SUMMER 2022 (3170722) Big Data Analytics [Q.1]

(a) What is Big data Analytics? List applications of it.

The process of analysis of large volumes of diverse data sets, using advanced analytic techniques is referred to as **Big Data Analytics**.

These diverse data sets include structured, semi-structured, and unstructured data, from different sources, and in different sizes from terabytes to zettabytes. We also reckon them as big data.

Big Data is a term that is used for data sets whose size or type is beyond the capturing, managing, and processing ability of traditional rotational databases. The database required to process big data should have low latency that traditional databases don't have.

List applications of it.

Healthcare

- Big Data has already started to create a huge difference in the healthcare sector.
- With the help of predictive analytics, medical professionals and HCPs are now able to provide personalized healthcare services to individual patients.
- Apart from that, fitness wearable's, telemedicine, remote monitoring – all powered by Big Data and AI – are helping change lives for the better.

Academia

- Big Data is also helping enhance education today.
- Education is no more limited to the physical bounds of the classroom – there are numerous online educational courses to learn from.
- Academic institutions are investing in digital courses powered by Big Data technologies to aid the all-round development of budding learners.

Banking

- The banking sector relies on Big Data for fraud detection.
- Big Data tools can efficiently detect fraudulent acts in real-time such as misuse of credit/debit cards, archival of inspection tracks, faulty alteration in customer stats, etc.

Manufacturing

- According to TCS Global Trend Study, the most significant benefit of Big Data in manufacturing is improving the supply strategies and product quality.
- In the manufacturing sector, Big data helps create a transparent infrastructure, thereby, predicting uncertainties and incompetencies that can affect the business adversely.

IT

- One of the largest users of Big Data, IT companies around the world are using Big Data to optimize their functioning, enhance employee productivity, and minimize risks in business operations.
- By combining Big Data technologies with ML and AI, the IT sector is continually powering innovation to find solutions even for the most complex of problems.

Retail

- Big Data has changed the way of working in traditional brick and mortar retail stores.
- Over the years, retailers have collected vast amounts of data from local demographic surveys, POS scanners, RFID, customer loyalty cards, store inventory, and so on.
- Now, they've started to leverage this data to create personalized customer experiences, boost sales, increase revenue, and deliver outstanding customer service.
- Retailers are even using smart sensors and Wi-Fi to track the movement of customers, the most frequented aisles, for how long customers linger in the aisles, among other things.
- They also gather social media data to understand what customers are saying about their brand, their services, and tweak their product design and marketing strategies accordingly.

Transportation

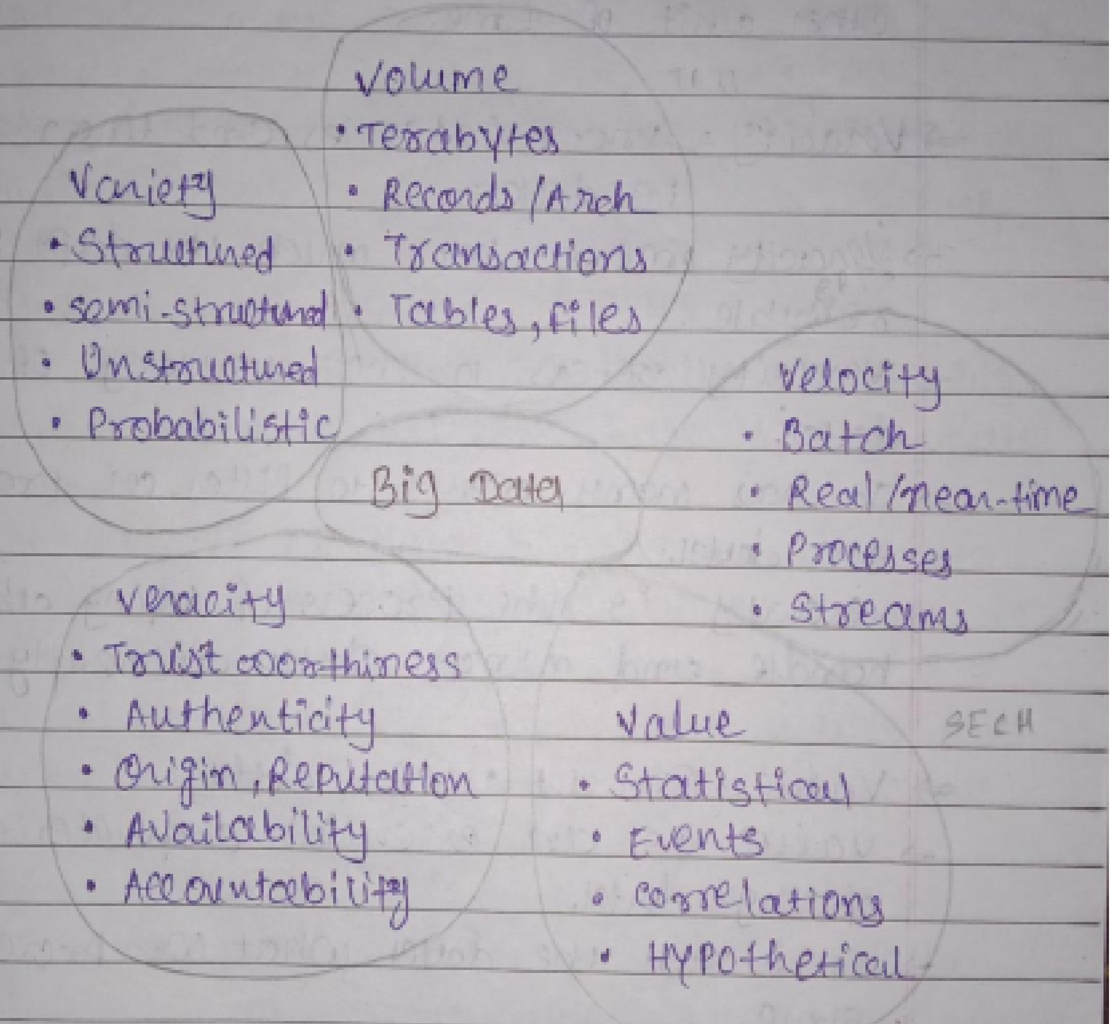
- Big Data Analytics holds immense value for the transportation industry.
- In countries across the world, both private and government-run transportation companies use Big Data technologies to optimize route planning, control traffic, manage road congestion, and improve services.
- Additionally, transportation services even use Big Data to revenue management, drive technological innovation, enhance logistics, and of course, to gain the upper hand in the market.

(b) Explain 4 V's of Big data.

→ Big data is a collection of data that is huge in volume, yet growing exponentially with time.

→ Big data is a combination of Structured, Semi-structured and unstructured data.

→ .



→ Volume - Huge amount of data

→ Volume refers to the amount of data that you have.

→ we measure the volume of our data in Gegabytes, zettabytes(ZB), and Yottabytes (YB).

→ According to the industry tends, the volume of the data will rise substantially in the coming years. largly

speed of generating data.

⇒ Velocity = High speed of accumulation of data:-

→ The speed at which companies receive, store and manage data - e.g., the specific number of social media posts or ~~seach~~ search queries received within a day, hour or other unit of time.

⇒ Veracity = Inconsistencies and uncertainty in data:-

→ Veracity means how much the data is reliable / valid.

→ Veracity refers to the accuracy of your data.

→ It has many ways to filter or translate the data.

→ veracity is the process of being able to handle and manage data efficiently.

⇒ Value:- Extract useful data. (Res)

→ Value is an essential characteristic of big data.

→ It is not the data what we process or store.

→ It is valuable and reliable data that we store, process and also analyze.

⇒ Variety:- different formats of data.

→ Big data can be structured, unstructured & semi-structured that are being collected from different sources.

(c) What is Hadoop? Briefly explain the core components of it.

Hadoop is a framework that uses distributed storage and parallel processing to store and manage big data. It is the software most used by data analysts to handle big data, and its market size continues to grow. There are three components of Hadoop:

1. Hadoop HDFS - Hadoop Distributed File System (HDFS) is the storage unit.
2. Hadoop MapReduce - Hadoop MapReduce is the processing unit.
3. Hadoop YARN - Yet Another Resource Negotiator (YARN) is a resource management unit.

Hadoop HDFS

⇨ Data is stored in a distributed manner in HDFS. There are two components of HDFS - name node and data node. While there is only one name node, there can be multiple data nodes.

⇨ HDFS is specially designed for storing huge datasets in commodity hardware. An enterprise version of a server costs roughly $10,000 per terabyte for the full processor. In case you need to buy 100 of these enterprise version servers, it will go up to a million dollars.

