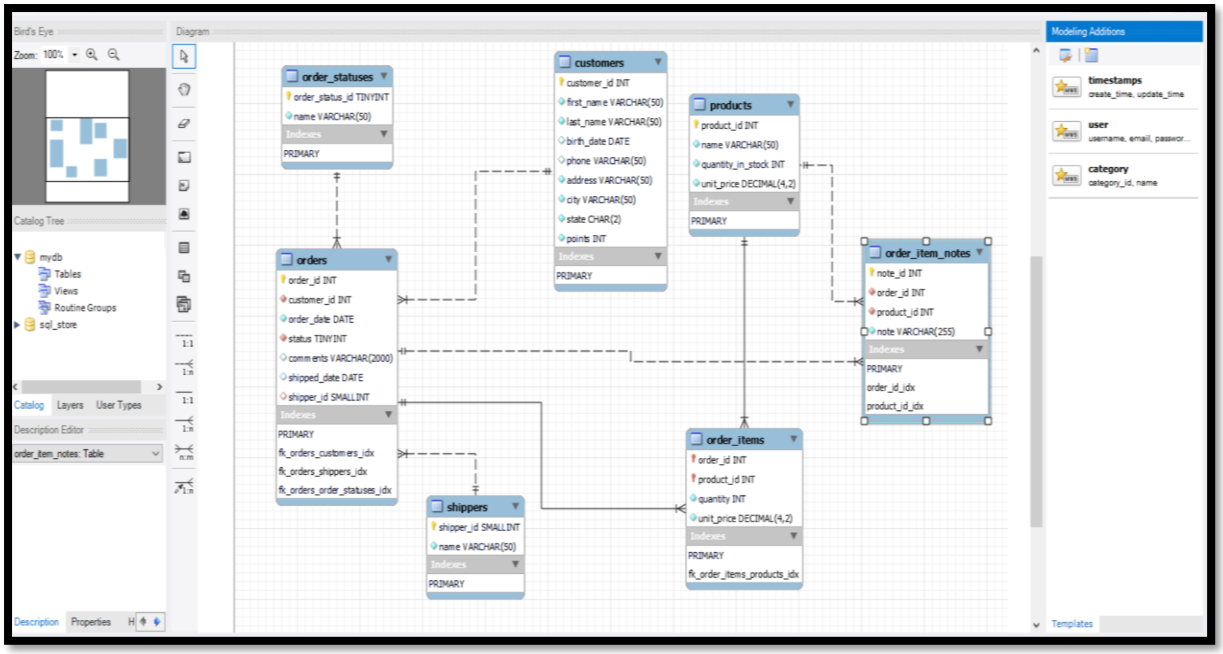
In this case, **note_id** is the Primary key and there will be two foreign keys **order_id** and **product_id** being referenced from **orders** and **products** table respectively. You can represent this by assigning them manually by double clicking on the order_item_notes table as shown below.



After manually entering the primary and foreign keys, it now shows the relationship between the order_item_notes table with order and products tables respectively.



The final EER diagram for the create-databases.sql is shown below.



REFLECTIVE

EER diagram from sql_store database shows all the tables it contains and their primary keys and how they communicate to other tables in the database using foreign keys. These diagrams are useful to understand the relationships between different entities in the database and how they communicate to each other. This is particularly helpful when we are performing joins to combine to tables based on their common attribute.