Near-Real-Time Data Warehouse for METRO Shopping Store

1. Executive Summary

This report presents a prototype for a near-real-time Data Warehouse (DW) designed for METRO Shopping Store in Pakistan. METRO is a large supermarket chain, dealing with thousands of customer transactions every day. To gain actionable insights and optimize sales strategies, it is crucial to analyze customer shopping behaviour promptly. By leveraging data warehousing techniques, we aim to build a near-real-time ETL pipeline that processes transactional data, enriches it with master data, and loads it into the DW for business intelligence analysis.

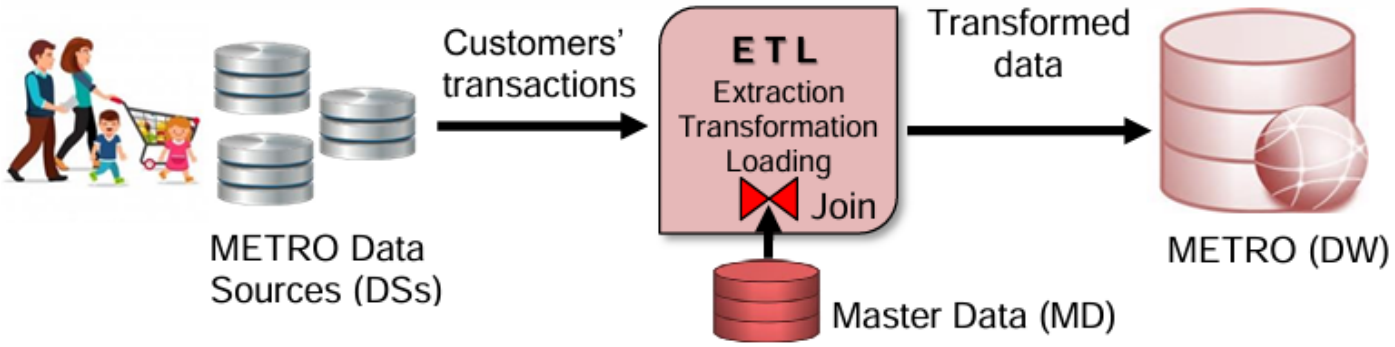


Figure 1: An overview of METRO DW

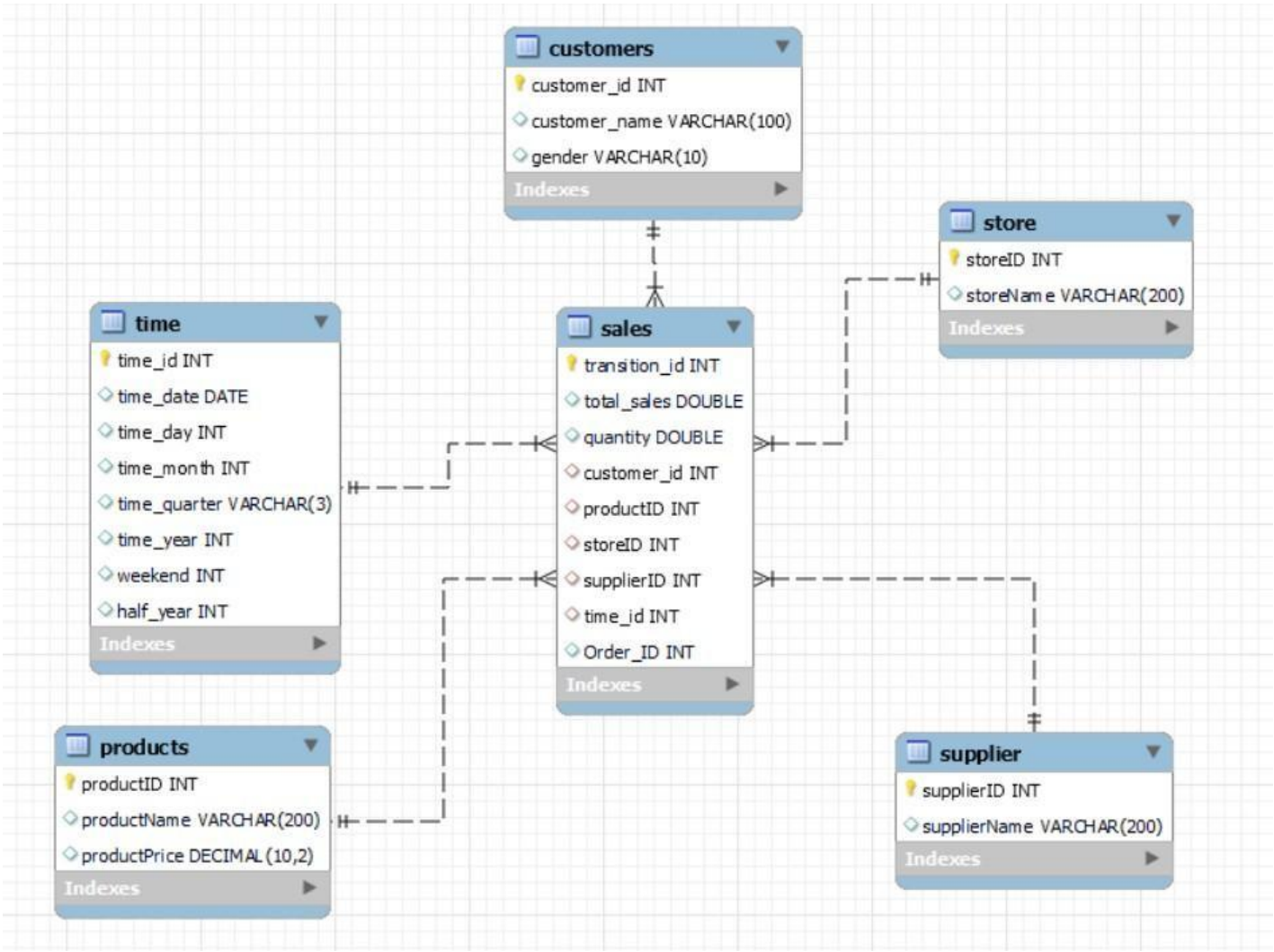
Objectives:

