NAME : FARHAS YS

MOB NO : 8951345782

TASK :03

Objective: Use SQL queries to extract and analyze data from a real-world e-commerce database.

Tools Used:

- SQLite (compatible with MySQL/PostgreSQL)

Dataset: Simulated E-commerce SQL Database with the following tables:

- customers
- orders
- order_items
- products
- categories

```
- Drop tables if they exist
DROP TABLE IF EXISTS order_items;
DROP TABLE IF EXISTS orders;
DROP TABLE IF EXISTS products;
DROP TABLE IF EXISTS customers;
DROP TABLE IF EXISTS categories;

-- Customers Table
CREATE TABLE customers (
    customer_id INTEGER PRIMARY KEY,
    first_name TEXT,
    last_name TEXT,
    email TEXT,
    join_date DATE
);

-- Categories Table
CREATE TABLE categories (
    category_id INTEGER PRIMARY KEY,
    name TEXT
);

-- Products Table
```

```
CREATE TABLE products (
    product_id INTEGER PRIMARY KEY,
    name TEXT,
    price DECIMAL(10, 2),
    category_id INTEGER,
    FOREIGN KEY (category_id) REFERENCES categories(category_id)
);

-- Orders Table
CREATE TABLE orders (
    order_id INTEGER PRIMARY KEY,
    customer_id INTEGER,
    order_date DATE,
    total_amount DECIMAL(10, 2),
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);

-- Order Items Table
CREATE TABLE order_items (
    order_item_id INTEGER PRIMARY KEY,
    order_id INTEGER,
    product_id INTEGER,
    quantity INTEGER,
    price DECIMAL(10, 2),
    FOREIGN KEY (order_id) REFERENCES orders(order_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id)
);
```

