**E-commerce System Final Documentation**

**Description:**

Welcome to our comprehensive e-commerce platform. Our project is the culmination of assignments focused on designing and implementing a robust database system tailored for an e-commerce environment. The primary goal of our project is to create an efficient database design for an e-commerce system. Throughout the assignments, we have undertaken a systematic approach to database design, beginning with a detailed analysis of the application's necessities. We delved into entity-relationship modeling, identifying entities, attributes, and relationships that form the spine of our database.

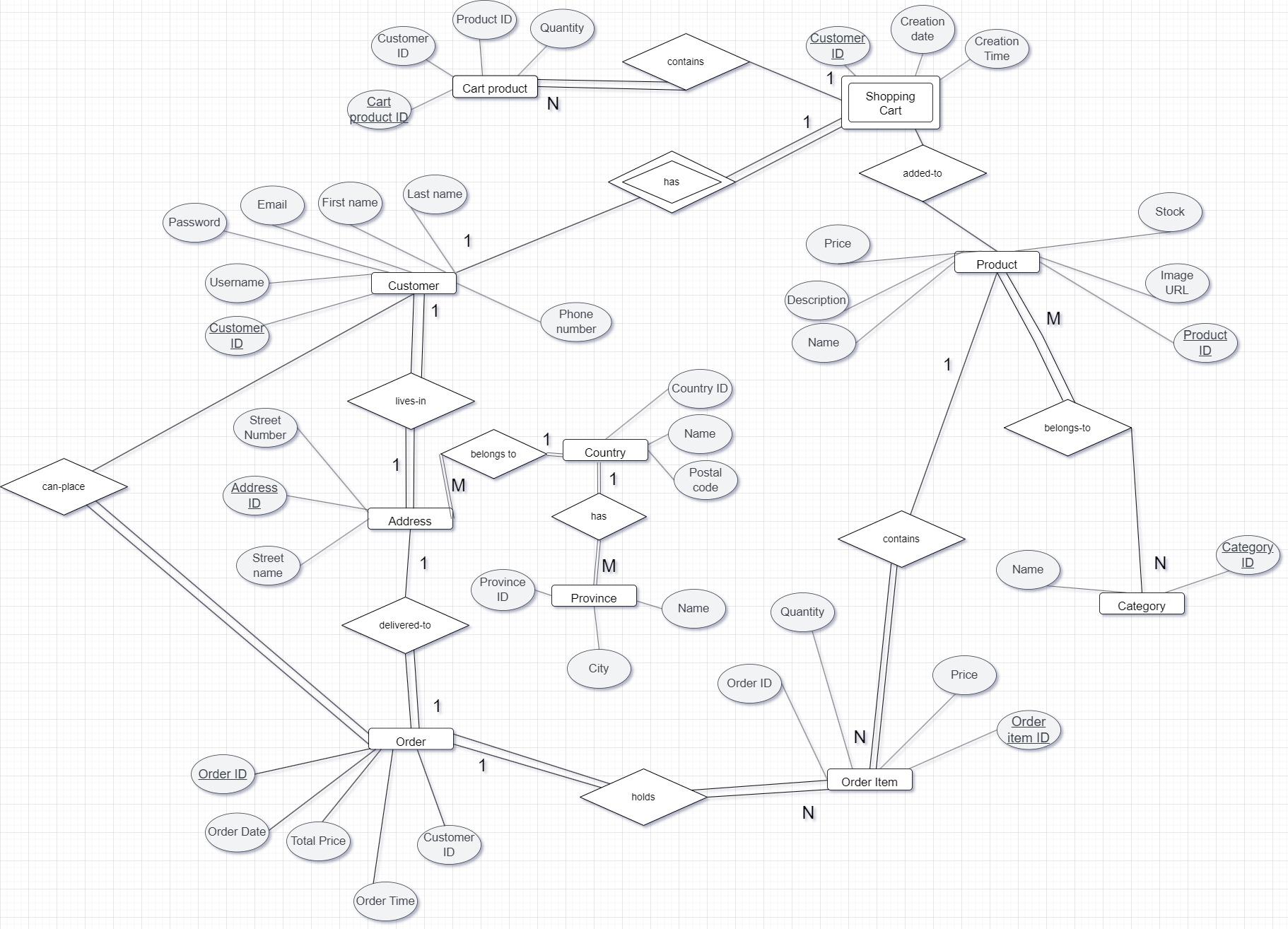
**In this document you will find:**

* Planning with ER Diagrams
* Source codes for the bash commands
* Source codes for our Bonus GUI design (in PHP)
* Functional Dependencies, 3NF and BCNF
* Relational Algebra

**To navigate**

* [Lab 01](#eh3qou1iklqc)
* [Lab 02](#86444g8hsw9s)
* [Lab 03](#198pq4kmq7dq)
* [Lab 04](#2po8aspsi9iw)
* [Lab 05](#119kk1wqhtpa)
* [Lab 06](#4db784ngf1nb)
* [Lab 07](#6zwhko4bezep)
* [Lab 08](#grfp0mrocjxn)
* [Lab 09](#dlut5gz3zozm)
* [RA](#ufuha5vz08m6)

Our final ER Diagram as show below:

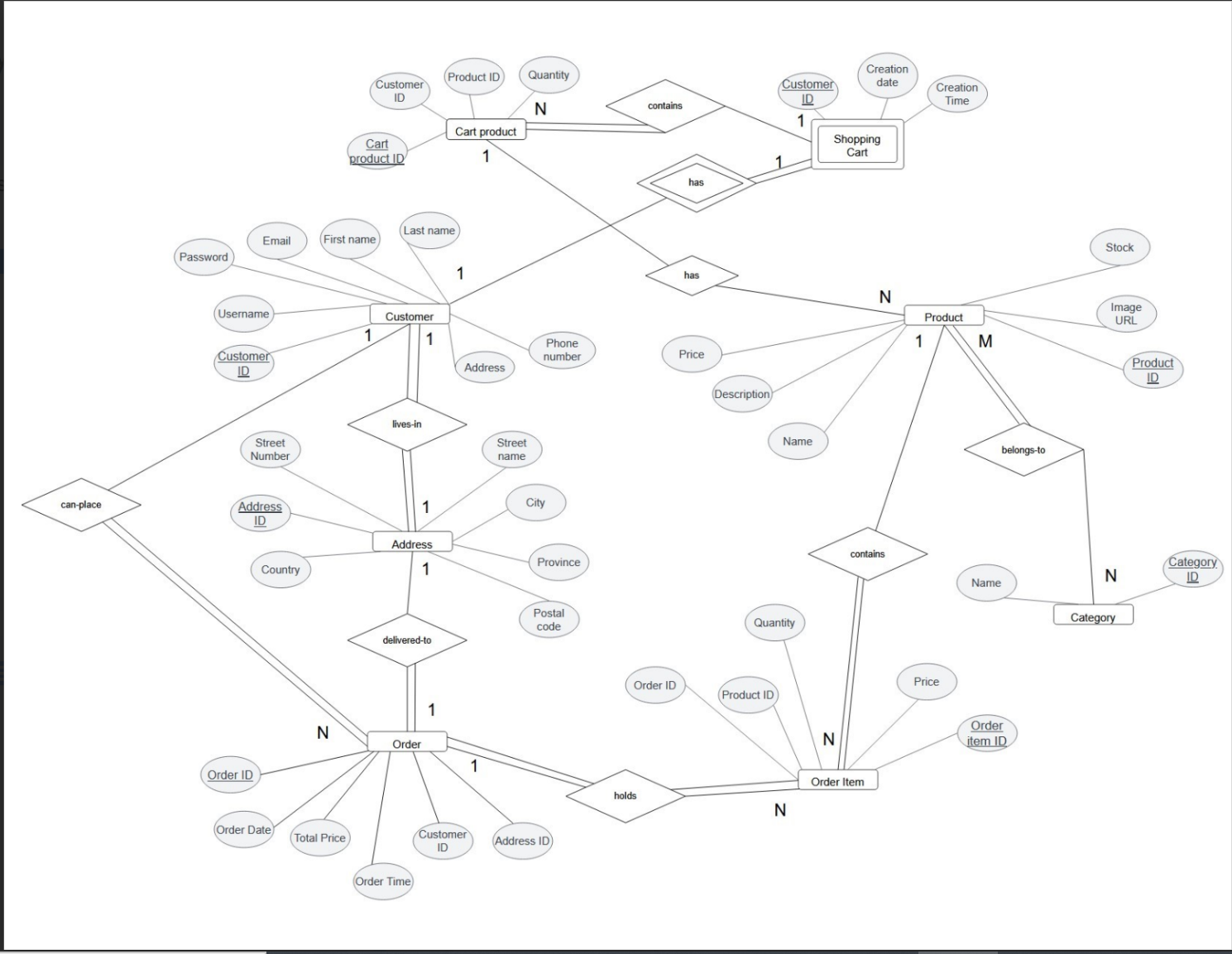
****

**Lab 01**

The first part of the lab was dedicated to forming groups. The goal was to foster an environment of shared responsibility, encouraging group members to exchange ideas and skills. Through lively discussions, we collectively identified key functionalities and entities crucial to the successful operation of an online shopping system. Varied perspectives were brought to the table, contributing rich insights into user interactions. We laid the groundwork for our database schema. Entities such as "User," "Product," "Order," and "Shopping Cart" emerged as pivotal components, each carrying a unique set of attributes that would shape the structure of our database.

**Lab02**

In the second lab, our focus transitioned from the conceptualization of database requirements to the actual design of an Entity-Relationship (ER) diagram. This pivotal phase served as the architectural blueprint for our e-commerce platform, translating abstract concepts into a visual representation.

****

**Lab03**

The third lab of our database catapulted us from the design domains into the tangible sphere of implementation. In Lab 3, our collective focus pivoted to the practical task of translating our crafted Entity-Relationship (ER) diagram into a fully functional SQL database for our e-commerce platform. This included table creation and data population.

Below is our Code:

-- CREATE THE TABLES:

-- Address table

CREATE TABLE Address (

AddressID *NUMBER*(5) PRIMARY KEY,

StreetNumber *VARCHAR2*(10),

StreetName *VARCHAR2*(100),

City *VARCHAR2*(50),

Province *VARCHAR2*(50),

Country *VARCHAR2*(50),

PostalCode *VARCHAR2*(10)

);

-- Customer table

CREATE TABLE Customer (

CustomerID *NUMBER*(5) PRIMARY KEY,

Username *VARCHAR2*(50) NOT NULL,

Passwd *VARCHAR2*(50) NOT NULL,

Email *VARCHAR2*(100),

FirstName *VARCHAR2*(50),

LastName *VARCHAR2*(50),

PhoneNumber *VARCHAR2*(20),

AddressID *NUMBER*(5) REFERENCES Address(AddressID)

);

-- Product table

CREATE TABLE Product (

ProductID *NUMBER*(5) PRIMARY KEY,

ProductName *VARCHAR2*(100) NOT NULL,

ProductDesc *VARCHAR2*(1000),

Price *NUMBER*(10, 2) NOT NULL,

Stock *NUMBER*(5) NOT NULL,

ImageURL *VARCHAR2*(255)

);

-- Category table

CREATE TABLE CategoryTable (

CategoryID *NUMBER*(5) PRIMARY KEY,

CategoryName *VARCHAR2*(100) NOT NULL

);

-- ProductCategory table

CREATE TABLE ProductCategory (

ProductCategoryID *NUMBER*(5) PRIMARY KEY,

ProductID *NUMBER*(5) REFERENCES Product(ProductID),

CategoryID *NUMBER*(5) REFERENCES CategoryTable(CategoryID)

);

-- Order table

CREATE TABLE OrderTable (

OrderID *NUMBER*(5) PRIMARY KEY,

CustomerID *NUMBER*(5) REFERENCES Customer(CustomerID),

AddressID *NUMBER*(5) REFERENCES Address(AddressID),

OrderDate *DATE* NOT NULL,

OrderTime *TIMESTAMP* NOT NULL,

TotalPrice *NUMBER*(10, 2) NOT NULL

);

-- OrderItem table

CREATE TABLE OrderItem (

OrderItemID *NUMBER*(5) PRIMARY KEY,

OrderID *NUMBER*(5) REFERENCES OrderTable(OrderID),

ProductID *NUMBER*(5) REFERENCES Product(ProductID),

Quantity *NUMBER*(5) NOT NULL,

Subtotal *NUMBER*(10, 2) NOT NULL

);

