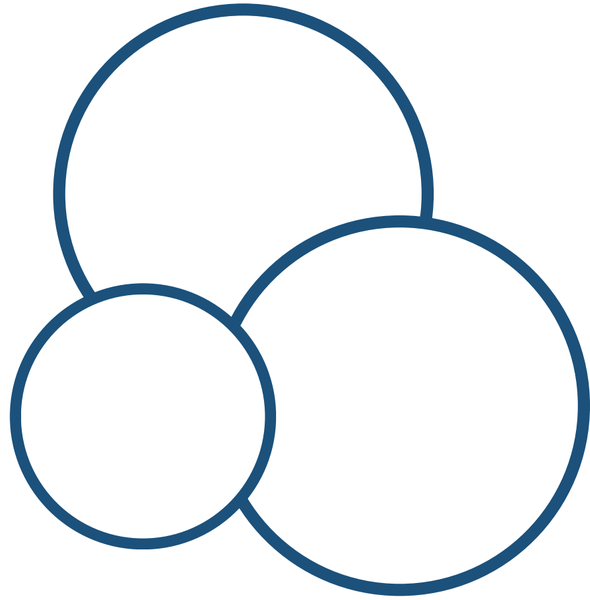
DATATECH LABS.

Data Engineering Course.

Week 3: Data Lakes and Data Warehousing With Modern Data Architectures

Outline: Week 3

- Data Lakes Design and Implementation
  - Understanding data lakes and their benefits
  - Data lake design patterns and best practices
- Data Warehousing and OLAP
  - Data warehouse architectures and design principles
  - Dimensional modeling and star schemas
  - OLAP concepts and cube operations
- Modern Data Architectures: Delta Lake and data lakehouse

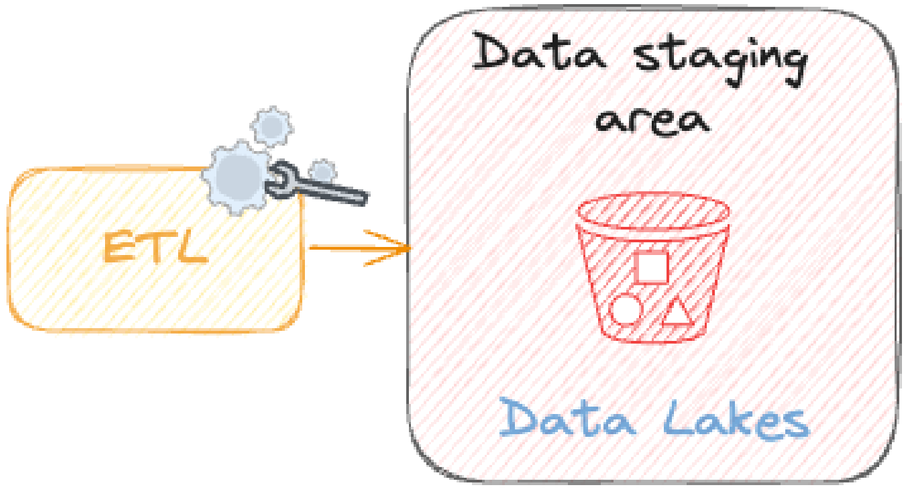


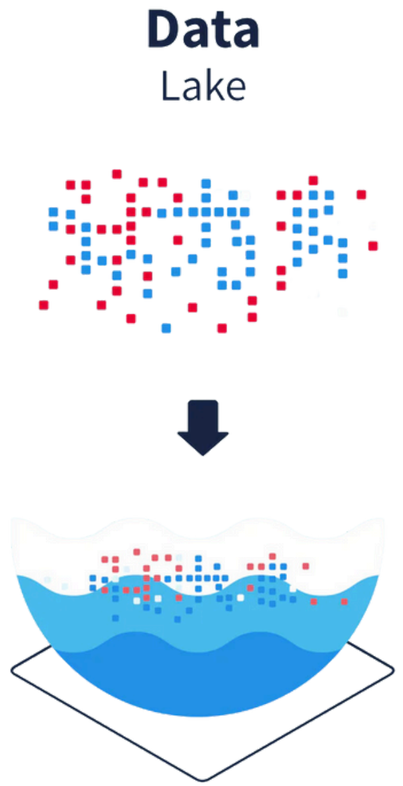
Data Lakes Design and Implementation

Data Lakes

Introduction to Data Lakes

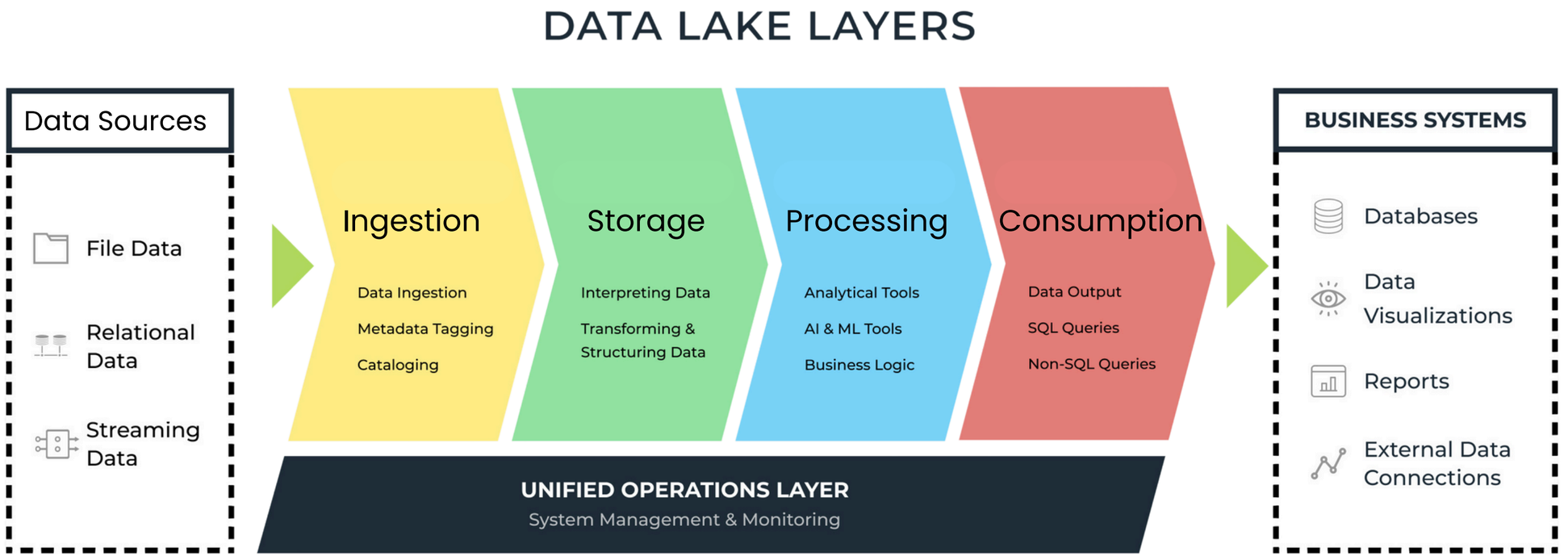
- **Definition**: A centralized repository that allows you to store all your structured and unstructured data at any scale.
- **Purpose**: To store raw data for future processing and analysis.
- Imagine the foldering system on your laptop.