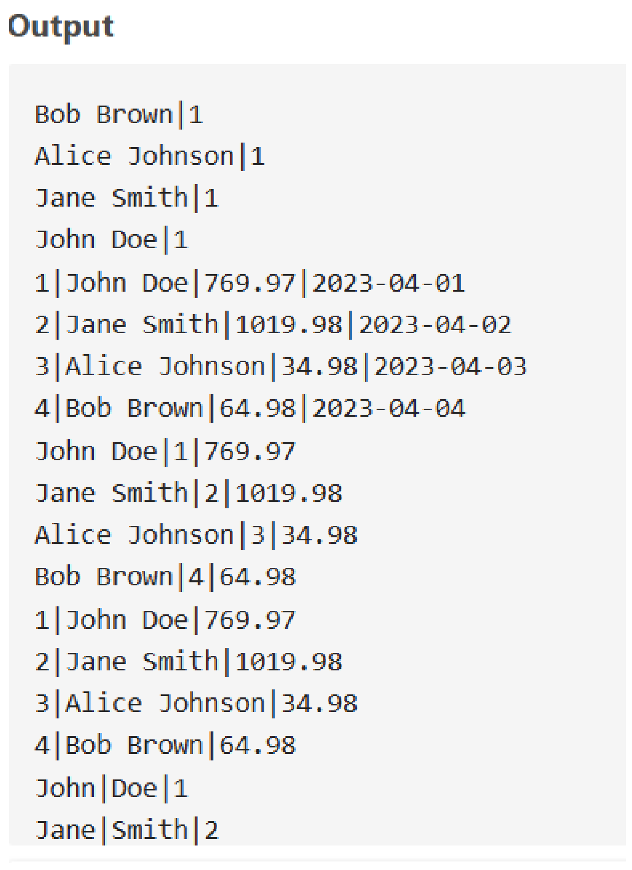
```
INSERT INTO customers VALUES
(1, 'John', 'Doe', 'john.doe@example.com', '2023-01-10'),
(2, 'Jane', 'Smith', 'jane.smith@example.com', '2023-02-14'),
(3, 'Alice', 'Johnson', 'alice.johnson@example.com', '2023-03-20'),
(4, 'Bob', 'Brown', 'bob.brown@example.com', '2023-04-05');

INSERT INTO categories VALUES
(1, 'Electronics'),
(2, 'Books'),
(3, 'Clothing'),
(4, 'Home & Kitchen');

INSERT INTO products VALUES
(1, 'Smartphone', 699.99, 1),
(2, 'Laptop', 999.99, 1),
(3, 'Fiction Novel', 19.99, 2),
(4, 'T-Shirt', 14.99, 3),
(5, 'Blender', 49.99, 4);

INSERT INTO orders VALUES
(1, 1, '2023-04-01', 769.97),
(2, 2, '2023-04-02', 1019.98),
(3, 3, '2023-04-03', 34.98),
(4, 4, '2023-04-04', 64.98);
```

```
INSERT INTO order_items VALUES
(1, 1, 1, 1, 699.99),
(2, 1, 4, 1, 14.99),
(3, 1, 3, 1, 19.99),
(4, 2, 2, 1, 999.99),
(5, 2, 4, 1, 14.99),
(6, 3, 3, 2, 19.99),
(7, 4, 4, 1, 14.99),
(8, 4, 5, 1, 49.99);


-- 1. SELECT + WHERE + ORDER BY + GROUP BY
SELECT c.first_name || ' ' || c.last_name AS customer_name,
COUNT(o.order_id) AS total_orders
FROM customers c
JOIN orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id
ORDER BY total_orders DESC;

-- 2. INNER JOIN
SELECT o.order_id, c.first_name || ' ' || c.last_name AS customer_name,
o.total_amount, o.order_date
FROM orders o
INNER JOIN customers c ON o.customer_id = c.customer_id;

-- 3. LEFT JOIN
SELECT c.first_name || ' ' || c.last_name AS customer_name, o.order_id,
o.total_amount
```

```
FROM customers c

LEFT JOIN orders o ON c.customer_id = o.customer_id;


-- 4. RIGHT JOIN (emulated with LEFT JOIN)

SELECT o.order_id, c.first_name || ' ' || c.last_name AS customer_name,
o.total_amount

FROM orders o

LEFT JOIN customers c ON o.customer_id = c.customer_id;


-- 5. Subquery: Customers who spent above average

SELECT first_name, last_name, customer_id

FROM customers

WHERE customer_id IN (

   SELECT customer_id

   FROM orders

   GROUP BY customer_id

   HAVING AVG(total_amount) > (SELECT AVG(total_amount) FROM
orders)

);


-- 6. Aggregates: SUM and AVG

SELECT

   SUM(total_amount) AS total_revenue,

   AVG(total_amount) AS average_order_value

FROM orders;


-- Top Selling Products (by revenue)

CREATE VIEW top_selling_products AS
```

```
SELECT
    p.name AS product_name,
    SUM(oi.quantity * oi.price) AS total_revenue
FROM order_items oi
JOIN products p ON oi.product_id = p.product_id
GROUP BY p.name
ORDER BY total_revenue DESC;


CREATE INDEX idx_orders_customer_id ON orders(customer_id);
CREATE INDEX idx_order_items_product_id ON order_items(product_id);
CREATE INDEX idx_products_category_id ON products(category_id);
```

OUTPUT:



Output

```
Bob Brown|1
Alice Johnson|1
Jane Smith|1
John Doe|1
1|John Doe|769.97|2023-04-01
2|Jane Smith|1019.98|2023-04-02
3|Alice Johnson|34.98|2023-04-03
4|Bob Brown|64.98|2023-04-04
John Doe|1|769.97
Jane Smith|2|1019.98
Alice Johnson|3|34.98
Bob Brown|4|64.98
1|John Doe|769.97
2|Jane Smith|1019.98
3|Alice Johnson|34.98
4|Bob Brown|64.98
John|Doe|1
Jane|Smith|2
```

```
Bob Brown|1
Alice Johnson|1
Jane Smith|1
John Doe|1
1|John Doe|769.97|2023-04-01
2|Jane Smith|1019.98|2023-04-02
3|Alice Johnson|34.98|2023-04-03
4|Bob Brown|64.98|2023-04-04
John Doe|1|769.97
Jane Smith|2|1019.98
Alice Johnson|3|34.98
Bob Brown|4|64.98
```

```
1|John Doe|769.97
2|Jane Smith|1019.98
3|Alice Johnson|34.98
4|Bob Brown|64.98
John|Doe|1
Jane|Smith|2
1889.91|472.4775

[Execution complete with exit code 0]
```