-- Create the ShoppingCart table

CREATE TABLE ShoppingCart (

CustomerID *NUMBER*(5) PRIMARY KEY,

CreationDate *DATE* NOT NULL,

CreationTime *TIMESTAMP* NOT NULL

);

-- CartProduct table

CREATE TABLE CartProduct (

CartProductID *NUMBER*(5) PRIMARY KEY,

CustomerID *NUMBER*(5) REFERENCES ShoppingCart(CustomerID),

ProductID *NUMBER*(5) REFERENCES Product(ProductID),

Quantity *NUMBER*(5) NOT NULL

);

-- POPULATE THE TABLES:

-- Insert data into the Address table

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (1, '123', 'Main St', 'Toronto', 'Ontario', 'CANADA', 'P7H0A8');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (2, '456', 'Oak Avenue', 'New York', 'New York', 'USA', '10001');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (3, '789', 'Cedar Lane', 'Los Angeles', 'California', 'USA', '90001');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (4, '1010', 'Maple Street', 'Chicago', 'Illinois', 'USA', '60601');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (5, '234', 'Pine Avenue', 'Houston', 'Texas', 'USA', '77001');

-- Insert data for Customer table

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (1, 'user1', 'password1', 'user1@example.com', 'John', 'Doe', '657-123-4567', 1);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (2, 'user2', 'password2', 'user2@example.com', 'Alice', 'Smith', '987-654-3210', 2);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (3, 'user3', 'password3', 'user3@example.com', 'Bob', 'Johnson', '555-123-4567', 3);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (4, 'user4', 'password4', 'user4@example.com', 'Emily', 'Brown', '444-789-0123', 4);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (5, 'user5', 'password5', 'user5@example.com', 'Grace', 'Lee', '111-222-3333', 5);

-- Insert data forProduct table

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (1, 'Product A', 'Description for A', 10.99 , 100, 'product\_a.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (2, 'Product B', 'Description for B', 19.99, 50, 'product\_b.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (3, 'Product C', 'Description for C', 7.49, 75, 'product\_c.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (4, 'Product D', 'Description for D', 24.95, 30, 'product\_d.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (5, 'Product E', 'Description for E', 15.99, 90, 'product\_e.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (6, 'Product F', 'Description for F', 12.50, 60, 'product\_f.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (7, 'Product G', 'Description for G', 29.99, 25, 'product\_g.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (8, 'Product H', 'Description for H', 8.99, 120, 'product\_h.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (9, 'Product I', 'Description for I', 14.75, 70, 'product\_i.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (10, 'Product J', 'Description for J', 17.49, 55, 'product\_j.jpg');

-- Insert data for the CategoryTable

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (1, 'Category 1');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (2, 'Category 2');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (3, 'Category 3');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (4, 'Category 4');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (5, 'Category 5');

-- Insert data ProductCategory table

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (1, 1, 1);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (2, 2, 1);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (3, 3, 2);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (4, 4, 2);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (5, 5, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (6, 6, 4);

-- Insert data for OrderTable

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (1, 1, 1, TO\_DATE('2023-09-25', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 10.99);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (2, 2, 2, TO\_DATE('2023-09-26', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 39.99);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (3, 3, 3, TO\_DATE('2023-09-27', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 17.49);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (4, 4, 4, TO\_DATE('2023-09-28', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 24.95);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (5, 5, 5, TO\_DATE('2023-09-29', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 35.99);

-- Insert data into the OrderItem table

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (1, 1, 1, 2, 21.98);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (2, 1, 2, 3, 59.97);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (3, 2, 3, 1, 7.49);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (4, 2, 4, 2, 49.90);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (5, 3, 5, 4, 63.96);

-- Insert data into the ShoppingCart table

INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (1, TO\_DATE('2023-09-25', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP);

INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (2, TO\_DATE('2023-09-26', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP);

INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (3, TO\_DATE('2023-09-27', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP);

-- Insert data into the CartProduct table

INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (1, 1, 1, 3);

INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (2, 1, 2, 2);

INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (3, 2, 3, 1);

-- MAKE SIMPLE QUERIES:

-- Get customer information for a specific customer using their CustomerID:

SELECT \* FROM Customer WHERE CustomerID = 1;

-- Get a list of all products:

SELECT \* FROM Product;

-- Get a list of categories:

SELECT \* FROM CategoryTable;

-- Get product details for a specific product by its ProductID:

SELECT \* FROM Product WHERE ProductID = 1;

-- Get orders placed by a specific customer:

SELECT \* FROM OrderTable WHERE CustomerID = 1;

-- Get order items for a specific order by its OrderID:

SELECT \* FROM OrderItem WHERE OrderID = 1;

-- Retrieve the products in a shopping cart for a specific customer:

SELECT \* FROM CartProduct WHERE CustomerID = 1;

-- Get total price of a specific order by its OrderID:

SELECT TotalPrice FROM OrderTable WHERE OrderID = 1;

-- Get address details for a specific address by its AddressID:

SELECT \* FROM Address WHERE AddressID = 1;

-- Drop the tables

-- Order Does not matter for these tables as they do not have foreign key constraints

DROP TABLE CartProduct;

DROP TABLE ProductCategory;

DROP TABLE ShoppingCart;

DROP TABLE OrderItem;

-- Later Drop these in Order

DROP TABLE OrderTable;

DROP TABLE CategoryTable;

DROP TABLE Product;

DROP TABLE Customer;

DROP TABLE Address;

**Lab04**

Lab 4 emerged as a pivotal moment, producing some simple and advanced queries that delved into the heart of our SQL database for the e-commerce platform. Part 1 just creates simple queries, and part 2 unleashes the power of advanced queries with views. Below is the code provided.

**Part 1 Simple Queries:**

-- Simple Queries for A4

-- Description: List all attributes of all products

SELECT \* FROM Product;

-- Description: List Product Name and their Price for all products

SELECT 'The Product: ', ProductName, ' costs ', Price

FROM Product;

-- Description: List the Unique First Name of all the customers

SELECT DISTINCT FirstName FROM Customer;

-- Description: List all attributes of products that costs more than $100.0 or

-- have more than 200 items in stock.

SELECT \* FROM Product

WHERE Price > 100.0 OR Stock > 200;

-- Description: List all the Subtotal of each product purchased by Customer1.

SELECT Subtotal AS Purchases\_by\_Customer1 FROM OrderItem

WHERE OrderID = 1;

-- Description: List the Order ID and Total Price of all orders sorted by Order

-- Date in ascending order (Earliest to Latest).

SELECT OrderID, TotalPrice FROM OrderTable

ORDER BY OrderDate;

-- Description: Count the number of customers living in each City from all the

-- addresses.

SELECT COUNT(AddressID) AS Number\_Of\_Customers, City FROM Address

GROUP BY City;

**Part 2 Advance Queries with Views:**

-- List the username and password of customer with an id greater than 3

SELECT username, passwd

FROM customer

WHERE customerid>3;

-- List the names of products that are popular (customers got more than one of)

SELECT productname

FROM product, cartproduct

Where quantity > 1

AND cartproduct.productid=product.productid;

-- List the customer id of customers that have spent less than $25 on their order

SELECT 'Customer ID is: ', customerid

FROM ordertable

WHERE totalprice < 25;

-- List the order IDs for orders that spent more than $50

SELECT orderid

FROM orderitem

WHERE subtotal>50;

-- List information of customers with an active shopping cart

SELECT c.\*

FROM customer c, shoppingcart s

WHERE c.customerid=s.customerid;

-- List the address id of customers that live in Toronto

SELECT addressid

FROM address

WHERE city='Toronto';

-- List all product ID's that are within the Clothing category

SELECT productid

FROM productcategory

WHERE categoryid='1';

-- List the category ID for the food category

SELECT categoryid

FROM categorytable

WHERE categoryname="food";

-- Count the number of products in each category.

SELECT CategoryName, COUNT(ProductCategoryID) AS NumberOfProducts

FROM CategoryTable

LEFT JOIN ProductCategory ON CategoryTable.CategoryID = ProductCategory.CategoryID

GROUP BY CategoryName;

-- List the total number of products in each shopping cart.

SELECT CustomerID, COUNT(ProductID) AS TotalProductsInCart

FROM ShoppingCart

LEFT JOIN CartProduct ON ShoppingCart.CustomerID = CartProduct.CustomerID

GROUP BY CustomerID;

-- List the products along with their quantities in the shopping cart of a specific customer.

-- In this query, we used tablesm aliases cp and p for a more advanced looking query

SELECT cp.CustomerID, p.ProductName, cp.Quantity

FROM CartProduct cp

JOIN Product p ON cp.ProductID = p.ProductID

WHERE cp.CustomerID = 1;

-- List customers who made orders and their respective cities:

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, a.City

FROM Customer c

JOIN Address a ON c.AddressID = a.AddressID

WHERE c.CustomerID IN (SELECT DISTINCT CustomerID FROM OrderTable);

-- List products and their categories:

SELECT p.ProductName, ct.CategoryName

FROM Product p

JOIN ProductCategory pc ON p.ProductID = pc.ProductID

JOIN CategoryTable ct ON pc.CategoryID = ct.CategoryID;

-- List customers and their total spending:

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

LEFT JOIN OrderTable o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName, c.LastName;

-- List customers who have both placed orders and added products to their shopping cart:

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName

FROM Customer c

WHERE EXISTS (SELECT 1 FROM OrderTable o WHERE o.CustomerID = c.CustomerID)

AND EXISTS (SELECT 1 FROM CartProduct cp WHERE cp.CustomerID = c.CustomerID);

--This query joins the OrderTable, Customer, and Address tables and orders the results by the order date in descending order.

SELECT o.OrderID, o.OrderDate, o.TotalPrice, c.FirstName || ' ' || c.LastName AS CustomerName, a.City, a.Province, a.Country

FROM OrderTable o

JOIN Customer c ON o.CustomerID = c.CustomerID

JOIN Address a ON o.AddressID = a.AddressID

ORDER BY o.OrderDate DESC;

-- VIEWS:

-- This view combines information from the Customer, OrderTable, and OrderItem tables to provide details about customer orders.

CREATE VIEW CustomerOrderDetails AS

SELECT o.OrderID, o.OrderDate, o.TotalPrice, c.FirstName || ' ' || c.LastName AS CustomerName, p.ProductName, oi.Quantity, oi.Subtotal

FROM OrderTable o

JOIN Customer c ON o.CustomerID = c.CustomerID

JOIN OrderItem oi ON o.OrderID = oi.OrderID

JOIN Product p ON oi.ProductID = p.ProductID;

-- This view shows products that are popular (ordered more than once) along with the total quantity ordered.

CREATE VIEW PopularProducts AS

SELECT p.ProductID, p.ProductName, SUM(oi.Quantity) AS TotalOrdered

FROM Product p

JOIN OrderItem oi ON p.ProductID = oi.ProductID

GROUP BY p.ProductID, p.ProductName

HAVING SUM(oi.Quantity) > 1;

-- This view counts the number of customers in each city.

CREATE VIEW CustomerCityCount AS

SELECT a.City, COUNT(c.CustomerID) AS CustomerCount

FROM Customer c

JOIN Address a ON c.AddressID = a.AddressID

GROUP BY a.City;

-- This view lists customers with active shopping carts.

CREATE VIEW ActiveShoppingCarts AS

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, s.CreationDate, s.CreationTime

FROM Customer c

JOIN ShoppingCart s ON c.CustomerID = s.CustomerID;

-- This view identifies high-value customers based on their total spending.

CREATE VIEW HighValueCustomers AS

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

LEFT JOIN OrderTable o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName, c.LastName

HAVING SUM(o.TotalPrice) > 100;

-- This view identifies product categories with high total sales based on the quantity sold.

CREATE VIEW HighValueProductCategories AS

SELECT pc.CategoryID, ct.CategoryName, SUM(oi.Quantity) AS TotalQuantitySold

FROM ProductCategory pc

JOIN OrderItem oi ON pc.ProductID = oi.ProductID

JOIN CategoryTable ct ON pc.CategoryID = ct.CategoryID

GROUP BY pc.CategoryID, ct.CategoryName

HAVING SUM(oi.Quantity) > 50;

**Lab05**

Lab 5 steered us into command-line finesse as we journeyed through the dynamic and powerful domain of Bash scripting for SQL command execution. This session helps Bash's versatility with the precision of SQL. We were given the sh files (6 of them), and all we had to do was paste our SQL command on it. Below the codes are provided.

