Module 1 Papers Solutions

May 2024:

Q1 a) Explain how big data problems are handled by Hadoop system. [5]

Explain using following content, as per the weightage given to the question

Traditional data storage and processing methods struggle with big data due to its massive volume, variety, and velocity. Here's how Hadoop tackles these challenges:

- **Distributed Storage:** Hadoop distributes data across a cluster of commodity hardware instead of relying on a single powerful machine. This allows for scaling storage capacity by simply adding more nodes to the cluster, making it a cost-effective solution.

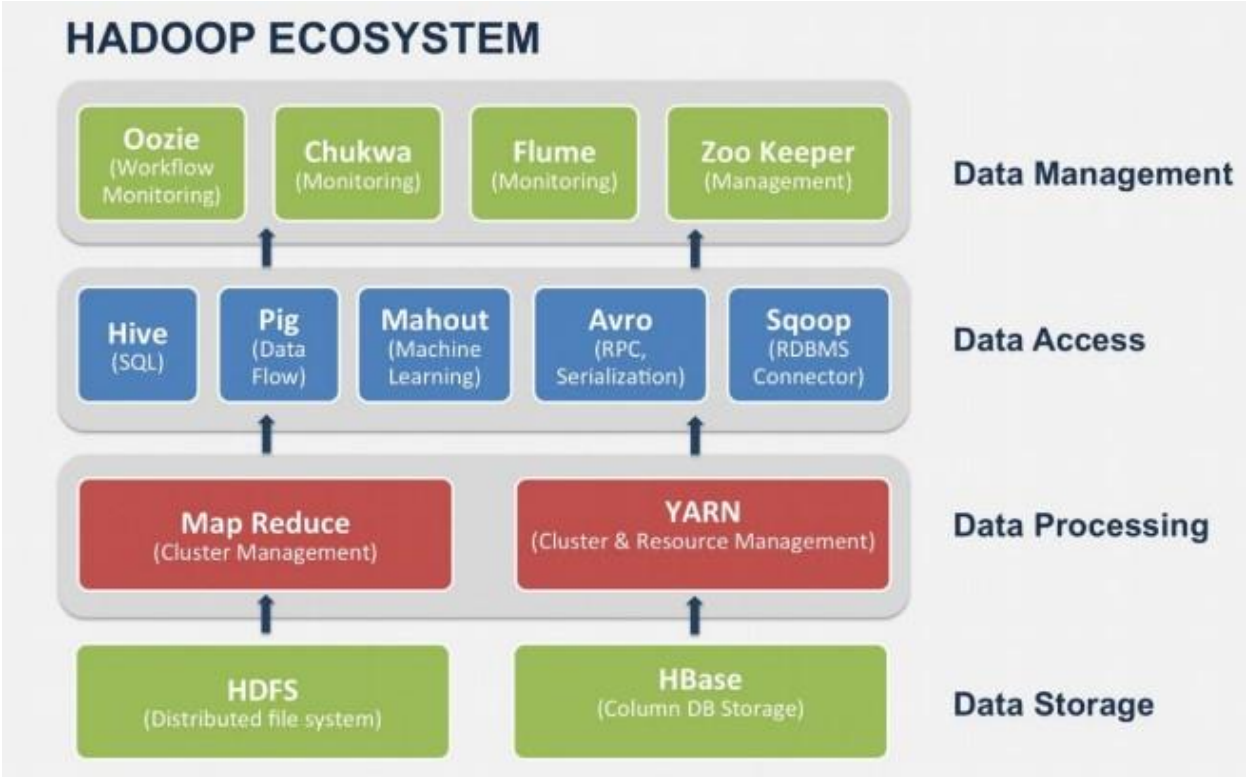
- **Parallel Processing:** Hadoop employs a programming paradigm called MapReduce to process data in parallel. MapReduce breaks down tasks into smaller, independent chunks that can be executed simultaneously on different nodes in the cluster. This significantly speeds up data processing compared to a single machine.
- **Handling Unstructured Data:** Unlike traditional databases that require structured data, Hadoop can handle any kind of data, including unstructured formats like text, audio, and video. This flexibility allows organizations to capture and analyze valuable insights from a wider range of data sources.

Hadoop's approach addresses several big data problems:

- **Scalability:** Traditional systems can't keep up with the ever-growing volume of data. Hadoop's distributed architecture allows for seamless horizontal scaling to accommodate increasing data storage and processing needs.

