

# **DATA WAREHOUSE**

## **Schema**

A schema defines the structure of data, including fields or columns, constraints, and the relationships between them. It serves as a blueprint for organizing data, outlining how information is arranged and connected within a system. In the context of databases or data models, a schema establishes the rules and format for storing and accessing data.

## **Data Type**

A data type specifies the nature of data, such as simple types like number, string, date, or float, and complex types like array, struct, or map. It defines the kind of values that can be stored in a field, determining how data is processed and interpreted. Data types are fundamental for ensuring data integrity and enabling appropriate operations on stored information.

## **Metadata**

Metadata provides additional information about data, including descriptions, functional fields, and authorship details. It includes technical information such as when data was created, last modified, and the context in which it exists. Metadata acts as a dictionary or guide, offering crucial details that help users understand and manage the primary data effectively.

## **Data formats**

- **Structured**

Highly organized and fits into a predefined model or schema. It typically includes data in tables with rows and columns, such as relational databases.

Structured data is easily queryable and suitable for analysis using SQL or other structured query languages.

- Semi-structured

Contains some structure but doesn't conform to a rigid schema. It might contain tags, metadata, or other markers that provide some level of organization. Examples include JSON, XML, and NoSQL databases. Although it doesn't adhere to a strict schema, it has some level of self-describing structure.

- Unstructured

Lacks a predefined data model or organization. It is raw and doesn't fit neatly into databases or spreadsheets. Examples include text documents, images, videos, social media posts, emails, and audio files. Analyzing unstructured data often involves techniques like natural language processing or machine learning.

## **Data Warehouse**

### ***Definition***

A data warehouse is a centralized repository that stores large volumes of structured and integrated data from multiple sources. It is designed to support business intelligence, reporting, and data analysis by providing a consolidated view of organizational data.

## **Bottom-Up Approach to Building a Data Warehouse**

The Bottom-up approach is a method for constructing a data warehouse that starts with individual, department-specific data marts and integrates them into a

comprehensive enterprise-wide data warehouse.

### ***Definition***

The Bottom-up approach begins by identifying and developing small, targeted data marts that serve the immediate needs of specific business units or departments. These data marts are built incrementally and can be integrated later into a larger, more comprehensive Enterprise Data Warehouse or EDW.

### ***Instructions***

1. Start Small

Begin by analyzing the data requirements of individual business units or departments.

2. Develop Targeted Marts:

Create Data Marts that are specific to a single area of the organization. This allows for quick delivery of insights and "quick wins" for specific teams.

3. Implement Incrementally:

Build and deploy these data marts one at a time, focusing on immediate needs.

4. Integrate Over Time:

As more data marts are created, they can be gradually integrated to form a larger, more holistic data warehouse. This approach allows for faster implementation and is more agile, as changes can be made to individual marts without disrupting the entire system.

## **Top-Down Approach to Building a Data Warehouse**

The Top-down approach is a strategic methodology for building a data warehouse that starts with a comprehensive, enterprise-wide view and then develops smaller, integrated components.

### ***Definition***

The Top-down approach begins with a strategic and comprehensive methodology that considers the entire organization's data requirements. It involves building a centralized Enterprise Data Warehouse or EDW first, which then serves as the foundation for developing various data marts for different departments.

### ***Instructions***

1. Strategic Planning:

Begin with a strategic plan that considers the data needs of the entire organization.

2. Build the Core:

Design and build a centralized Enterprise Data Warehouse or EDW that provides a holistic view of the company's data.

3. Develop Marts:

Create data marts from the central EDW to serve the specific needs of different departments.

4. Ensure Consistency:

This approach ensures data consistency across the organization by centralizing the data model and integration processes. While it provides a

comprehensive view, it can be more rigid and time-consuming to implement.

## **Build approach FEELT TABELL**

### **Operation Data Store or ODS**

Designed to integrate and consolidate data from various operational systems in near real-time. ODS focuses on transactional data and serves as an interim repository for data before it gets loaded into the data warehouse.

### **Enterprise Data Warehouse or EDW:**

A comprehensive and centralized repository that integrates data from various sources across an entire organization. It serves as the single source of truth for business intelligence and decision-making.

