Project name: E-Commerce Database Management System.

Objective:
The primary objective of this project is to design and implement a comprehensive relational database system to efficiently manage the various aspects of an e-commerce platform. This includes organizing and storing data related to users, products, categories, orders, order items, and product reviews. The system aims to facilitate seamless data retrieval and manipulation to support core business operations such as user management, product cataloging, order processing, and customer feedback collection. Additionally, the project seeks to demonstrate the use of SQL queries for generating insightful reports, including sales summaries, order tracking, and product ratings, thereby enhancing decision-making and improving overall operational efficiency in an e-commerce environment.

3. Introduction

E-commerce platforms require robust and efficient database systems to manage vast amounts of data generated by customers, products, and transactions. This project develops a relational database schema to represent key entities such as users, products, categories, orders, order items, and product reviews. The database supports essential operations like placing orders, categorizing products, maintaining user data, and analyzing sales and customer feedback through SQL queries.

4. Experimental Setup

- **Database:** MySQL Server (version 8.0 recommended)
- **Development Environment:** MySQL Workbench or any SQL IDE
- **Hardware:** Standard PC or laptop
- **Tools:** SQL editor for writing queries and managing the database

5. Entities and Description:

User: Stores customer details including user ID, username, email, and password.

Categories: Holds product category information such as category name and description.

Product: Contains product details including name, description, price, and category ID (foreign key).

Orders: Stores order information including order ID, user ID, order date, and total amount.

Order-Item: Details products ordered in each order, including product ID, quantity, and price per item.

Review: Contains customer reviews for products, including rating and comments.

6. Creating Tables and Inserting Data:

Creating Tables:

```
CREATE TABLE user (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    user_name VARCHAR(100) NOT NULL,
    email VARCHAR(50) UNIQUE,
    password VARCHAR(225)
);

CREATE TABLE categories (
    category_id INT PRIMARY KEY AUTO_INCREMENT,
    category_name VARCHAR(50) NOT NULL,
    category_description VARCHAR(100)
);

CREATE TABLE product (
    product_id INT PRIMARY KEY AUTO_INCREMENT,
    product_name VARCHAR(50) NOT NULL,
    product_description VARCHAR(100),
    product_price DECIMAL(10,2),
    category_id INT,
    FOREIGN KEY (category_id) REFERENCES categories(category_id)
```

```
);

CREATE TABLE orders (
    order_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    order_date DATETIME DEFAULT CURRENT_TIMESTAMP,
    total_amount DECIMAL(10,2),
    FOREIGN KEY (user_id) REFERENCES user(user_id)
);

CREATE TABLE order_item (
    item_id INT PRIMARY KEY AUTO_INCREMENT,
    order_id INT,
    product_id INT,
    quantity INT,
    price DECIMAL(10,2),
    FOREIGN KEY (order_id) REFERENCES orders(order_id),
    FOREIGN KEY (product_id) REFERENCES product(product_id)
);

CREATE TABLE review (
    review_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
```

```
    product_id INT,
    rating INT CHECK (rating BETWEEN 1 AND 5),
    comment VARCHAR(100),
    FOREIGN KEY (user_id) REFERENCES user(user_id),
    FOREIGN KEY (product_id) REFERENCES product(product_id)
);
```

Inserting simple data:

```
-- Users
INSERT INTO user (user_name, email, password) VALUES
('Bob', 'bob@example.com', 'pass456'),
('Charlie', 'charlie@example.com', 'charlie123'),
('David', 'david@example.com', 'david789'),
('Eve', 'eve@example.com', 'eve321'),
('Frank', 'frank@example.com', 'frank654'),
('Grace', 'grace@example.com', 'grace987'),
('Heidi', 'heidi@example.com', 'heidi111'),
('Ivan', 'ivan@example.com', 'ivan999'),
('Judy', 'judy@example.com', 'judy222'),
('Mallory', 'mallory@example.com', 'mallory333');

-- Categories
```

```
INSERT INTO categories (category_name, category_description) VALUES

('Home Appliances', 'Appliances for home use'),

('Books', 'Educational and fictional books'),

('Clothing', 'Men and Women clothing'),

('Furniture', 'Indoor and outdoor furniture'),

('Toys', 'Children toys and games'),

('Beauty', 'Cosmetics and skincare'),

('Sports', 'Sports equipment'),

('Stationery', 'School and office stationery'),

('Automotive', 'Car accessories and parts'),

('Grocery', 'Daily food and household items');


-- Products (with category_id specified)

INSERT INTO product (product_name, product_description, product_price, category_id) VALUES

('Refrigerator', 'Double door fridge', 499.99, 1),

('Washing Machine', 'Top load washer', 350.00, 1),

('Notebook', '100 pages ruled', 2.50, 8),

('Chair', 'Office chair', 45.99, 4),

('Lipstick', 'Red matte lipstick', 12.00, 6),

('Football', 'Size 5 standard football', 22.00, 7),

('T-Shirt', 'Cotton round-neck', 8.99, 3),

('Toy Car', 'Remote control car', 15.00, 5),

('Engine Oil', '5W-30 full synthetic', 25.00, 9),
```

('Rice Bag', '5kg premium rice', 6.99, 10);

-- Orders

INSERT INTO orders (user_id, total_amount) VALUES

(2, 45.99),

(3, 499.99),

(4, 12.00),

(5, 15.00),

(6, 8.99),

(7, 25.00),

(8, 6.99),

(9, 22.00),

(10, 350.00),

(1, 2.50);

