Name:	Shreeya Sunil Hudekar
Roll No:	13
Class/Sem:	TE/V
Experiment No.:	1
Title:	Data Warehouse Construction – Star schema and Snowflake schema
Date of Performance:	19/07/24
Date of Submission:	26/07/24
Marks:	
Sign of Faculty:	

Aim: To Build a Data Warehouse – Star Schema, Snowflake Schema and Fact Constellation Schema

Objective: A data warehouse is a large store of data collected from multiple sources within a business. The objective of a data warehouse system is to provide consolidated, flexible, meaningful data storage to the end user for reporting and analysis.

Theory:

In general, the warehouse design process consists of the following steps:

1. Choose a business process to model (e.g., orders, invoices, shipments, inventory, account administration, sales, or the general ledger). If the business process is organizational and involves multiple complex object collections, a data warehouse model should be followed. However, if the process is departmental and focuses on the analysis of one kind of business process, a data mart model should be chosen.
2. Choose the business process grain, which is the fundamental, atomic level of data to be represented in the fact table for this process (e.g., individual transactions, individual daily snapshots, and so on).
3. Choose the dimensions that will apply to each fact table record. Typical dimensions are time, item, customer, supplier, warehouse, transaction type, and status.
4. Choose the measures that will populate each fact table record. Typical measures are numeric additive quantities like dollars sold and units sold.

Steps to Draw Star, Snowflake, and Fact Constellation Schemas

1. Star Schema:

- Step 1: Identify the central fact table, which contains quantitative data (e.g., sales, revenue).
- Step 2: Determine the dimension tables related to the fact table, such as time, product, customer, etc.
- Step 3: Define the relationships between the fact table and each dimension table, usually a one-to-many relationship.
- Step 4: Draw the fact table at the center and connect it to each dimension table using lines, creating a star-like structure.

2. Snowflake Schema:

- Step 1: Start with the fact table as in the Star Schema.
- Step 2: Identify the dimension tables and further normalize them by breaking them into multiple related tables (e.g., split "Location" into "Country" and "City").
- Step 3: Establish relationships between these normalized dimension tables and the fact table.
- Step 4: Draw the fact table at the center, then connect it to the dimension tables, which in turn connect to their sub-tables, forming a snowflake-like structure.

3. Fact Constellation Schema:

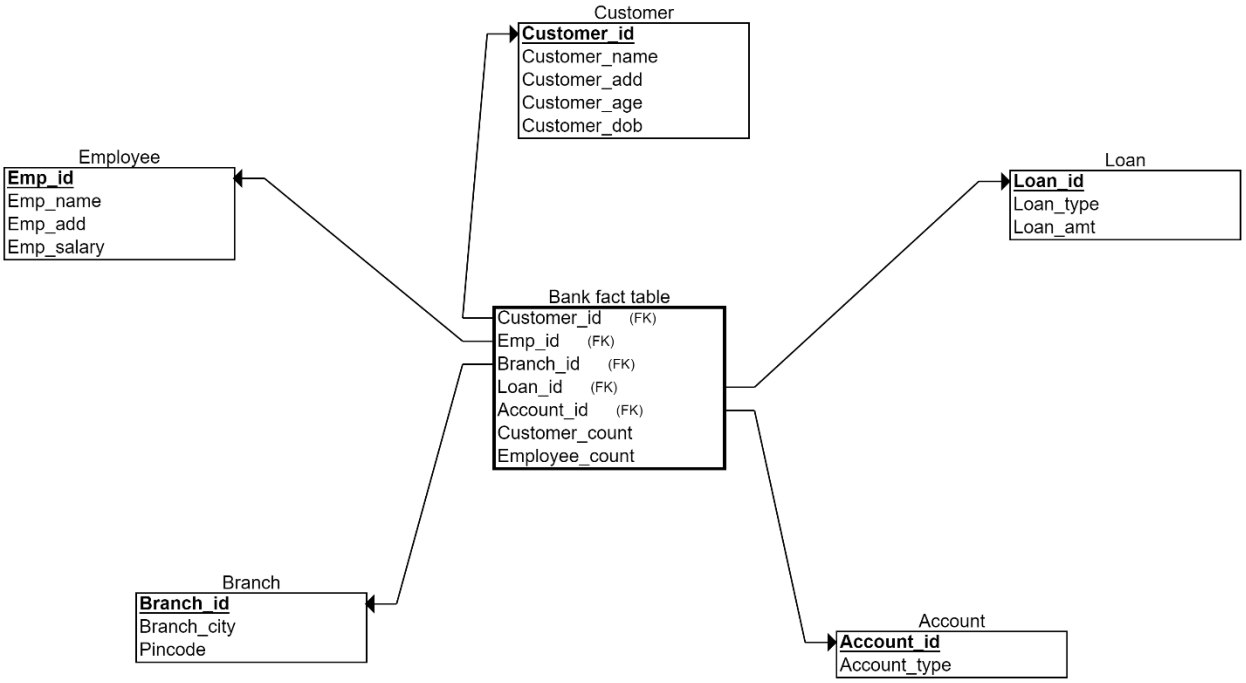
- Step 1: Identify multiple fact tables representing different processes or subjects (e.g., sales and inventory).
- Step 2: Identify the shared dimension tables that will connect to these fact tables.
- Step 3: Define relationships between each fact table and the shared dimension tables.
- Step 4: Draw all fact tables and connect them to the shared dimension tables, creating a constellation of facts and dimensions.

