# Data Engineering

# What is Data Engineering?

- Designing and building scalable systems for collecting, storing, and analyzing data at scale.
- Building robust ETL/ELT pipelines to transform raw data into usable formats.
- Managing data infrastructure (Cloud/On-prem) and ensuring high availability.
- Implementing data governance, quality checks, and monitoring to ensure reliability.

# Core Aspects of Data Engineering

- Data Pipelines (ETL/ELT):
  - Data engineers build pipelines to Extract, Transform, and Load (ETL) data from various sources into centralized systems like data lakes or data warehouses.
- Infrastructure & Storage:
  - They design and manage scalable, secure, and high-performance databases and cloud-based storage systems (e.g., AWS, GCP, Snowflake).
- Data Quality & Access:
  - A key goal is to ensure data is accurate, reliable, secure, and easily accessible for downstream applications.
- Tools & Technologies:
  - Data engineers use SQL, Python, and distributed computing tools like Apache Spark, Kafka, and Airflow.

# OLTP Vs OLAP

- OLTP (Online Transaction Processing)
- Optimized for transactional atomicity, consistency, and high concurrency.
- Handles high volumes of small, simple queries (Inserts, Updates, Deletes).
- Uses highly normalized schemas (3NF) to ensure data integrity.
- Example: Banking transactions, e-commerce order placement.

- OLAP (Online Analytical Processing)
- Optimized for complex aggregation and historical analysis on large datasets.
- Handles low volumes of compute-heavy read queries.
- Uses denormalized schemas (Star/Snowflake) for faster read performance.
- Example: Quarterly sales reporting, customer churn analysis.

# What is a Data Warehouse?

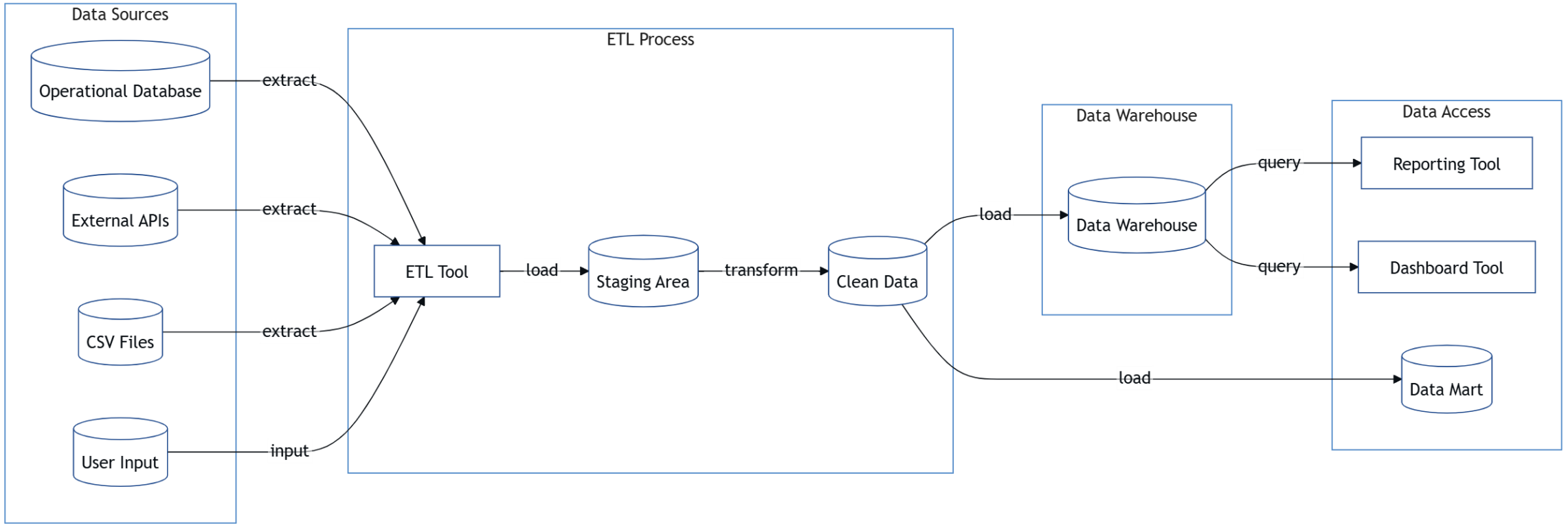
Centralized repository of integrated data