Data Lake Architecture

- Data Sources
- Ingestion Layer
- Storage Layer
- Processing Layer
- Consumption (Data access) Layer



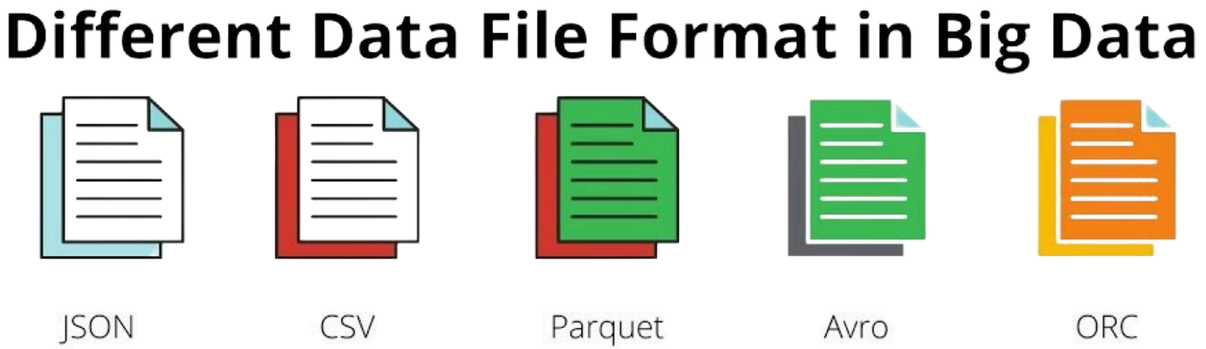
Data Lake Storage Technologies

- Hadoop Distributed File System (HDFS)
- Cloud object storage (e.g., Amazon S3, Azure Blob Storage, Google Cloud Storage)
- Delta Lake
- Apache Hudi
- Apache Iceberg

Data Ingestion in Data Lakes

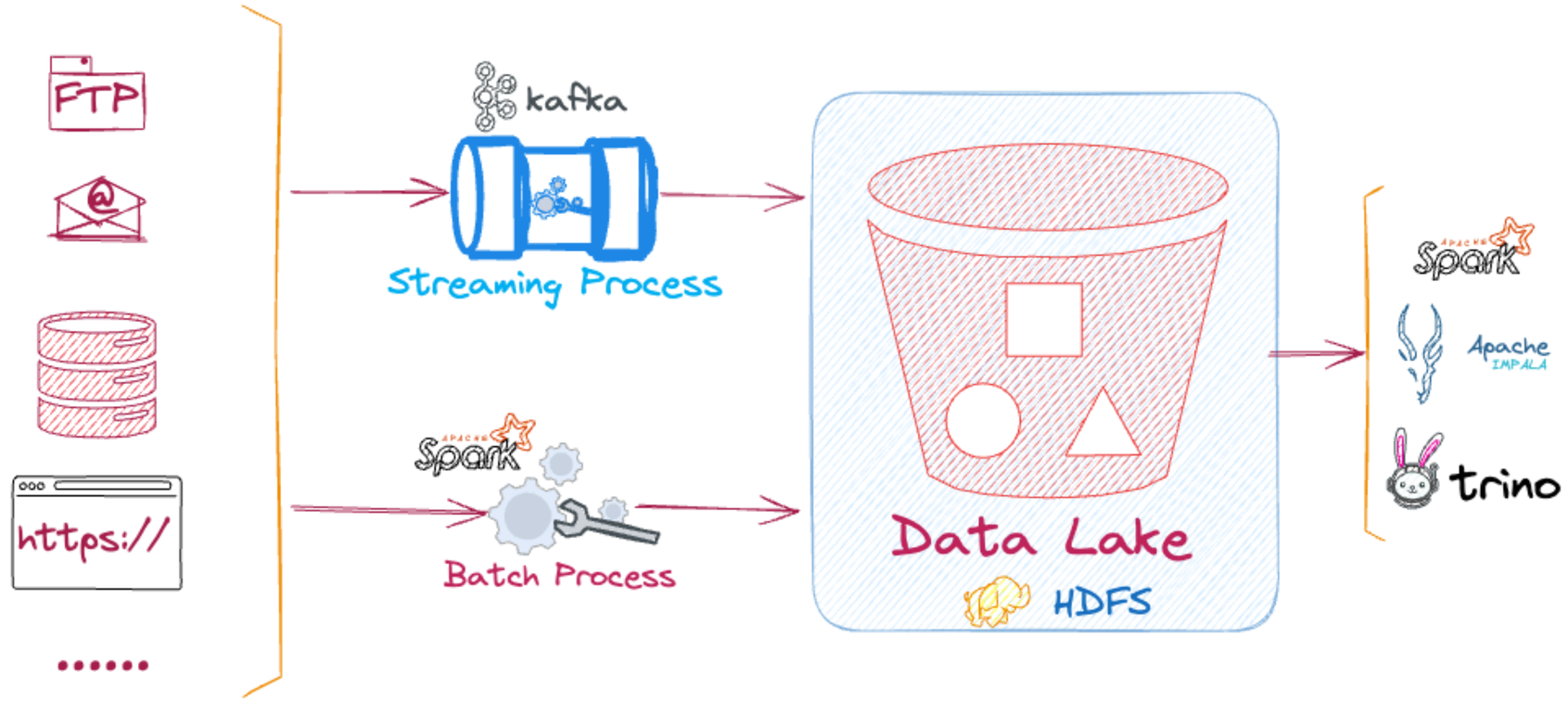
- Batch ingestion
- Stream ingestion
- Change Data Capture (CDC)
- Data migration tools

- Data Formats in Data Lakes:
  - CSV
  - JSON
  - Avro
  - Parquet
  - ORC (Optimized Row Columnar)



Data Processing in Data Lakes

- Batch processing (e.g., Hadoop MapReduce, Spark)
- Stream processing (e.g., Spark Streaming, Flink)
- Interactive queries (e.g., Presto, Impala)
- Machine learning pipelines



Data Governance in Data Lakes

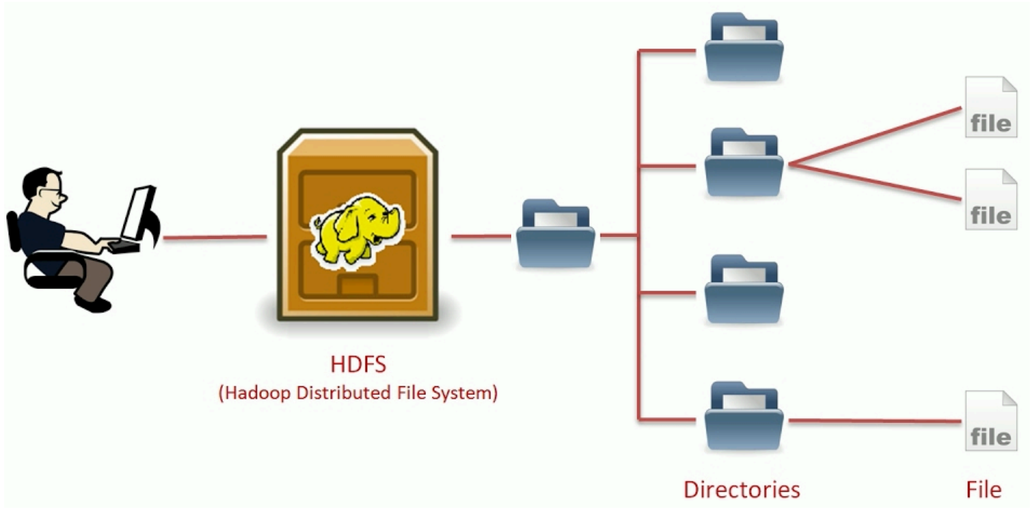
- Metadata management
- Data lineage
- Access control and security
- Data quality management
- Data lifecycle management

Challenges in Data Lake Implementation

- Data swamp prevention
- Schema evolution
- Data quality issues
- Performance optimization
- Cost management

