NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

NTU Academy for Professional and Continuing Education

(SCTP) Advanced Professional Certificate

Data Science and AI

Learning Objectives

Learners will understand:

- Data Warehouse, Ingestion Model and Dimensional Modeling

Learners will be able to:

- Developing a Dwh on Google Cloud Platform (GCP) BigQuery.
- Transform and load data into a dimensional model using Dbt

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

2.5 Data Warehouse

Module Overview

2.1 Introduction to Big Data and Data Engineering

2.2 Data Architecture

2.3 Data Encoding and Data Flow

2.4 Data Extraction and Web Scraping

2.5 Data Warehouse

2.6 Data Pipelines and Orchestration

2.7 Data Orchestration and Testing

2.8 Out of Core/Memory Processing

2.9 Big Data Ecosystem and Batch Processing

2.10 Event Streaming and Stream Processing

Agenda

- Data Warehouse Characteristics
- Ingestion Model: ETL vs ELT
- Cloud Data Warehouse
- Data Marts
- Dimensional Modeling
  - Fact and Dimension Tables
  - Star Schema
  - Snowflake Schema
  - Slowly Changing Dimensions (SCDs)

Recap: Data Warehouse

- ***Centralized*** repository for integrated, historical data from multiple sources.
- Designed for complex queries, reporting, and analysis for business intelligence.
- Data is ***extracted, transformed, and loaded*** (ETL) to create a unified view.
- Optimized for read-heavy workloads using indexing, partitioning, etc.

Recap: Data Warehouse

Data is often *denormalized* to reduce complex joins.

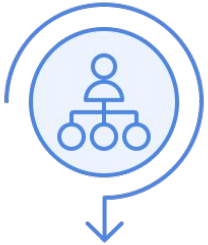
Shortcoming:

Complex, time-consuming, and costly to build and maintain.

Examples:

Snowflake, Amazon Redshift, Google BigQuery.

Data Warehouse Characteristics

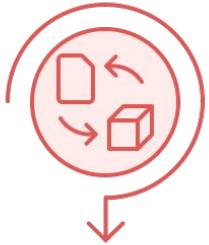


Domain-Driven

Organized around specific subjects.

Integrated

Data transformed into consistent format.



Nonvolatile

Data not altered in real-time.

Time-Variant

Maintains historical data over time.

Optimized Analytics

Designed for complex querying and reporting.

Data Warehouse Characteristics

- **Domain-Driven**: Data is organized around specific subjects (areas of interest).
- **Integrated**: Data from different sources/systems are collected, transformed, and integrated into a consistent format. This ensures that data from various parts of the organization can be analyzed together.
- **Nonvolatile**: Data is typically not altered or updated in real-time. Instead, it is periodically refreshed or loaded from source systems, ensuring that historical data remains intact and can be used for trend analysis.

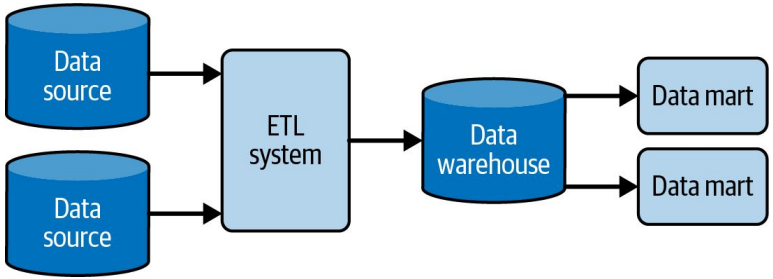
Data Warehouse Characteristics

- **Time-Variant**: Maintains historical data over time, allowing users to analyze trends and changes in data patterns across time periods.
- **Optimized for Analytics**: Designed for complex querying and reporting. Uses specialized structures and indexing to facilitate efficient data retrieval/analysis.

Ingestion Model: ETL

Traditionally, a data warehouse pulls data from application systems by using ETL via data pipelines or data systems.

