My SQL Task

Create a Database named ecommercetask.

Create database ecommercetask;

Use ecommercetask;

```
mysql> create database ecommercetask;
Query OK, 1 row affected (0.02 sec)

mysql> use ecommmercetask;
ERROR 1049 (42000): Unknown database 'ecommmercetask'
mysql> use ecommercetask;
Database changed
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| ecommerce          |
| ecommercetask      |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| task1              |
+--------------------+
7 rows in set (0.01 sec)
```

Create three tables: customers, orders, and products.

1)**Customers Table**:

create table customers(id INT AUTO_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) NOT NULL,

address VARCHAR(255));

```
mysql> create table customers(id INT AUTO_INCREMENT PRIMARY KEY,
    ->                   name VARCHAR(100) NOT NULL,
    ->                   email VARCHAR(100) NOT NULL,
    ->                   address VARCHAR(255));
Query OK, 0 rows affected (0.05 sec)

mysql> select * from customers;
Empty set (0.01 sec)

mysql> desc customers;
+---------+--------------+------+-----+---------+----------------+
| Field   | Type         | Null | Key | Default | Extra          |
+---------+--------------+------+-----+---------+----------------+
| id      | int          | NO   | PRI | NULL    | auto_increment |
| name    | varchar(100) | NO   |     | NULL    |                |
| email   | varchar(100) | NO   |     | NULL    |                |
| address | varchar(255) | YES  |     | NULL    |                |
+---------+--------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

2.**Orders Table**:

create table orders (id INT AUTO_INCREMENT PRIMARY KEY, customer_id INT, order_date DATE, total_amount DECIMAL(10, 2),

FOREIGN KEY (customer_id) REFERENCES customers(id));

```
mysql> create table orders (id INT AUTO_INCREMENT PRIMARY KEY,
    -> customer_id INT,
    -> order_date DATE,
    -> total_amount DECIMAL(10, 2),
    -> FOREIGN KEY (customer_id) REFERENCES customers(id));
Query OK, 0 rows affected (0.03 sec)

mysql> desc orders;
+--------------+---------------+------+-----+---------+----------------+
| Field        | Type          | Null | Key | Default | Extra          |
+--------------+---------------+------+-----+---------+----------------+
| id           | int           | NO   | PRI | NULL    | auto_increment |
| customer_id  | int           | YES  | MUL | NULL    |                |
| order_date   | date          | YES  |     | NULL    |                |
| total_amount | decimal(10,2) | YES  |     | NULL    |                |
+--------------+---------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

3.**Products Table:**

create table products(id INT PRIMARY KEY AUTO_INCREMENT, name VARCHAR(100) NOT NULL, price DECIMAL(10,2), description TEXT);

```
mysql> create table products(id INT PRIMARY KEY AUTO_INCREMENT,
    -> name VARCHAR(100) NOT NULL,
    -> price DECIMAL(10,2),
    -> description TEXT);
Query OK, 0 rows affected (0.03 sec)

mysql> desc products;
+-------------+---------------+------+-----+---------+----------------+
| Field       | Type          | Null | Key | Default | Extra          |
+-------------+---------------+------+-----+---------+----------------+
| id          | int           | NO   | PRI | NULL    | auto_increment |
| name        | varchar(100)  | NO   |     | NULL    |                |
| price       | decimal(10,2) | YES  |     | NULL    |                |
| description | text          | YES  |     | NULL    |                |
+-------------+---------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

Insert sample data:

Sample data for Customers Table:

insert into customers(name, email, address) values(

('John Doe', 'john.doe@example.com', '123 Elm St'),

('Jane Smith', 'jane.smith@example.com', '456 Maple Ave'),

('Alice Johnson', 'alice.j@example.com', '789 Oak Dr'),

);