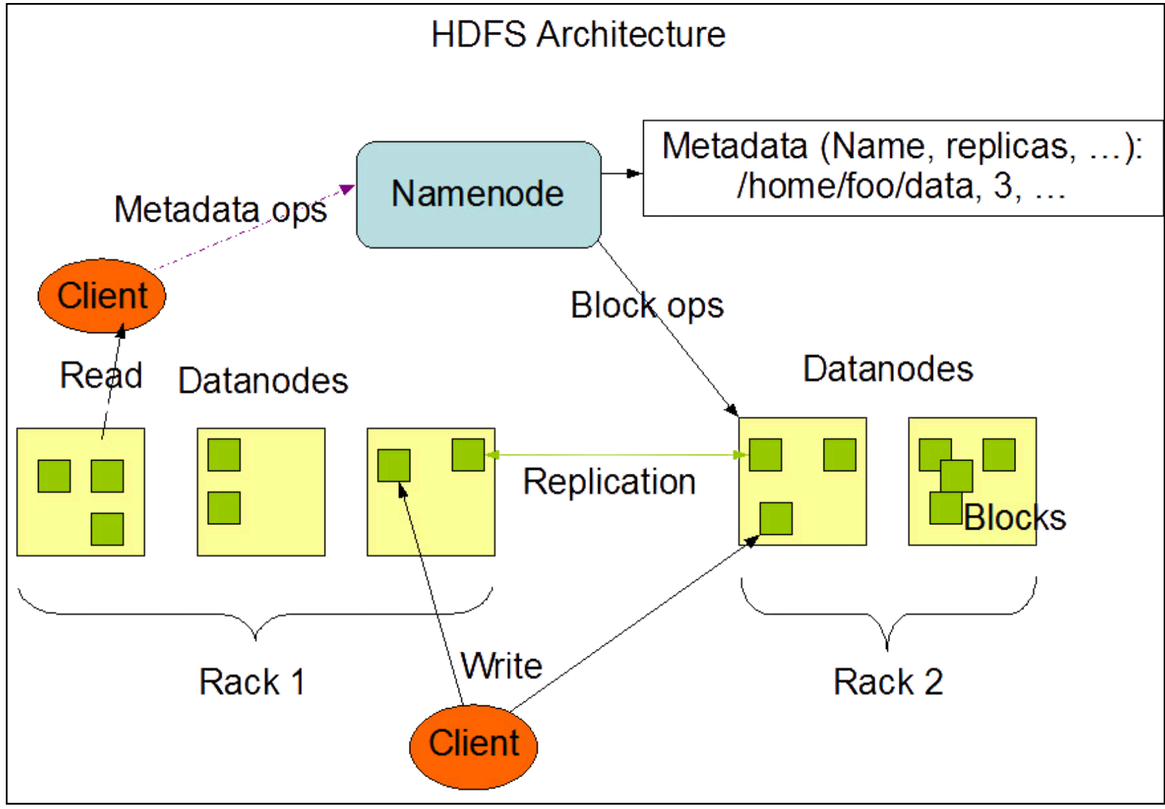
HDFS Data Lake

- Overview:
  - Stands for “Hadoop Distributed File System”.
  - It is designed to store large volumes of data across many machines and to provide high-throughput access to this data.
- Key Features:
  - **Scalability**: HDFS can scale to accommodate petabytes of data across thousands of machines.
  - **Fault Tolerance**: Data is replicated across multiple nodes, ensuring data availability even in case of hardware failures.
  - **High Throughput**: Optimized for large data sets, HDFS provides high-throughput access for big data applications.



HDFS Architecture and Functionality

- Architecture:
  - **NameNode**: Manages metadata and regulates access to files.
  - **DataNodes**: Store the actual data blocks and perform read/write operations as directed by the NameNode.
- Data Storage:
  - **Block Storage**: Data is split into large blocks (**default 128MB**) and distributed across DataNodes.
  - **Replication**: Each block is replicated across multiple DataNodes (default is 3 replicas) to ensure fault tolerance.

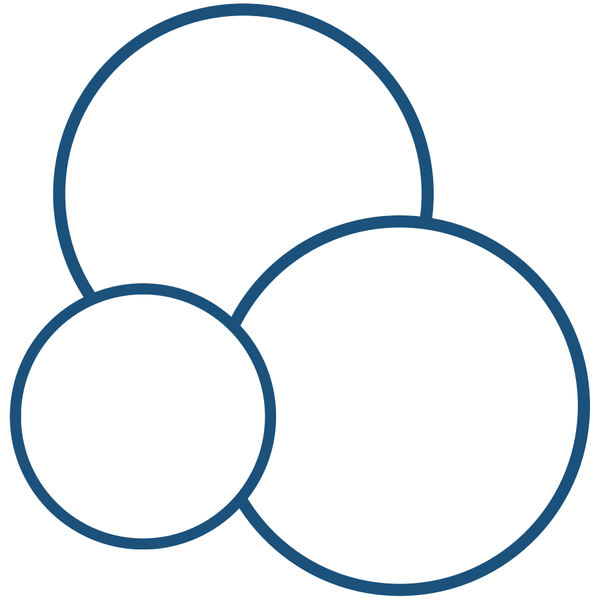


HDFS Architecture and Functionality

- Data Access:
  - **Write Once, Read Many**: Files are written once and read multiple times, optimizing for read-heavy operations typical in big data processing.
  - Streaming Data Access: HDFS is optimized for sequential read/write access rather than random access.

Hands-on:

Data Lake: HDFS

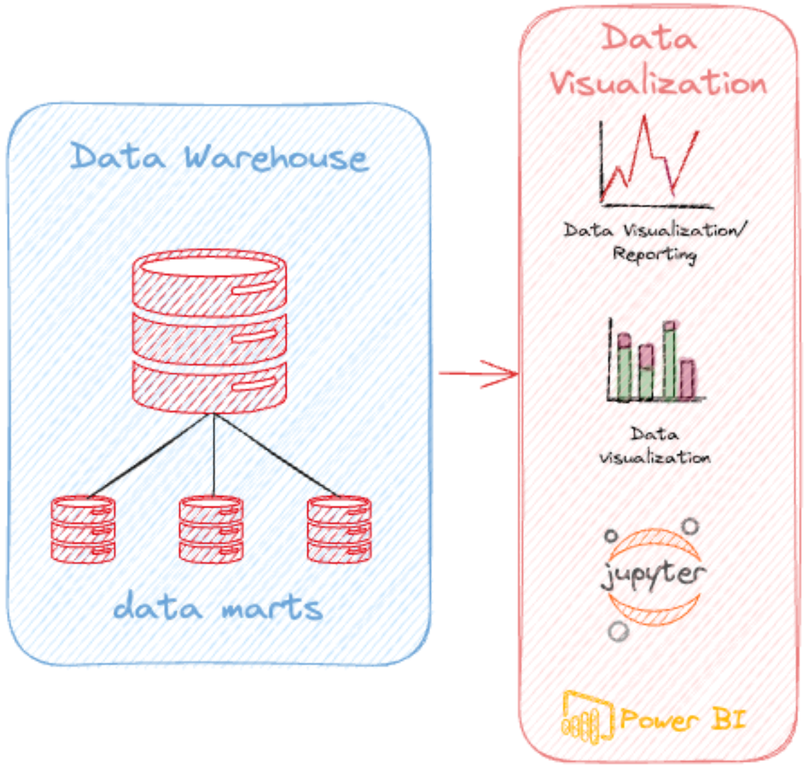


Data Warehousing and Modern Data Architectures

Data Warehousing

Introduction to Data Warehousing

- Definition: A system used for reporting and data analysis, considered a core component of business intelligence
- Purpose: To integrate data from multiple sources for complex querying and analysis

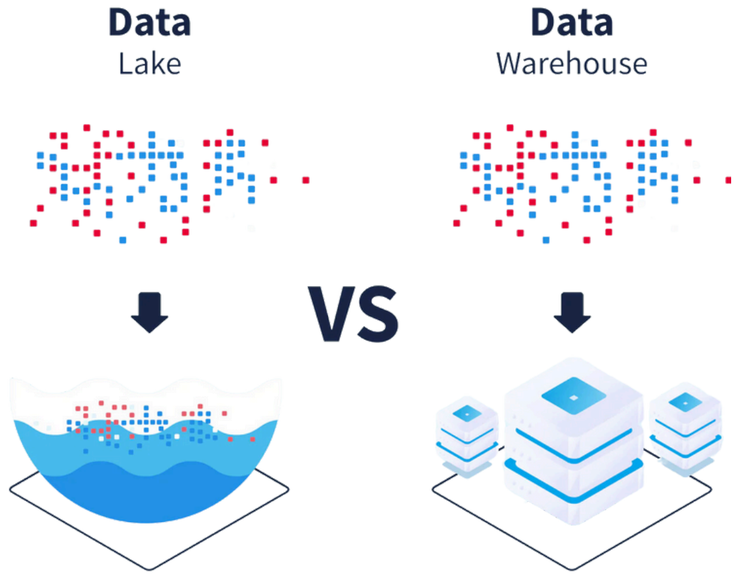


Data Warehouses vs. Transactional Databases

Aspect	Data Warehouses	Transactional Databases
Purpose	Analysis and reporting	Day-to-day transactions
Data	Historical, integrated	Current, detailed
Query type	Complex, ad-hoc	Simple, predefined
Users	Analysts, executives	Clerks, managers
Size	Large (TBs to PBs)	Small to medium

