Task:1. Database Design

1. Create the database named "TechShop"

=>CREATE DATABASE TechShop;

Use Techshop;

2. Define the schema for the Customers, Products, Orders, OrderDetails and Inventory tables based on the provided schema.

=>Customers Table

CREATE TABLE Customers(

CustomerId integer(5),

FirstName varchar(20),

LastName Varchar(20),

Email varchar(20) UNIQUE ,

Phone bigint(15) UNIQUE,

Address Varchar(20),

constraint customers\_customerId\_pk primary key(customerId)

);

A screen shot of a computer program

Description automatically generated

=>Products Table

CREATE TABLE Products(

ProductId integer(8),

ProductName varchar(20),

Description Varchar(50),

Price Decimal(20) ,

constraint products\_ProductId\_pk primary key(productId)

);

A black screen with white text

Description automatically generated

=>Orders Table

CREATE TABLE Orders(

OrderID integer(8),

CustomerId integer(5) ,

OrderDate Date,

TotalAmount Decimal(20),

constraint orders\_orderId\_pk primary key(orderId),

constraint orders\_customerId\_fk foreign key(customerId) references customers(customerId)

);

A black screen with white text

Description automatically generated

=>OrderDetails Table

CREATE TABLE OrderDetails(

OrderDetailID integer(10),

constraint ordersDetails\_OrderDetailID\_pk primary key(OrderDetailID),

OrderId integer(8),

constraint ordersDetails\_orderId\_fk foreign key(orderId) REFERENCES Orders(OrderID),

ProductID integer(8),

constraint ordersDetails\_productId\_fk foreign key(productId) REFERENCES Products(ProductId),

Quantity Integer(50)

);

A black screen with white text

Description automatically generated

=>Inventory Table

CREATE TABLE Inventory(

InventoryID integer(10),

constraint Inventory\_InventoryID\_pk Primary Key(InventoryID),

ProductID integer(8),

constraint Inventory\_productId\_fk foreign key(productId) REFERENCES Products(ProductID),

QuantityInStock integer(100),

LastStockUpdate Date

);

A black screen with white text

Description automatically generated

5. Insert at least 10 sample records into each of the following tables.

a. Customer :

insert into customers values(101,'Arav','Josh','arav@gmailcom',9876543212,'Bangalore');

insert into customers values(102,'Harsh','Singh','harsh@gmailcom',9865433276,'Hyderbad');

insert into customers values(103,'Siya','Gupta','siya@gmailcom',9766578654,'Mumbai');

insert into customers values(104,'Siri','Kumar','siri@gmailcom',9776589076,'Pune');

insert into customers values(105,'Sam','sharma','sam@gmailcom',9765432178,'Bangalore');

insert into customers values(106,'Dhruv','Patel','dhruv@gmailcom',7786543289,'chennai');

insert into customers values(107,'Manya','Agarwal','manya@gmailcom',7869540091,'lucknow');

insert into customers values(108,'Manoj','Jain','manoj@gmailcom',6789543005,'Goa');

insert into customers values(109,'Gagana','Joshi','gagana@gmailcom',8765455900,'Mysore');

insert into customers values(110,'Amulya','Bhatt','amulya@gmailcom',7865443109,'Indore');



b.Products :

insert into products values(1,'Phone','samsung',100000);

insert into products values(2,'Laptop','Lenovo',500000);

insert into products values(3,'TV','Sony',300000);

insert into products values(4,'Computer','Bosh',200000);

insert into products values(5,'Speaker','Echo',20000);

insert into products values(6,'Washing Machine','LG',70000);

insert into products values(7,'Refrigerator','IFB',90000);

insert into products values(8,'Bulb','panasonic',900);

insert into products values(9,'Camera','canon',80000);

insert into products values(10,'Tablet','apple',77000);

A screen shot of a computer

Description automatically generated

c.Orders :

insert into orders values(201,102,'2001-10-23',40000);

insert into orders values(202,108,'2008-11-03',7000);

insert into orders values(203,101,'1998-01-13',90000);

insert into orders values(204,103,'1995-12-05',88000);

insert into orders values(205,110,'2000-07-08',67000);

insert into orders values(206,109,'2008-05-18',68090);

insert into orders values(207,105,'2018-09-28',98050);

insert into orders values(208,104,'2009-06-16',8790);

insert into orders values(209,106,'2015-04-18',7899);

insert into orders values(210,107,'2019-08-10',77777);

