THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

Introduction to Big Data and Analytics
CSCI 6444

Big Data Storage and Databases

Prof. Roozbeh Haghnazar

Slides Credit:

Stephen H. Kaisler, D.Sc.and Prof. Roozbeh Haghnazar

OUTLINE – WEEKS 3

- **Final Project**
  - What are the stages and factors?
- **Introduction**
- **Challenges of Big Data Storage**
- **Characteristics Needed for Big Data Storage**
- **Solutions for Big Data Storage**
- **DBMS/BDMS for Different Data Types**
- **Row-Oriented vs Column-Oriented DBMS**
- **Data Lake vs DataBase vs Data Warehouse**
- **Introduction to Hadoop and HDFS**
- **Data Warehousing**

Final Project

Prof. Roozbeh Haghnazar

CRITERIA

- Big Data Dataset
- Target some challenges
- Have a pipeline
  - Storage
  - ETL
  - Data Modeling
  - Analyze (Predictive and Descriptive)
  - Visualize

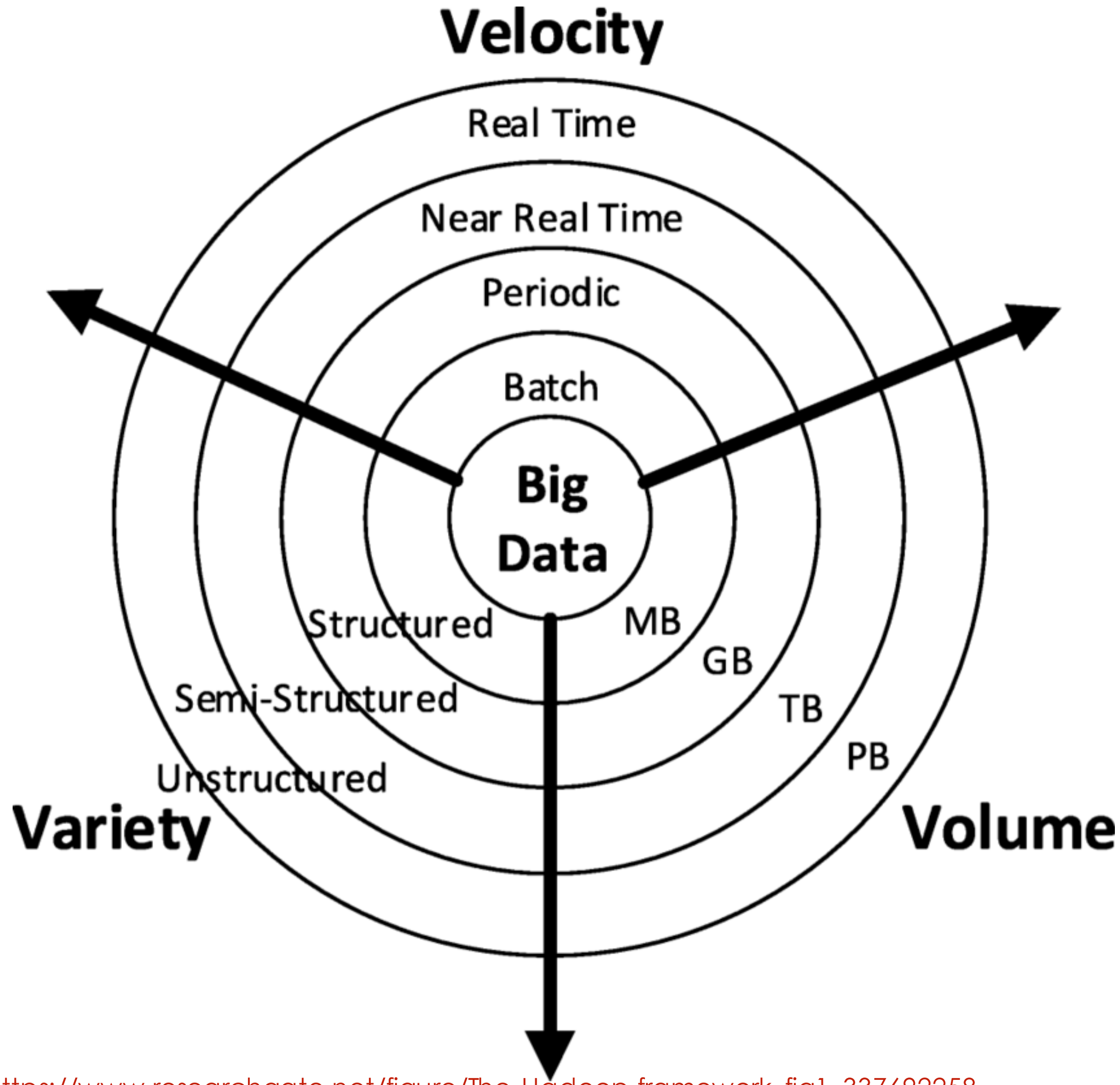
Introduction

Prof. Roozbeh Haghnazar

Objectives of Today's Session

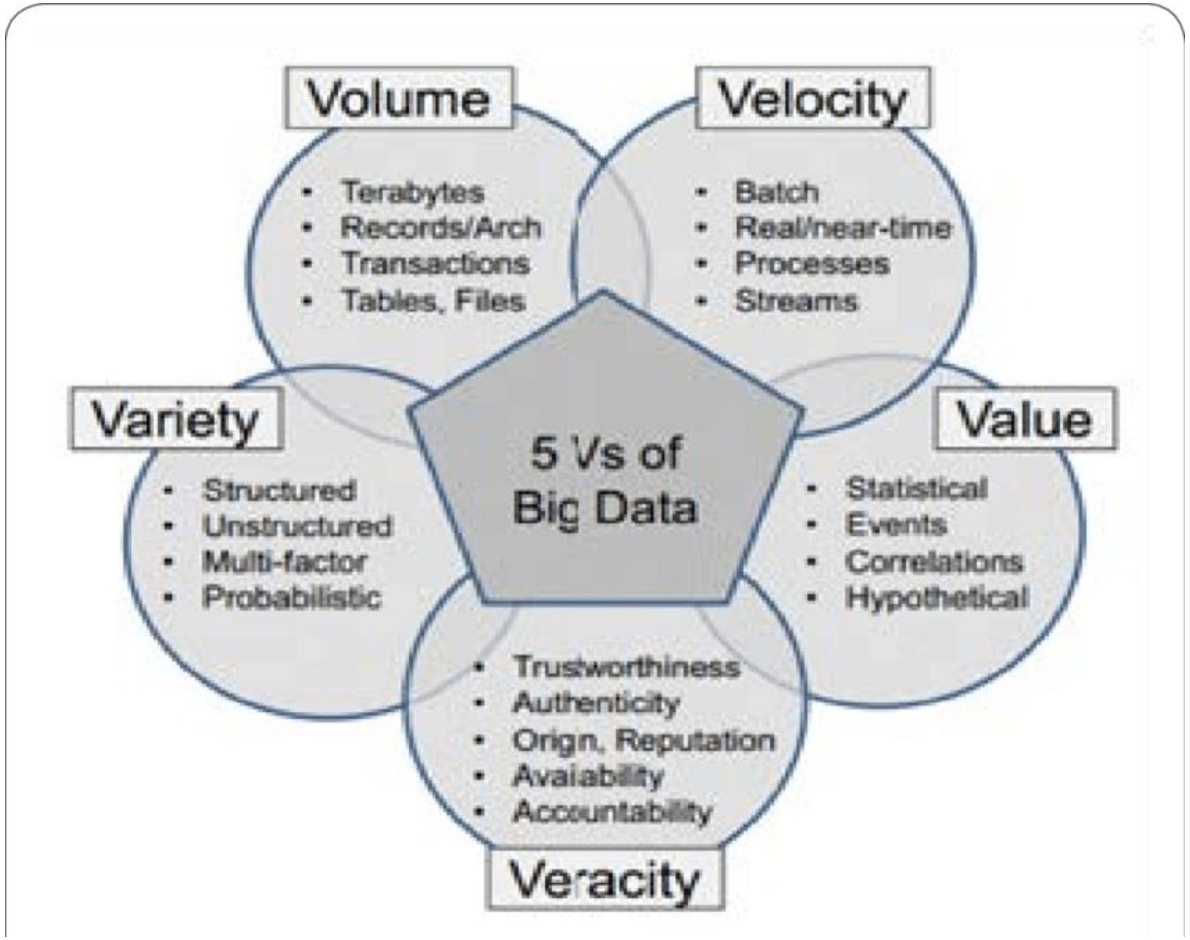
- Understand the challenges of Big Data storage.
- Learn about the characteristics necessary for effective Big Data storage.
- Differentiate between various DBMS/BDMS tailored for different data types.
- Explore the differences between row-oriented and column-oriented storage.
- Distinguish between Data Lakes, Databases, and Data Warehouses.