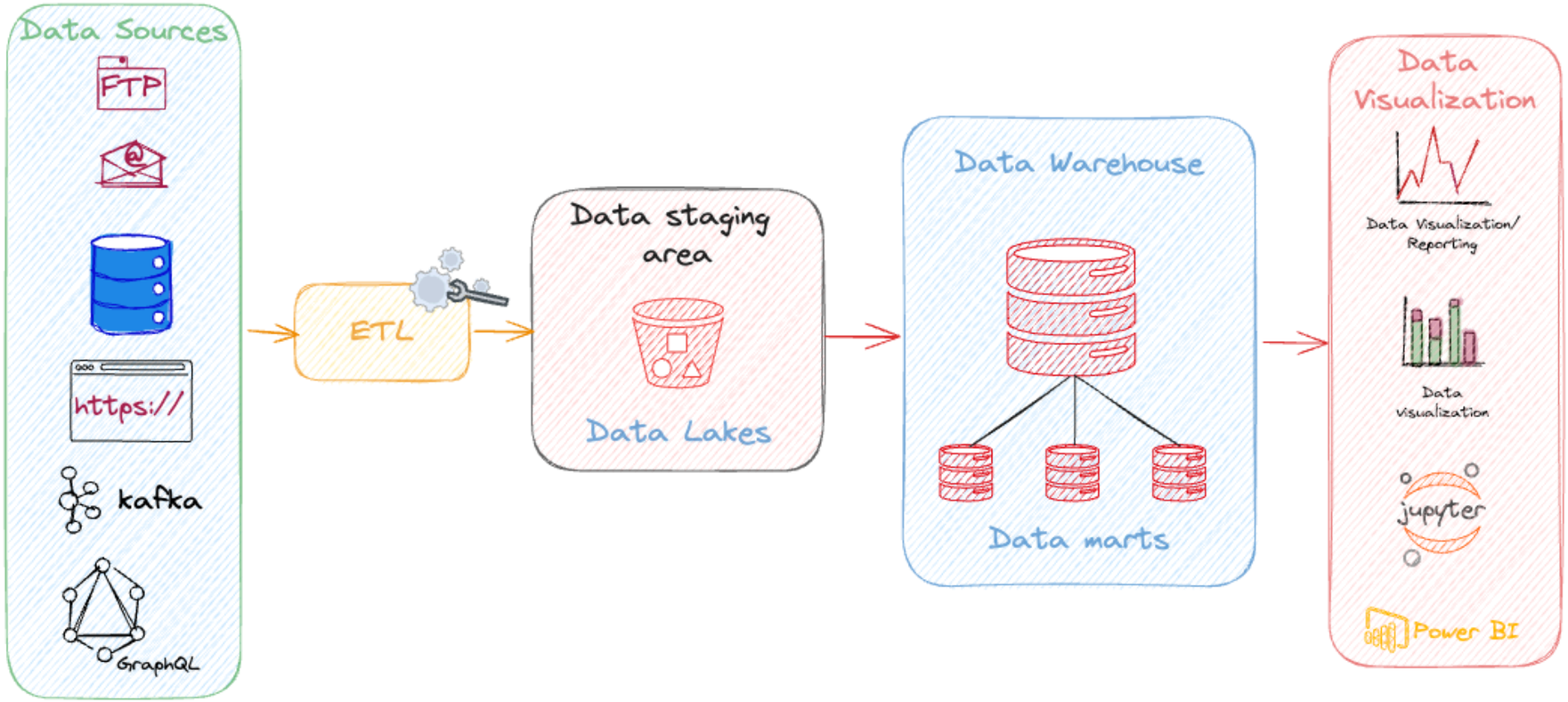
Data Warehouses vs. Data Lakes

Aspect	Data Warehouse	Data Lake
Data structure	Processed, structured	Raw, unstructured/semi-structured
Schema	Schema-on-write	Schema-on-read
Users	Business analysts	Data scientists, analysts
Agility	Less flexible	Highly flexible
Processing	ETL (Extract, Transform, Load)	ELT (Extract, Load, Transform)



