**A REPORT GENERATED USING SQL QUERIES SHOWCASING TRENDS OR PATTERNS**

This report presents a detailed analysis of employee sales performance using advanced SQL features. The focus of this analysis is to apply Subqueries, Common Table Expressions (CTEs), and Window Functions to a relational dataset in order to uncover performance trends, departmental differences, and individual contribution patterns. The task aligns with real-world business requirements where organizations depend on data to monitor, assess, and guide employee performance.

Two tables were created for this task: Employees and Sales. The Employees table contains employee specific information such as ID, name, and department, while the Sales table stores individual sales transactions linked to each employee. Together, they simulate a small organization’s internal sales structure, enabling the use of advanced SQL queries to derive analytical insights.

The first query used in this task was a Subquery. Subqueries are nested SQL statements used within other queries to create dynamic filtering or comparison criteria. Here, a subquery calculated the average sales amount and compared each employee’s total sales to that value. The outer query filtered those employees whose total sales exceeded the average. This approach helps quickly identify high performers in any dataset. Bhavya from the Sales department, based on this analysis, emerged as a clear top performer with a total significantly greater than the company average.

Next, a Common Table Expression (CTE) was implemented. CTEs simplify complex queries by creating temporary result sets that can be referred to within the main query. They’re excellent for breaking down logic and improving code readability. In this case, the CTE calculated total sales for each employee and was then filtered to focus on the Sales department only. This narrowed the analysis to compare internal performance within one team. Results showed that while two employees were in the Sales department, only one of them, Bhavya, was actively contributing a significant amount, indicating a gap that may need addressing.

The third and most insightful approach was the use of a Window Function specifically, RANK(). Window Functions are advanced SQL tools that allow operations across a set of table rows that are somehow related to the current row. They’re commonly used for ranking, running totals, and moving averages. In this case, RANK() was used to assign a rank to each sale based on its amount, from highest to lowest. This method highlighted not just total sales but individual transaction values. Again, Bhavya dominated the top of the rankings, while Diya and Chirag had lower ranked results, revealing both limited sales and possibly fewer opportunities.

These three techniques Subqueries, CTEs, and Window Functions provided multiple perspectives of the same dataset. From totals, to department specific summaries, to transaction level analysis, each method contributed unique value to the overall assessment. They allowed the creation of a multidimensional report that goes far beyond what basic SELECT statements can offer. This demonstrates why SQL is such a powerful language in both business intelligence and data science contexts.

In terms of real world application, these techniques are crucial in roles such as business analysts, database developers, HR data consultants, and operations managers. Subqueries can help segment high performers. CTEs are ideal for simplifying multistep logic in dashboards or reports. Window Functions are invaluable in generating rankings, comparative reports, or even trend graphs when paired with reporting tools. These tools empower data-driven decisions across departments.

This report identified several key trends. First, Bhavya’s consistent appearance across all top metrics makes her the organization’s top sales performer. Second, the Sales department, while active, suffers from internal imbalance suggesting the need for internal support, mentoring, or workload redistribution. Third, departments like HR show little or no participation in sales, which is understandable, but if this analysis were adapted to HR-specific KPIs, similar methods could reveal participation gaps there too.

From a learning perspective, this task has deepened my understanding of how to apply SQL analytically, not just technically. Writing a JOIN query is easy but asking the right question, choosing the right SQL structure, and interpreting the result takes a deeper level of understanding. Through this report, I have practiced building queries that tell a story one that management could actually use.

In conclusion, this analysis using Subqueries, CTEs, and Window Functions was successful in generating a meaningful, trend-focused report. SQL was not just a language here it was a thinking tool. I now feel more confident in using these advanced concepts to solve practical business problems and produce insights that matter.