Summary Report: Monthly Sales Trend Analysis

This report summarizes the key findings from the sales trend analysis performed on the "Online Sales Data" dataset. The objective was to analyze monthly revenue and order volume to identify trends over time.

Methodology

The raw sales data from Online Sales Data.csv was first loaded into an SQLite database table named online_sales. During this process, the Date, Total Revenue, and Transaction ID columns from the CSV were mapped and renamed to order_date, amount, and order_id respectively, to facilitate standard SQL operations.

A SQL query was then executed to:

- Extract the year and month from the order_date.
- Calculate the total amount (revenue) for each month.
- Count the number of unique order_id values (volume) for each month.
- Group these aggregations by year and month.
- Order the results chronologically.

Key Findings

The analysis successfully generated a monthly breakdown of sales performance. The resulting table provides a clear view of:

- **Monthly Revenue (total_revenue):** Shows the total sales amount generated in each month. This metric is crucial for understanding income trends and financial performance over time.
- **Monthly Order Volume (total_volume):** Indicates the number of distinct orders placed in each month. This metric helps in assessing customer activity and operational load.
- **Sales Trend over Time:** By observing the values across different sales_year_month periods, one can identify periods of growth, decline, or stability in both revenue and order volume. This information is vital for strategic planning, inventory management, and marketing efforts.

Key Learnings:

1. **Understanding and Interpreting Data Requirements:** I learned to break down a request for "Sales Trend Analysis" into specific technical requirements, such as calculating monthly revenue and order volume, and identifying the necessary data points (date, amount, order ID).
2. **Data Ingestion and Transformation:**
    - **CSV to Database Loading:** I gained practical experience in loading raw .csv data into an SQLite database, a common first step in many data analysis workflows.
    - **Column Mapping and Renaming:** I understood the importance of correctly identifying and mapping source CSV column names (Date, Total Revenue, Transaction ID) to logical, query-friendly database column names (order_date, amount, order_id). This highlighted the need for data cleaning and preparation before analysis.

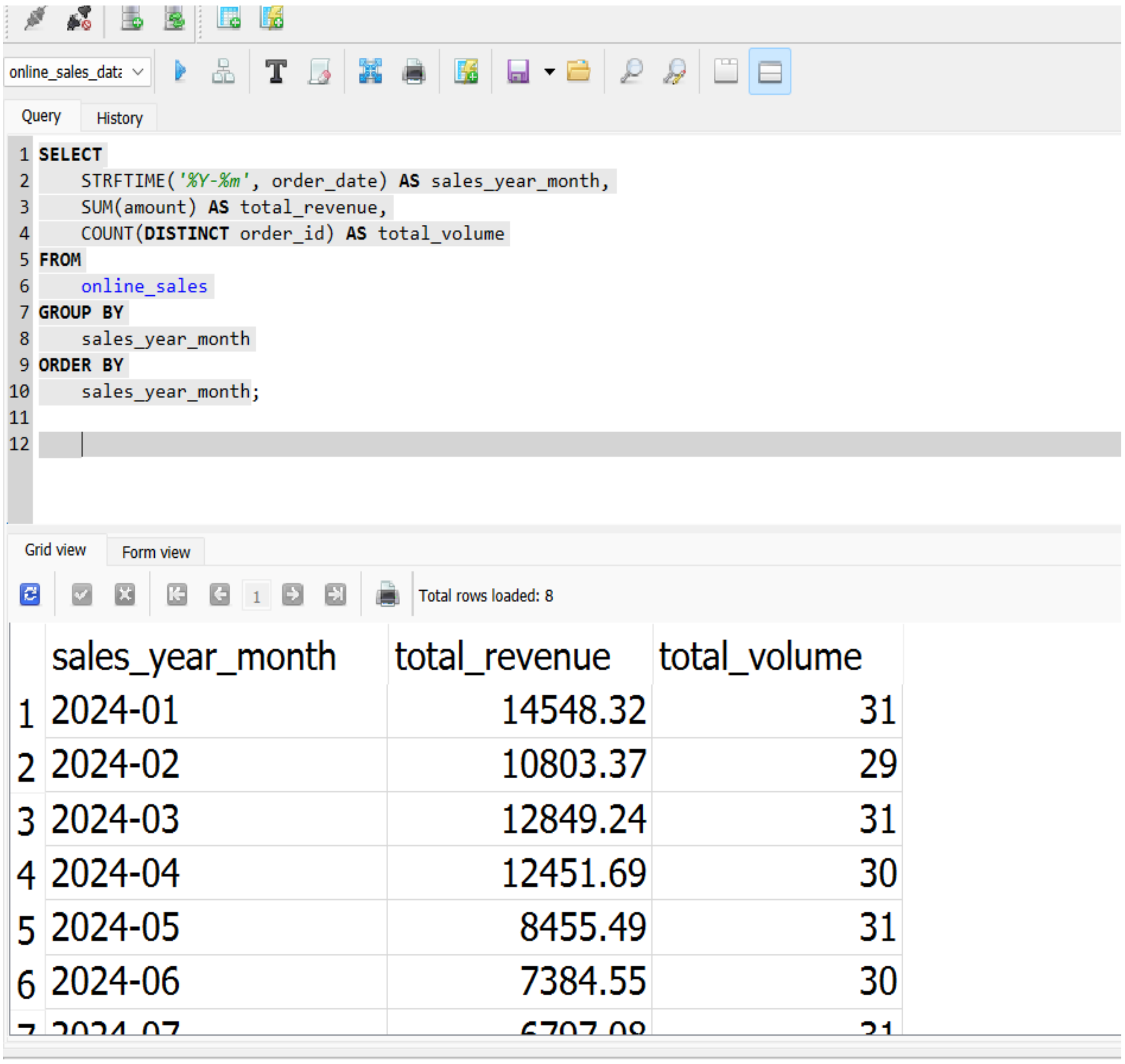
   - **Data Type Conversion:** The necessity of converting date strings in the CSV to proper datetime objects (e.g., pd.to_datetime in Python) was crucial for accurate date-based SQL functions like STRFTIME.