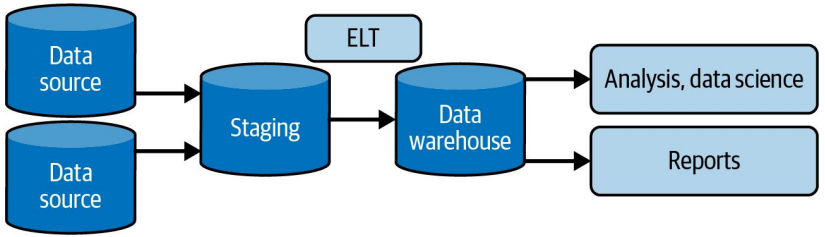
- The **extract** phase pulls data from source systems.
- The **transform** phase cleans and standardizes data, organizing and imposing business logic in a highly modeled form (dimensional modeling).
- The **load** phase pushes data into the data warehouse.



Data is loaded into multiple data marts that serve the analytical needs for specific lines or business and departments.

Ingestion Model: ELT

- With ELT, data gets moved from production systems into a ***staging area*** in the data warehouse.
- Staging in this setting indicates that the data is in a raw form.
- Transformations are handled directly in the data warehouse rather than in external data pipelines or systems like ETL.
- Gaining popularity and adoption due to the rise of cloud data warehouse and streaming.



Events are extracted periodically in a batch or streamed continuously via a CDC process, and loaded into a staging area in data warehouse by data engineers.

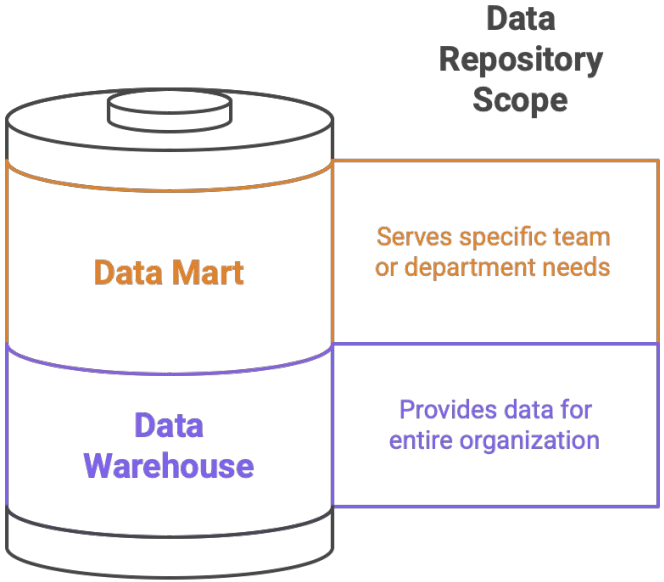
Transformed into dimensional models for analytics within the data warehouse by data analysts or analytics (BI) engineer.

Cloud Data Warehouses

	Traditional Data Warehouse	Cloud Data Warehouse
Data Storage Capacity	Limited by available systems and resources	Virtually limitless with parallel processing
Data Type Handling	Struggles with semi-structured/unstructured data	Optimized for all data types
Scaling	Tedious, time-consuming hardware/software reconfiguration	Instant on-demand scaling, both vertical and horizontal
Cost Structure	High upfront costs, 40% maintenance overhead	Pay-as-you-go model, 50% reduction in IT costs
Performance	Degrades with high volume and query complexity	Dynamic load balancing maintains performance
Availability	Depends on hardware quality and IT team skills	99.9% uptime guarantee from providers

Data Marts

- Subset of a data warehouse focused on providing data and information for a specific business area or department within an organization.
- Smaller and more specialized repository of data, designed to serve the needs of a particular group of users (department or team), rather than the entire organization.