Overview of the Big Data Landscape



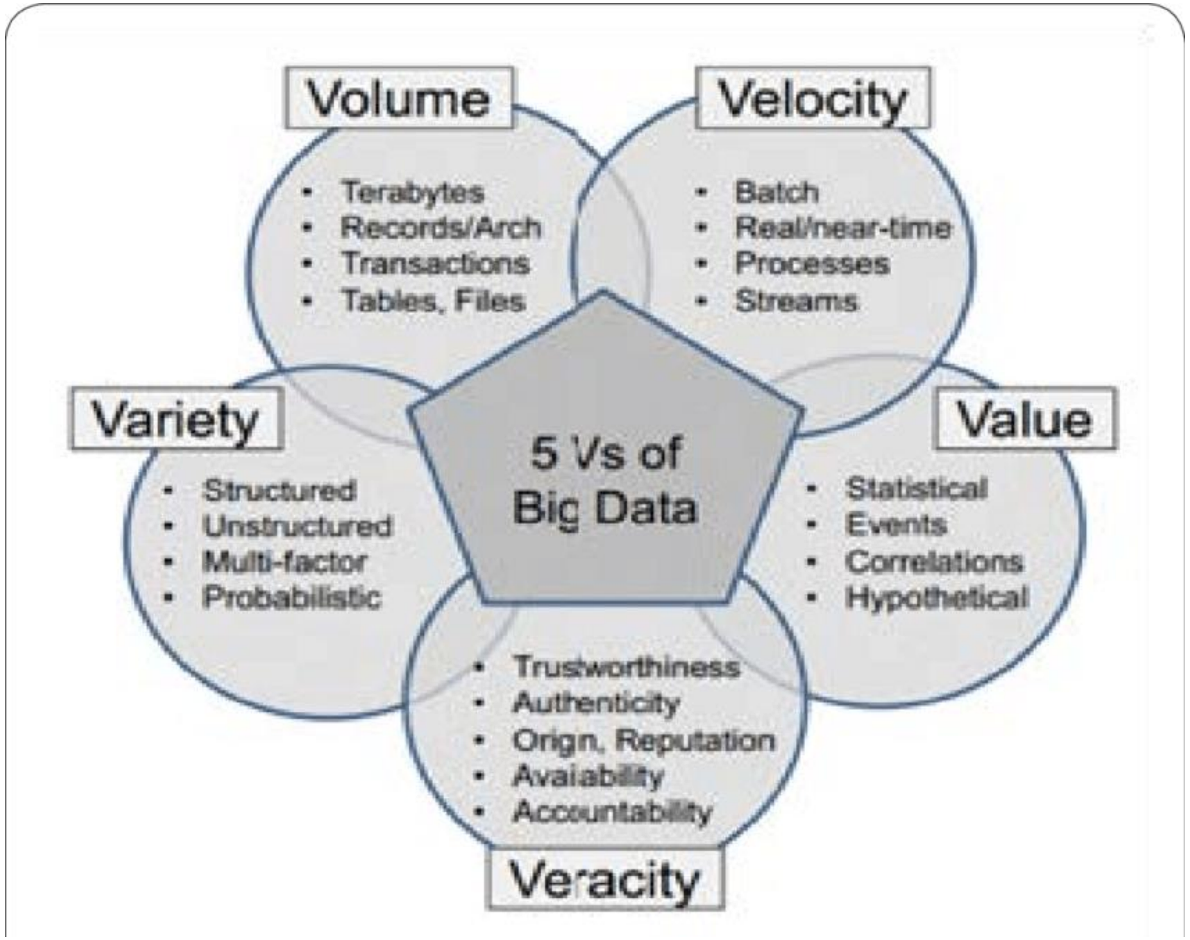
https://www.researchgate.net/figure/The-Hadoop-framework_fig1_337692258

Overview of the Big Data Landscape



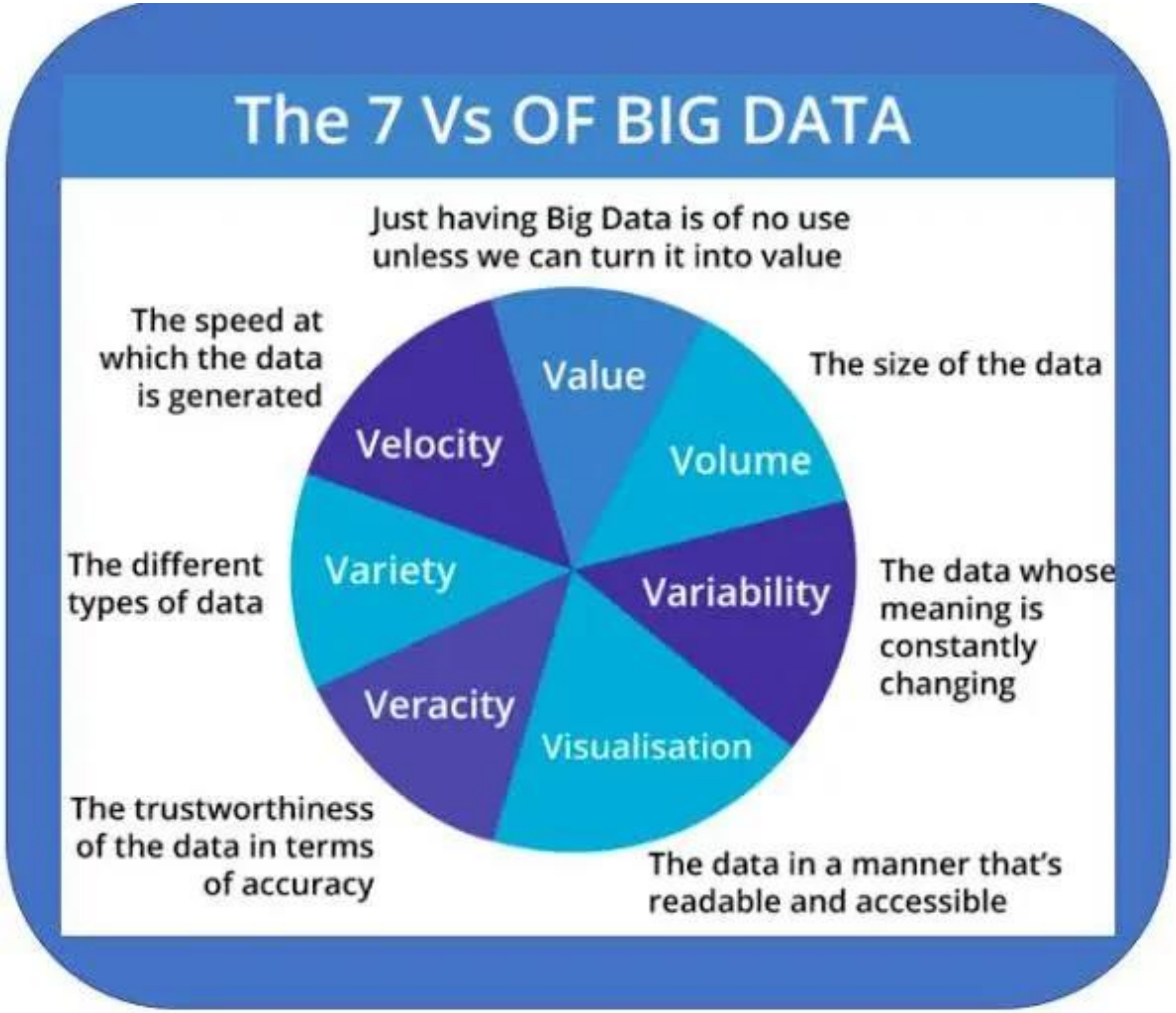
https://www.researchgate.net/figure/5Vs-of-big-data-10_fig1_325103690

Overview of the Big Data Landscape



https://www.researchgate.net/figure/5Vs-of-big-data-10_fig1_325103690

Overview of the Big Data Landscape


The 7 Vs OF BIG DATA
Just having Big Data is of no use unless we can turn it into value
Value
The speed at which the data is generated
Velocity
The size of the data
Volume
The different types of data
Variety
Variability
The data whose meaning is constantly changing
Veracity
Visualisation
The trustworthiness of the data in terms of accuracy
The data in a manner that's readable and accessible

https://bigdatapath.wordpress.com/2019/11/13/understanding-the-7-vs-of-big-data/

CHALLENGES IN BIG DATA STORAGE

- **Volume**
- **Velocity**
- **Variety**
- **Veracity**
- **Value**
- **Visualization**

VOLUME

- **Volume** refers to the vast amounts of data generated every second. This isn't just from conventional sources like sales transactions or social media posts, but also from sensors, machines, and more.
- **Examples**:
  - Social media platforms: Facebook processes over 500 terabytes of new data every day.
  - IoT: Sensors in various industries generate petabytes of data regularly.
- **Challenges & Implications**:
  - Traditional database systems struggle to handle such huge volumes of data.
  - Storing massive data requires innovative and cost-effective solutions.

VELOCITY

- **Velocity** is about the pace at which new data is generated, and how quickly it needs to be processed and acted upon.
- **Examples**:
  - High-frequency trading systems can generate millions of transactions per second.
  - Streaming services that process real-time data from millions of users simultaneously.
- **Challenges & Implications**:
  - Systems must handle and process data in near-real-time to extract its value.
  - The rapid generation of data might lead to bottlenecks if not processed efficiently.

VARIETY

- **Variety** refers to the diverse types of data. It can come in various formats and structures.
- **Examples**:
  - **Structured data**: Relational databases like SQL, where data is organized into tables, rows, and columns.
  - **Semi-structured data**: JSON or XML files where there is some structure but not as rigid as structured databases.
  - **Unstructured data**: Emails, videos, images, social media posts, etc.
- **Challenges & Implications**:
  - Data from different sources might need to be merged or joined, requiring data integration efforts.
  - Processing and storing diverse data types require flexible and versatile systems.

