Data Warehouse Semester Project

### Submitted to: Doc. Asif Naeem

### Submitted By: Maaz Asad (i180474)

### Section : A



# Project Overview

The goal of this project was to create a data warehouse of a sample retail operation database. We were provided with Transactional data and Master data. The main aim was to emulate the ETL process using the MESHJOIN algorithm and load data into the warehouse. The Data warehouse was in the form of a star schema where data was loaded after being manipulated by the MESHJOIN

# What is MeshJoin?

MeshJoin is an algorithm with the main objective of providing a near real time data warehouse by reducing the time taken for Disk I/O.

It consists of 4 major components. They are mentioned below:

1. Stream Buffer (Transactional data)
2. Disk Buffer (Master Data)
3. Queue
4. Hash Table

The Transactional data is temporarily stored in the Stream Buffer. When the buffer reaches a certain record size, it is pushed into a queue. As it is pushed into the queue, an entry is made in the Hash Table with relevant keys. While this process is working, a partition of disk is loaded into the Disk buffer. When a partition from disk is loaded into the Disk buffer, it is joined with all the stream partitions in the queue and all the entries in the Hash Tables are updated. The max size of the queue is equal to the number of partitions in the disk. When a partition in the queue is joined with all the Disk partitions it is de-queued and the joined records related to the stream partition from the hash table are inserted into the data warehouse and cleared from the hashtable to make space for more entries.

This whole process works in tandem with both the streams (transactional and master data) and queue and hashtable reading and joining and writing into the data warehouse.

# The Transactional Data and Master Data

We were provided with a sql file containing the transactional and master data.

The structure of these tables was as such:

**TRANSACTION:**

* **TRANSACTION\_ID** VarChar(8)
* **PRODUCT\_ID** VarChar(6)
* **CUSTOMER\_ID** VarChar(4)
* **CUSTOMER\_NAME** VarChar(30)
* **STORE\_ID** VarChar(3)
* **STORE\_NAME** VarChar(20)
* **T\_DATE** DATE
* **QUANTITY** NUMBER(3,0)

**MASTER:**

* **PRODUCT\_ID** VarChar(6)
* **PRODUCT\_NAME** VarChar(30)
* **SUPPLIER\_ID** VarChar(5)
* **SUPPLIER\_NAME** VarChar(30)
* **PRICE** NUMBER(5,2) DEFAULT 0.0

The Stream buffer and disk buffer designed as such to handle these records effectively so that the data is loaded correctly for the correct joins in the mesh join algorithm

Observing these tables we were able to create the Star schema for the data warehouse

# The Star Schema

By looking at the transaction and master data we created the following structures for the facts and dimension tables

## Dimension Tables:

**PRODUCTS**:

* PRODUCT\_ID VARCHAR(6)
* PRODUCT\_NAME VARCHAR(50)

**CUSTOMERS:**

* CUSTOMER\_ID VARCHAR(6)
* CUSTOMER\_NAME VARCHAR(50)

**SUPPLIERS:**

* SUPPLIER\_ID VARCHAR(5)
* SUPPLIER\_NAME VARCHAR(30)

**STORES:**

* STORE\_ID VARCHAR(4)
* STORE\_NAME VARCHAR(40)

**DATES**:

* DATE\_ID DATE
* DAY DECIMAL(2,0)
* MONTH DECIMAL(2,0)
* QUATER DECIMAL(1,0)
* YEAR DECIMAL(4,0)
* WEEK\_OF\_YEAR DECIMAL(2,0)
* WEEK\_DAY VARCHAR(9)

