TechShop Task1

```
CREATE DATABASE TechShop;
USE TechShop;


CREATE TABLE Customers(
CustomerID    INT          Primary key,
FirstName     Varchar(50),
LastName      Varchar(30),
Email         Varchar(100)  NOT NULL Unique  ,
Phone         BIGINT          NOT Null Unique,
Address       Varchar(250)
);

CREATE TABLE Products(
ProductID INT Primary key,
ProductName   VARCHAR(50),
ProdDescription  VARCHAR(1000),
Price INT
);

CREATE TABLE Orders(
OrderID INT Primary Key,
CustomerID INT,
FOREIGN KEY (CustomerID) References Customers(CustomerID),
OrderDate Date,
TotalAmount INT
);

CREATE TABLE OrderDetails(
OrderDetailID  INT PRIMARY KEY,
OrderID INT,
Foreign Key (OrderID) References Orders(OrderID),
ProductID INT ,
Foreign Key (ProductID) References Products(ProductID),
Quantity int
);

CREATE TABLE Inventory(
InventoryID INT Primary Key,
ProductID INT,
Foreign Key(ProductID) References Products(ProductID),
QuantityInStock INT,
LastStockUpdate Date
);


Show tables;

INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Phone, Address) VALUES
(1, 'Vinay', 'Pokala', 'vinay.pokala@gmail.com', 9876543210, '123 Main St, NY'),
(2, 'rahul', 'nalla', 'rahul.nalla@yahoo.com', 8765432109, '456 Elm St, CA'),
(3, 'saketh', 'tadaka', 'tadakasaketh@yahoo.com', 7654321098, '789 Oak St, TX'),
(4, 'tata', 'sasi', 'jaintr@outlook.com', 6543210987, '321 Pine St, FL'),
(5, 'giddi', 'vardhan', 'giddy@gmail.com', 5432109876, '147 Maple St, IL'),
```

