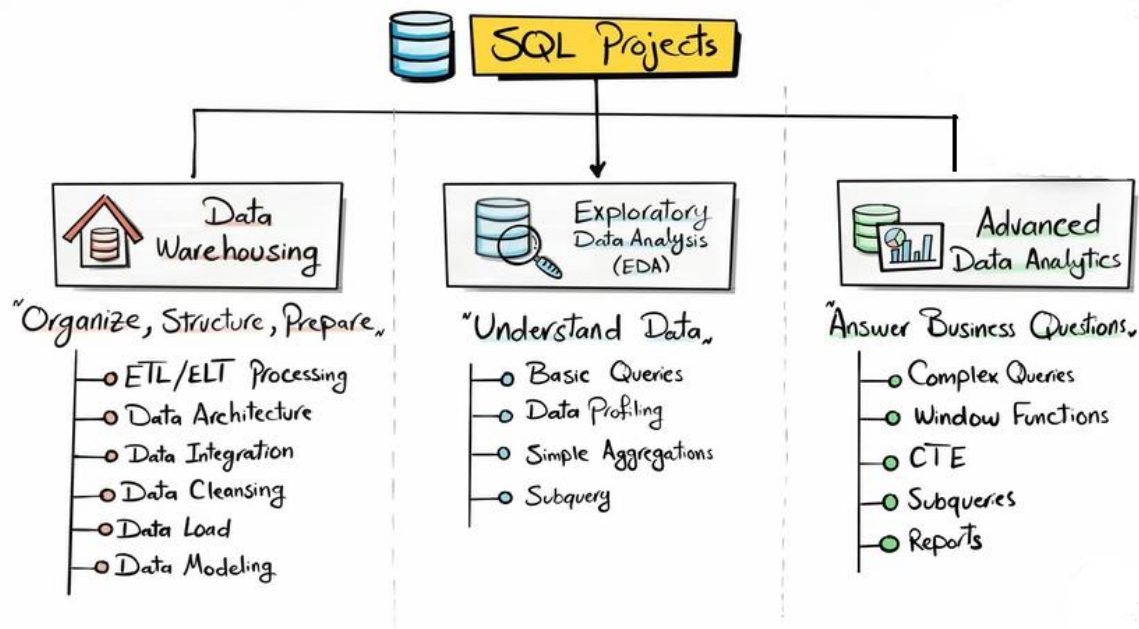
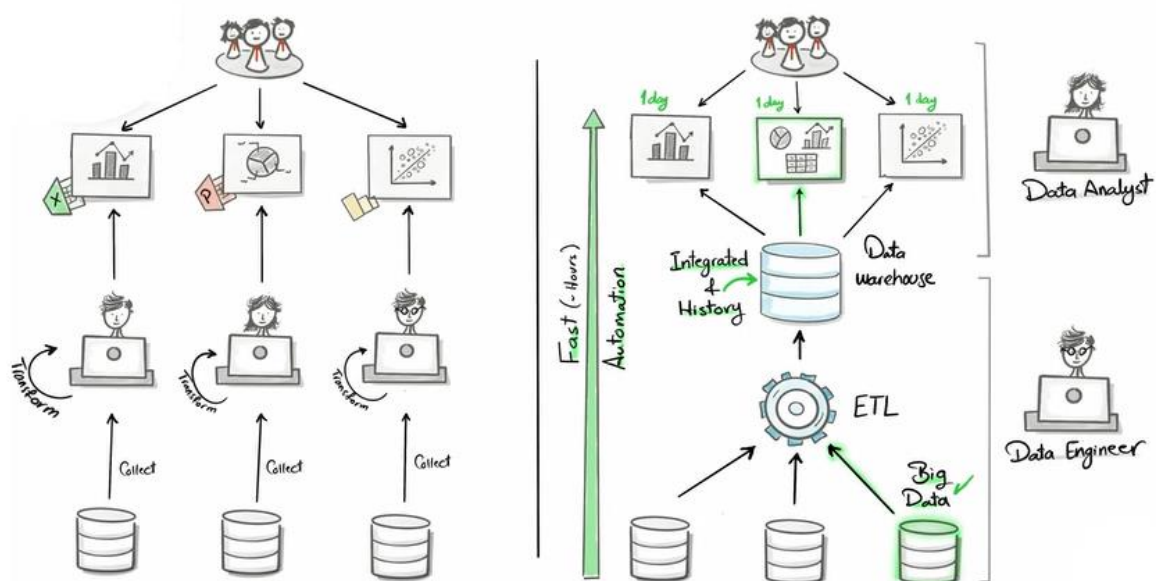


Data Warehouse Project

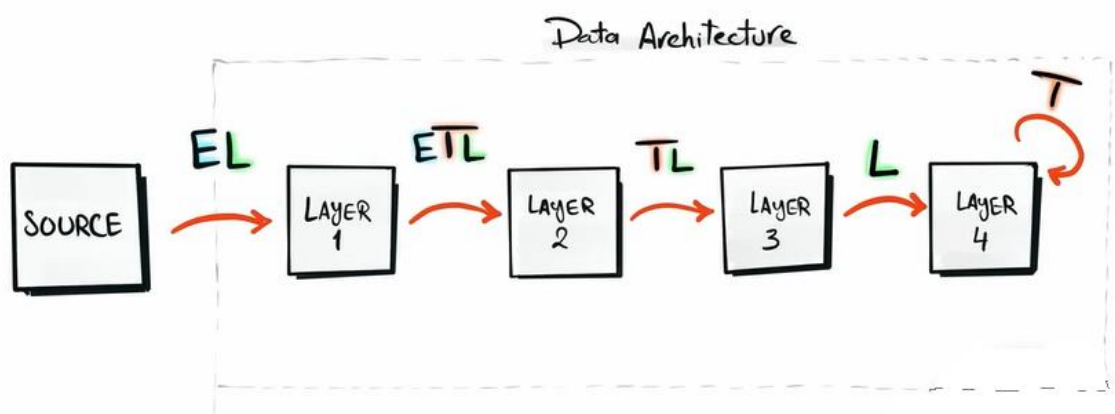
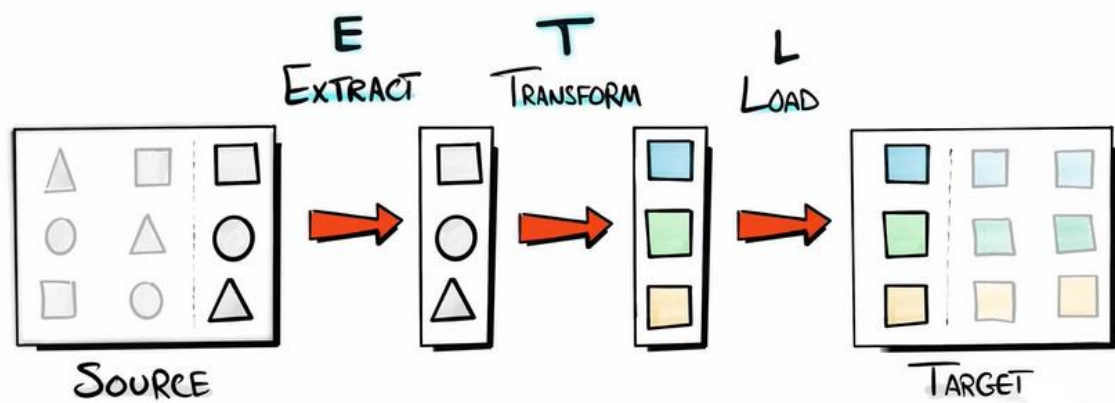