Create Tables SH file:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.cs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

-- Address table

CREATE TABLE Address (

AddressID NUMBER(5) PRIMARY KEY,

StreetNumber VARCHAR2(10),

StreetName VARCHAR2(100),

City VARCHAR2(50),

Province VARCHAR2(50),

Country VARCHAR2(50),

PostalCode VARCHAR2(10)

);

-- Customer table

CREATE TABLE Customer (

CustomerID NUMBER(5) PRIMARY KEY,

Username VARCHAR2(50) NOT NULL,

Passwd VARCHAR2(50) NOT NULL,

Email VARCHAR2(100),

FirstName VARCHAR2(50),

LastName VARCHAR2(50),

PhoneNumber VARCHAR2(20),

AddressID NUMBER(5) REFERENCES Address(AddressID)

);

-- Product table

CREATE TABLE Product (

ProductID NUMBER(5) PRIMARY KEY,

ProductName VARCHAR2(100) NOT NULL,

ProductDesc VARCHAR2(1000),

Price NUMBER(10, 2) NOT NULL,

Stock NUMBER(5) NOT NULL,

ImageURL VARCHAR2(255)

);

-- Category table

CREATE TABLE CategoryTable (

CategoryID NUMBER(5) PRIMARY KEY,

CategoryName VARCHAR2(100) NOT NULL

);

-- ProductCategory table

CREATE TABLE ProductCategory (

ProductCategoryID NUMBER(5) PRIMARY KEY,

ProductID NUMBER(5) REFERENCES Product(ProductID),

CategoryID NUMBER(5) REFERENCES CategoryTable(CategoryID)

);

-- Order table

CREATE TABLE OrderTable (

OrderID NUMBER(5) PRIMARY KEY,

CustomerID NUMBER(5) REFERENCES Customer(CustomerID),

AddressID NUMBER(5) REFERENCES Address(AddressID),

OrderDate DATE NOT NULL,

OrderTime TIMESTAMP NOT NULL,

TotalPrice NUMBER(10, 2) NOT NULL

);

-- OrderItem table

CREATE TABLE OrderItem (

OrderItemID NUMBER(5) PRIMARY KEY,

OrderID NUMBER(5) REFERENCES OrderTable(OrderID),

ProductID NUMBER(5) REFERENCES Product(ProductID),

Quantity NUMBER(5) NOT NULL,

Subtotal NUMBER(10, 2) NOT NULL

);

-- Create the ShoppingCart table

CREATE TABLE ShoppingCart (

CustomerID NUMBER(5) PRIMARY KEY,

CreationDate DATE NOT NULL,

CreationTime TIMESTAMP NOT NULL

);

-- CartProduct table

CREATE TABLE CartProduct (

CartProductID NUMBER(5) PRIMARY KEY,

CustomerID NUMBER(5) REFERENCES ShoppingCart(CustomerID),

ProductID NUMBER(5) REFERENCES Product(ProductID),

Quantity NUMBER(5) NOT NULL

);

exit;

EOF

Drop Table SH File:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

DROP TABLE CartProduct;

DROP TABLE ProductCategory;

DROP TABLE ShoppingCart;

DROP TABLE OrderItem;

DROP TABLE OrderTable;

DROP TABLE CategoryTable;

DROP TABLE Product;

DROP TABLE Customer;

DROP TABLE Address;

exit;

EOF

Menu SH File:

#!/bin/sh

StartMessage() {

echo "Starting the program..."

}

MainMenu()

{

while [ "$CHOICE" != "START" ]

do

#clear

echo "================================================================="

echo "| Oracle All Inclusive Tool

|"

echo "| Main Menu - Select Desired Operation(s):

|"

echo "| <CTRL-Z Anytime to Enter Interactive CMD Prompt>

|"

echo "-------------------------------------------------------------

----"

echo " $IS\_SELECTEDM M) View Manual"

echo " "

echo " $IS\_SELECTED1 1) Drop Tables"

echo " $IS\_SELECTED2 2) Create Tables"

echo " $IS\_SELECTED3 3) Populate Tables"

echo " $IS\_SELECTED4 4) Query Tables"

echo " $IS\_SELECTED4 5) View Tables"

echo " "

echo " $IS\_SELECTEDX X) Force/Stop/Kill Oracle DB"

echo " "

echo " $IS\_SELECTEDE E) End/Exit"

echo "Choose: "

read CHOICE

if [ "$CHOICE" = "0" ]

then

echo "Nothing Here"

elif [ "$CHOICE" = "1" ]

then

bash drop\_tables.sh

elif [ "$CHOICE" = "2" ]

then

bash create\_tables.sh

elif [ "$CHOICE" = "3" ]

then

bash populate\_tables.sh

elif [ "$CHOICE" = "4" ]

then

bash queries.sh

elif [ "$CHOICE" = "5" ]

then

bash view\_tables.sh

elif [ "$CHOICE" = "E" ]

then

exit

fi

done

}

#--COMMENTS BLOCK--

# Main Program

#--COMMENTS BLOCK--

ProgramStart()

{

StartMessage

while [ 1 ]

do

MainMenu

done

}

ProgramStart

Populate Table SH File:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.cs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

-- Insert data into the Address table

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (1, '123', 'Main St', 'Toronto', 'Ontario', 'CANADA', 'P7H0A8');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (2, '456', 'Oak Avenue', 'New York', 'New York', 'USA', '10001');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (3, '789', 'Cedar Lane', 'Los Angeles', 'California', 'USA', '90001');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (4, '1010', 'Maple Street', 'Chicago', 'Illinois', 'USA', '60601');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (5, '234', 'Pine Avenue', 'Houston', 'Texas', 'USA', '77001');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (6, '134', 'Brooke Road', 'Toronto', 'Ontario', 'Canada', 'M1Z001');

INSERT INTO Address (AddressID, StreetNumber, StreetName, City, Province, Country, PostalCode)

VALUES (7, '134', 'Brooklyn Street', 'New York', 'New York', 'USA', '90101');

-- Insert data for Customer table

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (1, 'user1', 'password1', 'user1@example.com', 'John', 'Doe', '657-123-4567', 1);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (2, 'user2', 'password2', 'user2@example.com', 'Alice', 'Smith', '987-654-3210', 2);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (3, 'user3', 'password3', 'user3@example.com', 'Bob', 'Johnson', '555-123-4567', 3);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (4, 'user4', 'password4', 'user4@example.com', 'Emily', 'Brown', '444-789-0123', 4);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (5, 'user5', 'password5', 'user5@example.com', 'Grace', 'Lee', '111-222-3333', 5);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (6, 'user6', 'password6', 'user6@example.com', 'Bob', 'Marley', '111-444-3333', 6);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (7, 'user7', 'password7', 'user7@example.com', 'Bob', 'Dylan', '000-434-2333', 7);

-- Insert data forProduct table

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (1, 'Men''s Cotton Shirt', 'High-quality cotton shirt for men', 29.99, 150, 'shirt.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (2, 'Women''s Leather Handbag', 'Stylish leather handbag for women', 79.99, 80, 'handbag.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (3, 'Organic All-Purpose Flour', 'Certified organic wheat flour', 5.99, 200, 'flour.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (4, 'Coca-Cola Classic', 'Classic carbonated soft drink', 1.99, 500, 'cocacola.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (5, 'Apple iPhone 13', 'Latest model with advanced features', 999.99, 30, 'iphone.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (6, 'Samsung 55" 4K Smart TV', 'Crystal-clear 4K Ultra HD television', 799.99, 10, 'tv.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (7, 'Bluetooth Wireless Earbuds', 'Wireless earbuds with long battery life', 49.99, 100, 'earbuds.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (8, 'Dell Inspiron Laptop', 'Powerful laptop for work and entertainment', 899.99, 40, 'laptop.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (9, 'Fitness Tracker Watch', 'Track your health and fitness activities', 29.99, 120, 'fitness.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (10, 'Chocolate Chip Cookies', 'Delicious homemade chocolate chip cookies', 3.99, 300, 'cookies.jpg');

-- Insert data for the CategoryTable

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (1, 'Clothing');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (2, 'Fashion Accessories');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (3, 'Electronics');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (4, 'Food');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (5, 'Fitness Equipment');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (6, 'Beverages');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (7, 'Home Appliances');

-- Insert data ProductCategory table

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (1, 1, 1);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (2, 1, 2);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (3, 2, 1);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (4, 3, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (5, 4, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (6, 4, 6);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (7, 5, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (8, 5, 2);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (9, 6, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (10, 6, 7);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (11, 7, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (12, 8, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (13, 9, 2);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (14, 9, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (15, 10, 4);

-- Insert data for OrderTable

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (1, 1, 1, TO\_DATE('2022-09-25', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 10.99);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (2, 2, 2, TO\_DATE('2020-09-16', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 39.99);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (3, 3, 3, TO\_DATE('2023-01-07', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 17.49);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (4, 4, 4, TO\_DATE('2023-10-18', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 24.95);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (5, 5, 5, TO\_DATE('2023-02-27', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 35.99);

-- Insert data into the OrderItem table

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (1, 1, 2, 2, 21.98);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (2, 1, 5, 3, 59.97);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (3, 2, 3, 1, 7.49);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (4, 2, 4, 2, 49.90);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (5, 3, 5, 4, 63.96);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (6, 4, 8, 1, 33.96);

-- Insert data into the ShoppingCart table

INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (1, TO\_DATE('2020-09-25', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP);

INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (2, TO\_DATE('2023-09-26', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP);

INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (3, TO\_DATE('2023-09-27', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP);

-- Insert data into the CartProduct table

INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (1, 1, 1, 3);

INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (2, 1, 2, 2);

INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (3, 2, 3, 1);

exit;

EOF

Queries SH File:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

-- List the Unique First Name of all the customers

SELECT DISTINCT FirstName FROM Customer;

-- List the Order ID and Total Price of all orders sorted by Order

-- Date in ascending order (Earliest to Latest).

SELECT OrderID, TotalPrice FROM OrderTable

ORDER BY OrderDate;

-- List the names of products that are popular (customers got more than one of)

SELECT productname

FROM product, cartproduct

Where quantity > 1

AND cartproduct.productid=product.productid;

-- List all product ID's that are within the Clothing category

SELECT productid

FROM productcategory

WHERE categoryid='1';

-- Count the number of products in each category.

SELECT CategoryName, COUNT(ProductCategoryID) AS NumberOfProducts

FROM CategoryTable

LEFT JOIN ProductCategory ON CategoryTable.CategoryID = ProductCategory.CategoryID

GROUP BY CategoryName;

-- List all customer id and name who have placed an order and or have an

-- active shopping cart

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName,

'Order' AS ActivityType

FROM Customer c

WHERE EXISTS (SELECT 1 FROM OrderTable o WHERE o.CustomerID = c.CustomerID)

UNION

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, 'Shopping Cart' AS ActivityType

FROM Customer c

WHERE EXISTS (SELECT 1 FROM CartProduct cp WHERE cp.CustomerID = c.CustomerID);

-- List all customers who placed an order but do not have products in their cart.

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName

FROM Customer c

WHERE EXISTS (SELECT 1 FROM OrderTable o WHERE o.CustomerID = c.CustomerID)

MINUS

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName

FROM Customer c

WHERE EXISTS (SELECT 1 FROM CartProduct cp WHERE cp.CustomerID = c.CustomerID);

-- List the products along with their quantities in the shopping cart of a specific customer.