- **Cost-Effectiveness:** Hadoop utilizes commodity hardware, making it a more economical solution compared to expensive high-end servers required for traditional systems.
- **Data Variety:** Hadoop can store and process any type of data, eliminating the need for separate systems for structured and unstructured data.
- **Data Velocity:** Big data is often generated in real-time. Hadoop's parallel processing capabilities enable faster data analysis, allowing organizations to gain insights from data streams quickly.

In conclusion, Hadoop provides a powerful and cost-effective framework for handling big data challenges by leveraging distributed storage, parallel processing, and its ability to handle various data formats.



Hadoop Ecosystem is a platform or a suite which provides various services to solve the big data problems. It includes Apache projects and various commercial tools and solutions. There are *four major elements of Hadoop* i.e. **HDFS, MapReduce, YARN, and Hadoop Common**. Most of the tools or solutions are used to supplement or support these major elements. All these tools work collectively to provide services such as absorption, analysis, storage and maintenance of data etc.

A) Data Storage:

1) HDFS: HDFS is the primary or major component of the Hadoop ecosystem and is responsible for storing large data sets of structured or unstructured data across various nodes and thereby maintaining the metadata in the form of log files.

- HDFS consists of two core components i.e.

1. Name node
2. Data Node

2) Hbase: It's a NoSQL database which supports all kinds of data and is thus capable of handling anything from a Hadoop Database. It provides capabilities of Google's BigTable, thus able to work on Big Data sets effectively.

B) Data Processing:

1) MapReduce:

By making the use of distributed and parallel algorithms, MapReduce makes it possible to carry over the processing's logic and helps to write applications which transform big data sets into a manageable one.

- MapReduce makes the use of two functions i.e. Map() and Reduce() whose task is:
    1. *Map()* Map generates a key-value pair based result which is later on processed by the Reduce() method.
    2. *Reduce(),* Reduce() takes the output generated by Map() as input and combines those tuples into a smaller set of tuples.

2) YARN: Yet Another Resource Negotiator, as the name implies, YARN is the one who helps to manage the resources across the clusters. In short, it performs scheduling and resource allocation for the Hadoop System.

C) Data Access

1) Hive: With the help of SQL methodology and interface, HIVE performs reading and writing of large data sets. However, its query language is called HQL (Hive Query Language).

2) Mahout: Mahout, allows Machine Learnability to a system or application. Machine Learning, as the name suggests, helps the system to develop itself based on some patterns, user/environmental interaction or on the basis of algorithms.

3) Pig: Pig was basically developed by Yahoo which works on a pig Latin language, which is a Query based language similar to SQL. It is a platform for structuring the data flow, processing and analyzing huge data sets.

4) Avro: Avro is a row-oriented remote procedure call and data serialization framework developed within Apache's Hadoop project.

5) Sqoop: Sqoop is a command-line interface application for transferring data between relational tables and Hadoop. It is used to import data from relational databases such as MySQL, Oracle to Hadoop HDFS, and export from Hadoop file systems to relational databases.

D) Data Management

1) Zookeeper: There was a huge issue of management of coordination and synchronization among the resources or the components of Hadoop which resulted in inconsistency.

2) Oozie: Oozie simply performs the task of a scheduler, thus scheduling jobs and binding them together as a single unit.

3) Chukwa: Chukwa is an open source data collection system for monitoring large distributed systems.

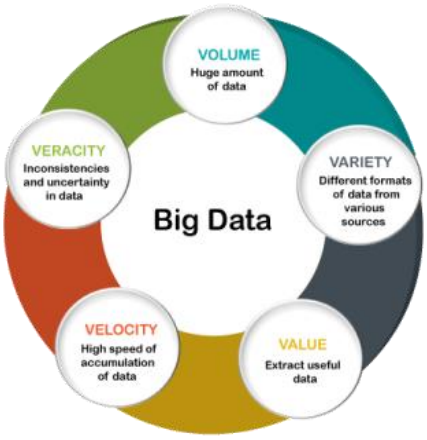
4) Flume : Flume is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data.

b) Mention four characteristics of big data and explain in detail. [5]

Explain any 4 out the following 5:

Big Data Characteristics (5 V's):

There are five v's of Big Data that explains the characteristics.

