Topic 2: Data Warehouse Architecture

At the end of this topic, students will be to discuss:

- Components of a data warehouse
    - Source systems
    - ETL processes
    - Data warehouse storage
    - Data marts
- Data warehouse design approaches (Top-down vs. Bottom-up)
- Architecture types

1. Overview of Data Warehouse Architecture

---

1. Components of a Data Warehouse

The architecture of a data warehouse typically consists of several key components that ensure the effective collection, storage, and processing of data for business intelligence and analytical purposes. Here are the main components:

1.1 Source Systems

- **Description**: These are the operational systems that provide raw data to the data warehouse. The data from these systems can include transactional, relational, and external data sources.
- **Types of Source Systems**:
    - **Operational Databases (OLTP)**: These systems support day-to-day business operations (e.g., Customer Relationship Management (CRM), Enterprise Resource Planning (ERP)).
    - **External Data Sources**: This could include third-party data (such as social media, external APIs, or market data).
    - **Flat Files**: Simple data files like CSV, Excel, or text files that may contain historical data from other systems.
- **Example**: A company's sales database or an inventory management system is a typical source system.

1.2 ETL Processes (Extract, Transform, Load)

- **Description**: The ETL process is responsible for extracting data from source systems, transforming it to fit the data warehouse schema, and loading it into the data warehouse.
- **Components of ETL**:

    - **Extract**: Pulls data from various source systems (e.g., OLTP databases, flat files).
    - **Transform**: Cleans, enriches, and converts the data into a consistent format. This may involve data validation, standardization, aggregation, and applying business rules.
    - **Load**: Loads the transformed data into the data warehouse storage, ensuring it's ready for querying and analysis.
- **ETL Tools**: Specialized tools like Informatica, Talend, or custom scripts that automate and streamline the ETL process.
- **Example**: Extracting raw sales data, transforming it into a consistent format (e.g., adjusting for time zones or calculating total sales), and loading it into the data warehouse.

1.3 Data Warehouse Storage

- **Description**: The central repository of a data warehouse stores transformed and cleansed data in a structured format for efficient querying and analysis.
- **Components**:
    - **Data Models**: Data is stored in models such as star schema or snowflake schema.
    - **Fact Tables**: Contain numeric data (e.g., sales volume, revenue) that is used for analysis.
    - **Dimension Tables**: Contain descriptive data (e.g., customer information, product categories) to provide context to the fact tables.
    - **Indexes**: Used to speed up query performance.
    - **Partitioning**: Data may be partitioned into smaller segments (e.g., by date) to improve query performance.
- **Storage Systems**: A variety of storage options like relational databases, cloud storage (Amazon Redshift, Google BigQuery), and on-premises data storage systems.

1.4 Data Marts

- **Description**: Data marts are subsets of the data warehouse, often focused on specific business areas, departments, or subject matter (e.g., marketing, sales).
- **Purpose**: They allow for quicker access to specialized data by users in specific departments without accessing the entire data warehouse.
- **Types of Data Marts**:
    - **Dependent Data Mart**: Extracted from the central data warehouse.
    - **Independent Data Mart**: A standalone system that sources data from external systems directly (rarely used in modern architectures).

- **Example**: A sales department's data mart may focus on sales transactions, customer demographics, and promotional performance, whereas a finance data mart may focus on financial performance metrics.

---

2. Data Warehouse Architecture

A **data warehouse architecture** defines how data is collected, stored, and processed for analytical purposes in a data warehouse system. It typically involves multiple layers that ensure efficient data management, transformation, and accessibility for business intelligence (BI) tools and decision-making.

---

1. Data Sources (Data Ingestion Layer)

- **Description**: The first layer is where data originates from various sources.
- **Components**:
  - **Operational Databases**: Relational databases that support day-to-day business operations (e.g., OLTP systems).
  - **External Data**: Third-party data, such as social media, web logs, or public datasets.
  - **Flat Files**: CSV, Excel, or text files that might be used for one-time imports.
  - **APIs and Web Services**: Integrations that fetch data from cloud applications and other services.
- **Example**: A retail company could pull data from its point-of-sale (POS) systems, online sales, and third-party suppliers.