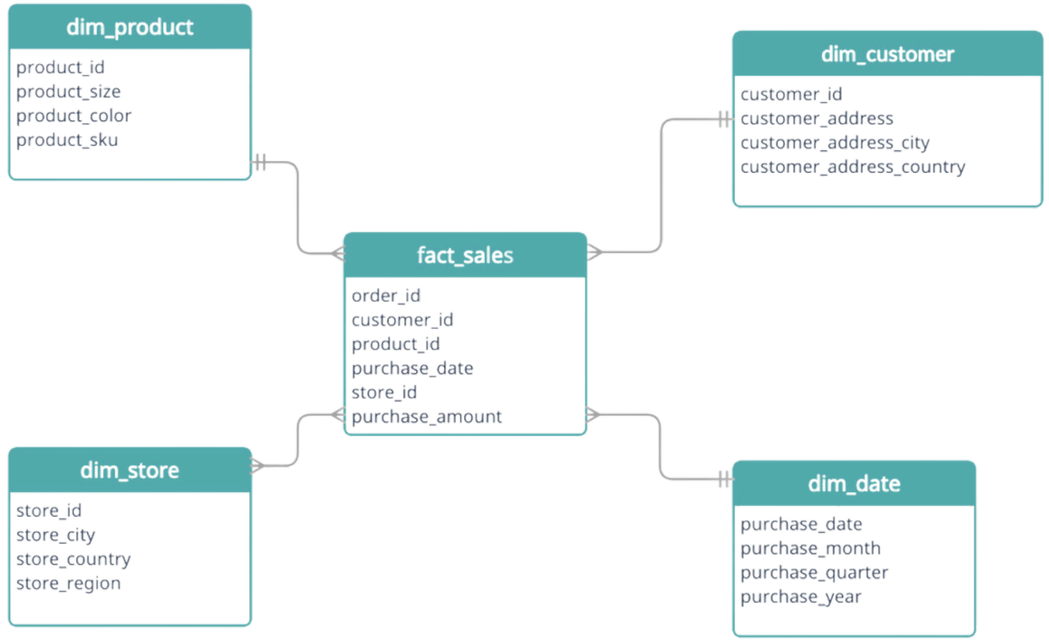
Data Warehouse Architecture

1. Source systems
2. ETL Processes
3. Data staging area
4. Data warehouse core
5. Data marts
6. Data consumption tools



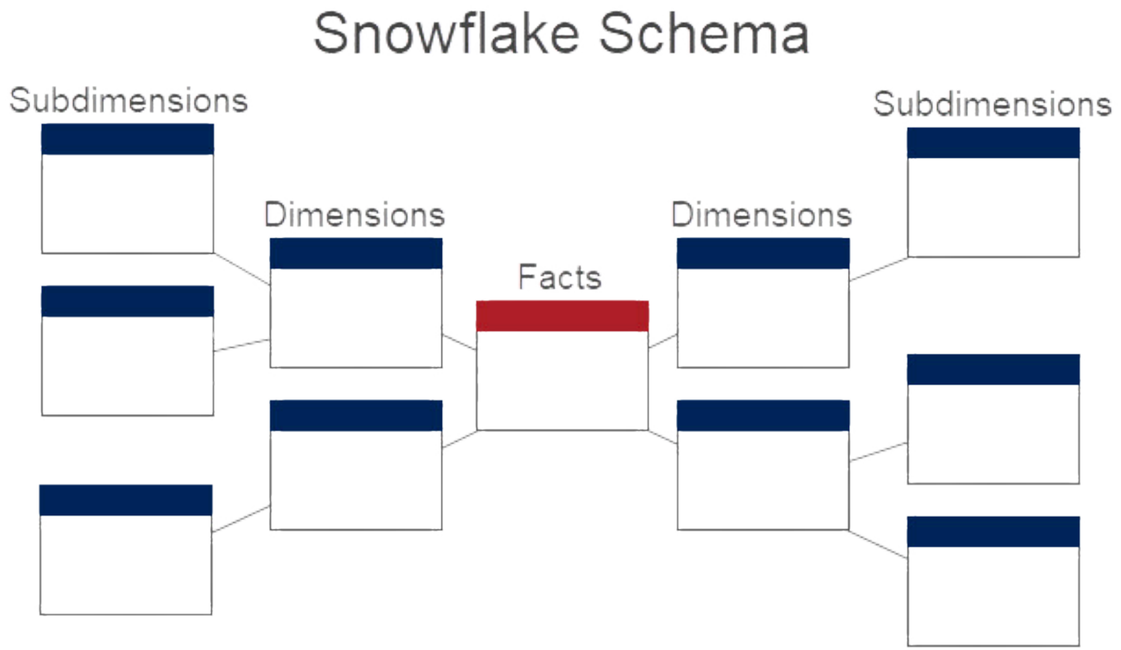
DW Schema Designs: Start Schema

- **Star Schema**: A type of database schema used in data warehousing. It consists of a central fact table connected to multiple dimension tables, resembling a star.
- **Structure**:
  - Fact Table: Central table containing quantitative data (e.g., sales, revenue).
  - Dimension Tables: Surrounding tables with descriptive attributes (e.g., time, product, location).
- **Advantages**:
  - Simpler queries
  - Faster performance for read operations
  - Easy to understand and maintain



DW Schema Designs: Snowflake Schema

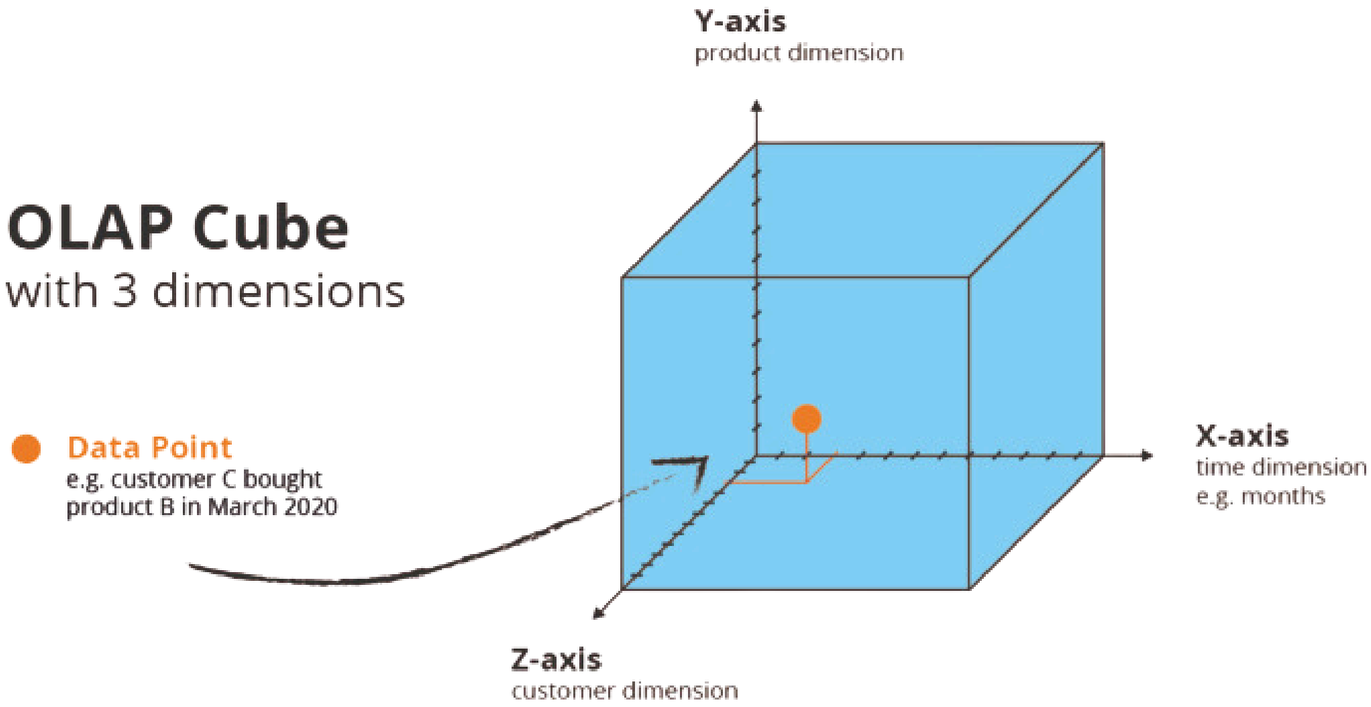
- **Snowflake Schema**: A more normalized version of the star schema where dimension tables are further broken down into related sub-dimension tables.
- **Structure**:
  - Fact Table: Same as in the star schema.
  - Normalized Dimension Tables: Hierarchically organized dimension tables.
- **Advantages**:
  - Reduced data redundancy
  - More organized and efficient storage
- **Considerations**:
  - Slightly more complex queries
  - May have slower query performance compared to a star schema



Intro to OLAP (Online Analytical Processing)

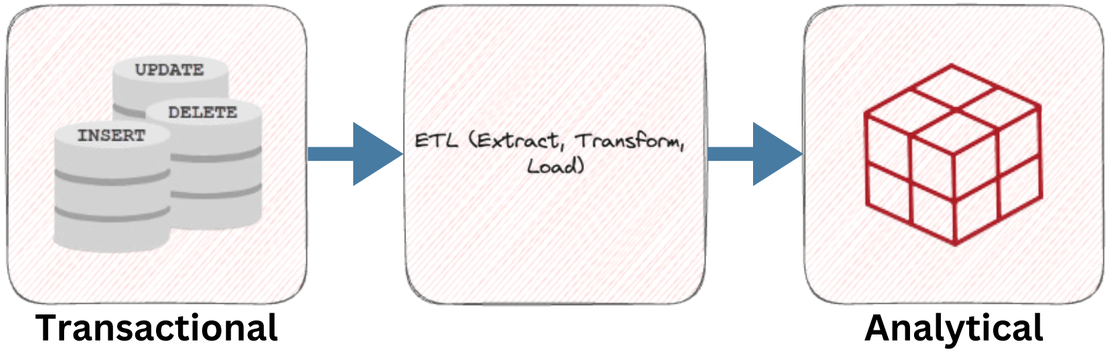
- **Definition**: Technology for analyzing multidimensional data from multiple perspectives
- Key operations: Slice, dice, drill-down, roll-up, pivot

- OLAP Cube:
  - Dimensions
  - Measures
  - Hierarchies



OLTP vs. OLAP

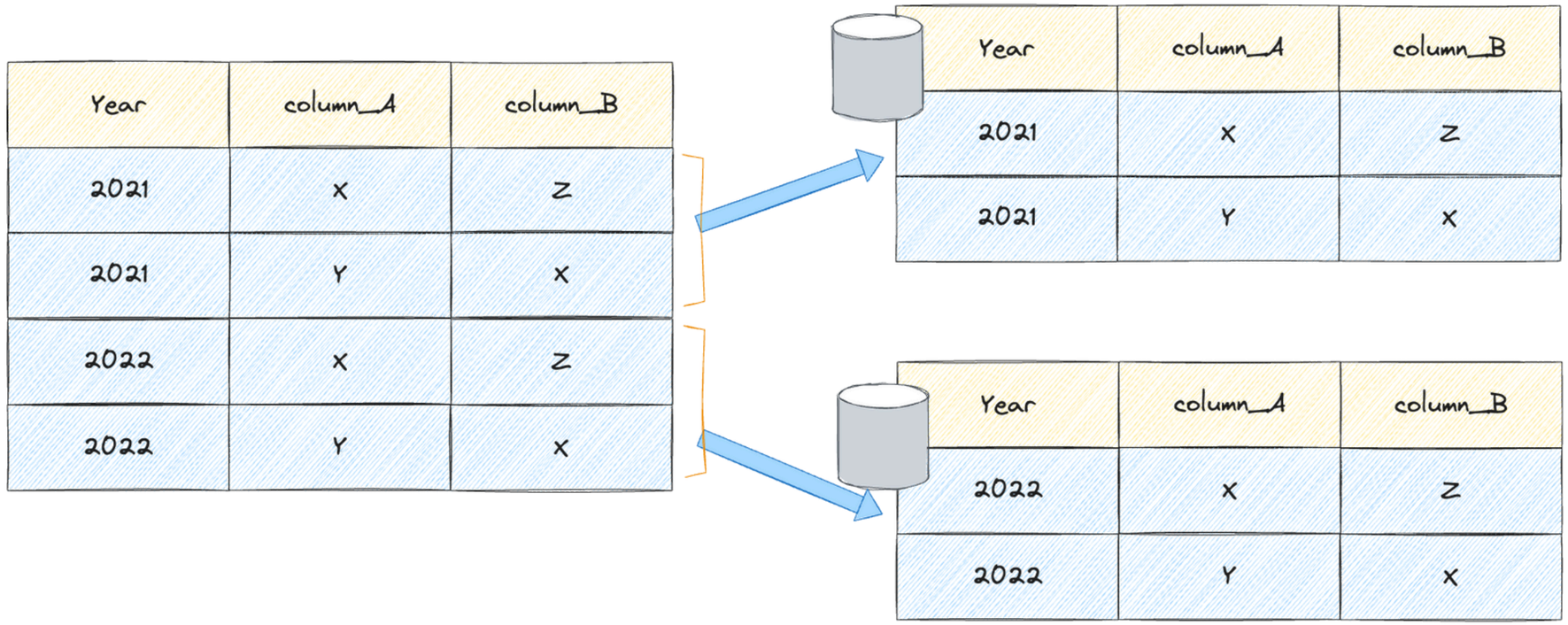
- OLTP (Online Transaction Processing)
- OLAP (Online Analytical Processing)