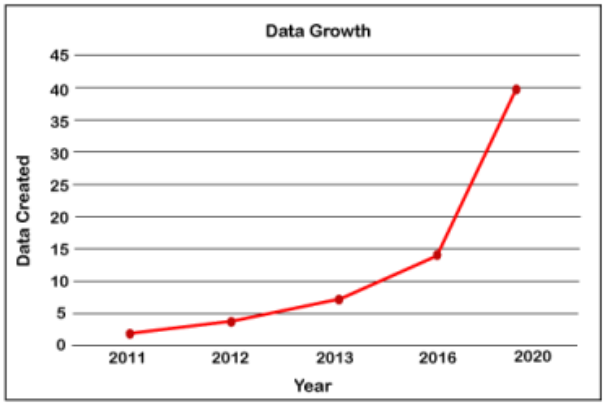


5 V's of Big Data

- o **Volume**
- o **Veracity**
- o **Variety**
- o **Value**
- o **Velocity**

Volume

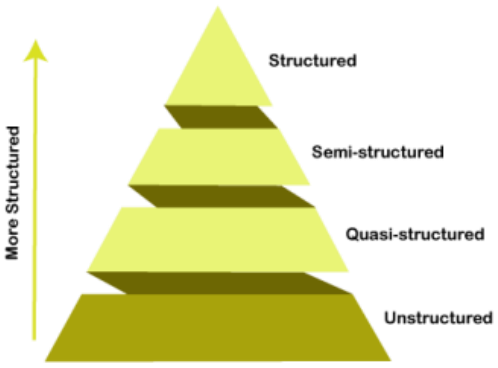
The name Big Data itself is related to an enormous size. Big Data is a vast 'volume' of data generated from many sources daily, such as **business processes, machines, social media platforms, networks, human interactions,** and many more. **Facebook** can generate approximately a **billion** messages, **4.5 billion** times that the "**Like**" button is recorded, and more than **350 million** new posts are uploaded each day. Big data technologies can handle large amounts of data.



Variety

Big Data can be **structured, unstructured, and semi-structured** that are being collected from different sources. Data will only be collected from **databases** and **sheets** in the past, But these days the data will comes in array forms, that are **PDFs, Emails, audios, SM posts, photos, videos,** etc.

The data is categorized as below:



a. **Structured data:** In Structured schema, along with all the required columns. It is in a tabular form. Structured Data is stored in the relational database management system.

b. **Semi-structured:** In Semi-structured, the schema is not appropriately defined, e.g., **JSON, XML, CSV, TSV**, and **email**. OLTP (**Online Transaction Processing**) systems are built to work with semi-structured data. It is stored in relations, i.e., **tables**.

c. Unstructured Data: All the **unstructured files, log files, audio files**, and **image** files are included in the unstructured data. Some organizations have much data available, but they did not know how to **derive** the value of data since the data is

raw.

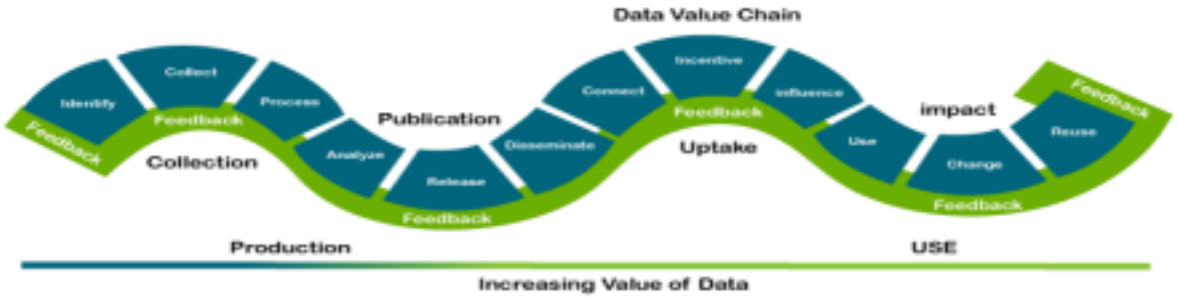
Example: Web server logs, i.e., the log file is created and maintained by some server that contains a list of **activities**.

Veracity

It refers to the quality and accuracy of data. Gathered data could have missing pieces, may be inaccurate or may not be able to provide real, valuable insight. Veracity, overall, refers to the level of trust there is in the collected data.

For example, **Facebook posts** with hashtags.

