**Task 3 – Database Migration Report**

As part of my SQL internship under CODTECH, Task 3 focused on the process of migrating data from a MySQL database to a PostgreSQL database. This task simulated a real-world scenario where organizations shift their systems or upgrade databases and need to ensure that the data remains accurate, complete, and properly structured throughout the transition. This task helped me understand the challenges of such migrations and gave me hands-on experience in handling data movement between two different database systems.

I started with an existing MySQL database named task2\_analysis which I had used in Task 2. It contained two related tables: Employees and Sales. The Employees table had basic information like EmployeeID, Name, and Department. The Sales table had SaleID, EmployeeID (as a foreign key), SaleAmount, and SaleDate. Both tables were fully populated with test data for realistic analysis. My goal was to replicate this schema and its data accurately in PostgreSQL.

First, I installed PostgreSQL and opened pgAdmin 4, which provides a user-friendly GUI. I created a new database named task3\_analysis. Then, I recreated both the Employees and Sales tables inside PostgreSQL by manually executing CREATE TABLE statements that matched the exact structure of the MySQL tables, including the primary key and foreign key constraints. This ensured that the structure was consistent across both systems.

After the structure was ready, I exported the data from MySQL into CSV format. For this, I used MySQL Workbench and right-clicked the Employees and Sales tables to export their data as CSV files. One key learning here was that the CSV format must match the destination database's expectations. Initially, I faced issues with delimiters, headers, and file encoding. The default export used semicolons, while PostgreSQL expected commas. I cleaned and corrected these CSV files in Notepad and Excel by ensuring commas were used as delimiters and headers matched the actual columns.

Next, I imported the cleaned CSV files into PostgreSQL. I attempted this first through pgAdmin’s Import tool, but ran into multiple errors due to permission issues and format mismatches. Eventually, I switched to the PostgreSQL terminal using the psql shell. Here, I used the \copy command instead of COPY because it works client-side and supports importing local files. This step was important because it showed me how important the right tool is when dealing with system-level operations. Using \copy, I successfully imported both CSV files into the task3\_analysis database.

To verify the migration, I ran SELECT \* queries on both tables inside pgAdmin and compared the results to the original MySQL dataset. All records matched exactly in both content and order. There were no missing or duplicate rows. This confirmed that data integrity was preserved during migration. I took screenshots of the output to document this step.

In terms of technical learnings, this task taught me several key things: how to maintain consistent schemas between two different database engines, how to export and import data using CSVs, how to troubleshoot format-related issues, and the difference between COPY and \copy in PostgreSQL. It also gave me exposure to working with permissions, character encoding, and handling foreign key constraints across platforms.

Real-world use cases of database migration include moving from local to cloud databases, switching between vendors due to licensing or performance, or upgrading older systems to modern ones. The concepts I learned here are directly applicable in those scenarios. I now feel more confident in handling migrations between MySQL and PostgreSQL and understand the attention to detail required to ensure data accuracy.

Overall, Task 3 was not just about copying data—it was about understanding data flow, structure compatibility, and reliability in the context of database management systems. It gave me a strong foundation in how professional database migrations are performed and helped me develop a systematic approach to troubleshooting issues that arise along the way.