Aspect	OLAP	OLTP
Purpose	Analysis and reporting	Transaction processing
Data updates	Periodic, batch updates	Continuous updates
Queries	Complex, ad-hoc	Simple, standardized
Data view	Multidimensional	Flat relational
Data volume	Large	Moderate

Data Warehouse Performance Tuning

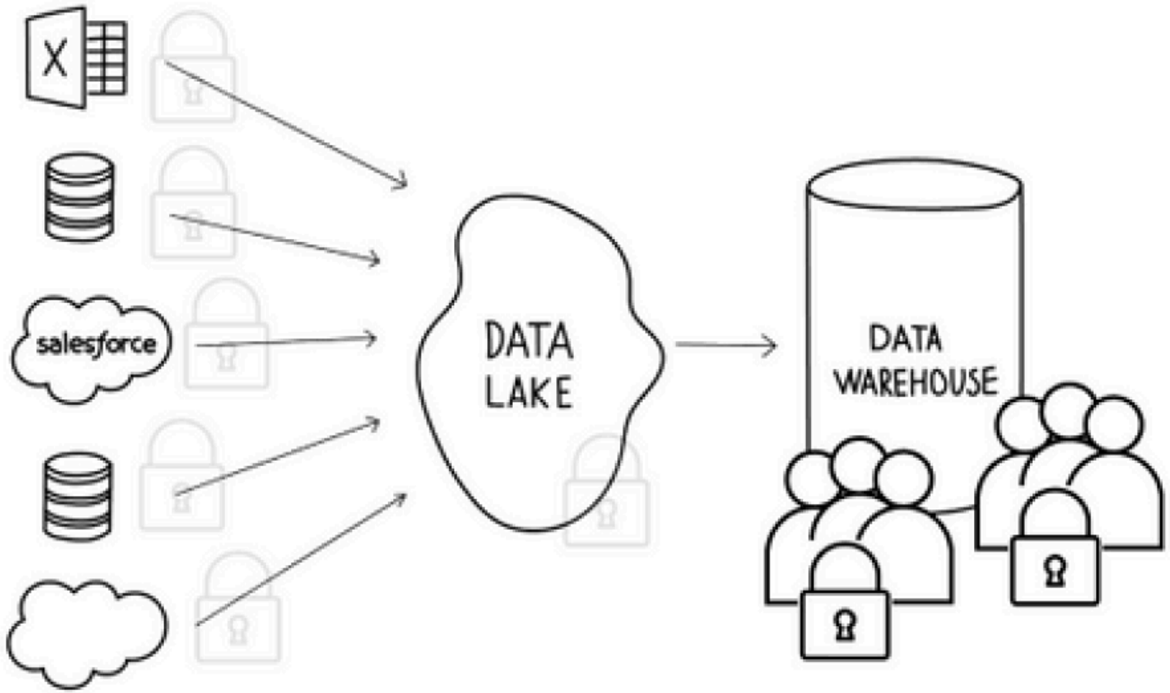
- Indexing strategies
- Partitioning
- Materialized views
- Query optimization



Partitioning

Security in Data Warehousing

- Access control
- Data encryption
- Auditing and monitoring
- Compliance considerations (e.g., GDPR, HIPAA)



Slowly Changing Dimensions (SCDs)

- Type 1: Overwrite
- Type 2: Add new row
- Type 3: Add new attribute
- Type 4: Use history table
- Type 6: Hybrid approach

First Time Load on 08-10-2020

product_id	product_name	unit_price	product_desc	eligible_promotion	Sync
100	Prod1	100.5	Prod desc 1	1	0
101	Prod2	50	Prod desc 2	0	0

product_key	product_id	product_name	unit_price	product_desc	eligible_promotion	start_date	end_date	active_flag
1	100	Prod1	100.5	Prod desc 1	1	08-10-2020		0
2	101	Prod2	50	Prod desc 2	0	08-10-2020		0

Second Time Load on 09-10-2020

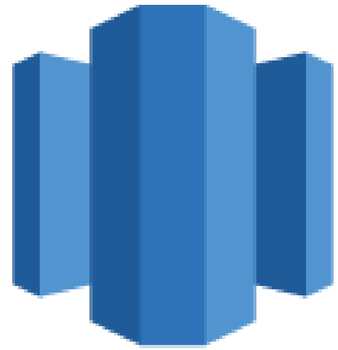
product_id	product_name	unit_price	product_desc	eligible_promotion	Sync
100	Prod1	100.5	Prod desc 1	1	1
101	Prod2	35.5	Prod desc 2	1	0
102	Prod3	40.25	Prod desc 3	0	0

product_key	product_id	product_name	unit_price	product_desc	eligible_promotion	start_date	end_date	active_flag
1	100	Prod1	100.5	Prod desc 1	1	08-10-2020		0
2	101	Prod2	50	Prod desc 2	0	08-10-2020	09-10-2020	1
3	101	Prod2	35.5	Prod desc 2	1	09-10-2020		0
4	102	Prod3	40.25	Prod desc 3	0	09-10-2020		0

Popular Data Warehouse Technologies

- Modern Data Warehouses:
  - Amazon Redshift
  - Google BigQuery
  - Apache Hive
  - Snowflake
  - Azure Synapse Analytics
  - Teradata

- Modern Data Warehouse Trends
  - Cloud-based data warehousing
  - Real-time data warehousing
  - Big data integration
  - Machine learning integration