-- In this query, we used tablesm aliases cp and p for a more advanced looking query

SELECT cp.CustomerID, p.ProductName, cp.Quantity

FROM CartProduct cp

JOIN Product p ON cp.ProductID = p.ProductID

WHERE cp.CustomerID = 1;

-- List customers who made orders and their respective cities:

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, a.City

FROM Customer c

JOIN Address a ON c.AddressID = a.AddressID

WHERE c.CustomerID IN (SELECT DISTINCT CustomerID FROM OrderTable);

-- List products and their categories:

SELECT p.ProductName, ct.CategoryName

FROM Product p

JOIN ProductCategory pc ON p.ProductID = pc.ProductID

JOIN CategoryTable ct ON pc.CategoryID = ct.CategoryID;

-- List customers and their total spending:

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

LEFT JOIN OrderTable o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName, c.LastName;

-- List customers who have both placed orders and added products to their shopping cart:

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName

FROM Customer c

WHERE EXISTS (SELECT 1 FROM OrderTable o WHERE o.CustomerID = c.CustomerID)

AND EXISTS (SELECT 1 FROM CartProduct cp WHERE cp.CustomerID = c.CustomerID);

--This query joins the OrderTable, Customer, and Address tables and orders the results by the order date in descending order.

SELECT o.OrderID, o.OrderDate, o.TotalPrice, c.FirstName || ' ' || c.LastName AS CustomerName, a.City, a.Province, a.Country

FROM OrderTable o

JOIN Customer c ON o.CustomerID = c.CustomerID

JOIN Address a ON o.AddressID = a.AddressID

ORDER BY o.OrderDate DESC;

-- List Cities Where Customers Have Spent More Than $30 in Total

SELECT a.City, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

JOIN OrderTable o ON c.CustomerID = o.CustomerID

JOIN Address a ON c.AddressID = a.AddressID

GROUP BY a.City

HAVING SUM(o.TotalPrice) > 30;

exit;

EOF

Views SH File:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

-- This view combines information from the Customer, OrderTable, and OrderItem tables to provide details about customer orders.

CREATE VIEW CustomerOrderDetails AS

SELECT o.OrderID, o.OrderDate, o.TotalPrice, c.FirstName || ' ' || c.LastName AS CustomerName, p.ProductName, oi.Quantity, oi.Subtotal

FROM OrderTable o

JOIN Customer c ON o.CustomerID = c.CustomerID

JOIN OrderItem oi ON o.OrderID = oi.OrderID

JOIN Product p ON oi.ProductID = p.ProductID;

-- This view shows products that are popular (ordered more than once) along with the total quantity ordered.

CREATE VIEW PopularProducts AS

SELECT p.ProductID, p.ProductName, SUM(oi.Quantity) AS TotalOrdered

FROM Product p

JOIN OrderItem oi ON p.ProductID = oi.ProductID

GROUP BY p.ProductID, p.ProductName

HAVING SUM(oi.Quantity) > 1;

-- This view counts the number of customers in each city.

CREATE VIEW CustomerCityCount AS

SELECT a.City, COUNT(c.CustomerID) AS CustomerCount

FROM Customer c

JOIN Address a ON c.AddressID = a.AddressID

GROUP BY a.City;

-- This view lists customers with active shopping carts.

CREATE VIEW ActiveShoppingCarts AS

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, s.CreationDate, s.CreationTime

FROM Customer c

JOIN ShoppingCart s ON c.CustomerID = s.CustomerID;

-- This view identifies high-value customers based on their total spending.

CREATE VIEW HighValueCustomers AS

SELECT CustomerNAME, TotalSpending FROM

(SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

INNER JOIN OrderTable o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName, c.LastName

ORDER BY TotalSpending DESC)

WHERE ROWNUM = 1;

-- This view identifies product categories with high total sales based on the quantity sold.

CREATE VIEW HighValueProductCategories AS

SELECT CategoryID, CategoryName, TotalQuantitySold FROM

(SELECT pc.CategoryID, ct.CategoryName, SUM(oi.Quantity) AS TotalQuantitySold

FROM ProductCategory pc

JOIN OrderItem oi ON pc.ProductID = oi.ProductID

JOIN CategoryTable ct ON pc.CategoryID = ct.CategoryID

GROUP BY pc.CategoryID, ct.CategoryName

ORDER BY TotalQuantitySold DESC)

WHERE ROWNUM = 1;

exit;

EOF

**Lab06**

Lab 6 delved into the fundamental concept of our database design. That is, functional dependencies shed light on the innate relationships within our database tables. In this lab, we had to show functional dependencies for each table. Below is our work.

| Primary Key | Foreign Key |
| --- | --- |

Address Table

| Address ID | Street Number | Street Name | City | Province | County | Postal Code |
| --- | --- | --- | --- | --- | --- | --- |

**{Address ID (PK)} →** {Street Number**,** Street Name, City, Province, Country, Postal Code}

Customer Table

| Customer ID | Address ID | Username | Password | Email | First Name | Last Name | Phone Number |
| --- | --- | --- | --- | --- | --- | --- | --- |

**{Customer ID (PK)} →** {Username, Password, Email, First Name, Last Name, Phone Number, Address ID}

**{Customer ID (PK)} →** {Address ID}

**{Customer ID (PK), Address ID} →** {First Name, Last Name}

Product Table

| Product ID | Product Name | Product Desc | Price | Stock | Image URL |
| --- | --- | --- | --- | --- | --- |

**{Product ID (PK)} →** {Product Name, Product Dec, Price, Stock, Image URL}

Category Table

| Category ID | Category Name |
| --- | --- |

**{Category ID (PK)} →** {Category Name}

Product Category Table

| Product Category ID | Product ID | Category ID |
| --- | --- | --- |

**{Product Category ID (PK)} →** {Product ID, Category ID}

**{Product ID} →** {Category ID} (This is Partial Dependency)

Order Table

| Order ID | Customer ID | Address ID | Order Date | Order Time | Total Price |
| --- | --- | --- | --- | --- | --- |

**{Order ID (PK), Customer ID, Address ID} →** {Order Date, Order Time, Total Price}

Order Item Table

| Order Item ID | Order ID | Product ID | Quantity | Subtotal |
| --- | --- | --- | --- | --- |

**{Order Item ID (PK), Order ID, Product ID} →** {Quantity, Subtotal}

Shopping Cart Table

| Customer ID | Creation Time | Creation Date |
| --- | --- | --- |

**{Customer ID (PK)} →** {Creation Time, Creation Date}

Cart Product Table

| Cart Product ID | Customer ID | Product ID | Quantity |
| --- | --- | --- | --- |

**{Cart Product ID (PK), Customer ID, Product ID} →** {Quantity}

**Lab07**

In Lab 7, our focus shifted to normalization, particularly honing in on the esteemed Third Normal Form (3NF). Armed with the Bernstein algorithm, we verified our tables' normalization status and undertook the normalization process when necessary.

For a table to be in 3NF:

It means that it should have a composite primary key (a primary key consisting of more than one attribute) and all non-key attributes must be fully functionally dependent on the entire composite primary key.

No transitive dependencies: A table is in 3NF if it does not have any transitive dependencies. A transitive dependency occurs when one non-key attribute depends on another non-key attribute, which in turn depends on the primary key. To eliminate transitive dependencies, we need to ensure that all non-key attributes are directly dependent on the entire primary key.

| Primary Key | Foreign Key |
| --- | --- |

Address Table

| Address ID | Street Number | Street Name | City | Province | County | Postal Code |
| --- | --- | --- | --- | --- | --- | --- |

**{Address ID (PK)} →** {Street Number**,** Street Name, City, Province, Country, Postal Code}

**{Country} →** {City, Province, Country, Postal Code}

**{Province} →** {City}

EXPLANATION:

This table is not in 3NF because we have some non-key attributes depending on other non-key attributes (transitive dependencies). In order to make the table to 3NF, there are a couple of methods, but we chose to do the Bernstein algorithm that was shown in class. After removing all the redundancies, decomposing and creating tables that satisfy the 3NF conditions, we would have the following three tables.

Address Table

| Address ID | Street Number | Street Name | County |
| --- | --- | --- | --- |

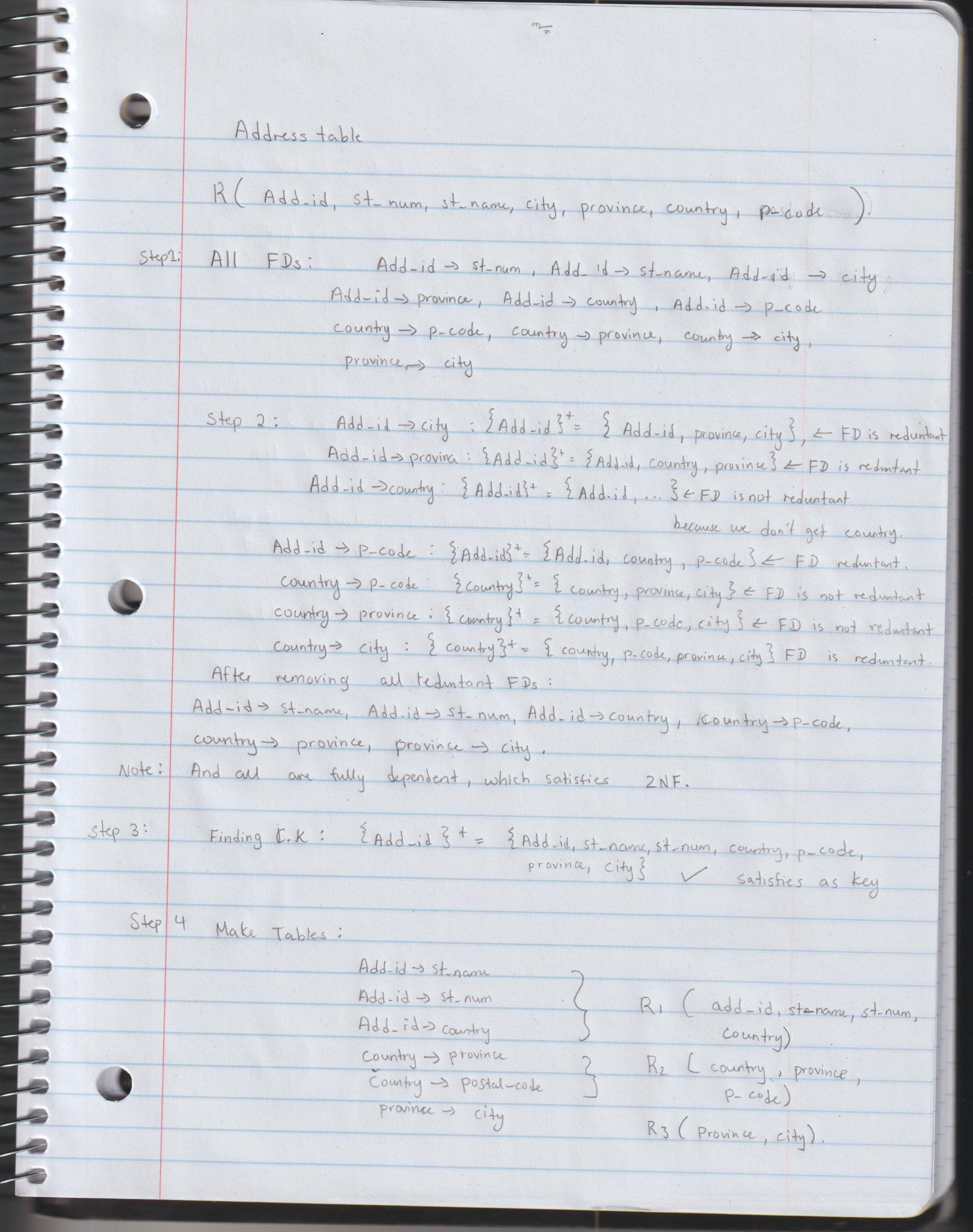
Country Table

| Country | Province | Postal Code |
| --- | --- | --- |

Province Table

| Province | City |
| --- | --- |

And below we have shown the work done using the Bernstein Algorithm to make the Address table to 3NF.



Customer Table

| Customer ID | Address ID | Username | Password | Email | First Name | Last Name | Phone Number |
| --- | --- | --- | --- | --- | --- | --- | --- |

**{Customer ID (PK)} →** {Username, Password, Email, First Name, Last Name, Phone Number, Address ID}

EXPLANATION:

This table is in 3NF already because we have a primary key (CUSTOMER ID), and all other non-key attributes depend on CUSTOMER ID. It also does not have any transitive dependencies because no non-key attribute depends on another non-key attribute. The foreign key ADDRESS ID helps break down transitive dependencies.