---

2. Data Staging Layer (ETL)

- **Description**: This layer temporarily stores raw, uncleaned data before it is processed and integrated into the data warehouse.
- **Components**:
  - **ETL Tools**: Extract, Transform, and Load (ETL) processes are used to clean, transform, and load data into the data warehouse.
  - **Staging Area**: A temporary storage area where data is held before it undergoes transformations (e.g., raw sales data might be stored temporarily before being aggregated).

    - **Data Transformation**: This includes tasks such as filtering, aggregation, data validation, and data enrichment to ensure data quality and consistency.
- **Example**: Extracting data from a CRM system, transforming it to standard formats, and loading it into the staging area.

---

3. Data Warehouse Layer

- **Description**: This is the core layer of the data warehouse, where data is structured for analysis. It is where cleansed and transformed data resides.
- **Components**:
    - **Data Warehouse**: A central repository of integrated, historical data from various sources.
    - **Schema Design**: Data is often organized in schemas, with common designs being:
        - **Star Schema**: Fact tables connected to dimension tables, used for analytical queries.
        - **Snowflake Schema**: A more complex variation of the star schema where dimension tables are normalized.
        - **Fact Tables**: Store numerical performance measures, like sales, revenue, etc.
        - **Dimension Tables**: Store descriptive data, such as products, time periods, or geographical regions.
- **Example**: A data warehouse for a retail chain might have a fact table for sales transactions and dimension tables for stores, customers, and products.

---

4. Data Mart Layer

- **Description**: Data marts are subsets of data warehouses designed for specific business units or departments. While a data warehouse contains the enterprise's entire data, a data mart contains more focused data for specific users or applications.
- **Components**:
    - **Subject-Oriented**: A data mart may focus on specific topics, like sales or marketing data.

    - **Extracted from Data Warehouse**: Data marts typically extract data from the data warehouse, though they can also be independently fed from source systems.
- **Example**: A marketing team's data mart may focus on customer behavior and campaign performance, while the finance team might have a separate mart for financial analysis.

---

5. Data Presentation Layer (BI Layer)

- **Description**: This layer provides the data in a format that is easily accessible to end-users for analysis and decision-making.
- **Components**:
    - **BI Tools**: Data visualization and reporting tools that access the data warehouse (e.g., Tableau, Power BI, Qlik).
    - **Ad-hoc Querying**: Allows users to query data directly to gain insights in real-time.
    - **Dashboards and Reports**: Predefined reports and dashboards for executive decision-making.
- **Example**: Executives can use dashboards to monitor sales performance and make decisions based on the most up-to-date data from the warehouse.

---

6. Data Governance Layer

- **Description**: This layer ensures that the data within the warehouse is secure, compliant with regulations, and governed effectively.
- **Components**:
    - **Data Quality**: Ensures that the data entering the warehouse is accurate, consistent, and timely.
    - **Security**: Role-based access controls (RBAC) to ensure data privacy and security.
    - **Compliance**: Ensures the data warehouse complies with industry regulations such as GDPR, HIPAA, etc.
- **Example**: Role-based permissions might allow only certain users to access sensitive financial data.

---

7. Metadata Layer

- **Description**: The metadata layer contains information about the data, such as its source, format, and transformations.
- **Components**:
    - **Metadata Repositories**: Store information about data definitions, data structures, and business rules.
    - **Data Lineage**: Tracks where the data came from and how it has been transformed.
    - **Business Glossary**: Provides definitions for terms used across the data warehouse for consistent understanding.
- **Example**: A metadata repository might provide details on how customer data was cleaned and transformed before being loaded into the warehouse.