VERACITY

- **Veracity** deals with the reliability of data. With so much data being generated, it's vital to determine the accuracy and trustworthiness of this data.
- **Examples**:
  - Fake news or reviews on the internet.
  - Sensor data that may occasionally send faulty or inaccurate readings.
- **Challenges & Implications**:
  - Poor data quality can lead to inaccurate insights, making it crucial to have data validation and cleaning processes.
  - Businesses must verify the sources of their data and ensure it's reliable before basing decisions on it.

VALUE

- **Value** refers to our ability to turn our data into value. It's all well and good having access to big data but unless we can turn it into value, it is useless.
- **Examples**:
  - Analyzing customer behavior data to enhance user experience or optimize marketing strategies.
  - Predictive analytics for industries like healthcare or finance to forecast future events.
- **Challenges & Implications**:
  - The need for skilled data scientists and analysts who can transform raw data into actionable insights.
  - The sheer volume of data can sometimes make it challenging to identify what's valuable and what's noise.

VISUALIZATION

- **1. Scale & Volume:**
  - **Challenge**: With vast amounts of data, visualizing everything on a single chart or graph can be overwhelming or simply impossible.
  - **Implication**: Details can be lost, patterns can be hidden, and significant data points might be overlooked.
- **2. Variety of Data Sources:**
  - **Challenge**: Big Data often comes from various sources and can be structured, semi-structured, or unstructured.
  - **Implication**: Integrating and visualizing this diverse data in a unified manner is complicated. Different data sources might require different visualization techniques.
- **3. Real-time Visualization:**
  - **Challenge**: With data continuously streaming in real-time (like stock market data or social media feeds), the visualization needs to be dynamic and update in real-time.
  - **Implication**: Ensuring smooth, continuous, and accurate visualization without lags is technologically challenging.
- **4. Accuracy and Veracity:**
  - **Challenge**: With so much data, ensuring that what's being visualized is accurate and reliable is crucial.
  - **Implication**: Faulty visualizations can lead to incorrect insights and decisions. Data quality plays a massive role in this challenge.
- **5. Complex Relationships:**
  - **Challenge**: Big Data isn't just about individual data points; it's about the relationships between them.
  - **Implication**: Traditional visualization methods might not capture complex interrelations, hierarchies, or networks inherent in the data.
- **6. Computational Efficiency:**
  - **Challenge**: Rendering visualizations for massive datasets can be computationally intensive, especially for interactive visualizations.
  - **Implication**: This can lead to slow performance, crashes, or even inaccurate visual representations if not handled correctly.
- **7. Cognitive Overload:**
  - **Challenge**: Even if we manage to represent Big Data visually, the human brain can only process so much information at once.
  - **Implication**: Overly complex visualizations can lead to confusion, misinterpretation, or simply overwhelm the viewer, making the visualization ineffective.
- **8. Adaptability:**
  - **Challenge**: As data changes and grows, the visualization techniques and tools used need to adapt accordingly.
  - **Implication**: Static visualization solutions can quickly become outdated or irrelevant, necessitating flexible and adaptable visualization approaches.
- **9. Interactivity:**
  - **Challenge**: Modern Big Data visualizations often benefit from being interactive, allowing users to drill down into specifics or view data from different angles.
  - **Implication**: Implementing interactivity without compromising performance or clarity is challenging.

CHARACTERISTICS NEEDED FOR BIG DATA STORAGE

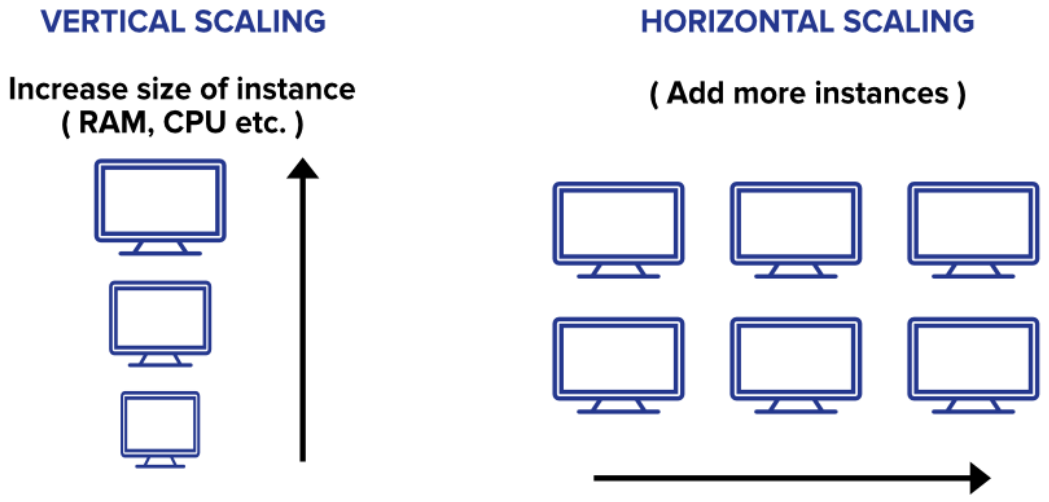
Prof. Roozbeh Haghnazar

Characteristics Needed for Big Data Storage

- Scalability
- Fault Tolerance
- Data Durability
- Distributed Architecture
- Low Latency Access

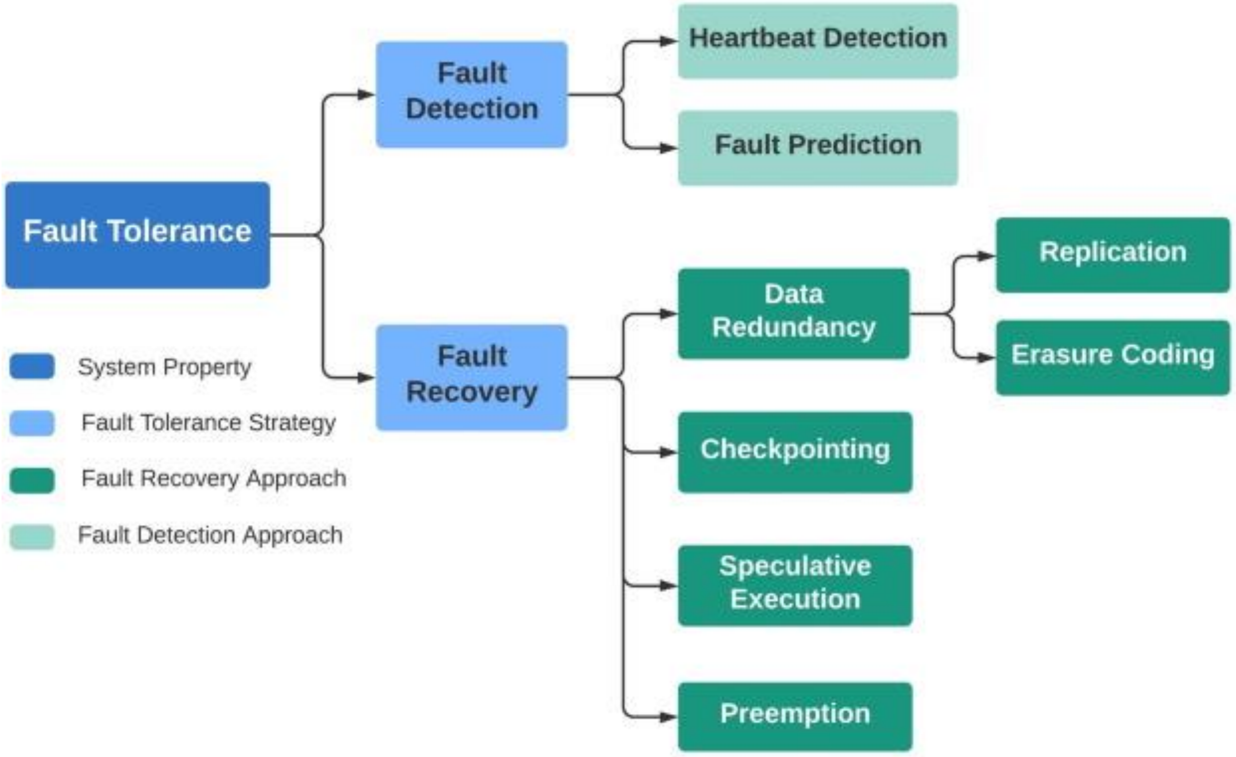
Scalability in Big Data Storage

- **Definition**: The ability of a system to handle increasing amounts of work or its potential t be enlarged to accommodate that growth.
- **Importance**: As data volume grows exponentially, systems must easily scale, whether vertically (adding more power to a single node) or horizontally (adding more nodes).
- **Examples**: Cloud storage solutions like AWS or database systems like Cassandra are designed with scalability in mind.



