ASSIGNMENT

TECHSHOP

(An Electronic Gadgets Shop)

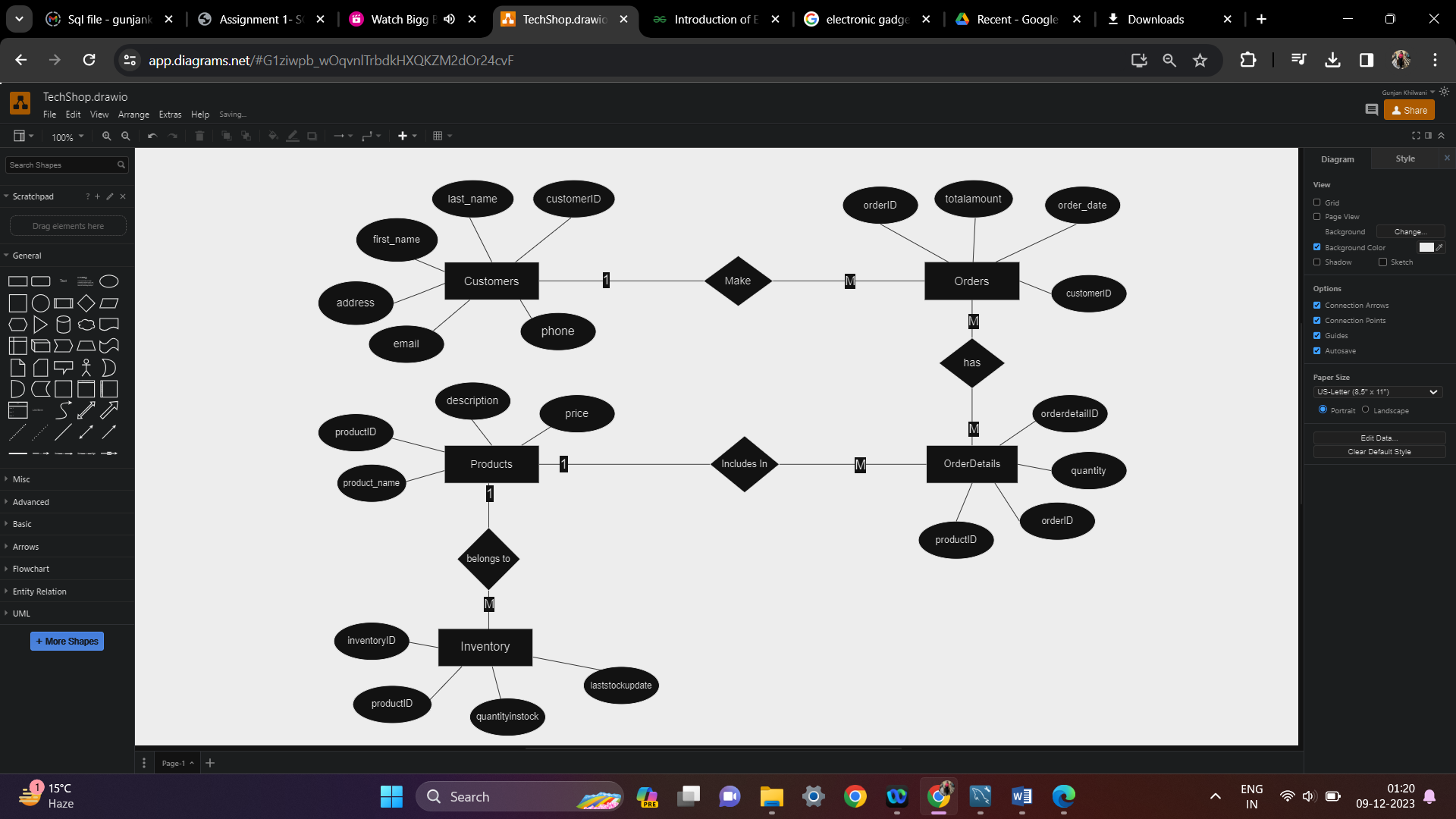
**Task 1: Database Design (Normalization)**

