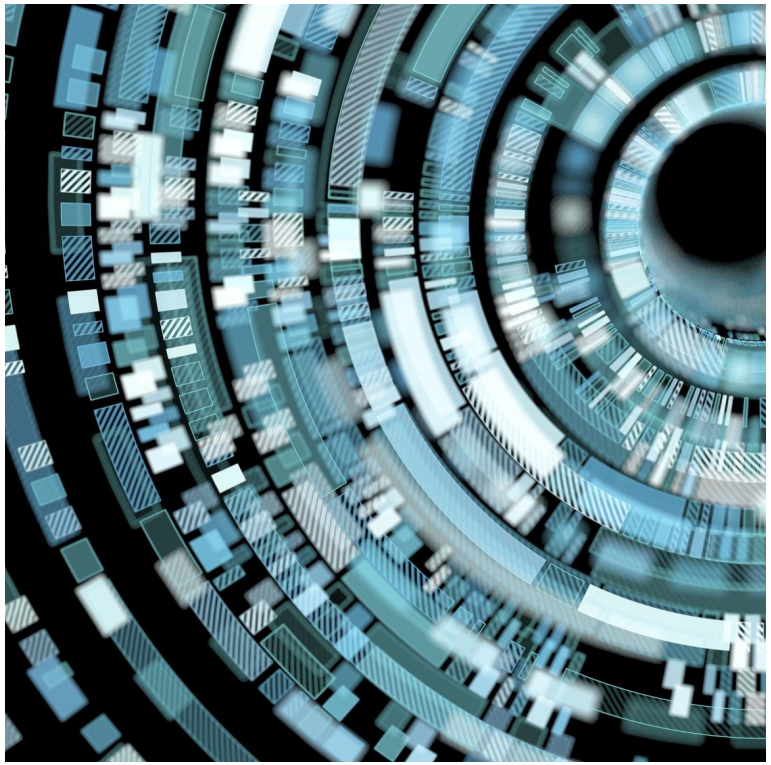
Introduction to Data Engineering

- Overview, Lifecycle, and Applications
- Pawan Kumar Sharma



What is Data Engineering ?

DEFINITION

IMPORTANCE

APPLICATIONS

Definition of Data Engineering

•The field of software engineering focused on the design, development, and management of systems that handle large volumes of data for Ensuring data is accessible, reliable, and ready for analysis and reporting.

Importance of Data Engineering

•Data engineering is crucial for ensuring that data is accessible, reliable, and ready for analysis. It supports data-driven decision-making, enhances business intelligence, and enables real-time analytics. By managing data pipelines, storage, and processing, data engineering allows organizations to extract valuable insights and maintain data quality and consistency.

Applications of Data Engineering

- Business Intelligence
- Machine Learning
- Data Warehousing
- Real-Time Analytics
- ETL (Extract, Transform, Load) Processes

Role of a Data Engineer

Responsibilities

Skills Required

Tools Used

Responsibilities of a Data Engineer

Data Pipeline Development

Data Integration

Database Management

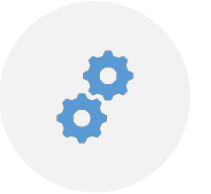
Data Quality Assurance

Collaboration

Skills Required for a Data Engineer

Proficiency in programming languages: Python, Java, Scala.

Knowledge of SQL and NoSQL databases.



Experience with ETL (Extract, Transform, Load) processes.

Tools Used by Data Engineers

- SQL Databases: MySQL, PostgreSQL, Oracle.
- NoSQL Databases: MongoDB, Cassandra, DynamoDB.
- ETL Tools(SSIS)
- Data pipeline tools(Airflow)

Data Engineering Lifecycle

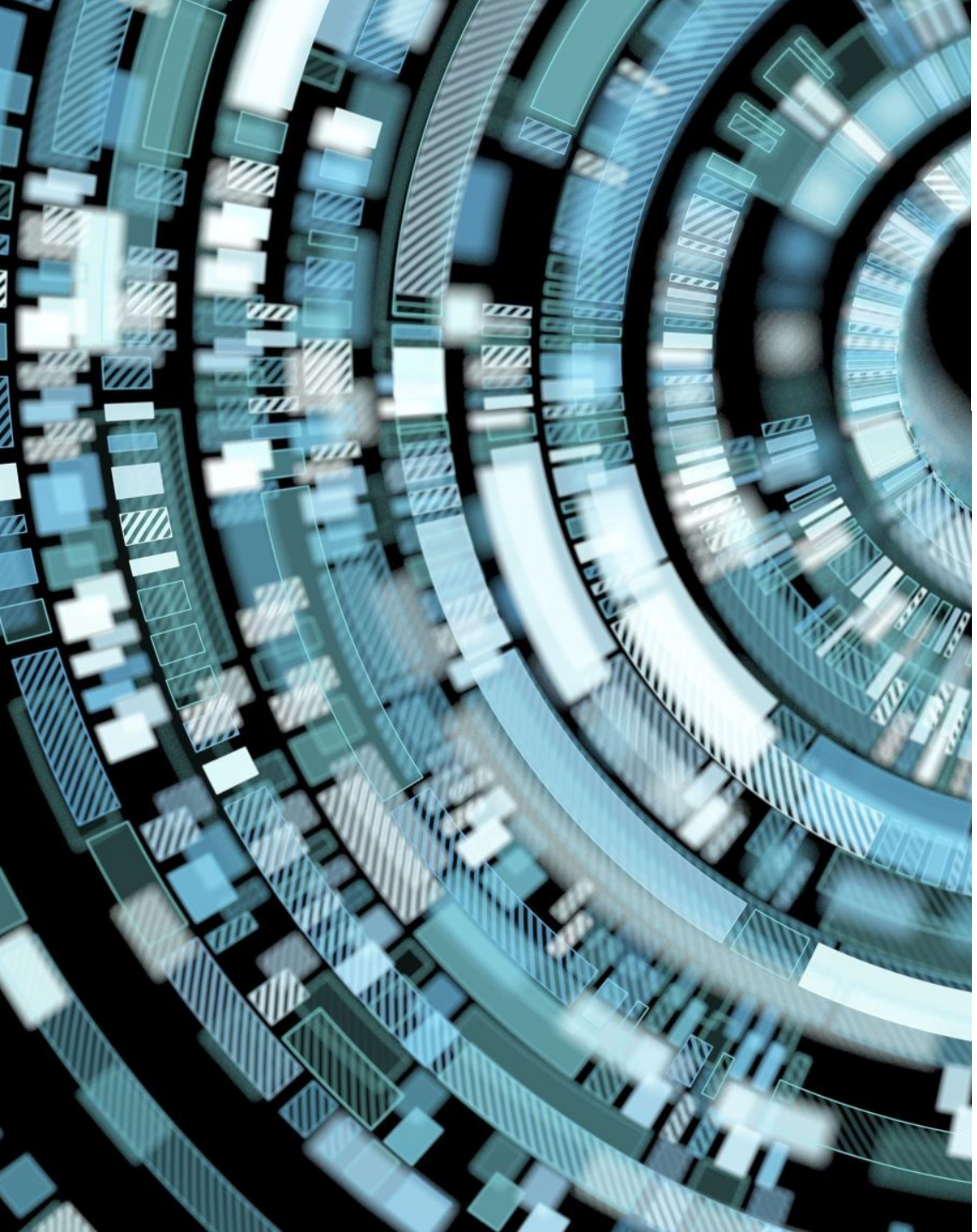
Data Generation

Data Collection

Data Storage

Data Processing

Data Analysis



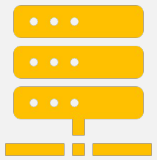
What is Data Generation ?

- Data generation is the process of creating data from various sources.
- Examples of data sources: Sensors, user interactions, social media, transactions, etc.

Types of Data Generated

Structured data: Tables, databases.

Unstructured data: Text, images, videos.



Semi-structured data: JSON, XML,HTML.

Importance of Data Generation

- Basis for informed decision-making.
- Enhances the capability to analyze trends and patterns.

What is Data Collection & Data Collection Techniques ?

The process of gathering and measuring information on variables of interest.

Surveys and questionnaires.

Online tracking tools.

Logs and event data.

Challenges in Data Collection

- Ensuring data accuracy.
- Managing large volumes of data.
- Privacy and security concerns.

What is Data Storage?

- The process of saving data in a systematic way for future use.

Types of Data Storage

- Relational databases: SQL, PostgreSQL.
- NoSQL databases: MongoDB, Cassandra.
- Data warehouses: Snowflake, Amazon Redshift.

What is Data Processing ?

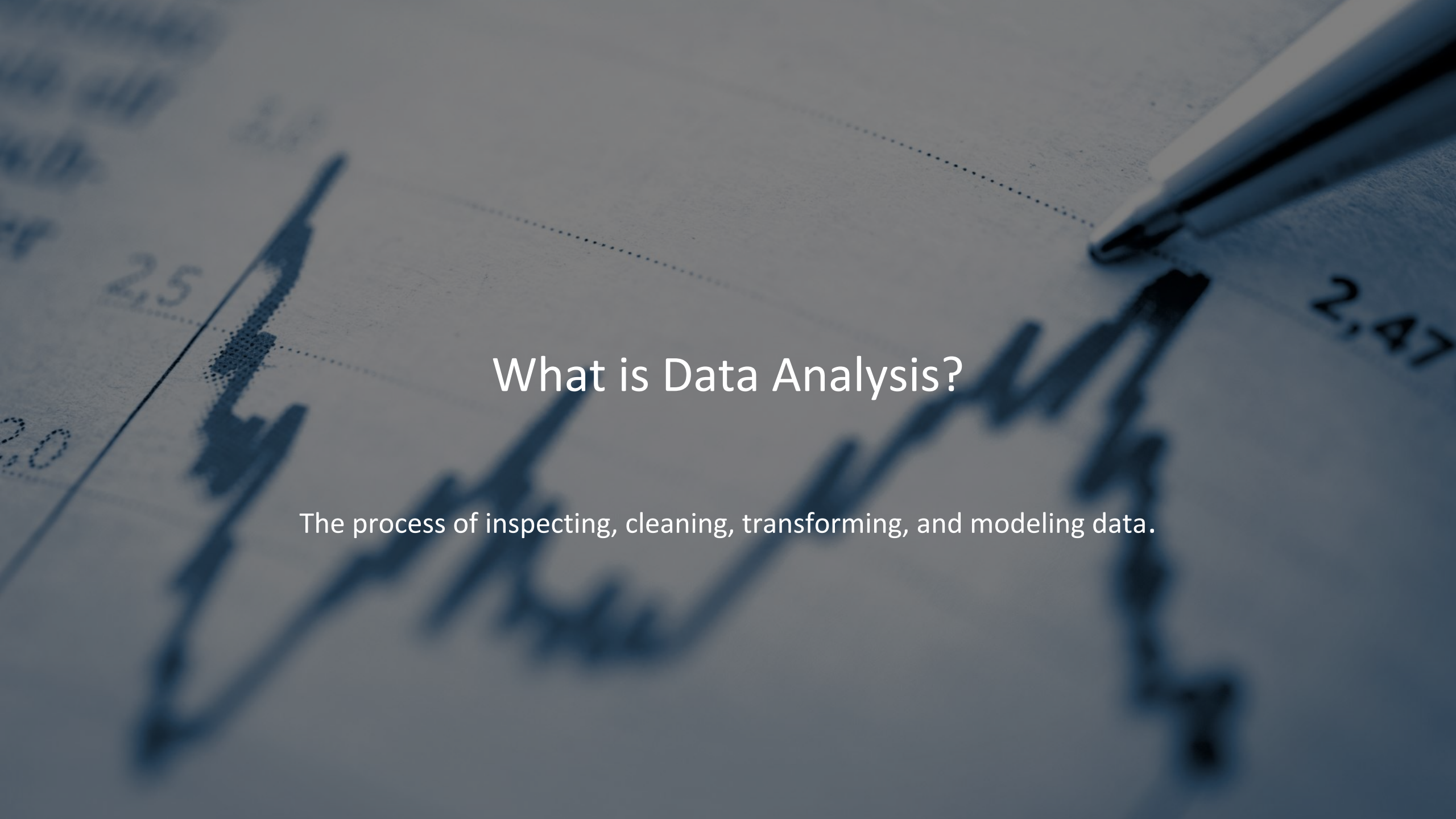
- The act of converting raw data into meaningful information.