Dimensional Modeling

- Dimensional modeling is a ***data modeling*** technique used in data warehousing.
- It aims to structure data for efficient analytical queries

Includes concepts like

- Fact Tables
- Dimension Tables
- Hierarchies
- Star Schema
- Snowflake Schema
- Degenerate
- Slowly Changing Dimensions (SCD)

Dimensional Modeling

Facts

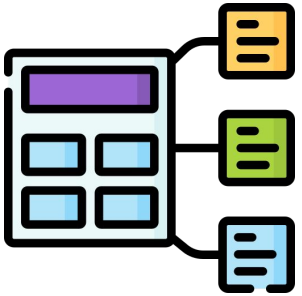
- Quantitative measurements or ***metrics*** of the business process.
- Core data points that represent the performance or behavior of a business, such as sales revenue, units sold, or profit.
- Stored in **fact tables**.

Dimensions

- Descriptive attributes that provide context and characteristics for the facts.
- Dimensions include things like time, geography, products, customers, and more.
- Each dimension is associated with one or more **dimension tables**.

Fact and Dimension Tables

- **Fact Tables**: Typically large and have foreign key relationships to dimension tables. Fact tables often contain aggregated data at *different levels of granularity* to support various analysis requirements.
- **Dimension Tables**: Allow users to slice and dice data in different ways. Dimension tables are usually smaller compared to fact tables.
- **Hierarchies**: Hierarchies represent the relationships between different levels of a dimension. For example, a time dimension hierarchy could have levels like year, quarter, month, and day. Hierarchies help users navigate and drill down into data.

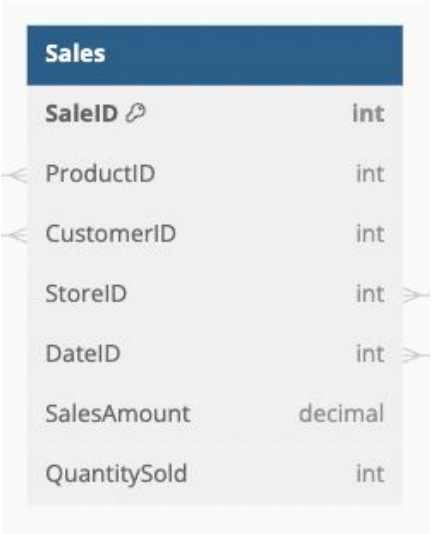


Star Schema

- A type of database schema designed to optimize querying and reporting performance for analytical purposes.
- The name comes from the visual representation of the schema, where the *fact table is at the center*, surrounded by dimension tables that resemble the points of a star.
- The *simplest* and most *widely used* approach to develop data warehouses and dimensional data marts.

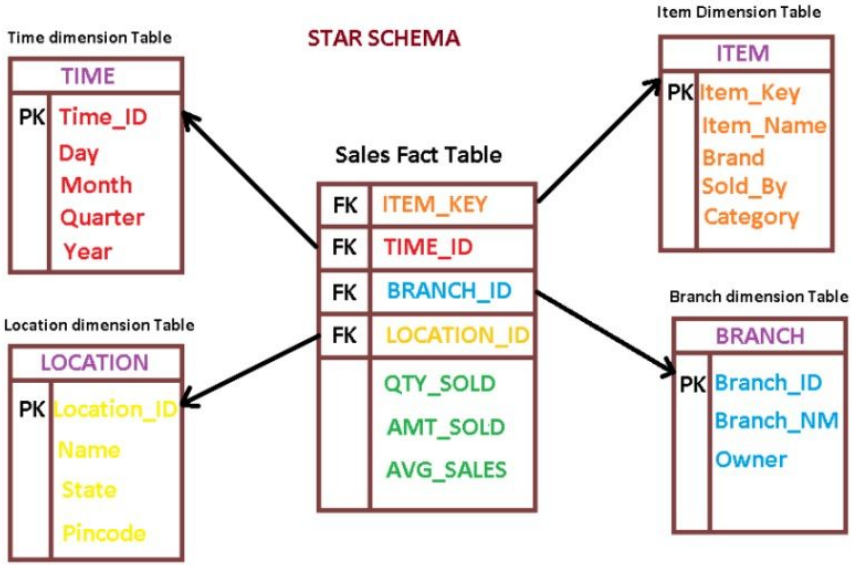
Star Schema

- **Fact Table**: Each row in the fact table corresponds to a specific event or transaction and contains the actual measurements. It often has foreign key references to dimension tables.
- **Dimension Tables**: Provide the context and descriptive attributes for the facts in the fact table.
- They contain the attributes that help define the “who”, “what”, “where”, “when”, and “why” aspects of the data.