Product Table

| Product ID | Product Name | Product Desc | Price | Stock | Image URL |
| --- | --- | --- | --- | --- | --- |

**{Product ID (PK)} →** {Product Name, Product Dec, Price, Stock, Image URL}

EXPLANATION:

This table is in 3NF already because we have a primary key (PRODUCT ID), and all other non-key attributes depend on PRODUCT ID. It also does not have any transitive dependencies because no non-key attribute depends on another non-key attribute.

Category Table

| Category ID | Category Name |
| --- | --- |

**{Category ID (PK)} →** {Category Name}

EXPLANATION:

This table is in 3NF already because we have a primary key (CATEGORY ID), and all other non-key attributes depend on CATEGORY ID. It also does not have any transitive dependencies because no non-key attribute depends on another non-key attribute.

Product Category Table

| Product Category ID | Product ID | Category ID |
| --- | --- | --- |

**{Product Category ID (PK)} →** {Product ID, Category ID}

EXPLANATION:

This table is in 3NF already because we have a primary key (PRODUCT CATEGORY ID), and all other non-key attributes depend on PRODUCT CATEGORY ID. It also does not have any transitive dependencies because no non-key attribute depends on another non-key attribute. the foreign keys PRODUCT ID and CATEGORY ID help break down transitive dependencies.

Order Table

| Order ID | Customer ID | Address ID | Order Date | Order Time | Total Price |
| --- | --- | --- | --- | --- | --- |

**{Order ID (PK), Customer ID, Address ID} →** {Order Date, Order Time, Total Price}

EXPLANATION:

This table is in 3NF already because we have a primary key (ORDER ID), and all other non-key attributes depend on ORDER ID. It also does not have any transitive dependencies because no non-key attribute depends on another non-key attribute. The foreign keys CUSTOMER ID and ADDRESS ID help break down transitive dependencies.

Order Item Table

| Order Item ID | Order ID | Product ID | Quantity | Subtotal |
| --- | --- | --- | --- | --- |

**{Order Item ID (PK), Order ID, Product ID} →** {Quantity, Subtotal}

EXPLANATION:

This table is in 3NF already because we have a primary key (ORDER ITEM ID), and all other non-key attributes depend on ORDER ITEM ID. It also does not have any transitive dependencies because no non-key attribute depends on another non-key attribute. The foreign keys ORDER ID and PRODUCT ID help break down transitive dependencies.

Shopping Cart Table

| Customer ID | Creation Time | Creation Date |
| --- | --- | --- |

**{Customer ID (PK)} →** {Creation Time, Creation Date}

EXPLANATION:

This table is in 3NF already because we have a primary key (CUSTOMER ID), and all other non-key attributes depend on CUSTOMER ID. It also does not have any transitive dependencies because no non-key attribute depends on another non-key attribute.

Cart Product Table

| Cart Product ID | Customer ID | Product ID | Quantity |
| --- | --- | --- | --- |

**{Cart Product ID (PK), Customer ID, Product ID} →** {Quantity}

EXPLANATION:

This table is in 3NF already because we have a primary key (CART PRODUCT ID), and all other non-key attributes depend on CART PRODUCT ID. It also does not have any transitive dependencies because no non-key attribute depends on another non-key attribute. The foreign keys CUSTOMER ID and PRODUCT ID help break down transitive dependencies.

**Lab08**

In Lab 8, our quest for normalization reached new heights as we delved into the intricacies of the Boyce-Codd Normal Form (BCNF). Building upon the foundations laid in previous labs, we directed our attention to verifying the normalization status of our database tables using the BCNF criterion.

For a table to be in BCNF:

It must satisfy 3NF, and for every non-trivial functional dependency, the determinant must be a superkey.

| Primary Key | Foreign Key |
| --- | --- |

Address Table

| Address ID | Street Number | Street Name | City | Province | County | Postal Code |
| --- | --- | --- | --- | --- | --- | --- |

**{Address ID (PK)} →** {Street Number**,** Street Name, City, Province, Country, Postal Code}

**{Country} →** {City, Province, Country, Postal Code}

**{Province} →** {City}

EXPLANATION:

This table is not in 3NF because we have some non-key attributes depending on other non-key attributes (transitive dependencies). In order to make the table to 3NF, there are a couple of methods, but we chose to do the Bernstein algorithm that was shown in class. After removing all the redundancies, decomposing and creating tables that satisfy the 3NF conditions, we would have the following three tables.

Address Table

| Address ID | Street Number | Street Name | County |
| --- | --- | --- | --- |

Country Table

| Country | Province | Postal Code |
| --- | --- | --- |

Province Table

| Province | City |
| --- | --- |

And below we have shown the work done using the Bernstein Algorithm to make the Address table to 3NF.

Address Table

| Address ID | Street Number | Street Name | County |
| --- | --- | --- | --- |

**{Address ID (PK)} →** {Street Number**,** Street Name, Country}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Address\_ID}. This means that there does not exist a functional dependency in which a non-candidate key determines {Address\_ID}.

Country Table

| Country | Province | Postal Code |
| --- | --- | --- |

**{Country (PK), Province} →** {Postal Code}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Country}. This means that there does not exist a functional dependency in which a non-candidate key determines {Country}.

Province Table

| Province | City |
| --- | --- |

**{Province (PK)} →** {City}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Province}. This means that there does not exist a functional dependency in which a non-candidate key determines {Province}.

Customer Table

| Customer ID | Address ID | Username | Password | Email | First Name | Last Name | Phone Number |
| --- | --- | --- | --- | --- | --- | --- | --- |

**{Customer ID (PK)} →** {Username, Password, Email, First Name, Last Name, Phone Number, Address ID}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Customer ID}. This means that there does not exist a functional dependency in which a non-candidate key determines {Customer ID}.

Product Table

| Product ID | Product Name | Product Desc | Price | Stock | Image URL |
| --- | --- | --- | --- | --- | --- |

**{Product ID (PK)} →** {Product Name, Product Dec, Price, Stock, Image URL}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Product ID}. This means that there does not exist a functional dependency in which a non-candidate key determines {Product ID}.

Category Table

| Category ID | Category Name |
| --- | --- |

**{Category ID (PK)} →** {Category Name}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Category ID}. This means that there does not exist a functional dependency in which a non-candidate key determines {Category ID}.

Product Category Table

| Product Category ID | Product ID | Category ID |
| --- | --- | --- |

**{Product Category ID (PK)} →** {Product ID, Category ID}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Product Category ID}. This means that there does not exist a functional dependency in which a non-candidate key determines {Product Category ID}.

Order Table

| Order ID | Customer ID | Address ID | Order Date | Order Time | Total Price |
| --- | --- | --- | --- | --- | --- |

**{Order ID (PK)} →** {Customer ID, Address ID, Order Date, Order Time, Total Price}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Order ID}. This means that there does not exist a functional dependency in which a non-candidate key determines {Order ID}.

Order Item Table

| Order Item ID | Order ID | Product ID | Quantity | Subtotal |
| --- | --- | --- | --- | --- |

**{Order Item ID (PK)} →** {Order ID, Product ID, Quantity, Subtotal}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Order Item ID}. This means that there does not exist a functional dependency in which a non-candidate key determines {Order Item ID}.

Shopping Cart Table

| Customer ID | Creation Time | Creation Date |
| --- | --- | --- |

**{Customer ID (PK)} →** {Creation Time, Creation Date}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Customer ID}. This means that there does not exist a functional dependency in which a non-candidate key determines {Customer ID}.

Cart Product Table

| Cart Product ID | Customer ID | Product ID | Quantity |
| --- | --- | --- | --- |

**{Cart Product ID (PK)} →** {Customer ID, Product ID, Quantity}

EXPLANATION:

This table is in BCNF because along with being in 3NF, the non-trivial functional dependencies all have a determinant of a minimal super key (candidate key). The candidate key for this table is {Cart Product ID}. This means that there does not exist a functional dependency in which a non-candidate key determines {Cart Product ID}.

**Lab09**

In Lab 9, we wrapped up our database design by translating our normalized tables into practical SQL commands. Additionally, we did the bonus assignment, crafting a user-friendly graphical interface using PHP to interact with our database.

Here is the updated Bash Commands:

Create Table SH file:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.cs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

-- Address table

CREATE TABLE ProvinceTable (

ProvinceID NUMBER(5) PRIMARY KEY,

Province VARCHAR2(50),

City VARCHAR2(50)

);

CREATE TABLE CountryTable (

CountryID NUMBER(5) PRIMARY KEY,

Country VARCHAR2(50),

PostalCode VARCHAR2(10),

ProvinceID NUMBER(5) REFERENCES ProvinceTable(ProvinceID)

);

CREATE TABLE Address (

AddressID NUMBER(5) PRIMARY KEY,

StreetNumber VARCHAR2(10),

StreetName VARCHAR2(100),

CountryID NUMBER(5) REFERENCES CountryTable(CountryID)

);

-- Customer table

CREATE TABLE Customer (

CustomerID NUMBER(5) PRIMARY KEY,

Username VARCHAR2(50) NOT NULL,

Passwd VARCHAR2(50) NOT NULL,

Email VARCHAR2(100),

FirstName VARCHAR2(50),

LastName VARCHAR2(50),

PhoneNumber VARCHAR2(20),

AddressID NUMBER(5) REFERENCES Address(AddressID)

);

-- Product table

CREATE TABLE Product (

ProductID NUMBER(5) PRIMARY KEY,

ProductName VARCHAR2(100) NOT NULL,

ProductDesc VARCHAR2(1000),

Price NUMBER(10, 2) NOT NULL,

Stock NUMBER(5) NOT NULL,

ImageURL VARCHAR2(255)

);

-- Category table

CREATE TABLE CategoryTable (

CategoryID NUMBER(5) PRIMARY KEY,

CategoryName VARCHAR2(100) NOT NULL

);

-- ProductCategory table

CREATE TABLE ProductCategory (

ProductCategoryID NUMBER(5) PRIMARY KEY,

ProductID NUMBER(5) REFERENCES Product(ProductID),

CategoryID NUMBER(5) REFERENCES CategoryTable(CategoryID)

);

-- Order table

CREATE TABLE OrderTable (

OrderID NUMBER(5) PRIMARY KEY,

CustomerID NUMBER(5) REFERENCES Customer(CustomerID),

AddressID NUMBER(5) REFERENCES Address(AddressID),

OrderDate DATE NOT NULL,

OrderTime TIMESTAMP NOT NULL,

TotalPrice NUMBER(10, 2) NOT NULL

);

-- OrderItem table

CREATE TABLE OrderItem (

OrderItemID NUMBER(5) PRIMARY KEY,

OrderID NUMBER(5) REFERENCES OrderTable(OrderID),

ProductID NUMBER(5) REFERENCES Product(ProductID),

Quantity NUMBER(5) NOT NULL,

Subtotal NUMBER(10, 2) NOT NULL

);

-- Create the ShoppingCart table

CREATE TABLE ShoppingCart (

CustomerID NUMBER(5) PRIMARY KEY,

CreationDate DATE NOT NULL,

CreationTime TIMESTAMP NOT NULL

);

-- CartProduct table

CREATE TABLE CartProduct (

CartProductID NUMBER(5) PRIMARY KEY,

CustomerID NUMBER(5) REFERENCES ShoppingCart(CustomerID),

ProductID NUMBER(5) REFERENCES Product(ProductID),

Quantity NUMBER(5) NOT NULL

);

exit;

EOF

Drop Table SH file:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

-- Drop first

DROP TABLE CartProduct;

DROP TABLE ProductCategory;

DROP TABLE ShoppingCart;

DROP TABLE OrderItem;

-- Later Drop these in Order

DROP TABLE OrderTable;

DROP TABLE CategoryTable;

DROP TABLE Product;

DROP TABLE Customer;

DROP TABLE Address;

DROP TABLE CountryTable;

DROP TABLE ProvinceTable;

exit;

EOF

Populate Table SH file:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

Sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.cs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

-- Insert data into the Province table

INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (1, 'Ontario', 'Toronto');

INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (2, 'New York', 'New York');

INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (3, 'California', 'Los Angeles');

INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (4, 'Illinois', 'Chicago');

INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (5, 'Texas', 'Houston');

-- Insert data into the Country table

INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (1, 'Canada', 'P7H0A8', 1);

INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (2, 'USA', '10001', 2);

INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (3, 'USA', '90001', 3);

INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (4, 'USA', '60601', 4);

INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (5, 'USA', '77001', 5);

INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (6, 'Canada', 'M1Z001', 1);

INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (7, 'USA', '90101', 2);

-- Insert data into the Address table

INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (1, '123', 'Main St', 1);

INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (2, '456', 'Oak Avenue', 2);

INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (3, '789', 'Cedar Lane', 3);

INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (4, '1010', 'Maple Street', 4);

INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (5, '234', 'Pine Avenue', 5);

INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (6, '134', 'Brooke Road',1);

INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (7, '134', 'Brooklyn Street', 2);

-- Insert data for Customer table

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (1, 'user1', 'password1', 'user1@example.com', 'John', 'Doe', '657-123-4567', 1);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (2, 'user2', 'password2', 'user2@example.com', 'Alice', 'Smith', '987-654-3210', 2);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (3, 'user3', 'password3', 'user3@example.com', 'Bob', 'Johnson', '555-123-4567', 3);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (4, 'user4', 'password4', 'user4@example.com', 'Emily', 'Brown', '444-789-0123', 4);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (5, 'user5', 'password5', 'user5@example.com', 'Grace', 'Lee', '111-222-3333', 5);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (6, 'user6', 'password6', 'user6@example.com', 'Bob', 'Marley', '111-444-3333', 6);

INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (7, 'user7', 'password7', 'user7@example.com', 'Bob', 'Dylan', '000-434-2333', 7);

-- Insert data forProduct table

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (1, 'Men''s Cotton Shirt', 'High-quality cotton shirt for men', 29.99, 150, 'shirt.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (2, 'Women''s Leather Handbag', 'Stylish leather handbag for women', 79.99, 80, 'handbag.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (3, 'Organic All-Purpose Flour', 'Certified organic wheat flour', 5.99, 200, 'flour.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (4, 'Coca-Cola Classic', 'Classic carbonated soft drink', 1.99, 500, 'cocacola.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (5, 'Apple iPhone 13', 'Latest model with advanced features', 999.99, 30, 'iphone.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (6, 'Samsung 55" 4K Smart TV', 'Crystal-clear 4K Ultra HD television', 799.99, 10, 'tv.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (7, 'Bluetooth Wireless Earbuds', 'Wireless earbuds with long battery life', 49.99, 100, 'earbuds.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (8, 'Dell Inspiron Laptop', 'Powerful laptop for work and entertainment', 899.99, 40, 'laptop.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (9, 'Fitness Tracker Watch', 'Track your health and fitness activities', 29.99, 120, 'fitness.jpg');

INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (10, 'Chocolate Chip Cookies', 'Delicious homemade chocolate chip cookies', 3.99, 300, 'cookies.jpg');

-- Insert data for the CategoryTable

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (1, 'Clothing');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (2, 'Fashion Accessories');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (3, 'Electronics');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (4, 'Food');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (5, 'Fitness Equipment');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (6, 'Beverages');

INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (7, 'Home Appliances');

-- Insert data ProductCategory table

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (1, 1, 1);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (2, 1, 2);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (3, 2, 1);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (4, 3, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (5, 4, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (6, 4, 6);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (7, 5, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (8, 5, 2);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (9, 6, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (10, 6, 7);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (11, 7, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (12, 8, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (13, 9, 2);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (14, 9, 3);

INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (15, 10, 4);

-- Insert data for OrderTable

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (1, 1, 1, TO\_DATE('2022-09-25', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 10.99);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (2, 2, 2, TO\_DATE('2020-09-16', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 39.99);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (3, 3, 3, TO\_DATE('2023-01-07', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 17.49);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (4, 4, 4, TO\_DATE('2023-10-18', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 24.95);

INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (5, 5, 5, TO\_DATE('2023-02-27', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP, 35.99);

-- Insert data into the OrderItem table

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (1, 1, 2, 2, 21.98);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (2, 1, 5, 3, 59.97);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (3, 2, 3, 1, 7.49);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (4, 2, 4, 2, 49.90);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (5, 3, 5, 4, 63.96);

INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (6, 4, 8, 1, 33.96);

-- Insert data into the ShoppingCart table

INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (1, TO\_DATE('2020-09-25', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP);

INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (2, TO\_DATE('2023-09-26', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP);

INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (3, TO\_DATE('2023-09-27', 'YYYY-MM-DD'), CURRENT\_TIMESTAMP);

-- Insert data into the CartProduct table

INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (1, 1, 1, 3);

INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (2, 1, 2, 2);

INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (3, 2, 3, 1);

exit;

EOF

Queries SH File:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

-- List all attributes of all products

SELECT \* FROM Product;

-- List Product Name and their Price for all products

SELECT 'The Product: ', ProductName, ' costs ', Price

FROM Product;

-- List the Unique First Name of all the customers

SELECT DISTINCT FirstName FROM Customer;

-- List all attributes of products that costs more than $100.0 or

-- have more than 200 items in stock.

SELECT \* FROM Product

WHERE Price > 100.0 OR Stock > 200;

-- List all the Subtotal of each product purchased by Customer1.

SELECT Subtotal AS Purchases\_by\_Customer1 FROM OrderItem

WHERE OrderID = 1;

-- List the Order ID and Total Price of all orders sorted by Order

-- Date in ascending order (Earliest to Latest).

SELECT OrderID, TotalPrice FROM OrderTable

ORDER BY OrderDate;

-- List the username and password of customer with an id greater than 3

SELECT username, passwd

FROM customer

WHERE customerid>3;

-- List the names of products that are popular (customers got more than one of)

SELECT productname

FROM product, cartproduct

Where quantity > 1

AND cartproduct.productid=product.productid;

-- List the customer id of customers that have spent less than $25 on their order

SELECT 'Customer ID is: ', customerid

FROM ordertable

WHERE totalprice < 25;

-- List the order IDs for orders that spent more than $50

SELECT orderid

FROM orderitem

WHERE subtotal>50;

-- List information of customers with an active shopping cart

SELECT c.\*

FROM customer c, shoppingcart s

WHERE c.customerid=s.customerid;

-- List all product ID's that are within the Clothing category

SELECT productid

FROM productcategory

WHERE categoryid='1';

-- List the category ID for the food category

SELECT categoryid

FROM categorytable

WHERE categoryname="food";

-- Count the number of products in each category.

SELECT CategoryName, COUNT(ProductCategoryID) AS NumberOfProducts

FROM CategoryTable

LEFT JOIN ProductCategory ON CategoryTable.CategoryID = ProductCategory.CategoryID

GROUP BY CategoryName;

-- List the total number of products in each shopping cart.

SELECT CustomerID, COUNT(ProductID) AS TotalProductsInCart

FROM ShoppingCart

LEFT JOIN CartProduct ON ShoppingCart.CustomerID = CartProduct.CustomerID

GROUP BY CustomerID;

-- List the products along with their quantities in the shopping cart of a specific customer.

-- In this query, we used tablesm aliases cp and p for a more advanced looking query

SELECT cp.CustomerID, p.ProductName, cp.Quantity

FROM CartProduct cp

JOIN Product p ON cp.ProductID = p.ProductID

WHERE cp.CustomerID = 1;

-- List products and their categories:

SELECT p.ProductName, ct.CategoryName

FROM Product p

JOIN ProductCategory pc ON p.ProductID = pc.ProductID

JOIN CategoryTable ct ON pc.CategoryID = ct.CategoryID;

-- List customers and their total spending:

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

LEFT JOIN OrderTable o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName, c.LastName;

-- List customers who have both placed orders and added products to their shopping cart:

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName

FROM Customer c

WHERE EXISTS (SELECT 1 FROM OrderTable o WHERE o.CustomerID = c.CustomerID)

AND EXISTS (SELECT 1 FROM CartProduct cp WHERE cp.CustomerID = c.CustomerID);

exit;

EOF

Views SH file:

#!/bin/sh

#export LD\_LIBRARY\_PATH=/usr/lib/oracle/12.1/client64/lib

sqlplus64 "username/password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs.ryerson.ca)(Port=1521))(CONNECT\_DATA=(SID=orcl)))" <<EOF

-- This view combines information from the Customer, OrderTable, and OrderItem tables to provide details about customer orders.

CREATE VIEW CustomerOrderDetails AS

SELECT o.OrderID, o.OrderDate, o.TotalPrice, c.FirstName || ' ' || c.LastName AS CustomerName, p.ProductName, oi.Quantity, oi.Subtotal

FROM OrderTable o

JOIN Customer c ON o.CustomerID = c.CustomerID

JOIN OrderItem oi ON o.OrderID = oi.OrderID

JOIN Product p ON oi.ProductID = p.ProductID;

-- This view shows products that are popular (ordered more than once) along with the total quantity ordered.

CREATE VIEW PopularProducts AS

SELECT p.ProductID, p.ProductName, SUM(oi.Quantity) AS TotalOrdered

FROM Product p

JOIN OrderItem oi ON p.ProductID = oi.ProductID

GROUP BY p.ProductID, p.ProductName

HAVING SUM(oi.Quantity) > 1;

-- This view lists customers with active shopping carts.

CREATE VIEW ActiveShoppingCarts AS

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, s.CreationDate, s.CreationTime

FROM Customer c

JOIN ShoppingCart s ON c.CustomerID = s.CustomerID;

-- This view identifies high-value customers based on their total spending.

CREATE VIEW HighValueCustomers AS

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

LEFT JOIN OrderTable o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName, c.LastName

HAVING SUM(o.TotalPrice) > 100;

-- This view identifies product categories with high total sales based on the quantity sold.

CREATE VIEW HighValueProductCategories AS

SELECT pc.CategoryID, ct.CategoryName, SUM(oi.Quantity) AS TotalQuantitySold

FROM ProductCategory pc

JOIN OrderItem oi ON pc.ProductID = oi.ProductID

JOIN CategoryTable ct ON pc.CategoryID = ct.CategoryID

GROUP BY pc.CategoryID, ct.CategoryName

HAVING SUM(oi.Quantity) > 50;

exit;

EOF

Now for the bonus marks:

Create Table PHP file:

<?php

*function* createTables($connect) {

$result= "";

$sql = "CREATE TABLE ProvinceTable (

ProvinceID *INT*(5) PRIMARY KEY,

Province *VARCHAR*(50),

City *VARCHAR*(50)

);";

if (mysqli\_query($connect, $sql)) {

$result .= "ProvinceTable Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "CREATE TABLE CountryTable (

CountryID *INT*(5) PRIMARY KEY,

Country *VARCHAR*(50),

PostalCode *VARCHAR*(10),

ProvinceID *INT*(5) REFERENCES ProvinceTable(ProvinceID)

);";

if (mysqli\_query($connect, $sql)) {

$result .= "CountryTable Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "CREATE TABLE Address (

AddressID *INT*(5) PRIMARY KEY,

StreetNumber *VARCHAR*(10),

StreetName *VARCHAR*(100),

CountryID *INT*(5) REFERENCES CountryTable(CountryID)

);";

if (mysqli\_query($connect, $sql)) {

$result .= "Address Created\n";

}

$sql = "CREATE TABLE Customer (

CustomerID *INT*(5) PRIMARY KEY,

Username *VARCHAR*(50) NOT NULL,

Passwd *VARCHAR*(50) NOT NULL,

Email *VARCHAR*(100),

FirstName *VARCHAR*(50),

LastName *VARCHAR*(50),

PhoneNumber *VARCHAR*(20),

AddressID *INT*(5) REFERENCES Address(AddressID)

);";

if (mysqli\_query($connect, $sql)) {

$result .= "Customer Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "CREATE TABLE Product (

ProductID *INT*(5) PRIMARY KEY,

ProductName *VARCHAR*(100) NOT NULL,

ProductDesc *VARCHAR*(1000),

Price *DECIMAL*(10, 2) NOT NULL,

Stock *INT*(5) NOT NULL,

ImageURL *VARCHAR*(255)

)";

if (mysqli\_query($connect, $sql)) {

$result .= "Product Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "CREATE TABLE CategoryTable (

CategoryID *INT*(5) PRIMARY KEY,

CategoryName *VARCHAR*(100) NOT NULL

);";

if (mysqli\_query($connect, $sql)) {

$result .= "CategoryTable Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "CREATE TABLE ProductCategory (

