CREATING AN LIBRARY DATABASE USING SQL THROUGH ORACLE SERVER

A Case Study submitted by:

*S Sri Harsha Teja*, VU22CSEN0101809

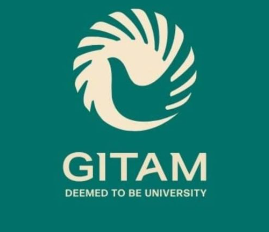
*G Praneeth*, VU22CSEN0101844

*U Sai Rahul*, VU22CSEN0101745

*K Manas Srikar*, VU22CSEN0101808

*Akhil Malla*, VU22CSEN0101649

Under the esteemed guidance of Mrs. P. Manasa Devi Assistant Professor

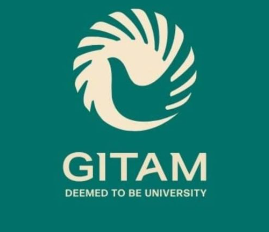


## DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING GITAM

(Deemed to be University) OCTOBER 2024

## DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING GITAM

(Deemed to be University) OCTOBER 2024



**DECLARATION**

I/We hereby declare that the case study entitled “**CREATING AN ONLINE RETAIL DATABASE USING SQL THROUGH ORACLE SERVER”** is an

original work done in the Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University) submitted in partial fulfillment of requirements of completing the course of CSEN2061- DATABASE MANAGEMENT SYSTEM LAB. The work hasn’t been lend to any other team or Individual.

## DATE:

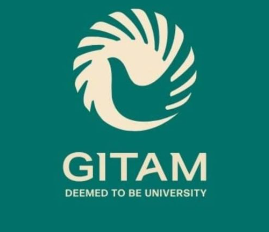
**TEAM DETAILS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **REGD.NO** | **NAME** | **SIGNATURE** |
| 1 | VU22CSEN0101809 | S Sri Harsha Teja |  |
| 2 | VU22CSEN0101844 | GVNSS  Praneeth |  |
| 3 | VU22CSEN0101745 | U Sai Rahul |  |
| 5 | VU22CSEN0101808 | K Manas Srikar |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 6 | VU22CSEN0101649 | Akhil Malla |  |

## DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING GITAM

(Deemed to be University) OCTOBER 2024



## CERTIFICATE

This is to certify that the following case study entitled “**CREATING AN ONLINE RETAIL DATABASE USING SQL THROUGH ORACLE**

**SERVER”** is a bonafide record done by Chintaluru Pavan

(VU22CSEN0101566), Satya Akshaya Pinnamaraju (VU22CSEN0101718) ,

Terlapu Harika Sai (VU22CSEN0100905), Malepati Nithish Reddy (VU22CSEN0101616), Gundala Nikhil (VU22CSEN0101706) submitted in partial fulfillment of requirements of completing the course of CSEN2061- DATABASE MANAGEMENT SYSTEM LAB.

PROJECT GUIDE HEAD OF THE DEPARTMENT

Mrs. P. MANASA DEVI GONDI LAKSHMEESHWARI ASSISTANT PROFESSOR

|  |  |  |
| --- | --- | --- |
|  | Table of Contents |  |
| Absract |  | 4 |
| Part-1 |  | 6-9 |
| ER Model |  | 7 |
| Conceptual Schema |  | 8 |
| Part-2:Relational Mode  Relational Model DDL Commands |  | 9-16  9 |
| DML Commands |  | 10 |
| DCL Commands |  | 12 |
| TCL Commands |  | 12 |
| DQL Command |  | 12 |
| Part-3:Advanced SQL |  | 16-18 |
| Nested Queries |  | 16 |
| Correlated Queries |  | 16 |
| Aggregate Operators |  | 17 |
| Relational Operators |  | 17 |
| Part-4:Joins And Views |  | 18-24 |
| Joins |  | 18 |
| Views |  | 19 |
| Display Table |  | 19 |

# Abstract

The aim of this Case Study is to create a database of online retail websites using SQL in Oracle. We will briefly look into what online retail is:

Online retail is a type of eCommerce whereby a business sells goods or services directly to consumers from a website. The website may be their own, or it may be owned by a larger retailer or marketplace like Amazon. Online retail is a similar concept to brick-and-mortar retail. Shoppers enter the store, search through an organized inventory of products, and then pay for their goods at

checkout. It’s just that online retail takes place over the Internet while brick- and-mortar is done in person.

Objectives of the Study This study aims to:

Develop a Structured Database: Create a database schema that organizes various online retail platforms and their attributes.

Utilize SQL for Data Management: Implement SQL commands to define tables, relationships, and constraints relevant to online retail data.

Analyze Data Trends: Use SQL queries to extract insights regarding consumer behavior and market trends from the database.

Methodology

The research will involve:

Database Creation: Utilizing Oracle SQL commands such as CREATE DATABASE and CREATE TABLE to establish the database structure.

Data Insertion: Populating the database with relevant data about online retailers, including product categories, pricing, and customer reviews.

Query Execution: Running SQL queries to analyze and retrieve data for further insights into online retail dynamics.

