SOUMYADEEP SINHA
WEEK 1 - DAY 1- 28-10-2024

DATA WAREHOUSING

ROLES AND RESPONSIBILITIES

1. **Data Engineer :**
    - ETL Process: Extracts, transforms, and loads data into databases (e.g., customers, sales, products).
    - Focuses on data integration and infrastructure.
2. **Data Analyst :**
    - ELT Process: Extracts data from data engineers, loads it into databases, and transforms it for analysis.
    - Analyses data to derive insights for decision-making.
3. **Data Scientist :**
    - Combines skills of a data engineer and machine learning engineer to make predictions and analyse complex data.
4. **CEO :**
    - Uses data analytics for financial insights, such as revenue comparisons and budget allocations.

TYPES OF DATA :

1. Structured Data

Structured data is **highly organised and easily searchable** because it follows a predefined schema, usually in a tabular format with rows and columns. This structure allows relational databases, such as SQL-based systems, to efficiently store, retrieve, and manage structured data.

Characteristics:

- Data is organised into tables with columns (attributes) and rows (records).
- Follows a predefined schema, making it easy to input, query, and analyse.
- Enforces data types and constraints (e.g., integers, strings, dates).
- Easily manageable with SQL queries.

Examples:

- Sales transactions: Tables with columns for date, customer ID, product ID, quantity, and amount.
- Employee records: Tables with columns for employee ID, name, position, department, and salary.
- Financial data: Tables with columns for account numbers, balances, and transaction types.

Use Cases:

- Used extensively in transactional databases (OLTP) and data warehouses (OLAP) for business intelligence, analytics, and reporting.
- Ideal for applications that require ACID (Atomicity, Consistency, Isolation, Durability) compliance, like banking and inventory management.

2. Unstructured Data

Unstructured data lacks a predefined structure, making it difficult to fit into traditional row-column databases. It doesn't conform to a fixed schema, so its format and organisation are inconsistent. This type of data is typically stored in file systems, object storage, or NoSQL databases that can handle diverse formats.

Characteristics:

- Doesn't follow a rigid schema or structure, which makes it challenging to search and organise.
- Often requires metadata or indexing techniques to enable retrieval and search.
- Can include a variety of formats, such as images, audio files, videos, and text documents.

Examples:

- Social media posts, emails, and messages, where structure varies widely and content is often free-form.
- Multimedia files like images, audio, and video, which have no inherent structure.
- Web content, such as HTML pages and blogs, where format and information vary.

Use Cases:

- Ideal for applications involving content management, social media analytics, media libraries, and customer service data.
- Often managed with NoSQL databases like MongoDB or object storage solutions like Amazon S3, which can handle diverse data types and scale horizontally.

3. Semi-Structured Data

Semi-structured data has some organisational structure but lacks the rigid schema associated with structured data. It's often organised in a hierarchical or key-value format, making it more flexible than structured data but more organised than unstructured data. Semi-structured data can be efficiently processed and stored in NoSQL databases, which support a schema-on-read approach, meaning the data can be processed without a predefined schema.

Characteristics:

- Contains tags or markers to separate data elements, making it possible to search and analyze parts of the data.
- Lacks a strict schema but has an internal structure (like key-value pairs or hierarchies).
- Often stored in formats that are human-readable and machine-readable, such as JSON, XML, or YAML.

Examples:

- JSON and XML documents, which organise data with key-value pairs or nested hierarchies.
- Email metadata (e.g., sender, recipient, timestamp), while the message body itself may be unstructured.
- Sensor data from IoT devices, where each device may have different types of data but follows a general format.

Use Cases:

- Commonly used in APIs, web data, and application logs, where data needs to be flexible but still searchable.
- Well-suited for NoSQL databases like MongoDB and Couchbase, which store semi-structured data in document-oriented or key-value formats.

DATA WAREHOUSE :

A Data Warehouse (DW) is a specialised system designed to store and manage large volumes of data collected from various sources to support reporting and analytical tasks. Unlike traditional Database Management Systems (DBMS), which primarily handle transactional data (e.g., day-to-day operations), data warehouses are optimised for analysing and querying vast datasets over long periods.

In a data warehouse, **fact tables** and **dimension tables** are fundamental components of the schema, particularly in star and snowflake schema designs.