ProductCategoryID *INT*(5) PRIMARY KEY,

ProductID *INT*(5) REFERENCES Product(ProductID),

CategoryID *INT*(5) REFERENCES CategoryTable(CategoryID)

);";

if (mysqli\_query($connect, $sql)) {

$result .= "ProductCategory Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "CREATE TABLE OrderTable (

OrderID *INT*(5) PRIMARY KEY,

CustomerID *INT*(5) REFERENCES Customer(CustomerID),

AddressID *INT*(5) REFERENCES Address(AddressID),

OrderDate *DATE* NOT NULL,

OrderTime *TIMESTAMP* NOT NULL,

TotalPrice *DECIMAL*(10, 2) NOT NULL

);";

if (mysqli\_query($connect, $sql)) {

$result .= "OrderTable Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "CREATE TABLE OrderItem (

OrderItemID *INT*(5) PRIMARY KEY,

OrderID *INT*(5) REFERENCES OrderTable(OrderID),

ProductID *INT*(5) REFERENCES Product(ProductID),

Quantity *INT*(5) NOT NULL,

Subtotal *DECIMAL*(10, 2) NOT NULL

);";

if (mysqli\_query($connect, $sql)) {

$result .= "OrderItem Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "CREATE TABLE ShoppingCart (

CustomerID *INT*(5) PRIMARY KEY,

CreationDate *DATE* NOT NULL,

CreationTime *TIMESTAMP* NOT NULL

);";

if (mysqli\_query($connect, $sql)) {

$result .= "ShoppingCart Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "CREATE TABLE CartProduct (

CartProductID *INT*(5) PRIMARY KEY,

CustomerID *INT*(5) REFERENCES ShoppingCart(CustomerID),

ProductID *INT*(5) REFERENCES Product(ProductID),

Quantity *INT*(5) NOT NULL

);";

if (mysqli\_query($connect, $sql)) {

$result .= "CartProduct Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

return $result;

}

Drop Table PHP file:

<?php

*function* dropTables($connect) {

$result= "";

$sql = "DROP TABLE ProvinceTable;";

if (mysqli\_query($connect, $sql)) {

$result .= "ProvinceTable Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "DROP TABLE CountryTable;";

if (mysqli\_query($connect, $sql)) {

$result .= "CountryTable Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "DROP TABLE Address;";

if (mysqli\_query($connect, $sql)) {

$result .= "Address Dropped\n";

}

$sql = "DROP TABLE Customer;";

if (mysqli\_query($connect, $sql)) {

$result .= "Customer Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "DROP TABLE Product";

if (mysqli\_query($connect, $sql)) {

$result .= "Product Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "DROP TABLE CategoryTable;";

if (mysqli\_query($connect, $sql)) {

$result .= "CategoryTable Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "DROP TABLE ProductCategory;";

if (mysqli\_query($connect, $sql)) {

$result .= "ProductCategory Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "DROP TABLE OrderTable;";

if (mysqli\_query($connect, $sql)) {

$result .= "OrderTable Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "DROP TABLE OrderItem;";

if (mysqli\_query($connect, $sql)) {

$result .= "OrderItem Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "DROP TABLE ShoppingCart;";

if (mysqli\_query($connect, $sql)) {

$result .= "ShoppingCart Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "DROP TABLE CartProduct;";

if (mysqli\_query($connect, $sql)) {

$result .= "CartProduct Dropped\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

return $result;

}

Index.php file:

<?php

include('createTables.php');

include('dropTables.php');

include('populateTables.php');

include('queries.php');

include('views.php');

$connect = mysqli\_connect("localhost", "h6sharif", "vVtga8OW",

"h6sharif") or die(mysqli\_error());

if(isset($\_POST['create'])) {

$result = createTables($connect);

}

if(isset($\_POST['drop'])) {

$result = dropTables($connect);

}

if(isset($\_POST['populate'])) {

$result = populateTables($connect);

}

if(isset($\_POST['run'])) {

$result = queryTables($connect);

}

if(isset($\_POST['view'])) {

$result = viewTables($connect);

}

?>

<!DOCTYPE html>

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>E-commerce</title>

<style>

body {

*font-family*: Arial, sans-serif;

*text-align*: center;

*margin*: 0;

*padding*: 0;

*background-color*: #9ba39c;

}

header {

*background-color*: #333;

*color*: #fff;

*padding*: 1em;

}

main {

*padding*: 2em;

}

.container {

*display*: flex;

*flex-direction*: column;

*align-items*: center;

}

h1 {

*margin-bottom*: 20px;

}

.button-container {

*display*: flex;

*justify-content*: center;

}

.button {

*margin*: 0 10px;

*padding*: 10px 20px;

*font-size*: 16px;

*cursor*: pointer;

*background-color*: #6d8570;

*color*: white;

*border*: none;

}

.button:hover {

*background-color*: #ddd;

*color*: black;

}

</style>

</head>

<body>

<header>

<h1>E-Commerce Showcase</h1>

</header>

<main>

<div class="container">

<div class="button-container">

<form method="post">

<input class="button" type="submit" name="create" value="Create Tables">

<input class="button" type="submit" name="drop" value="Drop Tables">

<input class="button" type="submit" name="populate" value="Populate Tables">

<input class="button" type="submit" name="run" value="Run Queries">

<input class="button" type="submit" name="view" value="View Tables">

</form>

</div>

</div>

<br>

<br>

<center>

<textarea style="font-weight: bold;" rows="50" cols="125"><?php echo $result; ?></textarea>

</center>

</main>

</body>

</html>

Populate Tables PHP file:

<?php

*function* populateTables($connect) {

$result = "";

$sql = "INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (1, 'Ontario', 'Toronto');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProvinceTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (2, 'New York', 'New York');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProvinceTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (3, 'California', 'Los Angeles');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProvinceTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (4, 'Illinois', 'Chicago');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProvinceTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProvinceTable (ProvinceID, Province, City)

VALUES (5, 'Texas', 'Houston');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProvinceTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (1, 'Canada', 'P7H0A8', 1);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CountryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (2, 'USA', '10001', 2);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CountryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (3, 'USA', '90001', 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CountryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (4, 'USA', '60601', 4);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CountryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (5, 'USA', '77001', 5);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CountryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (6, 'Canada', 'M1Z001', 1);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CountryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CountryTable (CountryID, Country, PostalCode, ProvinceID)

VALUES (7, 'USA', '90101', 2);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CountryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (1, '123', 'Main St', 1);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Address\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (2, '456', 'Oak Avenue', 2);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Address\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (3, '789', 'Cedar Lane', 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Address\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (4, '1010', 'Maple Street', 4);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Address\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (5, '234', 'Pine Avenue', 5);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Address\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (6, '134', 'Brooke Road',1);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Address\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Address (AddressID, StreetNumber, StreetName, CountryID)

VALUES (7, '134', 'Brooklyn Street', 2);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Address\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (1, 'user1', 'password1', 'user1@example.com', 'John', 'Doe', '657-123-4567', 1);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Customer\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (2, 'user2', 'password2', 'user2@example.com', 'Alice', 'Smith', '987-654-3210', 2);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Customer\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (3, 'user3', 'password3', 'user3@example.com', 'Bob', 'Johnson', '555-123-4567', 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Customer\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (4, 'user4', 'password4', 'user4@example.com', 'Emily', 'Brown', '444-789-0123', 4);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Customer\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (5, 'user5', 'password5', 'user5@example.com', 'Grace', 'Lee', '111-222-3333', 5);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Customer\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (6, 'user6', 'password6', 'user6@example.com', 'Bob', 'Marley', '111-444-3333', 6);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Customer\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Customer (CustomerID, Username, Passwd, Email, FirstName, LastName, PhoneNumber, AddressID)

VALUES (7, 'user7', 'password7', 'user7@example.com', 'Bob', 'Dylan', '000-434-2333', 7);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Customer\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (1, 'Men''s Cotton Shirt', 'High-quality cotton shirt for men', 29.99, 150, 'shirt.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (2, 'Women''s Leather Handbag', 'Stylish leather handbag for women', 79.99, 80, 'handbag.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (3, 'Organic All-Purpose Flour', 'Certified organic wheat flour', 5.99, 200, 'flour.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (4, 'Coca-Cola Classic', 'Classic carbonated soft drink', 1.99, 500, 'cocacola.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (5, 'Apple iPhone 13', 'Latest model with advanced features', 999.99, 30, 'iphone.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (6, 'Samsung 55\" 4K Smart TV', 'Crystal-clear 4K Ultra HD television', 799.99, 10, 'tv.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (7, 'Bluetooth Wireless Earbuds', 'Wireless earbuds with long battery life', 49.99, 100, 'earbuds.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (8, 'Dell Inspiron Laptop', 'Powerful laptop for work and entertainment', 899.99, 40, 'laptop.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (9, 'Fitness Tracker Watch', 'Track your health and fitness activities', 29.99, 120, 'fitness.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO Product (ProductID, ProductName, ProductDesc, Price, Stock, ImageURL)

VALUES (10, 'Chocolate Chip Cookies', 'Delicious homemade chocolate chip cookies', 3.99, 300, 'cookies.jpg');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into Product\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (1, 'Clothing');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CategoryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (2, 'Fashion Accessories');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CategoryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (3, 'Electronics');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CategoryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (4, 'Food');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CategoryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (5, 'Fitness Equipment');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CategoryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (6, 'Beverages');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CategoryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CategoryTable (CategoryID, CategoryName)

VALUES (7, 'Home Appliances');";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CategoryTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

// Insert data into the ProductCategory table

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (1, 1, 1);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (2, 1, 2);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (3, 2, 1);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (4, 3, 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (5, 4, 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (6, 4, 6);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (7, 5, 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (8, 5, 2);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (9, 6, 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (10, 6, 7);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (11, 7, 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (12, 8, 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (13, 9, 2);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (14, 9, 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ProductCategory (ProductCategoryID, ProductID, CategoryID)

VALUES (15, 10, 4);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ProductCategory\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

// Insert data into the OrderTable table

$sql = "INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (1, 1, 1, STR\_TO\_DATE('2022-09-25', '%Y-%m-%d'), CURRENT\_TIMESTAMP, 10.99);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (2, 2, 2, STR\_TO\_DATE('2020-09-16', '%Y-%m-%d'), CURRENT\_TIMESTAMP, 39.99);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (3, 3, 3, STR\_TO\_DATE('2023-01-07', '%Y-%m-%d'), CURRENT\_TIMESTAMP, 17.49);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (4, 4, 4, STR\_TO\_DATE('2023-10-18', '%Y-%m-%d'), CURRENT\_TIMESTAMP, 24.95);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO OrderTable (OrderID, CustomerID, AddressID, OrderDate, OrderTime, TotalPrice)

VALUES (5, 5, 5, STR\_TO\_DATE('2023-02-27', '%Y-%m-%d'), CURRENT\_TIMESTAMP, 35.99);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderTable\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

// Insert data into the OrderItem table

$sql = "INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (1, 1, 2, 2, 21.98);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderItem\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (2, 1, 5, 3, 59.97);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderItem\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (3, 2, 3, 1, 7.49);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderItem\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (4, 2, 4, 2, 49.90);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderItem\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (5, 3, 5, 4, 63.96);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderItem\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO OrderItem (OrderItemID, OrderID, ProductID, Quantity, Subtotal)

VALUES (6, 4, 8, 1, 33.96);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into OrderItem\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (1, STR\_TO\_DATE('2020-09-25', '%Y-%m-%d'), CURRENT\_TIMESTAMP);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ShoppingCart\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (2, STR\_TO\_DATE('2023-09-26', '%Y-%m-%d'), CURRENT\_TIMESTAMP);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ShoppingCart\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO ShoppingCart (CustomerID, CreationDate, CreationTime)

VALUES (3, STR\_TO\_DATE('2023-09-27', '%Y-%m-%d'), CURRENT\_TIMESTAMP);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into ShoppingCart\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (1, 1, 1, 3);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CartProduct\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (2, 1, 2, 2);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CartProduct\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "INSERT INTO CartProduct (CartProductID, CustomerID, ProductID, Quantity)

VALUES (3, 2, 3, 1);";

if (mysqli\_query($connect, $sql)) {

$result .= "Data inserted into CartProduct\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

return $result;

}

Queries PHP file:

<?php

*function* queryTables($connect) {

$result = "";