⇨ Hadoop enables you to use commodity machines as your data nodes. This way, you don't have to spend millions of dollars just on your data nodes. However, the name node is always an enterprise server.

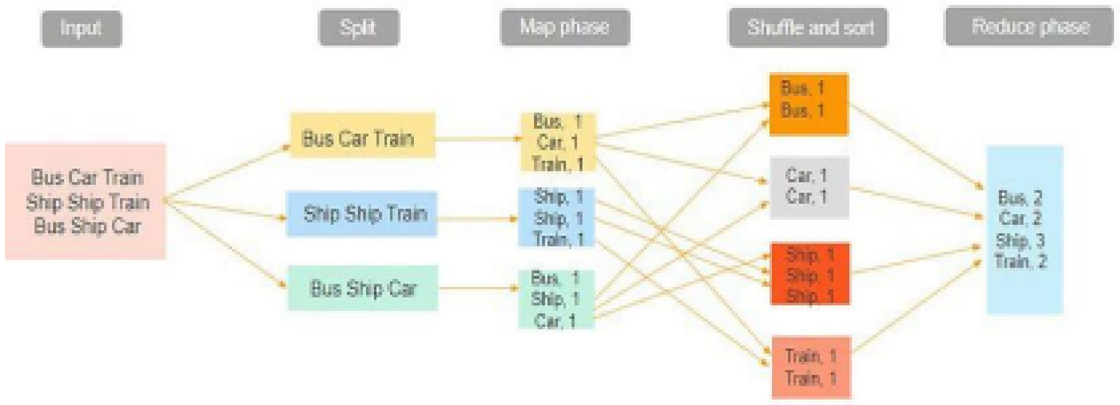
⇨ <u>Features of HDFS</u>

⇨ Provides distributed storage
⇨ Can be implemented on commodity hardware
⇨ Provides data security
⇨ Highly fault-tolerant - If one machine goes down, the data from that machine goes to the next machine

Hadoop MapReduce

⇨ Hadoop MapReduce is the processing unit of Hadoop. In the MapReduce approach, the processing is done at the slave nodes, and the final result is sent to the master node.

⇨ A data containing code is used to process the entire data. This coded data is usually very small in comparison to the data itself. You only need to send a few kilobytes worth of code to perform a heavy-duty process on computers.



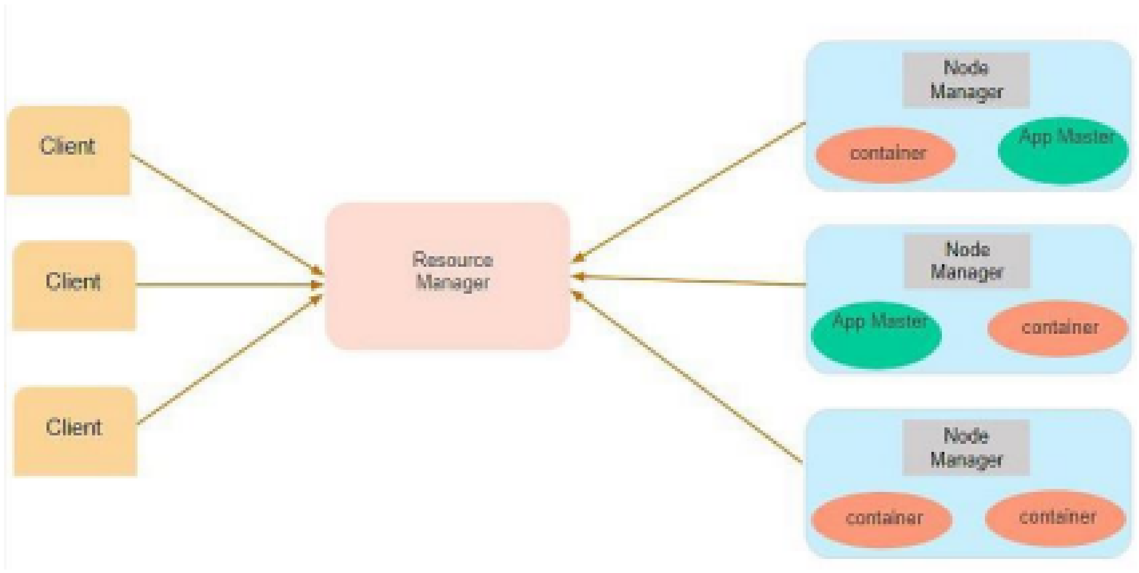
⇨ The input dataset is first split into chunks of data. In this example, the input has three lines of text with three separate entities - “bus car train,” “ship ship train,” “bus ship car.” The dataset is then split into three chunks, based on these entities, and processed parallelly.

⇨ In the map phase, the data is assigned a key and a value of 1. In this case, we have one bus, one car, one ship, and one train.

⇨ These key-value pairs are then shuffled and sorted together based on their keys. At the reduce phase, the aggregation takes place, and the final output is obtained.

Hadoop YARN

⇨ Hadoop YARN stands for Yet Another Resource Negotiator. It is the resource management unit of Hadoop and is available as a component of Hadoop version 2.
⇨ Hadoop YARN acts like an OS to Hadoop. It is a file system that is built on top of HDFS.
⇨ It is responsible for managing cluster resources to make sure you don't overload one machine.
⇨ It performs job scheduling to make sure that the jobs are scheduled in the right place



⇨ Suppose a client machine wants to do a query or fetch some code for data analysis. This job request goes to the resource manager (Hadoop Yarn), which is responsible for resource allocation and management.

[Q.2]

(a) How is Big data and Hadoop related?

Hadoop is a kind of framework that can handle the huge volume of Big Data and process it, whereas Big Data is just a large volume of the Data which can be in unstructured and structured data.

Big data and Hadoop are two different concepts but they are inter related.

In simple terms Big data is massive amount of data and Hadoop is the framework which is used to store, process, and analyze this data.

Data has evolved rapidly in the last decade. In the earlier days, there were less data generating sources and there was only one type of data being generated that was the structured data.

Only a single traditional database was enough to store and process this data. But as time passed by, the number of sources increased and data was generated in large amount all across the globe.

This generated data was of three types, structured data like excel records, semi structured data like mails and unstructured data like videos and images
It was now difficult for a traditional database to store, process and analyze this data.

This data is termed as big data, data which is huge to be stored, processed or analyzed using traditional databases.

Hadoop is a framework that manages big data storage in a distributed way and processes it parallelly.

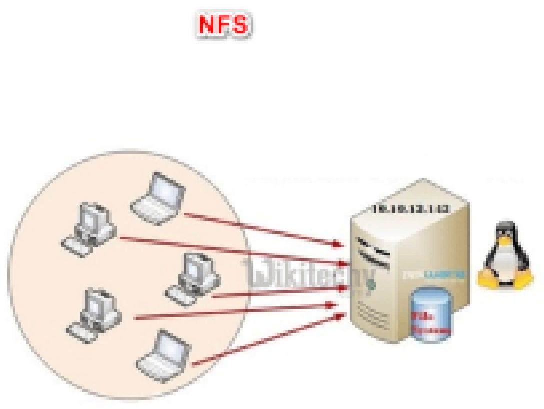
Hadoop has 3 components - HDFS, MapReduce, and YARN. Hadoop Distributed File System (HDFS) is specially designed for storing huge datasets in commodity hardware.

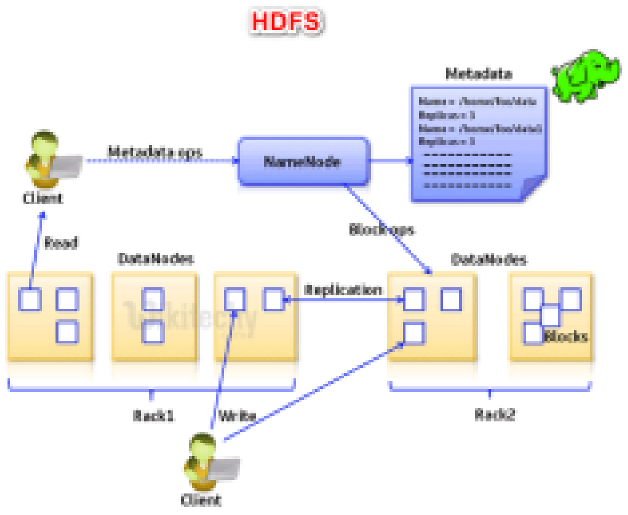
Hadoop MapReduce is a programming technique where huge data is processed in a parallel and distributed fashion.

YARN is similar to an OS of Hadoop.

(b) How HDFS is different from traditional NFS?

NFS (Network File System) is one of the oldest and popular distributed file storage systems. Whereas HDFS (Hadoop Distributed File System) is the recently used and popular one to handle big data.