Problem Statement:

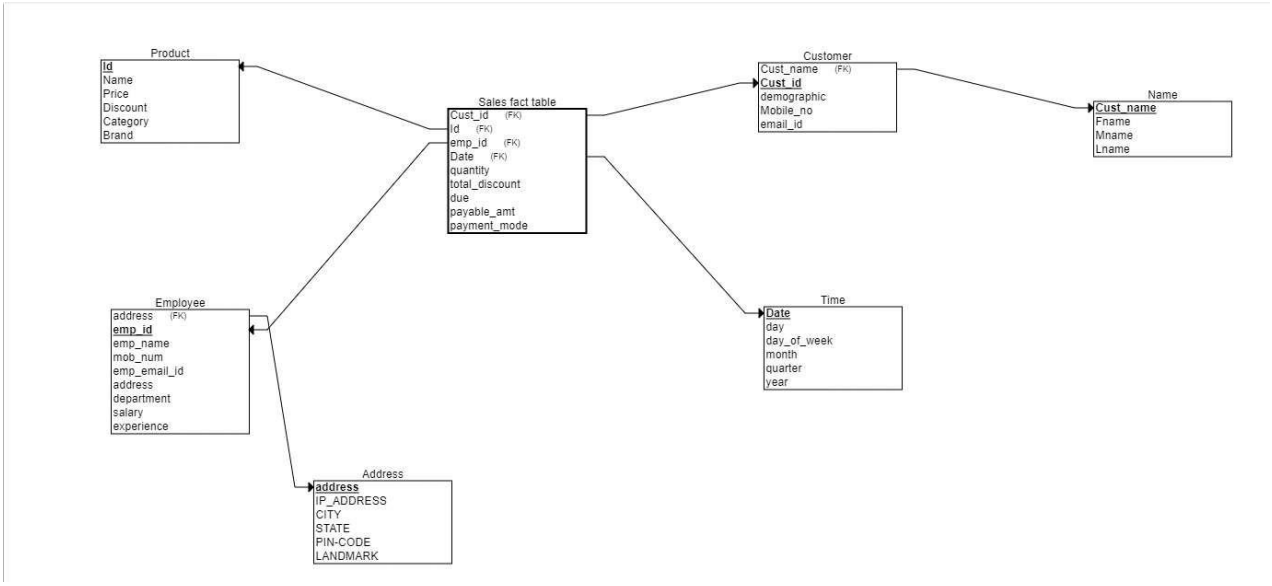
The organization faces challenges in efficiently organizing and analyzing large volumes of data due to an inadequately structured data storage system. This experiment aims to design and implement Star, Snowflake, and Galaxy schemas for a Data Warehouse to determine the most effective architecture for improving data retrieval, consistency, and scalability.