Techniques :

- Batch processing.
- Stream processing.
- Real-time processing.

Challenges in Data Processing

- Handling big data volumes.
- Ensuring data quality.
- Maintaining data integrity.

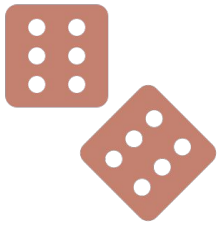


What is Data Analysis?

The process of inspecting, cleaning, transforming, and modeling data.

Types of Data Analysis

Descriptive analysis.



Predictive analysis.

Prescriptive analysis.

Data Generation and Collection

SOURCES OF DATA

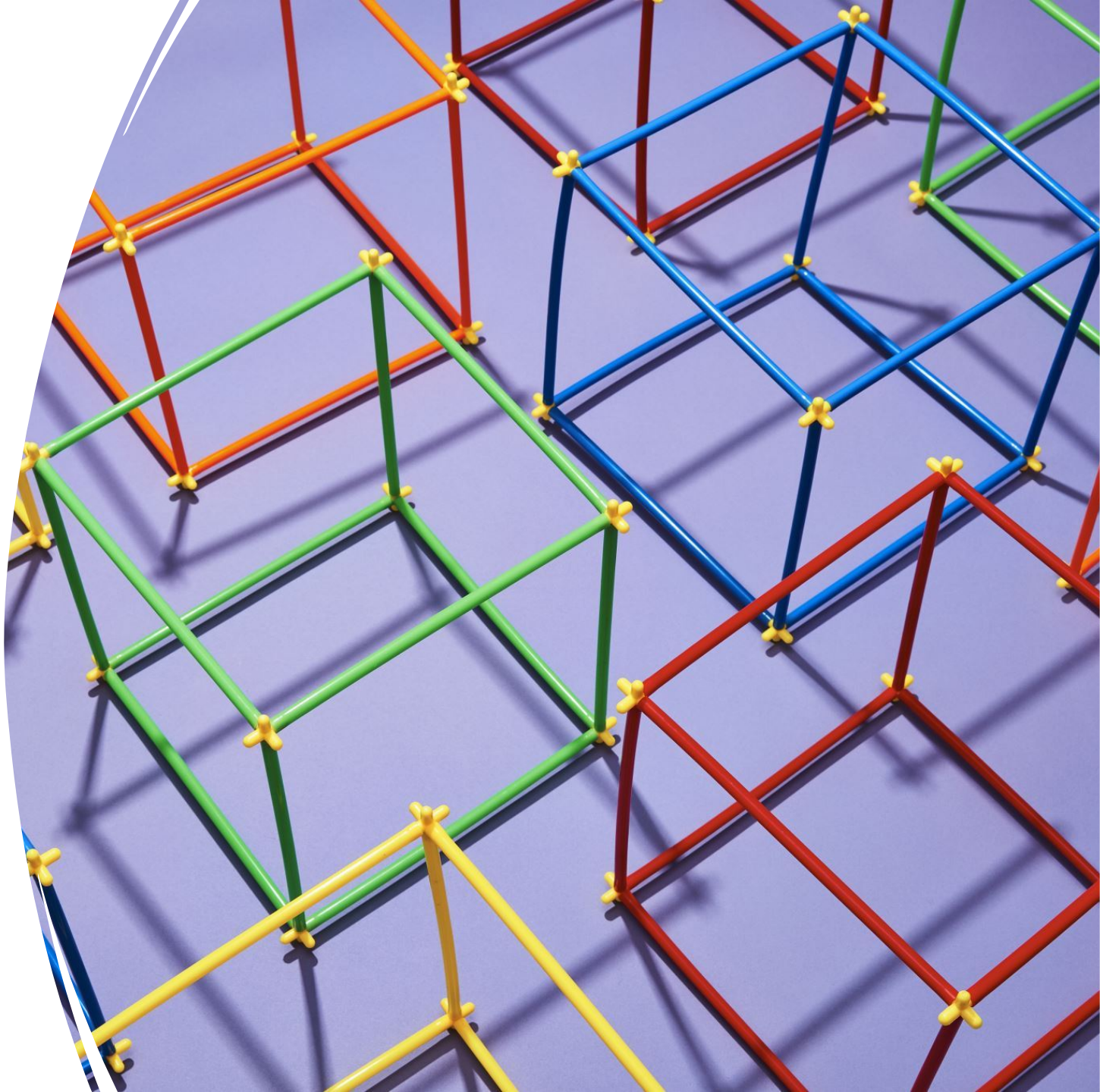
DATA LAKE

Data Sources

- Sources from which data is generated and collected for analysis.

Types of Data Sources

- **Internal Data Sources**: Data generated within an organization (e.g., transactional data, CRM data).
- **External Data Sources**: Data collected from outside the organization (e.g., social media, third-party data).



Data Lake

- A centralized repository that allows you to store all your structured and unstructured data at any scale.

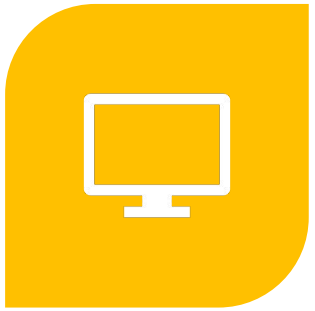
Features of a Data Lake

- **Scalability**: Can handle large volumes of data.
- **Flexibility**: Supports all data types (structured, semi-structured, unstructured).
- **Accessibility**: Data is easily accessible for processing and analysis.

Data Collection Methods

BATCH PROCESSING

STREAMING



WEB SCRAPING

APIS AND DATA EXTRACTION

Batch Processing

The collection and processing of data in large volumes at scheduled intervals.

Characteristics of Batch Processing

Scheduled Intervals: Data is processed at specific times (e.g., nightly, weekly).

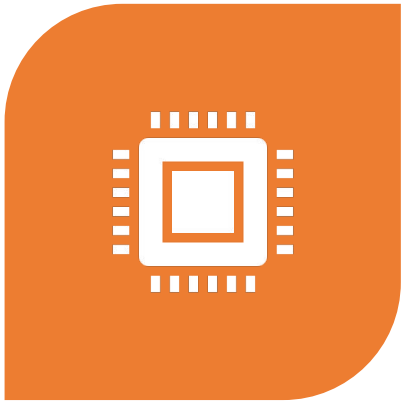
Large Volumes: Suitable for processing large datasets.

Non-Real-Time: Data is not processed in real-time, leading to some delay.

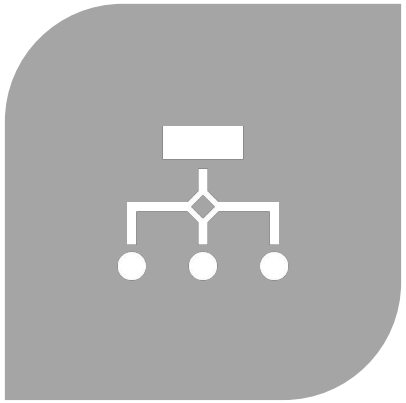
Streaming

The real-time collection and processing of data as it is generated.

Characteristics of Streaming



REAL-TIME PROCESSING: DATA IS PROCESSED AS SOON AS IT IS GENERATED.