-- Order Items

INSERT INTO order_item (order_id, product_id, quantity, price) VALUES

(2, 4, 1, 45.99),

(3, 1, 1, 499.99),

(4, 5, 1, 12.00),

(5, 8, 1, 15.00),

(6, 7, 1, 8.99),

(7, 9, 1, 25.00),

(8, 10, 1, 6.99),

(9, 6, 1, 22.00),

(10, 2, 1, 350.00);

-- Reviews

INSERT INTO review (user_id, product_id, rating, comment) VALUES

(2, 4, 4, 'Comfortable chair.'),

(3, 1, 5, 'Keeps everything cold!'),

(4, 5, 3, 'Good, but dries out fast.'),

(5, 8, 4, 'Fun toy for kids.'),

(6, 7, 5, 'Soft and comfortable.'),

(7, 9, 5, 'Works great for my car.'),

(8, 10, 4, 'Good quality rice.'),

(9, 6, 5, 'Perfect for matches.'),

(10, 2, 4, 'Cleans clothes well.'),

(1, 3, 5, 'Very useful notebook.');

7. Some Essential SQL Queries

- **List all orders with customer names:**

  SELECT o.order_id, u.user_name, o.order_date, o.total_amount

  FROM orders o

  JOIN user u ON o.user_id = u.user_id;

  Output:

order_id	user_name	order_date	total_amount
1	Charlie	2025-05-25 16:05:39	45.99
2	David	2025-05-25 16:05:39	499.99
3	Eve	2025-05-25 16:05:39	12.00
4	Frank	2025-05-25 16:05:39	15.00
5	Grace	2025-05-25 16:05:39	8.99
6	Heidi	2025-05-25 16:05:39	25.00
7	Ivan	2025-05-25 16:05:39	6.99
8	Judy	2025-05-25 16:05:39	22.00
9	Mallory	2025-05-25 16:05:39	350.00
10	Bob	2025-05-25 16:05:39	2.50

- **Show products with their category names:**

  SELECT p.product_name, c.category_name, p.product_price

  FROM product p

  JOIN categories c ON p.category_id = c.category_id;

  Output:

user_name	total_orders
Bob	1
Charlie	1
David	1
Eve	1
Frank	1
Grace	1
Heidi	1
Ivan	1
Judy	1
Mallory	1

- **Count total orders per user:**

  SELECT u.user_name, COUNT(o.order_id) AS total_orders

  FROM user u

  LEFT JOIN orders o ON u.user_id = o.user_id

  GROUP BY u.user_id;

product_name	total_sales
Refrigerator	499.99
Washing Machine	350.00
Chair	45.99
Engine Oil	25.00
Football	22.00
Toy Car	15.00
Lipstick	12.00
T-Shirt	8.99
Rice Bag	6.99

- **Calculate total sales per product:**

  SELECT p.product_name, SUM(oi.price * oi.quantity) AS total_sales
  FROM order_item oi
  JOIN product p ON oi.product_id = p.product_id
  GROUP BY p.product_id
  ORDER BY total_sales DESC;

  Output:

product_name	total_sales
Refrigerator	499.99
Washing Machine	350.00
Chair	45.99
Engine Oil	25.00
Football	22.00
Toy Car	15.00
Lipstick	12.00
T-Shirt	8.99
Rice Bag	6.99

- **Find average rating of each product:**

  SELECT p.product_name, AVG(r.rating) AS avg_rating

  FROM product p

  JOIN review r ON p.product_id = r.product_id

  GROUP BY p.product_id

  ORDER BY avg_rating DESC;

  Output:

product_name	avg_rating
Refrigerator	5.0000
Notebook	5.0000
Football	5.0000
T-Shirt	5.0000
Engine Oil	5.0000
Washing Machine	4.0000
Chair	4.0000
Toy Car	4.0000
Rice Bag	4.0000
Lipstick	3.0000

- **List products with 5-star reviews:**

```
SELECT DISTINCT p.product_name
FROM product p
JOIN review r ON p.product_id = r.product_id
WHERE r.rating = 5;
```

Output:

product_name
Refrigerator
T-Shirt
Engine Oil
Football
Notebook

- **Identify products not yet ordered:**

```
SELECT p.product_name

FROM product p
```

LEFT JOIN order_item oi ON p.product_id = oi.product_id

WHERE oi.product_id IS NULL;

product_name
Notebook

- **Count number of reviews per product:**

```
SELECT p.product_name, COUNT(r.review_id) AS total_reviews
FROM product p
LEFT JOIN review r ON p.product_id = r.product_id
GROUP BY p.product_id;
```

Output:

product_name	total_reviews
Refrigerator	1
Washing Machine	1
Notebook	1
Chair	1
Lipstick	1
Football	1
T-Shirt	1
Toy Car	1
Engine Oil	1
Rice Bag	1

Conclusion:

This project successfully demonstrates the design and implementation of a structured and efficient relational database system tailored for an e-commerce platform. By integrating multiple interconnected tables such as users, products, categories, orders, and reviews, the system ensures smooth handling of essential operations like product listing, order placement, inventory tracking,

and customer feedback. The use of SQL queries further allows for effective data analysis, reporting, and decision-making. Overall, this project not only reinforces foundational database management concepts but also highlights the practical application of SQL in solving real-world problems within the domain of online retail.