Amazon
Redshift

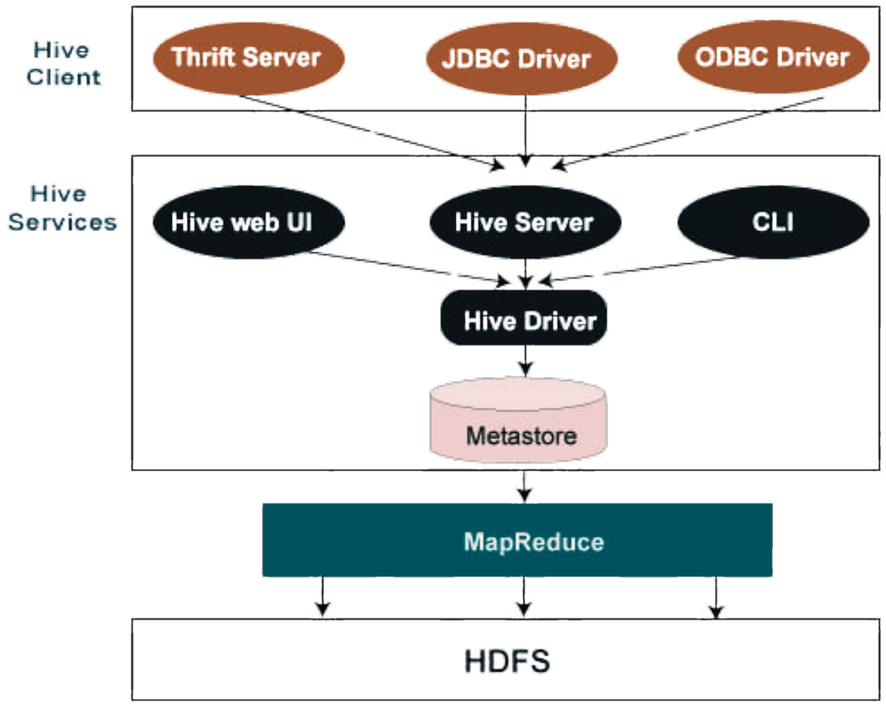
HIVE

Google
Big Query

snowflake®

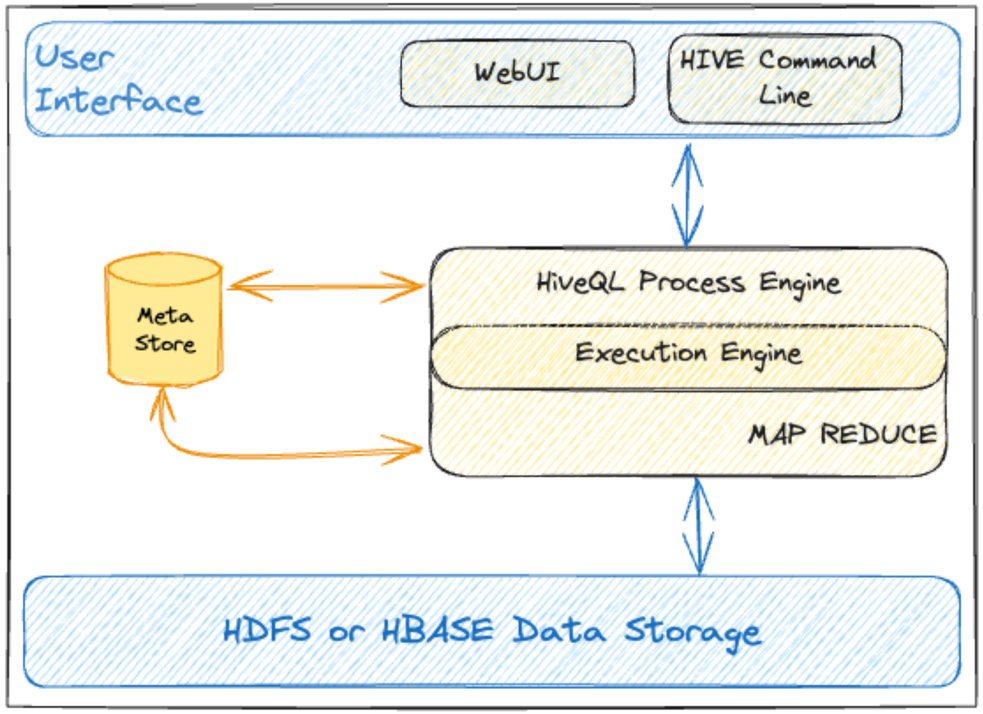
Big Data Warehousing: Apache Hive

- **Overview**:
  - Apache Hive is a data warehousing infrastructure built on top of Hadoop.
  - It provides a query language called HiveQL, similar to SQL, for querying and managing large datasets residing in distributed storage.
- **Key Features**:
  - Data Warehouse Infrastructure: Hive is designed for managing and querying large datasets stored in Hadoop HDFS.
  - HiveQL: A SQL-like language for querying data, making it accessible to users familiar with SQL.
  - Scalability: Supports massive data storage and processing needs.



How Apache Hive Works

- **Architecture**:
  - **MetaStore**: Stores metadata about tables, columns, partitions, and more.
  - **Driver**: Receives queries, compiles them, and manages their execution.
  - **Compiler**: Compiles HiveQL into MapReduce or Tez jobs.
  - **Execution Engine**: Executes the jobs on Hadoop, typically using MapReduce or Tez.
- **Workflow**:
  - User submits a HiveQL query.
  - The query is compiled into a series of MapReduce jobs.
  - Jobs are executed on Hadoop, and results are returned to the user.



Hands-on:

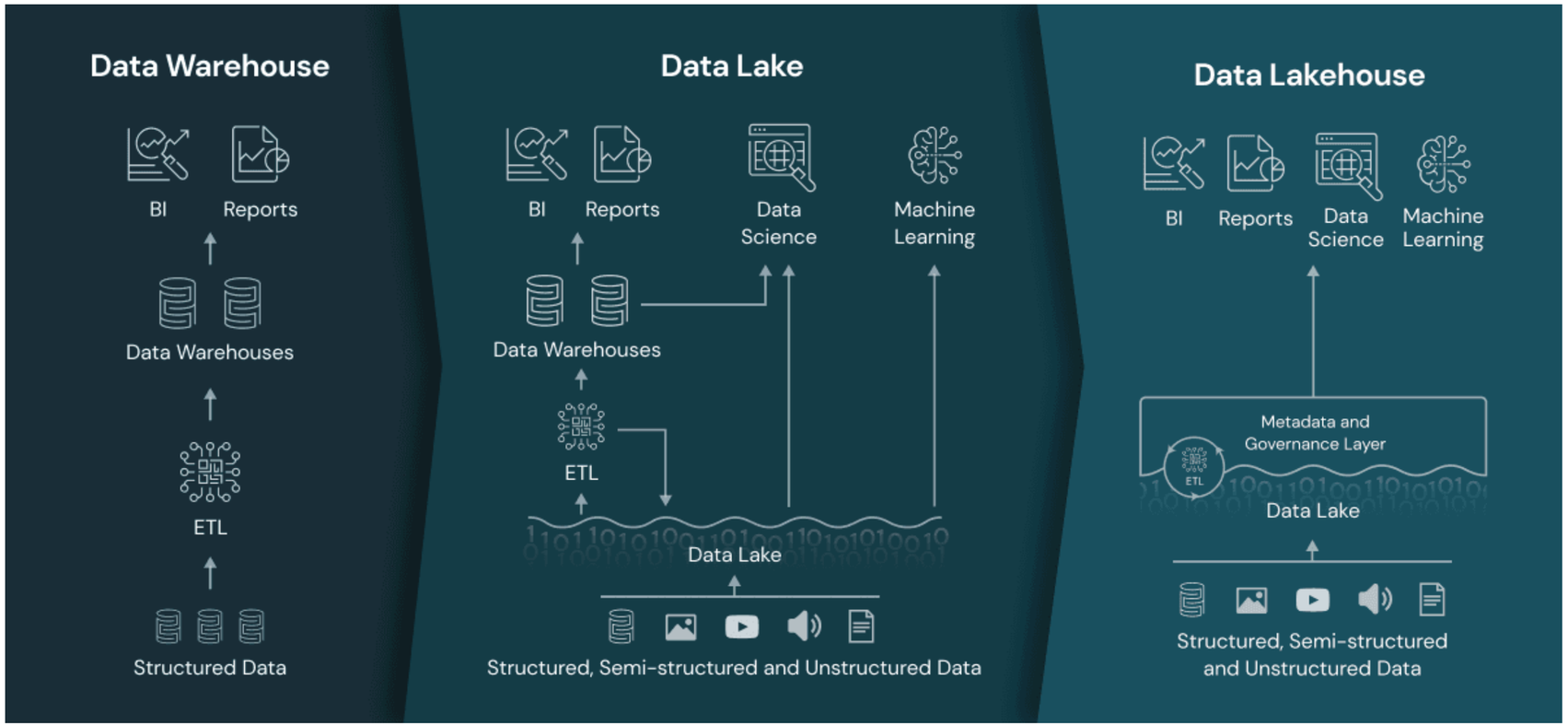
Data Warehousing: Apache Hive

Q&A and Discussion

Modern Data Architectures

Data Lakehouse Concept

- **Definition**: A new data management paradigm that combines the best elements of data lakes and data warehouses
- **Key features**:
  - ACID transactions
  - Schema enforcement
  - BI support
  - Decoupled storage and compute