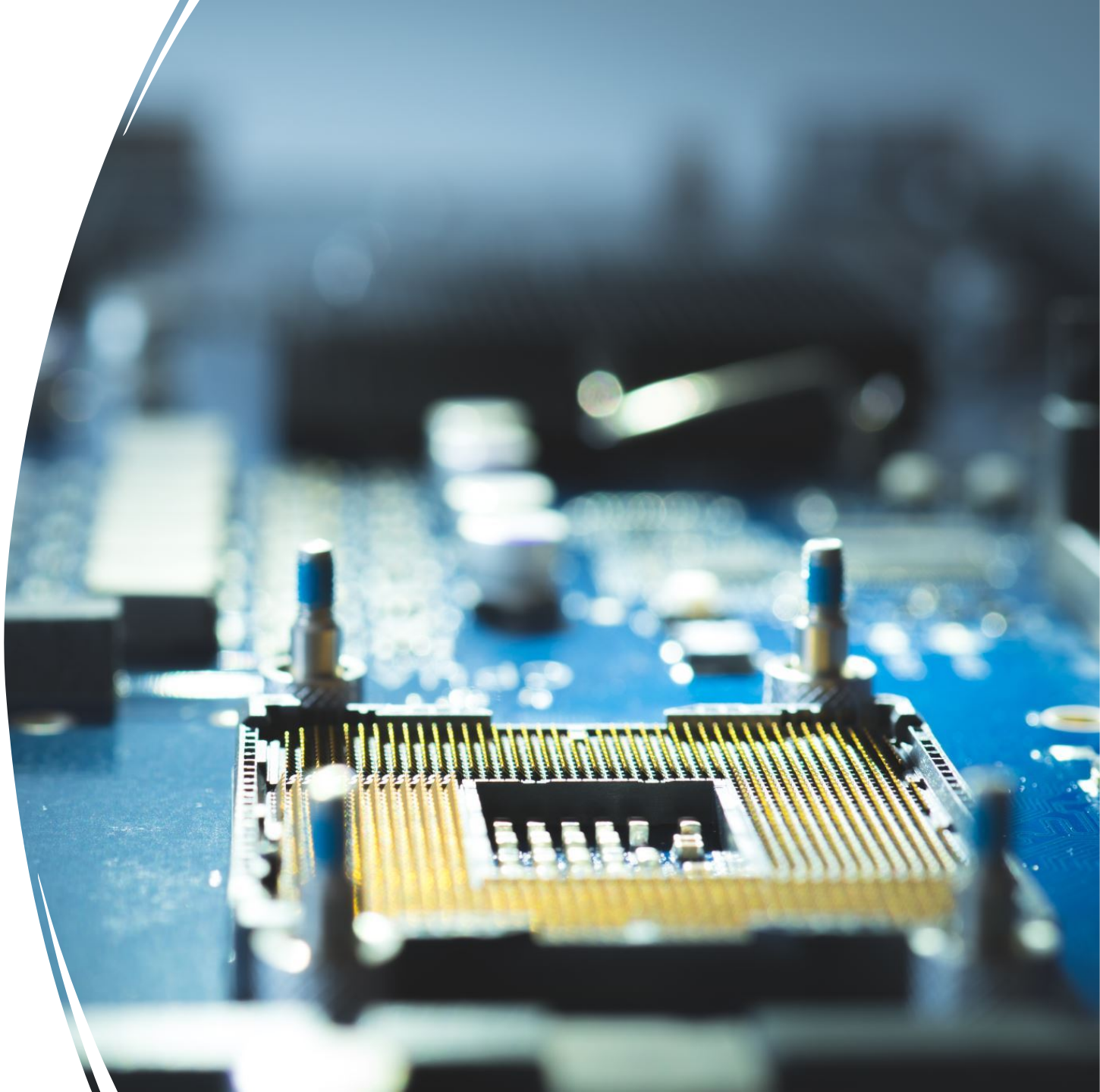


CONTINUOUS FLOW: DATA IS COLLECTED AND PROCESSED CONTINUOUSLY.

LOW LATENCY: IMMEDIATE INSIGHTS FROM DATA.

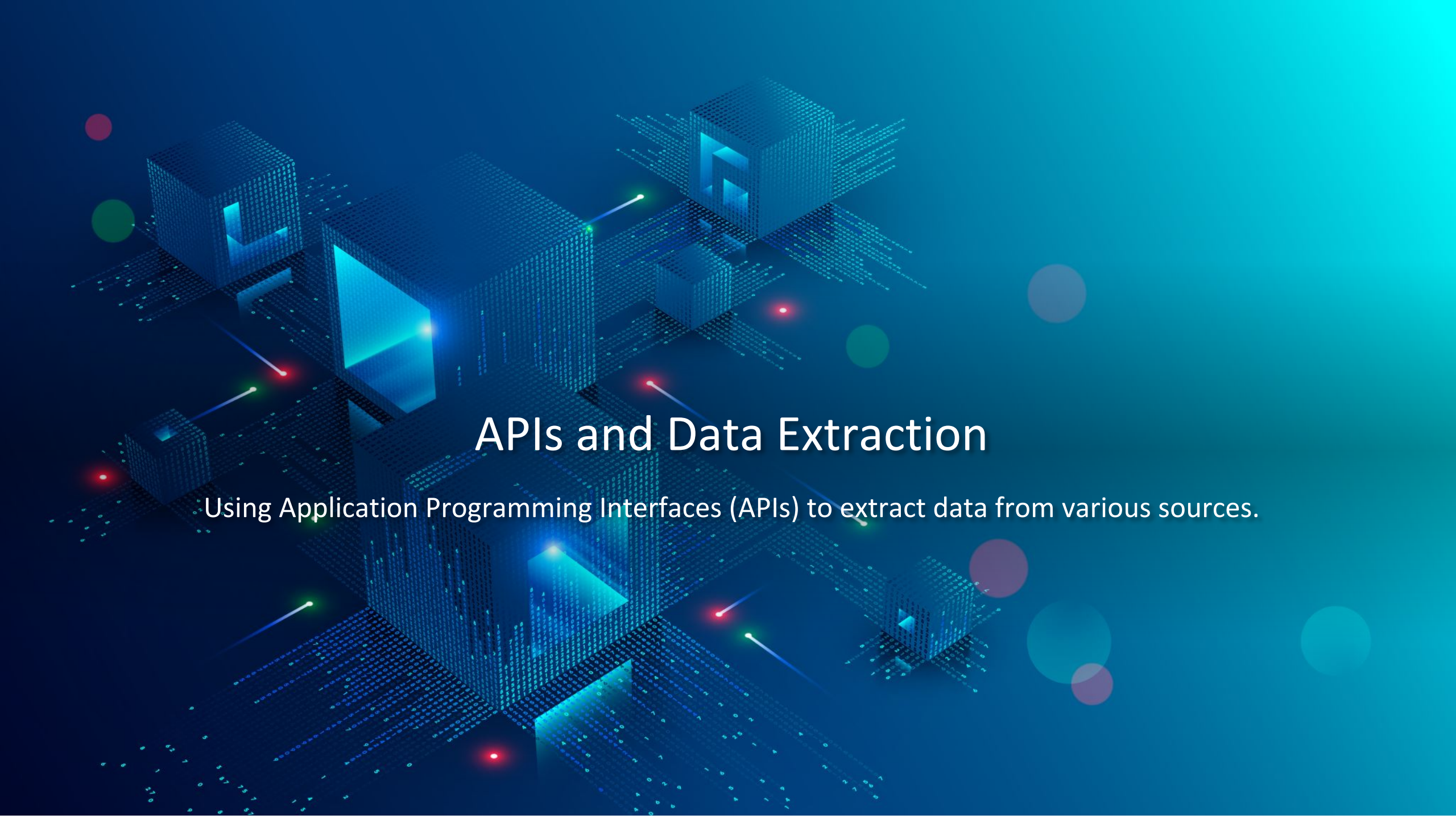
Web Scraping

- The automated extraction of data from websites.



Characteristics of Web Scraping

- **Automated Extraction**: Using bots or scripts to collect data from web pages.
- **Unstructured Data**: Often involves extracting data from unstructured or semi-structured sources.
- **Dynamic Content**: Can handle dynamic and frequently updated content.



APIs and Data Extraction

Using Application Programming Interfaces (APIs) to extract data from various sources.

Characteristics of APIs

- **Standardized Access**: Provides a standard way to access and retrieve data.
- **Real-Time or Batch**: Can be used for both real-time and batch data extraction.
- **Secure Access**: Often requires authentication and authorization.

Data Modeling Concepts

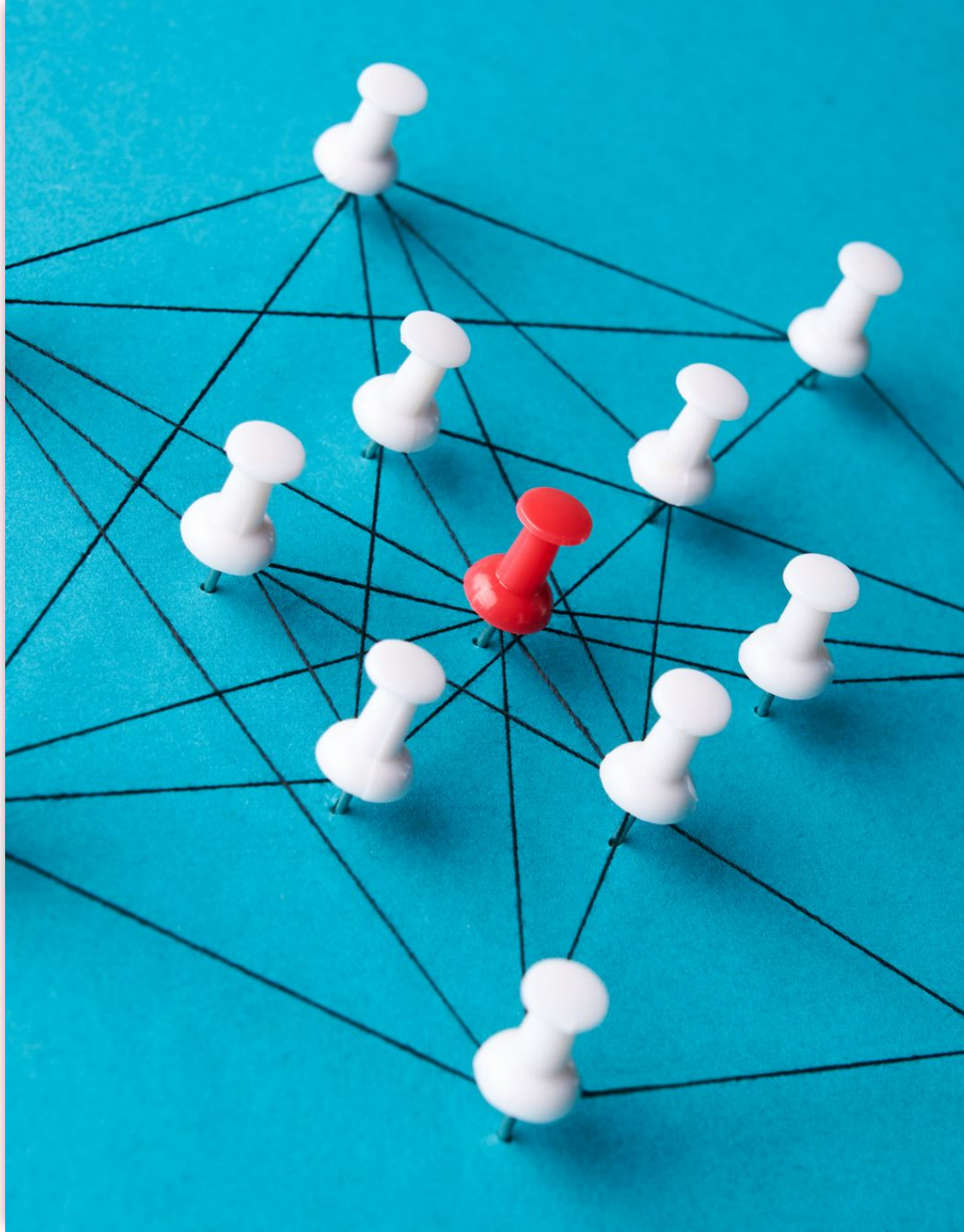
- E-R Diagrams
- Normalization
- Denormalization

E-R Diagrams (Entity-Relationship Diagrams)

- A graphical representation of entities and their relationships to each other within a database.