1. Fact Table

A fact table is the **central table in a star schema** that holds quantitative data or metrics (facts) about a business process. These facts are often numeric and aggregatable, such as sales amount, units sold, or revenue. Fact tables are designed to capture transactional data and can contain a large number of records.

Characteristics:

- Contains **measurable data**: Facts are typically numeric and additive (e.g., sales amount, quantity).
- Includes **foreign keys** that link to dimension tables, enabling context around the facts.
- Contains a **high volume of records**, as it captures each transaction or event.
- Usually normalised to improve performance and reduce data redundancy.

Example: A Sales Fact Table

Date_Key	Product_Key	Store_Key	Sales_Amount	Units_Sold
20231025	102	3	500.00	5
20231025	101	4	300.00	3
20231026	103	3	200.00	2

In this example, `Sales_Amount` and `Units_Sold` are the measurable facts, while `Date_Key`, `Product_Key`, and `Store_Key` are foreign keys linking to dimension tables.

2. Dimension Table

A dimension table is a **descriptive table that provides context** to the facts in a fact table. Dimension tables contain attributes (often textual or categorical) that describe aspects of a business process and allow users to filter, group, and explore data along various perspectives.

Characteristics:

- Stores **descriptive data**: Attributes are usually textual or categorical (e.g., product name, category).
- Contains a **primary key** that uniquely identifies each row and links to the fact table via foreign keys.
- Contains a **smaller number of records** compared to fact tables but has a wider range of descriptive columns.
- Often **denormalized** to include all relevant information for easy querying.

Example: A Product Dimension Table

Product_Key	Product_Name	Category	Brand	Price
101	Laptop	Electronics	Brand A	300.00
102	Office Chair	Furniture	Brand B	150.00
103	Wireless Mouse	Electronics	Brand C	20.00

Here, `Product_Key` is the primary key that links to the `Product_Key` in the fact table. Attributes like `Product_Name`, `Category`, and `Brand` provide descriptive details about each product.

Key Differences Between Fact Tables and Dimension Tables

Aspect	Fact Table	Dimension Table
Purpose	Stores quantitative data for analysis	Stores descriptive data to provide context
Data Type	Numeric facts (measurable metrics)	Textual attributes (descriptive details)
Key Type	Contains foreign keys to dimension tables	Contains primary keys that link to fact tables

Record Volume	High (large number of rows, grows quickly)	Relatively low (fewer rows, stable over time)
Example Data	Sales amount, quantity, revenue	Product names, categories, customer information

OLAP AND OLTP

OLAP (Online Analytical Processing) and OLTP (Online Transaction Processing) are two primary types of database systems, each serving different purposes:

OLAP (Online Analytical Processing)

OLAP systems are designed for **data analysis and reporting**, offering multidimensional views of historical data for business intelligence and strategic decisions. They are optimised for **read-heavy** operations **(80% read, 20% write)** due to complex, aggregation-based queries.

- **Use Cases**: Business intelligence, forecasting, trend analysis.
- **Data Structure**: Multidimensional cubes.
- **Operations**: Roll-up, drill-down, slice, and dice.
- **Tools**: **Microsoft SQL Server Analysis Services (SSAS)**, Oracle OLAP, IBM Cognos.
- **Example**: Data warehousing, where data is updated periodically and analysed for insights.

OLTP (Online Transaction Processing)

OLTP systems support **transactional data processing** for day-to-day operations. They are optimised for **write-heavy** operations, as they handle high volumes of simple transactions They are optimised for **write-heavy** operations **(20% read, 80% write)**.

- **Use Cases**: Daily business activities like order processing, banking transactions.
- **Data Structure**: Normalised tables.
- **Operations**: Insert, update, delete.
- **Systems**: MySQL, PostgreSQL, Oracle, Microsoft SQL Server.
- **Example**: Customer order systems requiring real-time updates.