Delta Lake

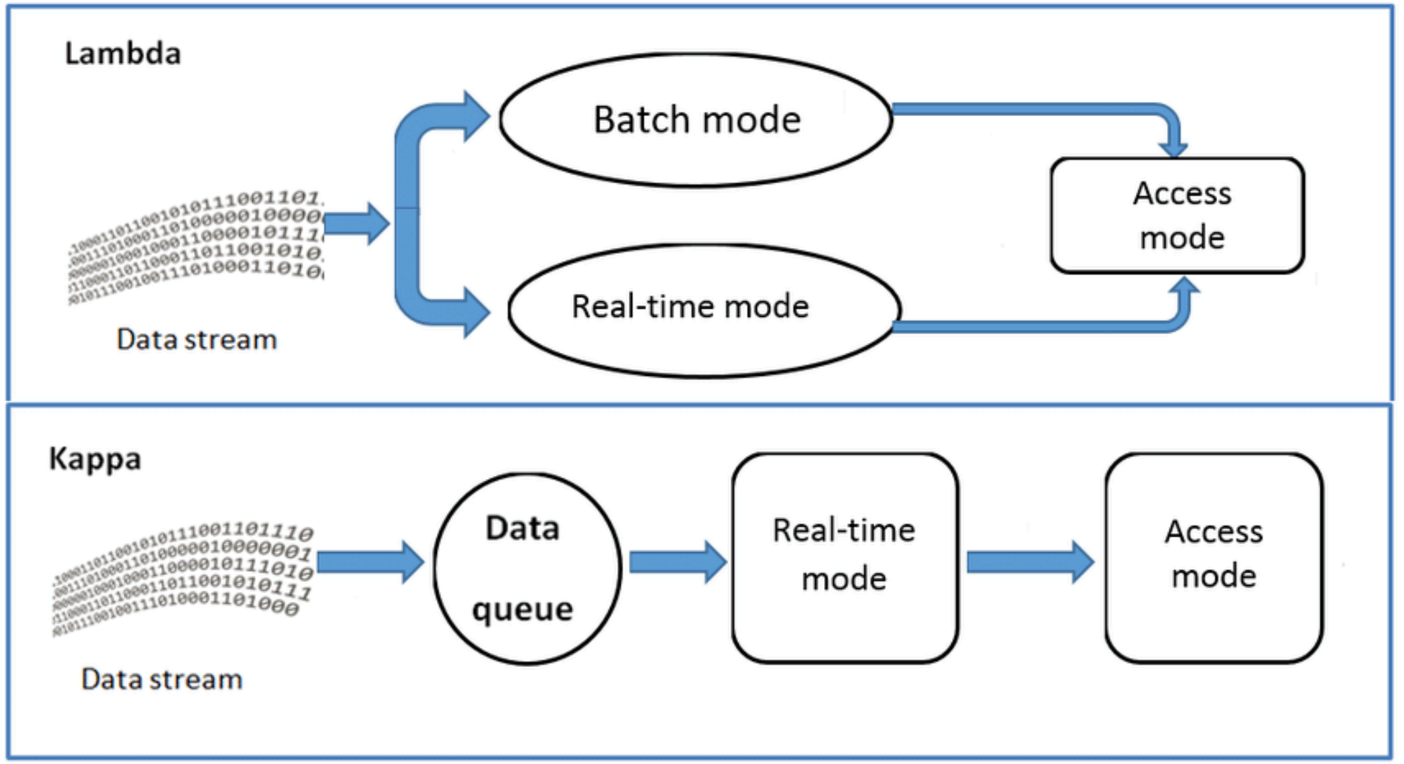
- Open-source storage layer for data lakes that gives the ability for CRUD operations in Data Lakes.
- Key features:
  - ACID transactions
  - Scalable metadata handling
  - Time travel (data versioning)
  - Schema enforcement and evolution
  - Audit history

Modern Data Architecture Patterns

- Lambda Architecture:
  - Batch layer
  - Speed layer
  - Serving layer

- Kappa Architecture:
  - Stream processing as the core
  - Reprocessing capabilities

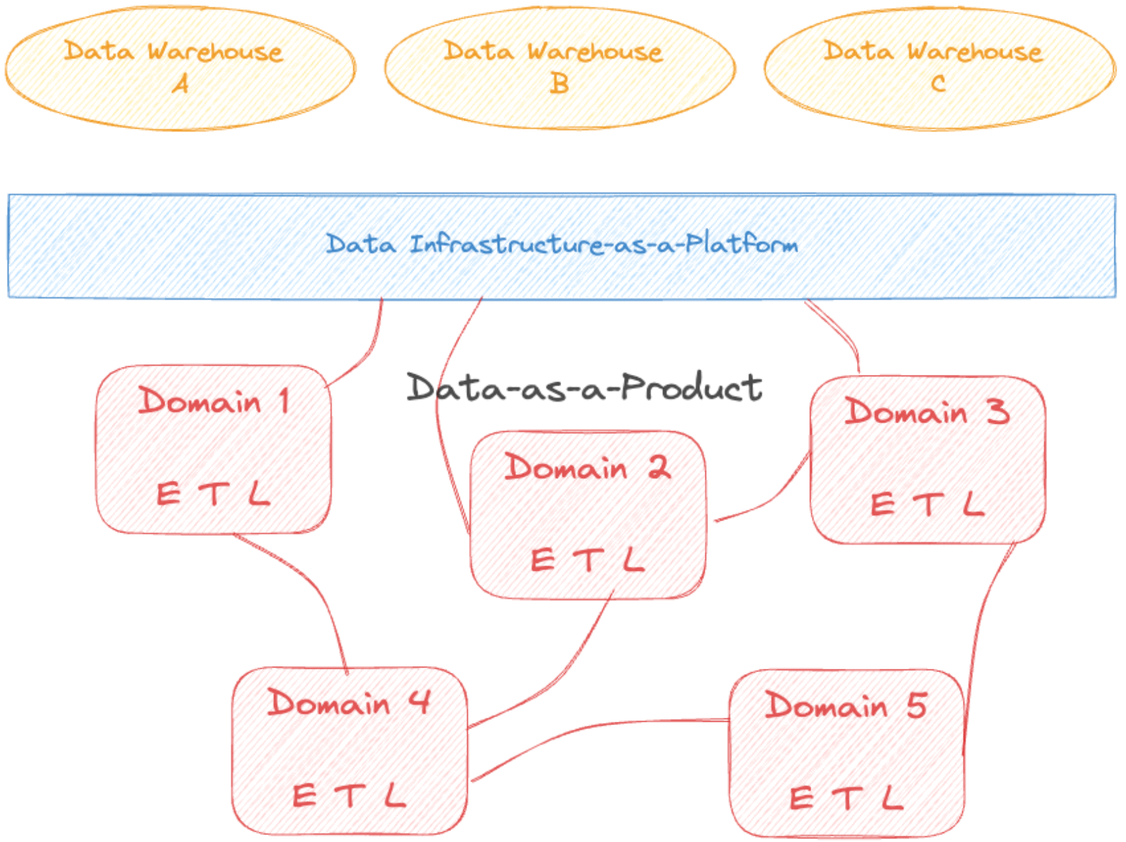
- Unified Architecture (Data Lakehouse)



Data Mesh Architecture

- **Definition**: A decentralized socio-technical approach to data management
- Key principles:
  - Domain-oriented decentralized data ownership
  - Data as a product
  - Self-serve data infrastructure
  - Federated computational governance

- **Discussion**: the trade-offs between centralized (e.g., data lake) and decentralized (e.g., data mesh) approaches to data architecture.



Implementing Data Quality in Modern Architectures

- Data profiling
- Data validation rules
- Anomaly detection
- Data quality scorecards
- Automated data cleansing

Future Trends in Data Architectures

- AI-driven data management
- Edge computing and IoT integration
- Blockchain for data integrity
- Quantum computing for data processing

Q&A and Discussion

MEET OUR TEAM

Omar AlSaghier

Sr. Data Engineer

DATATECH LABS.

THANK YOU

OUR CONTACT

DataTechLabs

datatechlabs.ai@gmail.com

datatechlabs.ai

Amman, Jordan