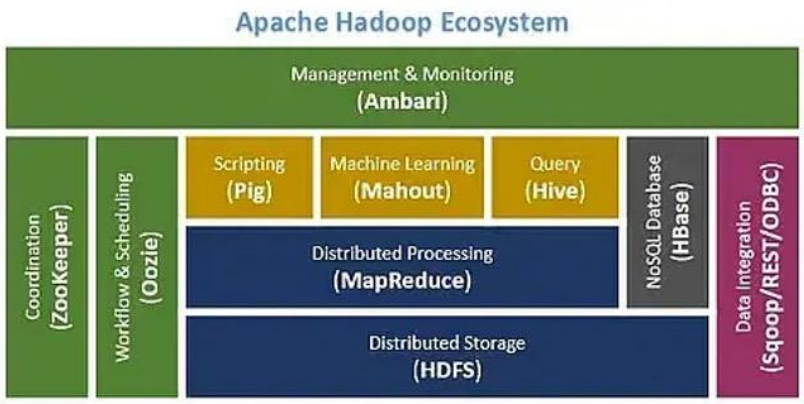
Hadoop

What is Hadoop?

Hadoop is an open-source framework used to store and process big data in a distributed and scalable way. It was developed by Apache and is built to handle massive datasets across clusters of computers.

Hadoop Ecosystem comprises:

1. **HDFS:** Hadoop, Hadoop Distributed File System, which simply stores data files as close to the original format as possible. It has capable to store very large files across multiple machines in a Hadoop cluster.
2. **Hbase:** It is a Hadoop database management system like HBase is an open-source, non-relational, distributed database that stores data in a column-oriented manner. It's designed to handle large, sparse datasets and provides fast, real-time access to data, making it suitable for various big data use cases.
3. **Hive** : Apache Hive is a data warehouse system built on top of Hadoop to analyze big data. Instead of writing complex MapReduce code it enables query, SQL-like syntax, called HiveQL and And Hive will automatically convert it into MapReduce jobs behind the scenes.
4. **Pig** is an easy-to-understand data flow language, helpful in analyzing Hadoop- based data. Pig scripts are automatically converted to MapReduce jobs by the Pig Interpreter, thus enabling SQL-type processing of Hadoop data.
5. **ZooKeeper**: Apache ZooKeeper is a centralized service used to manage configuration, coordination, and synchronization for distributed systems. It acts as the “brains” or “traffic controller” of a cluster by ensuring consistency and coordination across nodes in a fault-tolerant way.
6. **Oozie** is a workflow schedule system to manage Apache Hadoop Jobs. It chains together multiple big data tasks like MapReduce, Hive, Pig, or Shell scripts and run them in a defined sequence or schedule.
7. **Mahout**: Apache Mahout is an open-source project designed to build scalable machine learning algorithms. It was originally developed to run on top of Hadoop MapReduce, but has since evolved to support Apache Spark and Scala DSL for faster computation.
8. **Chukwa**: Apache Chukwa is a data collection and monitoring system built on top of Hadoop. It helps you collect logs and system metrics from various sources and then stores them in HDFS or other Hadoop-compatible storage for analysis.
9. **Sqoop**: Apache Sqoop is a tool designed to efficiently transfer bulk data between relational databases (like MySQL, Oracle, SQL Server, PostgreSQL) and Hadoop ecosystems (like HDFS, Hive, HBase).
10. **Ambari:** Apache Ambari is a web-based tool for provisioning, managing, monitoring, and securing Hadoop clusters. It is like the “cPanel” of Hadoop — instead of doing everything via the terminal or multiple config files, Ambari helps to monitor Hadoop components (like HDFS, YARN, Hive), to install new services, to restart failed nodes, to track metrics and alerts, and to manage user permissions and configurations.



What are NoSQL databases?