A black screen with white text

Description automatically generated

c.Ordersdetails :

insert into orderdetails values(1001,202,4,2);

insert into orderdetails values(1002,205,3,3);

insert into orderdetails values(1003,201,1,5);

insert into orderdetails values(1004,210,10,2);

insert into orderdetails values(1005,209,9,1);

insert into orderdetails values(1006,203,2,4);

insert into orderdetails values(1007,204,5,5);

insert into orderdetails values(1008,206,7,1);

insert into orderdetails values(1009,207,6,2);

insert into orderdetails values(1010,208,8,6);

A screen shot of a black screen

Description automatically generated

d.Inventory :

insert into inventory values (2001,3,10, ‘2024-01-18’);

insert into inventory values (2002,2,15 , ‘2023-12-28’);

insert into inventory values (2003,1,18 , ‘2023-11-12’);

insert into inventory values (2004,10,12 , ‘2024-03-10’);

insert into inventory values (2005,9,11, ‘2023-12-12’);

insert into inventory values (2006,4,10 , ‘2022-10-20’);

insert into inventory values (2007,5,16 , ‘2022-07-29’);

insert into inventory values (2008,6,8 , ‘2022-04-27’);

insert into inventory values (2009,7,17 , ‘2023-08-22’);

insert into inventory values (2010,8,13 , ‘2024-02-04’);

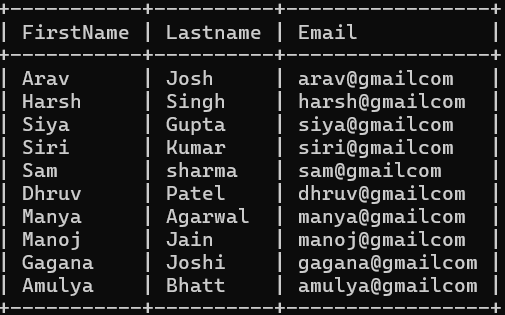
A screen shot of a computer

Description automatically generated

Tasks 2: Select, Where, Between, AND, LIKE:

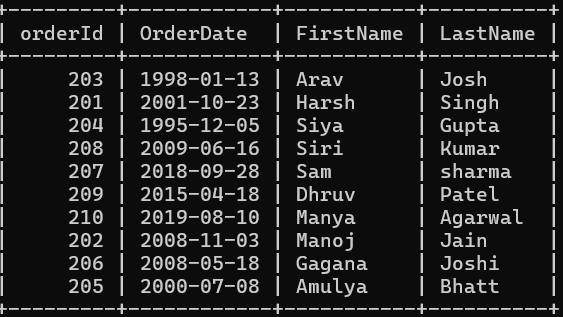
1. Write an SQL query to retrieve the names and emails of all customers.

->select FirstName, Lastname, Email from customers;



1. Write an SQL query to list all orders with their order dates and corresponding customer names.

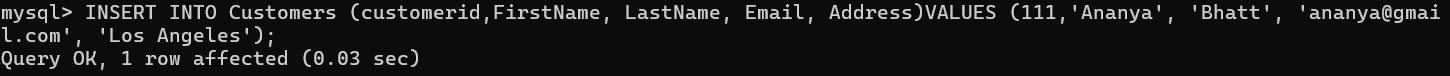
-> select o.orderId ,o.OrderDate ,c.FirstName,c.LastName from orders o ,customers c where o.customerid=c.customerid;



1. Write an SQL query to insert a new customer record into the "Customers" table. Include customer information such as name, email, and address.

-> INSERT INTO Customers (customerid,FirstName, LastName, Email, Address)VALUES (111,'Ananya', 'Bhatt', 'ananya@gmai

l.com', 'Los Angeles');



1. Write an SQL query to update the prices of all electronic gadgets in the "Products" table by increasing them by 10%.

->UPDATE Products

-> SET Price = Price \* 1.10;

select \* from products;

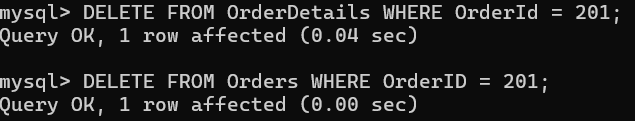
A screen shot of a computer

Description automatically generated

1. Write an SQL query to delete a specific order and its associated order details from the "Orders" and "OrderDetails" tables. Allow users to input the order ID as a parameter.

