Customer360 Project Report

Objective

The objective of this project was to create a Customer360 view for an online retailer by integrating data from multiple tables. This comprehensive view aimed to provide detailed insights into customer conversions, order history, and cumulative revenue, which would ultimately help in better understanding customer behaviour and making informed business decisions.

Method

To achieve this, we utilized several tables from the mban_db database: fact_tables.orders, fact_tables.conversions, dimensions.date_dimension, dimensions.product_dimension, and dimensions.customer_dimension. These tables were essential in constructing a unified and detailed view of each customer's activity. The first step in this project was to create a new schema named customer360. This schema was designed to store the final Customer360 view and any intermediate views required for data integration. Next, we created a CTE named ‘CustomerConversionData’ to capture static customer conversion details. This included conversion type, date, week, and channel. The use of SQL functions like ROW_NUMBER() and LEAD() allowed us to count conversions and find the next conversion week; this was important in order to track conversions for each customer. Another CTE named ‘FirstOrderPlaced’ was created to get the first order details relating to each conversion. The conversion events were linked to the first purchasing behaviour of the customer by aggregating the first order data, including the date, week, total paid, product details, and order number using the CTE.

Next, another CTE called ‘OrderHistory’ was created to aggregate weekly order data; this was used to capture metrics such as the number of orders placed, total before discounts, total discounts, and total paid in each week. Overall, this allowed for a comprehensive understanding of the costumes purchasing behaviour over time. A CTE named ‘AllWeeks’ was created in order to make sure all possible weeks within the conversion period for each customer was covered. A cross join between the date dimension and customer dimension tables was done and used by the CTE to generate all possible weeks–this was done so that weeks that did not have orders were accounted for in the analysis. A final CTE named ‘customer360_cte’ combined all previous CTEs to make a row from the conversion week to the next. Window functions were used to ensure efficient calculations for cumulative revenue for both the conversion period and lifetime revenue. This allowed for comprehensive data for detailed customer behaviour week by week. The data from the ‘customer360_cte’ was then ordered by customer ID, conversion number, and week counter to create the final Customer360 view. This final selection ensured that the data was presented in a logical and useful order, facilitating easy analysis and interpretation. Finally, the table was saved into the customer360 schema in our personal database.

Challenges Faced

We encountered difficulties in creating the customer360 schema in our personal database; this issue required adding ‘SELECT * INTO customer360.customer360_table FROM customer360_cte ORDER BY customer_id, conversion_number, week_counter’ to the end of our code to ensure that the table could be added to our personal database successfully. We also struggled a bit while ensuring accurate joins between the fact and dimensions tables. This required careful attention to data integrity and consistency to avoid errors and ensure the correctness of the integrated data. Handling cases where the next conversion week was not available also presented difficulties. We addressed this issue using conditional functions like ISNULL to maintain proper data continuity and avoid gaps in the timeline. Another significant challenge was efficiently aggregating order data while maintaining performance. The use of SQL window functions and optimized queries helped to address performance issues, ensuring that the final Customer360 view was both comprehensive and efficient–these optimizations were crucial for handling multiple datasets and complex calculations without compromising on speed.

Conclusion

The Customer360 view effectively integrates data to provide a comprehensive overview of customer behaviour. This project demonstrated the use of SQL for complex data integration, resulting in a valuable tool for understanding customer behaviour and supporting business decisions.