- Design and implement a star schema for DW to support multidimensional analysis.
- Apply Mesh Join algorithm for stream-relation joining of transactional data with master data.
- Build a near-real-time ETL pipeline to extract, transform, and load data into the DW.
- Execute OLAP queries to derive actionable insights for business strategies.

---

2. Schema for the Data Warehouse

The Data Warehouse is designed using the **Star Schema**, which offers simplicity and efficiency for multidimensional analysis. The star schema consists of a **fact table** at the centre, surrounded by various **dimension tables**. The schema is optimized for slicing, dicing, and analysing data across multiple business aspects.



Dimension Tables:

- **Customers Attributes**:
  customer_id (PK), customer_name, gender
- **Products Attributes**:
  product_id (PK), product_name, product_price
- **Store Attributes**:
  store_id (PK), store_name
- **Supplier Attributes**:
  supplier_id (PK), supplier_name
- **Time Attributes**:
  time_id (PK), time_date, time_day, time_month, time_quarter, time_year, weekend, half_year

Fact Table:

- **Sales Attributes**:

  transition_id (PK), total_sales, quantity, customer_id (FK), product_id (FK), store_id (FK), supplier_id (FK), time_id (FK), order_id

3. Mesh Join

The Mesh Join algorithm is used to process and enrich transactional data by joining it with master data (i.e., customer and product data). It is designed for stream-relation joins and helps efficiently integrate large, disk-based datasets with streaming transactional data.

