**ASSIGNMENT – SQL – ELECTRONIC GADGETS**

**Task:1. Database Design:**

**1. Create the database named "TechShop".**

**ANS:**

CREATE DATABASE TechShop;

--use the created database.

use techshop;

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

**ANS:**

CREATE TABLE customers(

customer\_id INT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

email VARCHAR(50),

phone VARCHAR(15),

address VARCHAR(100)

);

CREATE TABLE products(

product\_id INT PRIMARY KEY,

product\_name VARCHAR(50),

description VARCHAR(100),

price FLOAT

);

CREATE TABLE orders (

order\_id INT PRIMARY KEY,

customer\_id INT,

order\_date DATE,

total\_amount FLOAT,

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)

);

CREATE TABLE order\_details (

order\_detail\_id INT PRIMARY KEY,

order\_id INT,

product\_id INT,

quantity INT,

FOREIGN KEY (order\_id) REFERENCES orders(order\_id),

FOREIGN KEY (product\_id) REFERENCES products(Product\_id)

);

CREATE TABLE inventory (

inventory\_id INT PRIMARY KEY,

product\_id INT,

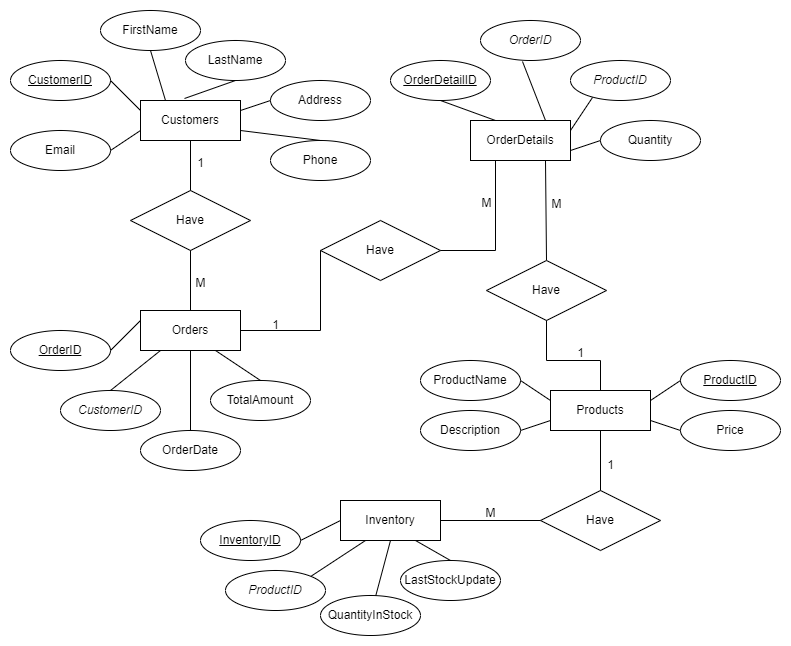
quantity\_instock INT,

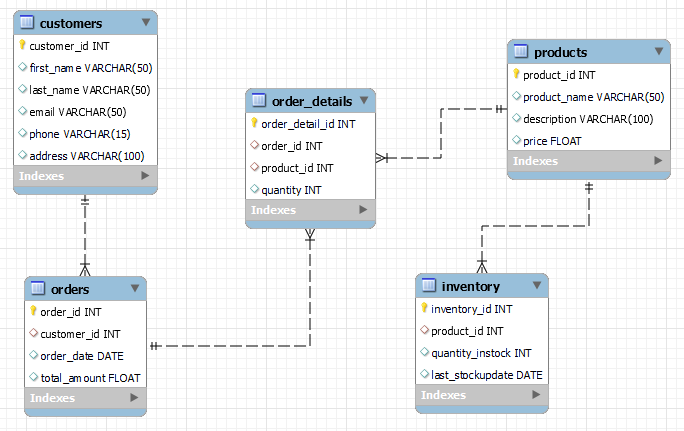
last\_stockupdate DATE,

FOREIGN KEY (product\_id) REFERENCES products(product\_id)

);

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





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

**ANS:**

Primary Key in the SQL Tables is,

* Customer\_id in customers table.
* Product\_id in products table.
* Order\_id in orders table.
* Order\_detail\_id in order\_details table.
* Inventory\_id in inventory table.