Star Schema Example

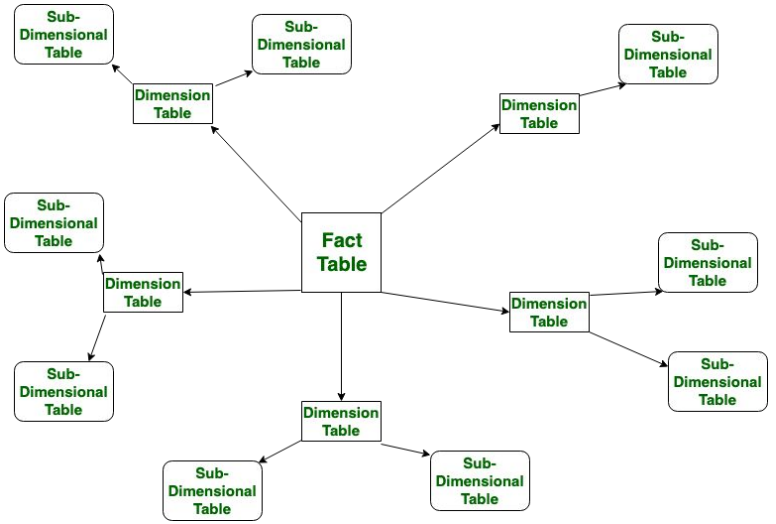
- The **fact table** is *Sales*, which contains the measures of sales quantity and sales amount.
- The **dimension tables** are *Product*, *Customer*, *Store*, and *Date*
- Each dimension table has a primary key that is referenced by a foreign key in the fact table.
- The fact table can be joined with the dimension tables to answer various business questions.
- Example query, "*How much revenue was generated by each product category in each store in each month?*".



Snowflake Schema

An extension of the star schema, designed to *improve data normalization* and *reduce redundancy* by further dividing dimension tables into sub-dimensions.

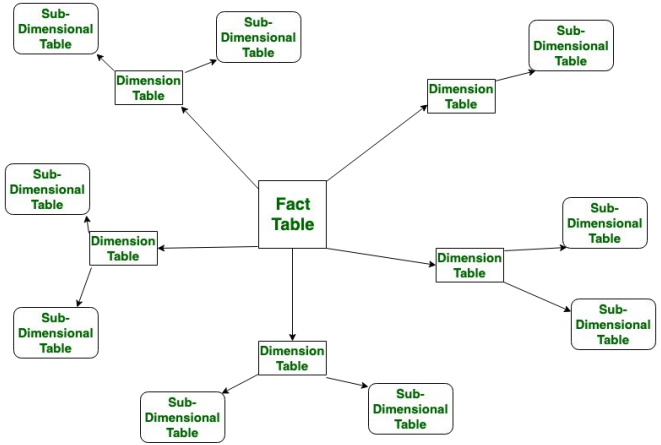
The name comes from the visual representation of the schema, where dimension tables look like snowflakes with their normalized structure.



Source: www.geekforgeeks.com

Snowflake Schema

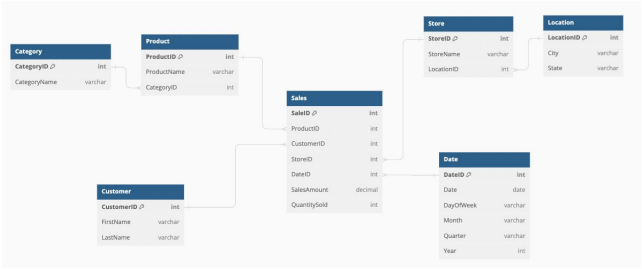
- Fact tables stay the same, but dimension tables are *normalized* into sub-dimensions.
- This means breaking down dimension attributes into multiple related tables.
- They contain more specific attribute information, reducing data redundancy and improving data integrity.