3. **SQL Aggregation and Time-Series Analysis:**
   - **Core Aggregation Functions:** I solidified my understanding and application of SUM() for calculating total revenue and COUNT(DISTINCT) for determining unique order volume.
   - **Grouping Data by Time:** The use of STRFTIME('%Y-%m', column) combined with GROUP BY was a key learning for aggregating data chronologically (by year and month), which is fundamental for any trend analysis.
   - **Ordering Results:** Applying ORDER BY ensures that the time-series data is presented in a logical, chronological sequence, making trends easier to visualize and interpret.
   - **Filtering Time Periods:** I learned how to use the WHERE clause with STRFTIME to filter data for specific years or months, allowing for focused analysis of particular periods.
4. **Debugging and Problem Solving:**
   - Encountering errors like "no such table" and "no such column" provided hands-on experience in debugging data pipeline issues. This involved systematically checking if the table existed, if the data was loaded correctly, and if the column names in the query matched those in the database. This reinforced the iterative nature of data analysis.
5. **Delivering Actionable Insights:**
   - The entire process culminated in generating a "results table" (as seen in the screenshot) that directly answers the objective of monthly revenue and order volume.
   - I also learned how to summarize the analysis, key findings, and technical steps into a comprehensive README and a summary report, essential skills for communicating data projects.

Overall, this task provided a practical foundation in using Python for data preparation and SQL for powerful time-series analysis, along with crucial debugging skills that are invaluable in real-world data scenarios.

Results Table

The analysis produced a detailed results table, as shown in the provided screenshot. Which typically includes columns like sales_year_month, total_revenue, and total_volume. This table serves as the primary deliverable, presenting the aggregated monthly sales data. Overall, this task provided a practical foundation in using Python for data preparation and SQL for powerful time-series analysis, along with crucial debugging skills that are invaluable in real-world data scenarios.

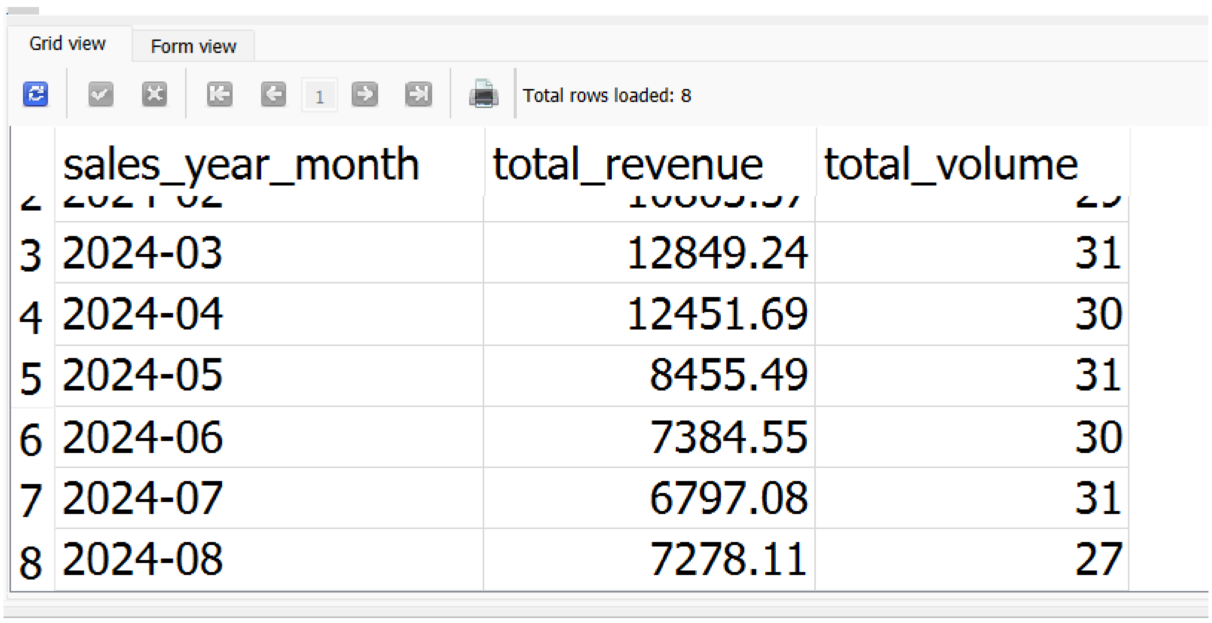


```sql
SELECT
    STRFTIME('%Y-%m', order_date) AS sales_year_month,
    SUM(amount) AS total_revenue,
    COUNT(DISTINCT order_id) AS total_volume
FROM
    online_sales
GROUP BY
    sales_year_month
ORDER BY
    sales_year_month;

```

Total rows loaded: 8

	sales_year_month	total_revenue	total_volume
1	2024-01	14548.32	31
2	2024-02	10803.37	29
3	2024-03	12849.24	31
4	2024-04	12451.69	30
5	2024-05	8455.49	31
6	2024-06	7384.55	30
7	2024-07	6797.08	31



Total rows loaded: 8

	sales_year_month	total_revenue	total_volume
2	2024-02	10803.37	29
3	2024-03	12849.24	31
4	2024-04	12451.69	30
5	2024-05	8455.49	31
6	2024-06	7384.55	30
7	2024-07	6797.08	31
8	2024-08	7278.11	27