ASSIGNMENT – SQL – ELECTRONIC GADGETS

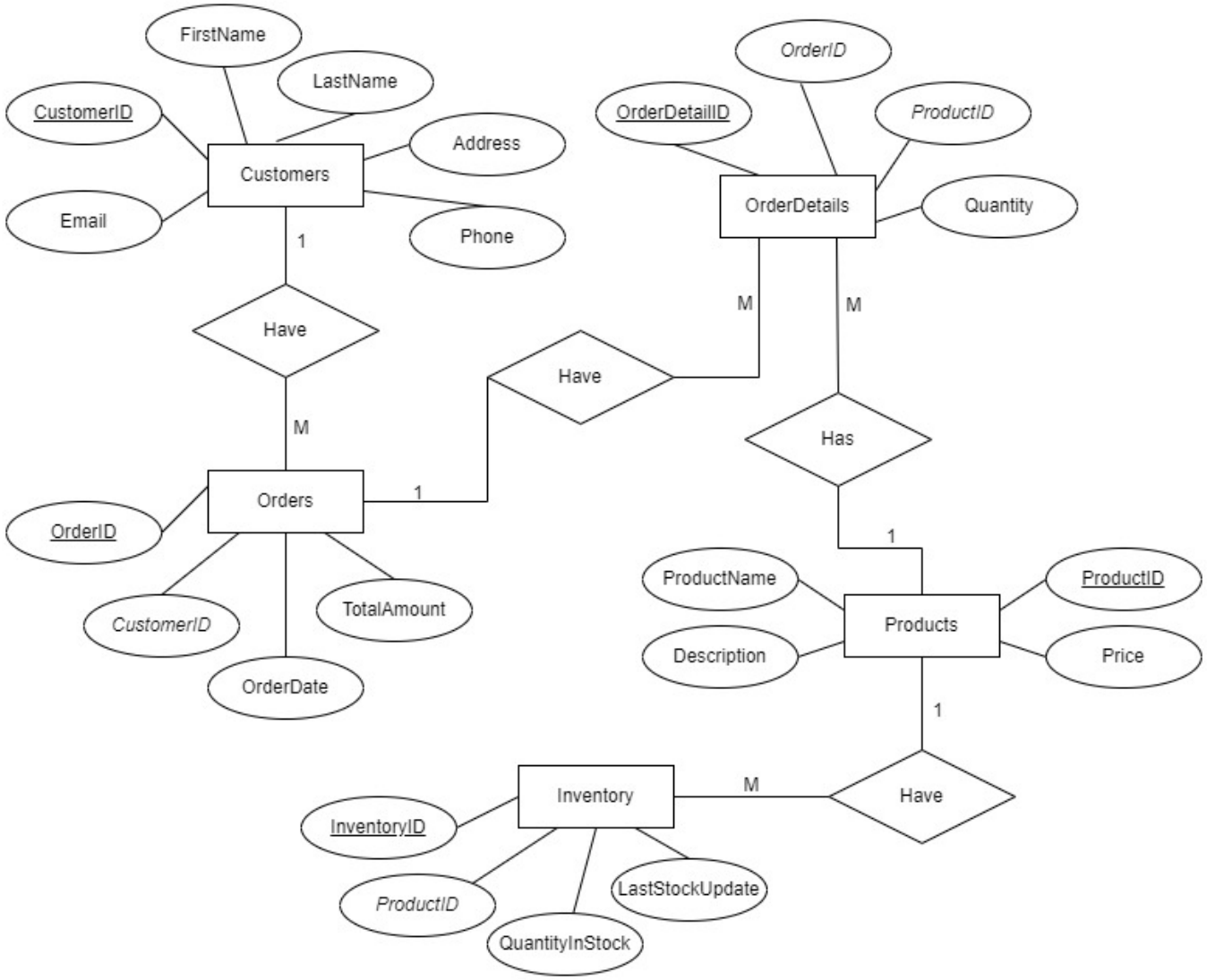
NIKHIL AGARWAL –
NIKHILAG113@GMAIL.COM

Task:1. Database Design:

```
-- 1. Create the database named "TechShop":

        create database techshop;


-- using techshop database for further storage
use techshop;


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


CREATE TABLE Customers (
    CustomerID INT AUTO_INCREMENT PRIMARY KEY,
    FirstName VARCHAR(255),
    LastName VARCHAR(255),
    Email VARCHAR(255),
    Phone VARCHAR(15),
    Address VARCHAR(255)
);


CREATE TABLE Products (
    ProductID INT AUTO_INCREMENT PRIMARY KEY,
    ProductName VARCHAR(255),
    Description TEXT,
    Price DECIMAL(10, 2)
);
```

```
CREATE TABLE Orders (
    OrderID INT AUTO_INCREMENT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE,
    TotalAmount DECIMAL(10, 2),
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);

CREATE TABLE OrderDetails (
    OrderDetailID INT AUTO_INCREMENT PRIMARY KEY,
    OrderID INT,
    ProductID INT,
    Quantity INT,
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);

CREATE TABLE Inventory (
    InventoryID INT AUTO_INCREMENT PRIMARY KEY,
    ProductID INT,
    QuantityInStock INT,
    LastStockUpdate DATE,
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
```

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



-- 4. Create appropriate Primary Key and Foreign Key constraints for referential integrity

(Primary Key and Foreign Key constraints are already included in the initial SQL statements)

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

- Customers
- Products
- Orders
- OrderDetails
- Inventory

-- Insert sample records into the Customers table

```
INSERT INTO Customers (FirstName, LastName, Email, Phone, Address)
VALUES
('John', 'Doe', 'john.doe@example.com', '123-456-7890', '123 Main St'),
('Jane', 'Smith', 'jane.smith@example.com', '987-654-3210', '456 Oak St'),
('Bob', 'Johnson', 'bob.johnson@example.com', '111-222-3333', '789 Pine St'),
('Alice', 'Williams', 'alice.williams@example.com', '444-555-6666', '101 Elm St'),
('Charlie', 'Brown', 'charlie.brown@example.com', '777-888-9999', '202 Maple St'),
('Eva', 'Garcia', 'eva.garcia@example.com', '333-444-5555', '303 Birch St'),
