CREATE DATABASE IF NOT EXISTS restaurant;

USE restaurant;

-- Create the items table

CREATE TABLE IF NOT EXISTS items (

item\_id INT PRIMARY KEY AUTO\_INCREMENT,

item\_name VARCHAR(255) NOT NULL,

item\_price DECIMAL(10, 2) NOT NULL

);

-- Create the tables table

CREATE TABLE IF NOT EXISTS tables (

table\_id INT PRIMARY KEY AUTO\_INCREMENT,

table\_name VARCHAR(255) NOT NULL

);

-- Create the users table

CREATE TABLE IF NOT EXISTS users (

user\_id INT PRIMARY KEY AUTO\_INCREMENT,

username VARCHAR(255) NOT NULL,

email VARCHAR(255) NOT NULL

);

-- Create the payments table

CREATE TABLE IF NOT EXISTS payments (

payment\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT,

table\_id INT,

total\_amount DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (user\_id) REFERENCES users(user\_id),

FOREIGN KEY (table\_id) REFERENCES tables(table\_id)

);

-- Sample data for items table

INSERT INTO items (item\_name, item\_price) VALUES

('Dish 1', 10.99),

('Dish 2', 15.99);

-- Sample data for tables table

INSERT INTO tables (table\_name) VALUES

('Table 1'),

('Table 2');

-- Sample data for users table

INSERT INTO users (username, email) VALUES

('User1', 'user1@example.com'),

('User2', 'user2@example.com');

-- Sample data for payments table

INSERT INTO payments (user\_id, table\_id, total\_amount) VALUES

(1, 1, 26.98),

(2, 2, 31.98);

-- 1. Retrieve all items from the items table:

SELECT \* FROM items;

-- 2. Retrieve all tables from the tables table:

SELECT \* FROM tables;

-- 3. Retrieve all users from the users table:

SELECT \* FROM users;

-- 4. Retrieve all payments from the payments table:

SELECT \* FROM payments;

-- 5. Retrieve the total number of items:

SELECT COUNT(\*) FROM items;

-- 6. Retrieve the total number of tables:

SELECT COUNT(\*) FROM tables;

-- 7. Retrieve the total number of users:

SELECT COUNT(\*) FROM users;

-- 8. Retrieve the total number of payments:

SELECT COUNT(\*) FROM payments;

-- 9. Retrieve items with prices less than 15.00:

SELECT \* FROM items WHERE item\_price < 15.00;

-- 10. Retrieve tables with names containing 'Table':

SELECT \* FROM tables WHERE table\_name LIKE '%Table%';

-- 11. Retrieve users with usernames starting with 'User':

SELECT \* FROM users WHERE username LIKE 'User%';

-- 12. Retrieve payments with total amount greater than 30.00:

SELECT \* FROM payments WHERE total\_amount > 30.00;

-- 13. Retrieve the item with the highest price:

SELECT \* FROM items ORDER BY item\_price DESC LIMIT 1;

-- 14. Retrieve the table with the lowest ID:

SELECT \* FROM tables ORDER BY table\_id LIMIT 1;

-- 15. Retrieve the user with the highest ID:

SELECT \* FROM users ORDER BY user\_id DESC LIMIT 1;

-- 16. Retrieve the payment with the lowest total amount:

SELECT \* FROM payments ORDER BY total\_amount LIMIT 1;

-- 17. Update the price of 'Dish 1' to 12.99:

UPDATE items SET item\_price = 12.99 WHERE item\_name = 'Dish 1';

-- 18. Update the name of 'Table 1' to 'Updated Table':

UPDATE tables SET table\_name = 'Updated Table' WHERE table\_id = 1;

-- 19. Update the email of 'User1' to 'updated\_user1@example.com':

UPDATE users SET email = 'updated\_user1@example.com' WHERE username = 'User1';

-- 20. Delete the item with item\_id 2:

DELETE FROM items WHERE item\_id = 2;

-- 21. Delete the table with table\_id 2:

DELETE FROM tables WHERE table\_id = 2;

-- 22. Delete the user with user\_id 2:

DELETE FROM users WHERE user\_id = 2;

-- 23. Delete the payment with payment\_id 2:

DELETE FROM payments WHERE payment\_id = 2;

-- 24. Retrieve items and their corresponding prices in ascending order:

SELECT item\_name, item\_price FROM items ORDER BY item\_price ASC;

-- 25. Retrieve users and their email addresses in alphabetical order:

SELECT username, email FROM users ORDER BY username;

-- 26. Retrieve payments with the user and table information:

SELECT payments.payment\_id, users.username, tables.table\_name, payments.total\_amount

FROM payments

JOIN users ON payments.user\_id = users.user\_id

JOIN tables ON payments.table\_id = tables.table\_id;

-- 27. Calculate the total revenue:

SELECT SUM(total\_amount) AS total\_revenue FROM payments;

-- 28. Retrieve the average item price:

SELECT AVG(item\_price) AS avg\_item\_price FROM items;

-- 29. Retrieve the user who spent the most:

SELECT users.username, MAX(payments.total\_amount) AS max\_spent

FROM users

JOIN payments ON users.user\_id = payments.user\_id;

-- 30. Retrieve items and their prices with a price between 10.00 and 20.00:

SELECT \* FROM items WHERE item\_price BETWEEN 10.00 AND 20.00;