Fault Tolerance

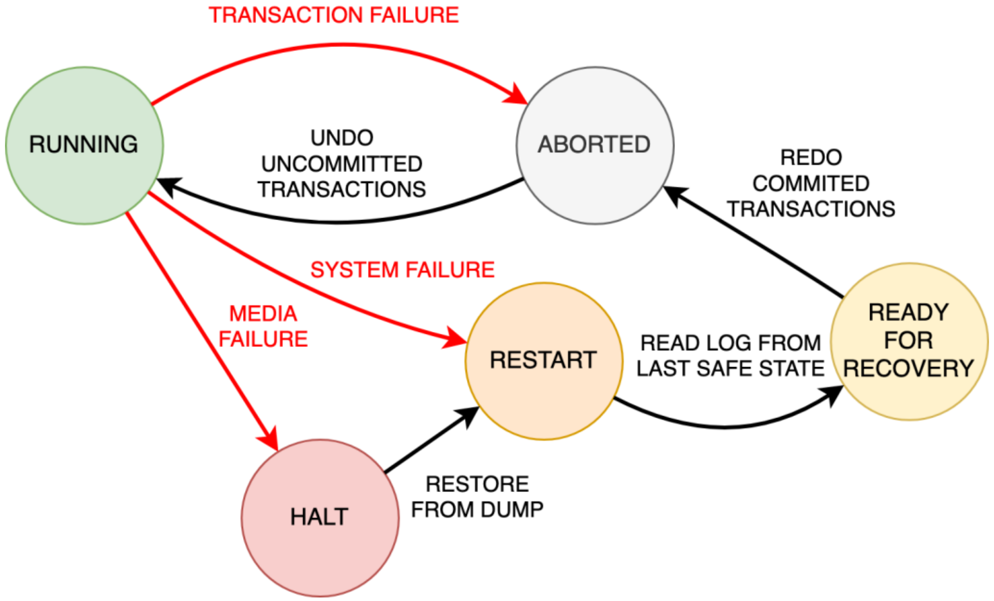
- **Definition**: The property that enables a system to continue operating correctly even in the presence of failures in some of its components.
- **Importance**: Failures are inevitable, especially in large systems. Effective Big Data storage systems need to operate without interruption even when individual components fail.
- **Examples**: Hadoop's HDFS replicates each piece of data multiple times across nodes to ensure data is not lost when nodes fail.



https://www.sciencedirect.com/science/article/pii/S2090447921002896

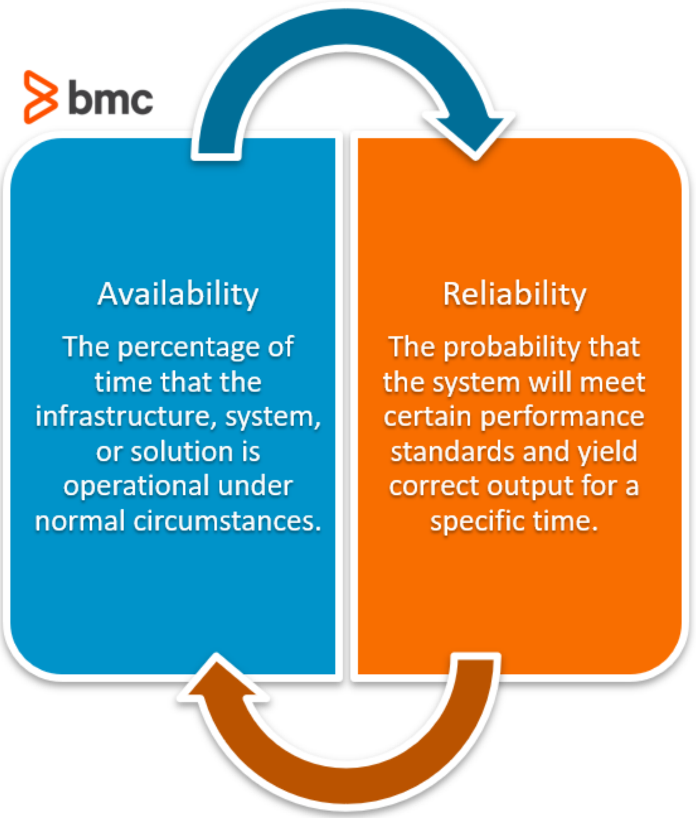
DATA DURABILITY

- **Definition**: The assurance that once data is stored, it will remain intact and available, even in the face of hardware failures or other issues.
- **Importance**: With the value derived from data analytics, ensuring that data isn't lost or corrupted is paramount.
- **Examples**: Amazon's S3 storage service boasts 99.999999999% (11 9's) annual durability for its objects.



https://en.wikipedia.org/wiki/Durability_%28database_systems%29

Durability vs Availability



bmc

Availability

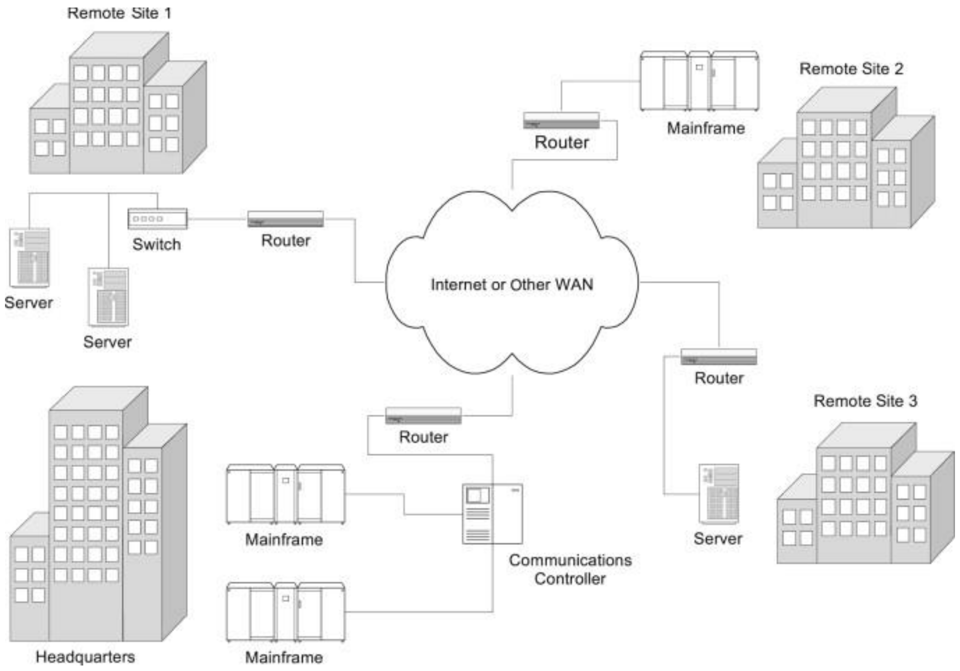
The percentage of time that the infrastructure, system, or solution is operational under normal circumstances.

Reliability

The probability that the system will meet certain performance standards and yield correct output for a specific time.

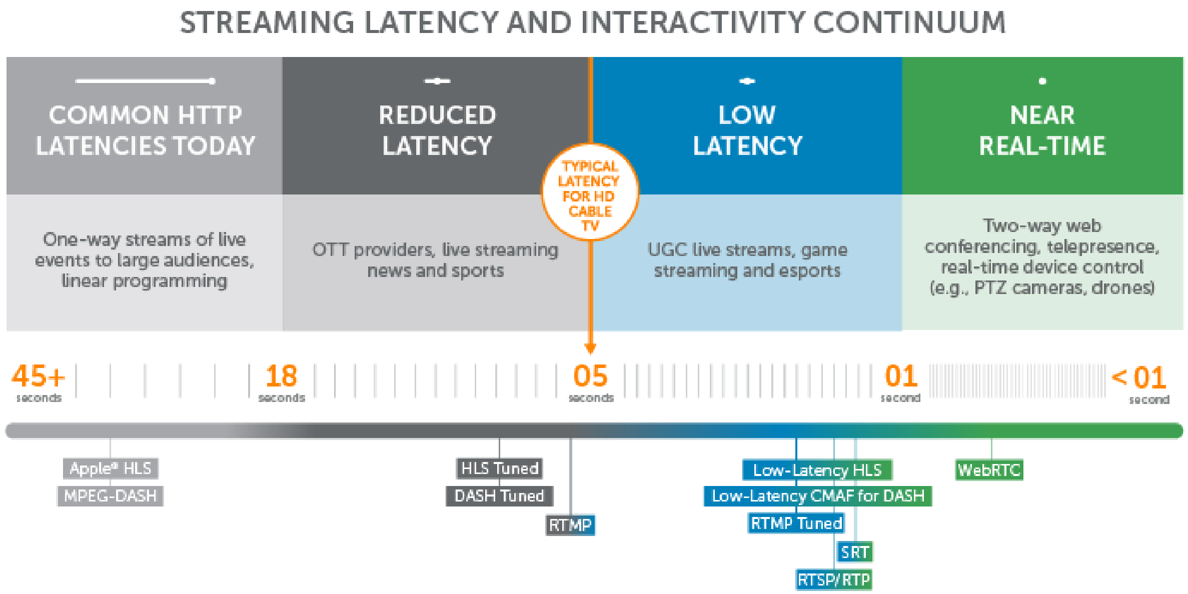
Distributed Architecture

- **Definition**: A method wherein storage and processing are spread across multiple devices or locations, often to enhance performance, scalability, and fault tolerance.
- **Importance**: Distributing data and computation allows systems to scale seamlessly, improve fault tolerance, and ensure data locality for faster processing.
- **Examples**: Distributed databases like Apache Cassandra or distributed processing systems like Apache Spark.