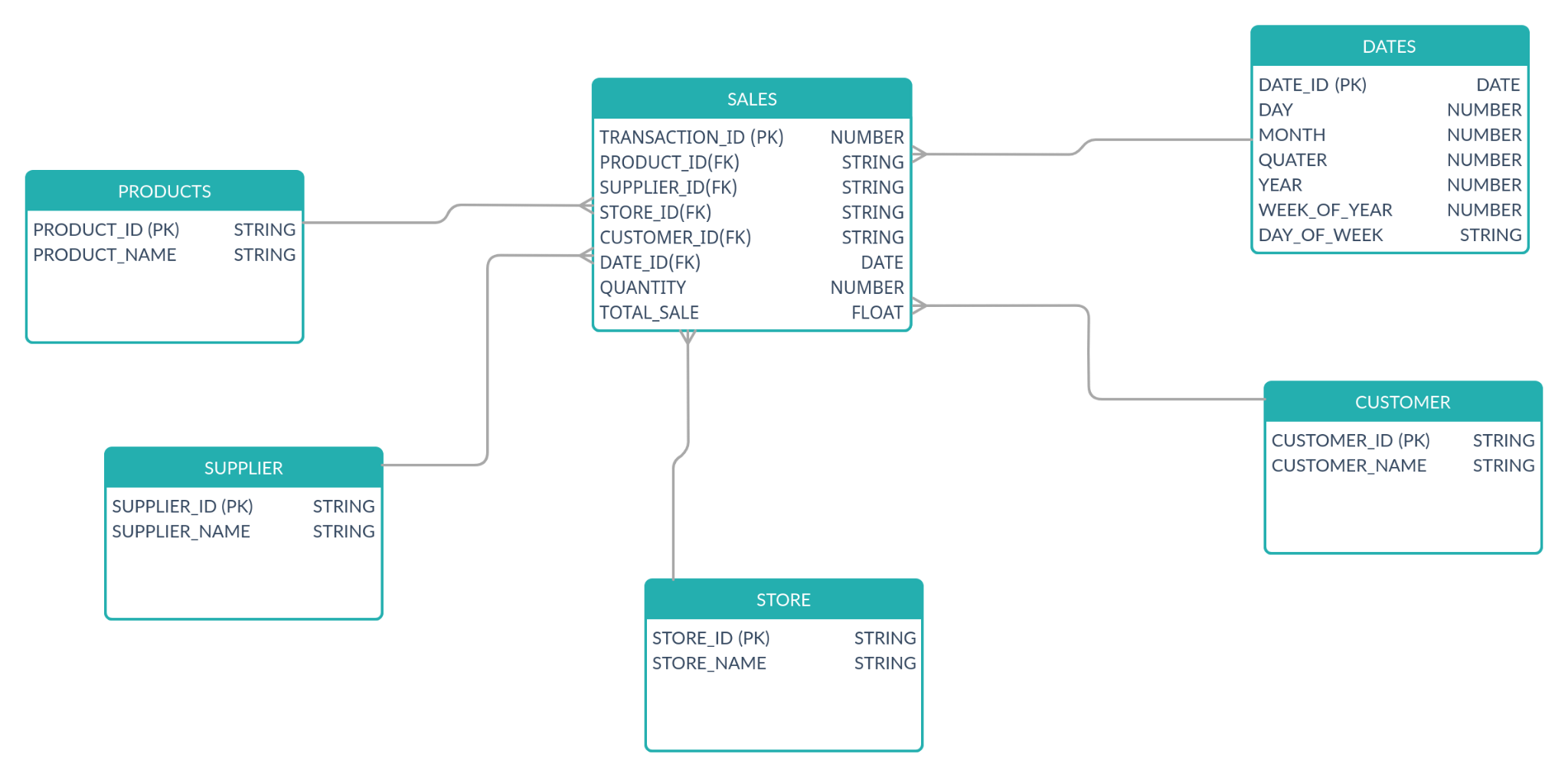
## Fact Table

**SALES:**

* TRANSACTION\_ID DECIMAL(8,0) PRIMARY KEY NOT NULL,
* PRODUCT\_ID VARCHAR(6) NOT NULL,
* SUPPLIER\_ID VARCHAR(5) NOT NULL,
* CUSTOMER\_ID VARCHAR(4) NOT NULL,
* STORE\_ID VARCHAR(4) NOT NULL,
* DATE\_ID DATE NOT NULL,
* QUANTITY DECIMAL(3,0) NOT NULL,
* TOTAL\_SALE DECIMAL(6,2) NOT NULL,

When data in the hash Table has been joined completely with all disk partitions to create the sales record. The resulting record is added into the sales table and other tables are updated respectively such that there are no duplicates. This ensures that the data in the data warehouse is consistent and properly loaded.

## Data Warehouse Star Schema Diagram



# Shortcomings of MeshJoin

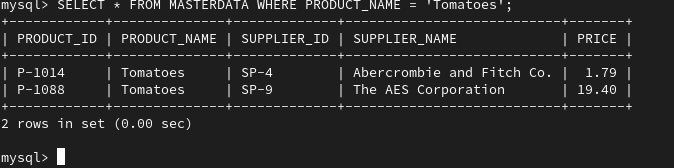
One major shortcoming of the MeshJoin algorithm is that we have to maintain a queue alongside a Hash Table of partitions of transactional data. This consumes memory so therefore would require a large amount of memory for successful processing.

We still cannot ignore the cost of Disk I/O that are required for reading the Master Data. As the dequeue and enqueue are dependent on the readiness of disk buffer, it can still lead to a hog in stream buffer causing for input stream to be slowed down and the.

# 

# Anomaly in the data

There are 2 products with the same name but have different prices and suppliers. This can lead to inconsistencies as the master data is being read in batches of 10. The 2nd batch will have the first tomatoes record while the 8th batch will have the other tomatoes record with a higher price.



# Learning Outcomes

The process of identifying the necessary dimensions and facts for a data warehouse given a database.

The implementation of MeshJoin enabled us to simulate a near real-time data warehouse where there was a constant incoming stream which was being divided into chunks. These chunks were processed efficiently using Hash Tables and Hash keys to ensure the data is loaded quickly into the warehouse and more incoming stream can be handled.

Importance of efficient storage of master data in a sequential format to ensure faster read times which adds onto the overall efficiency of the MeshJoin Algorithm.