NFS	HDFS
NFS can store and process small amount of data.	HDFS is mainly use to store and process big data.
Data is stored on a single dedicated hardware.	The data blocks are distributed on the local drives of hardware.
No reliability, data is not available in the case of machine failure.	Data is stored reliably, data is available even after machine failure.
NFS runs on single machine, no chances of data redundancy.	HDFS runs on a cluster of different machines, data redundancy may occur due to replication protocol.
Workgroup.	Larger than AFS.
Single Domain.	Multi Domain.
Client identity is trusted by default.	Client identity is what os tells. No Kerberos Auth.
Same System calls as of O/S.	Different Calls.Mainly used for non interactive programs.

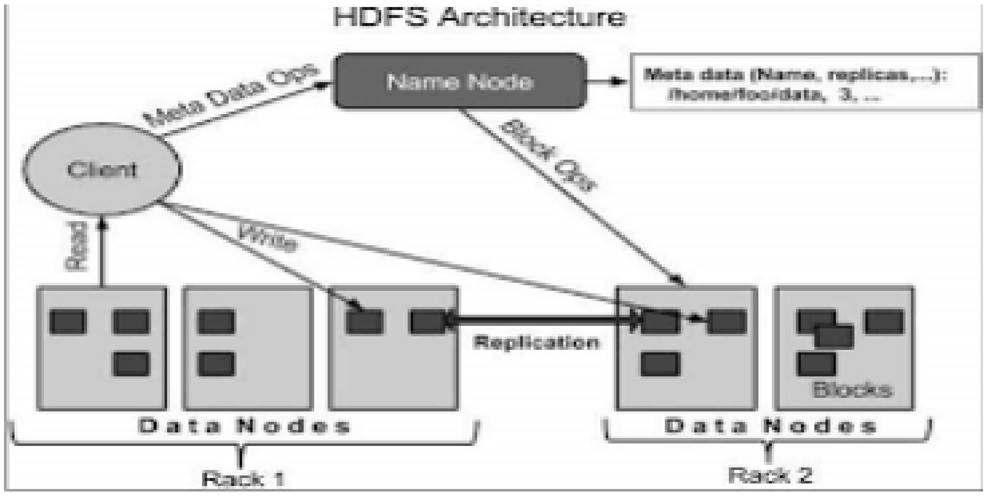
(c) Draw and Explain HDFS architecture. How can you restart NameNode and all the daemons in Hadoop?

This HDFS tutorial by DataFlair is designed to be an all in one package to answer all your questions about HDFS architecture.

Hadoop Distributed File System(HDFS) is the world's most reliable storage system. It is best known for its fault tolerance **and** high availability**. In this article about** HDFS Architecture Guide**, you can read all about Hadoop HDFS.**

HDFS **stores very large files running on a cluster of commodity hardware. It works on the principle of storage of less number of large files rather than the huge number of small files.**

HDFS stores data reliably even in the case of hardware failure. It provides high throughput by providing the data access in parallel.



Let's discuss each of the nodes in the Hadoop HDFS Architecture in detail.

HDFS NameNode

⇨ NameNode is the centerpiece of the Hadoop Distributed File System. ⇨ It maintains and manages the **file system namespace** and provides the right access permission to the clients.

⇨ The NameNode stores information about blocks locations, permissions, etc. on the local disk in the form of two files:

1. **Fsimage:** Fsimage stands for File System image. It contains the complete namespace of the Hadoop file system since the NameNode creation.
2. **Edit log:** It contains all the recent changes performed to the file system namespace to the most recent Fsimage.

HDFS DataNode

⇨ DataNodes are the slave nodes in Hadoop HDFS. DataNodes are **inexpensive commodity hardware**. They store blocks of a file.

Checkpoint Node

⇨ The Checkpoint node is a node that periodically creates checkpoints of the namespace.

⇨ Checkpoint Node in Hadoop first downloads Fsimage and edits from the Active Namenode.

⇨ Then it merges them (Fsimage and edits) locally, and at last, it uploads the new image back to the active NameNode.

⇨ It stores the latest checkpoint in a directory that has the same structure as the Namenode’s directory.

⇨ This permits the checkpointed image to be always available for reading by the NameNode if necessary.

Backup Node

⇨ A Backup node provides the same checkpointing functionality as the Checkpoint node.

⇨ In Hadoop, Backup node keeps an **in-memory, up-to-date copy** of the file system namespace. It is always synchronized with the active NameNode state.

⇨ It is not required for the backup node in HDFS architecture to download **Fsimage** and **edits files** from the active NameNode to create a checkpoint.

⇨ It already has an up-to-date state of the namespace state in memory.

⇨ The Backup node checkpoint process is more efficient as it only needs to save the namespace into the local Fsimage file and reset edits.

⇨ **NameNode supports one Backup node at a time**.

OR

(c) What is MapReduce? Explain working of various phases of MapReduce with appropriate example and diagram.

A MapReduce is a data processing tool which is used to process the data parallelly in a distributed form.

MapReduce architecture contains two core components as Daemon services responsible for running mapper and reducer tasks, monitoring, and re-executing the tasks on failure.

In Hadoop 2 onwards Resource Manager and Node Manager are the daemon services. When the job client submits a MapReduce job, these daemons come into action.

They are also responsible for parallel processing and fault-tolerance features of

MapReduce jobs.

In Hadoop 2 onwards resource management and job scheduling or monitoring functionalities are segregated by YARN (Yet Another Resource Negotiator) as different daemons.

Compared to Hadoop 1 with Job Tracker and Task Tracker, Hadoop 2 contains a global Resource Manager (RM) and Application Masters (AM) for each application.

Phases of the MapReduce model :

1. Mapper

· It is the first phase of MapReduce programming and contains the coding logic of the mapper function.

· The conditional logic is applied to the 'n' number of data blocks spread across various data nodes.

· Mapper function accepts key-value pairs as input as (k, v), where the key represents the offset address of each record and the value represents the entire record content.

· The output of the Mapper phase will also be in the key-value format as (k', v').

2. Shuffle and Sort

· The output of various mappers (k', v'), then goes into Shuffle and Sort phase. · All the duplicate values are removed, and different values are grouped together based on similar keys.

· The output of the Shuffle and Sort phase will be key-value pairs again as key and array of values (k, v[]).

3. Reducer

- The output of the Shuffle and Sort phase (k, v[]) will be the input of the Reducer phase.
- In this phase reducer function's logic is executed and all the values are aggregated against their corresponding keys.
- Reducer consolidates outputs of various mappers and computes the final job output.
- The final output is then written into a single file in an output directory of HDFS.

4. Combiner

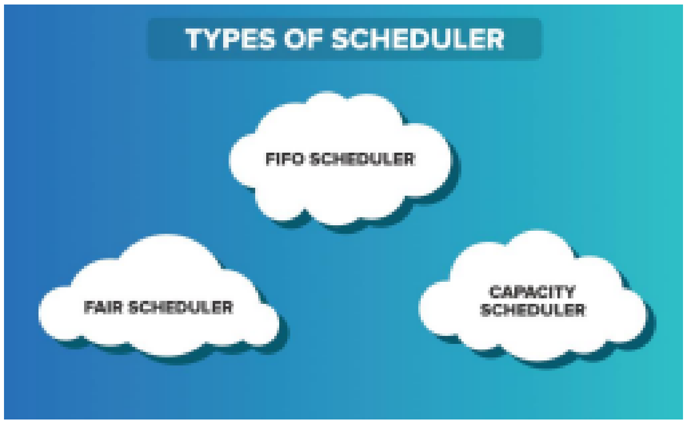
- It is an optional phase in the MapReduce model.
- The combiner phase is used to optimize the performance of MapReduce jobs. · In this phase, various outputs of the mappers are locally reduced at the node level.
- For example, if different mapper outputs (k, v) coming from a single node contains duplicates, then they get combined i.e. locally reduced as a single (k, v[]) output.
- This phase makes the Shuffle and Sort phase work even quicker thereby enabling additional performance in MapReduce jobs.

[Q.3]

(a) What do you mean by job scheduling in Hadoop? List different schedulers in Hadoop.

Job scheduling is the process of allocating system resources to many different tasks by an operating system (OS).

The system handles prioritized job queues that are awaiting CPU time and it should determine which job to be taken from which queue and the amount of time to be allocated for the job.