---

8. Data Integration Layer

- **Description**: This layer integrates data from multiple data sources into the warehouse for analysis and reporting.
- **Components**:
    - **Data Integration Tools**: Manage how different data sources interact and ensure consistency across various datasets.
    - **Data Aggregation**: Combines and summarizes data from various sources to ensure accurate and timely analysis.
- **Example**: Integrating customer data from CRM, purchase data from sales, and behavior data from web analytics platforms.

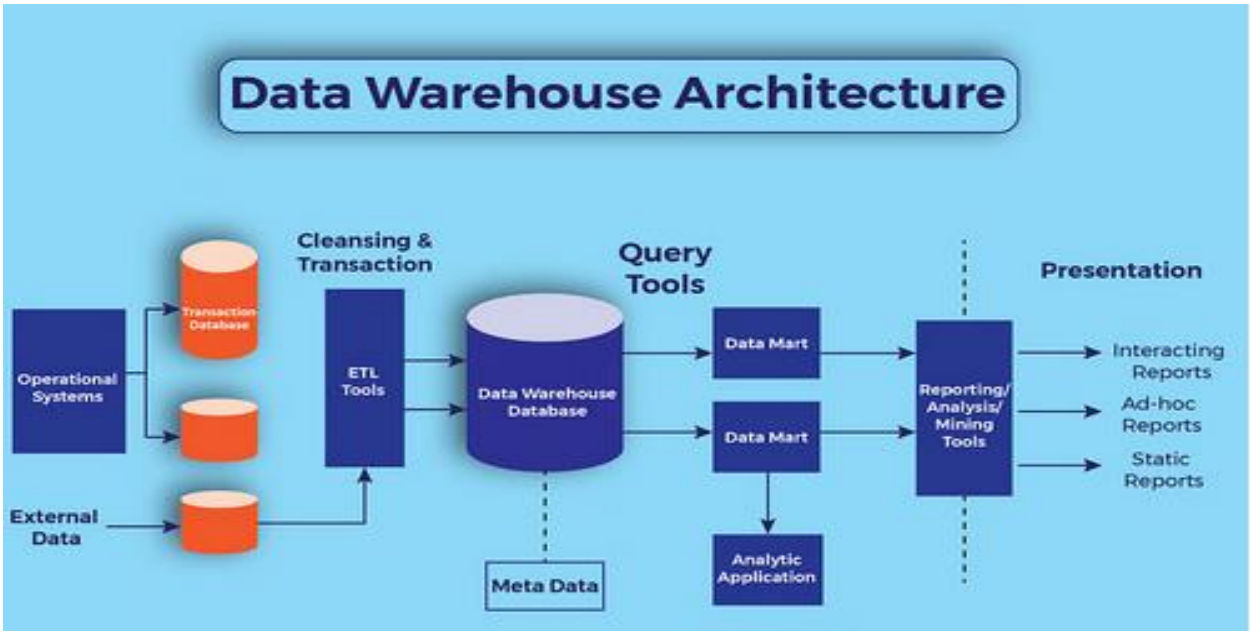
---

8. Query and Analysis Layer

- **Description**: This is where users interact with the data warehouse, running queries, generating reports, and performing analytics.
- **Components**:
    - **Ad-hoc Querying**: Allows users to write and execute their queries to extract data for immediate analysis.

- **OLAP Cubes (Online Analytical Processing)**: Multi-dimensional data structures that allow for fast querying of large datasets for complex analysis.
- **Reporting**: Tools and platforms that generate periodic or custom reports from the data stored in the warehouse.

- **Example**: A sales manager might run a query to check quarterly sales performance, while an analyst might use an OLAP cube to perform trend analysis.



3. Data Warehouse Design Approaches

Data warehouse design involves organizing and structuring data in a way that allows for efficient querying and reporting. There are two primary design approaches for building a data warehouse: **Top-down** and **Bottom-up**. These approaches differ in how data is integrated and structured within the organization.