```
mysql> INSERT INTO customers (name, email, address) VALUES
    -> ('John Doe', 'john.doe@example.com', '123 Elm St'),
    -> ('Jane Smith', 'jane.smith@example.com', '456 Maple Ave'),
    -> ('Alice Johnson', 'alice.j@example.com', '789 Oak Dr');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from customers;
+----+--------------+----------------------+--------------+
| id | name         | email                | address      |
+----+--------------+----------------------+--------------+
|  1 | John Doe     | john.doe@example.com | 123 Elm St   |
|  2 | Jane Smith   | jane.smith@example.com | 456 Maple Ave |
|  3 | Alice Johnson | alice.j@example.com | 789 Oak Dr   |
+----+--------------+----------------------+--------------+
3 rows in set (0.00 sec)
```

Sample data for orders:

Insert into orders (customer_id, order_date, total_amount) VALUES (1,

CURDATE(), 60.00), (2, CURDATE() - INTERVAL 15 DAY, 75.00), (1, CURDATE() -

INTERVAL 35 DAY, 80.00);

```
mysql> Insert into orders (customer_id, order_date, total_amount) VALUES (1, CURDATE(), 60.00), (2, CURDATE() - INTERVAL 15 DAY, 75.0
0), (1, CURDATE() - INTERVAL 35 DAY, 80.00);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>

mysql> select * from orders;
+----+-------------+------------+--------------+
| id | customer_id | order_date | total_amount |
+----+-------------+------------+--------------+
|  1 |           1 | 2024-11-06 |        60.00 |
|  2 |           2 | 2024-10-22 |        75.00 |
|  3 |           1 | 2024-10-02 |        80.00 |
+----+-------------+------------+--------------+
3 rows in set (0.00 sec)
```

Sample data for products Table:

INSERT INTO products (name, price, description)

VALUES ('Product A', 20.00, 'Description of Product A'),

('Product B', 35.00, 'Description of Product B'),

('Product C', 50.00, 'Description of Product C');

```
mysql>  INSERT INTO products (name, price, description)
    ->  VALUES ('Product A', 20.00, 'Description of Product A'),
    ->  ('Product B', 35.00, 'Description of Product B'),
    ->  ('Product C', 50.00, 'Description of Product C');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from products;
+----+-----------+-------+--------------------------+
| id | name      | price | description              |
+----+-----------+-------+--------------------------+
|  1 | Product A | 20.00 | Description of Product A |
|  2 | Product B | 35.00 | Description of Product B |
|  3 | Product C | 50.00 | Description of Product C |
+----+-----------+-------+--------------------------+
3 rows in set (0.00 sec)
```

Queries:

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

```
select DISTINCT c.* FROM customers c
JOIN orders o ON c.id = o.customer_id
WHERE o.order_date >= CURDATE() - INTERVAL 30 DAY;
```

```
mysql> select DISTINCT c.* FROM customers c
    -> JOIN orders o ON c.id = o.customer_id
    -> WHERE o.order_date >= CURDATE() - INTERVAL 30 DAY;
+----+------------+----------------------+---------------+
| id | name       | email                | address       |
+----+------------+----------------------+---------------+
|  1 | John Doe   | john.doe@example.com | 123 Elm St    |
|  2 | Jane Smith | jane.smith@example.com | 456 Maple Ave |
+----+------------+----------------------+---------------+
2 rows in set (0.01 sec)
```

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.id;
```

```
mysql> 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.id;
+------------+-------------+
| name       | total_spent |
+------------+-------------+
| John Doe   |      140.00 |
| Jane Smith |       75.00 |
+------------+-------------+
2 rows in set (0.00 sec)
```