There are mainly 3 types of Schedulers in Hadoop:

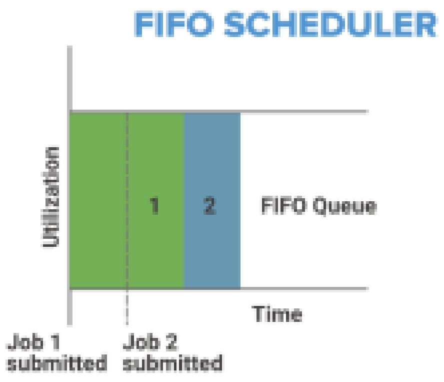
1. FIFO (First In First Out) Scheduler.
2. Capacity Scheduler.
3. Fair Scheduler.

1. FIFO Scheduler

As the name suggests FIFO i.e. First In First Out, so the tasks or application that comes first will be served first. This is the default Scheduler we use in Hadoop.

The tasks are placed in a queue and the tasks are performed in their submission order. In this method, once the job is scheduled, no intervention is allowed.

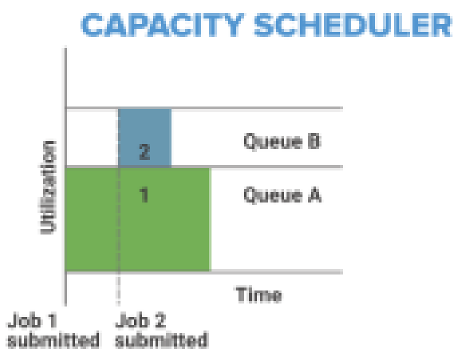
So sometimes the high-priority process has to wait for a long time since the priority of the task does not matter in this method.



2. Capacity Scheduler

In Capacity Scheduler we have multiple job queues for scheduling our tasks. The Capacity Scheduler allows multiple occupants to share a large size Hadoop cluster.

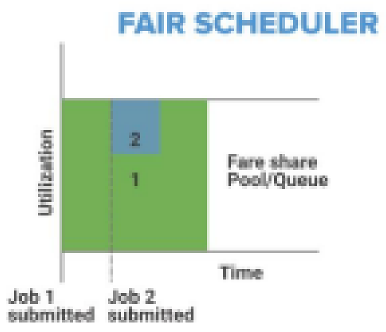
In Capacity Scheduler corresponding for each job queue, we provide some slots or cluster resources for performing job operation.



3. Fair Scheduler

The Fair Scheduler is very much similar to that of the capacity scheduler. The priority of the job is kept in consideration.

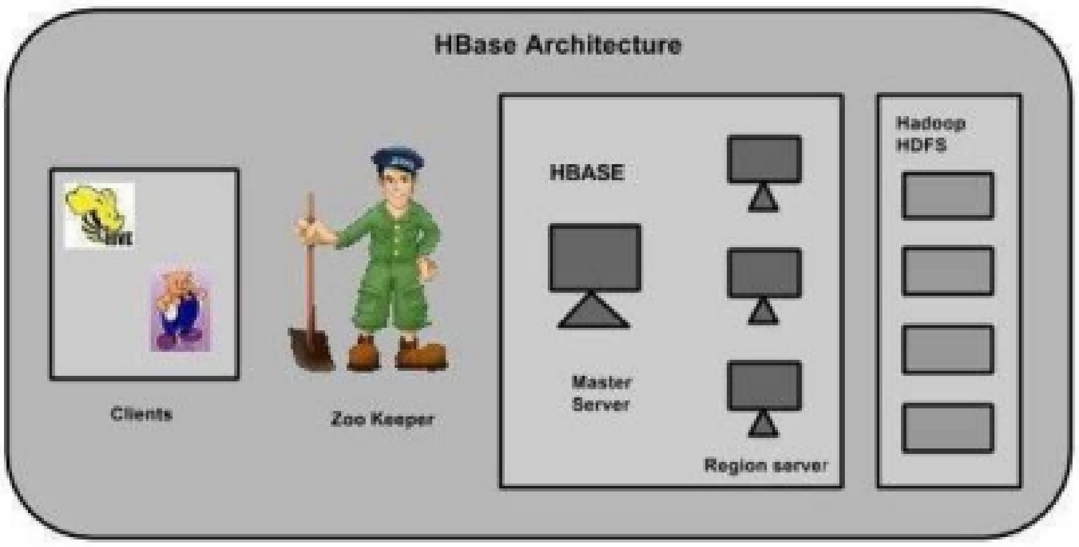
With the help of Fair Scheduler, the YARN applications can share the resources in the large Hadoop Cluster and these resources are maintained dynamically so no need for prior capacity.



(b) What are WAL, MemStore, Hfile and Hlog in HBase?

(c) Explain the architecture of HBase.

HBase, tables are split into regions and are served by the region servers. Regions are vertically divided by column families into “Stores”. Stores are saved as files in HDFS. Shown below is the architecture of HBase.



HBase has three major components: the client library, a master server, and region servers. Region servers can be added or removed as per requirement.

MasterServer

The master server -

- Assigns regions to the region servers and takes the help of Apache ZooKeeper for this task.
- Handles load balancing of the regions across region servers. It unloads the busy servers and shifts the regions to less occupied servers.
- Maintains the state of the cluster by negotiating the load balancing. • Is responsible for schema changes and other metadata operations such as creation of tables and column families.

Regions

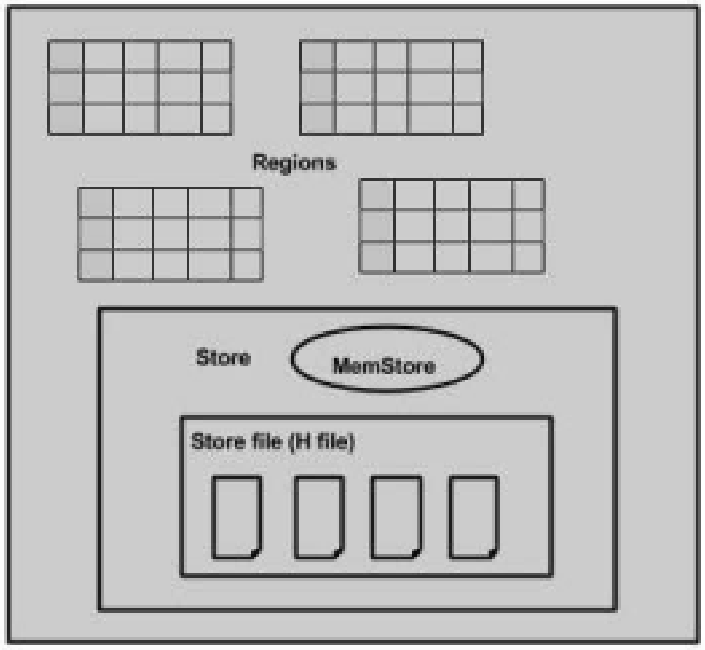
Regions are nothing but tables that are split up and spread across the region servers.

Region server

The region servers have regions that -

- Communicate with the client and handle data-related operations. • Handle read and write requests for all the regions under it.
- Decide the size of the region by following the region size thresholds.

When we take a deeper look into the region server, it contain regions and stores as shown below:



The store contains memory store and HFiles. Memstore is just like a cache memory. Anything that is entered into the HBase is stored here initially. Later, the data is transferred and saved in Hfiles as blocks and the memstore is flushed.

Zookeeper

- Zookeeper is an open-source project that provides services like maintaining configuration information, naming, providing distributed synchronization, etc. • Zookeeper has ephemeral nodes representing different region servers. Master servers use these nodes to discover available servers.
- In addition to availability, the nodes are also used to track server failures or network partitions.
- Clients communicate with region servers via zookeeper.
- In pseudo and standalone modes, HBase itself will take care of zookeeper.