### **Data Mart**

A smaller, more department- specific subset of an Enterprise Data Warehouse. It's designed to serve the needs of a particular business unit, department, or user group. Data marts are optimized for specific business functions or departments, allowing for faster query performance and analysis.

## **Preparation of a data warehouse**

- Data Integration

Process of bringing together data from multiple sources and transforming it into a single, consistent format.

- Data Modeling

Design the structure of the data warehouse. This includes defining the entities, attributes, and relationships between the entities.

- Data Loading

Identify and correct errors and inconsistencies in the data. This is an important step in the data warehousing process, as it ensures that the data is accurate and reliable.

## **Storage and analysis of a data warehouse**

- Data Loading

Populate the data warehouse with data from the source systems. This can be done on a batch or real-time basis.

- Metadata

Describe the structure of the data warehouse, the data types, and the relationships between the entities.

- Data Access and Reporting

Data access and reporting tools allow users to query the data warehouse and generate reports. These tools are typically used by business analysts, data scientists, and other decision-makers.

## **Fact table**

A Fact table in a data warehousing data model consists of one or more numeric facts of importance for a business.

### ***Remark***

Facts must be consistent with the granularity.

## **Types of facts**

- Additive

Can be summed up through all of the dimensions in the fact table.

- Semi-Additive

Can be summed up for some of the dimensions in the fact table, but not the others.

- Non-Additive

Cannot be summed up for any of the dimensions present in the fact table.

## **Features of a fact table**

The level of detail available in a given fact table as well as to the level of detail provided by a star schema.

It is usually given as the number of records per key within the table. In short, the grain of the fact table is the grain of the star schema.

In a Data Warehouse, dimensions represents all points of interest for business and an entry point for fact tables.

And once in your model and can be reused multiple times with different fact tables.

Consider a model containing multiple fact tables, representing different data marts. Now look for a dimension that is common to these facts tables.

Factless table captures the many-to-many relationship between dimensions but contains no numeric or text facts.

They are often used to record events or coverage information.

## **Normalisation**

Normalization is a process in database design that organizes data into separate tables to minimize redundancy and improve data integrity.

The purpose of normalization is to ensure that each piece of data is stored in only one place, reducing the risk of inconsistencies.

## **Denormalisation**

Denormalization involves combining related data from multiple tables into a single table to improve query performance.

The purpose of denormalization is to speed up queries by reducing the need for joins.

## **Star schema**

Star schema is a data warehouse schema where there is only one “fact table” and many denormalized dimension tables.

Fact table contains primary keys from all the dimension tables and other numeric columns of additive, numeric facts.

## **Grain of a star schema**

In Data warehousing, grain refers to the level of detail available in a given fact table as well as to the level of detail provided by a star schema.

It is usually given as the number of records per key within the table. In general, the grain of the fact table is the grain of the star schema.

## **Snowflake schema**

Snowflake schema contains normalized dimension tables in a tree-like structure with many nesting levels.

Snowflake schema is easier to maintain but queries require more joins because of nested levels.

## **Data Warehouse Processing Architecture**



- The scalability of the solution should be demonstrated by the ability to process a huge volume of data and stream it to different destinations, at high speed, in various formats. The data stream should be processed and presented in the required format, at the right time and location, with the minimum impact to the existing infrastructure.
- The architecture should be extensible; new functionality can be implemented in an existing service without regression or alteration. Newer technologies, such as artificial intelligence, can be implemented in an existing service by extending the service's APIs.
- Data security is a critical aspect of the data governance strategy. Data security controls at the source include establishing data access controls and data encryption. Data security controls at the perimeter include data security policies and monitoring access to the data.
- It should be simple and straightforward, and users should be able to work with the data in an efficient and effective manner.

## **Data Lake**

A data lake is a central repository that stores large volumes of raw and unprocessed data from various sources. It typically uses a flat architecture, storing data in its native format, such as CSV, JSON, or Parquet.

Data lakes offer scalability and flexibility, allowing organizations to collect and store vast amounts of data for future processing and analysis.

## **Data Lake use cases**

- Data exploration and analytics

Data scientists and analysts can explore and analyze diverse datasets within the data lake. This supports ad-hoc queries, data discovery, and the extraction of valuable insights.

- Big Data processing

Well-suited for processing big data workloads, including large-scale batch processing and parallel data processing. This is essential for applications in industries like finance, healthcare, and e-commerce.

