Internship Task Submission

Task 6: Sales Trend Analysis Using Aggregations

Internship Domain: Data Analytics / SQL

Submitted by: Kajal Prajapati

Date: [2025-06-11]

Objective:

Analyze monthly revenue and order volume using SQL aggregations.

Dataset: Online Sales Dataset from Kaggle

Table used: `Online Sales Data raw dataset.csv`

Columns used: `order_date`, `total_revenue`, `transaction_id`

Internship Project — Analyzing Monthly Sales Trends using SQL Aggregations

Introduction

Objective:

To analyze monthly revenue and order volume from online sales data using SQL aggregation functions such as `SUM()` and `COUNT()` along with `GROUP BY` and `ORDER BY`.

Tools Used:

- MySQL 8.0
- SQL
- Dataset: Online Sales Data from Kaggle (241 records)

Output:

The analysis helped identify trends in revenue and sales volume over 8 months.

Dataset Columns Used:

- order_date
- total_revenue
- transaction_id

SQL Query Section (Technical Details)

Database Creation & Table Structure:

```
CREATE DATABASE online_sales_analysis;
USE online_sales_analysis;

CREATE TABLE online_sales(
  transaction_id INT,
  order_date DATE,
  product_category VARCHAR(100),
  product_name VARCHAR(100),
  unit_sold INT,
  unit_price DECIMAL(10,2),
  total_revenue DECIMAL(12,2),
  region VARCHAR(100),
  payment_method VARCHAR(50)
);
```

SQL Query Section (Technical Details)

Data Import:

```
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL
Server 8.0\\Uploads\\Online Sales Data.csv'

INTO TABLE online_sales

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS

(transaction_id, @order_date, product_category,
product_name, unit_sold, unit_price, total_revenue,
region, payment_method)

SET order_date = STR_TO_DATE(@order_date,
'%m/%d/%Y');
```

SQL Query Section (Technical Details)

Final Aggregation Query:

```
SELECT
    YEAR(order_date) AS order_year,
    MONTH(order_date) AS order_month,
    SUM(total_revenue) AS total_revenue,
    COUNT(DISTINCT transaction_id) AS total_orders
FROM
    online_sales
GROUP BY
    order_year, order_month
ORDER BY
    order_year, order_month;
```

Result Table

order_year	order_month	total_revenue	total_orders
2024	1	14548.32	31
2024	2	10803.37	29
2024	3	12849.24	31
2024	4	12451.69	30
2024	5	8455.49	31
2024	6	7384.55	30
2024	7	6797.08	31
2024	8	7278.11	27

Conclusion:

The dataset showed 8 months of sales data. This SQL aggregation helped identify peak order volume and revenue months.

This analysis technique is helpful for business decisions related to sales seasonality and inventory planning.