Significance of the Study

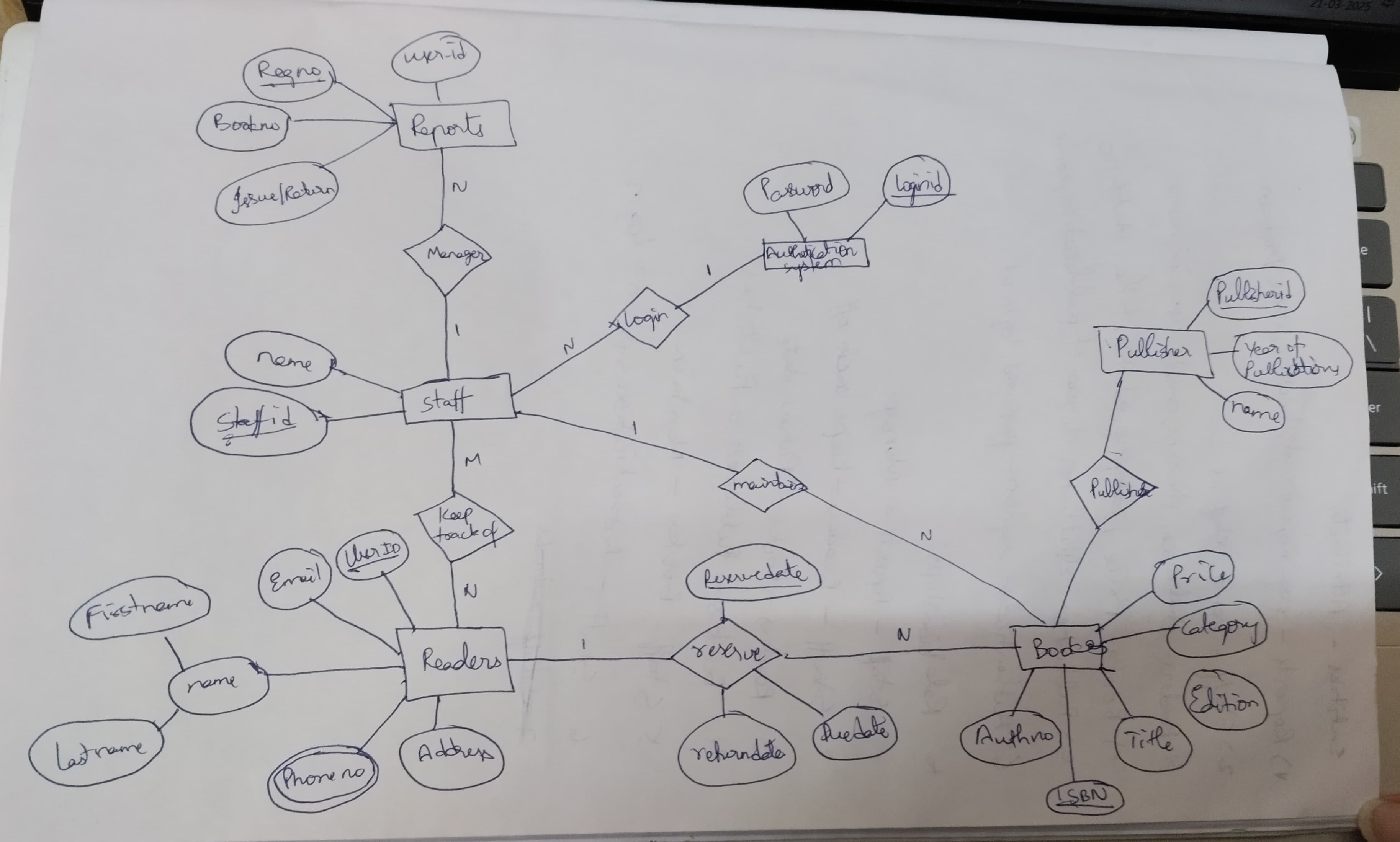
Understanding the landscape of online retail through a structured database is crucial for businesses aiming to optimize their eCommerce strategies. By

leveraging Oracle SQL, this study will provide valuable insights that can help stakeholders make informed decisions regarding product offerings, marketing strategies, and customer engagement practices.

In summary, this case study will contribute to the broader understanding of online retail dynamics and assist businesses in navigating the complexities of the digital marketplace through effective database management.

PART-1:

**ER MODEL**



# CONCEPTUAL SCHEMA :

Customer(

customer\_id: Primary Key,

contact\_no: INTEGER,

first\_name: VARCHAR2(10),

last\_name: VARCHAR2(10),

home\_address: VARCHAR2(100),

customer\_Password: VARCHAR2(20),

email\_address: VARCHAR2(50)

);

Account(

Acc\_no: INTEGER PRIMARY KEY,

Acc\_type: VARCHAR2(20),

Balance: INTEGER

);

Payment(

Payment\_ID: Primary Key,

Amount: INTEGER,

Payment\_Type: VARCHAR2(20),

Customer\_ID: INTEGER -- Foreign Key referencing Customer(customer\_id)

);

Product(

ProductID: Primary Key,

ProductName: VARCHAR2(100),

ProductDescription: VARCHAR2(255),

Price: DECIMAL(10, 2)

);

Orders(

OrderID: Primary Key,

CustomerID: INTEGER, -- Foreign Key referencing Customer(customer\_id)

OrderDate: DATE,

Status: VARCHAR2(20)

);

Order\_Item(

OrderItemID: Primary Key,

OrderID: INTEGER, -- Foreign Key referencing Orders(OrderID)

ProductID: INTEGER, -- Foreign Key referencing Product(ProductID)

Quantity: INTEGER

);

Cart(

Cart\_ID: Primary Key,

Total\_Cost: INTEGER,

Customer\_ID: INTEGER -- Foreign Key referencing Customer(customer\_id)

);

Cart\_Item(

Cart\_Item\_ID: Primary Key,

Quantity: INTEGER,

Total\_Cost: INTEGER,

Cart\_ID: INTEGER -- Foreign Key referencing Cart(Cart\_ID)

);

Discount(

Discount\_ID: Primary Key,

Discount\_Percentage: INTEGER,

Start\_Date: DATE,

End\_Date: DATE

);

Author(

Author\_ID: Primary Key,

Name: VARCHAR2(100),

Bio: CLOB

);

Category(

Category\_ID: Primary Key,

Name: VARCHAR2(50),

Description: CLOB

);

Book(

Book\_ID: Primary Key,

Title: VARCHAR2(100),

Author\_ID: INTEGER, -- Foreign Key referencing Author(Author\_ID)

ISBN: VARCHAR2(20),

Publisher\_ID: INTEGER, -- Foreign Key referencing Publisher(Publisher\_ID)

Category\_ID: INTEGER, -- Foreign Key referencing Category(Category\_ID)

Year\_Published: INTEGER,

Number\_of\_Copies: INTEGER

);

Member(

Member\_ID: Primary Key,

FirstName: VARCHAR2(50),

LastName: VARCHAR2(50),

PhoneNo: VARCHAR2(20),

EmailAddress: VARCHAR2(100),

JoinDate: DATE

);

Borrowing(

BorrowingID: Primary Key,

Member\_ID: INTEGER, -- Foreign Key referencing Member(Member\_ID)

Book\_ID: INTEGER, -- Foreign Key referencing Book(Book\_ID)

BorrowDate: DATE,

DueDate: DATE,

ReturnDate: DATE

);

Reservation(

ReservationID: Primary Key,

Member\_ID: INTEGER, -- Foreign Key referencing Member(Member\_ID)

Book\_ID: INTEGER, -- Foreign Key referencing Book(Book\_ID)

ReservationDate: DATE,

ExpiryDate: DATE

);

