Database Schema

Table: Users. Stores information about users, including their roles (e.g., customer, admin).

```
mysql> CREATE DATABASE IF NOT EXISTS ECommercePlatform;
Query OK, 1 row affected (0.02 sec)

mysql> USE ECommercePlatform;
Database changed
mysql> CREATE TABLE IF NOT EXISTS Users (
    ->     user_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     username VARCHAR(255) NOT NULL UNIQUE,
    ->     email VARCHAR(255) NOT NULL UNIQUE,
    ->     password_hash VARCHAR(255) NOT NULL,
    ->     role ENUM('customer', 'admin') NOT NULL DEFAULT
'customer',
    ->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql>
```

Table: Products. Stores details about the products available for sale.
Table: Orders. Stores information about orders placed by users.

```
mysql> CREATE TABLE IF NOT EXISTS Products (
    ->     product_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     product_name VARCHAR(255) NOT NULL,
    ->     category VARCHAR(100) NOT NULL,
    ->     description TEXT,
    ->     price DECIMAL(10, 2) NOT NULL,
    ->     stock_quantity INT NOT NULL DEFAULT 0,
    ->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP O
N UPDATE CURRENT_TIMESTAMP,
    ->     CONSTRAINT chk_price CHECK (price >= 0),
    ->     CONSTRAINT chk_stock_quantity CHECK (stock_quant
ity >= 0)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE IF NOT EXISTS Orders (
    ->     order_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     user_id INT NOT NULL,
    ->     order_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     total_amount DECIMAL(12, 2) NOT NULL,
    ->     order_status ENUM('pending', 'processing', 'ship
ped', 'delivered', 'cancelled') NOT NULL DEFAULT 'pending',

    ->     shipping_address TEXT,
    ->     billing_address TEXT,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id)
ON DELETE RESTRICT ON UPDATE CASCADE,
    ->     CONSTRAINT chk_total_amount CHECK (total_amount
>= 0)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql>
```

Table: OrderDetails. Stores the individual items included in each order.
Table: Payments. Stores information about payments made for orders.

```
mysql> CREATE TABLE IF NOT EXISTS OrderDetails (
    ->     order_detail_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     order_id INT NOT NULL,
    ->     product_id INT NOT NULL,
    ->     quantity INT NOT NULL,
    ->     unit_price DECIMAL(10, 2) NOT NULL,
    ->     FOREIGN KEY (order_id) REFERENCES Orders(order_i
d) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     FOREIGN KEY (product_id) REFERENCES Products(pro
duct_id) ON DELETE RESTRICT ON UPDATE CASCADE,
    ->     CONSTRAINT chk_quantity CHECK (quantity > 0),
    ->     CONSTRAINT chk_unit_price CHECK (unit_price >= 0
)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE IF NOT EXISTS Payments (
    ->     payment_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     order_id INT NOT NULL UNIQUE,
    ->     payment_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
,
    ->     payment_method ENUM('credit_card', 'debit_card',
 'paypal', 'bank_transfer') NOT NULL,
    ->     amount DECIMAL(12, 2) NOT NULL,
    ->     payment_status ENUM('pending', 'completed', 'fai
led', 'refunded') NOT NULL DEFAULT 'pending',
    ->     transaction_id VARCHAR(255) UNIQUE,
    ->     FOREIGN KEY (order_id) REFERENCES Orders(order_i
d) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     CONSTRAINT chk_payment_amount CHECK (amount >= 0
)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql>
```

Table: Reviews. Stores customer reviews for products.

```
mysql> CREATE TABLE IF NOT EXISTS Reviews (
    ->     review_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     product_id INT NOT NULL,
    ->     user_id INT NOT NULL,
    ->     rating INT NOT NULL,
    ->     review_text TEXT,
    ->     review_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    ->     FOREIGN KEY (product_id) REFERENCES Products(pro
duct_id) ON DELETE CASCADE ON UPDATE CASCADE,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id)
ON DELETE CASCADE ON UPDATE CASCADE,
    ->     CONSTRAINT chk_rating CHECK (rating >= 1 AND rat
ing <= 5)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql>
```

Users: Stores user credentials and roles. password_hash should store a securely hashed password.
Products: Contains product information, including stock_quantity. category helps in filtering.
Orders: Header information for each order, linked to a user. order_status tracks the order's progress.