Low Latency Access

- **Definition**: The ability to read or write data with minimal delay, ensuring real-time or near-real-time data processing and access.
- **Importance**: As businesses move towards real-time analytics and decision-making, quick access to data is essential.
- **Examples**: In-memory databases like Redis or high-performance databases like Google's Bigtable offer low latency data access.


STREAMING LATENCY AND INTERACTIVITY CONTINUUM
COMMON HTTP LATENCIES TODAY
REDUCED LATENCY
LOW LATENCY
NEAR REAL-TIME
TYPICAL LATENCY FOR HD CABLE TV
One-way streams of live events to large audiences, linear programming
OTT providers, live streaming news and sports
UGC live streams, game streaming and esports
Two-way web conferencing, telepresence, real-time device control (e.g., PTZ cameras, drones)
45+ seconds
18 seconds
05 seconds
01 second
< 01 second
Apple® HLS
MPEG-DASH
HLS Tuned
DASH Tuned
RTMP
Low-Latency HLS
Low-Latency CMAF for DASH
RTMP Tuned
SRT
RTSP/RTP
WebRTC

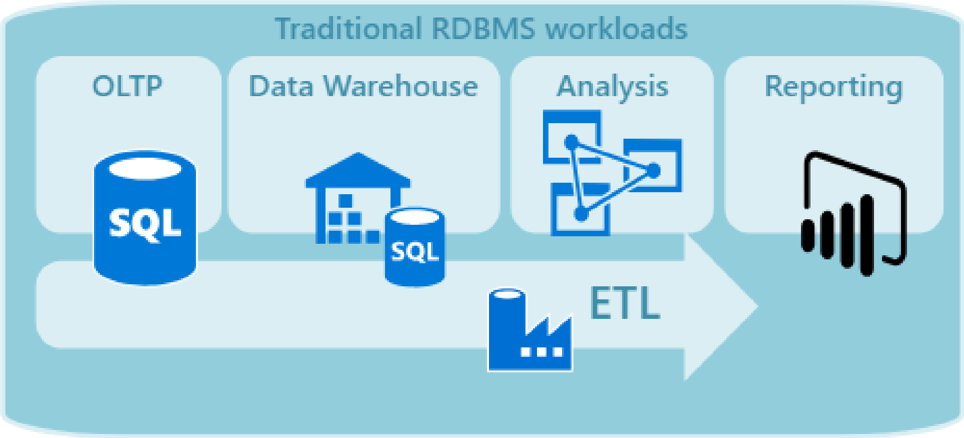
https://www.wowza.com/blog/ultra-low-latency

Solutions for Big Data Storage

Prof. Roozbeh Haghnazar

Traditional RDBMS Limitations in the Big Data Context

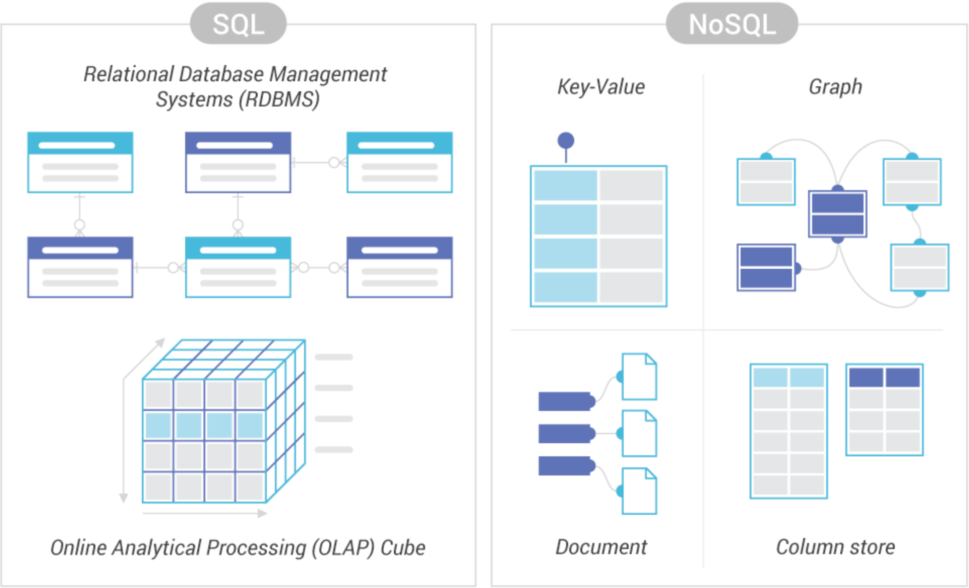
- **Definition**: RDBMS (Relational Database Management Systems) are databases designed to manage structured data and relationships using SQL.
- **Limitations**:
  - **Scalability**: Hard to scale out horizontally (across multiple servers).
  - **Fixed Schema**: Requires predefined schema, making it challenging to manage unstructured or semi-structured data.
  - **Performance**: Might suffer latency issues with very high volumes of data.
- **Note**: Mention that while RDBMS is robust and serves many applications well, the era of Big Data demands more flexibility and scalability.



https://learn.microsoft.com/en-us/azure/architecture/data-guide/databases-architecture-design

Evolution of NoSQL Databases

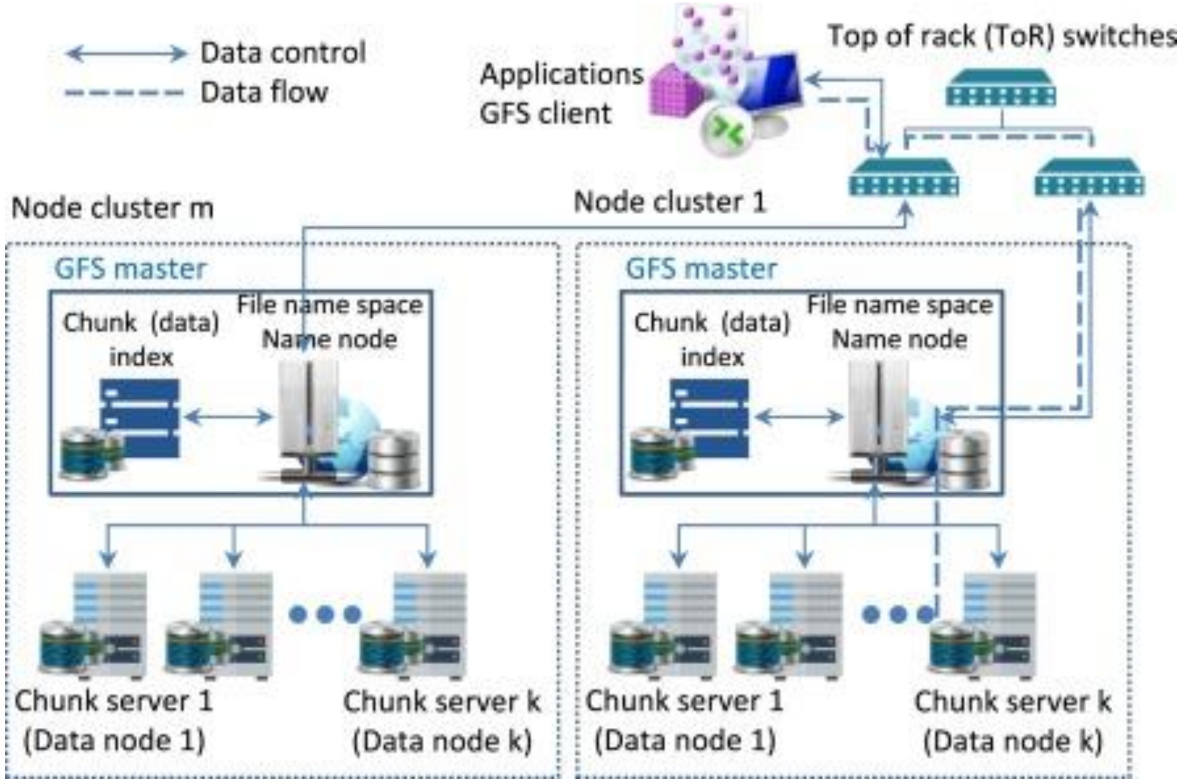
- **Definition**: NoSQL (Not Only SQL) databases are designed to allow for storage and retrieval of data that is modeled differently than the tabular relations used in RDBMS.
- **Key Features**:
  - **Flexible Schema**: Can accommodate structured, semi-structured, and unstructured data.
  - **Scalability**: Designed for horizontal scalability.
  - **Diverse Data Models**: Document, Key-Value, Column-Family, and Graph.
- **Note**: Highlight that NoSQL is not a replacement but an alternative based on specific data needs.