Return(

ReturnID: Primary Key,

BorrowingID: INTEGER, -- Foreign Key referencing Borrowing(BorrowingID)

ReturnDate: DATE,

Condition: VARCHAR2(50) -- e.g., Good, Damaged, Lost

);

Fines(

FineID: Primary Key,

Member\_ID: INTEGER, -- Foreign Key referencing Member(Member\_ID)

Amount: DECIMAL(10, 2),

FineDate: DATE,

Paid: CHAR(1) DEFAULT 'N' -- 'Y' for Yes, 'N' for No

);

Writes(

Author\_ID: INTEGER, -- Foreign Key referencing Author(Author\_ID)

Book\_ID: INTEGER, -- Foreign Key referencing Book(Book\_ID)

PRIMARY KEY (Author\_ID, Book\_ID)

);

Publishes(

Publisher\_ID: INTEGER, -- Foreign Key referencing Publisher(Publisher\_ID)

Book\_ID: INTEGER, -- Foreign Key referencing Book(Book\_ID)

PRIMARY KEY (Publisher\_ID, Book\_ID)

);

Borrows(

Member\_ID: INTEGER, -- Foreign Key referencing Member(Member\_ID)

BorrowingID: INTEGER, -- Foreign Key referencing Borrowing(BorrowingID)

PRIMARY KEY (Member\_ID, BorrowingID)

);

Reserves(

Member\_ID: INTEGER, -- Foreign Key referencing Member(Member\_ID)

ReservationID: INTEGER, -- Foreign Key referencing Reservation(ReservationID)

PRIMARY KEY (Member\_ID, ReservationID)

);

Has(

BorrowingID: INTEGER, -- Foreign Key referencing Borrowing(BorrowingID)

ReturnID: INTEGER, -- Foreign Key referencing Return(ReturnID)

PRIMARY KEY (BorrowingID, ReturnID)

);

Incurs(

Member\_ID: INTEGER, -- Foreign Key referencing Member(Member\_ID)

FineID: INTEGER, -- Foreign Key referencing Fines(FineID)

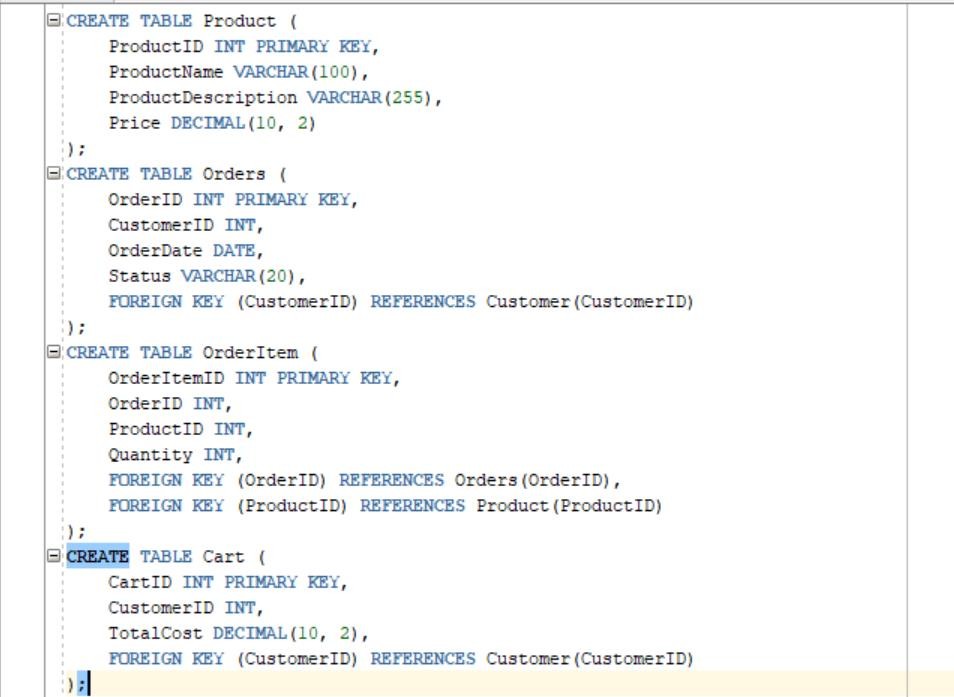
PRIMARY KEY (Member\_ID, FineID)

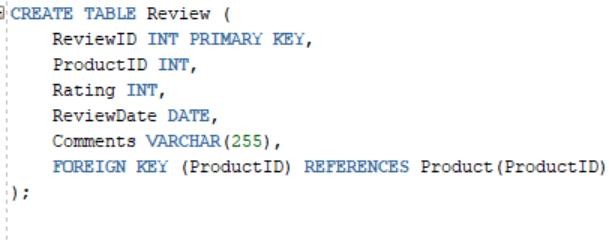
);

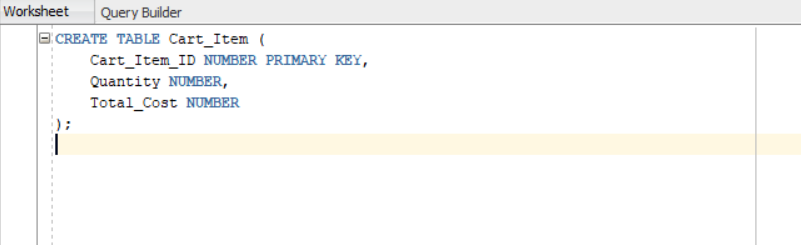
PART-2:

RELATIONAL MODELS: DDL COMMANDS

Creating Tables







# 

# 

# 

# 

# 

# 

# 

# 

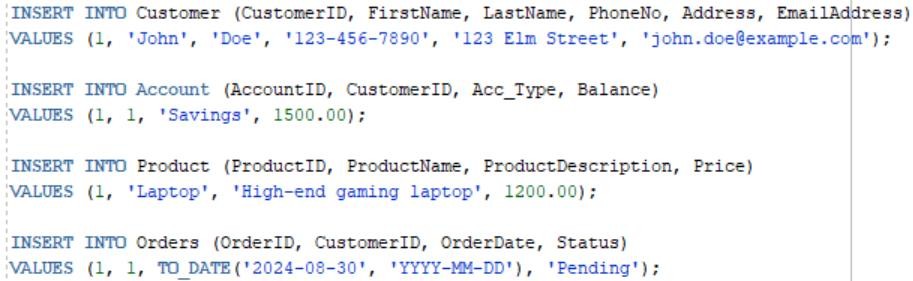
# 

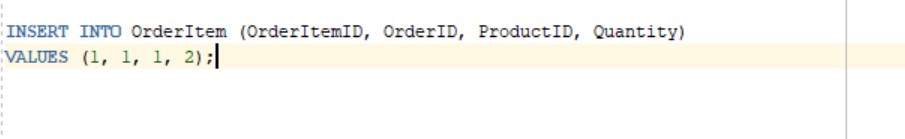
# 

# 

# DML COMMANDS:

Insertting data into Tables using INSERT INTO command





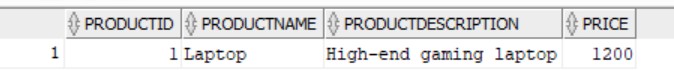


Display tables using SELECT command



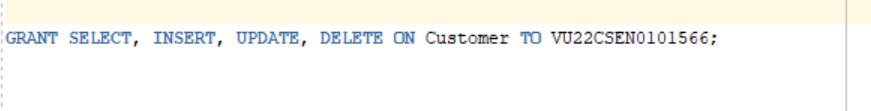






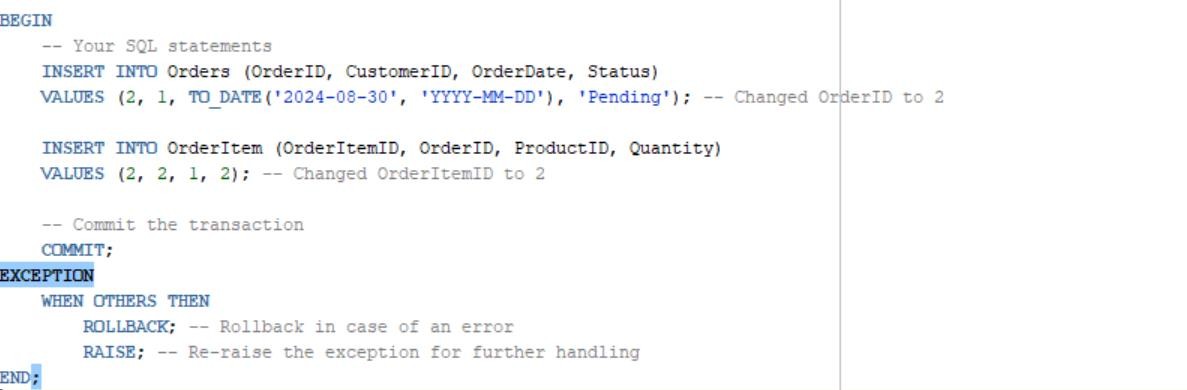
# DCL COMMANDS

GRANTING PERMISSIONS





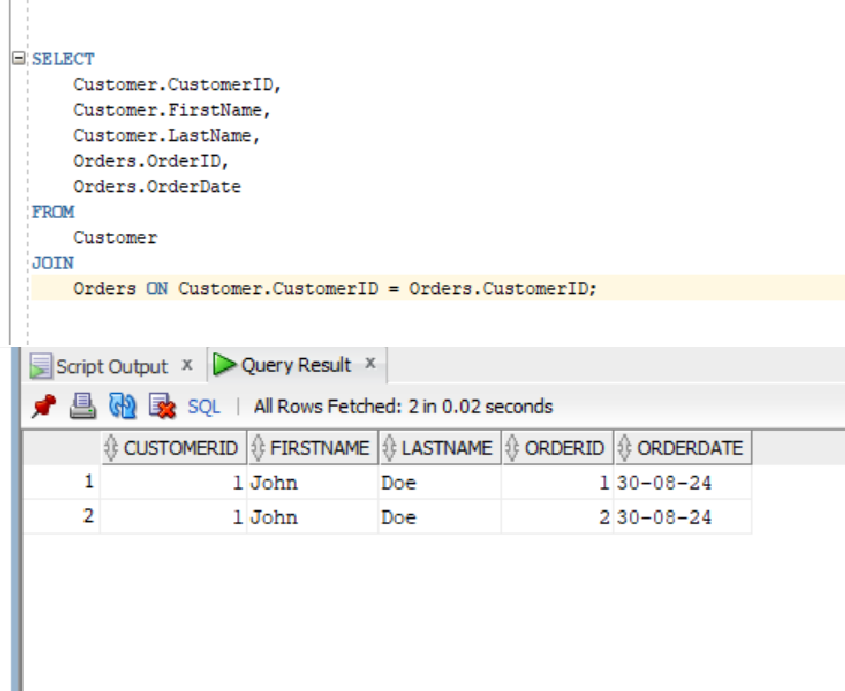
# TCL COMMANDS



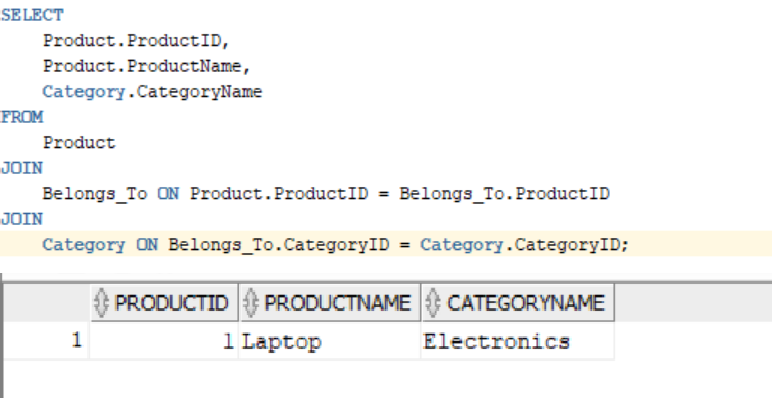


DQL COMMAND(SELECT):