('David', 'Lee', 'david.lee@example.com', '666-777-8888', '404 Cedar St'),
('Grace', 'Jones', 'grace.jones@example.com', '222-333-4444', '505 Walnut St'),
('Michael', 'Miller', 'michael.miller@example.com', '555-666-7777', '606 Pine St'),
('Sophia', 'Davis', 'sophia.davis@example.com', '888-999-0000', '707 Oak St');
```

```
-- Insert sample records into the Products table
INSERT INTO Products (ProductName, Description, Price)
VALUES
('Laptop', 'High-performance laptop with latest features', 899.99),
('Smartphone', 'Top-of-the-line smartphone with advanced camera', 699.99),
('Tablet', 'Lightweight and portable tablet for on-the-go use', 299.99),
('Headphones', 'Noise-canceling headphones with premium sound quality', 149.99),
('Smartwatch', 'Smartwatch with fitness tracking and notifications', 199.99),
('Camera', 'Professional-grade camera for photography enthusiasts', 1299.99),
('Drone', 'Quadcopter drone with HD camera for aerial photography', 799.99),
('Bluetooth Speaker', 'Portable Bluetooth speaker for wireless audio streaming', 79.99),
('Gaming Console', 'High-performance gaming console for immersive gaming', 499.99),
('Fitness Tracker', 'Fitness tracker for monitoring health and activity', 129.99);

-- Insert sample records into the Orders table
INSERT INTO Orders (CustomerID, OrderDate, TotalAmount)
VALUES
(1, '2023-01-15', 899.99),
(3, '2023-02-20', 299.99),
(5, '2023-03-10', 149.99),
(2, '2023-04-05', 199.99),
(7, '2023-05-18', 1299.99),
(4, '2023-06-22', 799.99),
(6, '2023-07-08', 79.99),
(8, '2023-08-30', 499.99),
(10, '2023-09-12', 129.99),
(9, '2023-10-25', 699.99);
```

```
-- Insert sample records into the OrderDetails table
INSERT INTO OrderDetails (OrderID, ProductID, Quantity)
VALUES
(1, 1, 1),
(2, 3, 2),
(3, 4, 3),
(4, 5, 1),
(5, 6, 1),
(6, 7, 1),
(7, 8, 2),
(8, 9, 1),
(9, 2, 1),
(10, 10, 2);

-- Insert sample records into the Inventory table
INSERT INTO Inventory (ProductID, QuantityInStock, LastStockUpdate)
VALUES
(1, 20, '2023-01-01'),
(2, 15, '2023-01-05'),
(3, 30, '2023-01-10'),
(4, 25, '2023-01-15'),
(5, 10, '2023-01-20'),
(6, 8, '2023-01-25'),
(7, 12, '2023-02-01'),
(8, 18, '2023-02-05'),
(9, 5, '2023-02-10'),
(10, 22, '2023-02-15');
```