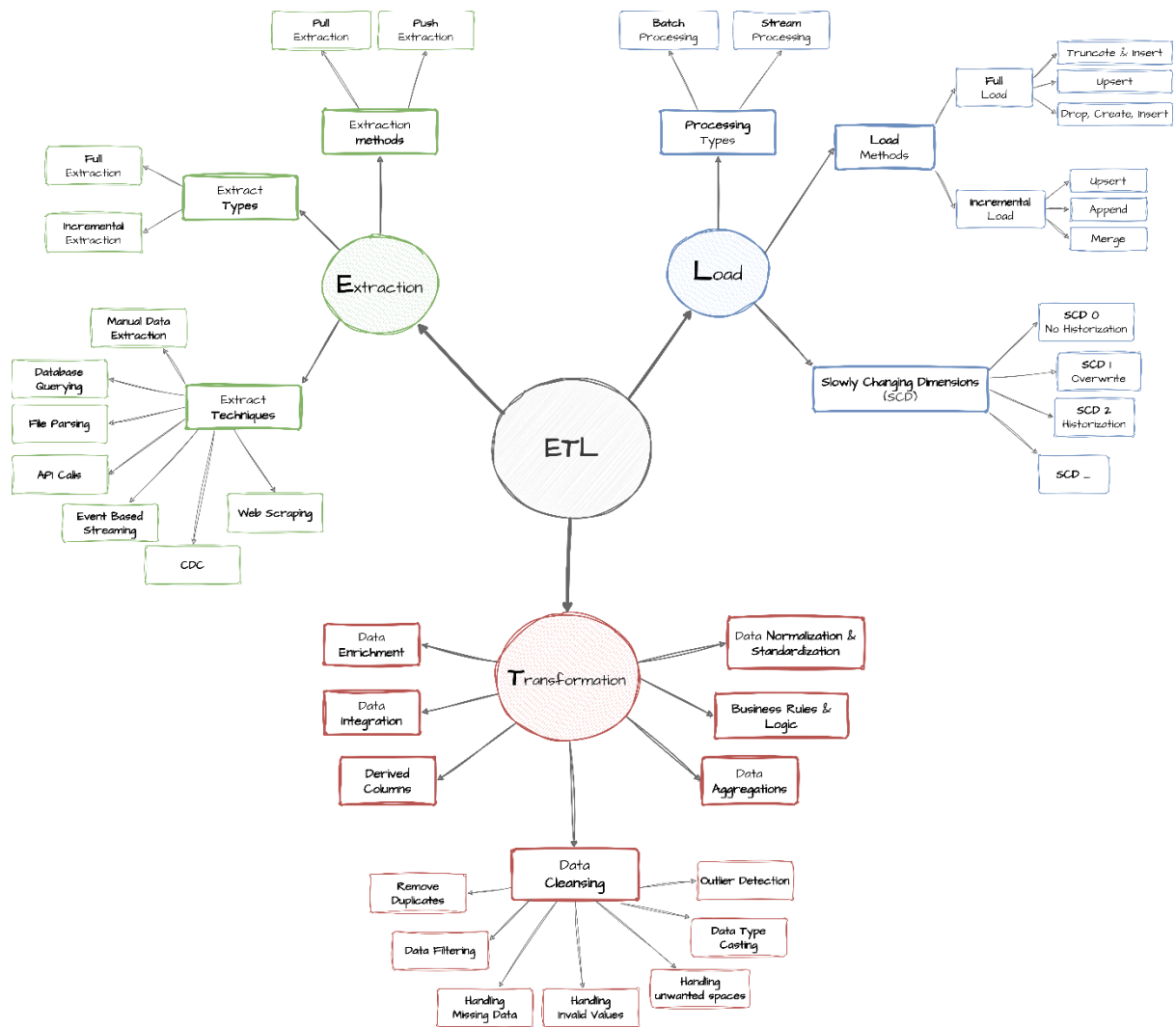
Data Warehouse:

A Data Warehouse is a Subject oriented, integrated, time variant and non-volatile collection of data in support of management's decision-making process.

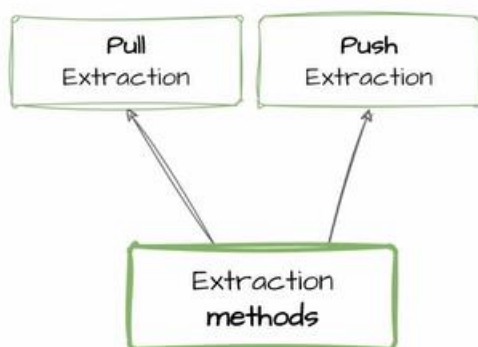


ETL





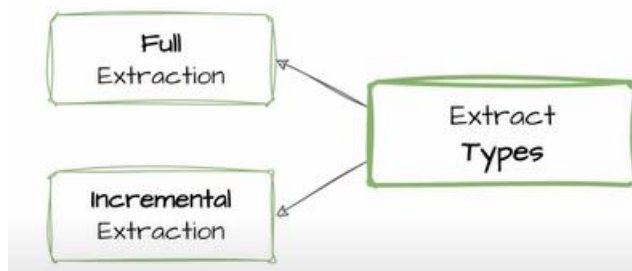
Extraction methods



Pull extraction: Pull data from source into Data Warehouse.

Push extraction: Source will push the data into Data Warehouse.

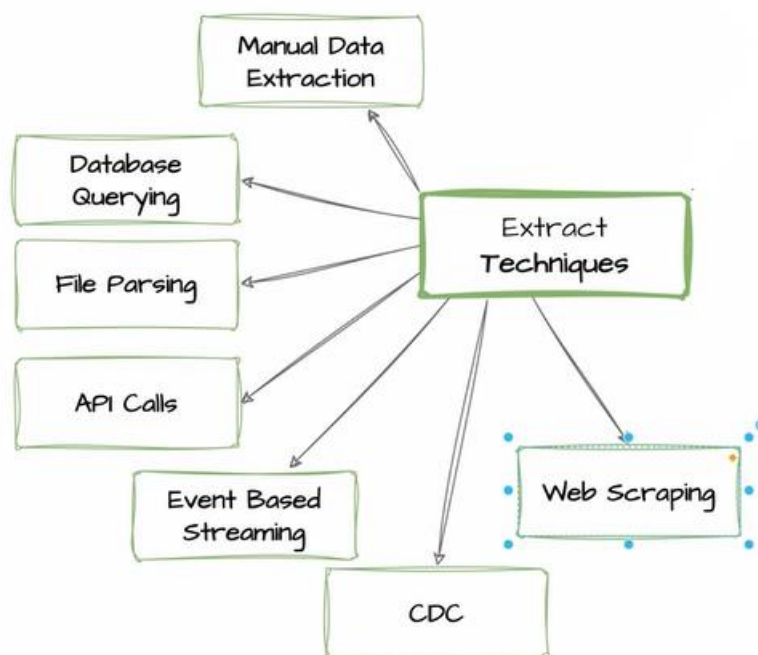
Extract types



Full extraction: Extract all the data at once.