-> DELETE FROM OrderDetails WHERE OrderId = 201;

-> DELETE FROM Orders WHERE OrderID = 201;



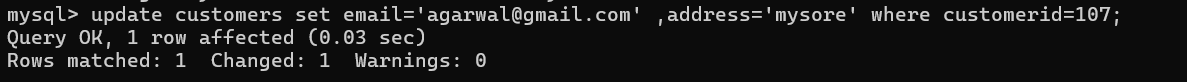
1. Write an SQL query to insert a new order into the "Orders" table. Include the customer ID, order date, and any other necessary information.

->INSERT INTO Orders values(211,109,'2023-10-25',29000);



1. Write an SQL query to update the contact information (e.g., email and address) of a specific customer in the "Customers" table. Allow users to input the customer ID and new contact information.

-> update customers set email='agarwal@gmail.com' ,address='mysore' where customerid=107;



A screen shot of a computer screen

Description automatically generated

1. Write an SQL query to recalculate and update the total cost of each order in the "Orders" table based on the prices and quantities in the "OrderDetails" table.

UPDATE Orders

-> SET TotalAmount = (

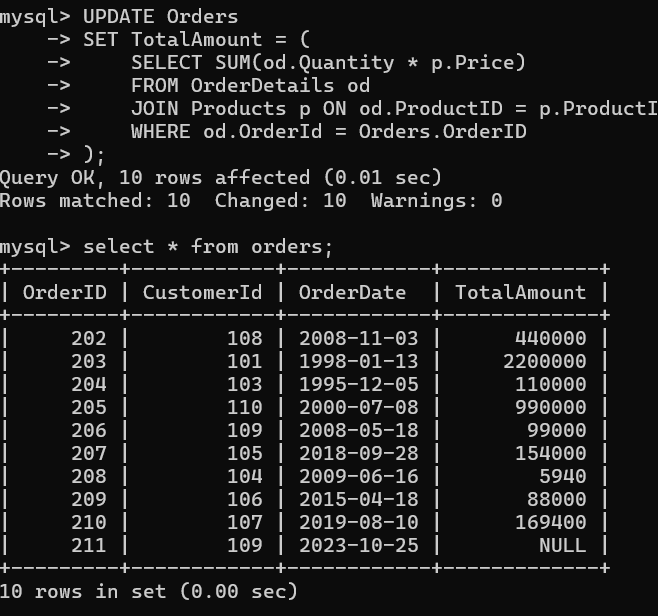
-> SELECT SUM(od.Quantity \* p.Price)

-> FROM OrderDetails od

-> JOIN Products p ON od.ProductID = p.ProductId

-> WHERE od.OrderId = Orders.OrderID

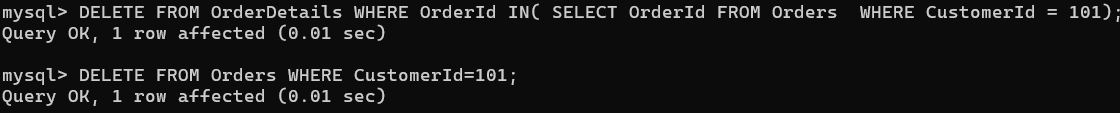
-> );



1. Write an SQL query to delete all orders and their associated order details for a specific customer from the "Orders" and "OrderDetails" tables. Allow users to input the customer ID as a parameter.

->DELETE FROM OrderDetails WHERE OrderId IN( SELECT OrderId FROM Orders WHERE CustomerId = 101);

->DELETE FROM Orders WHERE CustomerId=101;



1. Write an SQL query to insert a new electronic gadget product into the "Products" table, including product name, category, price, and any other relevant details.

-> INSERT INTO Products VALUES (11,'Smartwatch', 'boat', 25000);



1. Write an SQL query to update the status of a specific order in the "Orders" table (e.g., from "Pending" to "Shipped"). Allow users to input the order ID and the new status.

-> ALTER TABLE Orders ADD status VARCHAR(20);

-> UPDATE Orders SET status = 'pending' WHERE OrderID =209;

-> UPDATE Orders SET status = 'shipped' WHERE OrderID =209;

A screenshot of a computer screen

Description automatically generated

1. Write an SQL query to calculate and update the number of orders placed by each customer in the "Customers" table based on the data in the "Orders" table.

-> ALTER TABLE Customers

-> ADD OrderCount INT DEFAULT 0;

->UPDATE Customers AS c

-> SET OrderCount = (

-> SELECT COUNT(DISTINCT od.OrderId)

-> FROM OrderDetails AS od

-> JOIN Orders AS o ON od.OrderId = o.OrderId

-> WHERE o.CustomerId = c.CustomerId

-> );

A screenshot of a computer screen

Description automatically generated

Task 3. Aggregate functions, Having, Order By, GroupBy and Joins:

1. Write an SQL query to retrieve a list of all orders along with customer information (e.g., customer name) for each order.