Optimized for analytics and reporting, not transaction processing

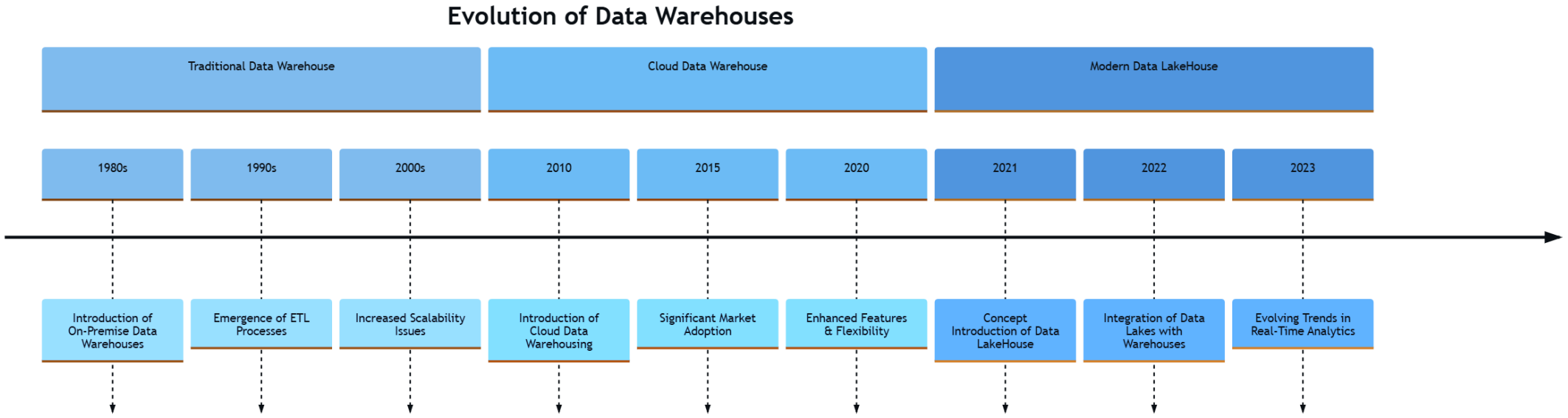
Stores historical and current data

Enables business intelligence, dashboards, and decision-making

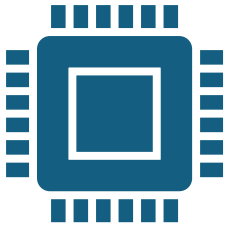
# Components of a Data Warehouse



# Evolution of Data Warehouses



# Evolution of Data Warehouses



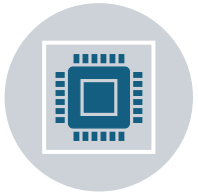
Traditional DW: On-premises, rigid schemas, high hardware costs

Cloud DW: Elastic compute & storage, pay-as-you-go

Modern DW: Lakehouse → Data Lake + Warehouse

# Key Features of Modern Data Warehouses

Scalability & Elasticity – manage petabytes of data



Separation of storage & compute – cost optimization

Schema evolution – adapt to change

Real-time & batch ingestion

Supports AI/ML workloads

# Examples of Modern Data Warehouses

Snowflake – Cloud-native, multi-cloud flexibility

Google BigQuery – Serverless, AI integration

Amazon Redshift – Scalable, AWS-native

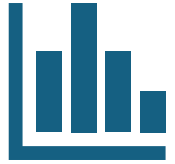
Databricks Lakehouse – Unified Data + AI platform (Spark)