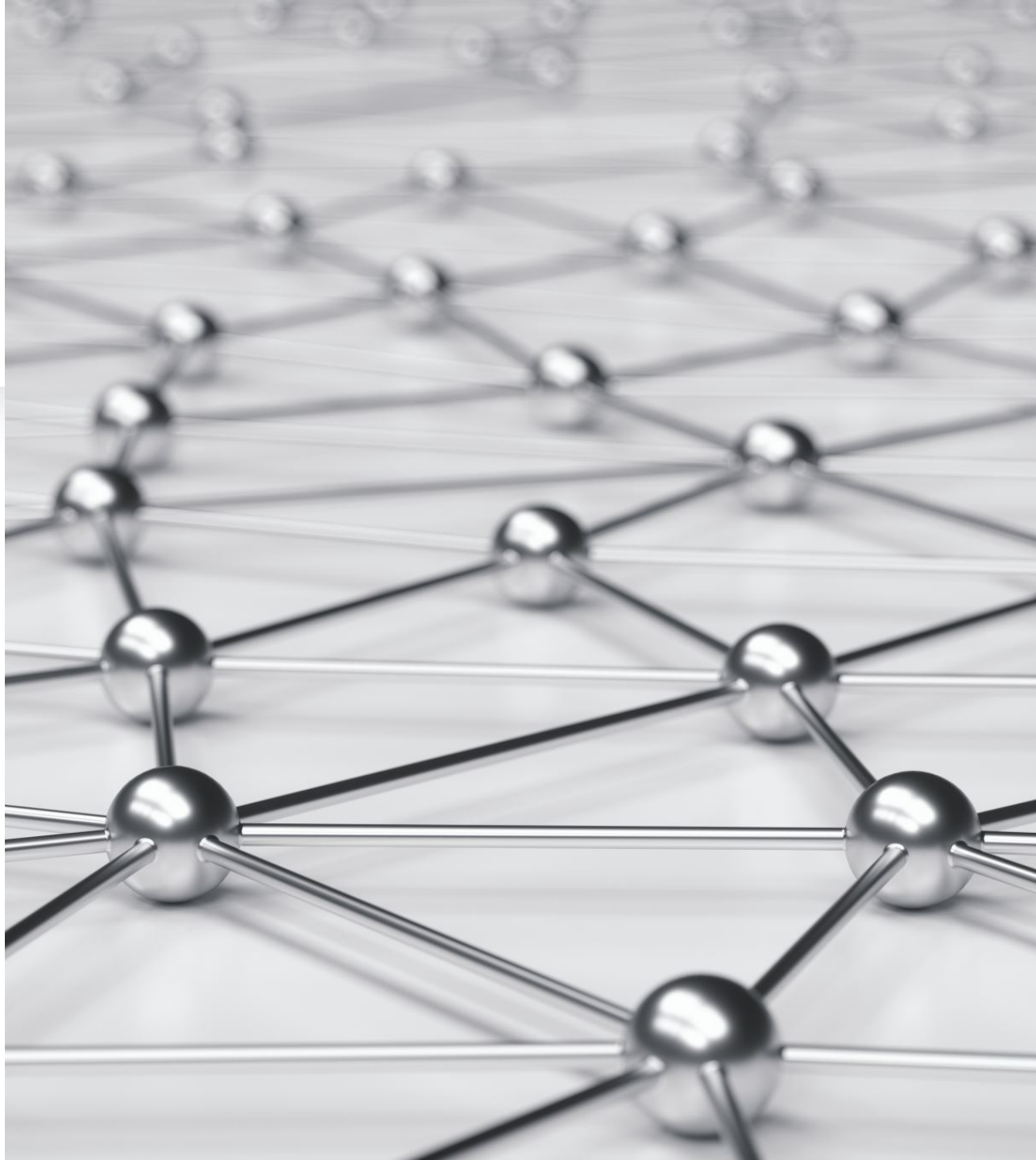
Components of E-R Diagrams

- **Entities**: Objects or concepts that can have data stored about them (e.g., Customer, Order).
- **Attributes**: Properties or details of entities (e.g., Customer Name, Order Date).
- **Relationships**: Connections between entities (e.g., Customers place Orders).



Types of Relationships

- **One-to-One**: Each entity in the relationship will have exactly one related entity.
- **One-to-Many**: One entity can have multiple related entities.
- **Many-to-Many**: Multiple entities can have multiple related entities.



Normalization

- The process of organizing data in a database to reduce redundancy and improve data integrity.

Goals of Normalization

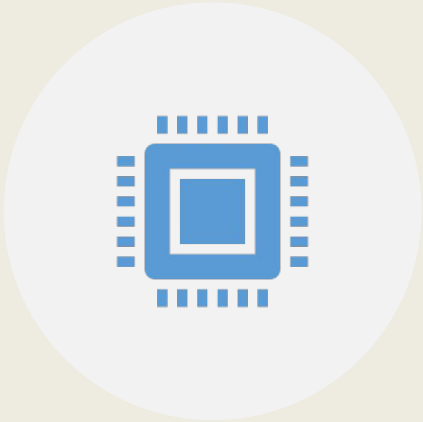
- **Eliminate Redundancy**: Reduce duplicate data.
- **Ensure Data Integrity**: Maintain consistent and accurate data.
- **Simplify Data Structure**: Make the database more flexible and easier to maintain.

Normal Forms

- **First Normal Form (1NF)**: Ensure each column contains atomic values, and each column has unique values.
- **Second Normal Form (2NF)**: Meet 1NF requirements and ensure that all non-key attributes are fully functionally dependent on the primary key.
- **Third Normal Form (3NF)**: Meet 2NF requirements and ensure that all attributes are only dependent on the primary key.

Advantages of Normalization

DATA INTEGRITY: MINIMIZES DATA ANOMALIES AND INCONSISTENCIES.



EFFICIENT STORAGE: REDUCES DATA REDUNDANCY, SAVING STORAGE SPACE.

IMPROVED QUERY PERFORMANCE: SIMPLIFIES COMPLEX QUERIES BY REDUCING THE NUMBER OF JOINS REQUIRED.

Denormalization

- The process of combining normalized tables to improve read performance at the cost of write performance and data redundancy.

When to Use Denormalization

- **Performance Optimization**: When read performance is critical, and normalized data requires complex joins.
- **Reporting and Analysis**: When the database is primarily used for reporting and data analysis.

Methods of Denormalization

Combining Tables: Merging tables that are frequently joined together.

Adding Redundant Data: Including duplicate data to avoid joins and improve read performance.

Precomputed Aggregations: Storing summary data to speed up query performance.

Advantages and Disadvantages of Denormalization

Advantages:

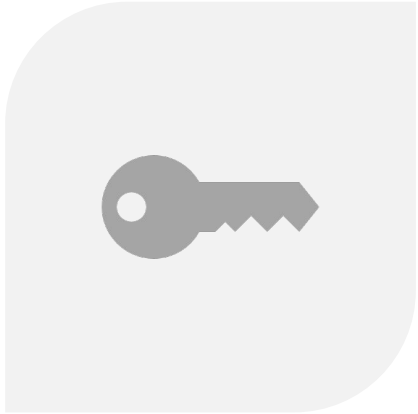
- Improved read performance.
- Simplified query structure.

Disadvantages:

- Increased data redundancy.
- Potential for data anomalies and inconsistencies.
- Increased storage requirements.

Data Storage: Relational Databases

SQL DATABASES

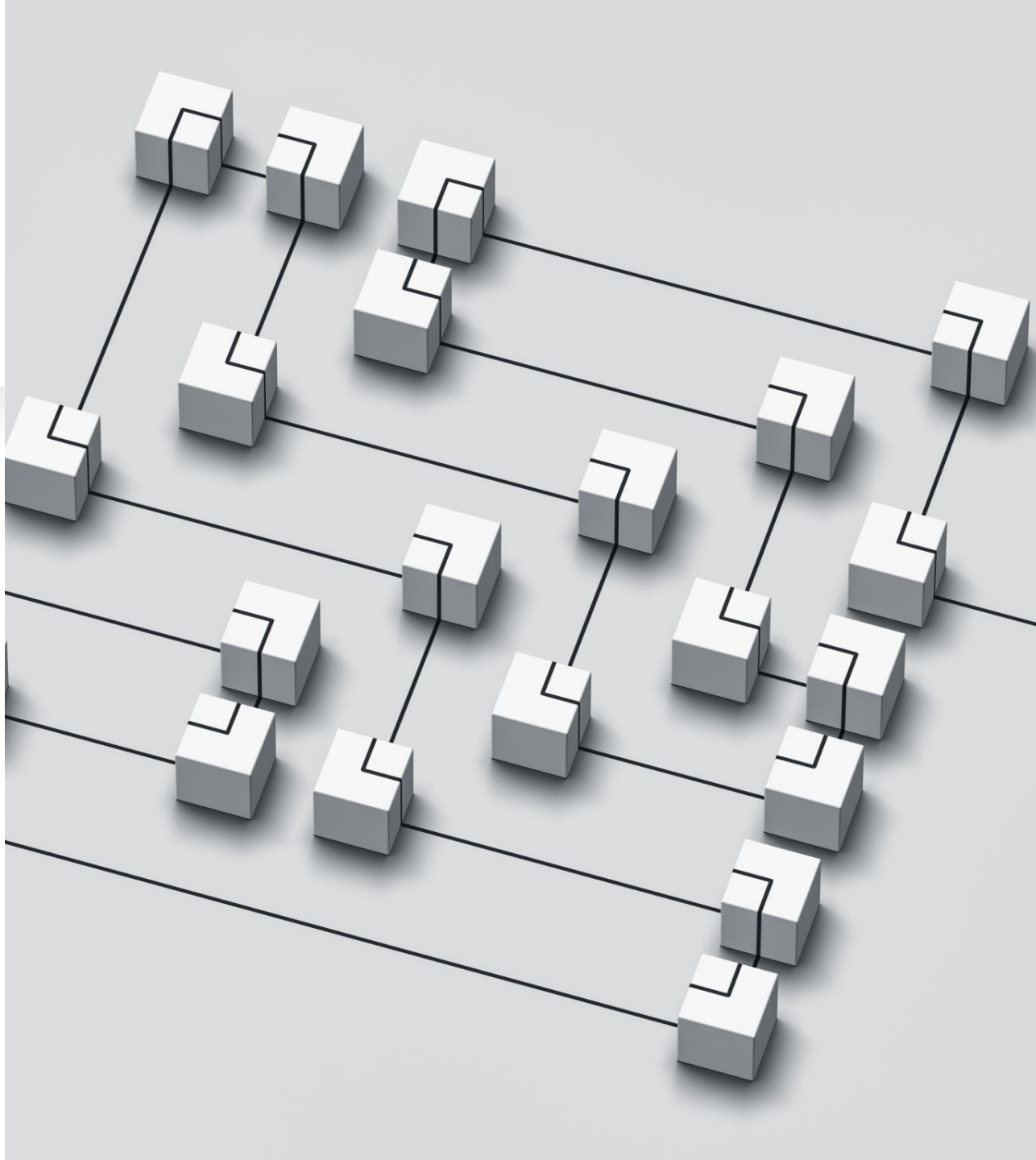


KEY CHARACTERISTICS

USE CASES

SQL Databases

- A type of database that uses Structured Query Language (SQL) for defining, manipulating, and querying data.