Foreign Key in the SQL Tables is,

* Customer\_id in orders table reference on customers table.
* Order\_id in order\_details table reference on orders table.
* Product\_id in order\_details table reference on products table.
* Product\_id in inventory table reference on products table.

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

* **Customers**
* **Products**
* **Orders**
* **OrderDetails**
* **Inventory**

**ANS:**

-- Insert sample records into the Customers table

INSERT INTO Customers (customer\_id, first\_name, last\_name, email, phone, address)

VALUES

(1,'gnnanavel','srinivasan','gvgod18@gmail.com',7903778934,'chennai'),

(2,'Rohit','kumar','rkumar23@gmail.com',7899378920,'delhi'),

(3,'kiran','prakash','kiranp3@gmail.com',7890378927,'kerala'),

(4,'srinivasan','venkat','srivenkat12@gmail.com',7892047890,'mumbai'),

(5,'praveen','singh','psingh88@gmail.com',9087124567,'pune'),

(6,'james','jones','j\_jones12@gmail.com',9057894689,'bangalore'),

(7,'virat','kohli','viratk76@gmail.com',9037489023,'kolkata'),

(8,'rohit','sharma','rohit5@gmail.com',8573082378,'mumbai'),

(9,'shreyas','iyer','shreyas3@gmail.com',9487234567,'chennai'),

(10,'Md','asjad','mdasjad45@gmail.com',9380472389,'hyderabad');

INSERT INTO products values

(1,'air conditioner','it cools',20000.0),

(2,'Fitness watch', 'Fitness monitoring watch', 130.0),

(3,'Ps5','gaming console by sony',49000.0),

(4,'Light bulb','100w filament bulb',80.0),

(5,'LG led tv','4k full HD tv',85000.0),

(6,'Smartphone','lastest hi-tech phone',20000.0),

(7,'Laptop','Gaming laptop hi-speed processor',95000.0),

(8,'BT Earphones','Wrieless earphone',2000.0),

(9,'Speaker','Dobly atoms quality sound',55000.0),

(10,'Table fan','on-charge also battery powered fan',25500.0);

-- Insert sample records into the Orders table

INSERT INTO orders VALUES

(1, 1, '2023-01-11', 10000.0),

(2, 5, '2023-02-10', 15000.0),

(3, 6, '2023-03-15', 59000.0),

(4, 3, '2023-04-25', 47000.0),

(5, 4, '2023-05-28', 65000.0),

(6, 2, '2023-06-12', 20000.0),

(7, 8, '2023-07-18', 125000.0),

(8, 9, '2023-08-30', 12500.0),

(9, 10, '2023-09-17', 45000.0),

(10, 7, '2023-10-25', 35000.0);

-- Insert sample records into the OrderDetails table

INSERT INTO order\_details VALUES

(1, 1, 1, 1),

(2, 2, 5, 2),

(3, 3, 4, 3),

(4, 4, 3, 1),

(5, 5, 10, 1),

(6, 6, 9, 1),

(7, 7, 2, 2),

(8, 8, 7, 1),

(9, 9, 8, 1),

(10, 10, 6, 2);

-- Insert sample records into the Inventory table

INSERT INTO inventory VALUES

(1, 1, 35, '2023-01-01'),

(2, 2, 19, '2023-01-05'),

(3, 3, 28, '2023-01-10'),

(4, 4, 22, '2023-01-15'),

(5, 5, 11, '2023-01-20'),

(6, 6, 9, '2023-01-25'),

(7, 7, 17, '2023-02-01'),

(8, 8, 8, '2023-02-05'),

(9, 9, 15, '2023-02-10'),

(10, 10, 25, '2023-02-15');

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

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

**ANS:**

SELECT first\_name, last\_name, email FROM customers;