OR

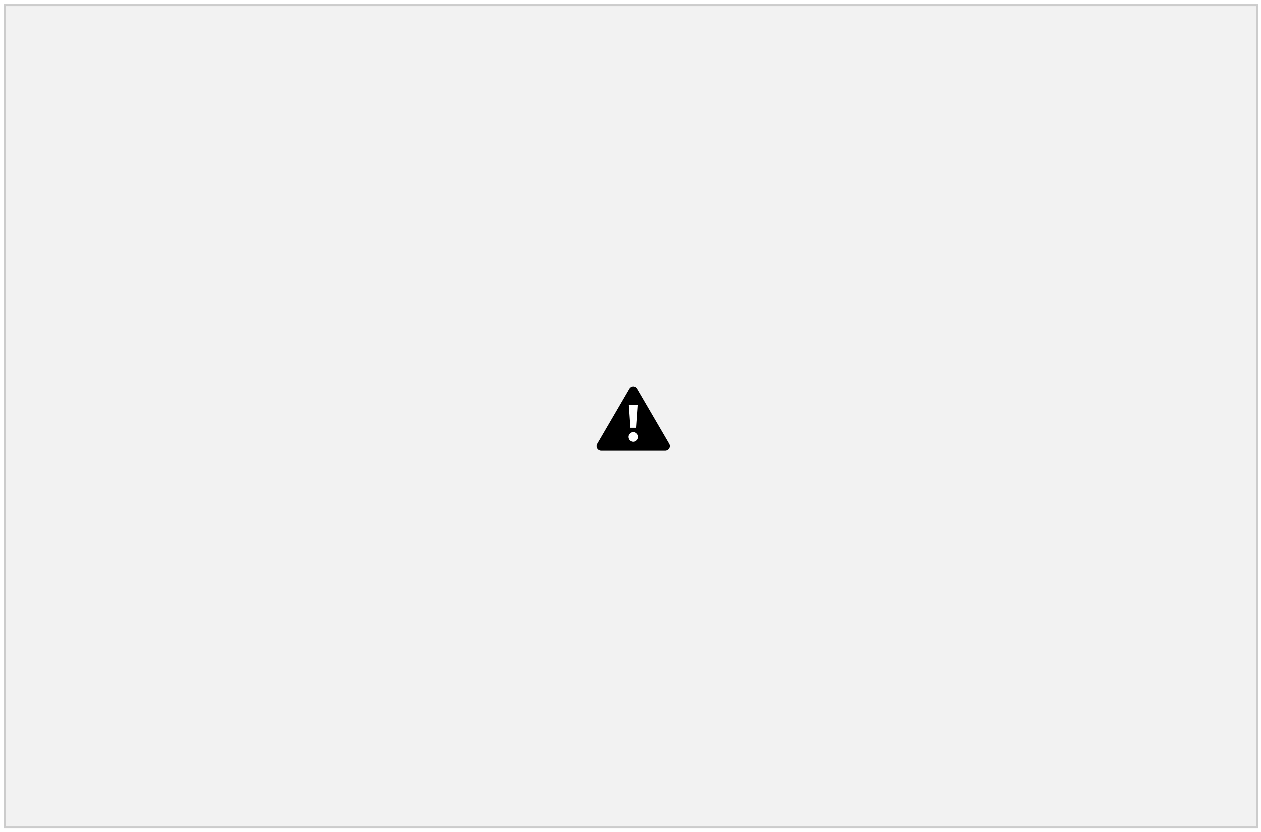
[Q.3]

(a) What is Zookeeper?

Zookeeper is an open-source project that provides services like maintaining configuration information, naming, providing distributed synchronization, etc.
Zookeeper has ephemeral nodes representing different region servers. Master servers use these nodes to discover available servers.
In addition to availability, the nodes are also used to track server failures or network partitions.

Clients communicate with region servers via zookeeper.
In pseudo and standalone modes, HBase itself will take care of zookeeper.

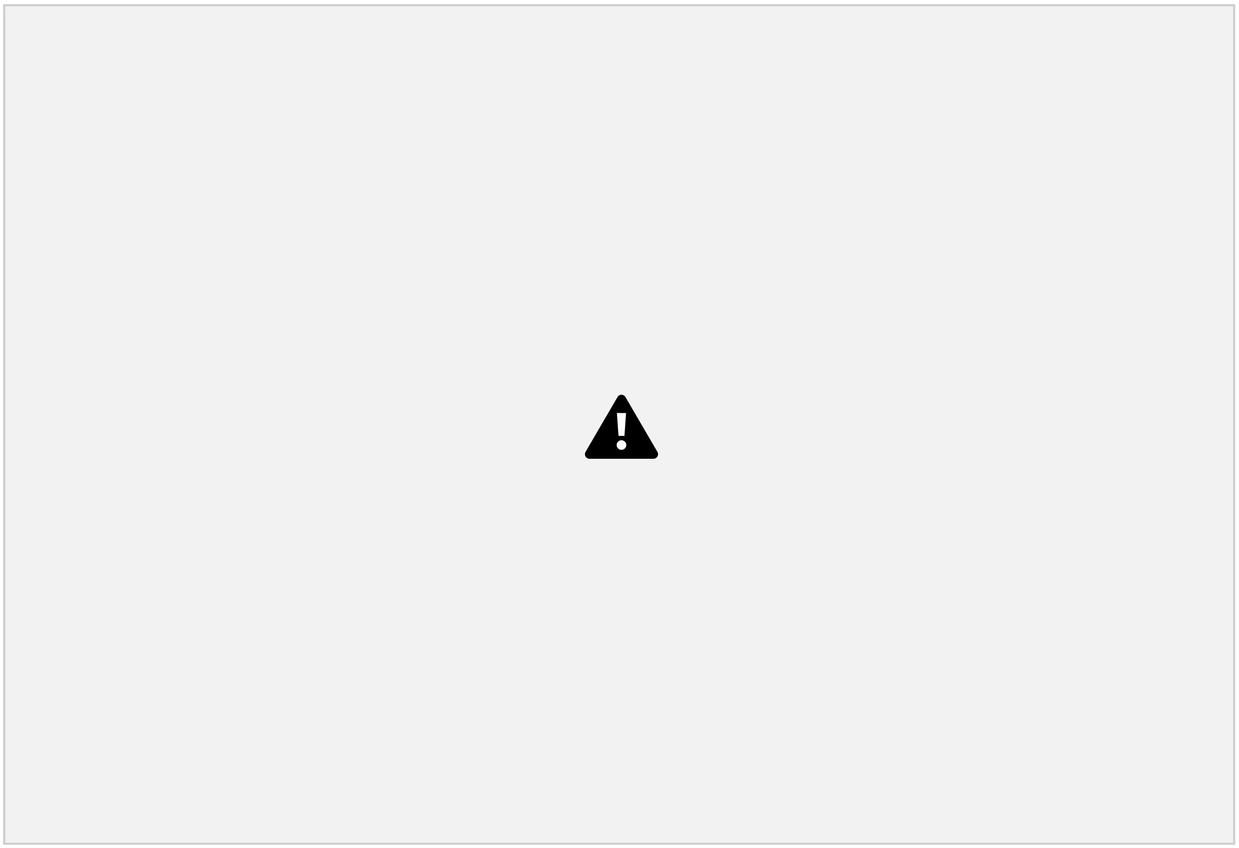
(b) Differentiate between HIVE and HBASE.



(c) What is NoSQL database? List the differences between NoSQL and relational databases. Explain in brief various types of NoSQL databases in practice.

NoSQL originally referring to non SQL or non relational is a database that provides a mechanism for storage and retrieval of data.

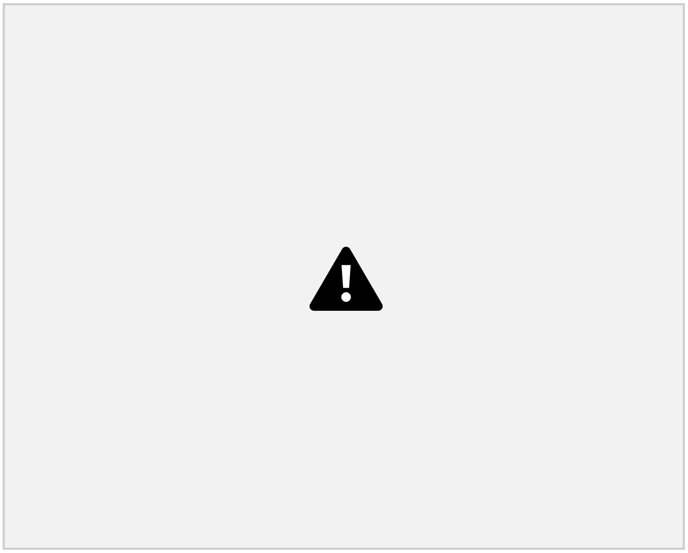
List the differences between NoSQL and relational databases.



Explain in brief various types of NoSQL databases in practice.

Types of NoSQL Database:

Document-based databases
Key-value stores
Column-oriented databases
Graph-based databases



Document-Based Database:

- ⇨ The document-based database is a nonrelational database. Instead of storing the data in rows and columns (tables), it uses the documents to store the data in the database

Key-Value Stores:

- ⇨ A key-value store is a nonrelational database. The simplest form of a NoSQL database is a key-value store. Every data element in the database is stored in key-value pairs.
- ⇨ Every data element in the database is stored in key-value pairs. The data can be retrieved by using a unique key allotted to each element in the database.

Column Oriented Databases:

- ⇨ A column-oriented database is a non-relational database that stores the data in columns instead of rows.
- ⇨ That means when we want to run analytics on a small number of columns, you can read those columns directly without consuming memory with the unwanted data.

Graph-Based databases:

- ⇨ Graph-based databases focus on the relationship between the elements. It stores the data in the form of nodes in the database.
- ⇨ The connections between the nodes are called links or relationships.

[Q.4]

(a) What is Pig?

