SQL Script for Database and Tables Creation

Create the database

CREATE DATABASE ecommerce;

Use the database

USE ecommerce;

Create the customers table

CREATE TABLE customers (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(255) NOT NULL, email VARCHAR(255) UNIQUE NOT NULL, address TEXT);

Create the products table

CREATE TABLE products (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(255) NOT NULL, price DECIMAL(10, 2) NOT NULL, description TEXT);

Create the orders table

CREATE TABLE orders (id INT AUTO_INCREMENT PRIMARY KEY, customer_id INT NOT NULL, order_date DATE NOT NULL, total_amount DECIMAL(10, 2) NOT NULL, FOREIGN KEY (customer_id) REFERENCES customers(id));

Insert sample data into customers

INSERT INTO customers (name, email, address) VALUES ('Arunkumar', 'ak@example.com', '123 main Street'), ('Ramesh', 'ramesh@example.com', '456 middle Avenue'), ('Suresh', 'suresh@example.com', '789 last Road');

Insert sample data into products

INSERT INTO products (name, price, description) VALUES ('Apple iphone', 20000, 'Iphone 16'), ('Samsung headphone', 3000, 'headphone with anc'), ('lg fridge', 8000, 'good refrigirater');

Insert sample data into orders

INSERT INTO orders (customer_id, order_date, total_amount) VALUES (1, '2024-12-09', 10000.00), (2, '2024-12-19', 2000.00), (3, '2024-11-19', 500.00);

1. Retrieve all customers who have placed an order in the last 30 days:

SELECT DISTINCT c.name, c.email FROM customers c JOIN orders o ON c.id = o.customer_id WHERE o.order_date >= CURDATE() - INTERVAL 30 DAY;

2. Get the total amount of all orders placed by each customer:

SELECT c.name, SUM(o.total_amount) AS total_spent FROM customers c JOIN orders o ON c.id = o.customer_id GROUP BY c.name;

3. Update the price of lg fridge to 8000.00:

UPDATE products SET price = 8000.00 WHERE name = 'lg fridge';

4. Add a new column discount to the products table:

ALTER TABLE products ADD COLUMN discount DECIMAL(5, 2) DEFAULT 0.00;

5. Retrieve the top 3 products with the highest price:

SELECT name, price FROM products ORDER BY price DESC LIMIT 3;

6. Join the orders and customers tables to retrieve the customer's name and order date for each order: SELECT c.name AS customer_name, o.order_date FROM customers c JOIN orders o ON c.id = o.customer_id;

7. Retrieve the orders with a total amount greater than 150.00: SELECT * FROM orders WHERE total_amount > 150.00;

Database Normalization

- Create a separate table for order_items:
- Create the order_items table CREATE TABLE order_items (id INT AUTO_INCREMENT PRIMARY KEY, order_id INT NOT NULL, product_id INT NOT NULL, quantity INT NOT NULL DEFAULT 1, FOREIGN KEY (order_id) REFERENCES orders(id), FOREIGN KEY (product_id) REFERENCES products(id));
- Update the orders table to remove product-related data ALTER TABLE orders DROP COLUMN total_amount;

Sample data for order_items: INSERT INTO order_items (order_id, product_id, quantity) VALUES (1, 1, 2), (1, 2, 1), (2, 3, 4); 9. Retrieve the average total of all orders: SELECT AVG(total_amount) AS average_order_total FROM (SELECT o.id, SUM(oi.quantity * p.price) AS total_amount FROM orders o JOIN order_items oi ON o.id = oi.order_id JOIN products p ON oi.product_id = p.id GROUP BY o.id) AS order_totals;