Incremental extraction: Extract data in incremental loads.

Extract techniques



Manual Data extraction: This involves a manual process where users download or copy data from various sources, like spreadsheets or databases, and then upload it to the data warehouse. While it's straightforward, it is labour-intensive and prone to human error, making it suitable for small-scale or ad-hoc data extraction tasks.

Database Querying: A widely-used method in data warehousing where SQL (Structured Query Language) or similar query languages are employed to retrieve specific data from relational databases. This allows for efficient extraction of structured data, which can then be transformed and loaded into the data warehouse.

File Parsing: Data stored in files (e.g., CSV, JSON, XML) is read and processed to extract meaningful information. This technique is common when dealing with batch data uploads, where files are parsed, cleaned, and formatted before being integrated into the warehouse.

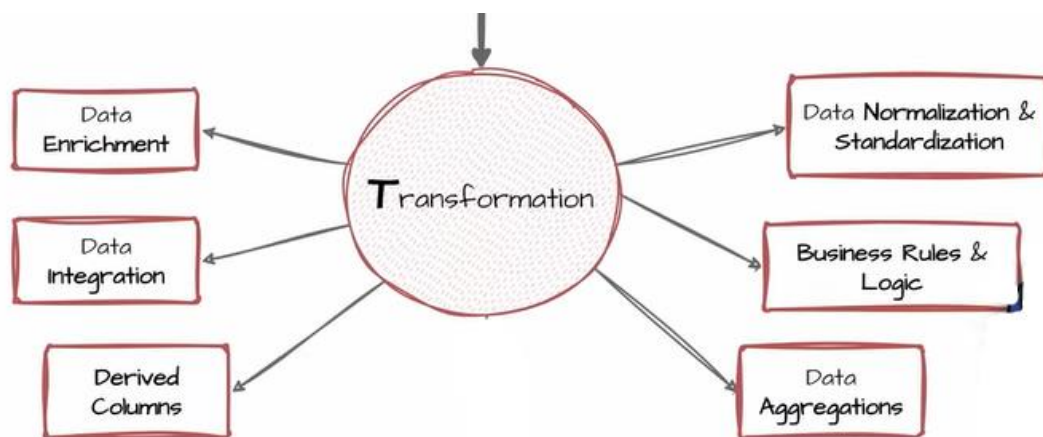
API Calls: APIs enable programmatic access to data from various systems or cloud platforms. Using API calls, data can be fetched dynamically and integrated into the warehouse. This method is particularly useful for accessing live or external data from third-party services.

Event Based Streaming: This is a real-time extraction technique where data is collected as events happen. Technologies like Apache Kafka or AWS Kinesis capture event streams, allowing them to be processed and ingested into the data warehouse for real-time analytics and decision-making.

CDC: Change Data Capture focuses on capturing only the incremental changes—new records, updates, and deletions—in the source data since the last extraction. This efficient approach reduces the processing load and ensures the warehouse reflects the most up-to-date data.

Web Scraping: When no direct data access methods like APIs are available, web scraping is used to extract data from website content. The scraped data is processed and structured before being stored in the data warehouse. It's often used for collecting public or unstructured data from web pages.

Transformation



Data Enrichment: This involves enhancing raw data by adding additional information or context to make it more meaningful. For instance, enriching customer data by appending demographic details, geographic locations, or purchase patterns from external data sources.

Data Integration: The process of combining data from multiple sources into a unified format. This typically includes merging databases, linking disparate systems, and consolidating datasets to provide a single, cohesive view for analysis in the data warehouse.

Derived Columns: These are new columns created by applying calculations, formulas, or transformations on existing data.

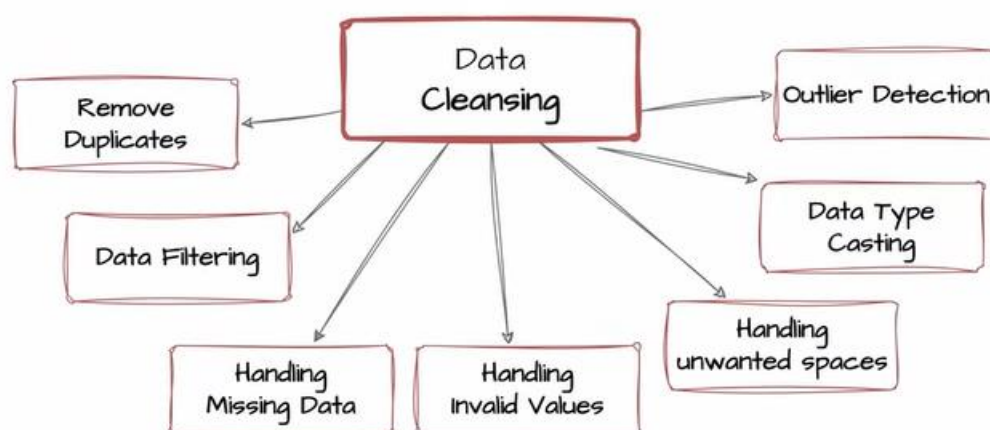
For example, calculating a “Total Price” column by multiplying the “Quantity” and “Unit Price” columns, or deriving age from a date of birth column.

Data Normalization and Standardization:

- Normalization involves organizing data to eliminate redundancy and ensure logical consistency, typically by splitting data into related tables. It’s essential for maintaining data integrity in databases.
- Standardization ensures that data is represented in a consistent format, such as using ISO standard date formats (YYYY-MM-DD) or converting all units of measurement to a standard system (e.g., converting weights from pounds to kilograms).

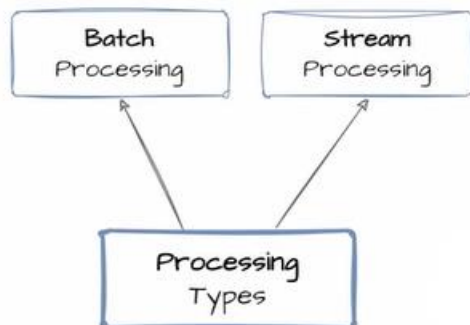
Business Rules and Logic: These refer to applying specific rules or conditions dictated by the organization to ensure data accuracy and relevance. For example, filtering out transactions under a certain amount, or flagging customers who meet certain credit score thresholds.