Pig Represents Big Data as data flows. Pig is a high-level platform or tool which is used to process the large datasets.

It provides a high-level of abstraction for processing over the MapReduce.

It provides a high-level scripting language, known as *Pig Latin* which is used to develop the data analysis codes.

(b) What are the features of MongoDB?

Schema-less Database:

⇨ It is the great feature provided by the MongoDB.

⇨ A Schema-less database means one collection can hold different types of documents in it. Or in other words, in the MongoDB database, a single collection can hold multiple documents and these documents may consist of the different numbers of fields, content, and size.

⇨ It is not necessary that the one document is similar to another document like in the relational databases.

⇨ Due to this cool feature, MongoDB provides great flexibility to databases.

Document Oriented:

⇨ In MongoDB, all the data stored in the documents instead of tables like in RDBMS.

⇨ In these documents, the data is stored in fields(key-value pair) instead of rows and columns which make the data much more flexible in comparison to RDBMS.

⇨ And each document contains its unique object id.

Indexing:

⇨ In MongoDB database, every field in the documents is indexed with primary and secondary indices this makes easier and takes less time to get or search data from the pool of the data.

⇨ If the data is not indexed, then database search each document with the specified query which takes lots of time and not so efficient.

Scalability:

⇨ MongoDB provides horizontal scalability with the help of sharding. ⇨

Sharding means to distribute data on multiple servers, here a large amount of data is partitioned into data chunks using the shard key, and these data chunks are evenly distributed across shards that reside across many physical servers.

⇨ It will also add new machines to a running database.

Replication:

⇨ MongoDB provides high availability and redundancy with the help of replication, it creates multiple copies of the data and sends these copies to a different server so that if one server fails, then the data is retrieved from another server.

Aggregation:

⇨ It allows to perform operations on the grouped data and get a single result or computed result.

⇨ It is similar to the SQL GROUPBY clause.

⇨ It provides three different aggregations i.e, aggregation pipeline, map reduce function, and single-purpose aggregation methods

High Performance:

⇨ The performance of MongoDB is very high and data persistence as compared to another database due to its features like scalability, indexing, replication, etc.

(c) Explain the concept of regions in HBase and storing Big data with HBase.

X

OR

[Q.4]

(a) How MongoDB is better than SQL database?

(b) Explain the Data Model of HBase.

HBase Data Model is a set of components that consists of Tables, Rows, Column families, Cells, Columns, and Versions.

HBase tables contain column families and rows with elements defined as Primary keys.

A column in HBase data model table represents attributes to the objects.

HBase Data Model consists of following elements,

- Set of tables
- Each table with column families and rows
- Each table must have an element defined as Primary Key.
- Row key acts as a Primary key in HBase.
- Any access to HBase tables uses this Primary Key
- Each column present in HBase denotes attribute corresponding to object

(c) Explain Pig data Model in detail and Discuss how it will help for effective data flow.

pig data model includes Pig's data types, how it handles concepts such as missing data, and how you can describe your data to Pig.

Pig has three complex data types: maps, tuples, and bags. All of these types can contain data of any type, including other complex types. So it is possible to have a map where the value field is a bag, which contains a tuple where one of the fields is a map.

Map :

A *map* in Pig is a chararray to data element mapping, where that element can be any Pig type, including a complex type.

The chararray is called a key and is used as an index to find the element, referred to as the value.

Because Pig does not know the type of the value, it will assume it is a bytearray.

Map constants are formed using brackets to delimit the map, a hash between keys and values, and a comma between key-value pairs.

For example, ['name'#'bob', 'age'#55] will create a map with two keys, "name" and "age". The first value is a chararray, and the second is an integer.

Tuple :

A *tuple* is a fixed-length, ordered collection of Pig data elements.

Tuples are divided into *fields*, with each field containing one data element. These elements can be of any type—they do not all need to be the same type.

Tuple constants use parentheses to indicate the tuple and commas to delimit fields in the tuple. For example, ('bob', 55) describes a tuple constant with two fields.

Bag :

A *bag* is an unordered collection of tuples. Because it has no order, it is not possible to reference tuples in a bag by position.

Like tuples, a bag can, but is not required to, have a schema associated with it. In the case of a bag, the schema describes all tuples within the bag.
Bag constants are constructed using braces, with tuples in the bag separated by commas. For example, {('bob', 55), ('sally', 52), ('john', 25)} constructs a bag with three tuples, each with two fields.

Bag is the one type in Pig that is not required to fit into memory. As you will see later, because bags are used to store collections when grouping, bags can become quite large.

Pig has the ability to spill bags to disk when necessary, keeping only partial sections of the bag in memory.

The size of the bag is limited to the amount of local disk available for spilling the bag.

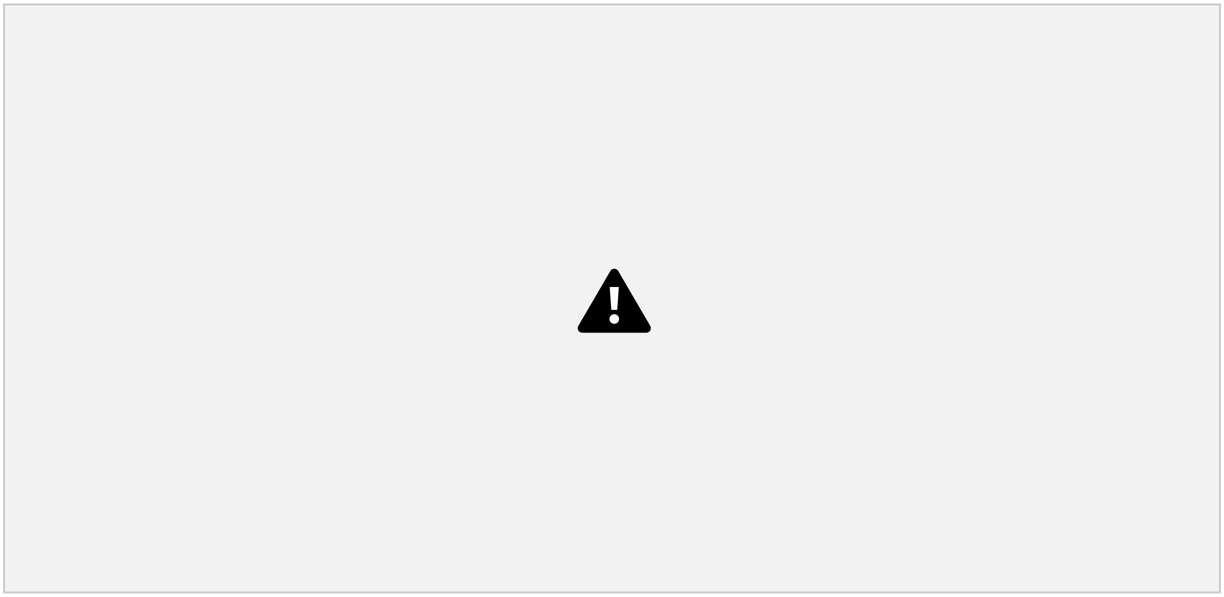
[Q.5]

(a) Write a short note on Spark.

Apache Spark is an open-source, distributed processing system used for big data workloads.

It utilizes in-memory caching and optimized query execution for fast queries against data of any size.

Simply put, Spark is a **fast and general engine for large-scale data processing**.



1. **Apache Spark Core** – Spark Core is the underlying general execution engine for the Spark platform that all other functionality is built upon. It provides in-memory computing and referencing datasets in external storage systems.
2. **Spark SQL** – Spark SQL is Apache Spark's module for working with structured data. The interfaces offered by Spark SQL provides Spark with more information about the structure of both the data and the computation being performed.
3. **Spark Streaming** – This component allows Spark to process real-time streaming data.

   ⇨ Data can be ingested from many sources like Kafka, Flume, and HDFS (Hadoop Distributed File System).

   ⇨ Then the data can be processed using complex algorithms and pushed out to file systems, databases, and live dashboards.