# ETL – Extract, Transform, Load

ETL = Extract, Transform, Load

Moves data from sources → Data Warehouse

Ensures clean, consistent, usable data



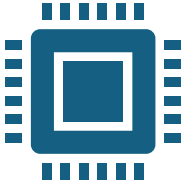
Foundation for reporting & analytics

# ETL Workflow

Data sources → Extraction

Business rules → Transformation

Final storage → Loading



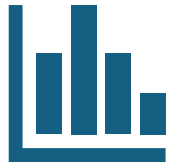
Runs in batches or real-time

# Importance of ETL

Ensures data quality & consistency

Integrates data from multiple systems

Enables faster reporting & insights



Foundation for advanced analytics

# Slowly Changing Dimensions (SCD)

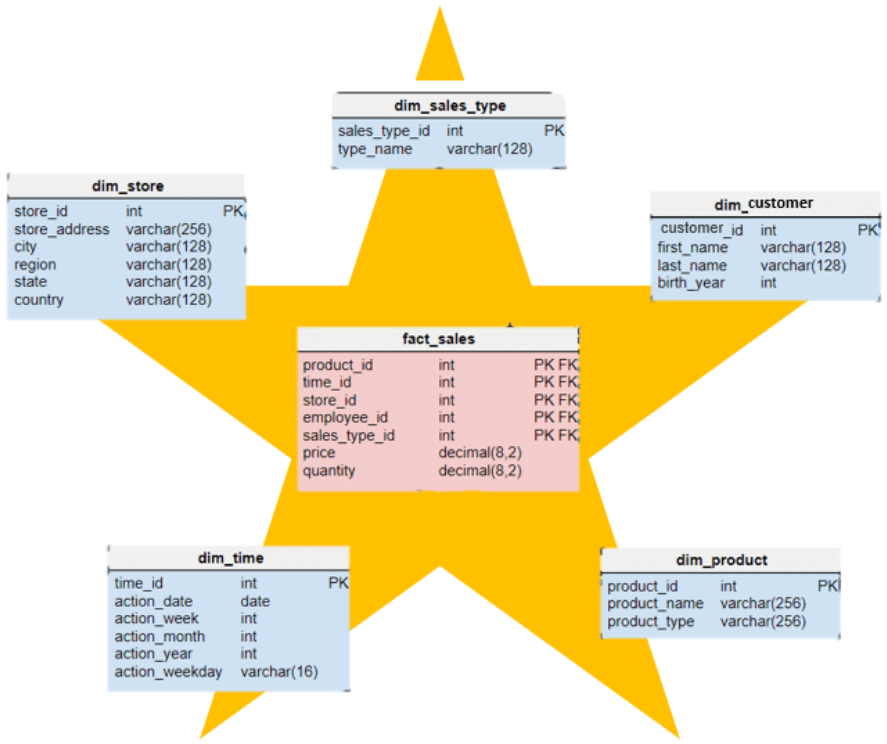
Manage historical changes in dimensional data

Common in customer, product, employee dimensions

Different types: SCD Type 1, 2, 3

Critical for accurate reporting & analytics

# Star Schema




- A data modelling technique, primarily used in data warehousing and business intelligence, that organizes data into a central fact table surrounded by dimension tables.
- Structure resembles a star, hence the name, and is designed to optimize query performance for reporting and analysis.

# Star Schema - Fact Table

- This table contains quantitative data (facts) about a specific business process, such as sales transactions, website visits, or sensor readings. It usually includes foreign keys that link to the dimension tables.
- The fact table is at the centre, and dimension tables are connected to it forming a star-like shape

# Star Schema - Dimensions

- These tables hold descriptive attributes (dimensions) that provide context to the facts, such as product information, customer details, time periods, or geographic locations.
- A star schema consists of a single fact table and multiple dimension tables.

# Benefits of a Star Schema

- The star schema is relatively simple to understand and implement, making it easier for users to navigate and query the data.
- By minimizing joins (typically only joining with dimension tables) and organizing data in a predictable manner, star schemas can significantly improve query performance for reporting and analysis.
- The structure allows for faster retrieval of data for business intelligence and reporting purposes.

# Snowflake Schema

- A schema serves as a logical grouping of database objects, such as tables, views, and stages, within a database.
- The snowflake schema consists of one fact table that is connected to many dimension tables, which can be connected to other dimension tables through a many-to-one relationship
- Tables in a snowflake schema are usually normalized to the third normal form.

# Star Schema Vs Snowflake Schema

| Feature | Star Schema | Snowflake Schema |
|---|---|---|
| Design | Simple, denormalized | Complex, normalized |
| Storage | More redundancy | Less redundancy |
| Query Performance | Fast (fewer joins) | Slower (more joins) |
| Ease of Use | Simple | Complex |
| Maintenance | More updates needed | Easier integrity maintenance |

# Data Warehouse Vs Data Lake Vs DataLakehouse

| Feature | Data Lake | Data Warehouse | Data Lakehouse |
|---|---|---|---|
| **Data Type** | Raw, unstructured & semi-structured | Structured & processed data | Both structured + unstructured data |
| **Schema** | Schema-on-read (flexible) | Schema-on-write (rigid) | Hybrid: schema-on-read + schema-on-write |
| **Use Case** | Data science, ML, big data analytics | Business intelligence, reporting | Unified analytics (BI + AI/ML) |
| **Cost** | Low-cost storage, high processing cost | Higher cost due to optimized storage | Balanced cost with flexibility |
| **Performance** | Slower for queries | Fast queries (optimized) | Near real-time, optimized for both |