Data Aggregations: Aggregation involves summarizing or combining data to provide a higher-level overview. Common examples include calculating totals, averages, counts, or percentages across various data dimensions—like the total sales per region or the average rating of a product.



Load

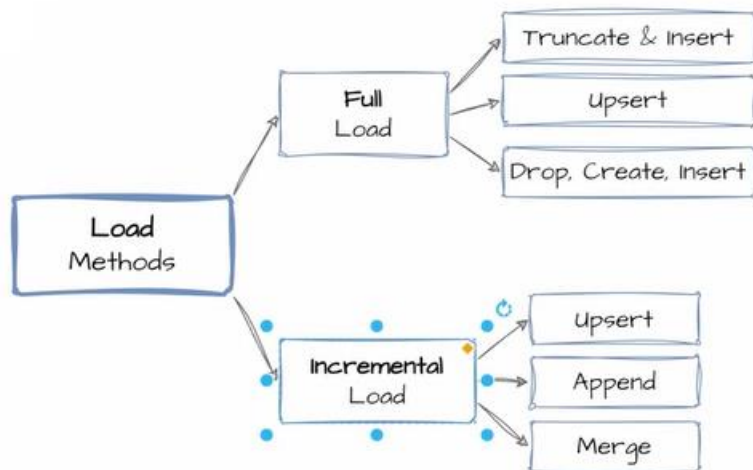
Processing types



Batch processing: Batch processing involves the collection of data over a period of time, which is then processed and loaded into the data warehouse in bulk as a single operation.

Stream processing: Stream processing involves continuously ingesting and processing data in real-time or near real-time as it becomes available, instead of waiting for a scheduled batch.

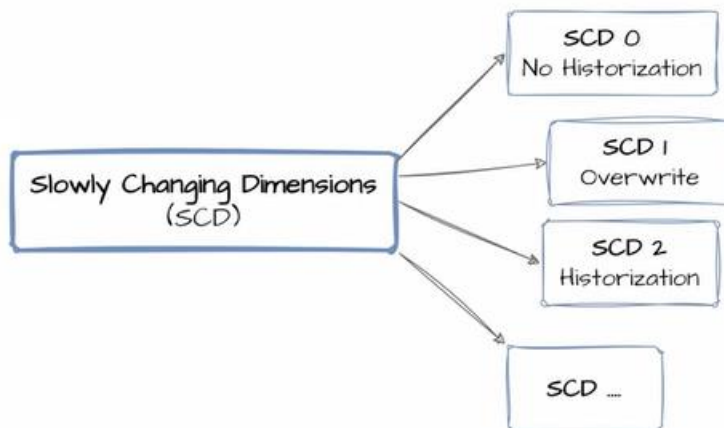
Load methods



Full load: Load all the data at once.

Incremental load: Load data in increments.

Slowly Changing Dimensions (SCD)

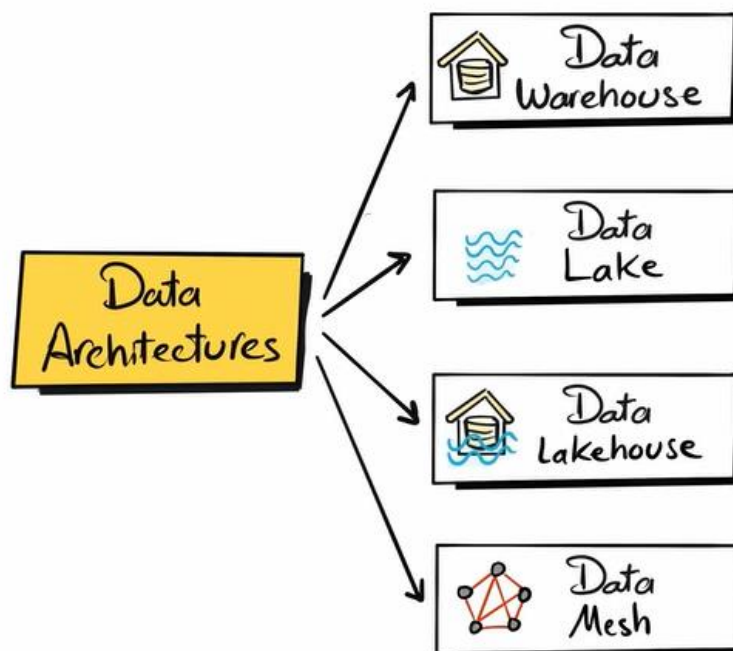


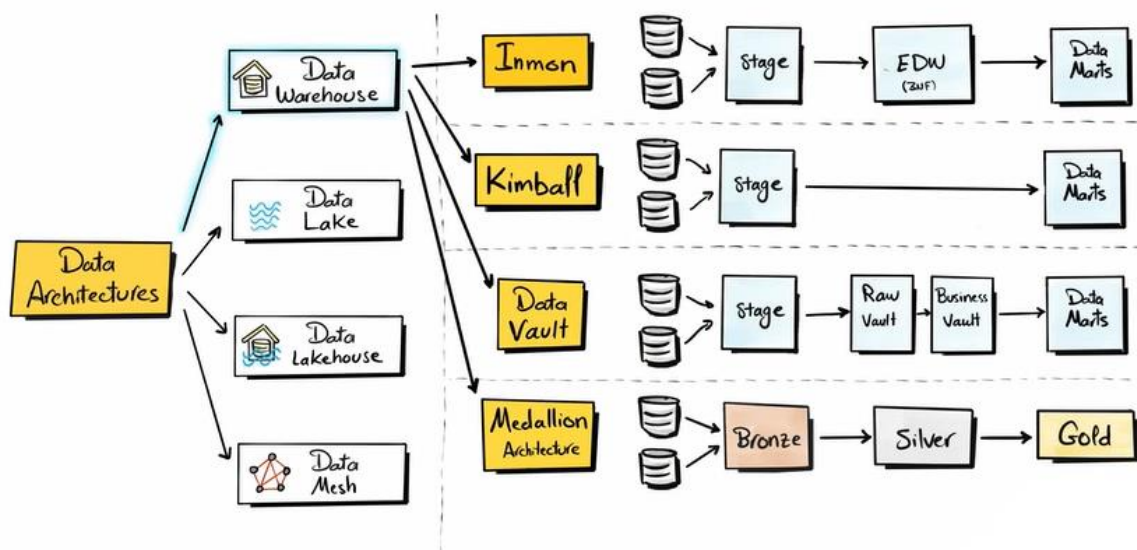
SCD 0: The original data remains unchanged, and any updates are ignored.

SCD 1: The new data overwrites the existing data, and no history is preserved.

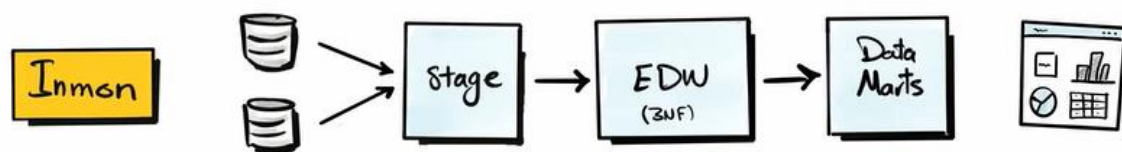
SCD 2: A separate table is created to store the historical data, while the main dimension table contains only the current data.