OrderDetails: Line items for each order, linking products to orders with specific quantities and the price at the time of purchase (unit_price).
Payments: Payment details for each order. payment_status tracks if the payment was successful.
Reviews: User feedback on products, including a rating (1-5).

Constraints:

- PRIMARY KEY uniquely identifies rows.
- FOREIGN KEY establishes relationships between tables. ON DELETE and ON UPDATE clauses define referential integrity actions.
- UNIQUE ensures certain columns have unique values (e.g., username, email).
- ENUM restricts values to a predefined set.
- CHECK constraints enforce specific conditions on data (e.g., price >= 0).
- DEFAULT CURRENT_TIMESTAMP automatically sets creation/update times.

Insert Sample Data

```
mysql> INSERT INTO Users (username, email, password_hash, role) VALUES
    -> ('john_doe', 'john.doe@example.com', 'hashed_password_123', 'customer'),
    -> ('jane_smith', 'jane.smith@example.com', 'hashed_password_456', 'customer'),
    -> ('admin_user', 'admin@example.com', 'hashed_admin_pass', 'admin'),
    -> ('sam_brown', 'sam.brown@example.com', 'hashed_password_789', 'customer'),
    -> ('lisa_green', 'lisa.green@example.com', 'hashed_password_abc', 'customer');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Products (product_name, category, description, price, stock_quantity) VALUES
    -> ('Laptop Pro 15', 'Electronics', 'High-performance laptop with 16GB RAM, 512GB SSD.', 1200.00, 50),
    -> ('Wireless Mouse', 'Electronics', 'Ergonomic wireless mouse with 5 buttons.', 25.00, 200),
    -> ('Mechanical Keyboard', 'Electronics', 'RGB Mechanical Keyboard with blue switches.', 75.00, 100),
    -> ('Python Programming Book', 'Books', 'Comprehensive guide to Python 3.', 45.00, 150),
    -> ('Coffee Maker Deluxe', 'Home Appliances', 'Automatic coffee maker with timer.', 60.00, 80),
    -> ('Smartphone X', 'Electronics', 'Latest generation smartphone with advanced camera.', 800.00, 70),
    -> ('The Art of SQL', 'Books', 'Master SQL from beginner to advanced.', 35.00, 120),
    -> ('Bluetooth Headphones', 'Electronics', 'Noise-cancelling over-ear Bluetooth headphones.', 150.00, 90),
    -> ('Yoga Mat Premium', 'Sports', 'Eco-friendly premium yoga mat.', 30.00, 200),
    -> ('Desktop Monitor 27"', 'Electronics', '27-inch 4K UHD Monitor.', 350.00, 40);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Orders (user_id, total_amount, order_status, shipping_address, billing_address) VALUES
    -> (1, 1225.00, 'delivered', '123 Main St, Anytown, USA', '123 Main St, Anytown, USA'),
    -> (1, 105.00, 'shipped', '123 Main St, Anytown, USA', '123 Main St, Anytown, USA');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Orders (user_id, total_amount, order_status, shipping_address, billing_address) VALUES
    -> (2, 80.00, 'processing', '456 Oak Ave, Otherville, USA', '456 Oak Ave, Otherville, USA'),
    -> (2, 950.00, 'pending', '456 Oak Ave, Otherville, USA', '456 Oak Ave, Otherville, USA');
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Orders (user_id, total_amount, order_status, shipping_address, billing_address) VALUES
    -> (4, 30.00, 'delivered', '789 Pine Ln, Sometown, USA', '789 Pine Ln, Sometown, USA');
Query OK, 1 row affected (0.00 sec)

mysql>
```

```
mysql> INSERT INTO OrderDetails (order_id, product_id, quantity, unit_price) VALUES
    -> (1, 1, 1, 1200.00),
    -> (1, 2, 1, 25.00);
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO OrderDetails (order_id, product_id, quantity, unit_price) VALUES
    -> (2, 3, 1, 75.00),
    -> (2, 4, 1, 45.00);
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO OrderDetails (order_id, product_id, quantity, unit_price) VALUES
    -> (3, 5, 1, 60.00),
    -> (3, 2, 1, 20.00);
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO OrderDetails (order_id, product_id, quantity, unit_price) VALUES
    -> (4, 6, 1, 800.00),
    -> (4, 8, 1, 150.00);
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO OrderDetails (order_id, product_id, quantity, unit_price) VALUES
    -> (5, 9, 1, 30.00);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO Payments (order_id, payment_method, amount, payment_status, transaction_id) VALUES
    -> (1, 'credit_card', 1225.00, 'completed', 'txn_1_cc_12345'),
    -> (2, 'paypal', 105.00, 'completed', 'txn_2_pp_67890'),
    -> (3, 'debit_card', 80.00, 'completed', 'txn_3_dc_abcde'),
    -> (4, 'credit_card', 950.00, 'pending', 'txn_4_cc_fghij'),
    -> (5, 'paypal', 30.00, 'completed', 'txn_5_pp_klmno');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
mysql> INSERT INTO Reviews (product_id, user_id, rating, review_text, review_date) VALUES
    -> (1, 1, 5, 'Excellent laptop, very fast and reliable!', '2025-01-10 10:00:00'),
    -> (1, 2, 4, 'Great value for the price, display is crisp.', '2025-01-12 14:30:00');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO Reviews (product_id, user_id, rating, review_text, review_date) VALUES
    -> (2, 1, 4, 'Comfortable mouse, good battery life.', '2025-01-11 09:00:00'),
    -> (2, 4, 5, 'Best wireless mouse I have used!', '2025-02-01 11:00:00');
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO Reviews (product_id, user_id, rating, review_text, review_date) VALUES
    -> (4, 2, 5, 'Very informative and well-written book for Python learners.', '2025-01-20 16:00:00');
Query OK, 1 row affected (0.00 sec)

mysql>
mysql> INSERT INTO Reviews (product_id, user_id, rating, review_text) VALUES
    -> (5, 1, 3, 'Makes good coffee, but a bit noisy.');
Query OK, 1 row affected (0.00 sec)

mysql>
mysql> INSERT INTO Reviews (product_id, user_id, rating, review_text, review_date) VALUES
    -> (6, 2, 5, 'Amazing camera and battery life!', '2025-03-05 12:00:00');
Query OK, 1 row affected (0.00 sec)

mysql>
```