-- Task 2: Select, Where, Between, AND, LIKE

```
-- 1. Retrieve the names and emails of all customers

SELECT FirstName, LastName, Email FROM Customers;


-- 2. List all orders with their order dates and corresponding customer names

SELECT Orders.OrderID, Orders.OrderDate, CONCAT(Customers.FirstName, ' ',
Customers.LastName) AS CustomerName

FROM Orders

JOIN Customers ON Orders.CustomerID = Customers.CustomerID;


-- 3. Insert a new customer record into the "Customers" table

INSERT INTO Customers (FirstName, LastName, Email, Phone, Address)

VALUES ('New', 'Customer', 'new.customer@example.com', '123-456-7890', '789 New St');




-- 4. Update the prices of all electronic gadgets in the "Products" table by increasing them by 10%

UPDATE Products

SET Price = Price * 1.10;



-- 5. Delete a specific order and its associated order details from the "Orders" and "OrderDetails"
tables

DELETE FROM OrderDetails WHERE OrderID = 1;

DELETE FROM Orders WHERE OrderID = 1;


-- 6. Insert a new order into the "Orders" table

INSERT INTO Orders (CustomerID, OrderDate, TotalAmount)

VALUES (1, '2023-12-07', 199.99);
```

```
-- 7. Update the contact information of a specific customer in the "Customers" table
UPDATE Customers
SET Email = 'updated.email@example.com', Address = 'Updated Address'
WHERE CustomerID = 1;

-- 8. Recalculate and update the total cost of each order in the "Orders" table
UPDATE Orders
SET TotalAmount = (
    SELECT SUM(Quantity * Price)
    FROM OrderDetails
    JOIN Products ON OrderDetails.ProductID = Products.ProductID
    WHERE OrderDetails.OrderID = Orders.OrderID
);

-- 9. Delete all orders and their associated order details for a specific customer
DELETE FROM OrderDetails WHERE OrderID IN (SELECT OrderID FROM Orders WHERE CustomerID = 1);
DELETE FROM Orders WHERE CustomerID = 1;

-- 10. Insert a new electronic gadget product into the "Products" table
INSERT INTO Products (ProductName, Description, Price)
VALUES ('New Gadget', 'Description of the new gadget', 499.99);


-- 11. Update the status of a specific order in the "Orders" table
alter table orders add status varchar(10) default 'pending';
UPDATE Orders
SET Status = 'Shipped'
WHERE OrderID = 1;
```

```
-- 12. Calculate and update the number of orders placed by each customer in the "Customers" table


alter table customers add NumOrders int;
UPDATE Customers
SET NumOrders = (
    SELECT COUNT(OrderID)
    FROM Orders
    WHERE Orders.CustomerID = Customers.CustomerID
);
```

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

```
-- 1. Retrieve a list of all orders along with customer information for each order

SELECT Orders.OrderID, Orders.OrderDate, CONCAT(Customers.FirstName, ' ',
Customers.LastName)

AS CustomerName

FROM Orders

JOIN Customers ON Orders.CustomerID = Customers.CustomerID;


-- 2. Find the total revenue generated by each electronic gadget product

SELECT Products.ProductID, Products.ProductName, SUM(OrderDetails.Quantity * Products.Price)

AS TotalRevenue

FROM Products

JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID

JOIN Orders ON OrderDetails.OrderID = Orders.OrderID

GROUP BY Products.ProductID, Products.ProductName

order by Products.ProductID, Products.ProductName;


-- 3. List all customers who have made at least one purchase

SELECT Customers.CustomerID, CONCAT(Customers.FirstName, ' ', Customers.LastName)

AS CustomerName, Customers.Email, Customers.Phone

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, Customers.FirstName,Customers.LastName, Customers.Email,
Customers.Phone

HAVING COUNT(Orders.OrderID) > 0;
```

-- 4. Find the most popular electronic gadget (highest total quantity ordered)

```
SELECT  Products.ProductID, Products.ProductName, SUM(OrderDetails.Quantity) AS
TotalQuantityOrdered

FROM Products

JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID

GROUP BY Products.ProductID, Products.ProductName

ORDER BY TotalQuantityOrdered DESC  limit 1;
```

-- 5. Retrieve a list of electronic gadgets along with their corresponding ID

```
SELECT ProductName, ProductId FROM Products;
```