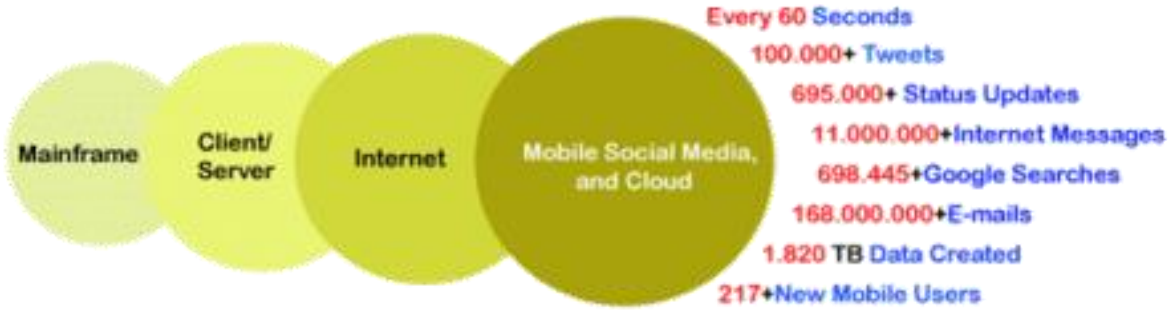
Value: This refers to the value that big data can provide, and it relates directly to what organizations can do with that collected data.



Velocity

Velocity plays an important role compared to others. Velocity creates the speed by which the data is created in **real-time**. It contains the linking of incoming **data sets speeds, rate of change**, and **activity bursts**. The primary aspect of Big Data is to provide demanding data rapidly.

Big data velocity deals with the speed at the data flows from sources like **application logs, business processes, networks, and social media sites, sensors, mobile devices,** etc.



December 2023:

Q1 a) What is the basic difference between traditional RDBMS and Hadoop? [5]

Traditional Data	Big Data
Traditional data is generated at the enterprise level.	Big data is generated outside the enterprise level.
Its volume ranges from Gigabytes to Terabytes.	Its volume ranges from Petabytes to Zettabytes or Exabytes.
Traditional database system deals with structured data.	Big data system deals with structured, semi structured,database, and unstructured data.
Traditional data is generated per hour or per day or more.	But big data is generated more frequently, mainly per seconds.
Traditional data source is centralized and it is managed in centralized form.	Big data source is distributed and it is managed in distributed form.
Data integration is very easy.	Data integration is very difficult.
Normal system configuration is capable of processing traditional data.	High system configuration is required to process big data.
The size of the data is very small.	The size is more than the traditional data size.
Traditional database tools (RDBMS) are required to perform any database operation.	Special kinds of database tools are required to perform any database schema-based operation. (Hadoop, MongoDB, NoSQL)
Normal functions can manipulate data.	Special kinds of functions can manipulate data.
Its data model is strict schema based and it is static.	Its data model is a flat schema based and it is dynamic.
Traditional data is stable and inter relationship.	Big data is not a stable and unknown relationship.
Traditional data is in manageable volume.	Big data is in huge volume which becomes unmanageable.

It is easy to manage and manipulate the data.	It is difficult to manage and manipulate the data.
Its data sources include ERP transaction data, CRM transaction data, financial data, organizational data, web transaction data etc.	Its data sources include social media, device data, sensor data, video, images, audio etc.

b) What are the 3 V's of big data? Give two big data case studies indicating respective V's with justification. [5]

Refer to V's from the previous solution. Pick any 2 case studies from the following

1. **Netflix: Personalization with Volume and Variety**

- **Challenge:** With millions of subscribers and billions of hours watched, Netflix deals with a massive **volume** of data on viewing habits, ratings, and searches. This data is also highly **varied**, including structured data (ratings) and unstructured data (search queries, viewing behavior).

- **Solution:** Netflix utilizes Hadoop to store and process this vast amount of data. Their recommendation engine analyzes user data to suggest personalized content, significantly impacting user engagement.

- **Justification:** The sheer volume of user data across diverse categories (movies, shows, documentaries) necessitates a scalable solution like Hadoop. This allows Netflix to analyze not just what users watch but also how and when they watch, providing valuable insights for personalization.

2. **Fraud Detection in Financial Services: Velocity and Variety**

- **Challenge:** Financial institutions constantly battle fraudulent transactions. Fraudulent activities often involve a high **velocity** of transactions occurring in real-time across various channels (online banking, credit cards). This data can be **varied**, including transaction details, location data, and user behavior patterns.

- **Solution:** Banks leverage Hadoop's ability to handle real-time data streams to identify suspicious transactions. Machine learning algorithms analyze transaction data to detect anomalies and patterns indicative of fraud, allowing for swift intervention.

- **Justification:** The speed and volume of transactions require real-time processing for effective fraud detection. Hadoop's ability to handle diverse data like transaction details and location data empowers banks to build more comprehensive fraud detection models.