Queries

- Retrieve the list of all products in a specific category.

```
mysql> SELECT product_id, product_name, description, price, stock_quantity
    -> FROM Products
    -> WHERE category = 'Electronics';
+------------+---------------------+-------------------------------------------------+---------+----------------+
| product_id | product_name        | description                                     | price   | stock_quantity |
+------------+---------------------+-------------------------------------------------+---------+----------------+
|          1 | Laptop Pro 15       | High-performance laptop with 16GB RAM, 512GB SSD. | 1200.00 |             50 |
|          2 | Wireless Mouse      | Ergonomic wireless mouse with 5 buttons.        |   25.00 |            200 |
|          3 | Mechanical Keyboard | RGB Mechanical Keyboard with blue switches.     |   75.00 |            100 |
|          6 | Smartphone X        | Latest generation smartphone with advanced camera. |  800.00 |             70 |
|          8 | Bluetooth Headphones | Noise-cancelling over-ear Bluetooth headphones. |  150.00 |             90 |
|         10 | Desktop Monitor 27" | 27-inch 4K UHD Monitor.                         |  350.00 |             40 |
+------------+---------------------+-------------------------------------------------+---------+----------------+
6 rows in set (0.01 sec)

mysql>
```

- Retrieve the details of a specific user by providing their user_id.

```
mysql> SELECT user_id, username, email, role, created_at
    -> FROM Users
    -> WHERE user_id = 1;
+---------+----------+-----------------------+----------+---------------------+
| user_id | username | email                 | role     | created_at          |
+---------+----------+-----------------------+----------+---------------------+
|       1 | john_doe | john.doe@example.com | customer | 2025-05-07 14:03:41 |
+---------+----------+-----------------------+----------+---------------------+
1 row in set (0.00 sec)

mysql>
```

- Retrieve the order history for a particular user.

```
mysql> SELECT
    ->     o.order_id,
    ->     o.order_date,
    ->     o.total_amount,
    ->     o.order_status,
    ->     o.shipping_address
    -> FROM Orders o
    -> WHERE o.user_id = 1
    -> ORDER BY o.order_date DESC;
+----------+---------------------+--------------+--------------+--------------------------+
| order_id | order_date          | total_amount | order_status | shipping_address         |
+----------+---------------------+--------------+--------------+--------------------------+
|        1 | 2025-05-07 14:04:23 |      1225.00 | delivered    | 123 Main St, Anytown, USA |
|        2 | 2025-05-07 14:04:23 |       105.00 | shipped      | 123 Main St, Anytown, USA |
+----------+---------------------+--------------+--------------+--------------------------+
2 rows in set (0.01 sec)

mysql>
```

