**Data Warehouse Project**

**Introduction:**

In this project, we were asked to implement a relational data warehouse using an algorithm called as “HYBRID JOIN”. We were asked to make a prototype warehouse for Electronica Business Chain in Pakistan.

**Steps involved:**

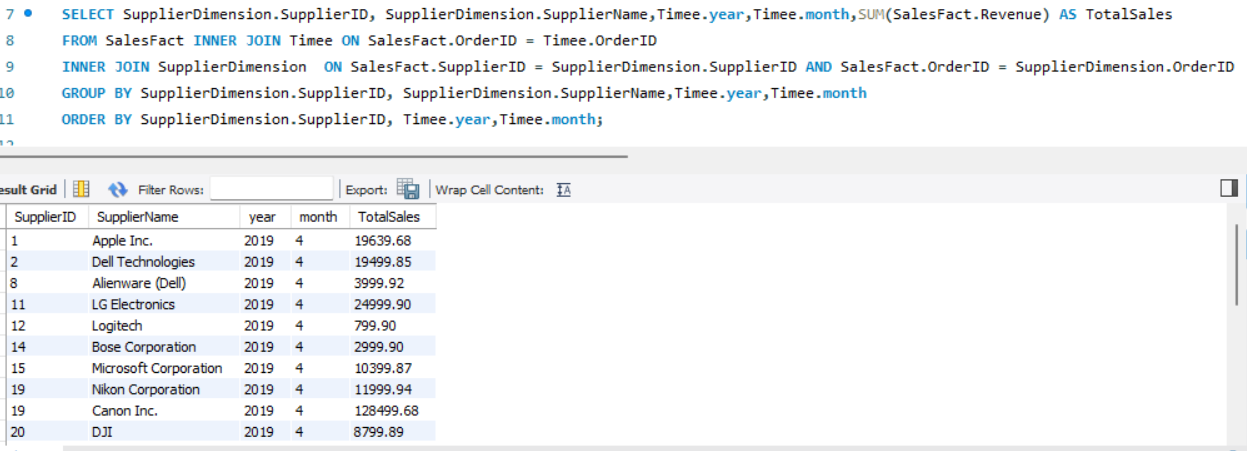
* We initiated by loading CSV files into MySQL Workbench and established a JDBC connection to both the 'Transaction' and 'Master' databases. Subsequently, we successfully retrieved transaction data as a stream.
* Initially, the first 1000 tuples were sent to a queue containing only product IDs. Simultaneously, a hash map was utilized, where the product IDs served as keys and the associated data was stored as values.
* During the 'processMasterData' function, while extracting master data, we performed a check to validate if the product ID from the transaction aligned with the product ID from the master data. Whenever a match occurred, the entire set of data linked to that particular product ID was directly dispatched to the data warehouse. This ensured that all the relevant data associated with the matched product ID was efficiently sent to the warehouse.

