# Project Report

By Aitazaz Kamran  
 (22i-2021)(DS-sec-B)

## Project Overview

This project focuses on building a data warehouse (DW) for a sales scenario and implementing the MESHJOIN algorithm in Java for loading transactional data into the DW. The project includes identifying appropriate dimension and fact tables, creating a star schema, analyzing data using slicing, dicing, drill-down techniques, and implementing materialized views.

## Schema for Data Warehouse (DW)

The schema is designed using a star schema approach. It consists of one fact table and several dimension tables. The fact table 'Sales' contains quantitative data, while dimension tables provide descriptive information. Below are the tables and their attributes:

Fact Table: Sales

- Attributes: Order\_ID (PK), Product\_ID (FK), Customer\_ID (FK), Supplier\_ID (FK), Time\_ID (FK), Store\_ID (FK), Quantity, Total\_Sale

Dimension Tables:

- Product: Product\_ID (PK), Product\_Name, Price  
- Customer: Customer\_ID (PK), Customer\_Name, Gender  
- Supplier: Supplier\_ID (PK), Supplier\_Name  
- Store: Store\_ID (PK), Store\_Name  
- Time\_Dimension: Time\_ID (PK), Transaction\_Date, Weekend, Time, Half\_Of\_Year, Month, Quarter, Year

## MESHJOIN Algorithm

The MESHJOIN (Mesh Join) algorithm has been introduced by Polyzotis in 2008 with the objective of implementing the Stream-Relation join operation in the transformation phase of ETL.

The main components of MESHJOIN are: The disk-buffer, which will be an array and used to load the incoming stream tuples from the stream relation. The in-memory buffer, which will be a hash table and used to load the tuples from the relation that will be joined with the incoming stream.

The Mesh, which is a graph structure that maps the incoming stream tuples to the tuples in the relation buffer that they may join with. The Join engine, which is responsible for performing the actual join operation between the incoming stream tuples and the relation tuples that are mapped to them by the Mesh.

**The MESHJOIN algorithm works as follows:**

The incoming stream tuples are loaded into the disk-buffer.

The relation tuples that will be joined with the stream are loaded into the in-memory buffer.

The Mesh is constructed by hashing the relation tuples in the in-memory buffer to their corresponding Mesh nodes.

The incoming stream tuples are then processed and mapped to the Mesh nodes.

The Join engine uses the Mesh to identify the relation tuples that can be joined with the incoming stream tuples.

The Join engine performs the join operation between the incoming stream tuples and the identified relation tuples.

The joined tuples are then sent to the output.

The MESHJOIN algorithm provides an efficient solution for performing Stream-Relation join operation during ETL transformations. By using a combination of disk-buffer, in-memory buffer, Mesh, and Join engine, it reduces the I/O operations and improves the performance of the join operation..

## Shortcomings of MESHJOIN

1. MESHJOIN is highly dependent on the partitioning strategy. Improper partitioning can lead to inefficient memory usage.  
2. Not resource friendly if the number of tables in the master data are increased.  
3. The algorithm requires multiple passes over the larger dataset, which can be time-consuming for extremely large datasets.

## What Did I Learn?

1. Designing a star schema for a data warehouse requires understanding the relationships between facts and dimensions.  
2. Implementing the MESHJOIN algorithm helped in understanding how to handle large and streaming datasets efficiently.  
3. Analyzing data using slicing, dicing, and drill-down techniques provided insights into multidimensional data analysis.