# Features of a Data Lakehouse

- **Transaction support:** Support for ACID transactions ensures consistency as multiple parties concurrently read or write data, typically using SQL.
- **Schema enforcement:** The Lakehouse supports Data Warehouse schema architectures such as star/snowflake-schemas
- **BI support:** Lakehouses enable using BI tools directly on the source data.
- **Support for diverse data types ranging from unstructured to structured data:** The lakehouse can be used to store, refine, analyze, and access data types needed for many new data applications, including images, video, audio, semi-structured data, and text
- **Support for diverse workloads:** Data Science, Machine Learning, and SQL and analytics.

# Designing a Data Warehouse

**Inmon Approach:**

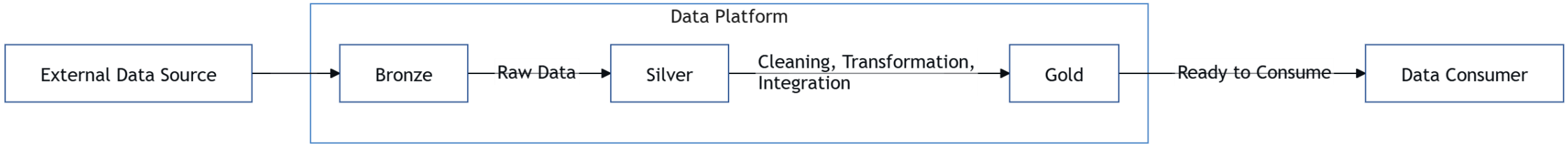
- A data warehousing methodology that prioritizes building a centralized, enterprise-wide data warehouse before creating departmental data marts.
- Focuses on creating a single, integrated, and normalized data model for the entire organization, ensuring data consistency and accuracy across all departments
- Data marts are then derived from this central warehouse to support specific business needs.

# Designing a Data Warehouse

**Medallion Architecture**

- Organizes data in a data lakehouse into three distinct layers: Bronze, Silver, and Gold.
- Each layer represents a progressive increase in data quality and usability, transforming raw data into high-value, analytics-ready datasets.
- Data is progressively refined and validated, leading to higher-quality datasets for analysis and decision-making.

# Medallion Architecture



# Medallion architecture

- Medallion architecture is a data design pattern used to organize data logically.
- Its goal is to incrementally and progressively improve the structure and quality of data as it flows through each layer of the architecture (from Bronze ⇒ Silver ⇒ Gold layer tables).
- Medallion architectures are sometimes also referred to as multi-hop architectures.
- By progressing data through these layers, organizations can incrementally improve data quality and reliability, making it more suitable for business intelligence and machine learning applications.

# Medallion Architecture – Bronze Layer

- This is the initial landing zone for all raw, untransformed data from various sources (databases, APIs, streaming services, etc.).
- Data is stored in its original format (CSV, JSON, Parquet, etc.) without modification or validation.
- The Bronze layer serves as a historical archive, providing a source of truth for auditing, compliance, and reprocessing.

# Medallion Architecture – Silver Layer

- Data from the Bronze layer is cleansed, validated, and transformed into a more structured and usable format.
- This involves deduplication, handling of null values, schema enforcement, and basic transformations to improve data quality.
- The Silver layer aims to provide an "Enterprise view" of key business entities and transactions.

# Medallion Architecture – Gold Layer

- Final layer contains highly refined, aggregated, and enriched data ready for consumption by business users, analysts, and machine learning models.
- Data is often denormalized and optimized for reporting and analytical queries, often using star schema designs.
- Designed to support specific business needs, such as dashboards, reports, and advanced analytics.