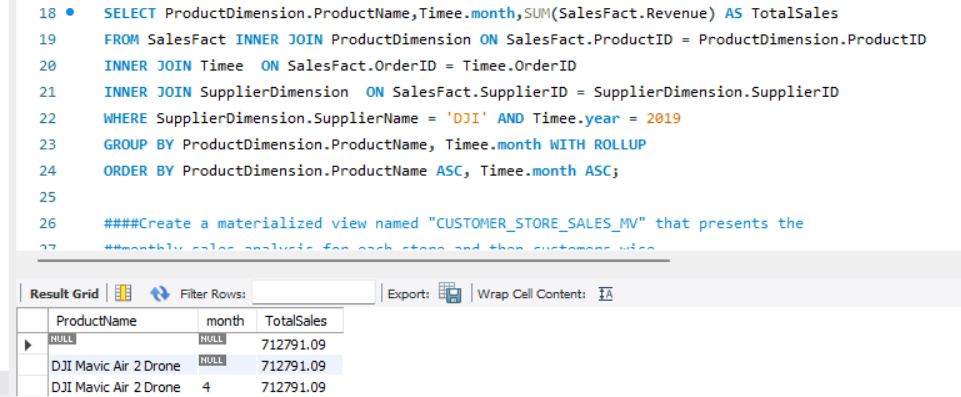
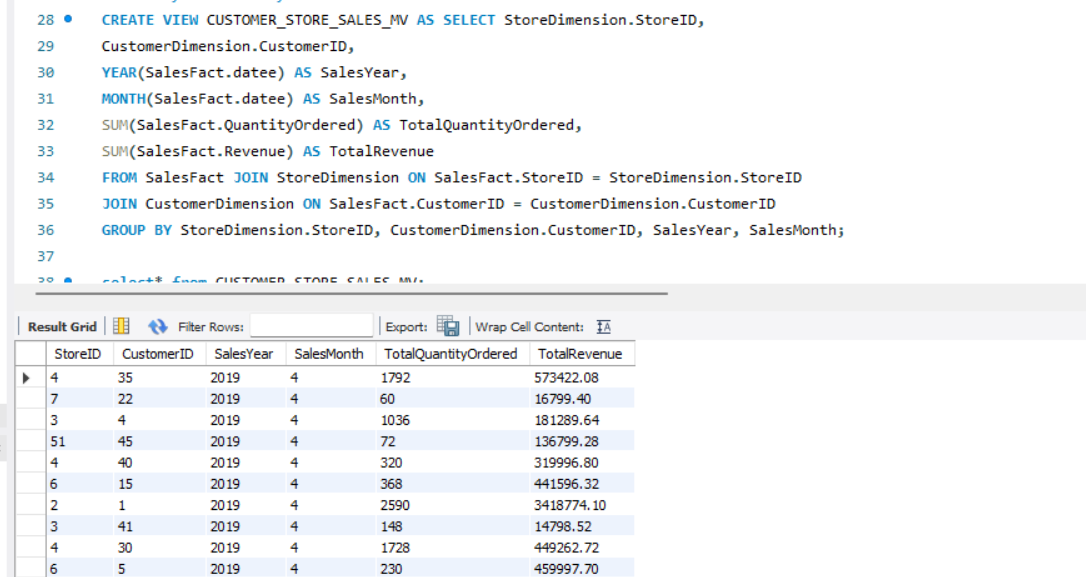
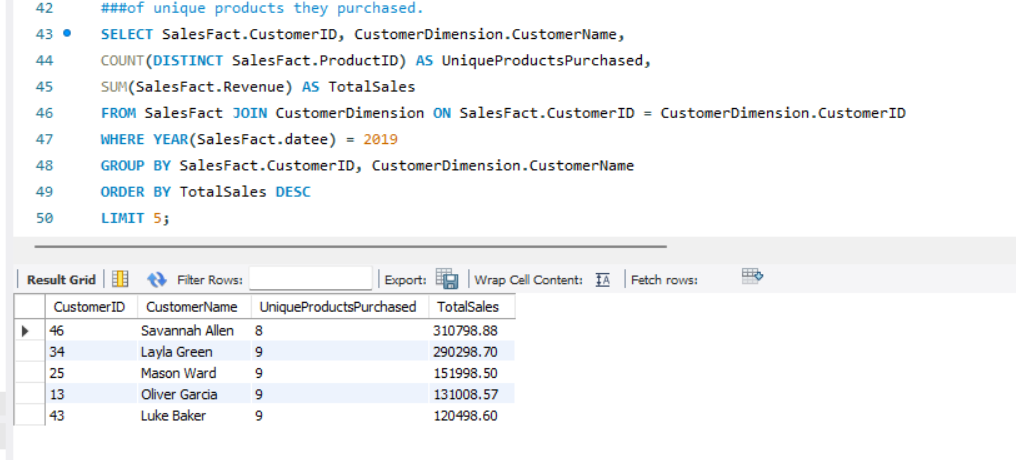
**Concept of Slice and Dice:**