- Machine Learning and AI

Provide a rich source of data for training machine learning models and implementing artificial intelligence applications. The diverse and voluminous data in the lake can be used to improve model accuracy and robustness.

- Log and Event data

Store logs, events, and other streaming data. This is crucial for monitoring system performance, tracking user activities, and troubleshooting issues.

- IoT Storage

Internet of Things or IoT devices generate vast amounts of data. Data lakes are ideal for storing and analyzing this data to gain insights into device behavior, predict maintenance needs, and optimize operations.

## **Datalakehouse**

Datalakehouse, also known as a unified analytics platform, is an evolution of the datalake concept.

It combines the strengths of a datalake with the capabilities of a data

warehouse.

A datalakehouse provides a unified and structured view of data by introducing schema enforcement and indexing on top of the raw data stored in the datalake. It enables organizations to perform both batch and real-time analytics on the same platform, offering a more streamlined and efficient data processing and analysis experience.

## **Data LakeHouse use cases**

- Analyze data near of real-time

Can accommodate streaming data alongside batch processing. This is beneficial for applications such as fraud detection, IoT analytics, and monitoring.

- Unified analytics

Allow organizations to perform unified analytics by integrating structured and unstructured data in a single platform. This supports comprehensive business intelligence and analytics, providing a holistic view of the data.

- Operation Analytics

Can support operational analytics by integrating data from sensors, machines, and production systems. This facilitates real-time monitoring, predictive maintenance, and process optimization.

- Advanced analytics and ML

Enable organizations to apply advanced analytics and machine learning algorithms to their data. The unified architecture simplifies data preparation and model training by leveraging both structured and unstructured data.

- 360 degree view

can consolidate data from various touchpoints, providing a 360-degree view. This supports personalized marketing, customer service, and experience optimization.

## **Data Lake features**

- ACID Transactions

Delta Lake provides ACID (Atomicity, Consistency, Isolation, and Durability) transactions, which ensures that data updates are always consistent and complete. This makes it well-suited for critical workloads where data integrity is paramount.

- Scalable Metadata

Delta Lake's metadata is designed to scale to petabyte-scale tables with billions of partitions and files. It uses a tiered storage approach to efficiently store and manage metadata across multiple levels.

- Time Travel

Delta Lake enables time travel, allowing users to query and analyze historical versions of data as they evolve over time. This is particularly useful for auditing purposes, identifying data lineage, and performing rollbacks.

- Unified Batch or Streaming

Delta Lake supports both batch and streaming workloads, providing a unified table format for both types of data processing. This eliminates the

need to manage separate tables for batch and streaming data, simplifying data management and processing.

- Schema Evolution or Enforcement

Delta Lake supports schema evolution, allowing users to add or modify table schema without disrupting existing data. It also provides schema enforcement, ensuring that data always adheres to the defined schema.

- Audit History

Delta Lake maintains a detailed audit history of all data changes, providing a comprehensive record of data lineage and modifications. This is valuable for auditing purposes and identifying data quality issues.

## **Data Visualisation**

Data visualization is the representation of information and data using charts, graphs, maps, and other visual tools.

These visualizations allow us to easily understand any patterns, trends, or outliers in a data set. Data visualization also presents data to the general public or specific audiences without technical knowledge in an accessible manner.

- Storytelling

People are drawn to colors and patterns in clothing, arts and culture, architecture, and more. Data is no different—colors and patterns allow us to visualize the story within the data.

- Accessibility

Information is shared in an accessible, easy-to-understand manner for a variety of audiences.

- Visualize relationships

It's easier to spot the relationships and patterns within a data set when the information is presented in a graph or chart.

- Exploration

More accessible data means more opportunities to explore, collaborate, and inform actionable decisions.

## **Data transformations**

- Filtering

Selecting a subset of rows or columns based on specified criteria.

- Sorting

Arranging data in a specific order, often based on one or more columns.

- Aggregation

Combining multiple rows into a single summary row, usually involves mathematical operations like sum, average, count, etc.

- Joining

Combining data from multiple sources based on a common key or attribute.

- Mapping: Replacing values in a column with corresponding values from a lookup table.

- Derivation

Creating new columns or calculations based on existing data.