Key Characteristics

- Data is organized into tables, each with rows and columns.
- Example: Customer table with columns for CustomerID, Name, Address, etc.

Use Cases

- Applications that require reliable transaction processing.
- Example: Banking systems, e-commerce platforms.

Data Storage: NoSQL Databases

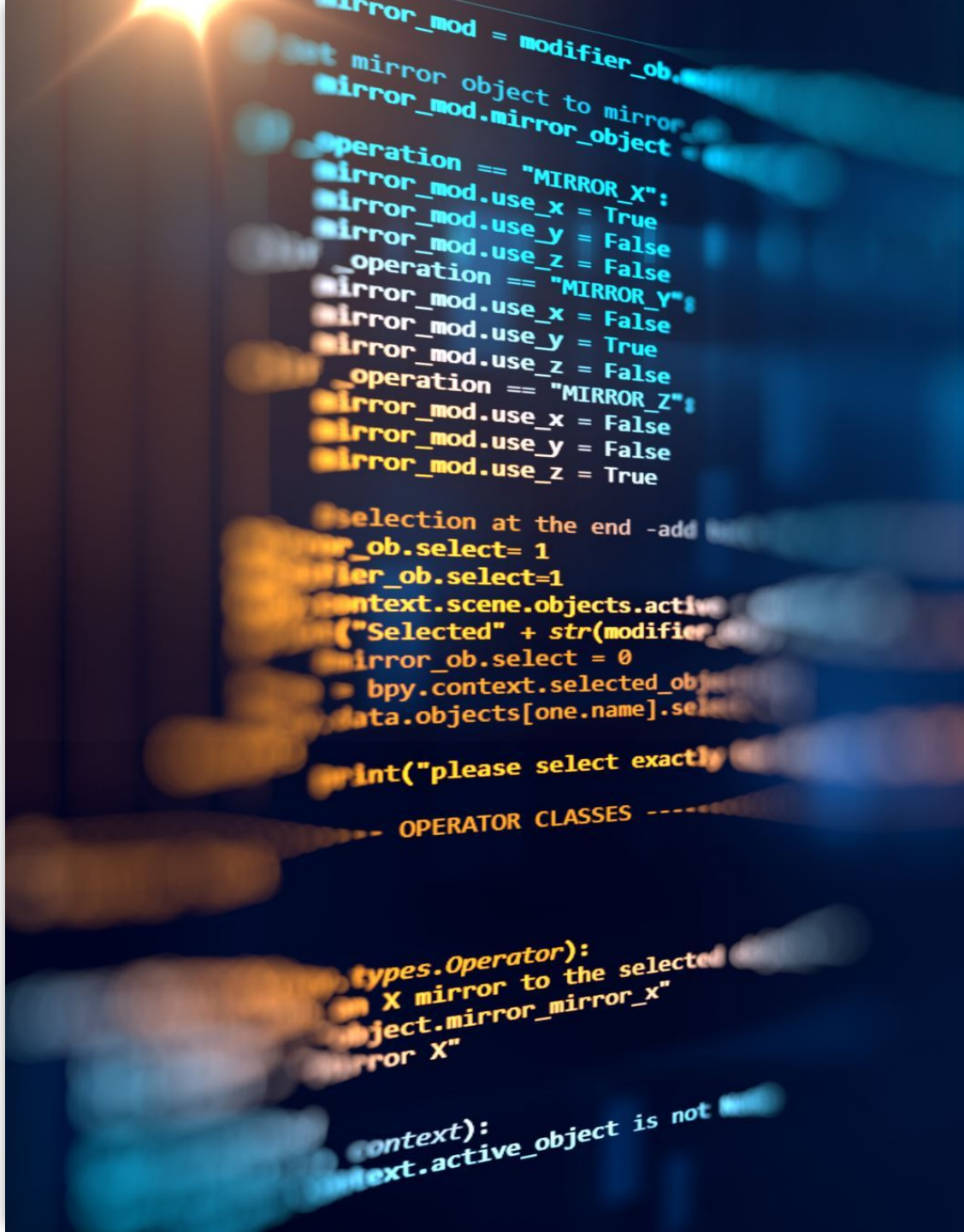
TYPES (DOCUMENT, KEY-VALUE, COLUMN, GRAPH)

KEY CHARACTERISTICS

USE CASES

Document Databases

- Store data in documents similar to JSON (JavaScript Object Notation).
- **Example**: MongoDB, CouchDB.



Key-Value Stores

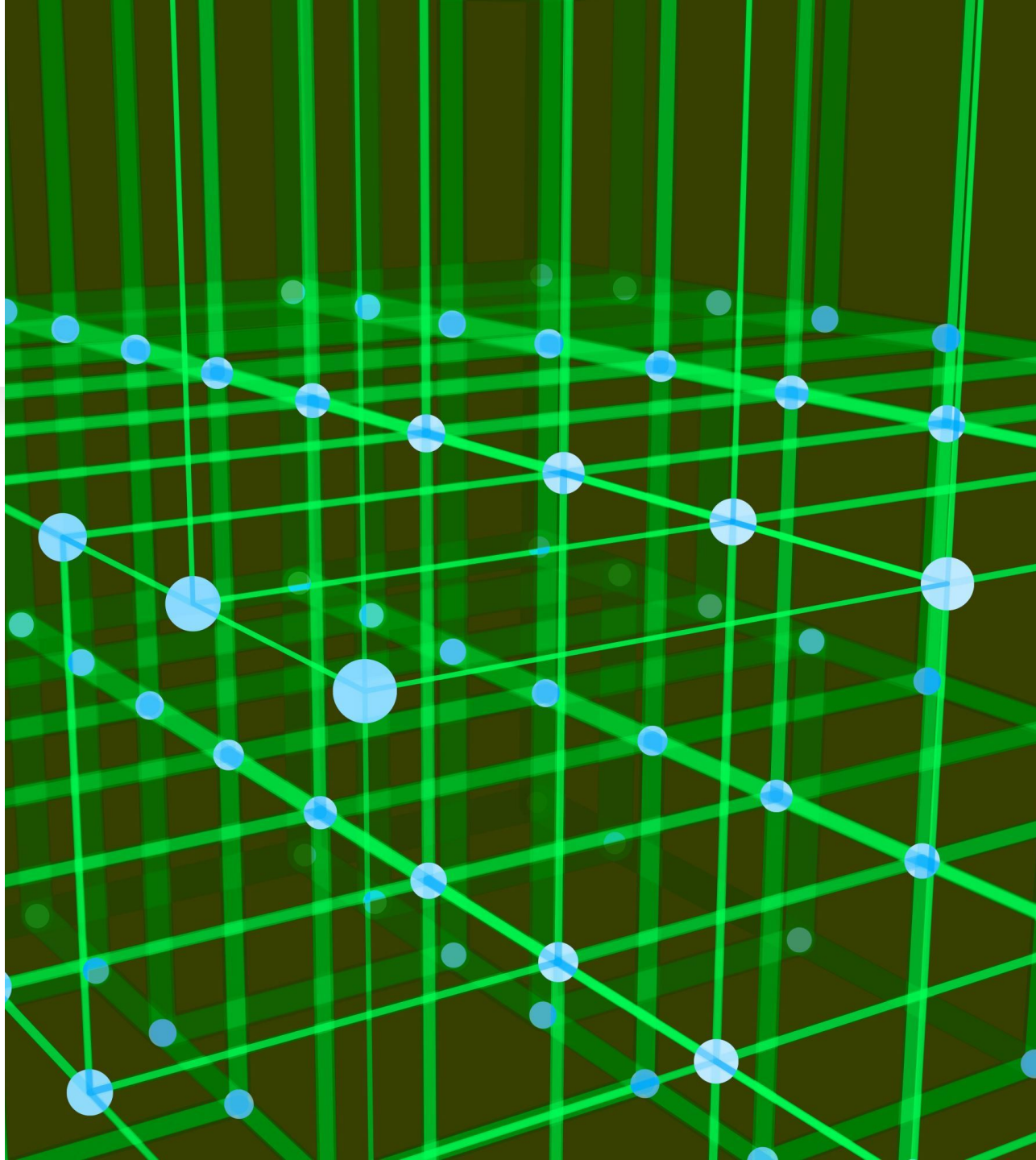
- Store data as a collection of key-value pairs.
- **Example**: Redis, DynamoDB.

Column-Family Stores

- Store data in columns rather than rows, optimized for read and write performance.
- **Example**: Apache Cassandra, HBase.

Graph Databases

- Store data in nodes, edges, and properties, ideal for data with complex relationships.
- **Example**: Neo4j, Amazon Neptune.



Key Characteristics

- Schema Flexibility
- Horizontal Scalability
- High Availability and Fault Tolerance
- Distributed Architecture
- Performance Optimization

Use Cases

- **Real-Time Big Data Applications**
- **Description**: Applications requiring real-time data processing and high throughput.
- **Example**: Analytics platforms, IoT applications.

Data Storage: Data Warehouses

DEFINITION

IMPORTANCE

EXAMPLES

Data Warehouses

- A data warehouse is a centralized repository for storing large volumes of structured data from multiple sources. It is designed for query and analysis rather than transaction processing.

Importance

Centralized Data Repository: Provides a single source of truth for an organization’s data.

Improved Data Quality: Standardizes and cleanses data from multiple sources.

Enhanced Business Intelligence: Facilitates complex queries and data analysis to support decision-making.