Shortcoming: The normalized structure introduces more complex relationships, requiring additional joins in queries, which can potentially slow down query performance. It also leads to more complex queries.

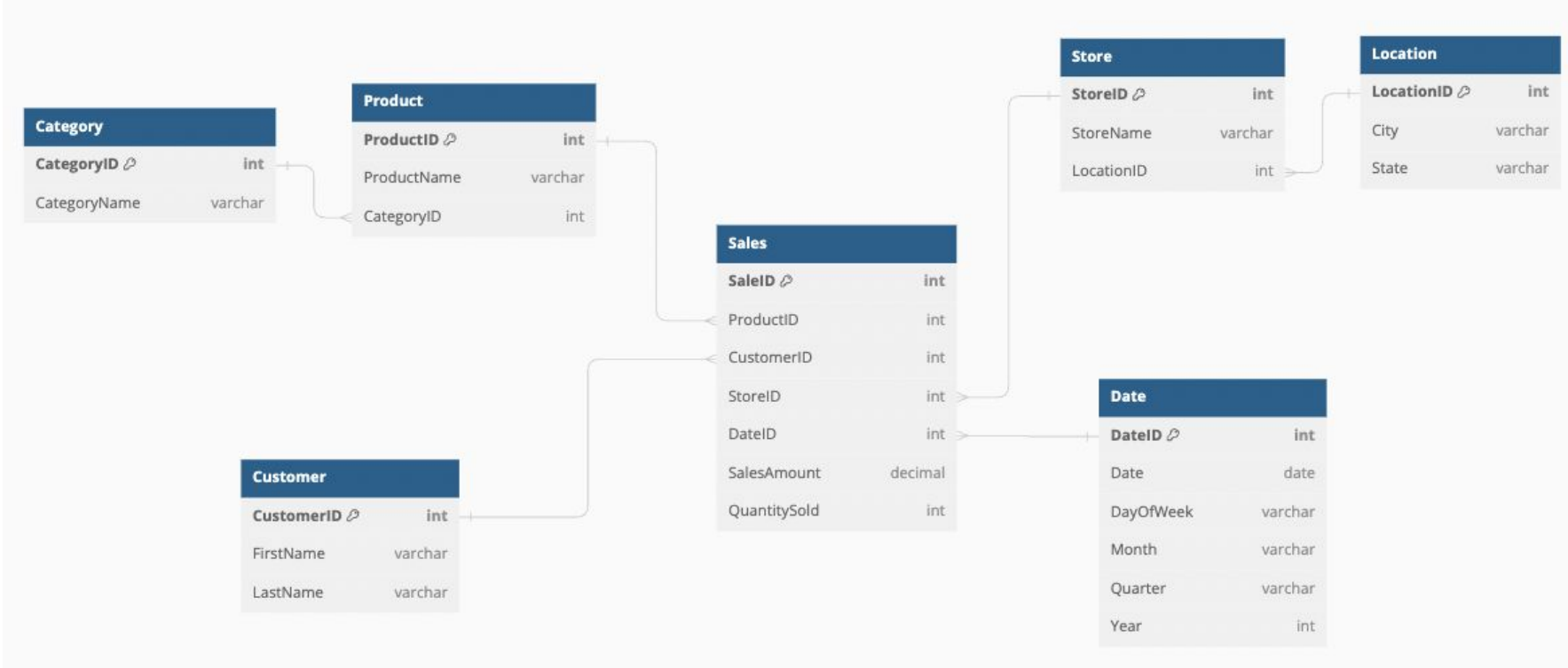
Snowflake Schema Example

- The dimension tables in the previous example are normalized into sub-dimensions.
- The "*Category*" and "*Location*" dimensions are further normalized from the "*Product*" and "*Store*" dimensions, respectively.
- While this provides benefits in terms of data redundancy and maintainability, it can require additional joins when querying compared to the simpler star schema.