-> select o.orderId ,o.orderDate , o.totalamount ,c.FirstName ,c.LastName,c.email from orders o inner join customers c on o.customerId=c.customerId;

A screenshot of a computer screen

Description automatically generated

1. Write an SQL query to find the total revenue generated by each electronic gadget product. Include the product name and the total revenue.

-> SELECT p.productName, SUM(od.quantity \* p.Price) AS total\_revenue

-> FROM products AS p

-> INNER JOIN orderdetails AS od ON p.productId = od.productId

-> GROUP BY p.productId, p.productName

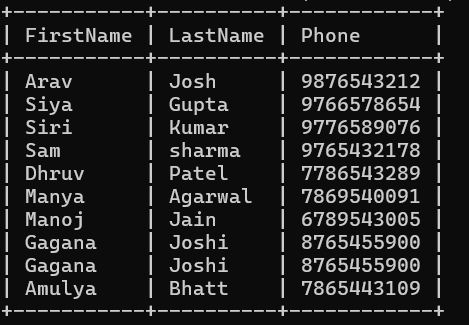
-> ORDER BY total\_revenue DESC;

A screen shot of a computer program

Description automatically generated

1. Write an SQL query to list all customers who have made at least one purchase. Include their names and contact information.

-> SELECT c.FirstName, c.LastName, c.Phone FROM Customers c INNER JOIN Orders o ON c.CustomerId = o.CustomerId;



1. Write an SQL query to find the most popular electronic gadget, which is the one with the highest total quantity ordered. Include the product name and the total quantity ordered.

-> select p.productname, od.quantity from products as p inner join orderdetails as od where p.productid=od.productid order by od.quantity desc limit 1;

A black screen with white text

Description automatically generated

1. Write an SQL query to retrieve a list of electronic gadgets along with their corresponding categories.

-> alter table products add(categories varchar(20));

update products set categories='communication' where productId=1;

update products set categories='microcomputers' where productId=2;

update products set categories='telecommunication' where productId=3;

update products set categories='microcomputer' where productId=4;

update products set categories='speech' where productId=5;

update products set categories='consumer' where productId=6;

update products set categories='consumer' where productId=7;

update products set categories='homeappliance' where productId=8;

update products set categories='images' where productId=9;

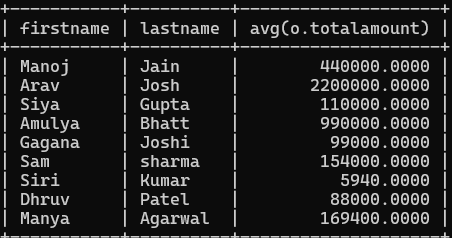
-> select productname ,categories from products;

A screen shot of a computer program

Description automatically generated

1. Write an SQL query to calculate the average order value for each customer. Include the customer's name and their average order value.

-> select c.firstname, c.lastname, avg(o.totalamount) from customers c inner join orders o on c.customerId=o.customerId group by c.firstname, c.lastname;



1. Write an SQL query to find the order with the highest total revenue. Include the order ID, customer information, and the total revenue.

-> SELECT o.OrderID, c.FirstName, c.LastName, SUM(od.Quantity \* p.Price) as totalrevenue FROM Orders AS o INNER JOIN Customers AS c ON o.CustomerId = c.CustomerId INNER JOIN OrderDetails AS od ON o.OrderID = od.OrderId INNER JOIN Products AS p ON od.ProductID = p.Pro

ductId GROUP BY o.OrderID, c.FirstName, c.LastName ORDER BY totalrevenue desc limit 1;

A black screen with white text

Description automatically generated

1. Write an SQL query to list electronic gadgets and the number of times each product has been ordered.

-> select p.productname ,od.quantity from products as p inner join orderdetails as od on p.productid=od.productid ;

A screen shot of a computer

Description automatically generated

1. Write an SQL query to find customers who have purchased a specific electronic gadget product. Allow users to input the product name as a parameter.

-> SELECT c.FirstName, c.LastName, c.Email

-> FROM Customers AS c JOIN Orders AS o ON c.CustomerId = o.CustomerId

-> JOIN OrderDetails AS od ON o.OrderID = od.OrderID

-> JOIN Products AS p ON od.ProductID = p.ProductID

-> WHERE p.ProductName = 'camera';

A screen shot of a black background

Description automatically generated

1. Write an SQL query to calculate the total revenue generated by all orders placed within a specific time period. Allow users to input the start and end dates as parameter.