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

**ANS:**

SELECT order\_id, order\_date, first\_name,last\_name

FROM orders,customers

WHERE orders.customer\_id=customers.customer\_id;

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

**ANS:**

INSERT INTO customers (customer\_id, first\_name, last\_name, email, phone, address)

VALUES (11, 'Manjula', 'meena', 'manjula7@gmail.com', '9345782987', 'pune');

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

**ANS:**

ALTER TABLE products ADD COLUMN category VARCHAR(25) DEFAULT 'Electronic Gadget';

UPDATE products SET price = price \* 1.10 WHERE category = 'Electronic Gadget';

**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.**

**ANS:**

SET @orderId = 1;

DELETE FROM order\_details WHERE order\_id = @orderId;

DELETE FROM orders WHERE order\_id = @orderId;

**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.**

**ANS:**

INSERT INTO orders (order\_id, customer\_id, order\_date, total\_amount)

VALUES (11, 1, '2023-11-27', 9800.0);

**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.**

**ANS:**

SET @customerId = 7;

SET @emaIl = 'vk\_viratkohli18@gmail.com';

SET @addreSs = 'chennai';

UPDATE customers

SET email = @emaIl, address = @addreSs

WHERE customer\_id = @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**

**ANS:**

UPDATE orders

SET total\_amount = (

SELECT SUM(quantity \* price)

FROM order\_details

JOIN products ON order\_details.product\_id = products.product\_id

WHERE order\_details.order\_id = orders.order\_id

);

**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.**

**ANS:**

SET @CusId = 10;

DELETE FROM order\_details WHERE order\_id IN (SELECT order\_id FROM orders WHERE customer\_id = @CusId);

DELETE FROM orders WHERE customer\_id = @CusId;

**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.**

**ANS:**

INSERT INTO products (product\_id,product\_name, description, price, category)

VALUES (11, 'Wireless charger', '48W fast charging charger wireless', 3500.0, 'Electronic Gadget');

**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.**

**ANS:**

ALTER TABLE orders ADD COLUMN status VARCHAR(25) DEFAULT 'Pending';

SET @ordId = 1;

SET @sta = 'Shipped';

UPDATE orders

SET status = @sta

WHERE order\_id = @ordId;

**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.**

**ANS:**

ALTER TABLE customers ADD COLUMN numbers\_oforders INT;

UPDATE customers

SET numbers\_oforders = (

SELECT COUNT(order\_id)

FROM orders

WHERE orders.customer\_id = customers.customer\_id

);

**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.**

**ANS:**

SELECT orders.order\_id, orders.order\_date,

CONCAT(customers.first\_name, ' ', customers.last\_name) AS CustomerName

FROM orders

JOIN customers ON orders.customer\_id = customers.customer\_id;

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

**ANS:**

SELECT p.product\_id, p.product\_name, SUM(o.quantity \* p.price) AS TotalRevenue

FROM order\_details o

JOIN products p ON o.product\_id = p.product\_id

WHERE p.category = 'Electronic Gadget'

GROUP BY p.product\_id;

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

**ANS:**

SELECT DISTINCT CONCAT(c.first\_name, ' ', c.last\_name) AS CustomerName, c.email, c.phone, c.address

FROM customers c

JOIN orders o ON c.customer\_id = o.customer\_id;

**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.**

**ANS:**

SELECT p.product\_id, p.product\_name, SUM(o.quantity) AS TotalQuantityOrdered

FROM order\_details o

JOIN products p ON p.product\_id = o.product\_id

GROUP BY p.product\_id

ORDER BY TotalQuantityOrdered DESC LIMIT 1;

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

**ANS:**

SELECT product\_name, category

FROM products

WHERE category = 'Electronic Gadget';

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

**ANS:**

SELECT c.customer\_id, CONCAT(c.first\_name, ' ', c.last\_name) AS CustomerName, AVG(o.total\_amount) AS AvgOrderValue

FROM customers c

JOIN orders o ON c.customer\_id = o.customer\_id

GROUP BY c.customer\_id;