3. **Precision Medicine: Volume, Variety, and Velocity for Healthcare Insights**

- **Challenge:** The healthcare industry generates a massive amount of patient data, including medical records, imaging scans, and sensor data from wearable devices. This data is highly **varied** (structured, unstructured, semi-structured) and constantly growing in **volume**. Additionally, real-time analysis of patient data streams (**velocity**) holds immense potential for personalized medicine and improved patient outcomes.
- **Solution:** Healthcare institutions are increasingly using Hadoop to store, manage, and analyze this complex data. By integrating patient data from various sources, researchers can identify patterns and trends that aid in early disease detection, personalized treatment plans, and even drug discovery.
- **Justification:** Hadoop's ability to handle all three V's of big data is crucial in this instance. The vast amount of patient data, combined with real-time sensor information, necessitates a scalable and versatile platform for analysis. This allows healthcare professionals to gain a more holistic view of a patient's health and make data-driven decisions for better care.

4. **Weather Prediction: Velocity and Volume for Mitigating Climate Risks**

- **Challenge:** Weather forecasting relies on a massive influx of data (**volume**) from weather stations, satellites, and radar systems in real-time (**velocity**). This constant stream of data requires rapid processing and analysis to generate accurate weather predictions.
- **Solution:** Meteorological agencies are adopting Hadoop to manage and analyze this ever-growing volume of weather data. By processing data in real-time, meteorologists can create more precise weather models, leading to improved forecasting accuracy.
- **Justification:** The rapid flow of weather data necessitates real-time processing for accurate predictions. Hadoop's ability to handle the sheer volume of data from various sources allows meteorologists to build more comprehensive weather models and issue timely warnings in case of severe weather events.

5. **Social Media Analytics: Volume and Variety for Understanding Customer Trends**

- **Challenge:** Social media platforms generate a vast amount of data (**volume**) in various formats (**variety**), including text posts, images, videos, and user interactions. Extracting meaningful insights from this diverse data can be a challenge.
- **Solution:** Social media companies utilize Hadoop to store and analyze this massive dataset. By analyzing user behavior and sentiment, they can identify trends, understand customer preferences, and even predict market shifts. This information is valuable for targeted advertising and product development.
- **Justification:** The sheer volume and variety of data generated on social media platforms necessitate a scalable and versatile solution. Hadoop allows social media companies to analyze not just the content itself but also the underlying emotions and user interactions, providing valuable insights into consumer behavior.

May 2023:

Q1 a) Distinguish between Name node and Data node. **[5]**

NameNode: The HDFS Orchestrator

- **Central Authority:** The NameNode serves as the authoritative directory service for the HDFS. It maintains the file system namespace, a comprehensive hierarchical structure that represents all files and directories stored within HDFS. This namespace includes critical metadata such as file permissions, ownership, and timestamps for efficient management and access control.
- **Master of Placement and Retrieval:** Beyond acting as a directory service, the NameNode meticulously tracks the location of each data block (a segmented portion of a file) across the distributed storage units (DataNodes) within the HDFS cluster. This allows the NameNode to efficiently orchestrate read and write requests by directing clients to the appropriate DataNodes for data access.
- **Client Interface:** As the central authority, the NameNode serves as the primary point of contact for all client applications interacting with the HDFS. Client applications seeking to create, delete, rename, or access files submit their requests to the NameNode, which then acts as the intermediary, issuing the necessary commands to the relevant DataNodes for execution.
- **Replication Manager:** Data redundancy is paramount for ensuring data availability in the event of DataNode failures. The NameNode plays a critical role in this process by determining the replication factor. This configurable parameter specifies the number of replicas of each data block to be stored across different DataNodes. By managing replication, the NameNode fosters data durability and enables the system to recover seamlessly from hardware issues within the cluster.

DataNode: The HDFS Storage Workhorse

- **Dedicated Storage Units:** Unlike the NameNode, which focuses on metadata management and orchestration, DataNodes serve as the dedicated storage units within the HDFS cluster. Their primary function is to store the actual data blocks that comprise the various files. These data blocks represent the raw bytes of information that constitute the content of the user's files.
- **Executing Instructions:** DataNodes operate under the direction of the NameNode. They receive commands from the NameNode to perform specific tasks such as creating new data blocks for storing incoming data, deleting redundant blocks to optimize storage space, or replicating existing blocks to maintain the configured replication factor as instructed by the NameNode.
- **Maintaining Heartbeats:** To ensure the NameNode has an accurate and up-to-date

picture of the overall HDFS health, DataNodes transmit periodic heartbeats. These heartbeats serve as status updates, informing the NameNode about the DataNode's operational state and the data blocks currently stored within its local storage.