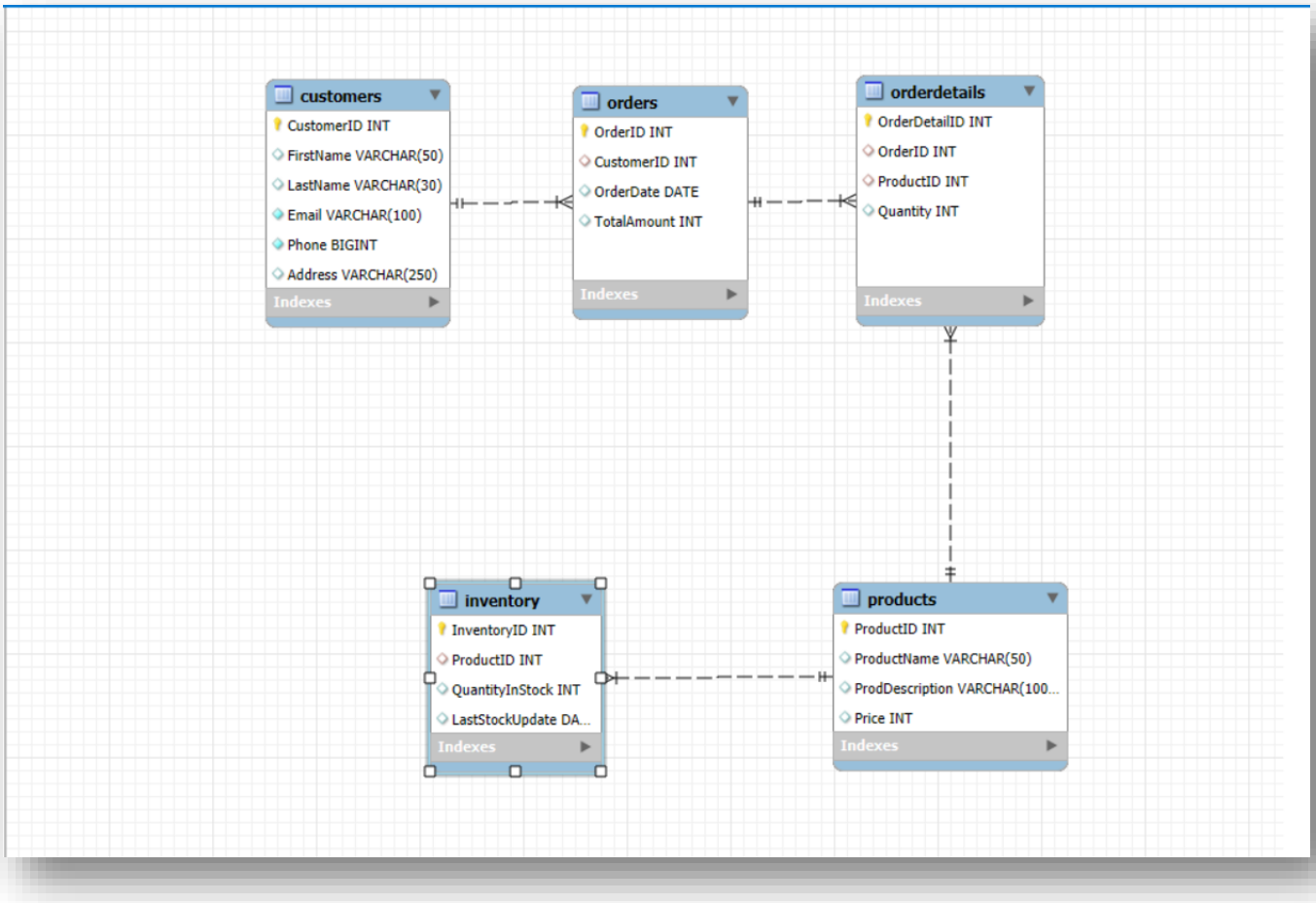
(6, 'shiva', 'cake', 'cake.t2s@gmail.com', 4321098765, '258 Birch St, WA'),
(7, 'nikhil', 'mitta', 'nikhil.lovefailure@yaahoo.com', 3210987654, '369 Cedar St, AZ'),
(8, 'luffy', 'Monkey', 'Joyboy@onepiece.com', 2109876543, '741 Walnut St, NV'),
(9, 'Zoro', 'Ashura', 'Kingofhell@onepiece.com', 1098765432, '852 Chestnut St, CO'),
(10, 'Sanji', 'Vinsmoke', 'Mr.Prince@onepiece.com', 9876504321, '963 Redwood St, MA');

INSERT INTO Products (ProductID, ProductName, ProdDescription, Price) VALUES
(1, 'Laptop', 'High-performance laptop with 16GB RAM and 512GB SSD', 1200),
(2, 'Smartphone', 'Latest 5G smartphone with 128GB storage', 800),
(3, 'Tablet', '10-inch tablet with high-resolution display', 500),
(4, 'Headphones', 'Wireless noise-canceling headphones', 200),
(5, 'Smartwatch', 'Fitness tracking smartwatch with heart rate monitor', 250),
(6, 'Keyboard', 'Mechanical keyboard with RGB lighting', 100),
(7, 'Mouse', 'Wireless ergonomic mouse', 50),
(8, 'Monitor', '27-inch 4K UHD monitor', 400),
(9, 'External Hard Drive', '2TB portable external hard drive', 150),
(10, 'Gaming Chair', 'Ergonomic gaming chair with lumbar support', 300);

INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount) VALUES
(1, 1, '2025-03-10', 1400),
(2, 2, '2025-03-11', 800),
(3, 3, '2025-03-12', 250),
(4, 4, '2025-03-13', 1600),
(5, 5, '2025-03-14', 550),
(6, 6, '2025-03-15', 450),
(7, 7, '2025-03-16', 100),
(8, 8, '2025-03-17', 900),
(9, 9, '2025-03-18', 300),
(10, 10, '2025-03-19', 500);

INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity) VALUES
(1, 1, 1, 1),
(2, 1, 6, 2),
(3, 2, 2, 1),
(4, 3, 5, 1),
(5, 4, 1, 1),
(6, 4, 8, 1),
(7, 5, 4, 2),
(8, 6, 7, 3),
(9, 7, 6, 1),
(10, 8, 3, 2);

INSERT INTO Inventory (InventoryID, ProductID, QuantityInStock, LastStockUpdate) VALUES
(1, 1, 50, '2025-03-01'),
(2, 2, 30, '2025-03-02'),
(3, 3, 40, '2025-03-03'),
(4, 4, 25, '2025-03-04'),
(5, 5, 20, '2025-03-05'),
(6, 6, 100, '2025-03-06'),
(7, 7, 150, '2025-03-07'),
(8, 8, 10, '2025-03-08'),
(9, 9, 35, '2025-03-09'),
(10, 10, 12, '2025-03-10');


customers
CustomerID INT
FirstName VARCHAR(50)
LastName VARCHAR(30)
Email VARCHAR(100)
Phone BIGINT
Address VARCHAR(250)
Indexes
orders
OrderID INT
CustomerID INT
OrderDate DATE
TotalAmount INT
Indexes
orderdetails
OrderDetailID INT
OrderID INT
ProductID INT
Quantity INT
Indexes
inventory
InventoryID INT
ProductID INT
QuantityInStock INT
LastStockUpdate DA...
Indexes
products
ProductID INT
ProductName VARCHAR(50)
ProdDescription VARCHAR(100...
Price INT
Indexes

TASK 2:

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

```
SELECT FirstName,LastName,Email From Customers;
```

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

```
SELECT o.OrderID,o.OrderDate ,CONCAT(c.FirstName,' ',c.LastName) as CustomerName
From Orders as o, Customers as c
WHERE o.CustomerID=c.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 (CustomerID ,FirstName, LastName, Email, Phone, Address)
VALUES (11, 'Chilliveri', 'Shivani', 'Shiva.ch@dumpyard.com' , 7395406361 , '1-2/23 Main, NY');
```

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*(10/100) WHERE ProductID>0;
```

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.

```
DELETE FROM OrderDetails WHERE OrderID=10;
DELETE FROM Orders WHERE OrderID=10;
```

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, CustomerID, OrderDate, TotalAmount)
VALUES (10, 11, '2025-03-21', 2500);
```

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.

```
UPDATE Customers SET Email = 'giddy@yahoo.com' , address= '157 Maple St, IL'

WHERE CustomerID=5;
```

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 SET TotalAmount=(SELECT SUM(OrderDetails.Quantity*OrderDetails.Price)
FROm OrderDetails WHERE Orders.OrderID=OrderDetails.OrderID GROUP BY
OrderDetails.OrderID) WHERE ORDERID>0;
```

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.

```
DELETE FROM OrderDetails WHERE OrderID IN (SELECT ORDERID FROM Orders

WHERE CustomerID=4);

DELETE FROM Orders WHERE CustomerID=4;
```

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, ProductName, ProdDescription, Price)

VALUES (11, 'Smartwatch', 'A high-tech smartwatch with fitness tracking features', 250);
```

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.

```
UPDATE Orders SET Orderstatus='Shipped' WHERE OrderID=3;
```

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.

```
UPDATE Customers SET NoOfOrders = ( SELECT COUNT(OrderID) FROM Orders WHERE
Orders.CustomerId=Customers.CustomerID)
WHERE CustomerID>0;
```

TASK 3:

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

```
SELECT   *   FROM ORDERS    INNER JOIN Customers

ON ORDERS.CustomerID=Customers.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 OD.ProductID , P.ProductName, (SUM(OD.Quantity)*P.Price) AS TotalRevenue
From OrderDetails as OD  Inner JOIN Products as p on OD.ProductID=p.ProductID
GROUP BY OD.ProductID,p.ProductName LIMIT 0, 1000;
```