SCD: Combines elements of Types (add new rows),(add new columns) by retaining current data, historical data, and partial historical attributes in the same table.





Inmon model



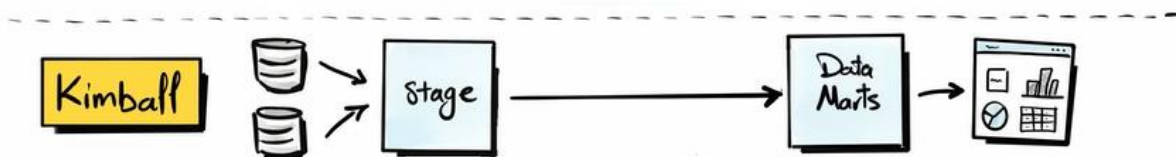
The Inmon Model, also referred to as the Top-Down Approach to data warehousing, was introduced by Bill Inmon, often called the "Father of Data Warehousing." It is a systematic and structured method for building data warehouses, focusing on enterprise-wide planning and design.

Key Features of the Inmon Model:

1. Enterprise-Wide Perspective:
 - The Inmon Model aims to create a centralized data warehouse that serves as the single source of truth for the entire organization.
 - Data is integrated across all subject areas, ensuring consistency and standardization.
2. Normalized Data Warehouse Design:

- The data warehouse is designed in a 3rd Normal Form (3NF) relational structure to avoid redundancy.
 - This structure provides flexibility and scalability, making it easier to adapt to organizational changes or integrate new data sources.
3. Data Marts Derived from the Data Warehouse:
- In the Inmon approach, data marts (which are smaller, department-specific subsets of the data) are created only after the centralized data warehouse is established.
 - These data marts are built for specific analytical needs, ensuring consistency with the enterprise-wide data warehouse.
4. ETL Process:
- Data from operational systems is extracted, transformed, and loaded (ETL) into the data warehouse.
 - The ETL process ensures data quality, consistency, and integration before populating the data warehouse.
5. Subject-Oriented and Time-Variant Data:
- The data warehouse is organized around subject areas (e.g., sales, finance, customers) rather than application-specific data.
 - It also captures historical data, making it time-variant and useful for trend analysis

Kimball model



The Kimball Model, also known as the Bottom-Up Approach to data warehousing, was proposed by Ralph Kimball. It emphasizes a simpler and faster way to build a data warehouse by focusing on business processes and departmental needs first. Here's a detailed explanation of the Kimball Model:

Key Features of the Kimball Model:

1. Business-Centric Approach:

- The Kimball methodology starts with understanding and addressing specific business processes or departmental needs.
- Individual data marts are created to support these processes, making it business-driven.

2. Dimensional Data Modelling:

- The model uses Star Schema or Snowflake Schema designs, which are denormalized structures optimized for querying and reporting.
- Dimensions (e.g., time, customer, product) and fact tables (e.g., sales, revenue) are central to this design, ensuring data is user-friendly for analysis.

3. Data Marts as Building Blocks:

- Each data mart is designed independently for a specific business area, such as sales, marketing, or finance.
- Over time, these data marts are integrated to form a logical data warehouse that serves the entire organization.

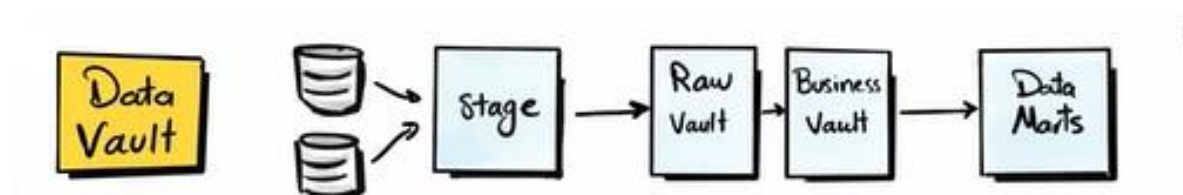
4. ETL (Extract, Transform, Load) Process:

- Data is extracted from source systems, transformed into a denormalized format, and loaded into the data marts.
- The focus is on rapid delivery of usable data for reporting and analysis.

5. Focus on User Experience:

- The Kimball approach prioritizes ease of access and performance for end-users, making it ideal for business intelligence and ad-hoc queries.

Data Vault



The Data Vault model is a data modelling approach specifically designed for data warehousing that emphasizes scalability, flexibility, and historical accuracy. It is ideal for handling large-scale, dynamic, and rapidly changing datasets. Below is a detailed explanation of its components and characteristics:

Core Components of Data Vault

1. Hub:
 - Represents the key business entities or core concepts (e.g., Customer, Product, Order).
 - Stores only the unique business key (natural key) and metadata, such as load timestamps and data sources.
 - Immutable: Once data is loaded into a hub, it is never modified or deleted.
2. Link:
 - Captures the relationships between hubs, representing many-to-many or complex relationships.
 - Ensures that changes in relationships can be recorded without impacting the hub structure.
 - Stores metadata like load timestamps to allow auditing.
3. Satellite:
 - Stores the descriptive attributes related to a hub or link (e.g., customer names, addresses, product descriptions).
 - Maintains a history of changes by adding new records rather than overwriting old data.
 - Metadata fields like timestamps enable tracking the validity period of each record.

Key Features of Data Vault

1. Modularity:
 - Hubs, links, and satellites are independent entities, allowing for easy updates or extensions without affecting the overall structure.
2. Scalability:
 - The highly normalized structure of the Data Vault model allows it to handle large volumes of data efficiently, making it suitable for enterprise-scale environments.

3. Flexibility:

- Easily adapts to schema changes and supports adding new data sources without redesigning the entire model.

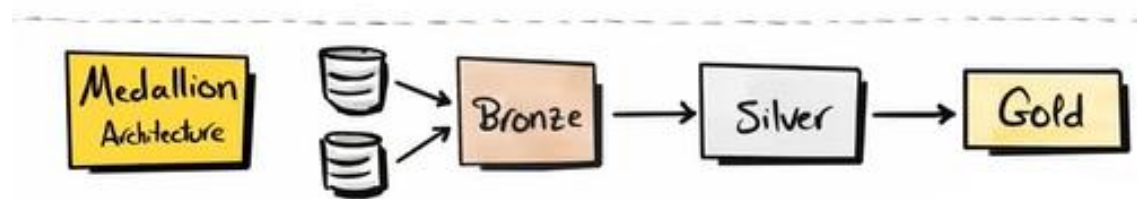
4. Auditability:

- Metadata tracking for all entities (hubs, links, satellites) ensures data lineage, traceability, and compliance with regulatory requirements.