* **Create the database named "TechShop"**

CREATE DATABASE TechShop;

USE TechShop;

* **Create an ERD (Entity Relationship Diagram) for the database.**



* **DATA DEFINITION LANGUAGE (DDL): CREATING TABLES**

CREATE TABLE Customers (

customerID INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

first\_name VARCHAR(20),

last\_name VARCHAR(20),

email VARCHAR(100),

phone VARCHAR(10),

address VARCHAR(255)

);

CREATE TABLE Categories (

categoryID INT PRIMARY KEY,

category\_name VARCHAR(120) NOT NULL

);

CREATE TABLE Products(

productID INT PRIMARY KEY,

categoryID INT,

product\_name VARCHAR(50),

price DECIMAL(10, 2),

FOREIGN KEY (categoryID) REFERENCES Categories(categoryID)

);

CREATE TABLE Orders(

orderID INT PRIMARY KEY,

customerID INT,

order\_date DATE,

order\_status VARCHAR(50),

totalamount DECIMAL(10, 2),

FOREIGN KEY(customerID) REFERENCES Customers(customerID) ON DELETE SET NULL );

CREATE TABLE OrderDetails(

orderdetailID INT PRIMARY KEY,

orderID INT,

productID INT,

quantity INT,

FOREIGN KEY(orderID) REFERENCES Orders(orderID) ON DELETE CASCADE, FOREIGN KEY(productID) REFERENCES Products(productID) ON DELETE SET NULL

);

CREATE TABLE Inventory(

inventoryID INT,

productID INT,

quantityinstock INT,

laststockupdate DATE,

FOREIGN KEY(productID) REFERENCES Products(productID) ON DELETE CASCADE

);

* **DATA MANIPULATION LANGUAGE (DML):**

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

INSERT INTO Customers

(customerID, first\_name, last\_name, email, phone, address) VALUES

(01, 'Monika', 'Arora', 'Mon01@gmail.com', '8796543987', 'Gurgaon'),

(02, 'Niharika', 'Verma', 'nihv6@gmail.com', '7645689076', 'Delhi'),

(03, 'Vishal', 'Singhal', 'singhalvishal@gmail.com', '9998765439', 'Mumbai'),

(04, 'Amitabh', 'Singh', 'singha89@gmail.com', '9876098765', 'Chennai'),

(05, 'Vivek', 'Bhati', 'bhativivek@gmail.com', '8976897654', 'Banglore'),

(06, 'Vipul', 'Diwan', 'vipul786@gmail.com', '7999865498', 'Chennai'),

(07, 'Kapil', 'Sharma', 'kapilsh34@gmail.com', '9876123567', 'Pune'),

(08, 'Ronak', 'Kapoor', 'ronakapoor@gmail.com', '9087789879', 'Pune'),

(09, 'Rashmi', 'Singh', 'rashmisingh@gmail.com', '8659098787', 'Delhi'),

(10, 'Akash', 'Agrawal', 'akash00@gmail.com', '8659876543', 'Hyderabad');

INSERT INTO Categories (categoryID, category\_name) VALUES

(01, 'Smart Phone'),

(02, 'Phone & Laptop Accessories'),

(03, 'Home Appliances'),

(04, 'Home Electronics'),

(05, 'IOS Phone'),

(06, 'Wireless & Wired Headphones'),

(07, 'Smart TV'),

(08, 'Speaker'),

(09, 'Smart Home'),

(10, 'Optical Instruments'),

(11, 'Computers');

INSERT INTO Products (productID, categoryID, product\_name, price) VALUES

(01, 08, 'DVD Player', 5997),

(02, 06, 'Headphones', 1599),

(03, 10, 'Camera', 7999),

(04, 02, 'Wireless Mouse', 499),

(05, 02, 'Mouse pad', 299),

(06, 11, 'Laptop', 60000),

(07, 07, 'LCD TV', 30000),

(8, 03, 'Heater', 4999),

(9, 02, 'C type charger', 799),

(10, 03, 'Washing Machine', 19000);

INSERT INTO Orders (orderid, order\_date, order\_status, customerID, totalamount) VALUES