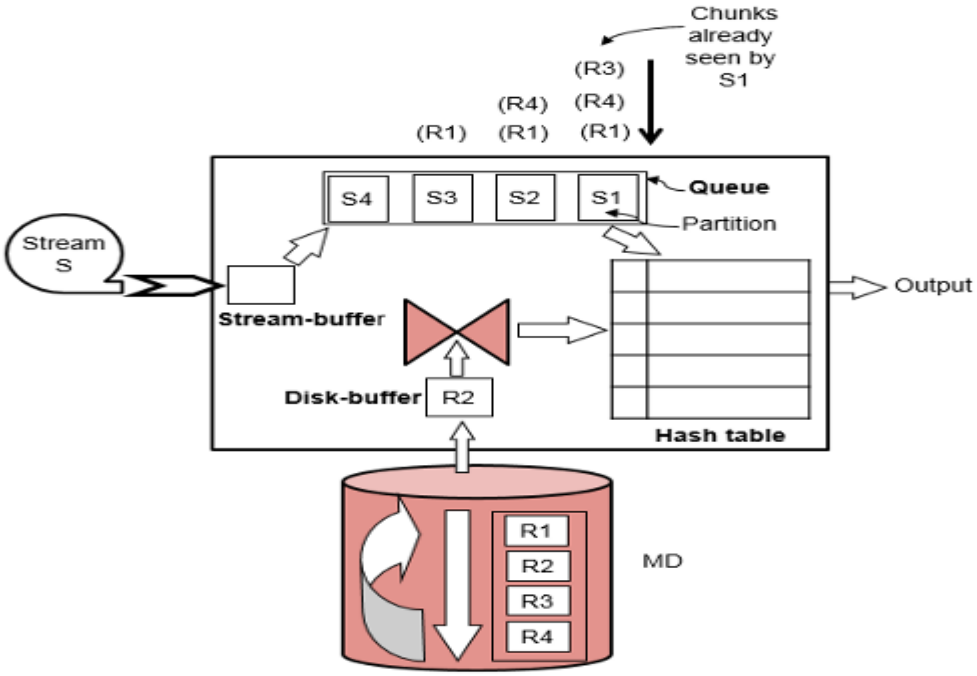


Figure : Working of MESHJOIN when R_2 is in memory but not yet processed

Mesh Join Components:

- **Stream Buffer**: Holds the incoming transactional data in chunks and stores it in a queue for processing.
- **Partitions for Master Data**: The customer and product master data are partitioned into manageable chunks, which are loaded into memory.
- **Hash Table**: Stores portions of transactional data temporarily for efficient probing during the join operation.
- **Queue**: Ensures that each chunk of transactional data is processed with all available master data partitions before being ejected.

Process:

1. **Reading Transactional Data**: Transaction data is read in chunks from a file and added to the stream buffer.
2. **Loading Master Data**: Master data partitions (customer and product) are loaded into the disk buffer.
3. **Joining**: Each chunk of transactional data is joined with the appropriate partition from the master data using the hash table.
4. **Data Enrichment**: After successful joins, the relevant attributes (e.g., total sales) are added to the transactional data.
5. **Loading to DW**: The enriched data is loaded into the appropriate dimension and fact tables in the Data Warehouse.
6. **Eviction**: Once all partitions have been joined for a chunk, the record is removed from memory.

4. Mesh Join Shortcomings

While the Mesh Join algorithm is efficient for stream-relation joins, it has a few limitations:

1. **Memory Consumption**: The size of the hash table and the number of in-memory buffers may limit the system's ability to handle large volumes of data concurrently. This may require optimization strategies.
2. **Hashing Overhead**: The need to probe through all master data for each transactional record can be slow, especially if the master data is large.
3. **Disk I/O Bottleneck**: Continuously loading master data partitions from disk may become a performance bottleneck, especially when dealing with high-frequency transactional data.

---

5. Learnings from the Project

Through the execution of this project, several key concepts were learned:

- **Data Warehouse Design**: I gained insights into the design and implementation of a star schema for multidimensional decision support, which is a cornerstone of business intelligence.
- **Real-Time ETL Pipelines**: I developed hands-on experience in constructing near-real-time ETL pipelines using Java, which allowed for the timely transformation and loading of data into the DW.
- **Algorithm Optimization**: The trade-offs in stream-relation join algorithms were explored. The limitations of memory, hashing, and disk I/O were evaluated, and potential optimization techniques were considered.
- **OLAP Analysis**: I learned how OLAP queries such as slicing, dicing, drill-down, and roll-up could be used to gain valuable business insights from the DW.

---

6. Conclusion

This project demonstrates the implementation of a near-real-time Data Warehouse for METRO Shopping Store. The use of a **star schema** design ensures efficient querying and analysis of sales data across various dimensions. The **Mesh Join algorithm** was successfully applied to integrate transactional data with master data, enriching it for effective business intelligence analysis. By executing various **OLAP queries**, actionable insights were gained, enabling data-driven decision-making for METRO.

The project highlights the importance of real-time data integration and analysis, especially in a fast-paced retail environment like METRO. The experience has not only deepened my understanding of data warehousing but also provided practical insights into stream-relation joins and the application of business intelligence in optimizing sales strategies.

---

7. References

- Polyzotis, N. (2008). MESHJOIN: A Stream-Relation Join Algorithm. *International Conference on Data Engineering*.
- Kimball, R., & Ross, M. (2011). *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling* (3rd ed.). Wiley.
- Inmon, W. H. (2005). *Building the Data Warehouse* (4th ed.). Wiley.