- Normalization

Ensuring data consistency by organizing it into a standardized format, often involves breaking down data into smaller, more manageable tables.

- Cleaning

Handling missing or incorrect data, dealing with duplicates, and ensuring data quality.

- Validation

Checking data integrity and accuracy to ensure it meets predefined standards.

- Data Type Conversion

Converting data from one type to another, such as changing a string to a date or a number.

- Splitting

Breaking down a column into multiple columns, usually when dealing with composite data.

- Concatenation

Combining data from multiple columns into a single column.

- Duplication Removal

Eliminating duplicate records from the dataset.

- Pivoting or Unpivoting

Transforming data from a wide format to a long format (or vice versa)

- Standardization

Ensuring consistency in units, formats, or naming conventions across the dataset.

- Enrichment

Adding additional information to the data from external sources.

- Masking or Anonymization

Protecting sensitive information by replacing, encrypting, or anonymizing certain data.

- Splitting by Business Logic

Dividing data into subsets based on business rules or criteria.

## **Aggregations**

Aggregations in the context of databases and data analysis involve the grouping and summarization of data. Here is a list of common aggregation functions used in SQL and other data analysis tools:

- Count:

Counts the number of rows in a result set

- Min:

Finds the minimum value in a column

- Max:

Finds the maximum value in a column

- Sum:

Calculates the sum of values in a numeric column



- Avg:

Computes the average value of a numeric column

- Count Distinct:

Count without duplicate values from a result set

## **Slowly Changing Dimension**

In the context of data warehousing and database design, Slowly Changing Dimensions (SCD) refer to the way in which historical data is managed when there are changes to the dimension attributes over time.

There are several types of Slowly Changing Dimensions, commonly denoted as SCD Type 1, SCD Type 2, and SCD Type 3.

### **SCD Type 1**

In SCD Type 1, when a change occurs, the existing dimension record is simply updated with the new data. There is no tracking of historical values, so only the current information is retained.

### **SCD Type 2**

In SCD Type 2, a new record is added to the dimension table for each change. The original record is marked as inactive, and the new record contains the updated information. This way, a history of changes is preserved.

### **SCD Type 3**

In SCD Type 3, a limited amount of historical data is maintained. Instead of creating a new record for each change, only certain attributes are updated, and additional columns are added to the table to capture this limited history.

### **SCD Type 4**

SCD Type 4 is a hybrid approach that combines elements of both SCD Type 1 and SCD Type 2. It typically involves creating a separate table to store historical changes while maintaining a current record in the main dimension table.

## **ELT**

ELT is a variation of the Extract, Transform, Load (ETL), a data integration process in which transformation takes place on an intermediate server before it is loaded into the target. In contrast, ELT allows raw data to be loaded directly into the target and transformed there.

With an ELT approach, a data extraction tool is used to obtain data from a source or sources, and the extracted data is stored in a staging area or database. Any required business rules and data integrity checks can be run on the data in the staging area before it is loaded into the data warehouse. All data transformations occur in the data warehouse after the data is loaded.

## **ELT Process**

- A more recent technology made possible by high-speed, cloud-based servers:

ELT is a relatively new technology made possible because of modern, cloud-based server technologies. Cloud-based data warehouses offer near-endless storage capabilities and scalable processing power. For example, platforms like Amazon Redshift and Google Big Query make ELT pipelines possible because of their incredible processing capabilities.

- Ingest anything and everything as the data becomes available:

ELT paired with a data lake lets you immediately ingest an ever-expanding pool of raw data as it becomes available. There's no requirement to transform the data into a special format before saving it in the data lake.

## **ELT Process**

- Transforms only the data you need:

ELT transforms only the data required for a particular analysis. Although it can slow down the process of analyzing the data, it offers more flexibility because you can transform the data in different ways on the fly to produce different types of metrics, forecasts, and reports. Conversely, with ETL, the entire ETL pipeline and the structure of the data in the OLAP warehouse may require modification if the previously decided upon structure doesn't allow for a new type of analysis.

- ELT has more specific use cases than ETL:

It's important to note that the tools and systems of ELT are still evolving, so they're not as reliable as ETL paired with an OLAP database. Although it takes more effort to set up, ETL provides more accurate insights when dealing with massive pools of data. Also, ELT developers who know how to use ELT technology are generally more difficult to find than ETL developers.