$sql = "SELECT \* FROM Product;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT 'The Product: ', ProductName, ' costs ', Price FROM Product;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT DISTINCT FirstName FROM Customer;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT \* FROM Product WHERE Price > 100.0 OR Stock > 200;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT Subtotal AS Purchases\_by\_Customer1 FROM OrderItem WHERE OrderID = 1;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT OrderID, TotalPrice FROM OrderTable ORDER BY OrderDate;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT username, passwd

FROM Customer

WHERE customerid>3;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT ProductName

FROM Product, CartProduct

Where quantity > 1

AND CartProduct.ProductID=Product.ProductID;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT 'Customer ID is: ', customerid

FROM OrderTable

WHERE totalprice < 25;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT orderid

FROM OrderItem

WHERE subtotal>50;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT c.\*

FROM Customer c, ShoppingCart s

WHERE c.customerid=s.customerid;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT productid

FROM ProductCategory

WHERE categoryid='1';";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT categoryid

FROM CategoryTable

WHERE categoryname='food';";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT CategoryName, COUNT(ProductCategoryID) AS NumberOfProducts

FROM CategoryTable

LEFT JOIN ProductCategory ON CategoryTable.CategoryID = ProductCategory.CategoryID

GROUP BY CategoryName;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT ShoppingCart.CustomerID, COUNT(ProductID) AS TotalProductsInCart

FROM ShoppingCart

LEFT JOIN CartProduct ON ShoppingCart.CustomerID = CartProduct.CustomerID

GROUP BY CustomerID;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql . mysqli\_error($connect);;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT cp.CustomerID, p.ProductName, cp.Quantity

FROM CartProduct cp

JOIN Product p ON cp.ProductID = p.ProductID

WHERE cp.CustomerID = 1;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT p.ProductName, ct.CategoryName

FROM Product p

JOIN ProductCategory pc ON p.ProductID = pc.ProductID

JOIN CategoryTable ct ON pc.CategoryID = ct.CategoryID;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

LEFT JOIN OrderTable o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName, c.LastName;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName

FROM Customer c

WHERE EXISTS (SELECT 1 FROM OrderTable o WHERE o.CustomerID = c.CustomerID)

AND EXISTS (SELECT 1 FROM CartProduct cp WHERE cp.CustomerID = c.CustomerID);";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "Error: No results found for " . $sql;

}

$result .= "---------------------------------------------------------------------------------------\n";

return $result;

}

?>

Views PHP file:

<?php

*function* viewTables($connect) {

$sql = "DROP VIEW CustomerOrderDetails;";

mysqli\_query($connect, $sql);

$sql = "DROP VIEW PopularProducts;";

mysqli\_query($connect, $sql);

$sql = "DROP VIEW ActiveShoppingCarts;";

mysqli\_query($connect, $sql);

$sql = "DROP VIEW HighValueCustomers;";

mysqli\_query($connect, $sql);

$sql = "DROP VIEW HighValueProductCategories;";

mysqli\_query($connect, $sql);

$result = "";

$sql = "CREATE VIEW CustomerOrderDetails AS

SELECT o.OrderID, o.OrderDate, o.TotalPrice, c.FirstName || ' ' || c.LastName AS CustomerName, p.ProductName, oi.Quantity, oi.Subtotal

FROM OrderTable o

JOIN Customer c ON o.CustomerID = c.CustomerID

JOIN OrderItem oi ON o.OrderID = oi.OrderID

JOIN Product p ON oi.ProductID = p.ProductID;";

if (mysqli\_query($connect, $sql)) {

$result .= "View CustomerOrderDetails Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "SELECT \* FROM CustomerOrderDetails;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "No results found for " . $sql . "\n";

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "CREATE VIEW PopularProducts AS

SELECT p.ProductID, p.ProductName, SUM(oi.Quantity) AS TotalOrdered

FROM Product p

JOIN OrderItem oi ON p.ProductID = oi.ProductID

GROUP BY p.ProductID, p.ProductName

HAVING SUM(oi.Quantity) > 1;";

if (mysqli\_query($connect, $sql)) {

$result .= "View PopularProducts Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "SELECT \* FROM PopularProducts;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "No results found for " . $sql . "\n";

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "CREATE VIEW ActiveShoppingCarts AS

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, s.CreationDate, s.CreationTime

FROM Customer c

JOIN ShoppingCart s ON c.CustomerID = s.CustomerID;";

if (mysqli\_query($connect, $sql)) {

$result .= "View ActiveShoppingCarts Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "SELECT \* FROM ActiveShoppingCarts;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "No results found for " . $sql . "\n";

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "CREATE VIEW HighValueCustomers AS

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

LEFT JOIN OrderTable o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName, c.LastName

HAVING SUM(o.TotalPrice) > 100;";

if (mysqli\_query($connect, $sql)) {

$result .= "View HighValueCustomers Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "SELECT \* FROM HighValueCustomers;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "No results found for " . $sql . "\n";

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "CREATE VIEW HighValueProductCategories AS

SELECT pc.CategoryID, ct.CategoryName, SUM(oi.Quantity) AS TotalQuantitySold

FROM ProductCategory pc

JOIN OrderItem oi ON pc.ProductID = oi.ProductID

JOIN CategoryTable ct ON pc.CategoryID = ct.CategoryID

GROUP BY pc.CategoryID, ct.CategoryName

HAVING SUM(oi.Quantity) > 50;";

if (mysqli\_query($connect, $sql)) {

$result .= "View HighValueProductCategories Created\n";

} else {

$result .= "Error: " . $sql . "\n" . mysqli\_error($connect) . "\n";

}

$sql = "SELECT \* FROM HighValueProductCategories;";

$temp = mysqli\_query($connect, $sql);

if (mysqli\_num\_rows($temp) > 0) {

$count=1;

$result .= "Query: $sql\n";

$result .= "---------------------------------------------------------------------------------------\n\n";

while($row = mysqli\_fetch\_assoc($temp)) {

$result .= "Result " . $count . " ";

$result .= print\_r($row, true);

$result .= "\n";

$count = $count +1;

}

} else {

$result .= "No results found for " . $sql . "\n";

}

$result .= "---------------------------------------------------------------------------------------\n";

$sql = "DROP VIEW CustomerOrderDetails;";

mysqli\_query($connect, $sql);

$sql = "DROP VIEW PopularProducts;";

mysqli\_query($connect, $sql);

$sql = "DROP VIEW ActiveShoppingCarts;";

mysqli\_query($connect, $sql);

$sql = "DROP VIEW HighValueCustomers;";

mysqli\_query($connect, $sql);

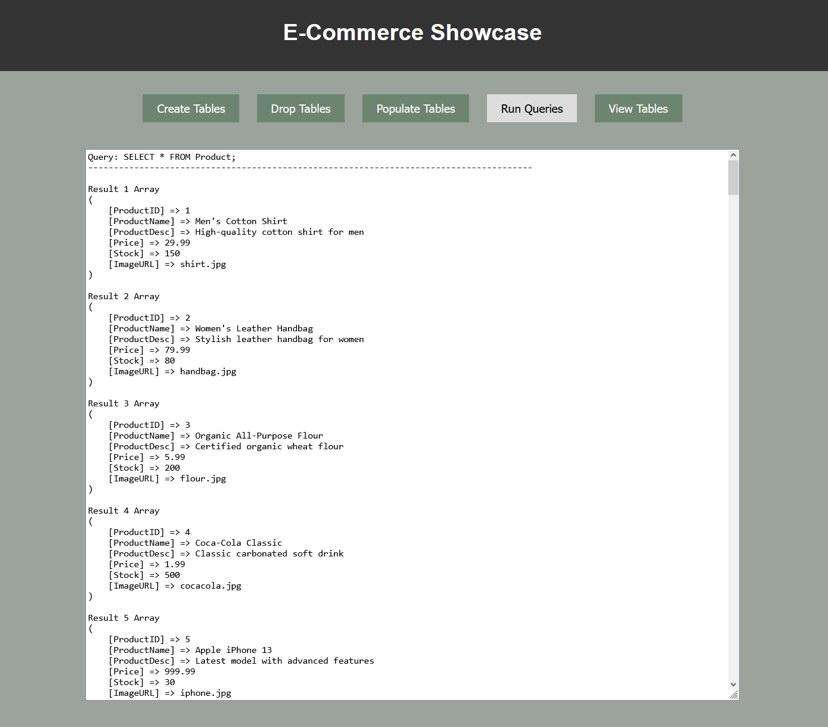
$sql = "DROP VIEW HighValueProductCategories;";

mysqli\_query($connect, $sql);

return $result;

}

?>

****

The figure above shows our GUI design run on the website.

**Product Usage Guide:**

To interact with our database system, access the graphical user interface (GUI) through the provided website link. Connecting to the school's Virtual Private Network (VPN) before navigating to the GUI is essential. Open your preferred web browser and enter the provided website link in the address bar. Ensure the school VPN is active before proceeding to the website and remains active throughout the process.

**Relational Algebra**

-- List all attributes of all products

SELECT \* FROM Product;

-- List Product Name and their Price for all products

SELECT 'The Product: ', ProductName, ' costs ', Price

FROM Product;

-- List the Unique First Name of all the customers

SELECT DISTINCT FirstName FROM Customer;

-- List all attributes of products that costs more than $100.0 or

-- have more than 200 items in stock.

SELECT \* FROM Product

WHERE Price > 100.0 OR Stock > 200;

-- List all the Subtotal of each product purchased by Customer1.

SELECT Subtotal AS Purchases\_by\_Customer1 FROM OrderItem

WHERE OrderID = 1;

-- List the Order ID and Total Price of all orders sorted by Order

-- Date in ascending order (Earliest to Latest).

SELECT OrderID, TotalPrice FROM OrderTable

ORDER BY OrderDate;

-- List the username and password of customer with an id greater than 3

SELECT username, passwd

FROM customer

WHERE customerid>3;

-- List the names of products that are popular (customers got more than one of)

SELECT productname

FROM product, cartproduct

Where quantity > 1

AND cartproduct.productid=product.productid;

-- List the customer id of customers that have spent less than $25 on their order

SELECT 'Customer ID is: ', customerid

FROM ordertable

WHERE totalprice < 25;

-- List the order IDs for orders that spent more than $50

SELECT orderid

FROM orderitem

WHERE subtotal>50;

-- List information of customers with an active shopping cart

SELECT c.\*

FROM customer c, shoppingcart s

WHERE c.customerid=s.customerid;

-- List all product ID's that are within the Clothing category

SELECT productid

FROM productcategory

WHERE categoryid='1';

-- List the category ID for the food category

SELECT categoryid

FROM categorytable

WHERE categoryname=’food’;

-- Count the number of products in each category.

SELECT CategoryName, COUNT(ProductCategoryID) AS NumberOfProducts

FROM CategoryTable

LEFT JOIN ProductCategory ON CategoryTable.CategoryID = ProductCategory.CategoryID

GROUP BY CategoryName;

-- List the total number of products in each shopping cart.

SELECT CustomerID, COUNT(ProductID) AS TotalProductsInCart

FROM ShoppingCart

LEFT JOIN CartProduct ON ShoppingCart.CustomerID = CartProduct.CustomerID

GROUP BY CustomerID;

-- List the products along with their quantities in the shopping cart of a specific customer.

-- In this query, we used tables aliases cp and p for a more advanced looking query

SELECT cp.CustomerID, p.ProductName, cp.Quantity

FROM CartProduct cp

JOIN Product p ON cp.ProductID = p.ProductID

WHERE cp.CustomerID = 1;

-- List products and their categories:

SELECT p.ProductName, ct.CategoryName

FROM Product p

JOIN ProductCategory pc ON p.ProductID = pc.ProductID

JOIN CategoryTable ct ON pc.CategoryID = ct.CategoryID;

-- List customers and their total spending:

SELECT c.CustomerID, c.FirstName || ' ' || c.LastName AS CustomerName, SUM(o.TotalPrice) AS TotalSpending

FROM Customer c

LEFT JOIN OrderTable o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName, c.LastName;

**Closing Remarks (Conclusion)**

Through subsequent labs, we delved into SQL queries' intricacies and normalization techniques. These efforts transformed our data-centric vision into a functional, user-friendly GUI accessible through a web interface. As we conclude this phase, our database system stands ready to serve its purpose efficiently.