-> SELECT SUM(o.TotalAmount) AS TotalRevenue

-> FROM Orders AS o

-> WHERE o.OrderDate BETWEEN '2019-08-10' AND '2022-08-29';

A screenshot of a computer screen

Description automatically generated

Task 4. Subquery and its type:

1.Write an SQL query to find out which customers have not placed any orders.

->SELECT c.CustomerId, c.FirstName, c.LastName

-> FROM Customers AS c

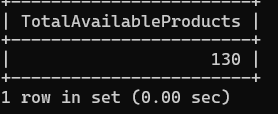
-> WHERE c.CustomerId NOT IN (SELECT o.CustomerId FROM Orders AS o);

A black and white screen with white text

Description automatically generated

2. Write an SQL query to find the total number of products available for sale.

->select sum(QuantityInStock) from inventory;



3. Write an SQL query to calculate the total revenue generated by TechShop.

-> select sum(totalamount) as revenue from orders;

A black background with white text

Description automatically generated

4. Write an SQL query to calculate the average quantity ordered for products in a specific category. Allow users to input the category name as a parameter.

-> SELECT p.ProductName, AVG(od.Quantity)

-> FROM OrderDetails AS od

-> INNER JOIN Products AS p ON od.ProductID = p.ProductId

-> GROUP BY p.ProductName;

A black screen with white text

Description automatically generated

5. Write an SQL query to calculate the total revenue generated by a specific customer. Allow users to input the customer ID as a parameter.

->SELECT o.CustomerId, SUM(od.Quantity \* p.Price) as revenue

-> FROM Orders AS o INNER JOIN OrderDetails AS od ON o.OrderID = od.OrderId

-> INNER JOIN Products AS p ON od.ProductID = p.ProductId

-> GROUP BY o.CustomerId;

A screenshot of a black screen

Description automatically generated

6. Write an SQL query to find the customers who have placed the most orders. List their names and the number of orders they've placed.

-> SELECT c.firstName,c.lastName,COUNT(o.orderId) AS order\_count

-> FROM customers AS c INNER JOIN orders AS o ON c.customerId = o.customerId

-> GROUP BY c.customerId, c.firstName, c.lastName

-> ORDER BY order\_count DESC;



7. Write an SQL query to find the most popular product category, which is the one with the highest total quantity ordered across all orders.

->SELECT c.firstName,c.lastName,p.productName,SUM(od.quantity) AS total\_quantity\_ordered

-> FROM customers AS c INNER JOIN orders AS o ON c.customerId = o.customerId

-> INNER JOIN orderdetails AS od ON o.orderId = od.orderId

-> INNER JOIN products AS p ON od.productId = p.productId

-> GROUP BY c.customerId, c.firstName, c.lastName, p.productName

-> ORDER BY total\_quantity\_ordered DESC limit 1;

A black screen with white text

Description automatically generated

8. Write an SQL query to find the customer who has spent the most money (highest total revenue) on electronic gadgets. List their name and total spending.

->SELECT c.firstName, c.lastName, SUM(od.quantity \* p.price) AS total\_spending

-> FROM customers AS c

-> INNER JOIN orders AS o ON c.customerId = o.customerId

-> INNER JOIN orderdetails AS od ON o.orderId = od.orderId

-> INNER JOIN products AS p ON od.productId = p.productId

-> GROUP BY c.customerId

-> ORDER BY total\_spending DESC

-> LIMIT 1;

A black screen with white text

Description automatically generated

9. Write an SQL query to calculate the average order value (total revenue divided by the number of orders) for all customers.

->SELECT AVG(order\_value) AS average\_order\_value FROM (

-> SELECT o.customerId,SUM(od.quantity \* p.price) AS order\_value

-> FROM orders AS o INNER JOIN orderdetails AS od ON o.orderId = od.orderId

-> INNER JOIN products AS p ON od.productId = p.productId

-> GROUP BY o.orderId)as order\_summary;

A black screen with white text

Description automatically generated

10. Write an SQL query to find the total number of orders placed by each customer and list their names along with the order count.

->SELECT c.firstName, COUNT(o.orderId) AS total\_orders

-> FROM customers AS c

-> LEFT JOIN orders AS o ON c.customerId = o.customerId

-> GROUP BY c.firstName;

A screen shot of a black and white screen

Description automatically generated