3.Update the price of Product C to 45.00:

update products SET price=45.00 where name=”Product C”;

```
mysql> select * from products;
+----+-----------+-------+--------------------------+
| id | name      | price | description              |
+----+-----------+-------+--------------------------+
|  1 | Product A | 20.00 | Description of Product A |
|  2 | Product B | 35.00 | Description of Product B |
|  3 | Product C | 50.00 | Description of Product C |
+----+-----------+-------+--------------------------+
3 rows in set (0.00 sec)

mysql> update products SET price=45.00 where name="Product C";
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from products;
+----+-----------+-------+--------------------------+
| id | name      | price | description              |
+----+-----------+-------+--------------------------+
|  1 | Product A | 20.00 | Description of Product A |
|  2 | Product B | 35.00 | Description of Product B |
|  3 | Product C | 45.00 | Description of Product C |
+----+-----------+-------+--------------------------+
3 rows in set (0.00 sec)
```

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

alter table products add discount decimal(5,2) DEFAULT 0.00;

```
mysql> alter table products add discount decimal(5,2) DEFAULT 0.00;
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> select * from products;
+----+-----------+-------+--------------------------+----------+
| id | name      | price | description              | discount |
+----+-----------+-------+--------------------------+----------+
|  1 | Product A | 20.00 | Description of Product A |     0.00 |
|  2 | Product B | 35.00 | Description of Product B |     0.00 |
|  3 | Product C | 45.00 | Description of Product C |     0.00 |
+----+-----------+-------+--------------------------+----------+
3 rows in set (0.00 sec)
```

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

We have to add more products in product table using insert query,

Insert into products (name, price, description) values('Product D', 60.00, 'Description of Product D'),

('Product E', 50.00, 'Description of Product E'),

('Product F', 90.00, 'Description of Product F');

```
mysql>      Insert into products (name, price, description)
    ->      values('Product D', 60.00, 'Description of Product D'),
    ->      ('Product E', 50.00, 'Description of Product E'),
    ->      ('Product F', 90.00, 'Description of Product F');
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from products;
+----+-----------+-------+-------------------------+----------+
| id | name      | price | description             | discount |
+----+-----------+-------+-------------------------+----------+
|  1 | Product A | 20.00 | Description of Product A |     0.00 |
|  2 | Product B | 35.00 | Description of Product B |     0.00 |
|  3 | Product C | 45.00 | Description of Product C |     0.00 |
|  4 | Product D | 60.00 | Description of Product D |     0.00 |
|  5 | Product E | 50.00 | Description of Product E |     0.00 |
|  6 | Product F | 90.00 | Description of Product F |     0.00 |
+----+-----------+-------+-------------------------+----------+
6 rows in set (0.00 sec)
```

Now, Retrieve the top 3 products with the highest price,

SELECT * FROM products ORDER BY price DESC LIMIT 3;

```
mysql> SELECT * FROM products ORDER BY price DESC LIMIT 3;
+----+-----------+-------+-------------------------+----------+
| id | name      | price | description             | discount |
+----+-----------+-------+-------------------------+----------+
|  6 | Product F | 90.00 | Description of Product F |     0.00 |
|  4 | Product D | 60.00 | Description of Product D |     0.00 |
|  5 | Product E | 50.00 | Description of Product E |     0.00 |
+----+-----------+-------+-------------------------+----------+
3 rows in set (0.00 sec)
```

6.Get the names of customers who have ordered Product A.

Lets, create order_items Table,

create table order_items (id INT AUTO_INCREMENT PRIMARY KEY,

order_id INT,

product_id INT,

quantity INT DEFAULT 1,

FOREIGN KEY (order_id) REFERENCES orders(id),

FOREIGN KEY (product_id) REFERENCES products(id));

```
mysql> create table order_items ( id INT AUTO_INCREMENT PRIMARY KEY,
    ->               order_id INT,
    ->               product_id INT,
    ->               quantity INT DEFAULT 1,
    ->               FOREIGN KEY (order_id) REFERENCES orders(id),
    ->              FOREIGN KEY (product_id) REFERENCES products(id));
Query OK, 0 rows affected (0.05 sec)

mysql> select * from order_items;
Empty set (0.00 sec)

mysql> desc order_items;
+------------+------+------+-----+---------+----------------+
| Field      | Type | Null | Key | Default | Extra          |
+------------+------+------+-----+---------+----------------+
| id         | int  | NO   | PRI | NULL    | auto_increment |
| order_id   | int  | YES  | MUL | NULL    |                |
| product_id | int  | YES  | MUL | NULL    |                |
| quantity   | int  | YES  |     | 1       |                |
+------------+------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

Insert Sample Data into order_items,

Insert into order_items (order_id, product_id, quantity) values (1, 1, 2),(1, 2, 1), (2, 1, 1),(3, 3, 3);

