Data Engineering Lifecycle

- Schema: Defines the hierarchical organization of data
- Schemaless: Application defines the schema as data is written
- Fixed schema: Enforced in the DB to which application writes must conform
- Ingestion:
  - Push: Source system writes data out to a target
  - Pull: Data is retrieved from a source system
- Featurization: Extract and enhance data features useful for training ML models
- BI:
  - Describe a business's past and current state
  - Data is stored in a clean but fairly raw form with minimal postprocessing business logic
- Operational Analytics:
  - Fine-grained details of operations, e.g., live view of inventory
  - Focused on present and doesn't concern historical trends
- Security: Principle of least priviledge
- Data Management: Encompasses the set of best practices DE will use to accomplish the task of managing the data lifecycle technically and strategically
- Data Governance: Ensure quality, integrity, security, and usability of the data collected by an organization
- Metadata:
  - Business: Non-technical questions about who, what, where, and how and provides a DE with the right context and definitions to properly use the data
  - Technical: Describes the data created and used by systems across the DE lifecycle
  - Data Lineage: Tracks the origin and changes to data
  - Schema: Describes structure of data stored in a system
  - Operational: Used to determine whether a process succeeded or failed and the data involved in the process
- Data Accountability: Assigning an individual to govern a portion of data
- Data Quality:
  - Quality tests, ensuring data conformance to schema expectations, data completions, and precision
  - Accuracy: Is the data factuall correct? Duplicated values? Are the numeric values accurate?
  - Completeness: Are the records complete? Do all required fields contain valid values?
- Master Data: Business entities (employees, customers etc.)
- Master Data Management: Practice of building consistent entity definitions known as golden records
- Data Modeling and Design: Process for converting data into usable form
- Data Lineage:
  - Provides a trail of data's evolution as it moves through various systems and workflows
  - Tracks both the systems that process the data and the upstream data it depends on
- Data Integration: Process of integrating data across tools and processes. Happens through general-purpose APIs rather than DB connections
- Orchestration: Coordinating many jobs to run as quickly and efficiently as possible on a scheduled cadence (DAGs)

Designing Data Architecture

Architecture Concepts

- Domain: Real-world subject area, can contain multiple services
- Service: Set of functionality whose goal is accomplish a task, each service has particular tasks
- Scalability: Increase capacity of a system to improve performance
- Elasticity: Ability of a scalable system to scale dynamically
- Availability: Percentage of time an IT service or component is in an operable state
- Reliability: System's probability of meeting defined standards in performing
- Horizontal scaling: Add more machines to satisfy load and resource requirements where a leader node distributes tasks to worker nodes → Increase availability and reliability
- Vertical scaling: Increase CPU, memory etc. → Cannot offer high availability and reliability
- Architecture tiers:
  - Single tier: Database and applications are tightly coupled, residing on a single server. Good for testing but not for production
  - Multi tier: Composed of multiple layers (data, applications, business logic etc.), that are bottom-up and hierarchical
  - Three tier: Consists of data, application logic, and presentation tiers
  - Monoliths: Consists of a single codebase running on a single machine that provides both the application logic and user interface
  - Microservices: Comprise separate, decentralized, and loosely coupled services where each service has a specific function and is decoupled from other services

Types of Data Architecture

- Data Warehouse:
  - Central data hub used for reporting and analysis
  - Data is typically highly formatted and structures for analytics use cases
  - Organizational DW: Organizes data associated with certain business team structure and processes
  - ETL:
    - Extract: Pull data from source systems
    - Transform: Clean and standardize data, organize and impose business logic
    - Load: Push data into the DW target DB systems/data marts that serve the analytical needs
  - ELT:
    - Raw data gets moved directly from production into staging area in the DW
    - Transformations are handled directly in the DW
    - Intention: Take advantage of massive computational power of cloud DWs
    - Data is processed in batches, and the transformed output is written into tables, views etc.
- Cloud Data Warehouse: Data is housed in object storage, allowing virtually limitless storage
- Data Mart: More refined subset of a warehouse designed to serve analytics and reporting, focused on single sub-organizations, departments etc. (each organization has its own DM)
- Data Lake: Central location for (un-/semi-)structured data