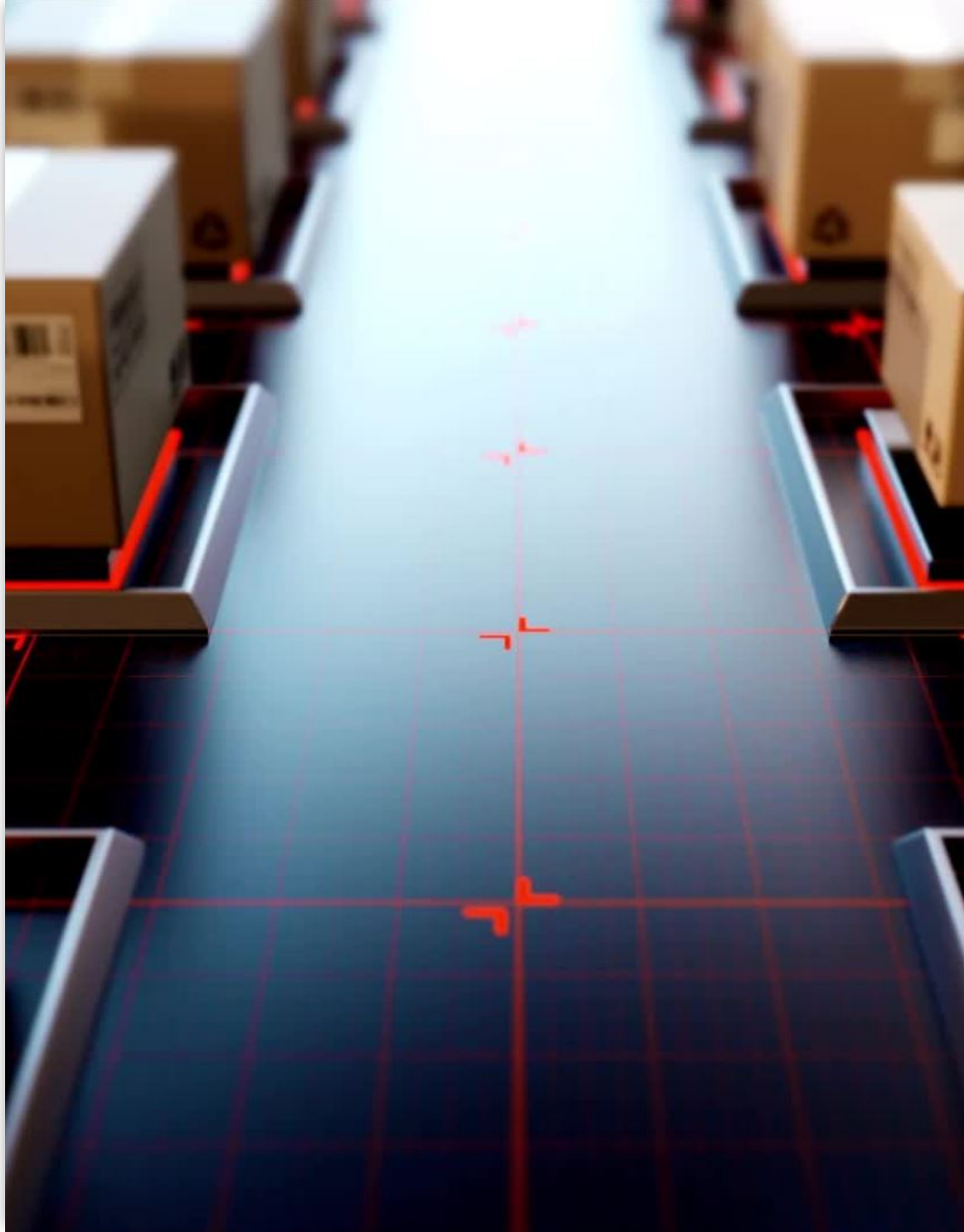
Examples

Amazon Redshift

- Description: Fully managed data warehouse service in the cloud.
- Key Features: Scalability, high performance, integration with AWS ecosystem.

Google BigQuery

- Description: Serverless, highly scalable, and cost-effective multi-cloud data warehouse.
- Key Features: Real-time analytics, machine learning integration, seamless data querying.



Data Processing and Transformation

ETL (EXTRACT, TRANSFORM, LOAD) PROCESSES

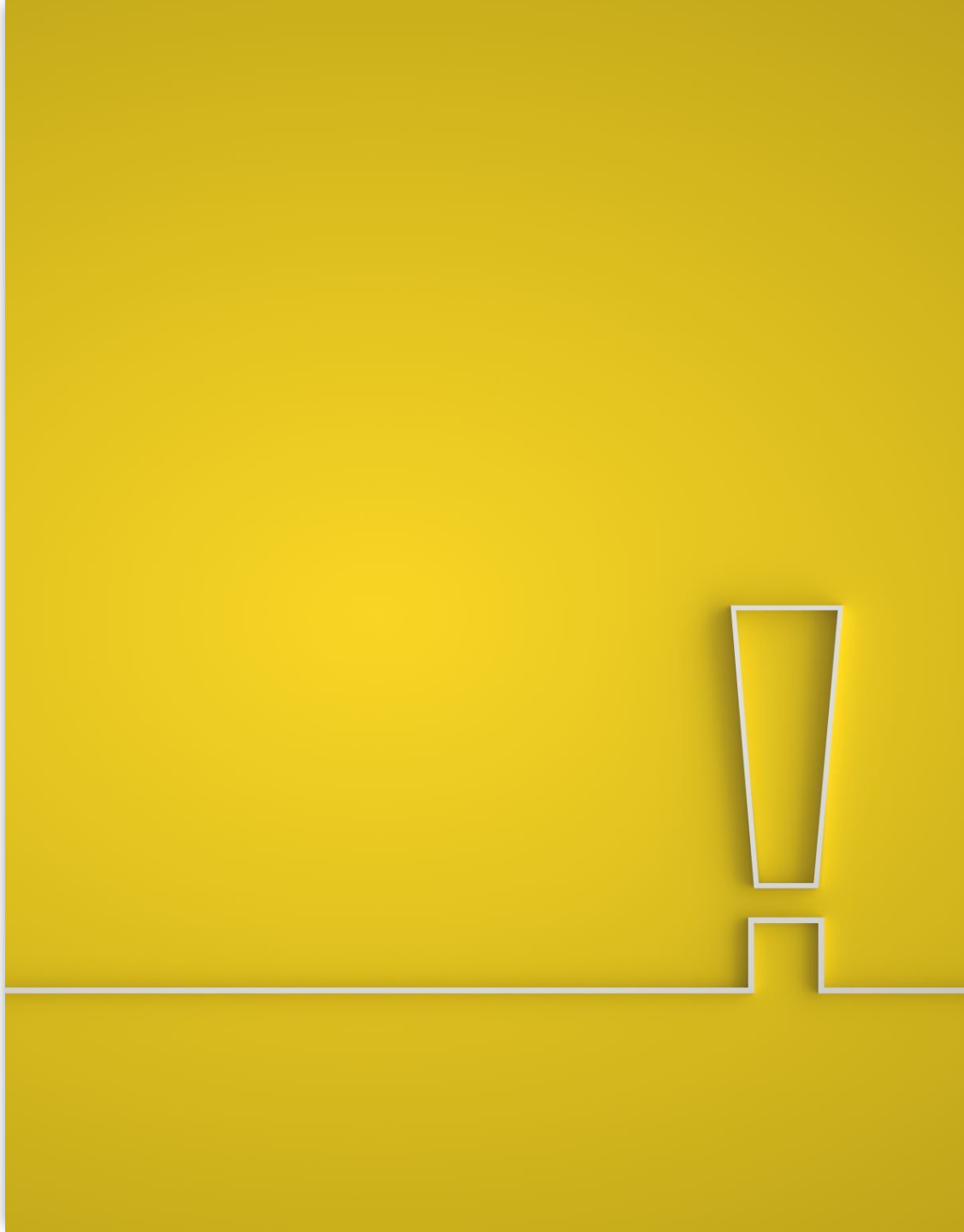
DATA CLEANING TECHNIQUES

ETL (Extract, Transform, Load) Processes

- ETL is a process that involves extracting data from various sources, transforming it into a suitable format, and loading it into a target system, such as a data warehouse.

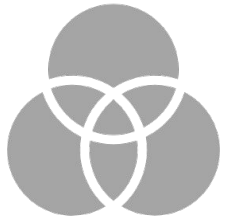
Data Cleaning Techniques

- **Handling Missing Data**: Techniques include deletion, imputation, and using default values.
- **Removing Duplicates**: Identifying and removing duplicate records to prevent redundancy.



Batch Processing vs. Stream Processing

Definitions

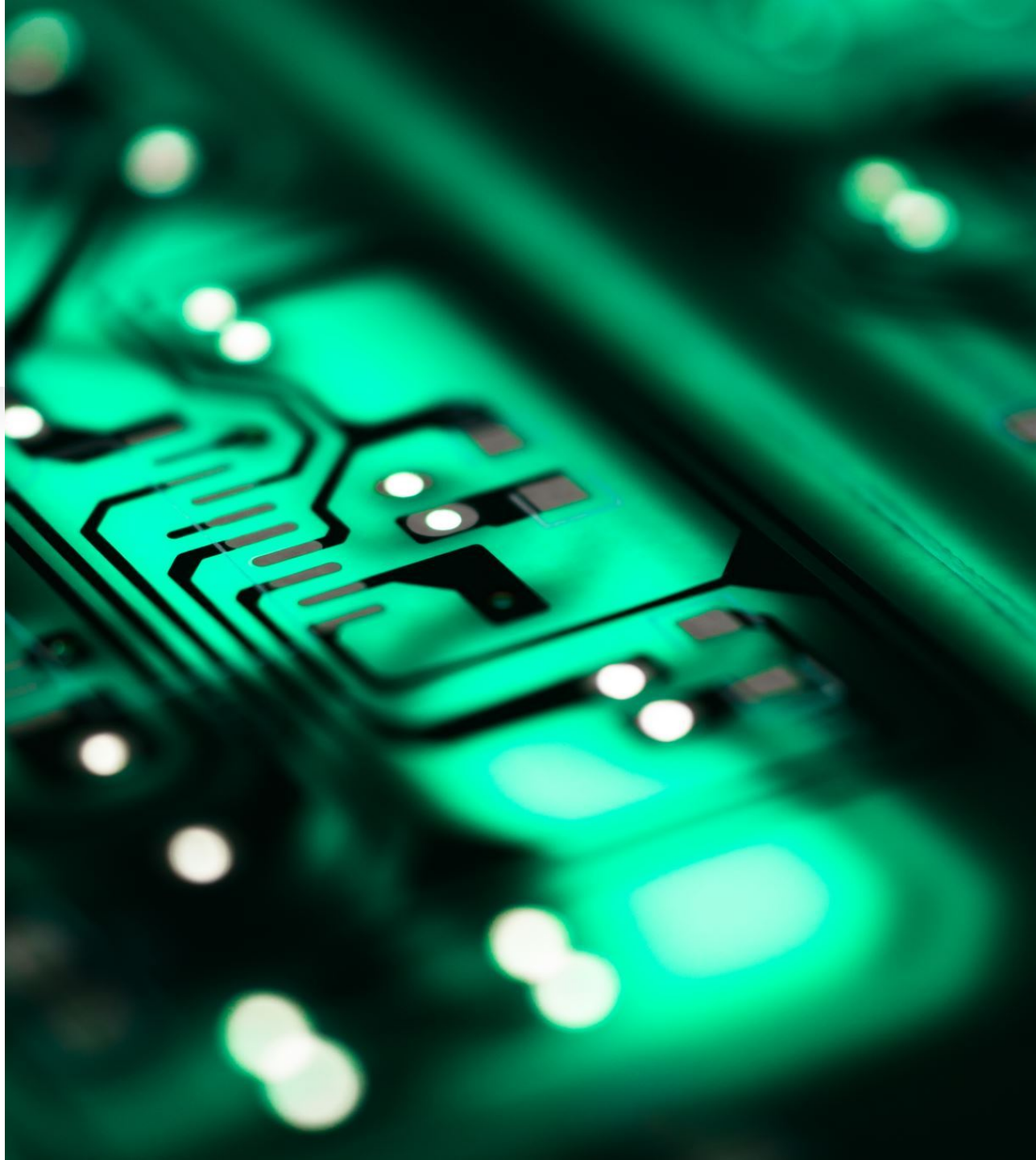


Differences

Use Cases

Batch Processing

- Batch processing is the execution of a series of jobs in a program on a computer without manual intervention. Data is collected, processed, and stored in batches at scheduled intervals.