SQL
Relational Database Management Systems (RDBMS)
Online Analytical Processing (OLAP) Cube
NoSQL
Key-Value
Graph
Document
Column store

https://www.scylladb.com/learn/nosql/nosql-vs-sql/

Introduction to Distributed File Systems

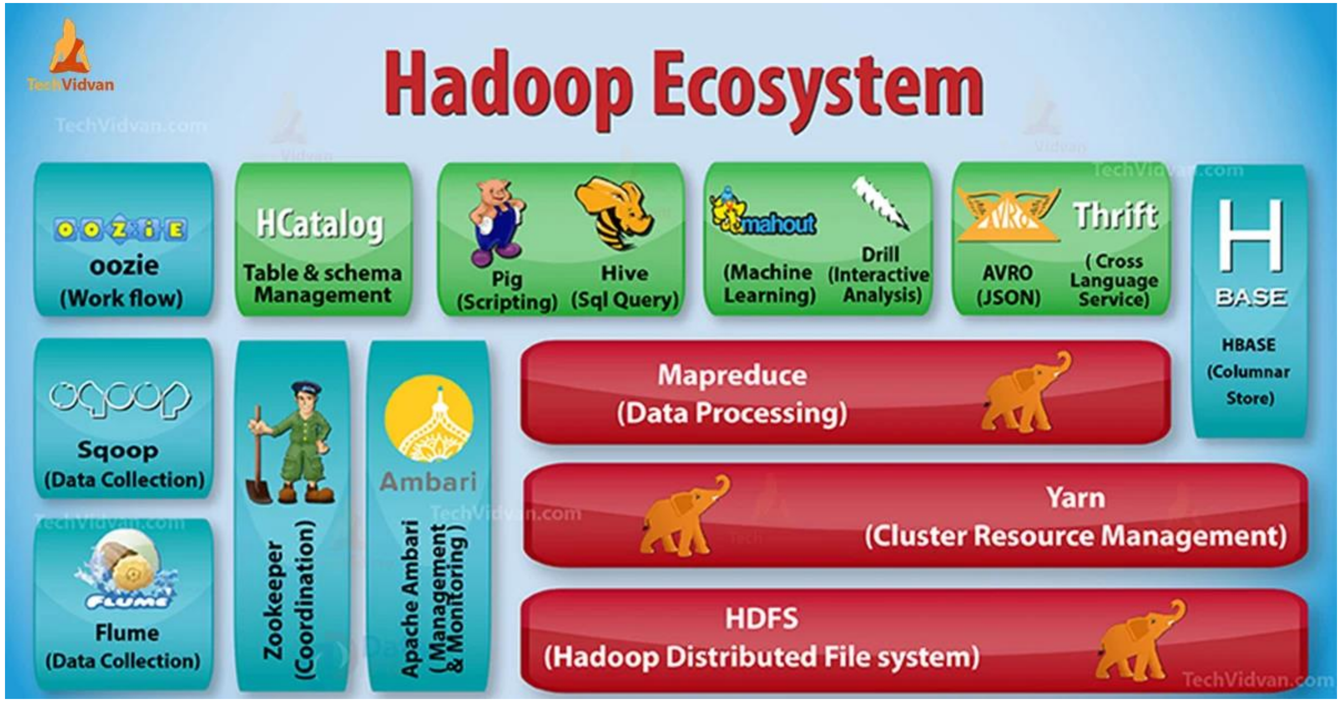
- **Definition**: A distributed file system allows files to be stored across multiple nodes or machines, ensuring data redundancy, fault tolerance, and high availability.
- **Benefits**:
  - **Fault Tolerance**: Data replication across nodes.
  - **Scalability**: Seamless addition of nodes.
  - **Optimized for Big Data**: Designed to manage petabytes of data.
- **Examples**: Mention systems like Google File System (GFS) and its inspiration to other systems.



https://www.sciencedirect.com/topics/computer-science/hadoop-distributed-file-system

Why Hadoop & HDFS?

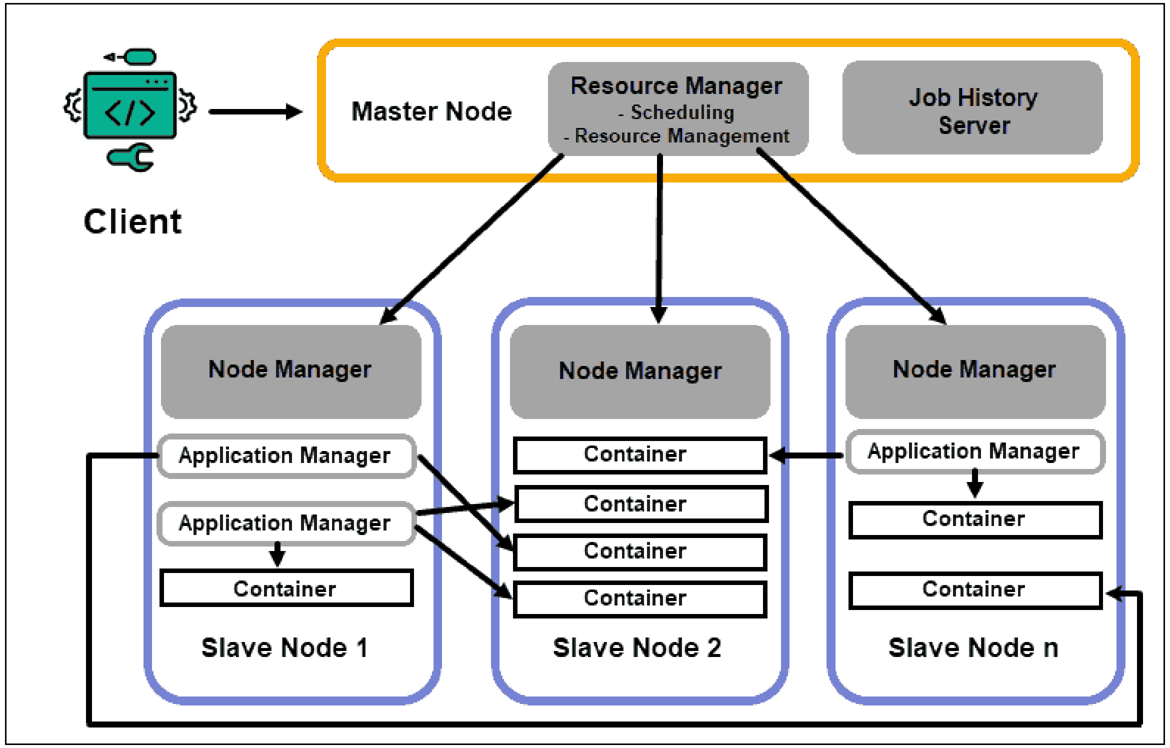
- **Introduction**: Hadoop, developed by the Apache Software Foundation, is an open-source framework designed to process vast amounts of data across clusters of computers using simple programming models. One of its core components is the Hadoop Distributed File System (HDFS).
- **Integration with other tools:** Hadoop does not work in isolation. Its ecosystem comprises various tools like Pig (for data manipulation), Hive (data warehousing), YARN (a scheduler), and many more. These tools together provide a comprehensive Big Data solution.



WHY HADOOP & HDFS?

Key Features & Benefits:

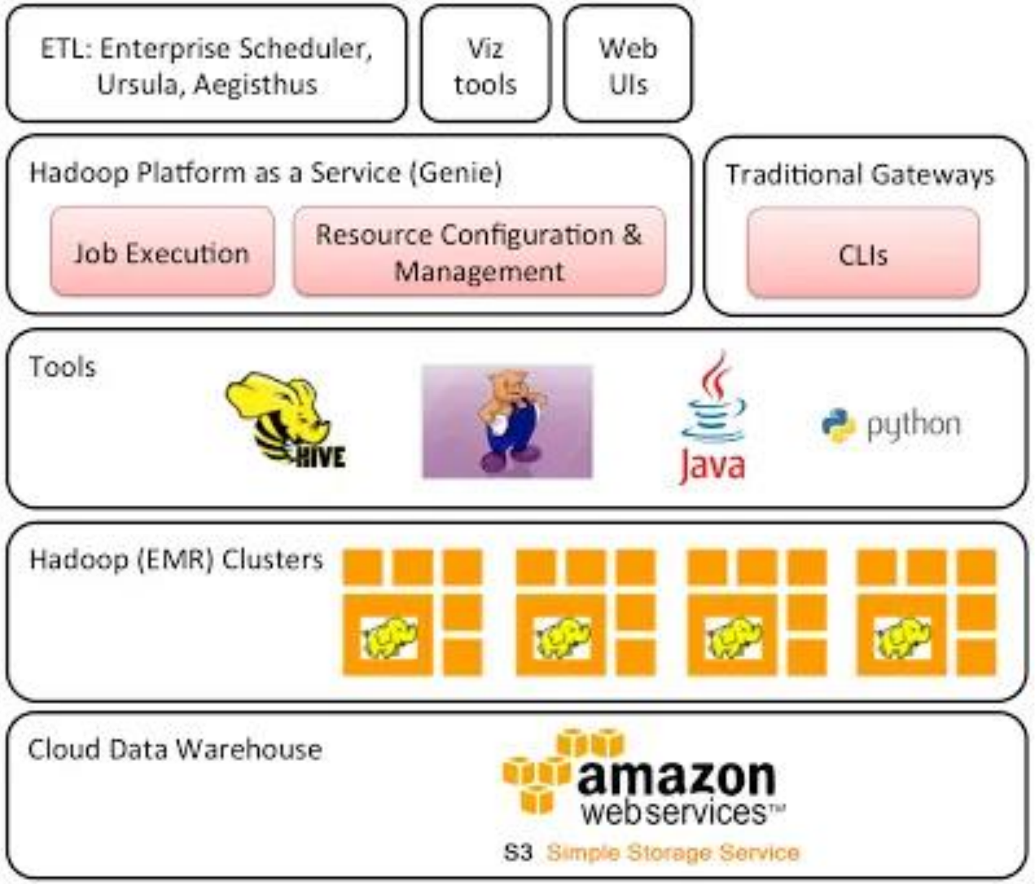
- **Distributed Storage**:
  - HDFS breaks down large data files into smaller blocks (typically 128 MB or 256 MB) and stores multiple copies of these blocks across the cluster's nodes. This ensures both data durability and high data access speeds.
- **Fault Tolerance & Reliability**:
  - HDFS is inherently resilient to node failures. When a node fails, data processing can redirect to another location where a copy of the data exists, ensuring system reliability and data integrity.
- **Scalability**:
  - Hadoop clusters can be easily scaled by adding more nodes. This makes it a cost-effective solution, as organizations can start small and grow their Hadoop infrastructure as their data needs increase.
- **Cost-Effective**:
  - Hadoop is designed to run on commodity hardware, making it an affordable solution for storing and processing massive datasets.
- **Data Locality**:
  - Hadoop processes data on the same server where the data is stored, reducing data transfer times and boosting processing speed—a concept known as data locality.