Construction of Star schema, Snowflake schema and Fact Constellation Schema:

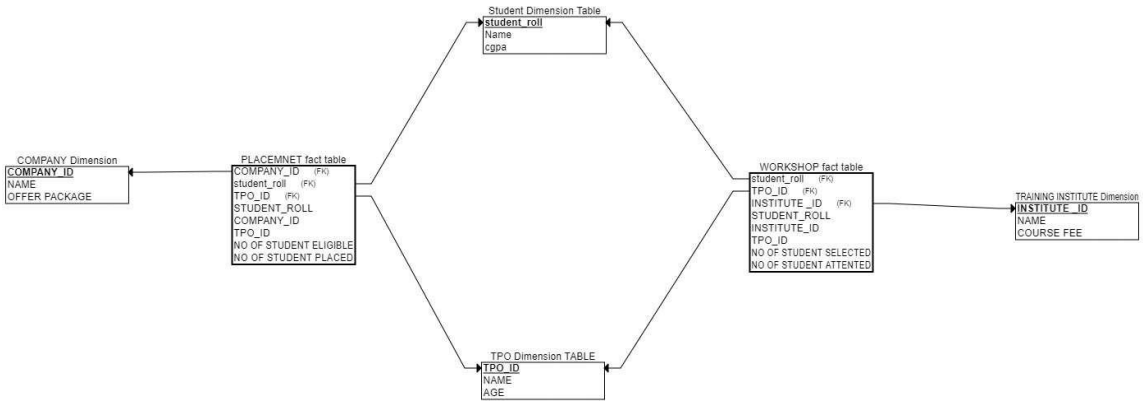
1) STAR Schema



2) Snowflake Schema



Galaxy Schema:



Conclusion:

After designing and implementing the Star, Snowflake, and Galaxy schemas, the experiment revealed distinct advantages and trade-offs for each architecture. The Star schema offered simplicity and faster query performance for straightforward analytical needs, while the Snowflake schema provided better storage efficiency through normalization at the cost of slightly more complex queries. The Galaxy schema, with its ability to handle multiple subject areas, proved to be the most flexible and scalable solution, suitable for complex and large-scale data environments. The choice of schema should be guided by the specific requirements of the data warehouse, balancing the need for performance, efficiency, and scalability.

1. How does the Snowflake Schema compare to the Star Schema in terms of ease of maintenance and scalability?

Ease of Maintenance:

Snowflake Schema**:** Due to its normalized structure, the Snowflake schema has multiple related tables, which makes it more complex to maintain. Changes in the schema, such as adding or modifying dimensions, often require adjustments across several tables, increasing the maintenance burden.
Star Schema: The denormalized structure of the Star schema, with fewer tables, simplifies maintenance. Changes are easier to implement because the dimensions are not split into multiple tables, resulting in a more straightforward schema with less complexity.

Scalability:
Snowflake Schema: The Snowflake schema, with its normalized tables, is better suited for scalability as it reduces data redundancy and storage costs. This efficiency becomes more important as the size of the data warehouse grows, making it easier to manage large datasets.
Star Schema: While the Star schema can handle large datasets, its denormalized nature may lead to data redundancy, which can affect storage efficiency and potentially slow down performance as the data warehouse scales. However, it offers faster query performance due to its simpler joins.

2. How do you manage the complexity of ETL (Extract, Transform, Load) processes in a Fact Constellation Schema?

To manage ETL complexity in a Fact Constellation (Galaxy) schema:

Modular ETL Design: Break ETL into smaller, manageable modules for each fact table and its dimensions.

Metadata Management: Use metadata systems to track data lineage and dependencies.

Automation: Implement ETL automation and orchestration tools for efficient task management.

Incremental Loading: Use incremental loading to process only data changes, reducing ETL load.

Documentation and Monitoring: Maintain documentation and monitor ETL performance to ensure data quality and troubleshoot issues.