Stream Processing

- Stream processing involves the continuous ingestion and processing of data in real-time as it is generated or received.

Differences Between Batch and Stream Processing

Processing Model:

Batch Processing: Processes data in bulk at scheduled times.

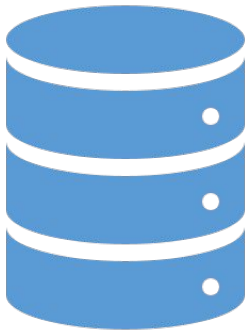
Stream Processing: Processes data continuously in real-time.

Latency:

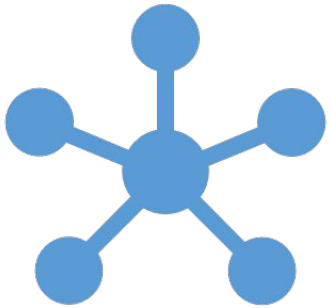
Batch Processing: High latency due to scheduled intervals.

Stream Processing: Low latency, near real-time.

Data Integration



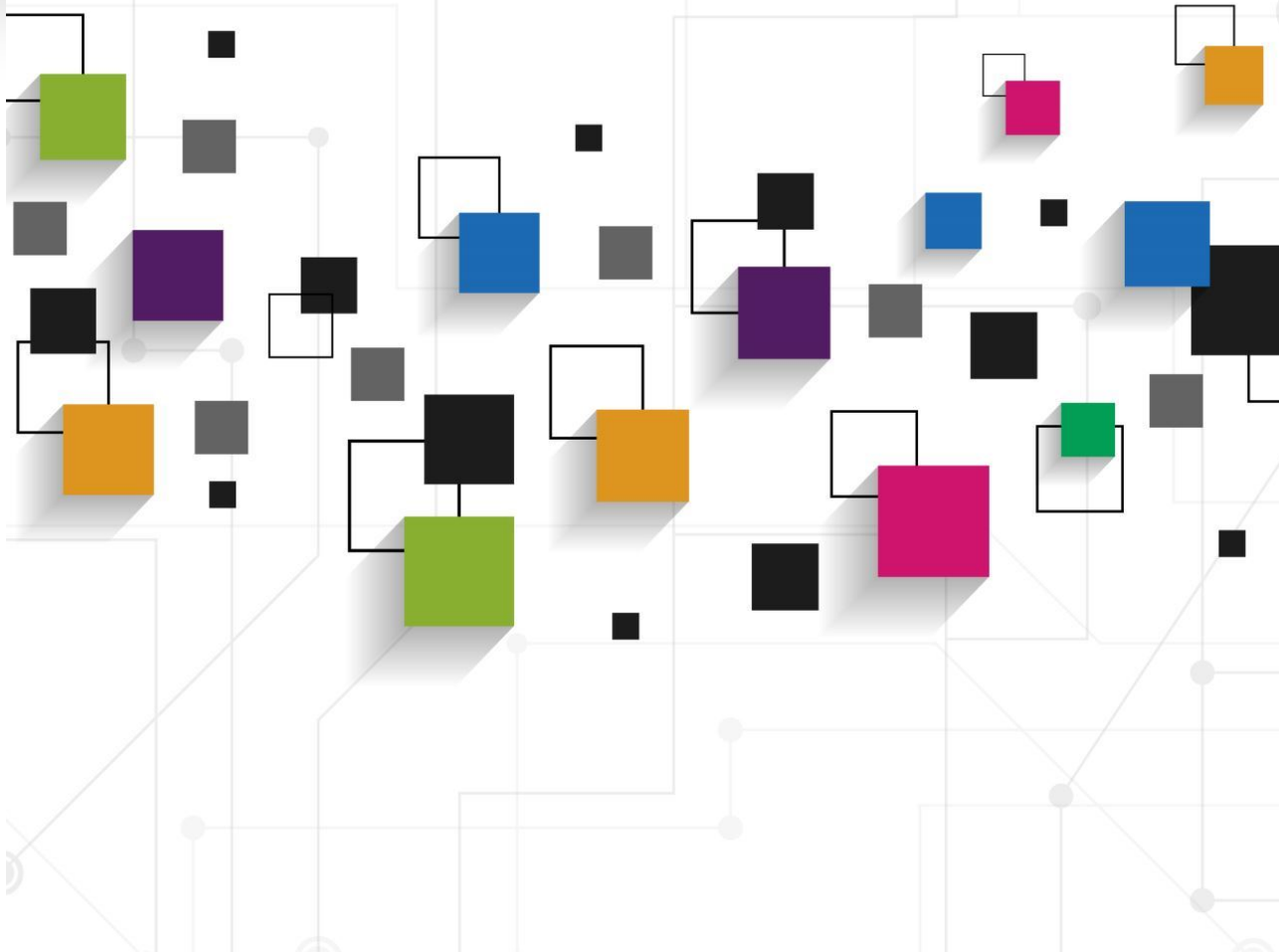
Combining Data from Multiple Sources



Importance of Data Integration

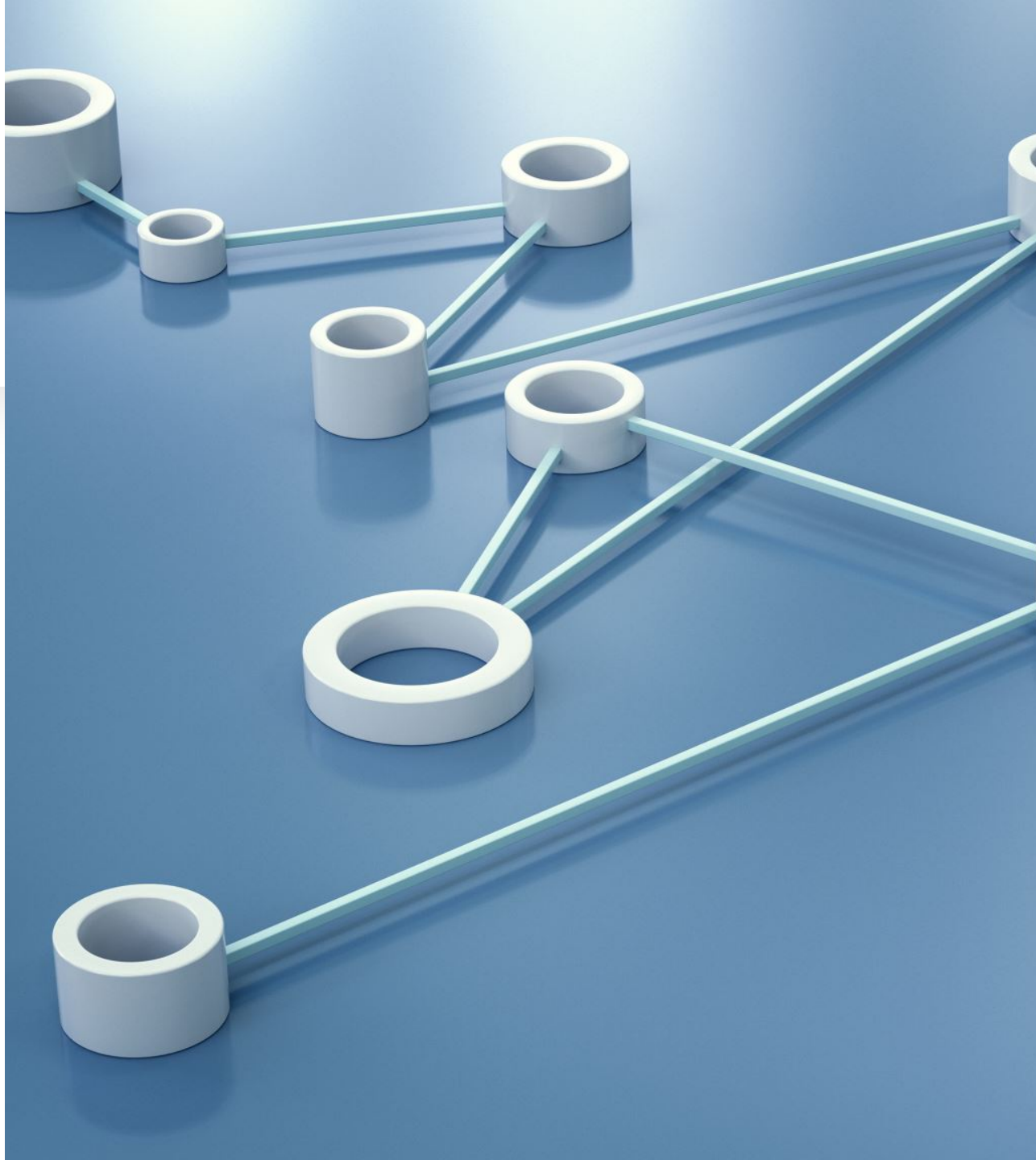
Data Integration

- Data integration is the process of combining data from different sources to provide a unified view. It involves consolidating data, harmonizing formats, and ensuring consistency across multiple data sources.



Importance of Data Integration

- **Unified View**: Provides a single view of data from disparate sources, improving decision-making.
- **Data Consistency**: Ensures consistency and accuracy across different datasets.



Data Integration Tools

Data Integration Tools

Use Cases

Tools for Data Integration

- **Microsoft SQL Server Integration Services (SSIS)**
- **Description**: A platform for data integration and workflow applications.
- **Features**: Data extraction, transformation, and loading, supports complex workflows.
- **Use Cases**: ETL processes, data warehousing, and data migration.