(full diagram next slide)

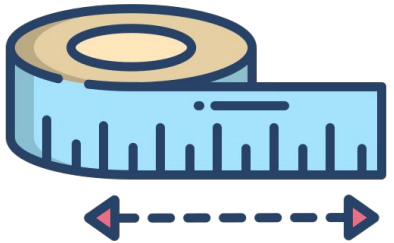
Snowflake Schema Example



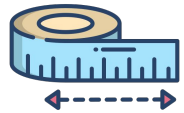
Slowly Changing Dimensions (SCDs)

A concept in data warehousing that deals with managing ***changes to dimension*** data over time.

Dimension attributes can change over time, and SCDs provide strategies for maintaining historical accuracy while accommodating these changes.



Slowly Changing Dimensions (SCDs)



Type 1 SCD - *Overwrite*

When a change occurs in a dimension attribute, the old value is simply overwritten with the new one. This approach doesn't keep a historical record of changes and is suitable when historical data is not important.

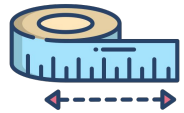
Example: If the supplier relocates the headquarters to Illinois the record would be overwritten.

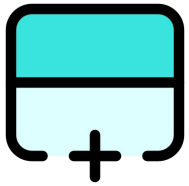
Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State
123	ABC	Acme Supply Co	CA

→

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State
123	ABC	Acme Supply Co	IL

Slowly Changing Dimensions (SCDs)





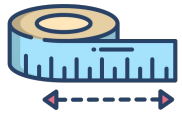
Type 2 SCD - *Add New Row*

When a change occurs in a dimension attribute, a new row is added to the dimension table to represent the new value. This maintains historical data by creating a new entry for each change, allowing you to track how dimension attributes change over time.

Example: If the supplier relocates to Illinois the version numbers will be incremented sequentially.

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State	Version
123	ABC	Acme Supply Co	CA	0
124	ABC	Acme Supply Co	IL	1

Slowly Changing Dimensions (SCDs)



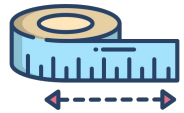
Method 2: Add “start_date” and “end_date” columns.

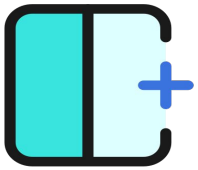
Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State	Start_Date	End_Date
123	ABC	Acme Supply Co	CA	2000-01-01T00:00:00	2004-12-22T00:00:00
124	ABC	Acme Supply Co	IL	2004-12-22T00:00:00	NULL

Method 3: Use an “effective date” and a “current_flag”.

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State	Effective_Date	Current_Flag
123	ABC	Acme Supply Co	CA	2000-01-01T00:00:00	N
124	ABC	Acme Supply Co	IL	2004-12-22T00:00:00	Y

Slowly Changing Dimensions (SCDs)





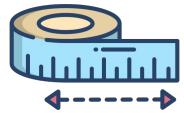
Type 3 SCD - *Add New Attribute*

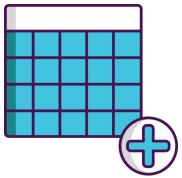
Keeps both the old and new values in the same row, with an additional column to store the previous value. This allows you to see the current and previous values of an attribute without needing to create new rows. However, this approach is limited in its ability to track multiple changes.

Rename “original_supplier” and add “current_supplier” column.

Supplier_Key	Supplier_Code	Supplier_Name	Original_Supplier_State	Effective_Date	Current_Supplier_State
123	ABC	Acme Supply Co	CA	2004-12-22T00:00:00	IL

Slowly Changing Dimensions (SCDs)





Type 4 SCD - *Historical Mini-Dimension*

Separate mini-dimension (history table) is created to store historical changes, while the main dimension table retains only the current values. This reduces the size of the main dimension table and improves performance.