- Retrieve the products in an order along with their quantities and prices.

```
mysql> SELECT
    ->     p.product_name,
    ->     od.quantity,
    ->     od.unit_price,
    ->     (od.quantity * od.unit_price) AS item_total
    -> FROM OrderDetails od
    -> JOIN Products p ON od.product_id = p.product_id
    -> WHERE od.order_id = 1;
+---------------+----------+------------+------------+
| product_name  | quantity | unit_price | item_total |
+---------------+----------+------------+------------+
| Laptop Pro 15 |        1 |    1200.00 |    1200.00 |
| Wireless Mouse |        1 |      25.00 |      25.00 |
+---------------+----------+------------+------------+
2 rows in set (0.01 sec)

mysql>
```

- Retrieve the average rating of a product.

```
mysql> SELECT
    ->     p.product_name,
    ->     AVG(r.rating) AS average_rating,
    ->     COUNT(r.review_id) AS number_of_reviews
    -> FROM Products p
    -> JOIN Reviews r ON p.product_id = r.product_id
    -> GROUP BY p.product_id, p.product_name
    -> ORDER BY average_rating DESC;
+-------------------------+----------------+--------------------+
| product_name            | average_rating | number_of_reviews  |
+-------------------------+----------------+--------------------+
| Python Programming Book |         5.0000 |                  1 |
| Smartphone X            |         5.0000 |                  1 |
| Laptop Pro 15           |         4.5000 |                  2 |
| Wireless Mouse          |         4.5000 |                  2 |
| Coffee Maker Deluxe     |         3.0000 |                  1 |
+-------------------------+----------------+--------------------+
5 rows in set (0.01 sec)
```

- Retrieve the total revenue for a given month.

```
mysql> SELECT
    ->     SUM(py.amount) AS total_revenue
    -> FROM Payments py
    -> JOIN Orders o ON py.order_id = o.order_id
    -> WHERE py.payment_status = 'completed'
    ->   AND o.order_status IN ('shipped', 'delivered')
    ->   AND YEAR(py.payment_date) = 2025
    ->   AND MONTH(py.payment_date) = 5;
+---------------+
| total_revenue |
+---------------+
|       1360.00 |
+---------------+
1 row in set (0.00 sec)

mysql>
```

Data Modification

- Add a new product to the inventory.

```
mysql> INSERT INTO Products (product_name, category, description, price, stock_quantity)
    -> VALUES ('Gaming Mouse Pro', 'Electronics', 'High DPI gaming mouse with customizable buttons.', 55.00, 75);
Query OK, 1 row affected (0.02 sec)

mysql>
```

- Place a new order for a user.

```
mysql> INSERT INTO Orders (user_id, total_amount, order_status, shipping_address, billing_address)
    -> VALUES (2, 95.00, 'pending', '456 Oak Ave, Otherville, USA', '456 Oak Ave, Otherville, USA');
Query OK, 1 row affected (0.00 sec)

mysql>
mysql> SET @new_order_id = LAST_INSERT_ID();
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> INSERT INTO OrderDetails (order_id, product_id, quantity, unit_price)
    -> VALUES (@new_order_id, 7, 1, 35.00),
    ->        (@new_order_id, 9, 2, 30.00);
Query OK, 2 rows affected (0.02 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
mysql> UPDATE Products SET stock_quantity = stock_quantity - 1 WHERE product_id = 7;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> UPDATE Products SET stock_quantity = stock_quantity - 2 WHERE product_id = 9;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
mysql> INSERT INTO Payments (order_id, payment_method, amount, payment_status, transaction_id)
    -> VALUES (@new_order_id, 'paypal', 95.00, 'pending', CONCAT('txn_new_', @new_order_id));
Query OK, 1 row affected (0.02 sec)

mysql>
```

- Update the stock quantity of a product.

```
mysql> UPDATE Products
    -> SET stock_quantity = stock_quantity + 50
    -> WHERE product_id = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

- Remove a user's review.