Data Quality and Consistency

Importance

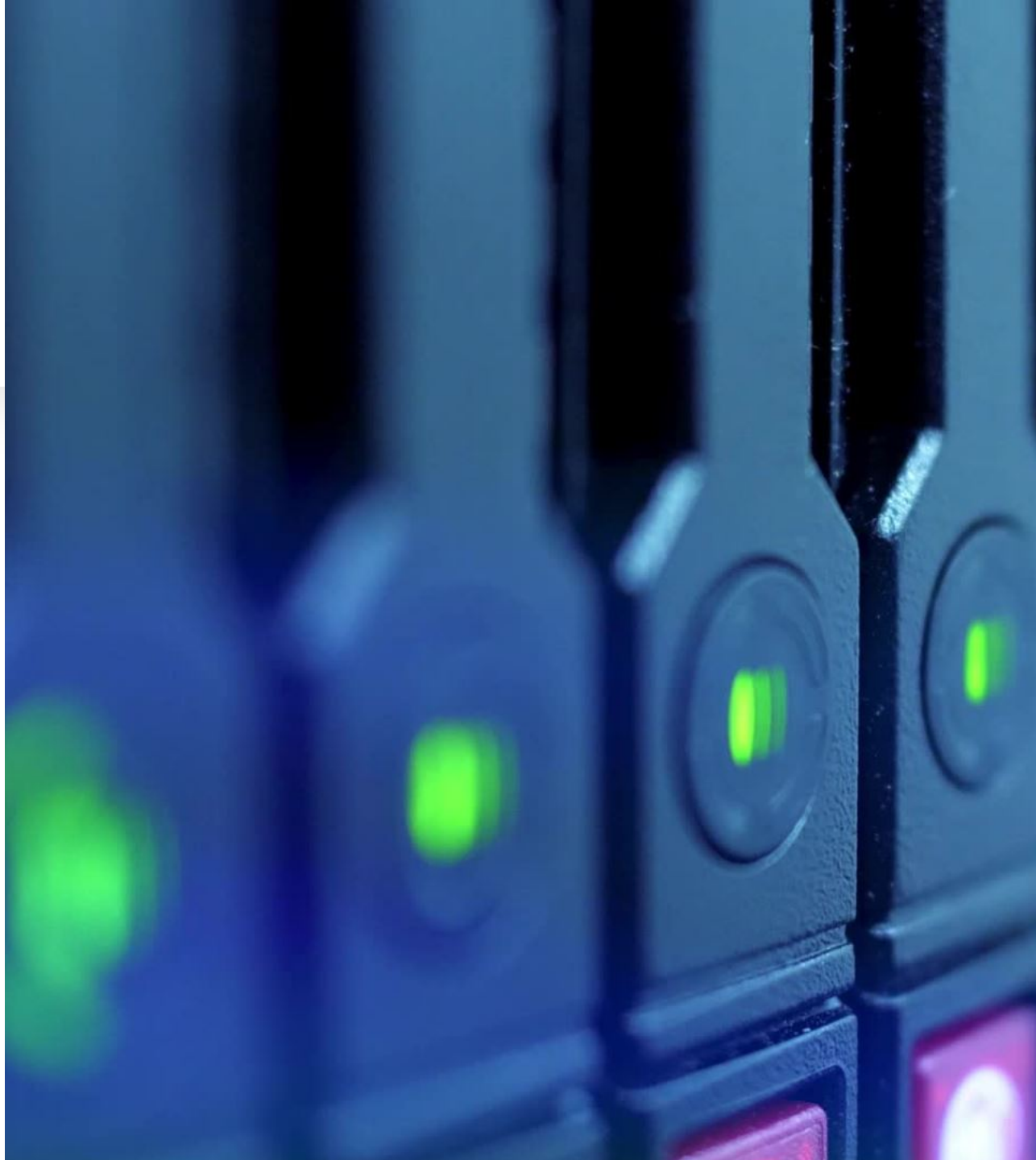
Techniques to Ensure Quality and Consistency

Importance of Data Quality

- **Informed Decision-Making**: Accurate data ensures that decisions are based on reliable information.
- **Operational Efficiency**: Reduces errors and improves process efficiency.

Data Quality Tools

- Talend Data Quality
- Informatica Data Quality
- OpenRefine.



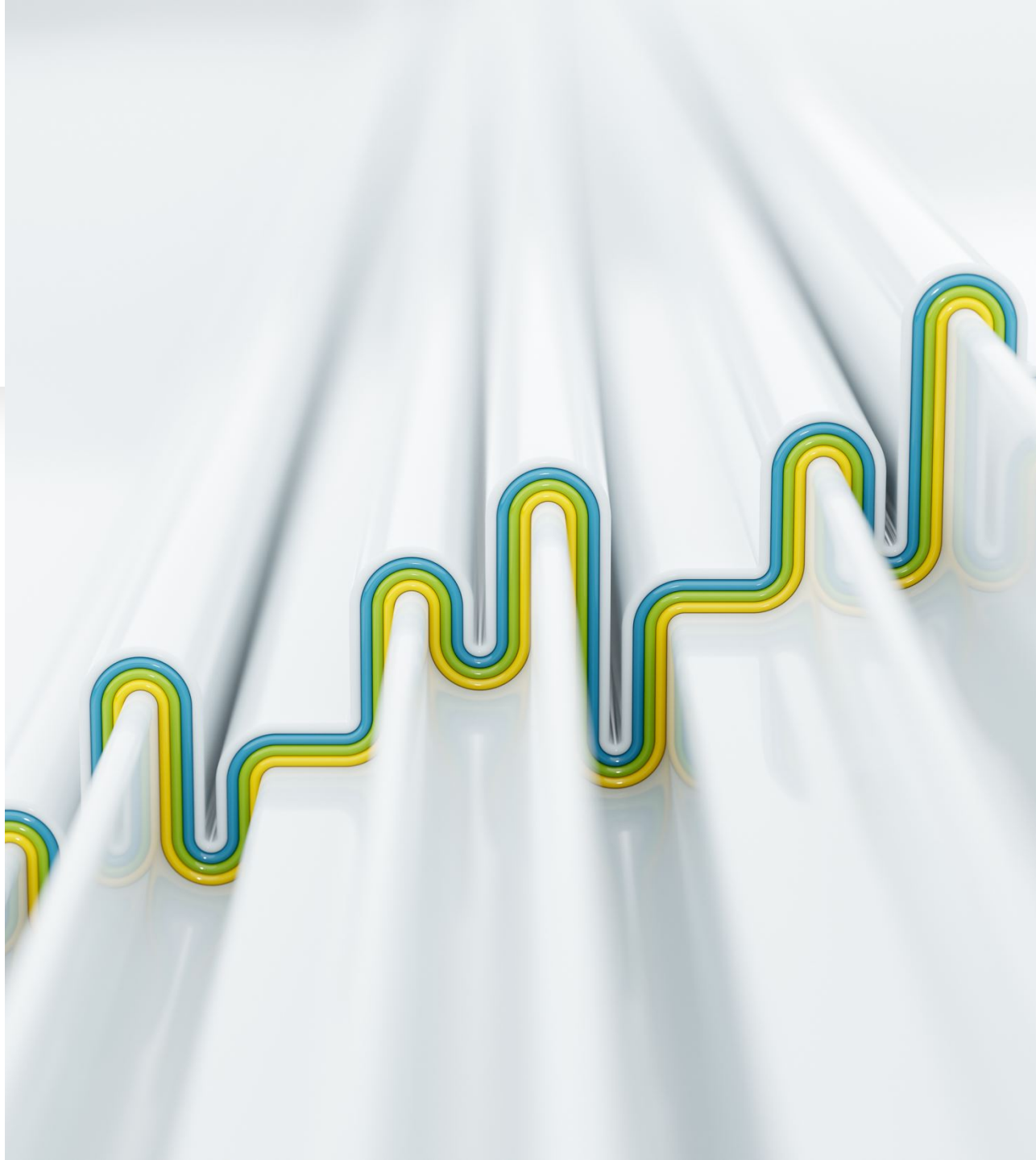
Designing Data Pipelines

Concepts

Best Practices

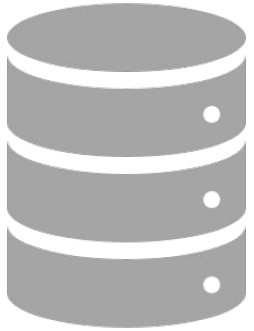
What is a Data Pipeline?

- A data pipeline is a series of processes that automates the collection, transformation, and storage of data from various sources to a destination for analysis and reporting.



Implementing Data Pipelines

Tools and Technologies



Workflow Orchestration Tools (e.g., Apache Airflow)

Tools for Designing and Managing Data Pipelines

Apache Airflow

Description: An open-source tool for scheduling and managing complex data workflows.

Features: DAGs (Directed Acyclic Graphs), extensibility, monitoring.

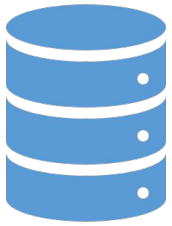
Use Cases: Workflow orchestration, ETL job scheduling.

Monitoring and Maintaining Data Pipelines

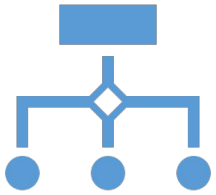
Importance

Techniques and Tools

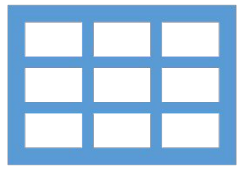
Importance of Data Pipelines



Efficiency: Automates repetitive data tasks, reducing manual effort.



Integration: Combines data from disparate sources into a unified system.

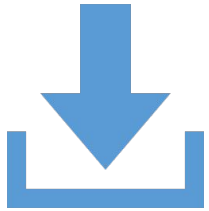


Scalability: Handles large volumes of data efficiently.

Data Visualization with Matplotlib

Basic Plots: Line, Bar, Histogram, Scatter

Customizing Plots: Titles, Labels, Legends, Annotations



Saving and Exporting Visualizations

What is Matplotlib?

Definition: Matplotlib is a widely-used Python library for creating static, interactive, and animated visualizations in Python.

Features: Versatile plotting capabilities, integration with Pandas and NumPy.

Importance of Data Visualization

Insight: Helps in understanding data patterns, trends, and anomalies.

Communication: Effective tool for presenting data findings to stakeholders.

Exploration: Assists in data exploration and hypothesis testing.

Data Visualization with Seaborn

- Distribution Plots: Histograms, KDE Plots
- Categorical Plots: Bar Plots, Count Plots, Box Plots, Violin Plots
- Relational Plots: Scatter Plots, Line Plots

What is Seaborn?

Definition: Seaborn is a Python visualization library based on Matplotlib that provides a high-level interface for drawing attractive and informative statistical graphics.

Features: Built-in themes, color palettes, and easy-to-create complex plots.