# Medallion Architecture

| Question | Bronze | Silver | Gold |
| --- | --- | --- | --- |
| What happens in this layer? | Raw data ingestion | Data cleaning and validation | Dimensional modeling and aggregation |
| Who is the intended user? | • Data engineers<br>• Data operations<br>• Compliance and audit teams | • Data engineers<br>• Data analysts (use the Silver layer for a more refined dataset that still retains detailed information necessary for in-depth analysis)<br>• Data scientists (build models and perform advanced analytics) | • Business analysts and BI developers<br>• Data scientists and machine learning (ML) engineers<br>• Executives and decision makers<br>• Operational teams |

# What is a Data Lake?

- A data lake is a centralized repository that stores vast amounts of raw data in its native format
- Data lakes can accommodate structured, semi-structured, and unstructured data.
- Suitable for big data analytics, machine learning, and other advanced analytical workloads

# Characteristics of a Data Lake

- **Centralized Repository:** Data is stored in a single location, making it easier to access and manage.
- **Native Format:** Data is stored in its original format, without the need for pre-processing or transformation.
- **Scalability:** Data lakes can easily scale to accommodate large volumes of data.
- **Flexibility:** They can handle diverse data types, including structured, semi-structured, and unstructured data.
- **Cost-effectiveness:** Cloud-based data lakes often offer cost-effective storage solutions for big data.
- **Support for advanced analytics:** Data lakes are designed to support a wide range of analytics workloads, including machine learning, AI, and business intelligence.

# Hadoop Overview

- Open-source framework for big data processing
- Stores and processes massive datasets
- Built on distributed computing
- Core modules: HDFS, MapReduce, YARN

# Hadoop Overview

- Open-source framework for big data processing
- Stores and processes massive datasets
- Built on distributed computing
- Core modules: HDFS, MapReduce, YARN

# Advantages of Hadoop

- Scalability – handles petabytes of data
- Fault tolerance – data replication ensures reliability
- Cost efficiency – uses commodity hardware
- Flexibility – processes structured & unstructured data

# Hadoop as a Data Lake

- Hadoop can serve as the foundational technology for building a data lake.
- Hadoop (HDFS) forms the robust and scalable storage backbone required for a functional and efficient data lake

# Apache Spark

- Apache Spark is a multi-language engine for executing data engineering, data science, and machine learning on single-node machines or clusters.
- Apache Spark is built on an advanced distributed SQL engine for large-scale data
- Spark SQL works on structured tables and unstructured data such as JSON or images
- Support for ANSI SQL: Use the same SQL you are already comfortable  with
- Spark SQL adapts the execution plan at runtime, such as automatically setting the number of reducers and join algorithms

# Data Lake vs. Data Warehouse vs. Lakehouse

## Data Warehouse

- Structured data only with strict Schema-on-write enforcement.
- Coupled compute and storage, often leading to high scaling costs.
- Best for: Standard Business Intelligence (BI) and reporting.

## Data Lake

- Stores raw, semi-structured, and unstructured data (Schema-on-read).
- Decoupled low-cost object storage (e.g., S3, ADLS) but lacks ACID transactions.
- Best for: Machine Learning, archival, and big data exploration.

## Data Lakehouse

- Hybrid architecture adding a metadata layer (e.g., Delta Lake, Iceberg) over object storage.
- Combines low-cost storage of a Lake with ACID management of a Warehouse.
- Best for: Unified BI and AI workloads on a single platform.

# Batch Processing Vs Stream Processing

- Batch Processing
  - Processes bounded datasets in large chunks at scheduled intervals (Daily, Weekly).
  - High latency (hours to days) but optimized for high throughput and compression.
  - Ideal for: Payroll, compliance reporting, and historical trend analysis.

- Streaming Processing
  - Processes unbounded data continuously as events arrive (Event-driven).
  - Low latency (milliseconds to seconds) for real-time decision-making.
  - Ideal for: Fraud detection, real-time monitoring, and stock trading.