NoSQL databases are non-relational database systems that store and retrieve data differently than traditional SQL databases that can handle sizes of Big Data. They are non-relational, open-source, distributed databases which are:

- capable to handle the following types of data
    - structured,
    - semi-structured and
    - less structured,
- with no fixed table schema,
- with no JOINS,
- with no multi document transactions and (therefore)
- relaxing one or more ACID proper- ties.

The ACID properties of Atomicity, Consistency, Isolation and Durability are not strictly adhered to in NoSQL databases. In contract, NoSQL databases adhere to CAP (Consistency, Availability and Partition tolerance) properties.

What are the different types of noSQL Databases?

There are five main types of NoSQL databases:

1. Document databases

Document databases, also called document-oriented databases or a document store, are used to store and query semi-structured data. Data is stored in a JSON-like document similar to the data objects that developers use in application code, making it easier to create and update applications without referencing a primary schema. Document databases are most commonly used for blogging platforms, ecommerce, real-time analytics, and content management systems.

The three important representatives of document storage systems, Mon- goDB, SimpleDB, and CouchDB.

- MongoDB is an open-source document-oriented database. MongoDB stores documents as Binary JSON (BSON) objects, which is similar to objects.
- SimpleDB is a distributed database and a web service of Amazon. Data in SimpleDB is organized into various domains in which data may be stored, acquired, and queried. SimpleDB allows users to use SQL to run queries.
- Apache CouchDB is a document-oriented database written in Erlang. Data in CouchDB is organized into documents that consist of fields named by key words/names and values, and are stored and accessed as JSON objects. Every document is provided with a unique identifier. CouchDB allows access to database documents through the RESTful HTTP API.

2. Key-value databases

Key-value databases, also referred to as key-value stores, are the simplest type of NoSQL databases. Data is stored in a "key-value" structure, where a unique key is paired with a value such as a string, number, boolean, or complex objects. Key-value stores are most commonly used for user preferences, shopping carts, and user profiles in web applications. (e.g., Key: First name; value : Rama; Key: Last name; value: Mainu)

Over the past few years, many key-value databases have appeared as moti- vated by Amazon's Dynamo system.

Several other representative key-value databases: Dynamo, Voldemort.

Dynamo is a highly available and expandable distributed key-value data storage system. It is used to manage the store status of some core services in the Amazon e-Commerce Platform. Amazon e-Commerce Platform provides multiple services and data storage that can be realized with key access.

Voldemort is also a key-value storage system, which was initially developed for and is still used by LinkedIn. Key words and values in Voldemort are composite objects constituted by tables and images.

The Voldemort interface includes three simple operations: reading, writing, and deletion, all of which are confirmed by key words.

Other key-value storage systems include Redis, Tokyo Cabinet and Tokyo Tyrant, Memcached and MemcacheDB, Riak and Scalaris.

3. Column-oriented databases

Column-oriented databases, or wide-column stores, store and read data in rows and are organized as a set of columns. They are optimal for analytics use cases, where you may need to query across specific columns in a database and aggregate the value of a given column quickly. The column-oriented databases are mainly inspired by Google's BigTable. Wide-column stores are most commonly used for catalogs, fraud detection, and recommendation engines.

BigTable

BigTable is a distributed, structured data storage system, which is de- signed to process the large-scale (PB class) data among thousands of commercial servers. The basic data structure of BigTable is a multi- dimension sequenced mapping with sparse, distributed, and persistent stor- age.

BigTable is based on many fundamental components of Google, including GFS, cluster management system, SSTable file format, and Chubby. GFS is used to store data and log files.

Cassandra

Cassandra is a distributed storage system to manage the huge amount of structured data distributed among multiple commercial servers. The system was developed by Facebook and became an open-source tool in 2008.

It adopts the ideas and concepts of both Amazon Dynamo and Google BigTable, especially integrating the distributed system technology of Dy- namo with the BigTable data model.

Tables in Cassandra are in the form of distributed four-dimensional struc- tured mapping, where the four dimensions include row, column family, col- umn, and super column.

