SQL Data Analysis Report (Task 3)

Ecommerce Database

1. Introduction

This document contains SQL queries performed on an ecommerce database to demonstrate data extraction, manipulation, and analysis. The database consists of four tables:

- **Customers** (customer details)
- **Products** (product catalog)
- **Orders** (order transactions)
- **Order_items** (individual items in orders)

2. Database Schema & Sample Data

- **Tables Created**

```
mysql> CREATE TABLE customers (
    ->     customer_id INT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     email VARCHAR(100),
    ->     city VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.11 sec)

mysql>
mysql> CREATE TABLE products (
    ->     product_id INT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     category VARCHAR(50),
    ->     price DECIMAL(10,2)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> CREATE TABLE orders (
    ->     order_id INT PRIMARY KEY,
    ->     customer_id INT,
    ->     order_date DATE,
    ->     total_amount DECIMAL(10,2),
    ->     FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql>
mysql> CREATE TABLE order_items (
    ->     item_id INT PRIMARY KEY,
    ->     order_id INT,
    ->     product_id INT,
    ->     quantity INT,
    ->     FOREIGN KEY (order_id) REFERENCES orders(order_id),
    ->     FOREIGN KEY (product_id) REFERENCES products(product_id)
    -> );
Query OK, 0 rows affected (0.05 sec)
```

- **Sample Data Inserted**

```
mysql> INSERT INTO customers VALUES
    -> (1, 'Ananya Sharma', 'ananya@example.com', 'Mumbai'),
    -> (2, 'Raj Verma', 'raj@example.com', 'Delhi'),
    -> (3, 'Simran Kaur', 'simran@example.com', 'Bangalore');
Query OK, 3 rows affected (0.02 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO products VALUES
    -> (101, 'Laptop', 'Electronics', 55000.00),
    -> (102, 'Smartphone', 'Electronics', 22000.00),
    -> (103, 'Book', 'Stationery', 500.00);
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO orders VALUES
    -> (1001, 1, '2024-05-10', 77000.00),
    -> (1002, 2, '2024-05-12', 500.00),
    -> (1003, 3, '2024-05-15', 22000.00);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
mysql> INSERT INTO order_items VALUES
    -> (1, 1001, 101, 1),
    -> (2, 1001, 102, 1),
    -> (3, 1002, 103, 1),
    -> (4, 1003, 102, 1);
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

3. SQL Queries & Analysis

3.1 Basic Data Retrieval

- **Get all customers from Delhi**

```
mysql> SELECT * FROM customers WHERE city = 'Delhi';
+-------------+-----------+-----------------+-------+
| customer_id | name      | email           | city  |
+-------------+-----------+-----------------+-------+
|           2 | Raj Verma | raj@example.com | Delhi |
+-------------+-----------+-----------------+-------+
1 row in set (0.00 sec)
```

- **List products by price (highest to lowest)**

```
mysql> SELECT * FROM products ORDER BY price DESC;
+------------+------------+-------------+----------+
| product_id | name       | category    | price    |
+------------+------------+-------------+----------+
|        101 | Laptop     | Electronics | 55000.00 |
|        102 | Smartphone | Electronics | 22000.00 |
|        103 | Book       | Stationery  |   500.00 |
+------------+------------+-------------+----------+
3 rows in set (0.00 sec)
```

3.2 Aggregations & Grouping

- **Total amount spent by each customer**

```
mysql> SELECT customer_id, SUM(total_amount) AS total_spent
    -> FROM orders
    -> GROUP BY customer_id;
+-------------+-------------+
| customer_id | total_spent |
+-------------+-------------+
|           1 |    77000.00 |
|           2 |      500.00 |
|           3 |    22000.00 |
+-------------+-------------+
3 rows in set (0.00 sec)
```

3.3 Joins (Combining Tables)

- **Order details with customer names**

```
mysql> SELECT o.order_id, c.name AS customer_name, o.order_date, o.total_amount
    -> FROM orders o
    -> INNER JOIN customers c ON o.customer_id = c.customer_id;
+----------+---------------+------------+--------------+
| order_id | customer_name | order_date | total_amount |
+----------+---------------+------------+--------------+
|     1001 | Ananya Sharma | 2024-05-10 |     77000.00 |
|     1002 | Raj Verma     | 2024-05-12 |       500.00 |
|     1003 | Simran Kaur   | 2024-05-15 |     22000.00 |
+----------+---------------+------------+--------------+
3 rows in set (0.00 sec)
```

3.4 Subqueries

- **Customers who spent more than average**

```
mysql> SELECT * FROM customers
    -> WHERE customer_id IN (
    ->     SELECT customer_id FROM orders
    ->     GROUP BY customer_id
    ->     HAVING SUM(total_amount) > (
    ->         SELECT AVG(total_amount) FROM orders
    ->     )
    -> );
+-------------+--------------+--------------------+--------+
| customer_id | name         | email              | city   |
+-------------+--------------+--------------------+--------+
|           1 | Ananya Sharma | ananya@example.com | Mumbai |
+-------------+--------------+--------------------+--------+
1 row in set (0.01 sec)
```

3.5 Views (Stored Queries)

- **Create a view for order summaries**

```
mysql> CREATE VIEW order_summary AS
    -> SELECT o.order_id, c.name, o.total_amount
    -> FROM orders o
    -> JOIN customers c ON o.customer_id = c.customer_id;
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM order_summary;
+----------+--------------+--------------+
| order_id | name         | total_amount |
+----------+--------------+--------------+
|     1001 | Ananya Sharma |     77000.00 |
|     1002 | Raj Verma     |       500.00 |
|     1003 | Simran Kaur   |     22000.00 |
+----------+--------------+--------------+
3 rows in set (0.00 sec)
```

3.6 Handling NULL Values

- **Replace NULL emails with "No Email"**

```
mysql> SELECT IFNULL(email, 'No Email') FROM customers;
+--------------------------+
| IFNULL(email, 'No Email') |
+--------------------------+
| ananya@example.com       |
| raj@example.com          |
| simran@example.com       |
+--------------------------+
3 rows in set (0.00 sec)
```

4. Conclusion

This report demonstrates SQL skills including:

✓ Basic queries (SELECT, WHERE, ORDER BY)

✓ Aggregations (SUM, AVG, GROUP BY)

✓ Joins (INNER JOIN)

✓ Subqueries (nested queries)

✓ Views (stored SQL queries)

✓ NULL handling (IFNULL)

Prepared by: Chirag Sawant

Date: 29/05/2005