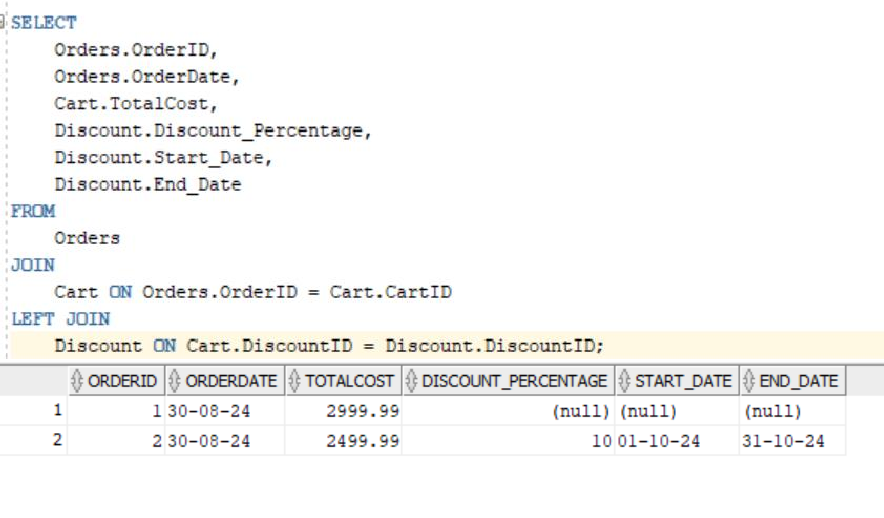
# Retrieve Customer Details and Their Orders