5. Historical Data Management:

- Data Vault captures a complete history of changes, preserving all versions of data for detailed analysis.

Medallion model



The Medallion Model is a data architecture approach that organizes data into distinct layers or "medallions." Each layer represents a level of data refinement and quality, helping to streamline the process of data ingestion, transformation, and analysis. Though it's more commonly associated with data lake houses, its principles can also apply to modern data warehousing. Here's a breakdown:

Layers in the Medallion Model

1. Bronze Layer (Raw Data):

- Purpose: This layer stores raw, unprocessed data directly ingested from various source systems.
- Characteristics:
 - Retains data in its original format (e.g., CSV, JSON, XML, Parquet).
 - May include duplicates, inconsistencies, or missing values.
 - Acts as an immutable source of truth for all incoming data, supporting audits and traceability.
- Application in Warehousing:

- Raw logs, transactional data dumps, or IoT data streams are preserved for future processing or lineage purposes.

2. Silver Layer (Cleaned and Curated Data):

- Purpose: Provides data that has been cleaned, transformed, and partially processed.
- Characteristics:
 - Data is deduplicated, validated, and enriched for consistency and accuracy.
 - Lightweight transformations (e.g., filtering, joining datasets) are applied.
- Application in Warehousing:
 - Ready-to-use data for intermediate analytics or machine learning use cases.
 - Facilitates exploration and experimentation before final transformations.

3. Gold Layer (Business-Ready Data):




- Purpose: Contains highly refined and aggregated data optimized for specific business use cases.
- Characteristics:
 - Tailored for reporting, dashboards, and operational analytics.
 - Often organized using dimensional models (e.g., star schemas) to enhance performance and usability.
- Application in Warehousing:
 - Data for executive dashboards, key performance indicators (KPIs), and domain-specific insights.

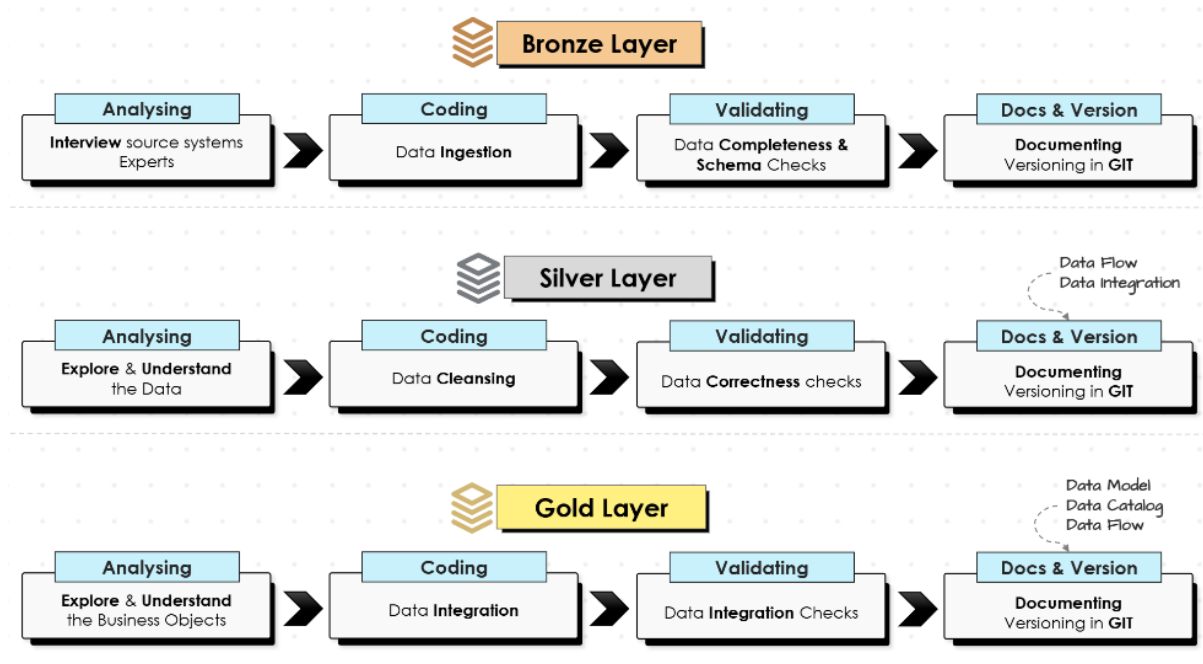
Features and Benefits

- Scalability: Supports large volumes of data by separating workloads into layers, allowing for efficient storage and processing.
- Traceability: Retaining raw data in the bronze layer ensures lineage and the ability to reproduce intermediate datasets.
- Flexibility: Layers can be expanded or modified independently as business requirements evolve.

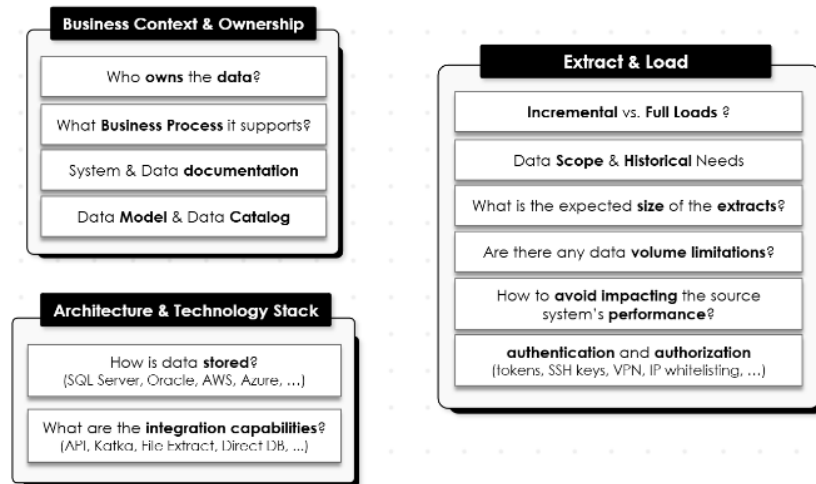
- **Quality Control:** Each layer progressively improves data quality, making it trustworthy for decision-making.
- **Support for BI and Analytics:** The gold layer enables business intelligence, while earlier layers support experimentation and advanced analytics.