https://phoenixnap.com/kb/apache-hadoop-architecture-explained

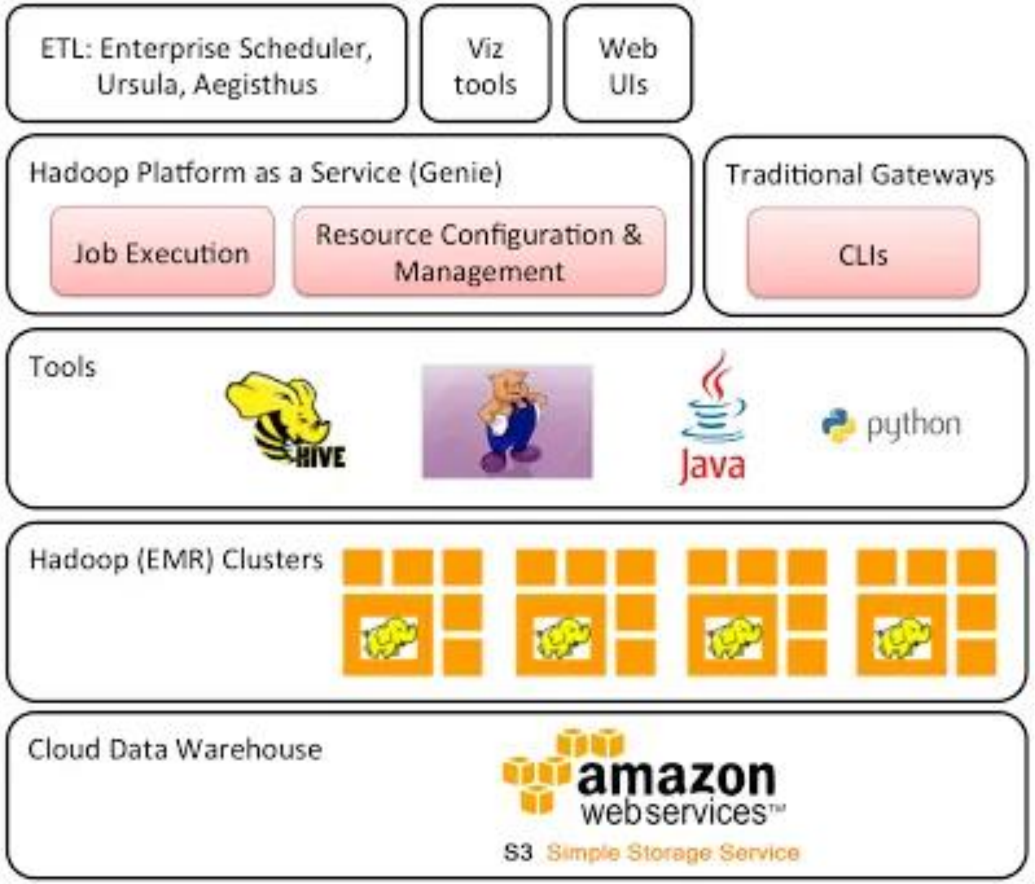
HADOOP PLATFORM AS A SERVICE IN THE CLOUD

- **Background**:
  - Hadoop is a standard for managing large datasets; Netflix's warehouse scales in petabytes.
  - Netflix's cloud-based architecture allows infinite scaling both in data and computational power.
  - Netflix introduced "Genie" as their in-house Hadoop Platform as a Service (PaaS) that offers REST-ful APIs for job management.
- **Architecture**:
  - Traditional Hadoop warehouses use Hadoop Distributed File System (HDFS).
  - Netflix's approach: Data stored on Amazon's Storage Service (S3) for durability, elasticity, and scalability.
  - Downsides: S3's read/write can be slower than HDFS.
- **Cluster Management**:
  - Netflix uses Amazon's Elastic MapReduce (EMR) for Hadoop.
  - S3's use allows dynamic creation of multiple Hadoop clusters accessing the same data without replication.
  - They resize clusters based on need without worrying about data loss.
- **Tools Used**:
  - Hive (for queries & analytics), Pig (for ETL and algorithms), Java for complex algorithms.
  - Developers use "gateways" to access Hadoop clusters and execute commands.


ETL: Enterprise Scheduler, Ursula, Aegisthus
Viz tools
Web UIs
Hadoop Platform as a Service (Genie)
Job Execution
Resource Configuration & Management
Traditional Gateways
CLIs
Tools
HIVE
Java
python
Hadoop (EMR) Clusters
Cloud Data Warehouse
amazon webservices™
S3 Simple Storage Service

Hadoop Platform as a Service in the Cloud

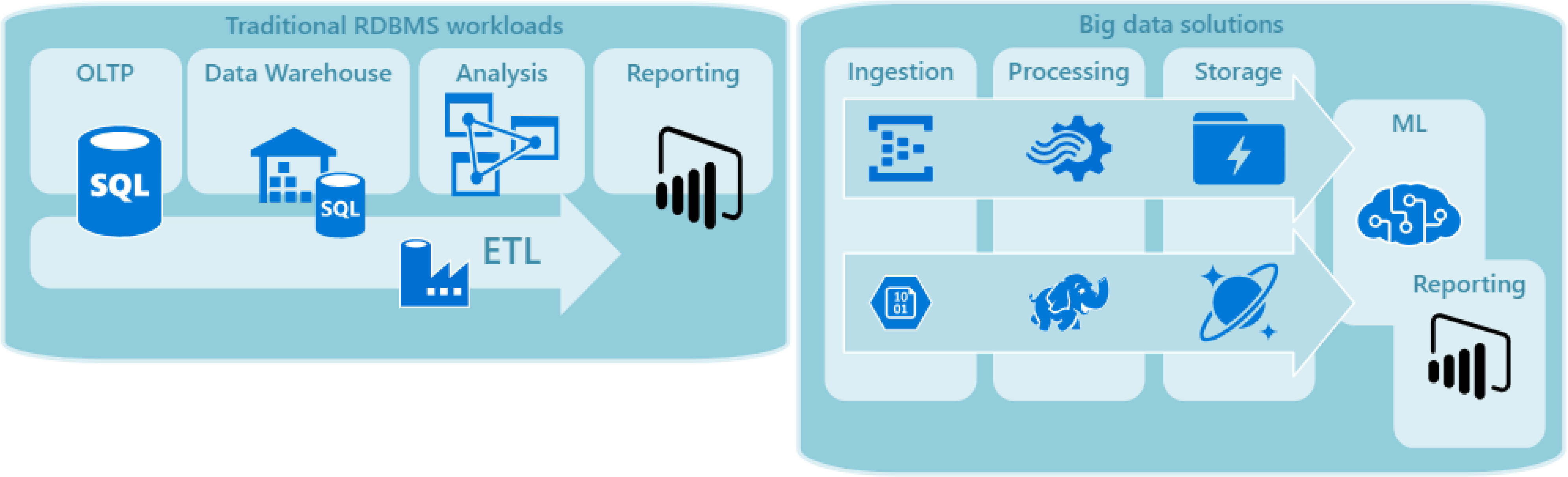
- **Introducing Genie**:
  - Provides an abstraction level for job management without provisioning Hadoop clusters.
  - Why Genie? To manage loosely coupled ETL processes across cloud and on-premises, handle increasing job loads, and abstract physical cluster details.
  - Genie Services:
    - **Execution Service** - Offers a RESTful API for job submissions and management.
    - **Configuration Service** - A repository of active clusters and their metadata.
- **Dynamic Resource Management**:
  - Genie aids in efficiently utilizing available resources based on load.
  - Different deployment options for upgrades, ensuring no job loss.
- **Deployment Status**:
  - Genie is extensively used in Netflix's production environment, ensuring scalability and fault tolerance.