-- 6. Calculate the average order value for each customer

```
SELECT Customers.CustomerID, CONCAT(Customers.FirstName, ' ', Customers.LastName) AS
CustomerName,

AVG(Orders.TotalAmount) AS AvgOrderValue

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName;
```

-- 7. Find the order with the highest total revenue

```
SELECT  Orders.OrderID, CONCAT(Customers.FirstName, ' ', Customers.LastName) AS
CustomerName, SUM(OrderDetails.Quantity * Products.Price) AS TotalRevenue

FROM Orders

JOIN Customers ON Orders.CustomerID = Customers.CustomerID

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

JOIN Products ON OrderDetails.ProductID = Products.ProductID

GROUP BY Orders.OrderID, Customers.FirstName, Customers.LastName

ORDER BY TotalRevenue DESC limit 1;
```

```
-- 8. List electronic gadgets and the number of times each product has been ordered
SELECT Products.ProductName, COUNT(OrderDetails.OrderID) AS OrderCount
FROM Products
JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID
GROUP BY Products.ProductName;

-- 9. Find customers who have purchased a specific electronic gadget product
SELECT Customers.CustomerID, CONCAT(Customers.FirstName, ' ', Customers.LastName)
AS CustomerName, Customers.Email, Customers.Phone
FROM Customers
JOIN Orders ON Customers.CustomerID = Orders.CustomerID
JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
JOIN Products ON OrderDetails.ProductID = Products.ProductID
WHERE Products.ProductName = 'Smartphone';

-- 10. Calculate the total revenue generated by all orders placed within a specific time period

SELECT Customers.CustomerID, CONCAT(FirstName, ' ', LastName) AS CustomerName,
COUNT(OrderID)
AS OrderCount
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
WHERE OrderDate BETWEEN '2023-01-01' AND '2023-12-31'
GROUP BY Customers.CustomerID, FirstName, LastName;
```

-- Task 4: Subquery and its type

```
-- 1. Find which customers have not placed any orders
SELECT CustomerID, CONCAT(FirstName, ' ', LastName) AS CustomerName
FROM Customers
WHERE CustomerID NOT IN (SELECT DISTINCT CustomerID FROM Orders);

-- 2. Find the total number of products available for sale
SELECT COUNT(*) AS TotalProducts
FROM Products;

-- 3. Calculate the total revenue generated by TechShop
SELECT SUM(TotalAmount) AS TotalRevenue
FROM Orders;

-- 4. Calculate the average quantity ordered for products in a specific category
SELECT OrderID, AVG(OrderDetails.Quantity) AS AvgQuantityOrdered FROM OrderDetails
join Products on OrderDetails.OrderID = Products.ProductID group by OrderDetails.OrderID;


-- 5. Calculate the total revenue generated by a specific customer
SELECT SUM(TotalAmount) AS TotalRevenue
FROM Orders WHERE CustomerID = 2;

-- 6. Find the customers who have placed the most orders
SELECT  customers.CustomerID, CONCAT(FirstName, ' ', LastName) AS CustomerName,
COUNT(OrderID) AS OrderCount
FROM orders
left JOIN customers ON customers.CustomerID = orders.CustomerID
GROUP BY CustomerID, FirstName, LastName
ORDER BY OrderCount DESC limit 1;
```

```
-- 7. Find the most popular product category (highest total quantity ordered)

SELECT  ProductName, SUM(OrderDetails.Quantity) AS TotalQuantityOrdered

FROM Products

JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID

GROUP BY ProductName

ORDER BY TotalQuantityOrdered DESC limit 1;


-- 8. Find the customer who has spent the most money on electronic gadgets

SELECT  Customers.CustomerID, CONCAT(FirstName, ' ', LastName) AS CustomerName,
SUM(OrderDetails.Quantity * Products.Price) AS TotalSpending

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

JOIN Products ON OrderDetails.ProductID = Products.ProductID

GROUP BY Customers.CustomerID, FirstName, LastName

ORDER BY TotalSpending DESC limit 1;


-- 9. Calculate the average order value for all customers

SELECT AVG(TotalAmount) AS AvgOrderValue

FROM Orders;


-- 10. Find the total number of orders placed by each customer and list their names along with the
order count

SELECT Customers.CustomerID, CONCAT(FirstName, ' ', LastName) AS CustomerName,
COUNT(OrderID) AS OrderCount

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, FirstName, LastName;
```

THANK YOU