```
mysql> DELETE FROM Reviews
    -> WHERE review_id = 2;
Query OK, 1 row affected (0.00 sec)

mysql>
```

Complex Queries

- Identify the top-selling products.

```
mysql> SELECT
    ->     p.product_id,
    ->     p.product_name,
    ->     p.category,
    ->     SUM(od.quantity) AS total_quantity_sold
    -> FROM Products p
    -> JOIN OrderDetails od ON p.product_id = od.product_id
    -> JOIN Orders o ON od.order_id = o.order_id
    -> WHERE o.order_status IN ('shipped', 'delivered')
    -> GROUP BY p.product_id, p.product_name, p.category
    -> ORDER BY total_quantity_sold DESC
    -> LIMIT 5;
+------------+-------------------------+-------------+---------------------+
| product_id | product_name            | category    | total_quantity_sold |
+------------+-------------------------+-------------+---------------------+
|          1 | Laptop Pro 15           | Electronics |                   1 |
|          2 | Wireless Mouse          | Electronics |                   1 |
|          3 | Mechanical Keyboard     | Electronics |                   1 |
|          4 | Python Programming Book | Books       |                   1 |
|          9 | Yoga Mat Premium        | Sports      |                   1 |
+------------+-------------------------+-------------+---------------------+
5 rows in set (0.01 sec)
```

- Find users who have placed orders exceeding a certain amount.

```
mysql> SELECT
    ->     u.user_id,
    ->     u.username,
    ->     u.email,
    ->     SUM(o.total_amount) AS total_spent_by_user
    -> FROM Users u
    -> JOIN Orders o ON u.user_id = o.user_id
    -> WHERE o.order_status IN ('shipped', 'delivered')
    -> GROUP BY u.user_id, u.username, u.email
    -> HAVING SUM(o.total_amount) > 1000.00
    -> ORDER BY total_spent_by_user DESC;
+---------+----------+----------------------+---------------------+
| user_id | username | email                | total_spent_by_user |
+---------+----------+----------------------+---------------------+
|       1 | john_doe | john.doe@example.com |             1330.00 |
+---------+----------+----------------------+---------------------+
1 row in set (0.00 sec)
```

- Calculate the overall average rating for each product category.

```
mysql> SELECT
    ->     p.category,
    ->     AVG(r.rating) AS average_category_rating,
    ->     COUNT(DISTINCT p.product_id) AS products_in_category,
    ->     COUNT(r.review_id) AS total_reviews_in_category
    -> FROM Products p
    -> JOIN Reviews r ON p.product_id = r.product_id
    -> GROUP BY p.category
    -> ORDER BY average_category_rating DESC;
+-----------------+--------------------------+----------------------+---------------------------+
| category        | average_category_rating | products_in_category | total_reviews_in_category |
+-----------------+--------------------------+----------------------+---------------------------+
| Books           |                   5.0000 |                    1 |                         1 |
| Electronics     |                   5.0000 |                    3 |                         3 |
| Home Appliances |                   3.0000 |                    1 |                         1 |
+-----------------+--------------------------+----------------------+---------------------------+
3 rows in set (0.01 sec)
```

Advanced Topics

- Automatically update the order status based on order processing.

```
mysql> DELIMITER //
mysql>
mysql> CREATE TRIGGER AfterPaymentCompleteUpdateOrderStatus
    -> AFTER UPDATE ON Payments
    -> FOR EACH ROW
    -> BEGIN
    ->     IF NEW.payment_status = 'completed' AND OLD.payment_status <> 'completed' THEN
    ->         UPDATE Orders
    ->         SET order_status = 'processing'
    ->         WHERE order_id = NEW.order_id AND order_status = 'pending';
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
```

- Generate a report on the most active users.

```
mysql> CREATE PROCEDURE GetMostActiveUsers(IN limit_count INT)
    -> BEGIN
    ->     SELECT
    ->         u.user_id,
    ->         u.username,
    ->         u.email,
    ->         COUNT(o.order_id) AS total_orders_placed
    ->     FROM Users u
    ->     JOIN Orders o ON u.user_id = o.user_id
    ->     WHERE o.order_status IN ('shipped', 'delivered', 'processing') -- Consider active/completed orders
    ->     GROUP BY u.user_id, u.username, u.email
    ->     ORDER BY total_orders_placed DESC
    ->     LIMIT limit_count;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
```