ETL: Enterprise Scheduler, Ursula, Aegisthus
Viz tools
Web UIs
Hadoop Platform as a Service (Genie)
Job Execution
Resource Configuration & Management
Traditional Gateways
CLIs
Tools
HIVE
Java
python
Hadoop (EMR) Clusters
Cloud Data Warehouse
amazon webservices™
S3 Simple Storage Service

https://netflixtechblog.com/hadoop-platform-as-a-service-in-the-cloud-c23f35f965e7

DATABASES ARCHITECTURE DESIGN



ALTERNATIVES FOR HADOOP AS A SERVICE ON THE CLOUD

- Hadoop as a service on the cloud refers to cloud offerings where vendors provide Hadoop-based big data analytics platforms. The advantage of these services is that they abstract away the complexities of deploying, managing, and scaling Hadoop clusters. Here are some of the popular cloud-based Hadoop services and alternatives:
- **Amazon EMR (Elastic MapReduce)**
  - Managed by Amazon Web Services (AWS).
  - Supports various popular Hadoop ecosystem tools such as Spark, HBase, Hive, and Pig.
  - Integrates well with other AWS services.
- **Google Cloud Dataproc**
  - Managed by Google Cloud Platform (GCP).
  - Offers fast deployments of Spark, Hadoop, Pig, and Hive.
  - Per-minute billing and easy integration with other GCP services.
- **Azure HDInsight**
  - Managed by Microsoft Azure.
  - Supports various Hadoop ecosystem tools including Spark, Hive, HBase, Storm, and Kafka.
  - Integrates with other Azure services and offers enterprise-level security.
- **Databricks**
  - A unified analytics platform on the cloud.
  - Built around Apache Spark, it offers collaborative notebooks, integrated workflows, and robust security.
  - Available on AWS and Azure.

ALTERNATIVES FOR HADOOP AS A SERVICE ON THE CLOUD

- **Cloudera Data Platform (CDP) - Public Cloud**
  - The cloud-native platform from Cloudera, built with a focus on hybrid cloud.
  - Includes multi-function analytics with common security and governance.
- **IBM Cloud Analytics Engine**
  - Combines IBM's analytics and Hadoop services.
  - Provides separate compute and storage for flexibility.
- **Qubole**
  - A self-service platform for big data analytics built on AWS, Azure, and Oracle Cloud Infrastructure.
  - Supports auto-scaling with a range of big data tools, including Hadoop, Spark, and Presto.
- **Oracle Big Data Cloud Service**
  - A managed Hadoop and Spark cloud service.
  - Integrated with Oracle's suite of cloud services and database technologies.

Row-Oriented vs Column-Oriented DBMS

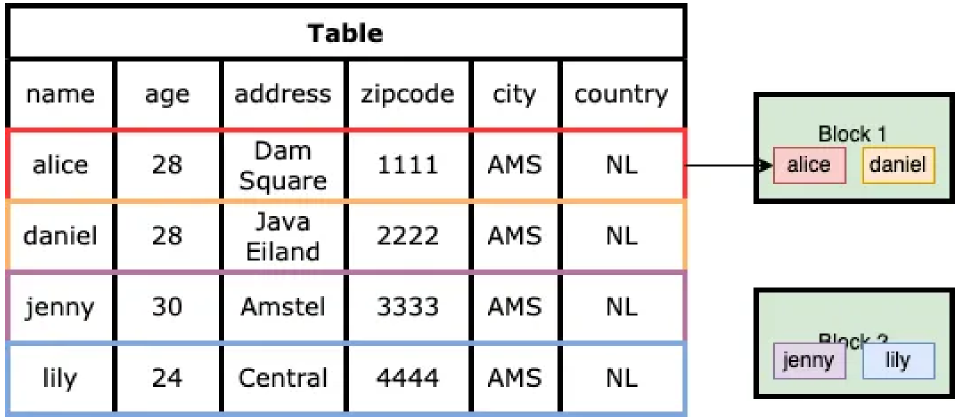
Prof. Roozbeh Haghnazar

INTRODUCTION

- Traditional database systems have been designed with a primary focus on row-oriented storage.
- With the advent of big data and analytical processing needs, column-oriented storage has gained prominence.
- This section delves into the fundamental differences, advantages, and disadvantages of each.

Traditional Row-Oriented DBMS

- **Definition**
  - In row-oriented DBMS, data is stored in rows, one after the other. Each row represents a single record.
- **Use Case**
  - Best suited for OLTP (Online Transaction Processing) systems where operations typically fetch or modify one record at a time.



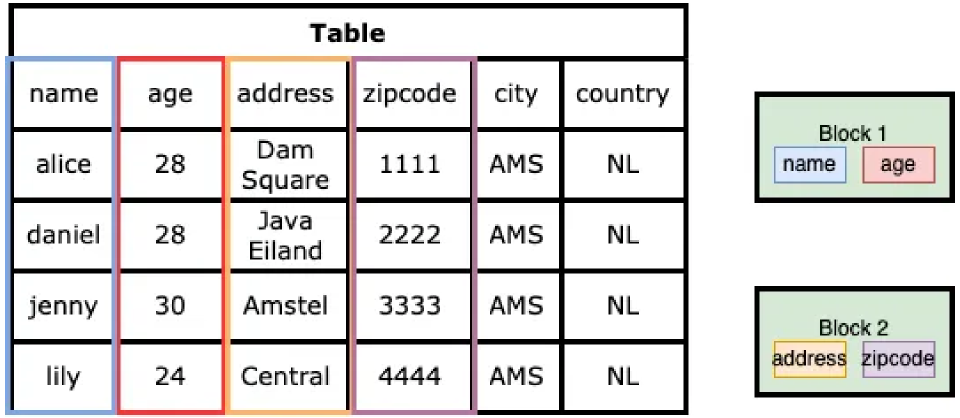
Table

name	age	address	zipcode	city	country
alice	28	Dam Square	1111	AMS	NL
daniel	28	Java Eiland	2222	AMS	NL
jenny	30	Amstel	3333	AMS	NL
lily	24	Central	4444	AMS	NL

https://towardsdatascience.com/understand-columnar-and-row-based-database-2cd29ae35bd0

COLUMN-ORIENTED DBMS

- **Definition**
  - In a column-oriented DBMS, data is stored column by column. Each column from multiple rows is stored together.
- **Use Case**
  - Best suited for OLAP (Online Analytical Processing) systems where operations involve scanning large datasets for specific columns.



Table

name	age	address	zipcode	city	country
alice	28	Dam Square	1111	AMS	NL
daniel	28	Java Eiland	2222	AMS	NL
jenny	30	Amstel	3333	AMS	NL
lily	24	Central	4444	AMS	NL

Block 1

name age

Block 2

address zipcode

https://towardsdatascience.com/understand-columnar-and-row-based-database-2cd29ae35bd0

Advantages and Disadvantages of Row-Oriented Storage

- **Advantages**
  - **Optimized for Transactional Operations**: Fast for write operations, especially inserts.
  - **Data Locality**: Data for a single record is stored together, making retrievals fast for single records.
  - **Familiarity**: Many traditional and widely used databases (like MySQL, MS SQL) use row-oriented storage.
- **Disadvantages**
  - **Analytical Performance**: Scanning large datasets for specific columns can be slow.
  - **Storage Overhead**: Storing metadata for each row can introduce overhead.

ADVANTAGES AND DISADVANTAGES OF COLUMN-ORIENTED STORAGE

- **Advantages**
  - **Optimized for Analytics**: Fast scans over large datasets for specific columns.
  - **Compression Benefits**: Similar data in columns can be compressed more effectively.
  - **Flexibility**: Addition of new columns can be easier.
- **Disadvantages**
  - **Row-wise Operations**: Retrieving or updating a single record can be slower compared to row-oriented storage.
  - **Complexity**: Columnar databases can be more complex in terms of design and maintenance.

WHICH TO CHOOSE? - FACTORS TO CONSIDER

- Type of Workload: OLTP vs. OLAP
- Data Volume: Large datasets with repetitive column data favor column-oriented storage.
- Query Patterns: Do queries scan columns or retrieve specific rows?
- Maintenance Complexity: Familiarity with the system and its requirements.
- Future Scaling Needs: How might the data grow or change in the future?

OLTP VS OLAP

Definition & Main Focus

- **OLTP (Online Transaction Processing)**
  - Focuses on transactional operations.
  - Examples: Online banking, order processing, retail sales.
- **OLAP (Online Analytical Processing)**
  - Focuses on data analysis and complex queries.
  - Examples: Business intelligence, data warehousing, reporting.

Characteristics

- **OLTP**
  - **High Transaction Volume**: Designed to handle many short online transactions.
  - **Consistency**: Emphasizes atomicity, consistency, isolation, and durability (ACID properties).
  - **Simple Queries**: Typically involves accessing only a few records.
  - **Normalized Schemas**: Reduces data redundancy.
- **OLAP**
  - **Complex Queries**: Often scans millions of rows.
  - **Read-Intensive**: Fewer writes compared to reads.
  - **Denormalized or Star Schema**: Facilitates faster analytical processing.
  - **Aggregation**: Supports complex calculations and aggregations.

OLTP vs OLAP

Database Design

- **OLTP**
  - **Row-Oriented Storage**: Optimized for quick, routine transactional tasks.
- **OLAP**
  - **Column-Oriented Storage**: Optimized for column-wise querying, which aids in analytical processing.

Examples of Systems

- **OLTP:** MySQL, PostgreSQL, MS SQL Server, Oracle DB.
- **OLAP:** Apache Druid, Amazon Redshift, Google BigQuery, Snowflake, VERTICA.