- Data Lakehouse:
  - Incorporates the controls, data management, and data structures found in a DW while housing data in object storage and supporting variety of querying and transformation engines
  - ACID (atomicity, consistency, isolation, durability) transactions
- Data Mesh: Divide of data between operational and analytical data
- Lambda Architecture:
  - Systems operating independently of each other- batch, streaming, serving
  - Source system is immutable and append-only, sending data to two destinations for processing (stream and batch)
  - In-stream processing intends to serve data with the lowest possible latency
  - Batch layer processes data and transforms it into a system (e.g., DW)
  - Serving layer provides a combined view by aggregating query results from the two layers
- Architecture for IoT:
  - Distributed collection of devices with an internet connection
  - Devices: Collecting and transmitting data to a downstream destination
  - IoT gateway: Hub for connecting devices and securely routing devices to appropriate destination on the internet

Technologies Across the Data Engineering Lifecycle

- FinOps:
  - Evolve cloud financial management discipline and cultural practice that enables organizations to get maximum business value by helping engineering, finance, technology, and business teams to collaborate
  - Fully operationalize financial accountability and business value by applying DevOps practices of monitoring and dynamically adjusting systems
- Serverless:
  - Run applications without managing servers behind the scenes → ne need for backend infrastructure
  - Automated scaling from zero to extremely high usage rates
  - Billed as pay-as-you-go
  - Suffer from inherent overhead inefficiency
  - Critical to monitor and model
  - Don't make sense when usage and cost exceed ongoing of running and maintaining a server
  - Typically run on containers

Data Generation in Source Systems

- Online Analytical Processing System (OLTP):
  - Built to run large analytics queries
  - Constantly listens for incoming queries → suitable for interactive analytics
  - Inefficient at handling lookups of individual records
- Change Data Capture: Extracts each change event occurring in a DB
- Logs:
  - Captures information about events that occur in systems
  - Track events and event metadata
  - Minimum: Who, what, when
- CRUD: Represents the four basic operations of persistent storage
- Insert-Only:
  - Retains history directly in a table
  - No updates of data but create new record with timestamp
  - Tables grow large, record lookups require extra overhead
- Messages:
  - Raw data communicated across two or more systems
  - Typically sent through a message queue from a publisher to a consumer
  - Discrete and singular signals in an event-driven system
- Stream:
  - Append-only log of event records
  - Accumulated in an ordered sequence
- Time:
  - Event time: Indicates when an event is generated in a source system
  - Ingestion time: Indicates when an event is ingested from a source system into a message queue
  - Process and processing time
- REST: Representational state transfer
- Webhook: Event-based data-transmission pattern
- Message queues:
  - Mechanism to asynchronously send data between discrete systems using a publish and subscribe model
  - Allows applications and systems to be decoupled from each other

Application Databases (OLTP)

- OLTP: Online Transaction Processing
- Store the state of an application, reads and writes individual data records at a high rate
- Support low latency and work well when lots of users might be interacting with the application simultaneously
- Less suited for analytics, where a single query must scan a vast amount of data
- ACID transactions:
  - Atomic: Several changes committed as a unit
  - Consistency: Any DB read will return the last written version of the retrieved item
  - Isolation: DB will be consistent with sequential execution of updates in order they were submitted

- Durability: Committed data will never be lost

Relational DBs

- Data is stored in a table of relations (rows)
- Each relation contains multiple fields (columns)
- Each relation in the table has the same schema
- Tables are typically indexed by a PK
- FK: Field with values connected with the values of PKs in other tables
- ACID compliant
- Ideal for storing rapidly changing application states

Storage

- Underlies Ingestion, Transformation, and Serving
- Storage abstractions: Data Lake, Lakehouse, Platform, Cloud Data Warehouse
- Storage systems: HDFS, RDBMS, Cache/memory-based, Object, and Streaming storage
- Indexes:
  - Provide a map of the table for particular fields and allow extremely fast lookup of individual records
  - Used for primary keys
- Columnar serialization: Allows a DB to scan only the columns required for a particular query

Storage Abstractions

Data Warehouse

- Standard OLAP data architecture for data centralization
- CDWs are often used to organize data into a Data Lake
- CDWs cannot handle truly unstructured data unlike a true Data Lake
- Can be coupled with object storage to provide a complete data Lake solution