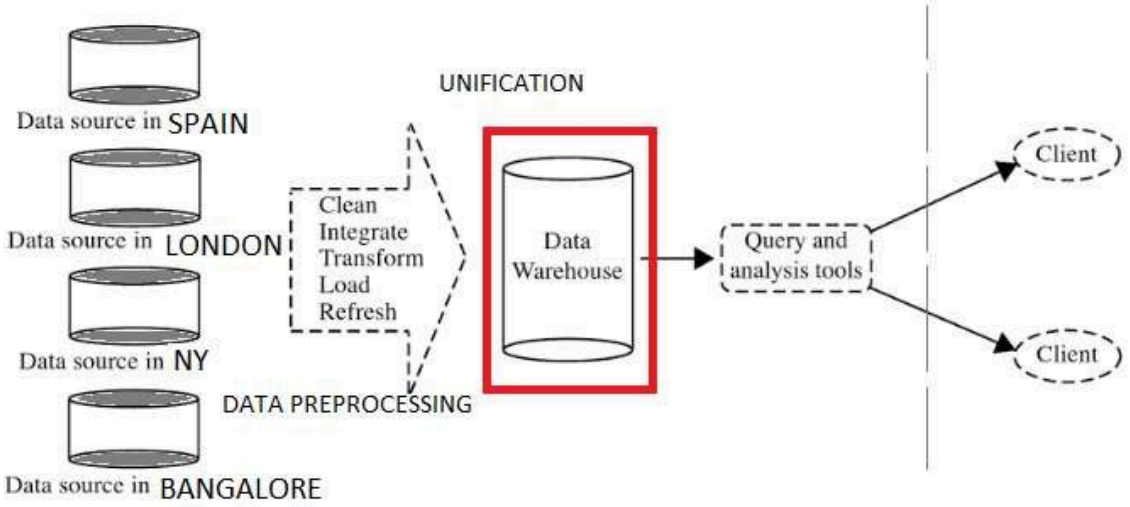
How Data Warehousing Works?

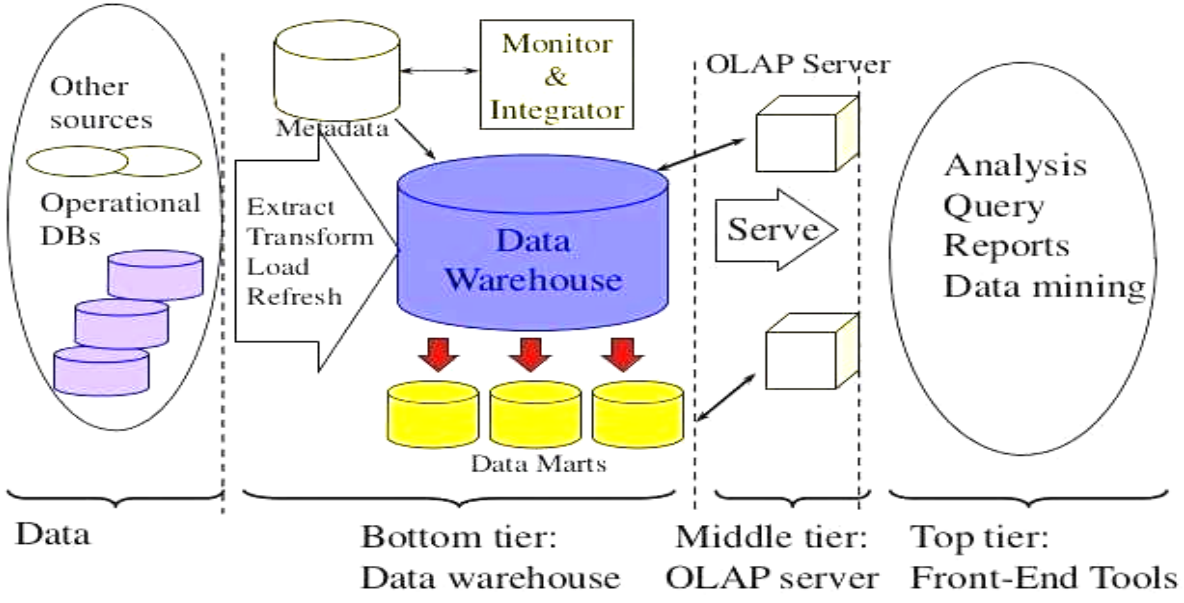
The architecture of a data warehouse consists of three tiers. The bottom one is the database server, where data is loaded and stored. The middle one is the analytics engine that analyses the data.The top one is the front-end client representing the result through analysis, reporting, and data mining tools. The entire work of a data warehouse depends on these tiers.