- **Singular Focus on Data Storage:** DataNodes strictly adhere to a block-oriented approach. They are unconcerned with the overall file system structure or the semantics of the data itself. Their primary function is to faithfully store and retrieve data blocks based on the directives received from the NameNode.

Parameter	**NameNode**	**DataNode**
Role	Central authority	Distributed storage unit
Function	Manages file system namespace, directs data access, manages replication	Stores data blocks, executes NameNode instructions
Data Stored	Metadata (file system structure, permissions, locations)	Actual data blocks (raw bytes of files)
Client Interaction	Primary point of contact for client applications	None (operates under NameNode instructions)
Focus	Orchestration, management, file system hierarchy	Data storage and retrieval (block-oriented)
Communication	Receives requests from clients, instructs DataNodes	Sends heartbeats to NameNode, receives instructions
Redundancy	Not replicated (single point of failure)	Can be replicated for fault tolerance

c) Mention four characteristics of big data. Elaborate these characteristics with respect to social media websites. **[5]**

Big Data in the context of social media websites:

Volume:

- **Social media platforms generate a staggering amount of data every day.** This includes text posts, images, videos, audio clips, location data, and user interactions (likes, shares, comments). The sheer volume of this data makes traditional data storage and processing methods inadequate.
- **Example:** Facebook processes billions of messages, likes, and post uploads daily.

Veracity:

- **The accuracy and trustworthiness of social media data can be a challenge.** Fake news, misinformation, and spam can pollute the data pool. Social media platforms

employ various techniques to verify user identities and flag suspicious content, but ensuring complete veracity remains an ongoing battle.

- **Example:** Social media bots can artificially inflate the volume of likes and shares, making it difficult to gauge the authenticity of user engagement.

Variety:

- **Social media data comes in a multitude of formats.** Text posts, images, videos, audio recordings, and even emojis all contribute to the data landscape. This variety necessitates a flexible data management system that can handle structured, semi-structured, and unstructured data formats.
- **Example:** A single social media post might contain text, hashtags, an image, and location data, all requiring different processing techniques for analysis.

Value:

- **Extracting meaningful insights from the vast amount of social media data is crucial to unlock its value.** Businesses can leverage this data for targeted advertising, market research, and understanding customer sentiment. Additionally, social media data can be used for social good, such as tracking disease outbreaks or identifying potential humanitarian crises.
- **Example:** By analyzing user posts and interactions, a clothing brand can identify trending styles and preferences, informing their design and marketing strategies.

Velocity:

- **Social media data is constantly being generated in real-time.** Users are constantly posting, sharing, and commenting, creating a high-velocity data stream. Social media platforms need to be able to process and analyze this data quickly to gain real-time insights and respond to trends or events as they unfold.
- **Example:** During a breaking news event, social media sentiment analysis can be used to gauge public opinion in real-time, allowing organizations to tailor their communication strategies accordingly.

In conclusion, the 5 V's of Big Data (Volume, Veracity, Variety, Value, and Velocity) are all crucial aspects to consider when managing and analyzing social media data. Social media platforms generate a massive amount of data in various formats, with varying degrees of accuracy, and at an ever-increasing speed. By effectively addressing these challenges, valuable insights can be extracted to benefit businesses, researchers, and society as a whole.

December 2022:

Q1. c) Why is HDFS more suited for applications having large datasets and not when there are small files? Elaborate. [5]

HDFS: A Champion for Large Datasets

The Hadoop Distributed File System (HDFS) reigns supreme in the realm of big data due to its architectural choices that optimize performance for massive datasets. However, its strengths become limitations when dealing with a multitude of small files. Let's delve into the factors that contribute to HDFS's prowess with large datasets:

- **Block-Based Advantage:** HDFS employs a block-based storage strategy. Data is segmented into sizable blocks (typically ranging from 64MB to 128MB by default). This approach significantly reduces metadata overhead compared to storing each tiny file individually. When dealing with large files, the metadata becomes a negligible fraction of the overall data size. This minimizes storage space wasted on metadata and streamlines data management.

- **Parallel Processing Powerhouse:** HDFS integrates seamlessly with parallel processing frameworks like MapReduce. This framework distributes tasks across numerous compute nodes within a cluster, enabling significantly faster processing of large files. By dividing the work into smaller chunks, HDFS leverages the collective power of the cluster to handle massive datasets efficiently. This parallel processing capability makes HDFS ideal for large-scale data analytics tasks.

- **Scalability for Growth:** HDFS boasts horizontal scalability, allowing for seamless addition of more nodes to the cluster as your data storage needs expand. This ability to scale storage capacity and processing power in tandem makes HDFS an ideal solution for managing ever-growing datasets. As your data volume increases, you can simply add more nodes to the cluster, ensuring efficient storage and processing capabilities.