Top-Down Design (Inmon Approach)

- **Description**: Proposed by William Inmon, the top-down approach emphasizes the creation of an enterprise-wide data warehouse as the first step, with subsequent creation of data marts.
- **Process**:
  1. **Data Warehouse First**: Start by building a central, integrated data warehouse that stores all the organization's data.
  2. **Data Marts**: After the data warehouse is in place, specific data marts are created for particular business areas, leveraging the central warehouse as the source.

- **Benefits**:
  - Centralized control of data, ensuring consistency and quality.
  - Data warehouse serves as a comprehensive source of truth for the entire organization.
- **Drawbacks**:
  - Longer initial implementation time as the enterprise-wide data warehouse must be built first.
- **Example**: A large corporation may build a central data warehouse and then create separate data marts for departments like HR, sales, and finance.

Bottom-Up Design (Kimball Approach)

- **Description**: Proposed by Ralph Kimball, the bottom-up approach starts by building smaller, subject-oriented data marts and gradually integrating them into a data warehouse.
- **Process**:
  1. **Data Marts First**: Create data marts for specific business areas or departments first. These marts are designed for quick implementation and user access.
  2. **Enterprise Data Warehouse**: Eventually, integrate all the data marts into a central data warehouse.
- **Benefits**:
  - Faster implementation, as data marts can be deployed individually and provide value immediately.
  - Easier to tailor data marts to the needs of specific departments.
- **Drawbacks**:
  - Possible data silos or inconsistencies between data marts.
  - Lack of centralized control initially.
- **Example**: A company may first create a data mart for the sales department to quickly provide sales insights, and then later integrate it with the finance data mart and build the central data warehouse.

Hybrid Approach

- **Overview**: The hybrid approach combines elements of both the top-down and bottom-up approaches. It starts with creating a high-level framework for the enterprise data warehouse, then proceeds to build data marts incrementally.
- **Steps**:
  1. Create a broad structure for the enterprise data warehouse.
  2. Begin with building key data marts to serve critical business areas.
  3. Gradually integrate these data marts into the larger data warehouse.

- **Advantages**:
  - Balances speed and scalability.
  - Allows for early wins in terms of business area functionality while maintaining enterprise consistency.
- **Disadvantages**:
  - May be more complex to manage.
  - Requires clear communication between teams to ensure integration is successful.

---

4. Data Warehouse Architecture Types

Data warehouse architecture refers to the structure and design of a data warehouse system, outlining how data is stored, processed, and accessed for analytical purposes. There are several key **data warehouse architecture types** that are commonly used, each with its specific advantages, use cases, and complexity levels.

There are several architectures for building a data warehouse, depending on the needs, scalability, and integration complexity of the organization. Here are the most common ones:

---

1. Single-Tier Architecture

- **Description**: In a single-tier architecture, the goal is to have minimal data storage layers. A single-tier architecture is designed to minimize data redundancy by directly storing all data in a single layer or repository. It is typically used for smaller applications or scenarios with a limited scope where complex analysis is not needed.
- **Key Characteristics**:
  - All data is stored and managed in one layer, typically in a database.
  - There is minimal separation between the storage, transformation, and presentation layers.
- **Advantages**:
  - Simple to implement and manage.
  - Reduces storage costs due to lack of redundancy.
- **Disadvantages**:
  - Scalability issues for large data sets.
  - Limited support for complex analytical processing.
  - Not suitable for large-scale enterprise data warehousing.

- **Components**:
  - Typically, it involves only a single layer where both raw and processed data reside.

    - This approach is rare in large-scale systems as it does not scale well for complex queries or large datasets.
- **Use Case**: Small-scale systems or specific departmental applications with simpler data needs.
- **Example**: A small business or department with minimal data processing requirements.