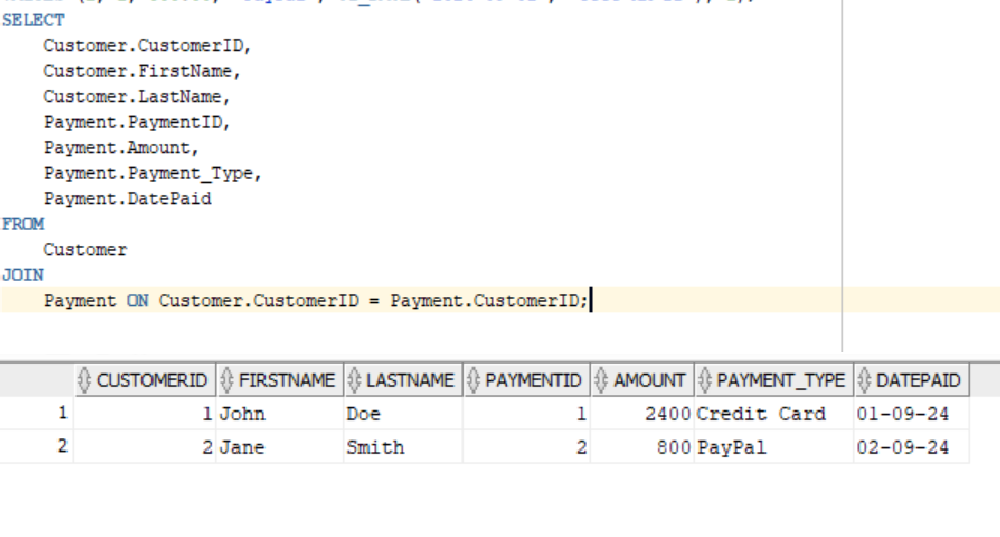


1. Retrieve Products with Their Categories

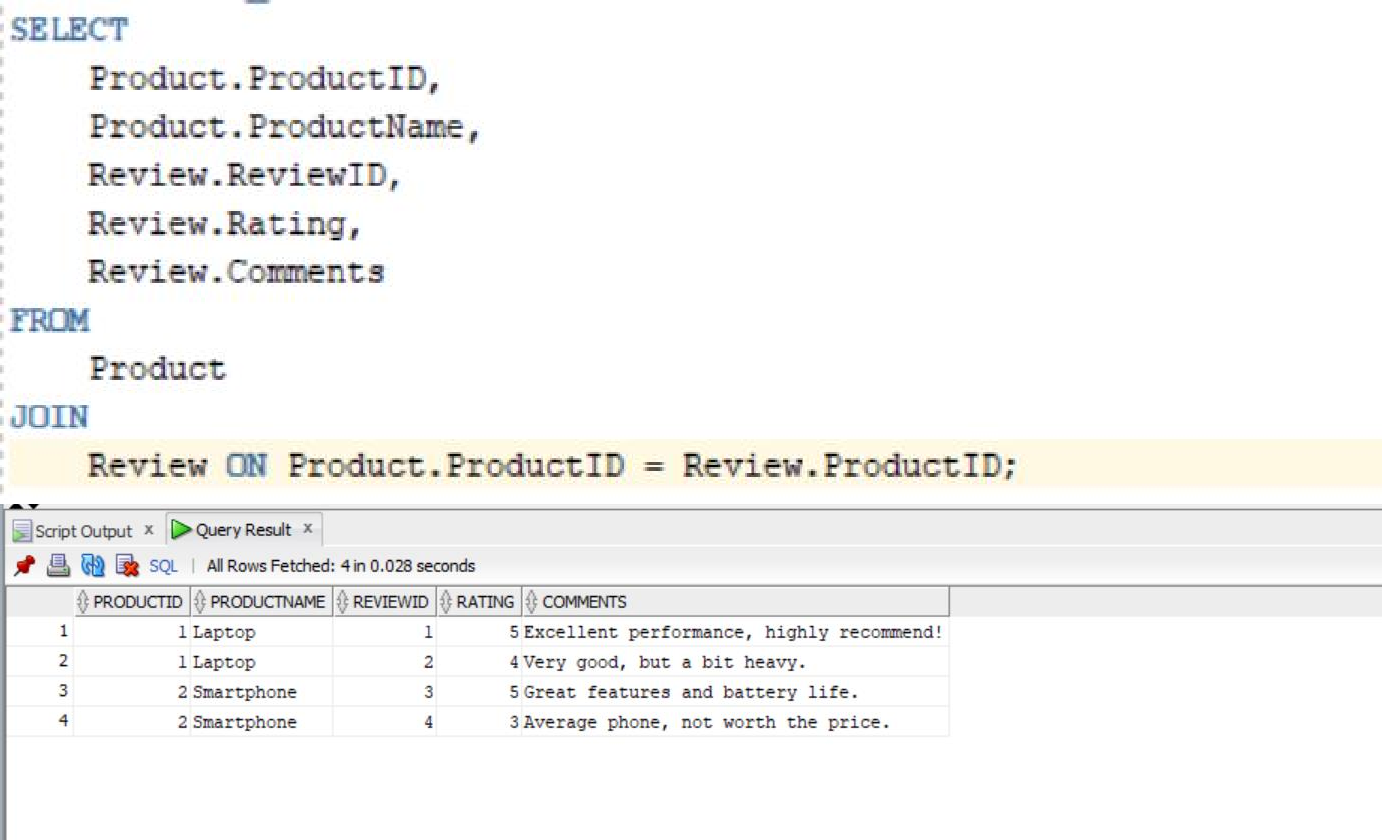


# Retrieve All Orders with Their Total Cost and Discount Details

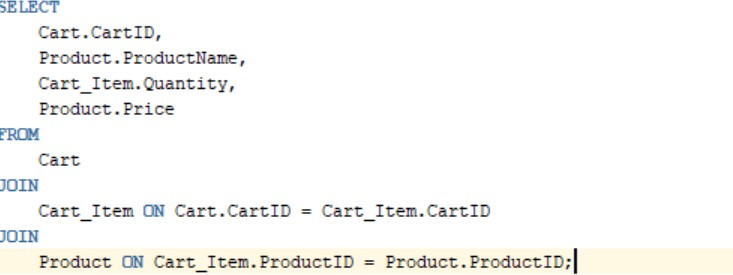


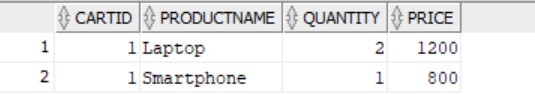
1. Retrieve Customer Payment Details

# Retrieve Products and Their Reviews

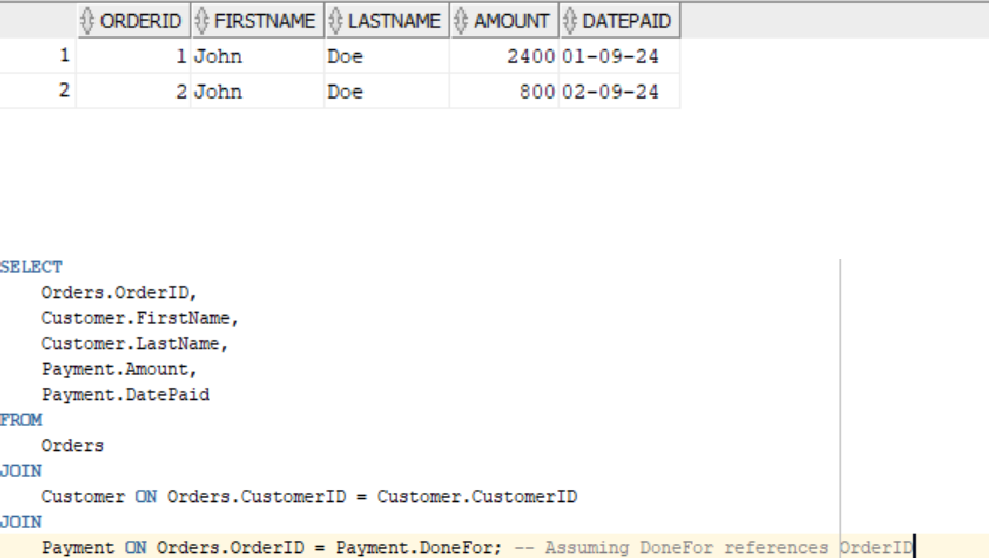


1. Retrieve Cart Details and Associated Products





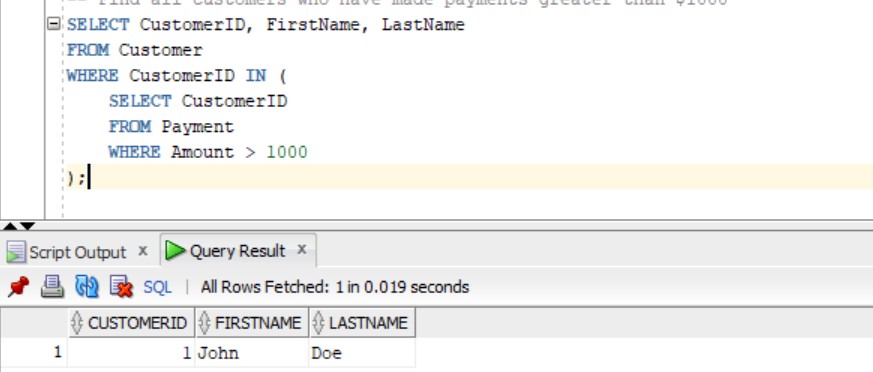
# Retrieve Orders with Customer and Payment Information



PART-3:

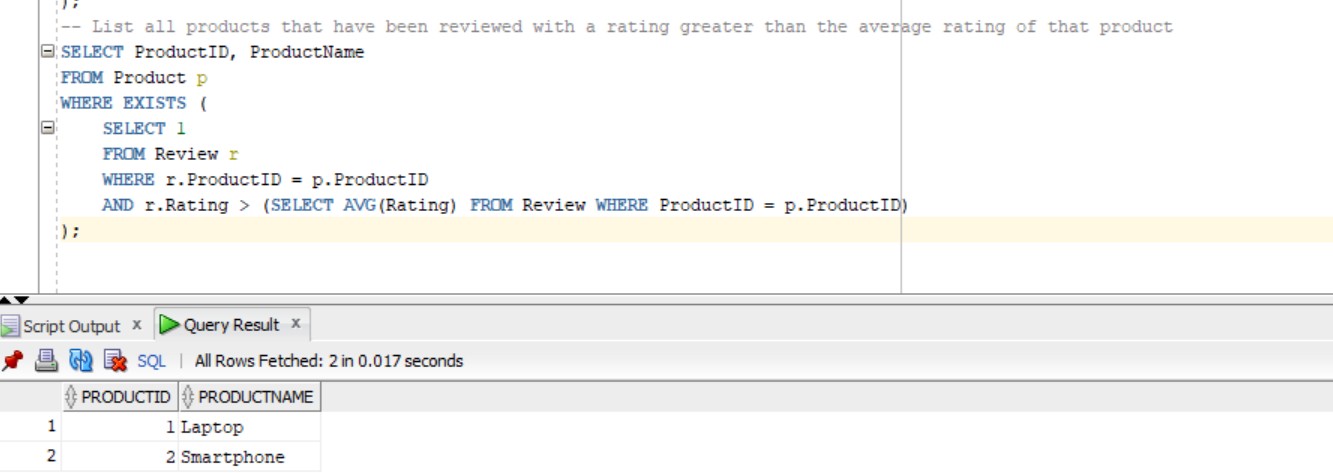
# NESTED QUERIES:

A nested query is a query within another query. Here's an example based on your schema:



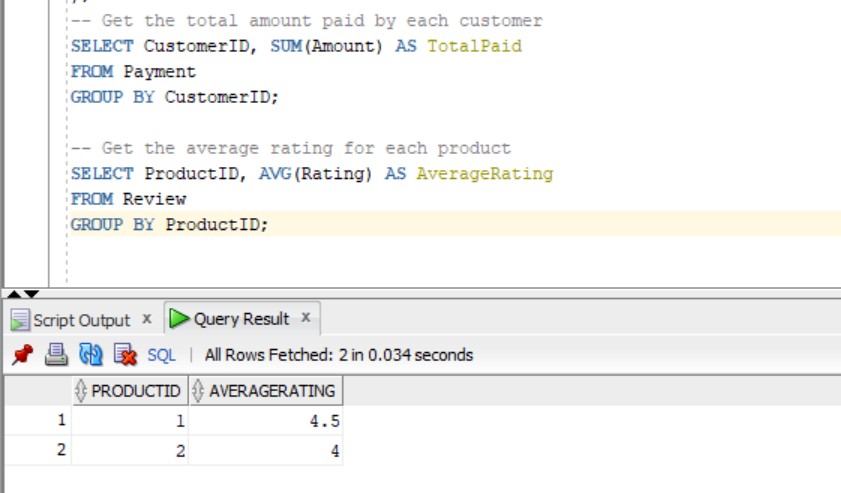
# CORRELATED QUERIES:

A correlated query refers to another query in its Where clause. Here's an example:



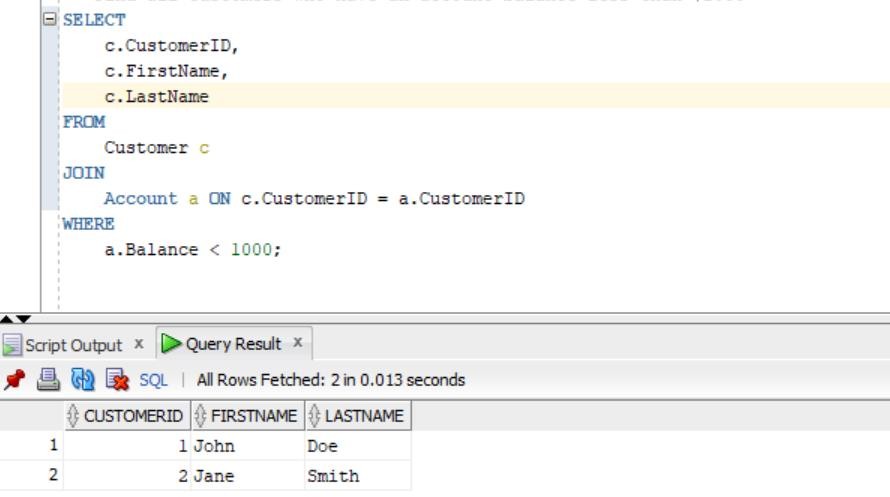
# AGGREGATE OPERATORS:

Aggregate operators perform calculations on a set of values. Here’s how you can use them:



# RELATIONAL OPERATORS:

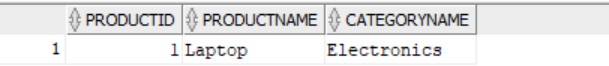
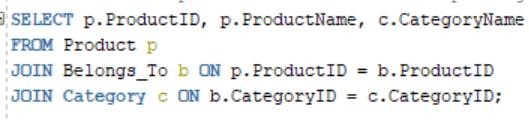
Relational operators can be used within the Where clause to compare values. Here’s an example:

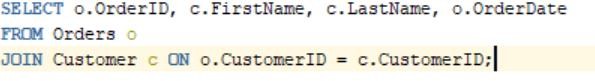


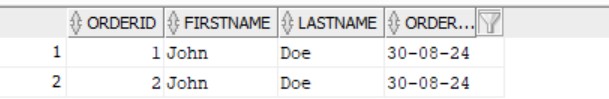
PART-4:

JOINS:

You can perform various types of joins to combine data from multiple tables. Here are a few examples:

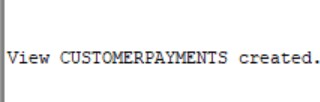
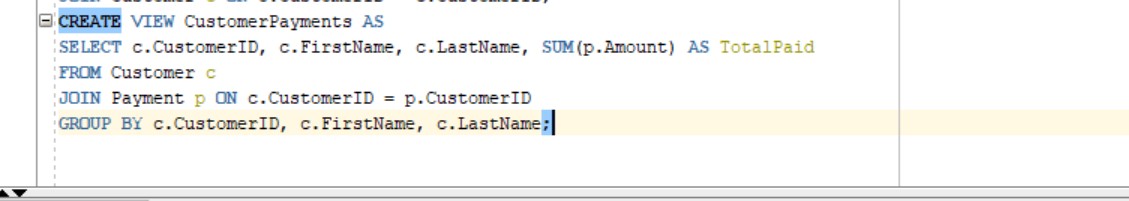






# VIEWS:

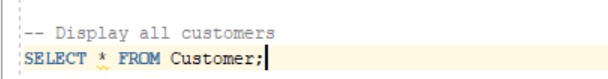
You can create a view to simplify complex queries. Here’s how to create a view for customer payments:

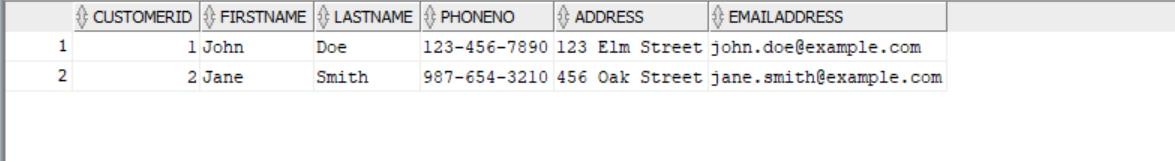


# DISPLAY TABLES:

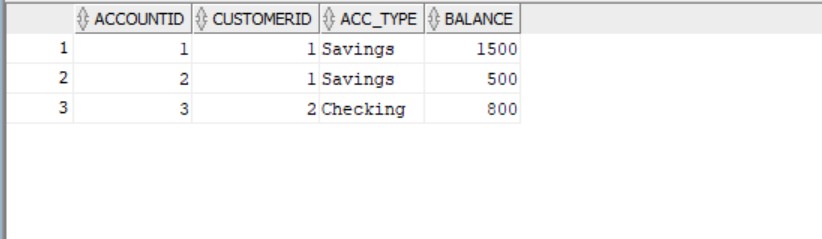
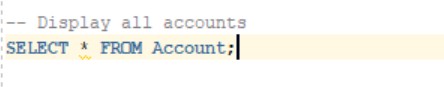
To display the contents of the tables you created, you can use the following SELECT statements:

