Interview Questions Related To Above Task:

Q1.What is the difference between WHERE and HAVING in SQL?

Ans:-The WHERE clause filters rows before any groupings or aggregations are performed. The HAVING clause filters groups after the GROUP BY and aggregate functions have been applied. You use WHERE for standard row-level filtering and HAVING to filter on the result of an aggregate function (e.g., HAVING SUM(Sales) > 5000).

Q2.How does GROUP BY work? Can we use it without aggregate functions?
Ans:-GROUP BY works by collecting all rows that have the same value in a specified column (or columns) into a single summary row. Its primary purpose is to be used with aggregate functions (SUM, COUNT, AVG, etc.) to perform a calculation on each group. If you use GROUP BY without an aggregate function, it behaves similarly to the DISTINCT keyword, showing only one unique row for each group.

Q3.Explain different types of JOINs. When would you use a LEFT JOIN?

- INNER JOIN: Returns only the rows that have matching values in both tables.
- LEFT JOIN: Returns all rows from the left table, and the matched rows from the right table. If there is no match, the columns from the right table will be NULL.
- RIGHT JOIN: Returns all rows from the right table, and the matched rows from the left table.
- FULL OUTER JOIN: Returns all rows when there is a match in either the left or the right table. You would use a
  LEFT JOIN when you want to retrieve all records from one table regardless of whether they have a match in the other. For example, to get a list of all customers and their corresponding orders, you would
  LEFT JOIN the Customers table with the Orders table to ensure that customers who haven't placed any orders are still included in the result.

Q4.What is normalization? Why is it important in databases?
Ans:-Normalization is the process of organizing the columns and tables in a relational database to minimize data redundancy and improve data integrity. It involves dividing larger tables into smaller, well-structured tables and defining relationships between them. It's important because it:

- Reduces Redundancy: Prevents the same data from being stored in multiple places.

- Improves Data Integrity: Ensures that changes to data are consistent across the database.
- Makes the Database More Efficient: Smaller tables can be queried faster.

Q5.Write a query to find the second-highest salary from an employee table.

- **Get unique sales**

```
SELECT DISTINCT Weekly_Sales
FROM walmart_sales
ORDER BY Weekly_Sales DESC
LIMIT 2;
```

- **Wrap it with subquery**

```
SELECT MAX(Weekly_Sales) AS Second_Highest
FROM walmart_sales
WHERE Weekly_Sales < (
  SELECT MAX(Weekly_Sales) FROM walmart_sales
);
```