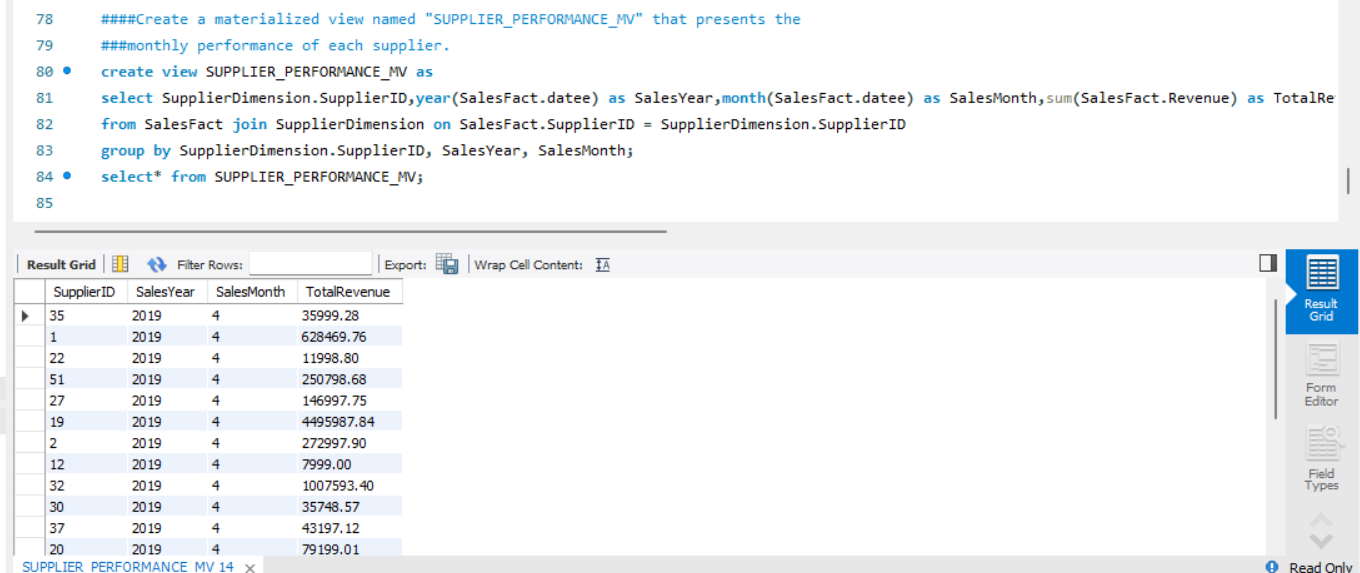
* Slicing involves the concept of focusing on a specific slice of data. For example, focusing on a specific region or a specific product.
* Dicing allows you to ask query across different dimensions. For Example
* Show the sales of “Electronics” across region of “North America”

**OLAP:**

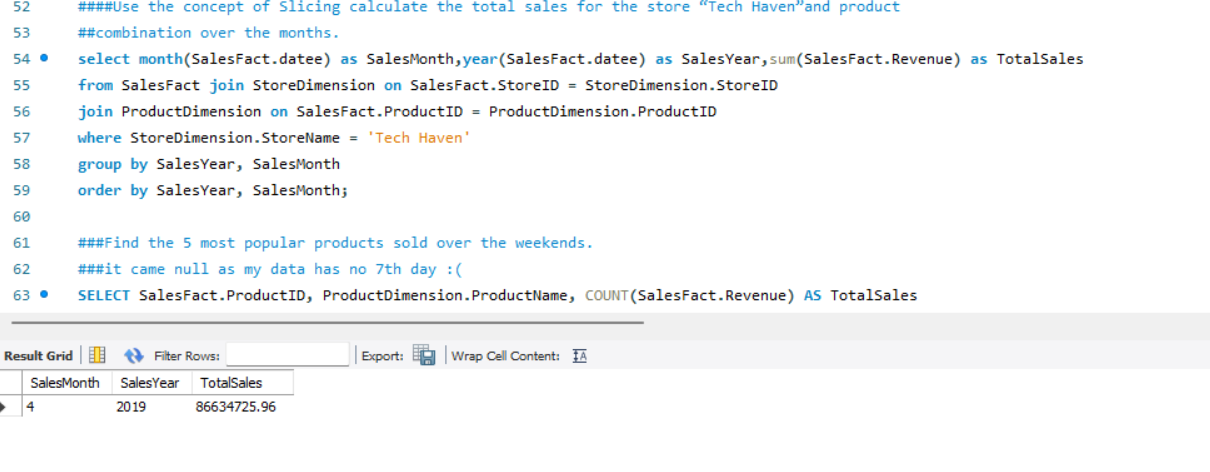
1. Present total sales of all products supplied by each supplier with respect to quarter and month using drill down concept.