4. Graph databases

Graph databases organize data as nodes in a graph, focusing on the relationships between data elements. The connections between nodes (edges) are stored as first-class elements, enabling richer representations of data relationships while offering more simplified storage and navigation. Graph databases are most commonly used in systems that map relationships, including social media platforms, reservation systems, fraud detection systems, and logistics applications.

5. In-memory databases

In-memory databases store data in memory in order to provide ultra-low latency for real-time applications. Redis and Valkey are examples of in-memory NoSQL databases. In-memory databases are most commonly used for caching, messaging, streaming, and real-time analytics.

Graph Databases

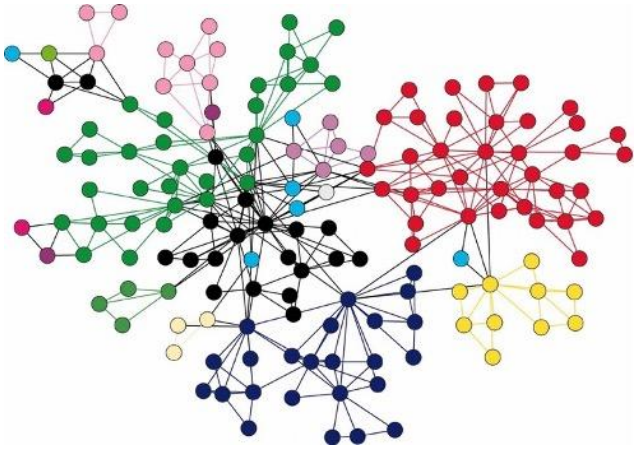
A graph database is a specialized database management system that stores and manages data as a network of nodes and edges, representing relationships between data points.

Nodes: Represent individual data entities or objects.

Edges: Represent the relationships or connections between nodes.

Properties: Attributes associated with nodes and edges, providing additional information about them.

Graph databases enable storage of entities and the relationships between them. In the graph database, entities are represented as nodes which have properties. Properties are represented as edges of the nodes with directionality being considered. Thus, graph databases enable us to not only store the data once but also allow us to interpret the same, in various ways, depending on the relationships indicated by edges. This feature is a unique strength of graph databases.



Popular Graph Database Examples:

Neo4j: A widely used and powerful graph database platform.

Amazon Neptune: A fully managed graph database service on AWS:.

ArangoDB: A multi-model database that supports graph data.

Dgraph: A distributed graph database known for its high performance.

JanusGraph: A distributed graph database optimized for scalability and performance.

TigerGraph: A proprietary graph database with features for data visualization and analysis.

Why Use a Graph Database

In complex Relationships, Graph databases are excellent for handling data with intricate and interconnected relationships, such as social networks, knowledge graphs, and recommendation systems.

- **Show Connections Clearly**: Think of a graph database like a map of a social network. It's very easy to see how everything is connected without having to dig through lots of data.
- **Find Relationships Fast**: In a graph database, finding out how one piece of data is related to another is quick. It's like being able to instantly see who your friends-of-friends are, without needing to search through every record one by one.
- **Easy to Expand and Change**: Graph databases are flexible. If we want to add new types of connections or new pieces of data, we can simply add more dots and lines without having to redo the entire system.

Some Real Applications

- Recommendation engines: People often search for similar products as others do. That is, if one person buys shoelaces after buying sneakers, there's a good chance others will do the same. Recommendation engines look for people who are connected by their purchases and then look for other closely connected products in the graph.
- Fraud detection: Good customers rarely commit fraud, and fraudsters often follow the same pattern again and again. Building out a graph of transactions can identify fraud by flagging suspicious patterns that often have no connection to legitimate transactions.
- Knowledge networks: Some artificial intelligence researchers have been creating graphs of facts and the connections between them so that computers can approximate human reasoning by following paths.
- Routing: Finding a path in the world is much like finding a path in an abstract graph that's modeling the roads. If the intersections are nodes, and the streets between them the links or edges, then the graph is a good abstract representation of the world. Choosing a path for an autonomous car just requires a sequence of nodes and links between them.