```
mysql>        Insert into order_items (order_id, product_id, quantity)
    ->        values (1, 1, 2),(1, 2, 1), (2, 1, 1),(3, 3, 3);
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from order_items;
+----+----------+------------+----------+
| id | order_id | product_id | quantity |
+----+----------+------------+----------+
|  1 |        1 |          1 |        2 |
|  2 |        1 |          2 |        1 |
|  3 |        2 |          1 |        1 |
|  4 |        3 |          3 |        3 |
+----+----------+------------+----------+
4 rows in set (0.00 sec)
```

Retrieve Customer Names Who Have Ordered Product A;

```
select DISTINCT c.name FROM customers c    JOIN orders o ON c.id =
o.customer_id
JOIN order_items oi ON o.id = oi.order_id
JOIN products p ON oi.product_id = p.id
WHERE p.name = 'Product A';
```

```
mysql>    select DISTINCT c.name FROM customers c
    ->    JOIN orders o ON c.id = o.customer_id
    ->   JOIN order_items oi ON o.id = oi.order_id
    ->   JOIN products p ON oi.product_id = p.id
    ->   WHERE p.name = 'Product A';
+-------------+
| name        |
+-------------+
| John Doe    |
| Jane Smith  |
+-------------+
2 rows in set (0.01 sec)
```

7.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 orders o

JOIN

customers c ON o.customer_id = c.id;

```
mysql>
mysql> SELECT c.name AS customer_name, o.order_date FROM orders o JOIN customers c ON o.customer_id = c.id;
+---------------+------------+
| customer_name | order_date |
+---------------+------------+
| John Doe      | 2024-11-06 |
| Jane Smith    | 2024-10-22 |
| John Doe      | 2024-10-02 |
+---------------+------------+
3 rows in set (0.00 sec)
```

8.Retrieve the orders with a total amount greater than 150.00:

select * from orders where total_amount > 150.00;

```
mysql> select * from orders where total_amount > 150.00;
Empty set (0.00 sec)

mysql> select * from orders;
+----+-------------+------------+--------------+
| id | customer_id | order_date | total_amount |
+----+-------------+------------+--------------+
|  1 |           1 | 2024-11-06 |        60.00 |
|  2 |           2 | 2024-10-22 |        75.00 |
|  3 |           1 | 2024-10-02 |        80.00 |
+----+-------------+------------+--------------+
3 rows in set (0.00 sec)
```

9.Normalize the database by creating a separate table for order items and updating the orders table to reference the order_items table:

create table order_items (id INT AUTO_INCREMENT PRIMARY KEY, order_id INT, product_id INT, quantity INT,

FOREIGN KEY (order_id) REFERENCES orders(id),

FOREIGN KEY (product_id) REFERENCES products(id));

```
mysql> select * from order_items;
+----+----------+------------+----------+
| id | order_id | product_id | quantity |
+----+----------+------------+----------+
|  1 |        1 |          1 |        2 |
|  2 |        1 |          2 |        1 |
|  3 |        2 |          1 |        1 |
|  4 |        3 |          3 |        3 |
+----+----------+------------+----------+
4 rows in set (0.00 sec)
```

10.Retrieve the average total of all orders:

select AVG(total_amount) AS average_order_total FROM orders;

```
mysql> select AVG(total_amount) AS average_order_total FROM orders;
+---------------------+
| average_order_total |
+---------------------+
|           71.666667 |
+---------------------+
1 row in set (0.00 sec)
```