Add a new “supplier history” table

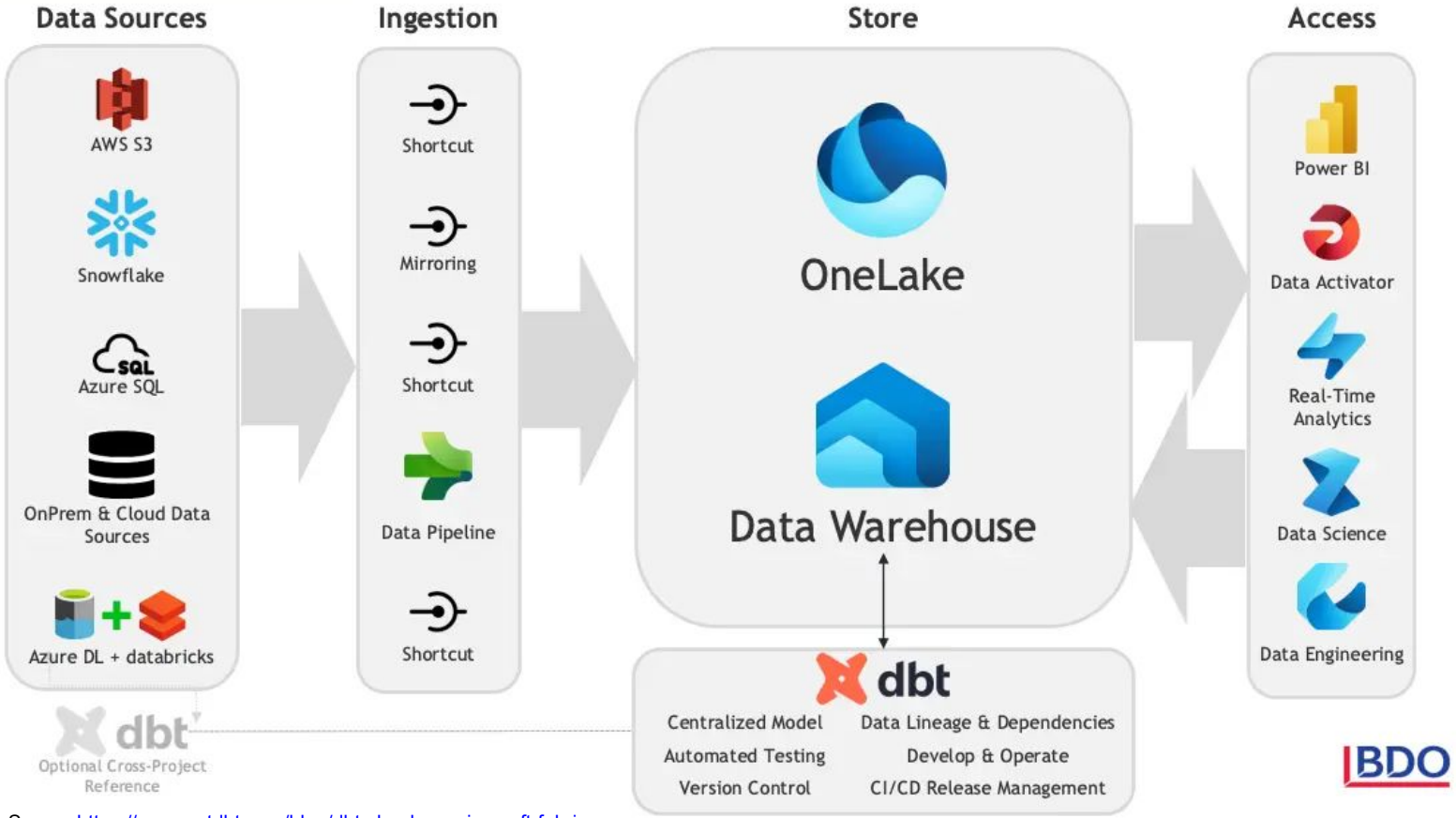
Supplier

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State
124	ABC	Acme & Johnson Supply Co	IL

Supplier_History

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State	Create_Date
123	ABC	Acme Supply Co	CA	2003-06-14T00:00:00
124	ABC	Acme & Johnson Supply Co	IL	2004-12-22T00:00:00