---

2. Two-Tier Architecture

- **Description**: A two-tier architecture consists of a data staging area (or operational database) and the data warehouse. The two-tier architecture separates the database layer (where data is stored) from the presentation layer (where data is accessed by users), but there is still limited separation between the business logic and storage layers. This type of architecture is more common in smaller to medium-sized data warehousing scenarios.
- The two-tier architecture separates the database layer (where data is stored) from the presentation layer (where data is accessed by users), but there is still limited separation between the business logic and storage layers. This type of architecture is more common in smaller to medium-sized data warehousing scenarios.

- **Components**:
    - **Tier 1**: The source systems or operational database from which raw data is extracted.
    - **Tier 2**: The data warehouse where data is transformed and stored for querying.
- **Benefits**:
    - Easier to implement than a three-tier system.
    - Suitable for smaller organizations or those with less complex reporting requirements.
- **Drawbacks**:
    - Performance can degrade as the data warehouse grows and as more users query the data.
    - The database server may become a bottleneck as more clients access the system simultaneously.
- **Example**: A business with limited transactional data may use a two-tier architecture for simplicity and cost-effectiveness.

3. Three-Tier Architecture

- **Description**: This is the most common architecture for data warehouses and is more scalable and complex. It separates the data extraction, storage, and presentation layers into three distinct tiers. The three-tier architecture is the most commonly used in enterprise data warehouses and separates the architecture into three distinct layers: the **data layer**, the **application layer**, and the **presentation layer**. This separation allows for better scalability, flexibility, and performance.
- **Key Characteristics**:
    1. **Data Layer**: This layer stores the raw data, often structured in relational databases or other data storage systems.
    2. **Application Layer**: This layer handles the business logic, data transformation, cleaning, and ETL (Extract, Transform, Load) processing.
    3. **Presentation Layer**: The final layer is responsible for reporting, analysis, and data visualization, where users interact with the system.
- **Components**:
    - **Tier 1 (Data Sources)**: This layer includes operational databases, external sources, and data lakes.
    - **Tier 2 (Data Staging and ETL)**: Data is extracted, cleaned, transformed, and loaded into the warehouse.
    - **Tier 3 (Data Warehouse and BI Tools)**: The central data warehouse stores integrated data, which is accessed via BI tools and reporting platforms.
- **Benefits**:
    - Scalable, efficient, and flexible for large-scale enterprise systems.
    - Allows for centralized control and integration across different business units.
    - Improved performance by separating concerns (data storage, business logic, and presentation).
    - Enhanced security and data integrity, as the data layer is isolated.
- **Disadvantages**:
    - More complex to implement and maintain.
    - Can be costly due to the need for separate components and infrastructure for each layer.
- **Example**: A large multinational company with multiple departments, regions, and a vast amount of data will likely use a three-tier architecture to efficiently manage its data.

---

4. Cloud-Based Architecture

- **Description**: In a cloud-based data warehouse architecture, the data warehouse and its components are hosted on cloud platforms (e.g., AWS, Google Cloud, Microsoft

Azure). With the rise of cloud technologies, many organizations are adopting cloud-based data warehouses, which provide flexibility and scalability. Cloud architectures often use platforms like Amazon Redshift, Google BigQuery, or Snowflake.

- **Key Characteristics**:
  - The data warehouse is hosted and managed in the cloud.
  - Scalable infrastructure that can grow with data and processing needs.
  - Typically integrated with cloud services for ETL processing, analytics, and machine learning.

-
- **Components**:
  - Data storage, ETL tools, and BI tools are hosted in the cloud.
  - Offers scalability, flexibility, and lower infrastructure costs.
- **Benefits**:
  - Scalable and flexible infrastructure that can handle massive volumes of data.
  - Lower upfront capital expenditure and better disaster recovery options.
  - Easier to implement and maintain than on-premises data warehouses.
  - Seamless integration with other cloud-based tools and services.
- **Disadvantages**:
  - Dependency on internet connectivity and the cloud provider.
  - Security and compliance concerns with storing data off-premises.
- **Example**: Companies like Netflix and Airbnb use cloud-based data warehouses to handle large amounts of data efficiently.