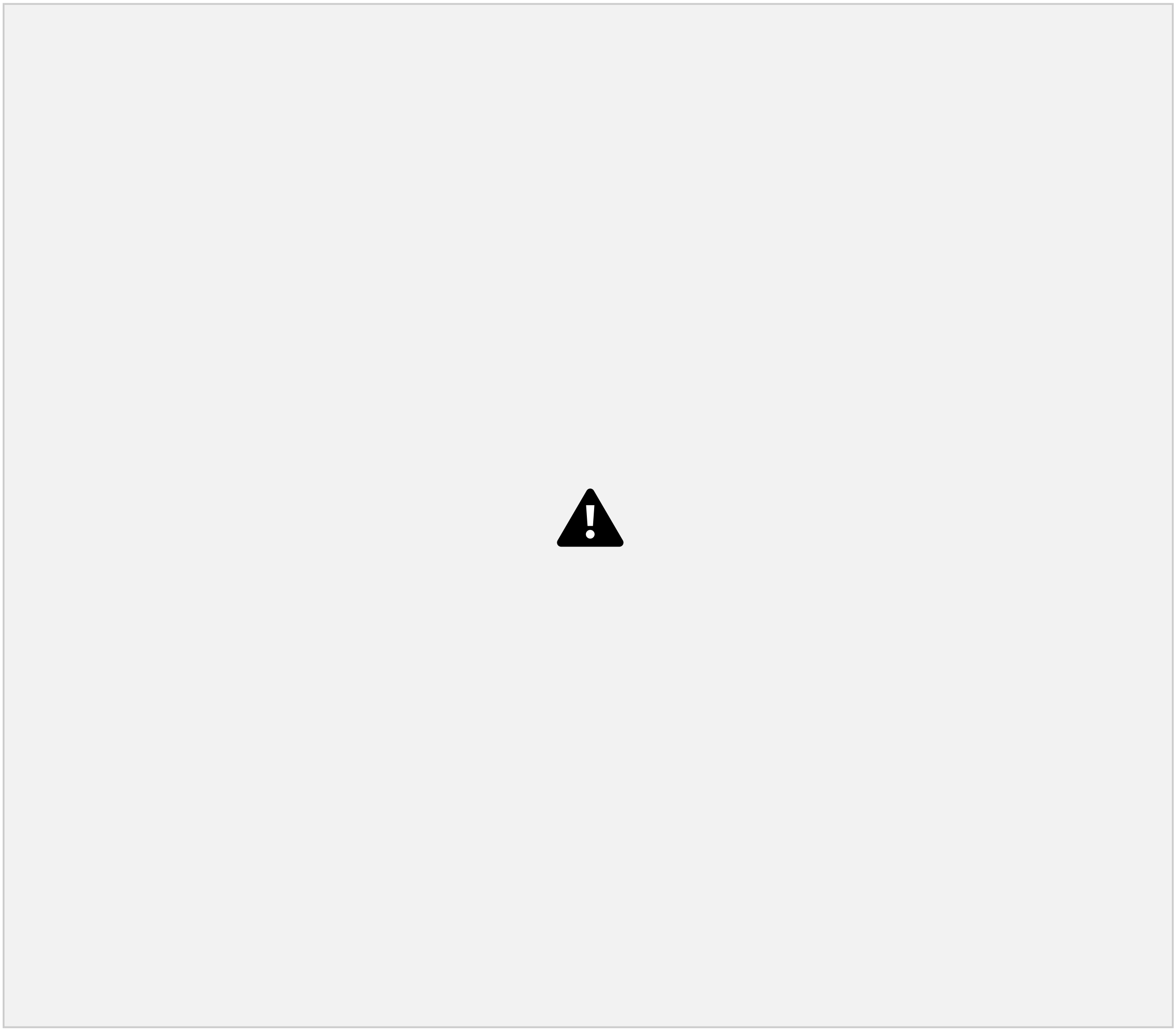
4. **MLlib (Machine Learning Library)** – Apache Spark is equipped with a rich library known as MLlib.

   ⇨ This library contains a wide array of machine learning algorithms classification, regression, clustering, and collaborative filtering.

   ⇨ It also includes other tools for constructing, evaluating, and tuning ML Pipelines. All these functionalities help Spark scale out across a cluster.

5. **GraphX** – Spark also comes with a library to manipulate graph databases and perform computations called GraphX. GraphX unifies ETL (Extract, Transform, and Load) process, exploratory analysis, and iterative graph computation within a single system.

(b) Write difference between MongoDB and Hadoop.



(c) Explain CRUD operations in MongoDB.

CRUD operations describe the conventions of a user-interface that let users view, search, and modify parts of the database.

MongoDB documents are modified by connecting to a server, querying the proper documents, and then changing the setting properties before sending the data back to the database to be updated. CRUD is data-oriented, and it's standardized according to HTTP action verbs.

When it comes to the individual CRUD operations:

• The Create operation is used to insert new documents in the MongoDB database. • The Read operation is used to query a document in the database.

• The Update operation is used to modify existing documents in the database. • The Delete operation is used to remove documents in the database.

Create Operations

⇨ For MongoDB CRUD, if the specified collection doesn't exist, the create operation will create the collection when it's executed.

⇨ Create operations in MongoDB target a single collection, not multiple collections.

⇨ Insert operations in MongoDB are atomic on a single document level.

⇨ MongoDB provides two different create operations that you can use to insert documents into a collection:

- db.collection.insertOne()
- db.collection.insertMany()

Read Operations

⇨ The read operations allow you to supply special query filters and criteria that let you specify which documents you want.

⇨ The MongoDB documentation contains more information on the available query filters.

⇨ Query modifiers may also be used to change how many results are returned.

⇨ MongoDB has two methods of reading documents from a collection:

- db.collection.find()
- db.collection.findOne()

Update Operations

⇨ Like create operations, update operations operate on a single collection, and they are atomic at a single document level.

⇨ An update operation takes filters and criteria to select the documents you want to update.

⇨ For MongoDB CRUD, there are three different methods of updating documents:

- db.collection.updateOne()
- db.collection.updateMany()
- db.collection.replaceOne()

Delete Operations

⇨ Delete operations operate on a single collection, like update and create operations.
⇨ Delete operations are also atomic for a single document. You can provide delete operations with filters and criteria in order to specify which documents you would like to delete from a collection.
⇨ The filter options rely on the same syntax that read operations utilize. ⇨ MongoDB has two different methods of deleting records from a collection:

- db.collection.deleteOne()
- db.collection.deleteMany()

OR

[Q.5]

(a) What are the features of MongoDB?

Q.4 (B)

(b) Explain Replication and scaling feature of MongoDB.

(c) What is RDD? State and Explain RDD operations.

The RDD (Resilient Distributed Dataset) is the Spark's core abstraction. It is a collection of elements, partitioned across the nodes of the cluster so that we can execute various parallel operations on it.

There are two ways to create RDDs:

- o Parallelizing an existing data in the driver program
- o Referencing a dataset in an external storage system, such as a shared filesystem, HDFS, HBase, or any data source offering a Hadoop InputFormat.

RDD Operations

⇨ The RDD provides the two types of operations:
- o Transformation
- o Action

<u>Transformation</u>

⇨ In Spark, the role of transformation is to create a new dataset from an existing one. The transformations are considered lazy as they only computed when an action requires a result to be returned to the driver program.

1. map(func) : It returns a new distributed dataset formed by passing each element of the source through a function func.

2. filter(func): It returns a new dataset formed by selecting those elements of the source on which func returns true.

3. flatMap(func): Here, each input item can be mapped to zero or more output items, so func should return a sequence rather than a single item.

4. intersection(otherDataset) : It returns a new RDD that contains the intersection of elements in the source dataset and the argument.

5. pipe(command, [envVars]): Pipe each partition of the RDD through a shell command, e.g. a Perl or bash script.

Action

⇨ In Spark, the role of action is to return a value to the driver program after running a computation on the dataset.

1. count() : It returns the number of elements in the dataset.

2. first() : It returns the first element of the dataset (similar to

take(1)). 3. take(n) : It returns an array with the first n elements of

the dataset.