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

```
SELECT FirstName, LastName, Phone, Email   From Customers as c
  JOIN Orders as 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.ProductName , p.ProductID , d.Quant
FROM Products as p
INNER JOIN
(SELECT ProductID ,SUM(Quantity) as quant     FROM Orderdetails GROUP BY ProductID)AS d
On d.ProductID=p.ProductID
 WHERE   d.quant= (SELECT MAX(Quants)
FROM (SELECT SUM(Quantity) as Quants FROM OrderDetails Group By ProductID) AS maxq);
```

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

```
SELECT ProductID, ProductName, Category FROM Products;
```

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 c.FirstName,c.LastName ,b.TotalAmount/b.Quant as OrderValue
FROM Customers AS c
LEFT JOIN (SELECT CustomerID,TotalAmount,a.Quant FROM Orders INNER JOIN (SELECT
ORDERID, SUM(Quantity) as Quant FROM OrderDetails  GROUP BY ORDERID )AS a
ON a.orderID=Orders.OrderID)as b   ON b.CustomerID=c.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  C.*,o.OrderID , o.TotalAmount       FROM Customers AS c
INNER JOIN (SELECT OrderID, CustomerID ,TotalAmount     FROM Orders WHERE TotalAmount=
(SELECT MAX(TotalAmount) FROm Orders))AS o
 ON C.CustomerID=o.CustomerID;
```

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

```
SELECT p.ProductName,p.ProductID, COUNT(o.OrderID) AS NoOfTimesOrdered
From Products AS p
LEFT Join OrderDetails AS o On o.productid=p.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.

```
SELECT c.CustomerID, c.FirstName, c.LastName, c.Email, c.Phone
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 = 'Mouse';
```

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.

```
SELECT SUM(totalAmount) AS totalRevenue FROM Orders
WHERE orderdate BETWEEN '2025-03-10' AND '2025-03-15';
```

TASK 4:

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

```
SELECT c.FirstName, c.LastName, c.Phone, c.Email FROM Customers AS c

LEFT JOIN Orders as o ON o.CustomerID=c.CustomerID

WHERE o.OrderID is NULL;
```

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

```
SELECT SUM(QuantityInStock) AS TotalAvailableProducts FROM Inventory;
```

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

```
SELECT SUM(totalAmount) AS TotalRevenue 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.

```
SELECT p.Category, (SUM(o.Quantity)/COUNT(o.Quantity)) AS AvgOrders

FROM Products AS p

LEFT JOIN OrderDetails AS o

ON p.ProductID=o.ProductID

GROUP BY p.Category;
```

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 c.CustomerID, c.FirstName, c.LastName ,o.TotalAmount FROM Customers AS c

LEFT JOIN Orders AS o ON c.CustomerID=o.CustomerID WHERE c.CustomerID=1;
```

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 CustomerID, FirstName, LastName , NoOfOrders

FROM Customers WHERE NoOfOrders= (SELECT MAX(NoOfOrders) FROM Customers );
```

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 p.Category, SUM(o.Quantity) AS TotalQuantity

FROM Products AS p

LEFT JOIN OrderDetails AS o ON o.ProductID=p.ProductID

GROUP BY p.Category;
```

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.CustomerID, c.FirstName, c.LastName, o.TotalRevenue FROM Customers AS c

INNER JOIN (SELECT CustomerID ,SUM(TotalAmount) AS TotalRevenue FROM Orders GROUP BY CustomerID) AS o   ON c.CustomerID=o.CustomerID

WHERE o.TotalRevenue = (SELECT MAX(TotalRevenue)  FROM(SELECT CustomerID ,SUM(TotalAmount) AS TotalRevenue FROM Orders GROUP BY CustomerID)as max_revenue) ;
```

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

```
SELECT c.CustomerID, c.FirstName, c.LastName, (o.TOrders/o.TotalRevenue) AS AvgOrderValue

FROM Customers AS c

INNER JOIN (SELECT CustomerID, COUNT(OrderID) AS TOrders , SUM(TotalAmount) AS TotalRevenue FROM ORDERS GROUP BY CustomerID) AS o

ON c.CustomerID=o.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, FirstName, LastName, NoOfOrders FROM Customers;
```