

The screenshot shows the MySQL Workbench interface. On the left, the 'Schemas' pane lists the 'adventureworks' database and its tables. The 'Query Editor' in the center contains a SQL query. The 'Results' pane at the bottom displays the output of the query as a table with 5 columns: Month, SalesCount, TotalSales, running_total_amt, and running_total_cmt. The data is filtered for the year 2006 and grouped by month.

Query:

```

SELECT
  Month,
  SalesCount,
  ROUND(TotalSales) AS TotalSales,
  ROUND(SUM(TotalSales) OVER (ORDER BY Month)) AS running_total_amt
FROM
  (SELECT
    DATE_FORMAT(OrderDate, '%Y-%m') AS Month,
    COUNT(*) AS SalesCount,
    SUM(TotalDue) AS TotalSales
  FROM
    adventureworks.salesorderheader
  WHERE
    YEAR(OrderDate) = 2006
  GROUP BY
    Month)

```

Results:

| Month | SalesCount | TotalSales | running_total_amt | running_total_cmt |
|-------|------------|------------|-------------------|-------------------|
| 01 | 228 | 1462449 | 1462449 | 228 |
| 02 | 250 | 2749105 | 4211554 | 478 |
| 03 | 263 | 2350568 | 6562122 | 741 |
| 04 | 244 | 1727690 | 8289811 | 985 |
| 05 | 299 | 3299799 | 11589610 | 1284 |
| 06 | 282 | 1920507 | 13510117 | 1566 |
| 07 | 325 | 3253419 | 16763536 | 1891 |
| 08 | 420 | 4663508 | 21427044 | 2311 |
| 09 | 309 | 3638980 | 25066024 | 2620 |
| 10 | 302 | 2488759 | 27554783 | 2922 |
| 11 | 326 | 3809633 | 31364416 | 3248 |
| 12 | 444 | 3099432 | 34463848 | 3692 |

```

SELECT
  Month,
  SalesCount,
  ROUND(TotalSales) AS TotalSales,
  ROUND(SUM(TotalSales) OVER (ORDER BY Month)) AS running_total_amt
FROM
  (SELECT
    DATE_FORMAT(OrderDate, '%Y-%m') AS Month,
    COUNT(*) AS SalesCount,
    SUM(TotalDue) AS TotalSales
  FROM
    adventureworks.salesorderheader
  WHERE
    YEAR(OrderDate) = 2006
  GROUP BY
    Month)

```

ORDER BY

Month) AS sales_data;

Explain:

To get the query that calculates cumulative sum of **SalesCount** and **TotalSales** ordered by **Month**:

1. Start with the inner query that retrieves the data needed for the report.
 - This query uses the **DATE_FORMAT** function to format the **OrderDate** into the desired format (MM).
 - It then uses the **COUNT** function to count the number of orders for each month, and **SUM** function to calculate the total sales for each month.
 - The data is grouped by **Month**, and ordered by **Month** to ensure the cumulative sum is calculated in the correct order.
2. Wrap the inner query with the outer query to calculate the cumulative sum of **SalesCount** and **TotalSales** using the **SUM** function with the **OVER** clause.
 - The cumulative sum is calculated for each row in the result set, ordered by **Month**.
 - The result set includes columns for **Month**, **SalesCount**, **TotalSales**, cumulative sum of **TotalSales**, and cumulative sum of **SalesCount**.
3. Round the values in the **TotalSales** and cumulative sum columns using the **ROUND** function.