Customer Table:

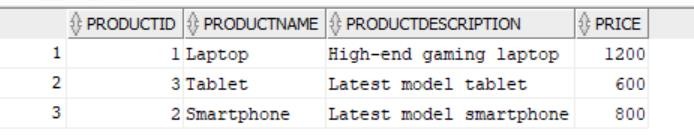
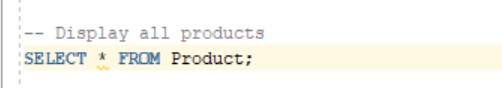




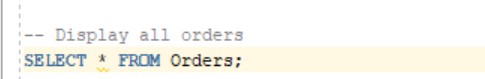
Account Table:

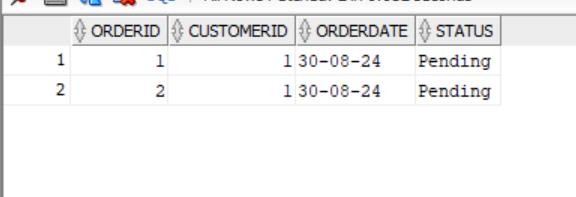


Product Table:

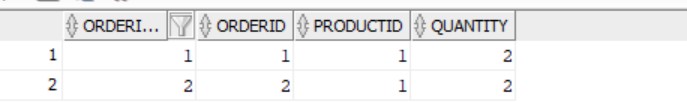


Orders Table:

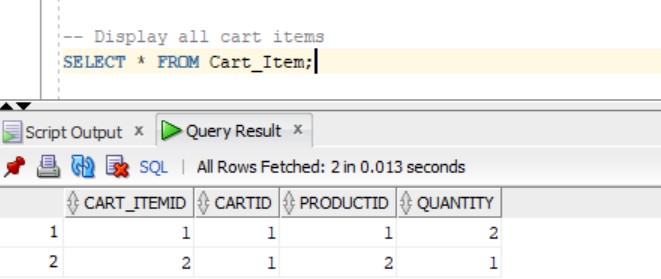




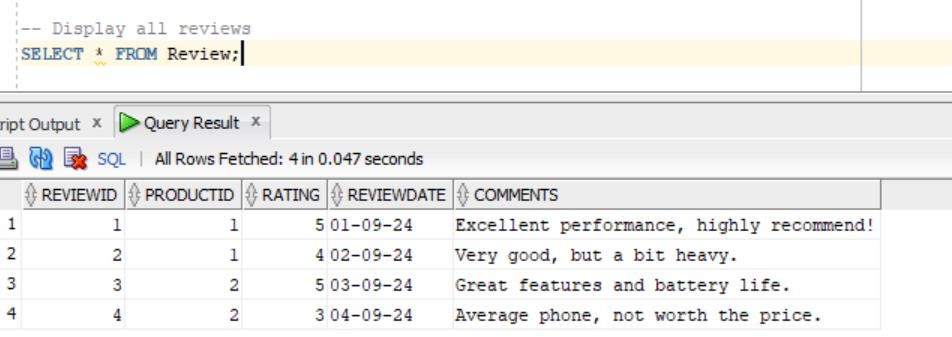
OrderItem Table:



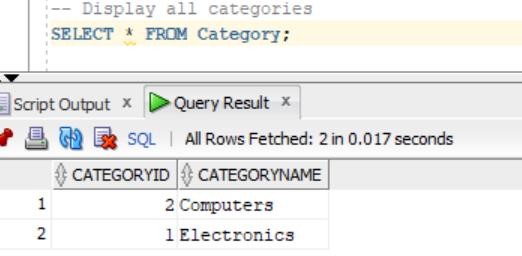
Cart\_Item Table:



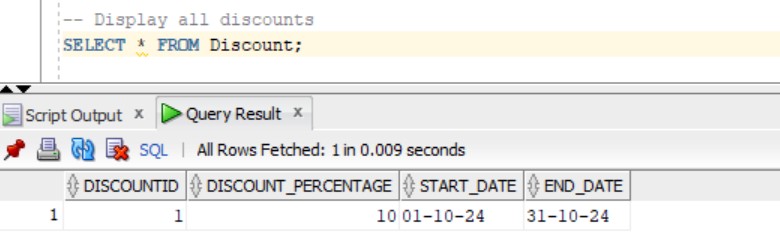
Review Table:



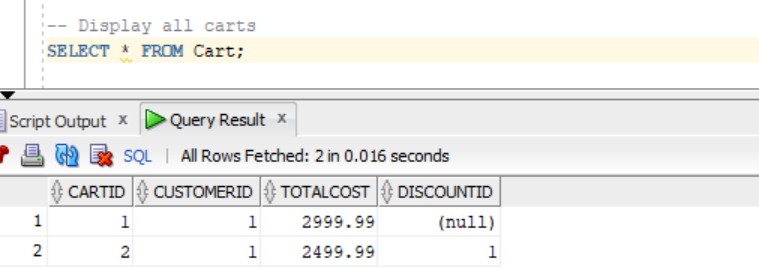
Category Table:

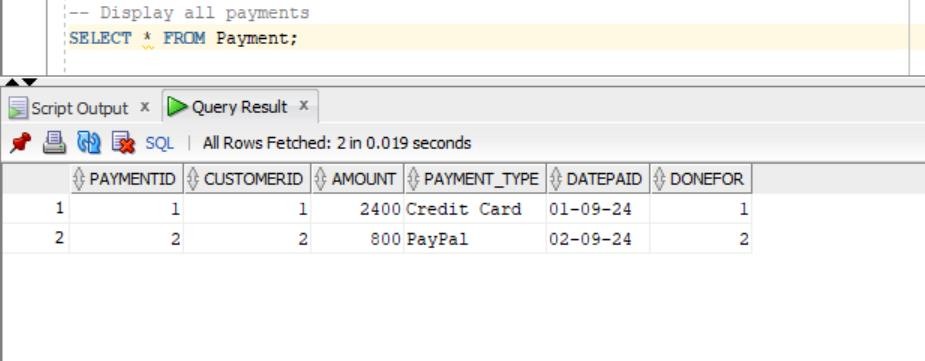


Discount Table:



Cart Table:



Payment Tab