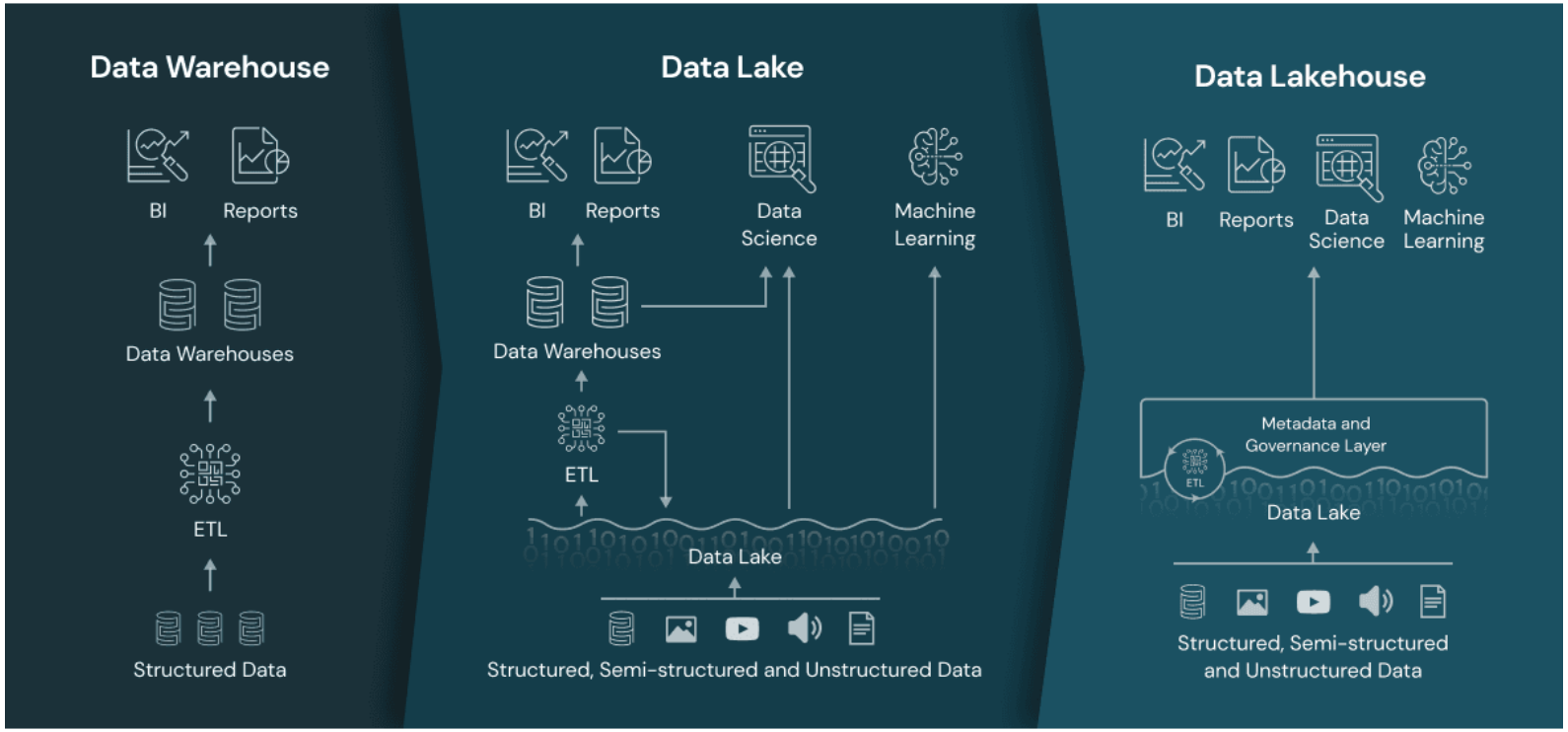
# Why Enterprises Use Lakehouse

- Unified Platform: Supports both SQL analytics (BI) and Machine Learning (AI) on the same data, eliminating silos.
- Cost Efficiency: Leverages low-cost cloud object storage instead of expensive proprietary compute for storage.
- Data Governance: Enforces ACID transactions and schema validation to prevent "Data Swamps."
- Open Standards: Uses open formats (Parquet, Avro) and table formats (Iceberg, Delta) to prevent vendor lock-in.
- Simplified Architecture: Removes the need to maintain separate pipelines for copying data between a Lake and a Warehouse.

# What is a Data LakeHouse?

- A data lakehouse is a data management system that combines the benefits of data lakes and data warehouses.
- A data lakehouse provides scalable storage and processing capabilities for modern organizations that want to avoid isolated systems for processing different workloads, like machine learning (ML) and business intelligence (BI).
- A data lakehouse can help establish a single source of truth, eliminate redundant costs, and ensure data freshness.

# Data Warehouse Vs Data Lake Vs DataLakehouse



# Databricks Lakehouse

- Unified architecture for integration, storage, processing, governance, sharing, analytics and AI.
- Unified approach to how you work with structured and unstructured data.
- Unified end-to-end view of data lineage and provenance. One toolset for Python and SQL, notebooks and IDEs, batch and streaming, and all major cloud providers.