(01, '2023-02-08', 'pending', 04, 0),

(02, '2022-08-12', 'shipped', 07, 0),

(03, '2023-09-09', 'pending', 05, 0),

(04, '2022-10-04', 'shipped', 04, 0),

(05, '2023-08-27', 'shipped', 03, 0),

(06, '2023-10-04', 'shipped', 06, 0),

(07, '2022-01-28', 'pending', 07, 0),

(08, '2023-07-05', 'pending', 01, 0),

(09, '2023-09-11', 'pending', 05, 0),

(10, '2022-10-29', 'pending', 02, 0);

INSERT INTO OrderDetails (orderdetailID, orderID, productID, quantity) VALUES

(01, 05, 05, 04),

(02, 06, 06, 08),

(03, 07, 08, 05),

(04, 02, 04, 07),

(05, 09, 02, 01),

(06, 08, 10, 05),

(07, 09, 03, 08),

(08, 09, 06, 03),

(09, 03, 10, 06),

(10, 07, 01, 07);

INSERT INTO Inventory (inventoryID, productID, quantityinstock, laststockupdate) VALUES

(01, 08, 101, '2023-04-01'),

(02, 09, 201, '2023-05-16'),

(03, 01, 301, '2023-08-17'),

(04, 07, 401, '2023-09-18'),

(05, 10, 51, '2023-10-04'),

(06, 02, 81, '2022-05-20'),

(07, 06, 91, '2023-02-27'),

(08, 03, 181, '2023-08-08'),

(09, 05, 71, '2022-02-09'),

(10, 04, 901, '2023-10-27');

**Task 2 : SELECT, WHERE, BETWEEN AND, LIKE**

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

SELECT CONCAT(first\_name, ' ', last\_name) AS name, email FROM Customers;

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

SELECT orderid, order\_date, CONCAT(first\_name, last\_name) AS Name

FROM Orders, Customers

WHERE Customers.customerid = Orders.customerid;

**3. 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 VALUES

(11, 'Neeraj', 'Jain','Jaineeraj@gmail.com', '9876549824', 'Gurgaon');

**4. 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+(price\*0.1) ;

**5. 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.**

SET @id = 10;

DELETE FROM Orders WHERE orderID = @id;

**6. 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(orderid, order\_date, order\_status, customerid, totalamount) VALUES

(12, '2023-02-14', 'shipped', 09, 2000);

**7. 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.**

SET @customerid = 02;

SET @address = 'Hyderabad';

SET @email = 'niharikaverma@gmail.com';

UPDATE customers SET email = @email , address = @address WHERE customerID = @customerid;

**8. 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 O, Orderdetails D

SET totalamount = (

SELECT SUM(price\*quantity) FROM Orderdetails D

JOIN Products P

ON P.productID = D.productID

WHERE D.orderID = O.orderID);

**9. 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.**

SET @customerid = 04;

DELETE FROM Orders WHERE customerID = @customerid;

**10. 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 (productID, categoryID, product\_name, price) VALUES

(14, 02, 'Earphone', 399);

**11. 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.**

SET @orderid = 01;

SET @status = 'SHIPPED';

UPDATE Orders SET status = @status WHERE orderID = @orderid;

**12. 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;

UPDATE Customers

SET ordercount = (

SELECT COUNT(orderID) FROM Orders

WHERE Customers.customerID = Orders.customerID );

**TASK 3 : AGGREGATE FUNCTIONS, HAVING, ORDER BY, GROUP BY 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 C.first\_name, C.last\_name, O.\* FROM ORDERS O JOIN CUSTOMERS C

ON O.customerID = C.customerID;

**2. 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.productid, product\_name, price\*SUM(quantity) AS TotalRevenue

FROM Products P

JOIN Orderdetails D

ON P.productID = D.productID

GROUP BY productID;

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

SELECT C.customerID, C.first\_name, C.last\_name, C.email

FROM CUSTOMERS C

JOIN ORDERS O

ON C.customerID = O.customerID

GROUP BY C.customerID;