Data Layers

	 Bronze Layer	 Silver Layer	 Gold Layer
Definition	Raw, unprocessed data as-is from sources	Clean & standardized data	Business-Ready data
Objective	Traceability & Debugging	(Intermediate Layer) Prepare Data for Analysis	Provide data to be consumed for reporting & Analytics
Object Type	Tables	Tables	Views
Load Method	Full Load (Truncate & Insert)	Full Load (Truncate & Insert)	None
Data Transformation	None (as-is)	<ul style="list-style-type: none"> - Data Cleaning - Data Standardization - Data Normalization - Derived Columns - Data Enrichment 	<ul style="list-style-type: none"> - Data Integration - Data Aggregation - Business Logic & Rules
Data Modeling	None (as-is)	None (as-is)	<ul style="list-style-type: none"> - Start Schema - Aggregated Objects - Flat Tables
Target Audience	- Data Engineers	<ul style="list-style-type: none"> - Data Analysts - Data Engineers 	<ul style="list-style-type: none"> - Data Analysts - Business Users

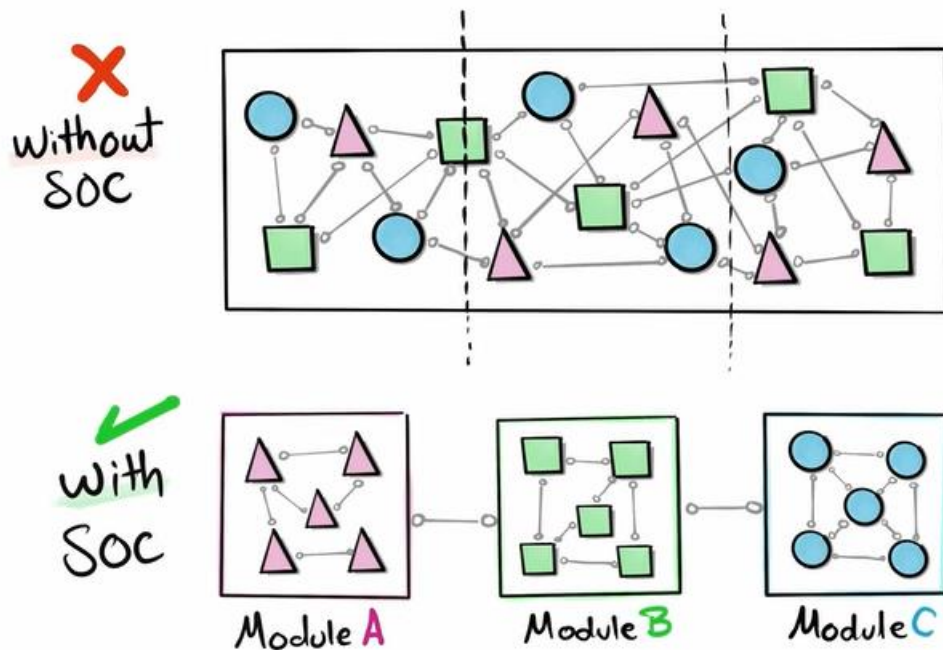


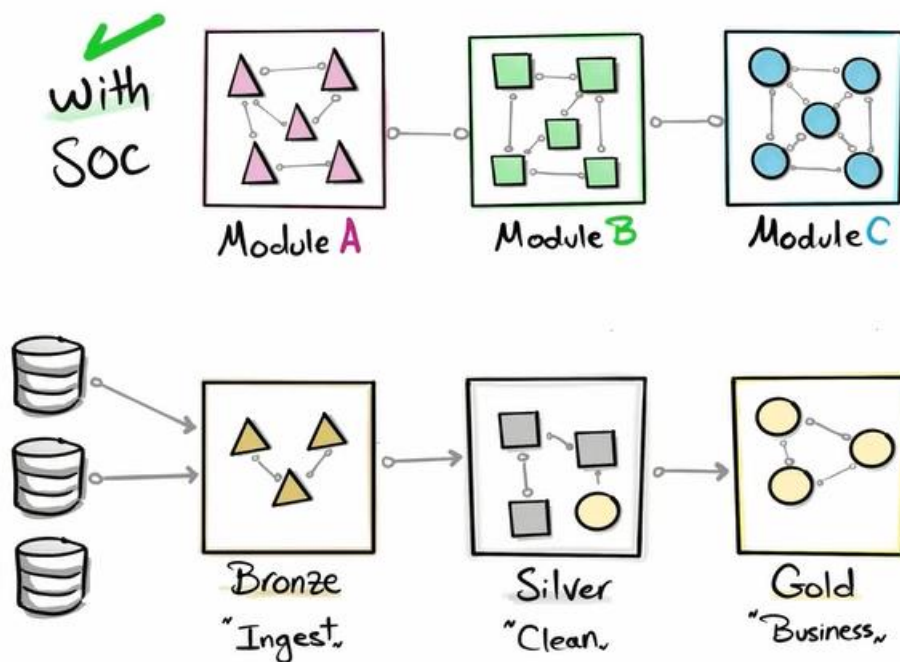
Source System Interview



Design Data layers

SOC-Separation of concerns



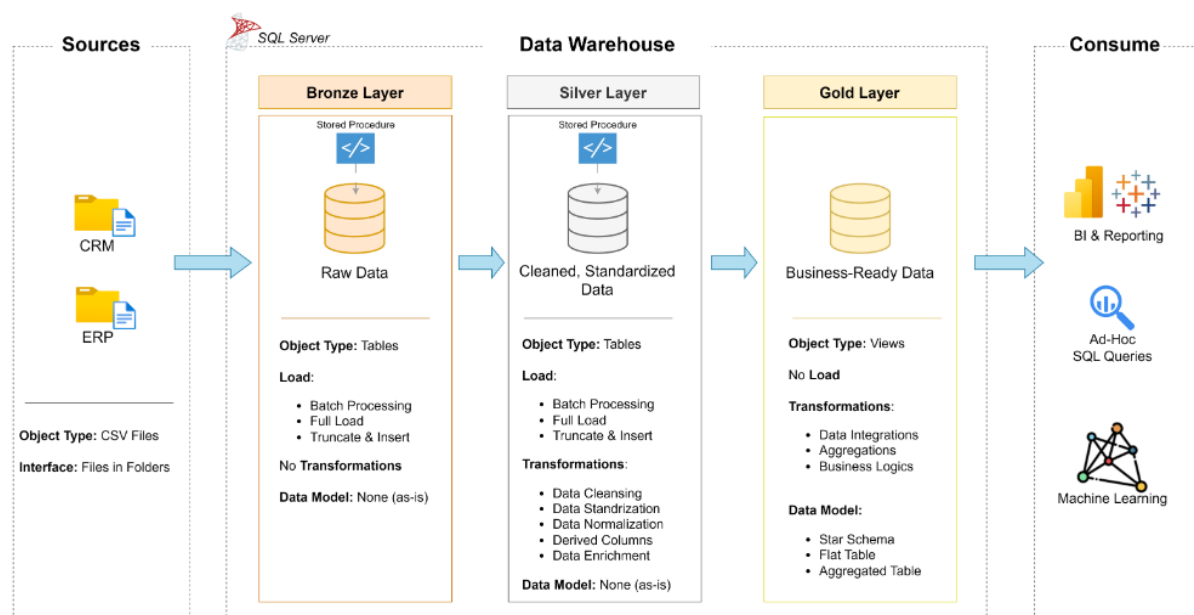


In data warehousing, Separation of Concerns (SoC) refers to the architectural design principle where different aspects or functionalities of a system are divided into distinct, independent modules or layers. This makes each module responsible for a specific concern, allowing for better manageability, scalability, and maintainability of the data warehouse.