Data warehousing is the process of collecting, storing, and managing large amounts of data from different sources to enable reporting and analysis. Here's a brief overview of how it works:

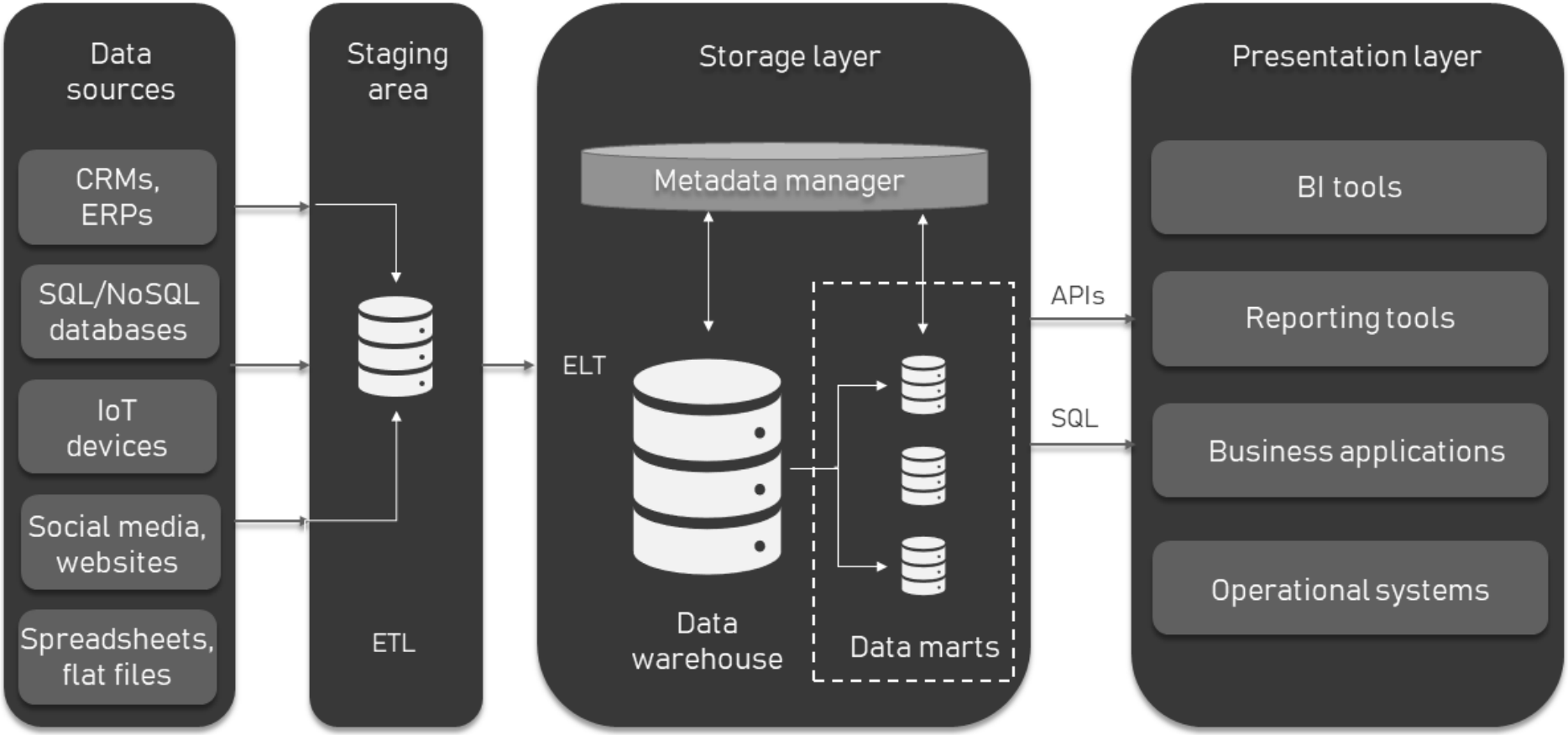
1. **Data Collection**:
    - Data is gathered from multiple **source systems** like transactional databases, CRM, ERP, and external data sources.
2. **ETL Process (Extract, Transform, Load)**:
    - **Extract**: Data is pulled from the source systems.
    - **Transform**: Data is cleaned, formatted, and transformed to fit the warehouse's structure.
    - **Load**: Transformed data is loaded into the data warehouse.
3. **Data Storage**:
    - Data is stored in a **central database** (the data warehouse) using an optimized schema, typically a **star** or **snowflake schema**, to support complex queries and analysis.
4. **Data Access**:
    - Users access data through **OLAP (Online Analytical Processing)** tools for multidimensional analysis, generating reports and dashboards.
5. **Data Refresh**:
    - Data is periodically updated (daily, weekly, or monthly) to ensure the warehouse contains recent information for analysis.
6. **Reporting and Analytics**:
    - Business users can run complex queries, generate insights, and create reports for decision-making, leveraging the warehouse's consolidated and structured data.