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

**ANS:**

SELECT o.order\_id,CONCAT(c.first\_name, ' ', c.last\_name) AS CustomerName,

total\_amount AS TotalRevenue

FROM orders o

JOIN customers c ON o.customer\_id = c.customer\_id

ORDER BY total\_amount DESC LIMIT 1;

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

**ANS:**

SELECT p.product\_id, p.product\_name, COUNT(d.order\_id) AS OrderCount

FROM products p

JOIN order\_details d ON p.product\_id = d.product\_id

WHERE p.category = 'Electronic Gadget'

GROUP BY p.product\_id, p.product\_name;

**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.**

**ANS:**

SET @prodName = 'Laptop';

SELECT c.customer\_id, CONCAT(c.first\_name, ' ', c.last\_name) AS CustomerName,

c.email, c.phone FROM customers c

JOIN orders o ON c.customer\_id = o.customer\_id

JOIN order\_details d ON o.order\_id = d.order\_id

JOIN products p ON d.product\_id = p.product\_id

WHERE p.product\_name = @prodName;

**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.**

**ANS:**

SET @startDate = '2023-05-10';

SET @endDate = '2023-10-01';

SELECT SUM(total\_amount) AS TotalRevenue

FROM orders

WHERE order\_date BETWEEN @startDate AND @endDate;

**Task 4. Subquery and its type:**

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

**ANS:**

SELECT customer\_id, CONCAT(first\_name, ' ', last\_name) AS CustomerName

FROM customers

WHERE customer\_id NOT IN (SELECT DISTINCT customer\_id FROM orders);

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

**ANS:**

SELECT COUNT(\*) AS TotalProducts

FROM products

WHERE product\_id IN (SELECT DISTINCT product\_id FROM order\_details);

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

**ANS:**

SELECT SUM((SELECT SUM(d.quantity \* p.price) FROM order\_details d, products p

WHERE d.order\_id = orders.order\_id)) 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.**

**ANS:**

SET @categoryName = 'Electronic Gadget';

SELECT AVG(quantity) AS Average\_Quantity\_Ordered

FROM order\_details

WHERE product\_id IN (SELECT product\_id FROM products WHERE category = @categoryName);

**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.**

**ANS:**

SET @customerId = 4;

SELECT SUM(d.quantity \* p.price) AS TotalRevenue

FROM order\_details d, products p

WHERE order\_id IN (SELECT order\_id FROM orders WHERE customer\_id = @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.**

**ANS:**

SELECT customer\_id, CONCAT(first\_name, ' ', last\_name) AS CustomerName,

( SELECT COUNT(\*) FROM orders o WHERE o.customer\_id = c.customer\_id

) AS NumberOfOrders FROM customers c

ORDER BY NumberOfOrders 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.**

**ANS:**

SELECT p.category,(SELECT SUM(quantity) FROM order\_details d

WHERE p.product\_id = d.product\_id) AS TotalQuantityOrdered

FROM products p

ORDER BY TotalQuantityOrdered 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.**

**ANS:**

SELECT c.customer\_id, CONCAT(c.first\_name, ' ', c.last\_name) AS CustomerName,

(SELECT total\_amount FROM orders o

WHERE o.customer\_id = c.customer\_id) AS Total\_Spending

FROM customers c WHERE c.customer\_id = (SELECT customer\_id

FROM orders

ORDER BY orders.total\_amount 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.**

**ANS:**

SELECT c.customer\_id, CONCAT(c.first\_name, ' ', c.last\_name) AS CustomerName,

( SELECT AVG(d.quantity \* p.price) FROM products p, order\_details d, orders o

WHERE o.customer\_id = c.customer\_id

AND o.order\_id = d.order\_id) AS Average\_Order\_Value

FROM customers c;

**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.**

**ANS:**

SELECT CONCAT(c.first\_name,' ',c.last\_name) AS CustomerName, (

SELECT COUNT(order\_id) FROM orders o

WHERE o.customer\_id = c.customer\_id) AS Order\_Count

FROM customers c;