1. **Data Ingestion Layer:** Handles the extraction, transformation, and loading (ETL/ELT) of data from source systems into the data warehouse. This layer focuses on integrating diverse data sources.
2. **Data Storage Layer:** Manages the organization, structuring, and storage of data in the warehouse. It deals with schemas like star or snowflake models.
3. **Data Access Layer:** Ensures that users can retrieve data efficiently, often involving query engines or APIs.
4. **Analytical Layer:** Focuses on enabling business intelligence (BI), dashboards, and analytics tools to leverage the data effectively.
5. **Security and Governance Layer:** Ensures proper authorization, authentication, data masking, and compliance with regulations like GDPR.

By separating these concerns, teams can work on specific areas without disrupting others, improve performance, and implement changes or updates more seamlessly.

Data Architecture

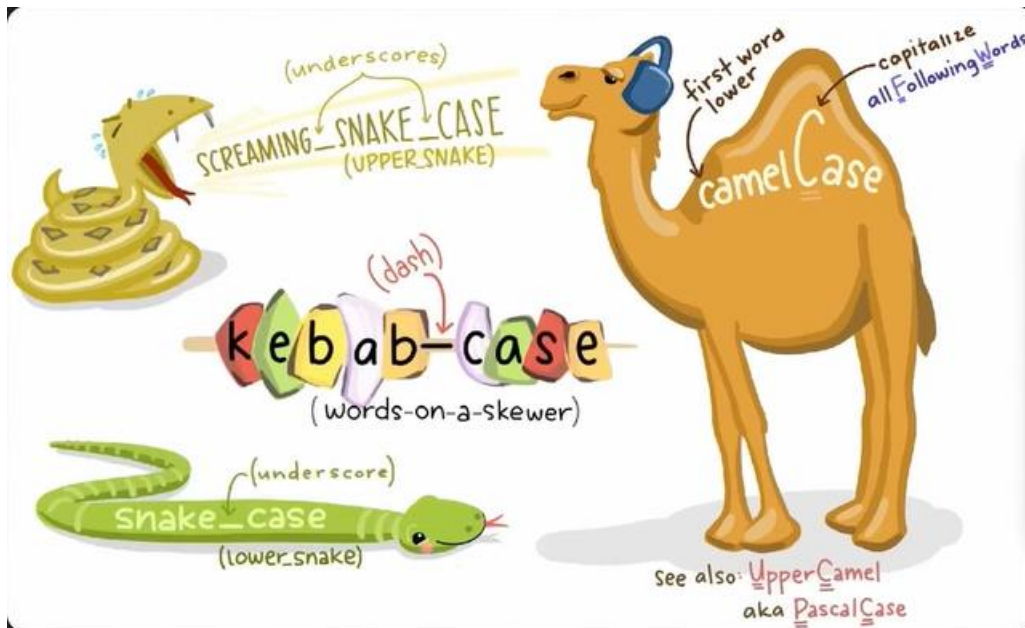


Naming Conventions

Naming Conventions

Set of Rules or Guidelines
for naming anything in the project.

- Database
- Schema
- Tables
- Store Procedures ...



Camel case: Camel case is a way of writing or formatting text where words are joined together without spaces, and each word starts with a capital letter, except for the very first word. It gets its name because the capital letters in the middle of the text resemble the humps of a camel.

For example:

- thisIsCamelCase
- myFavoriteBook

Kebab case: Kebab case is another text formatting style where words are written in lowercase and separated by hyphens (-). It's called "kebab case" because the hyphens resemble skewers in a kebab. This style is commonly used in web development, particularly for naming files, URLs, and CSS classes, as it's both readable and URL-friendly.

For example:

- this-is-kebab-case
- my-favourite-book

Screaming snake case: Screaming snake case is a text formatting style where all letters are written in uppercase, and words are separated by underscores (_). It's commonly used in programming to name constants, making it visually distinct and easy to recognize.

For example:

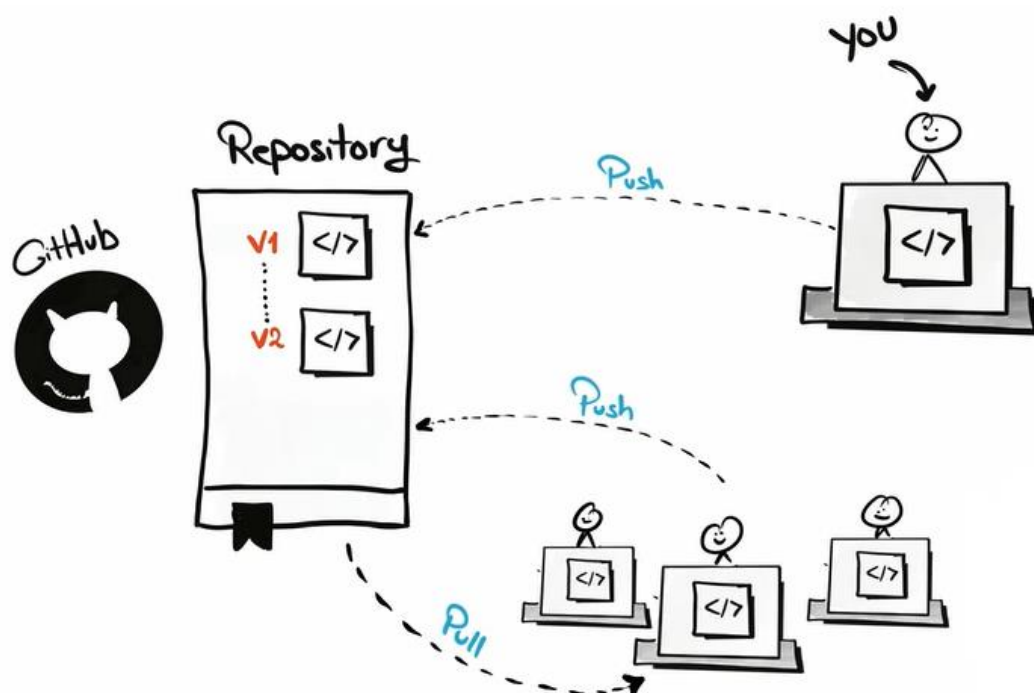
- THIS_IS_SCREAMING_SNAKE_CASE
- MY_FAVORITE_BOOK

Snake case: Snake case is a text formatting style where all letters are in lowercase (or sometimes uppercase), and words are separated by underscores (_). It's called "snake case" because the underscores slithering between the words resemble a snake.

For example:

- `this_is_snake_case`
- `my_favorite_book`

Github



Markdown (.md)

Lightweight markup language that you can use
to add formatting elements to plaintext text documents.

What is Markdown?

see [Markdown](www.markdownlink.com)

> Markdown is a lightweight markup language

List of tips

1. **One asterisk Italicizes**

2. ****Two asterisks emphasize****

What is Markdown?

see [Markdown](#)

Markdown is a lightweight markup language

List of tips

1. *One asterisk Italicizes*

2. **Two asterisks emphasize**