Small Files: A Challenge for HDFS

While HDFS excels at handling large files, it encounters several hurdles when confronted with a significant number of small files. Let's explore the reasons behind this performance bottleneck:

- **Metadata Burden:** For small files, the metadata (file information stored by the NameNode) can be just as large as the file itself. This creates a substantial metadata overhead, not only consuming valuable storage space but also impacting overall performance. The sheer number of small files translates to a significant increase in metadata that the NameNode needs to manage, leading to inefficiencies. This metadata

overhead can become a significant bottleneck for the system, hindering its ability to handle numerous small files effectively.

- **I/O Bottleneck:** Accessing a plethora of small files necessitates a substantial number of I/O operations (reading/writing) on the DataNodes. This frequent disk access drastically slows down performance compared to reading a large file, which requires fewer I/O operations. The constant barrage of I/O requests for small files creates a bottleneck, hindering HDFS's ability to handle them efficiently. The inherent overhead associated with numerous I/O operations significantly impacts performance for small file workloads.

- **NameNode Under Strain:** The NameNode serves as the central authority, managing the entire file system namespace, including all files and their locations. With a massive influx of small files, the NameNode's memory usage can skyrocket. This excessive memory consumption can significantly impact its performance and potentially become a bottleneck for the entire system. The NameNode's capacity to manage the namespace efficiently can be overwhelmed by a large number of small files, limiting its ability to handle them effectively.

Alternative Approaches for Small File Management

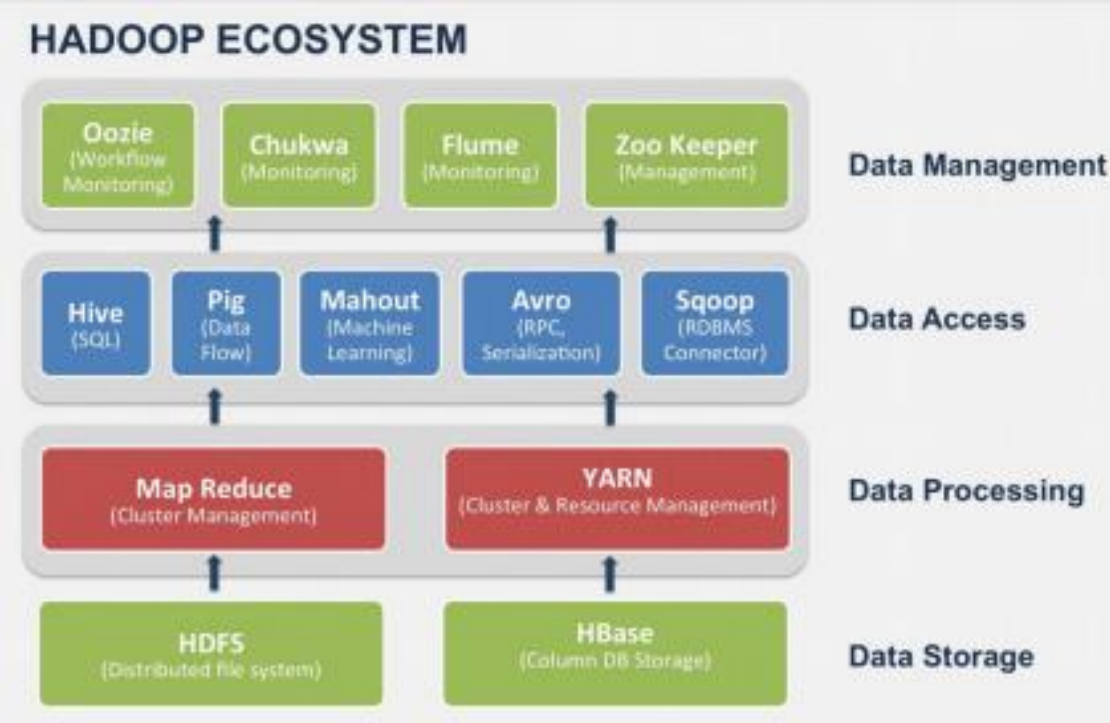
Here are some strategies to consider when dealing with small files within a big data ecosystem:

- **Archiving for Less Frequently Used Data:** If small files are not accessed frequently, consider archiving them to a separate file system specifically designed for small files. This approach frees up valuable storage space within HDFS and allows it to focus on its strength: handling large datasets efficiently. By archiving less frequently accessed small files, you can optimize storage utilization within HDFS.