**4. 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.product\_name, SUM(D.quantity) AS Units\_sold

FROM PRODUCTS P

JOIN ORDERDETAILS D

ON P.productID = D.productID

GROUP BY P.productID

ORDER BY Units\_sold DESC LIMIT 1;

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

SELECT P.product\_name, C.category\_name

FROM Products P

JOIN Categories C

ON P.categoryID = C.categoryID;

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

SELECT CONCAT(C.first\_name,' ',C.last\_name) CustName,

AVG(O.totalamount) AS AvgOrderValue

FROM Customers C JOIN Orders O

ON C.customerID = O.customerID

GROUP BY O.customerID;

**7. 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, first\_name, last\_name, address, O.totalamount AS TotalRevenue

FROM ORDERS O JOIN CUSTOMERS C

ON O.customerID = C.customerID

ORDER BY TotalRevenue DESC LIMIT 1;

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

SELECT P.productID, P.product\_name, COUNT(D.orderDetailID) AS TotalOrders

FROM Products P JOIN OrderDetails D ON P.productID = D.productID

GROUP BY productID;

**9. 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.**

SET @product = 'DVD player';

SELECT C.first\_name, C.last\_name

FROM customers C

JOIN Orders O ON C.customerID = O.customerID

JOIN OrderDetails D ON D.orderID = O.orderID

JOIN Products P on P.productID = D.productID

WHERE product\_name = @product;

**10. 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 parameters.**

SET @startDate = '2023-01-07';

SET @endDate = '2023-11-20';

SELECT SUM(totalamount) AS TotalRevenue

FROM ORDERS O

JOIN ORDERDETAILS D ON O.orderID = D.orderID

WHERE order\_date BETWEEN @startDate AND @endDate;

**TASK 4: SUBQUERY AND ITS TYPES**

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

SELECT \* FROM Customers WHERE customerID NOT IN (

SELECT distinct(customerID) FROM ORDERS

);

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

SELECT COUNT(\*) as Total\_Products from INVENTORY WHERE QuantityInStock>0;

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

SELECT SUM(totalamount) as "Total Revenue" FROM ORDERS;

**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.**

SET @cate\_name = 'Phone & Laptop Accessories';

SELECT AVG(quantity) FROM OrderDetails

WHERE productid IN (SELECT productid FROM Products WHERE categoryid = (

SELECT categoryID FROM Categories WHERE category\_name = @cate\_name)

);

**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.**

SET @customerid = 03;

SELECT customerID, SUM(totalamount)

FROM orders

WHERE customerID=@customerid

GROUP BY customerID;

**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

(SELECT CONCAT(first\_name,' ',last\_name) FROM Customers

WHERE Orders.customerid=customerid) AS

"Customer Name", COUNT(orderid) as "Total Orders"

FROM orders GROUP BY customerid ORDER BY count(\*) DESC LIMIT 1;

**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.category\_name, SUM(quantity) AS TotalQuantity FROM OrderDetails D

JOIN Products P ON P.productID = D.productID

JOIN Categories C ON C.categoryID = P.categoryID

GROUP BY C.categoryID

ORDER BY SUM(quantity) DESC LIMIT 1;

**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 CustomerID,

(select concat(first\_name,' ',last\_name) FROM Customers

WHERE customerID = O.customerID) AS "Customer Name",

(SELECT SUM(totalamount) FROM Orders

WHERE customerID = O.CustomerID) as "Total Spending"

FROM Orders O

WHERE customerID IN (

SELECT customerID FROM OrderDetails

)

GROUP BY customerID

ORDER BY "Total Spending" DESC limit 1;

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

SELECT customerID,

(SELECT CONCAT(first\_name," ",last\_name) FROM Customers C

WHERE C.customerid = O.customerid) AS 'Customer Name',

SUM(totalamount)/COUNT(orderid) AS "Avg Order Value"

FROM Orders O

GROUP BY customerid;

**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 customerID,

(SELECT CONCAT(first\_name,' ', last\_name) FROM Customers C

WHERE C.customerid=O.customerid) AS 'Customer Name', COUNT(orderid) AS 'Order Count' FROM orders O GROUP BY customerid;