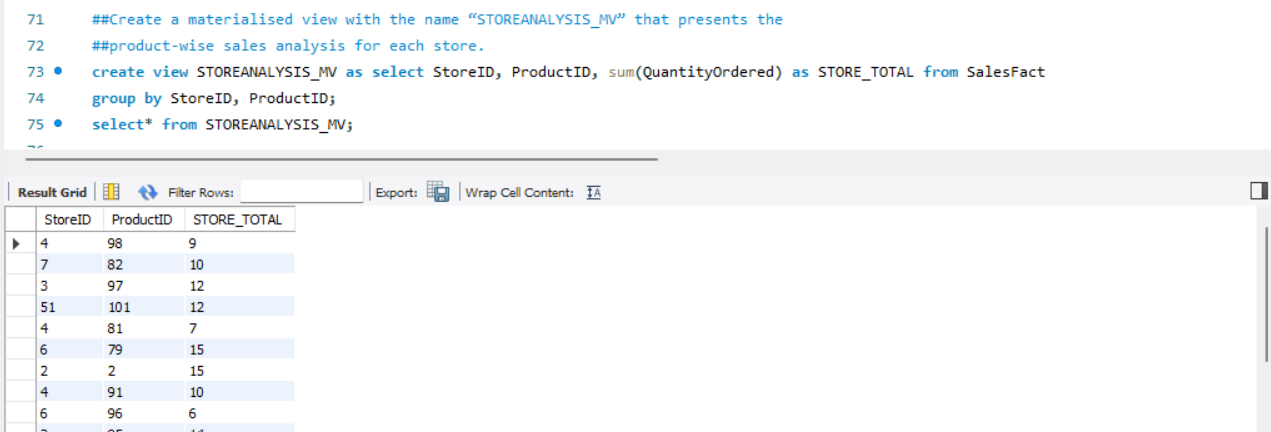
1. Find total sales of product with respect to month using feature of rollup on month and feature of dicing on supplier with name "DJI" and Year as "2019". You will use the grouping sets feature to achieve rollup. Your output should be sequentially ordered according to product and month.
2. Create a materialized view named "CUSTOMER\_STORE\_SALES\_MV" that presents the monthly sales analysis for each store and then customers wise.
3. Identify the top 5 customers with the highest total sales in 2019, considering the number of unique products they purchased.
4. Create a materialized view named "SUPPLIER\_PERFORMANCE\_MV" that presents the monthly performance of each supplier.



1. Use the concept of Slicing calculate the total sales for the store “Tech Haven”and product combination over the months.



1. Create a materialised view with the name “STOREANALYSIS\_MV” that presents the product-wise sales analysis for each store.



**What I learnt from the project:**

Even though I initially felt overwhelmed by the task of implementing the algorithm, I persevered and learned a great deal. This project highlighted the incredible ease that a data warehouse provides for executing OLAP queries. Handling the same query in a standard database could have been time-consuming in a real business scenario. Additionally, I've come to appreciate how a small modification can significantly enhance the efficiency of an algorithm. Initially, I struggled to grasp the concept of queues within the process, but now I see how pivotal they are in the entire algorithm, which operates at a much more advanced level.

**Short comings in hybrid join:**

**Resource Intensiveness:** Depending on the implementation, Hybrid Join might consume considerable computing resources, especially memory and processing power, which could affect the overall system performance.

**Large Data Sets:** When dealing with exceptionally large datasets, the Hybrid Join approach might encounter scalability challenges. Handling large volumes of data efficiently might become complex, impacting performance.

**Sensitivity to Memory Budgets:** While Hybrid Join shows better performance across various memory budgets, it might still have limitations concerning extremely constrained memory environments. In scenarios of severely limited memory, its advantages over other join algorithms might diminish.