In short, a data warehouse integrates and organises data from multiple sources into a central repository, making it accessible and useful for business analysis and strategic decision-making.


UNIFICATION
Data source in SPAIN
Data source in LONDON
Data source in NY
Data source in BANGALORE
Clean
Integrate
Transform
Load
Refresh
DATA PREPROCESSING
Data Warehouse
Query and analysis tools
Client
Client


Other sources
Operational DBs
Extract
Transform
Load
Refresh
Metadata
Monitor & Integrator
Data Warehouse
Data Marts
OLAP Server
Serve
Analysis
Query
Reports
Data mining
Data
Bottom tier: Data warehouse
Middle tier: OLAP server
Top tier: Front-End Tools

ENTERPRISE DATA WAREHOUSE COMPONENTS



DATA MARTS :

A data mart is a focused subset of a data warehouse designed to serve a specific department or business function, like sales or finance. It holds only the data relevant to that area, enabling faster access and more efficient analysis. Data marts improve query performance, are cost-effective, and can be implemented faster than a full data warehouse.

Types:

1. Dependent: Extracted from a central data warehouse.
2. Independent: Created directly from operational systems without a central data warehouse.
3. Hybrid: Combines data from a data warehouse and external sources.

Benefits: Faster access to relevant data, simplified views, better performance for department-specific queries, and customization for department needs.

TYPES OF SCHEMA IN DATA WAREHOUSING

1. Star Schema

- **Structure**: Consists of a central **fact table** (containing quantitative data, like sales) linked to multiple **dimension tables** (holding descriptive data, like customer, time, product).
- **Design**: Dimension tables are directly connected to the fact table, resembling a star shape.
- **Advantages**: Simple design, easy to understand, and optimised for **query performance**.
- **Use Case**: Commonly used in OLAP systems for quick aggregations and reporting.

2. Snowflake Schema

- **Structure**: Similar to the star schema but with **normalised dimension tables**. Dimension tables are further broken down into related sub-dimension tables.
- **Design**: More complex, resembling a snowflake shape due to additional layers in dimension tables.
- **Advantages**: Reduces data redundancy and saves storage space.
- **Use Case**: Suitable when storage optimization is important, though it may increase query complexity.

3. Galaxy Schema (or Fact Constellation Schema)

- **Structure**: Combines multiple star schemas, where **multiple fact tables** share dimension tables.
- **Design**: Complex structure that looks like a galaxy or constellation due to multiple interconnected fact and dimension tables.
- **Advantages**: Supports complex data warehousing scenarios and **multiple business processes**.
- **Use Case**: Used in large data warehouses with diverse analytical requirements (e.g., when an organisation needs to analyse sales and inventory data together).

SUMMARY

Data Management

- **Data Marts**: Subsets of data warehouses, organised by subject for faster retrieval (e.g., categorising movies).
- **Operational Data Store (ODS)**: Supports real-time reporting and operational analysis.
- **Archiving**: Periodic movement of data from OLTP systems to OLAP for historical analysis.

Data Structures

- **Entity-Relationship Diagram (ERD)**: Used in OLTP for master and transactional tables.
- **Dimensional Modelling**: In OLAP, uses dimension and fact tables for analysis.

Schema Types

- **Star Schema**: Dimension tables are directly connected to a central fact table.
- **Snowflake Schema**: Normalised dimension tables, breaking data into related sub-tables.
- **Galaxy Schema**: Combines multiple star schemas for complex data structures.