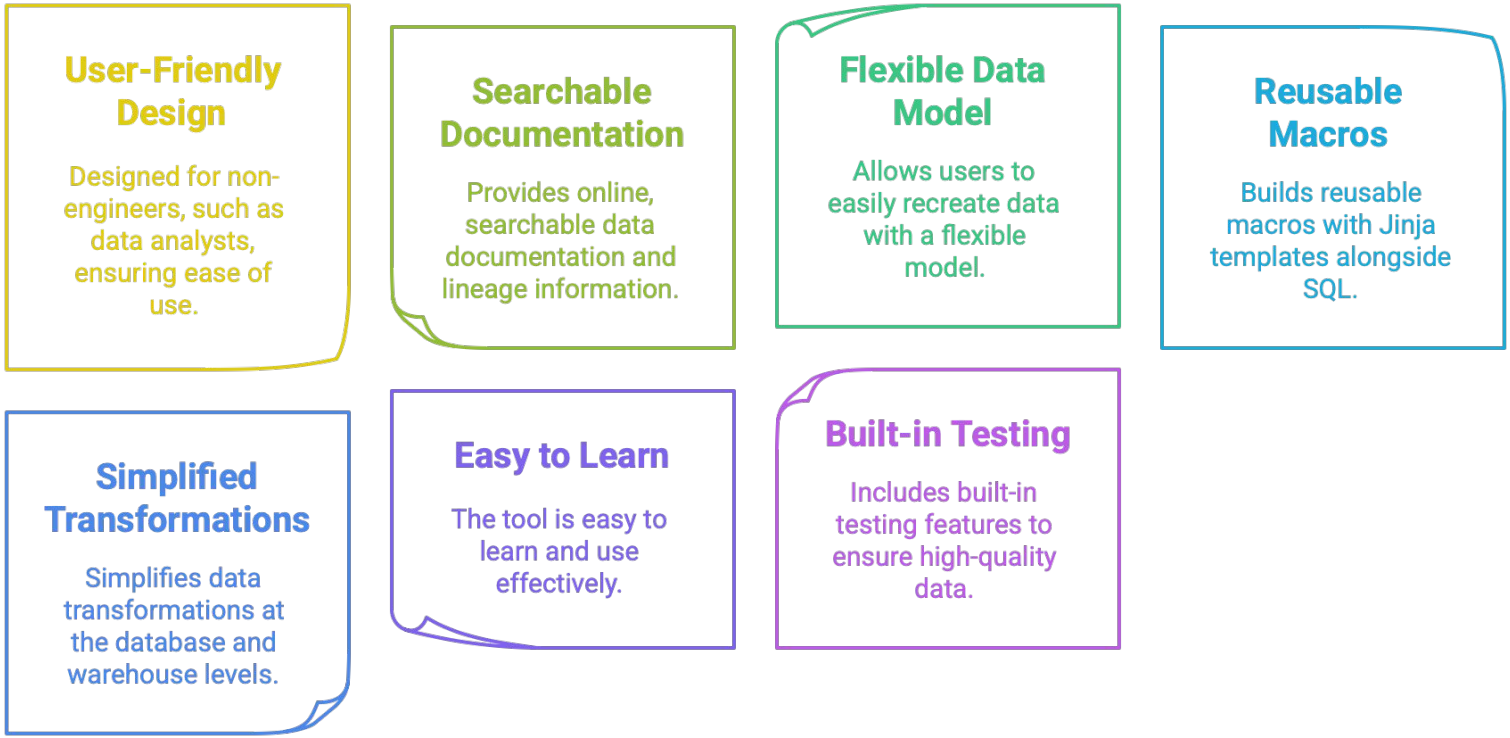
DBT in the Data Architecture



Source https://www.getdbt.com/blog/dbt-cloud-on-microsoft-fabric

Advantages of DBT

Dbt is preferred by a number of organizations to other tools simply because of the pros it offers:




BREAK
TIME!!!

Code-along

Jupyter Notebook

Activity

End of Lesson - Exit Ticket

Survey Link

https://www.menti.com