- **Data Serialization for Efficiency:** In some cases, it might be beneficial to combine or serialize small files into a larger file format. This can significantly improve storage efficiency within HDFS by reducing metadata overhead and minimizing I/O operations. By transforming numerous small files into a single, larger file, you can leverage HDFS's block-based storage and parallel processing capabilities more effectively. Serialization can offer a significant performance improvement for specific use cases involving small files.

- **Specialized Tools for Small File Workloads:** For applications primarily dealing with small files, consider distributed file systems like Apache Kudu or Ceph. These file systems are specifically designed to handle a multitude of small files efficiently, offering optimizations that address the challenges HDFS faces in such scenarios. By choosing the right tool for the job, you can ensure optimal performance for your big data application, regardless of file size. Utilizing specialized tools can be a more efficient approach for managing large numbers of small files within a big data environment.

Q4 a) What are the Core Hadoop components? Explain in detail. [10]



MapReduce

- MapReduce is a programming model for processing large data sets with a parallel, distributed algorithm on a cluster
- How does it solve our previously mentioned problems?

MapReduce is highly scalable and can be used across many computers.

- Many small machines can be used to process jobs that normally could not be processed by a large machine.

How MapReduce works?

- MapReduce is a method for distributing a task across multiple nodes
- Each node processes data stored on that node
  - Where possible
- Consists of two phases:
  - Map
  - Reduce
- Two phases: Map and Reduce
- Large scale parallel data processing.
- Mapreduce provides:
  - Automatic parallelization and distribution
  - Fault-tolerance
  - Status and monitoring
- Map phase
  - Processes input key/value pair

- Produces set of intermediate pairs

• Reduce Phase
- Combines all intermediate values for a particular key
- Produces a set of merged output values (usually just one)

Features of MapReduce

• Automatic parallelization and distribution
• Fault tolerance
• Status and monitoring tools
• A clean abstraction for programmers
- MapReduce programs are usually written in Java
- Can be written in any language using *Hadoop Streaming* (see later)
- All of Hadoop is written in Java

• MapReduce abstracts all the 'housekeeping' away from the developer
- Developer can concentrate simply on working the Map and Reduce functions

Working of Hive:-

Translates HiveQL statements into a set of MapReduce Jobs, which are then executed on a Hadoop Cluster

PIG

• is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs.
• Pig is widely used and accepted by Yahoo!, Twitter, Netflix etc
• Map reduce programmers need to think of programmes in terms of map and reduce functions.
• It normally requires JAVA programmers. • Pig requires its own scripting language named Pig Latin'.

Features

• Join Datasets.
• Sort Datasets.
• Data Types.
• Group By.

PIG Components

Pig Latin :

• Command based language.
• Designed specifically for data transformation and flow expression**.**

Execution Environment :

• The environment in which Pig Latin commands are executed .
• Currently it supports local and Hadoop modes.

Pig compiler converts Pig Latin into Map Reduce

- Compiler strives to optimize execution.

Execution Modes

- There are 2 execution modes supported by Pig :
- To enter into different modes :

$Pig –x local (for local mode)

$Pig –x mapreduce (for hadoop mode)

HBASE

- HBase is a column-oriented database built on the top of HDFS.
- HBase is a Hadoop application used to read/write random access to very large databases.
- HBase comes at a scaling problem from the opposite direction.
- It is built from the ground-up to scale linearly just by adding nodes.

Features

- Scalability.
- Distributed
- Variable schema
- Auto partitioning
- Convenient base classes for backing Hadoop.
- Mapreduce jobs can run on Hbase tables.
- Extensible jruby-based shell

SQOOP

- The name Sqoop is derived from SQL + Hadoop.
- Sqoop = SQL to Hadoop and Hadoop to SQL.
- Sqoop is a tool designed to transfer data between Hadoop and relational database servers. It is used to import data from relational databases like MySql, Oracle to Hadoop HDFS and export data from Hadoop HDFS to relational databases like MySql, Oracle.
- It is provided by Apache software Foundation.

Sqoop – Import :

- Importing data from MySql database to Hadoop HDFS helps in importing individual tables from RDBMS to HDFS.
- Each row in a table is treated as a record in HDFS.
- All records are stored as text data in text files or binary data in sequence files.

Sqoop – Export :

- Export command helps us to export content from HDFS to RDBMS.
- The files which are given as input to the Sqoop contain records, which are called rows in the table.
- Those are read and parsed into a set of records and delimited with user specified delimiter.