---

5. Hub-and-Spoke Architecture

- **Overview**: Hub-and-spoke architecture is a variation of the three-tier model, commonly used for large-scale enterprise data warehouses. The central "hub" is the enterprise data warehouse (EDW), and the "spokes" are the departmental data marts. This architecture is designed for data integration across different business functions.
- **Key Characteristics**:
  - The central hub contains a large, integrated data warehouse with data from various sources.
  - Spokes represent data marts that focus on specific business areas (e.g., sales, finance) and are fed by the central warehouse.
- **Advantages**:
  - Centralized control of the data, ensuring consistency across departments.

    - Scalability: Data marts can be added or removed as the business grows or changes.
    - Better performance for departmental reporting as each department has its data mart.
- **Disadvantages**:
    - Higher implementation complexity and cost.
    - The central warehouse can become a bottleneck if not properly designed and managed.
    - Requires careful management of data consistency between the hub and spoke systems.

6. Data Vault Architecture

- **Overview**: The data vault model focuses on scalability and flexibility in handling large, complex datasets. It separates the data into three main components: **hubs**, **links**, and **satellites**. Data vault architecture is well-suited for dynamic environments where data sources are frequently changing.
- **Key Characteristics**:
    - **Hubs** represent the core business entities (e.g., customer, product).
    - **Links** capture relationships between business entities.
    - **Satellites** store descriptive, historical data related to the hubs and links.
- **Advantages**:
    - Highly scalable and flexible for growing and changing datasets.
    - Well-suited for integrating data from multiple sources with varying structures.
    - Can handle complex historical data and multiple data versions.
- **Disadvantages**:
    - Complex to implement and maintain.
    - May require specialized expertise for management and development.

7. Kimball Architecture (Dimensional Model)

- **Overview**: The **Kimball methodology** focuses on organizing data into **facts** (quantitative data) and **dimensions** (descriptive data) in a way that is optimized for querying and reporting. This architecture is often implemented using a **star schema** or **snowflake schema**.
- **Key Characteristics**:
    - Data is organized into a central **fact table**, which stores transactional data (e.g., sales, purchases).

    - Surrounding **dimension tables** contain descriptive data (e.g., time, location, product).
    - The data warehouse is designed for fast query processing and reporting.
- **Advantages**:
    - Simple and intuitive design, making it easy for end-users to query.
    - Fast performance for analytical queries.
    - Easy to scale for adding more data marts or expanding reporting capabilities.
- **Disadvantages**:
    - Potential for data redundancy in dimension tables (especially in snowflake schema).
    - May require more storage space due to the denormalization of data.

8. Inmon Architecture (Top-Down Approach)

- **Overview**: The **Inmon approach**, named after Bill Inmon, focuses on building an **enterprise data warehouse (EDW)** first, which integrates data from across the organization. **Data marts** are then created for specific departments or business areas, drawing from the EDW.
- **Key Characteristics**:
    - The EDW serves as the central repository that integrates data from all sources.
    - Data marts are created based on specific needs (e.g., for sales, HR, finance) and draw data from the EDW.
    - Emphasizes the use of **normalized** data to avoid redundancy.
- **Advantages**:
    - Centralized data storage ensures consistency and avoids duplication.
    - Enables comprehensive reporting and analysis across the enterprise.
    - Better suited for large, complex organizations.
- **Disadvantages**:
    - Longer implementation time as the central EDW needs to be established first.
    - May be costly due to the complexity of integrating data from various sources.

Note:

The choice of data warehouse architecture depends on factors such as the scale of operations, complexity of data, budget, and performance requirements. For large, complex organizations, **Hub-and-Spoke** or **Inmon's EDW approach** may be suitable, while organizations looking for simplicity and faster implementation might opt for **Kimball's**

dimensional model or **cloud-based architectures**. Each architecture has its strengths and weaknesses, so it is essential to align the architecture with business goals and data needs