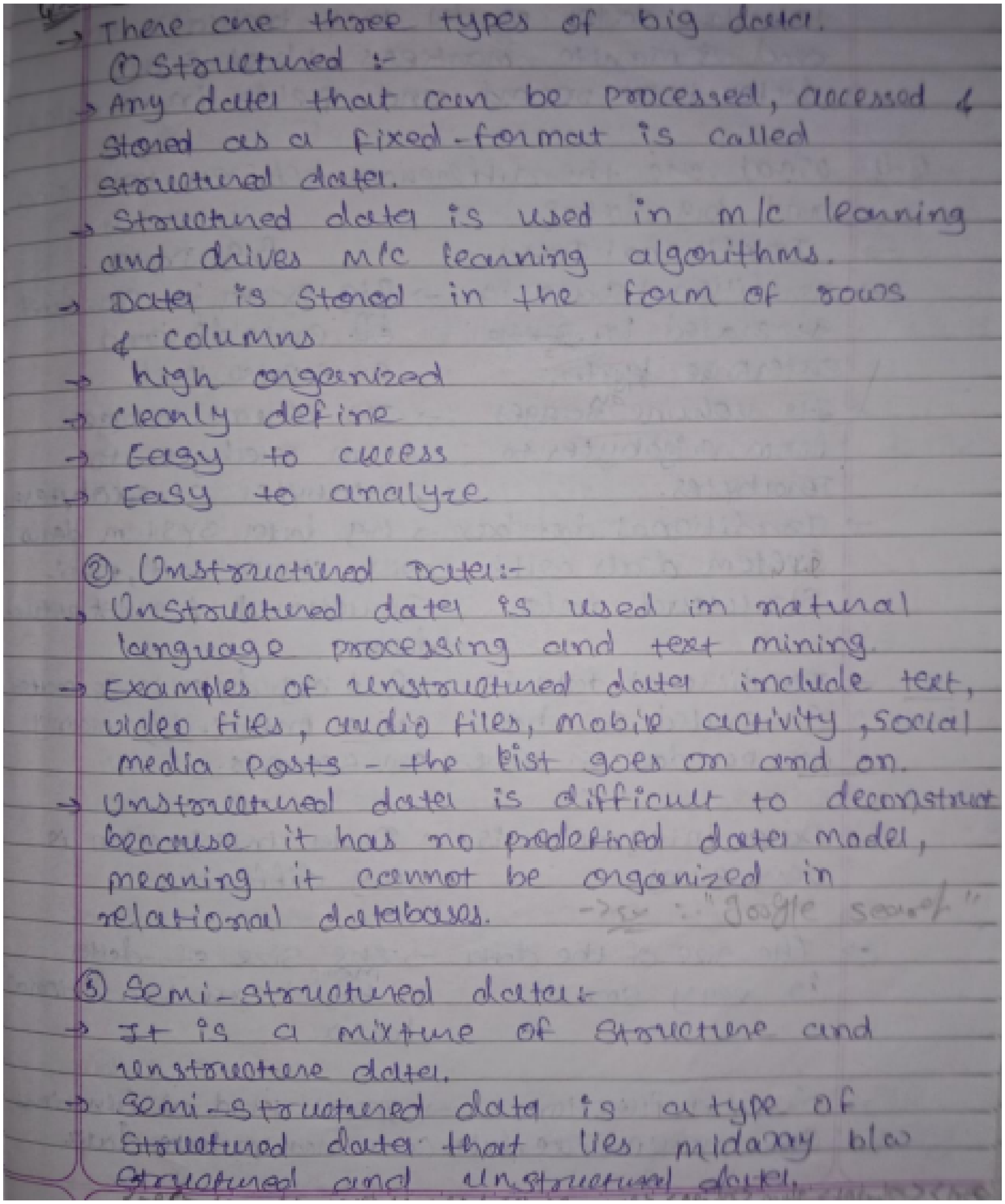
4. countByKey() : It is only available on RDDs of type (K, V). Thus, it returns a hashmap of (K, Int) pairs with the count of each key.

5. foreach(func) : It runs a function func on each element of the dataset for side effects such as updating an Accumulator or interacting with external storage systems.

****

Big Data Analytics (3170722) WINTER 2021 **[Q.1]**

(a) Explain types of Big Data



→ There are three types of big data.

① Structured :-

→ Any data that can be processed, accessed & stored as a fixed-format is called structured data.

→ Structured data is used in m/c learning and drives m/c learning algorithms.

→ Data is stored in the form of rows & columns

→ high organized

→ cleanly define

→ Easy to access

→ Easy to analyze

② Unstructured Data:-

→ Unstructured data is used in natural language processing and text mining.

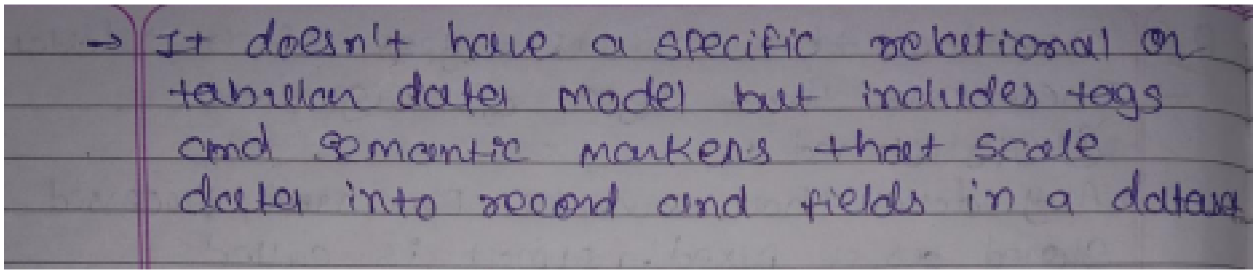
→ Examples of unstructured data include text, video files, audio files, mobile activity, social media posts - the list goes on and on.

→ Unstructured data is difficult to deconstruct because it has no predefined data model, meaning it cannot be organized in relational databases. ->ex :- "Google search"

③ Semi-structured data:-

→ It is a mixture of structure and unstructure data.

→ Semi-structured data is a type of structured data that lies midway b/w structured and unstructured data.


→ It doesn't have a specific relational or tabular data model but includes tags and semantic markers that scale data into record and fields in a dataset

(b) Describe Traditional vs. Big Data business approach.

Traditional information, it's hard to keep up the precision and private as the nature of the information is high and so as to store such a huge amount of information is costly.

It influences the information breaking down which likewise declines the final product of exactness and secrecy.

However, big information makes this work a lot less complex and bother free when contrasted with customary information.

Likewise, it gives high precision and makes the outcomes increasingly exact.

Data Relationship

Enormous information contains a huge amount of information which makes the database relationship difficult to comprehend.

It influences the information thing which likewise makes the understanding level annoying.

Anything but difficult to experience all information and data without confronting an excessive amount of difficulty.

It likewise helps in making sense of the connection between information and information things without any problem.

Data storage size

The size of capacity in information is significant.

In Traditional Data, it’s difficult to store a lot of information.

The main certain sum can be put away; in any case, Big Data can store immense voluminous information without any problem.

The conventional database can spare information in the number of gigabytes to terabytes.

All things considered, the large information can spare several terabytes, petabytes, and significantly more.

It likewise helps in setting aside the measure of cash that spends on the customary database for capacity.

By putting away enormous information decreases additional sources and cash.

(c) What is Big Data? Explain Challenges of Conventional System.

Big Data is a collection of data that is huge in volume, yet growing exponentially with time.

→ 1.) Lack of proper understanding of big data.
→ Companies fail in their Big Data initiatives due to insufficient understanding.
→ Employees may not know what data is, its storage, processing, importance and sources.
→ Data professionals may know what is going on, but others may not have a clear picture.

2) Data growth issues
→ One of the most pressing challenges of Big Data is storing all these huge sets of data properly.
→ The amount of data being stored in data ~~cent~~ centers and databases of companies is increasing rapidly.
→ As these data sets grow exponentially with time, it gets extremly difficult to handle.

3) Confusion while Big Data tools selection.
→ Companies often get confused while selecting the best tool for Big Data analysis and storage
→ Is HBase or Cassandra the best technology for data storage?
→ Is Hadoop MapReduce good enough or will Spark be a better option for data analytics and storage?

4) Lack of data professionals

→ To run these modern technologies and Big Data tools, companies need skilled data professionals.

→ These professionals will include data scientists, data analysts and data engineers who are experienced in working with the tools and making sense out of huge data sets.

5) Securing data

→ Securing these huge sets of data is one of the daunting challenges of Big data.

→ Often companies are so busy in understanding storing and analyzing their data sets that they push data security for later stages.

6) Integrating data from a variety of sources

→ Data in an organization comes from a variety of sources, such as social media pages, ERP applications, customer logs, financial reports, e-mails, presentations and reports created by employees.

→ Combining all this data to prepare reports is a challenging task.

[Q.2]

(a) Explain basic Components of Analyzing the Data with Hadoop.

Hadoop is a framework that uses distributed storage and parallel processing to store and manage Big Data. It is the most commonly used software to handle Big Data. There are three components of Hadoop.

1. Hadoop HDFS - Hadoop Distributed File System (HDFS) is the storage unit of Hadoop.
2. Hadoop MapReduce - Hadoop MapReduce is the processing unit of Hadoop.
3. Hadoop YARN - Hadoop YARN is a resource management unit of Hadoop.

Hadoop HDFS

⇨ Data is stored in a distributed manner in HDFS. There are two components of HDFS - name node and data node. While there is only one name node, there can be multiple data nodes.

Hadoop MapReduce

⇨ Hadoop MapReduce is the processing unit of Hadoop. In the MapReduce approach, the processing is done at the slave nodes, and the final result is sent to the master node.

Hadoop YARN

⇨ Hadoop YARN stands for Yet Another Resource Negotiator. It is the resource management unit of Hadoop and is available as a component of Hadoop version 2.

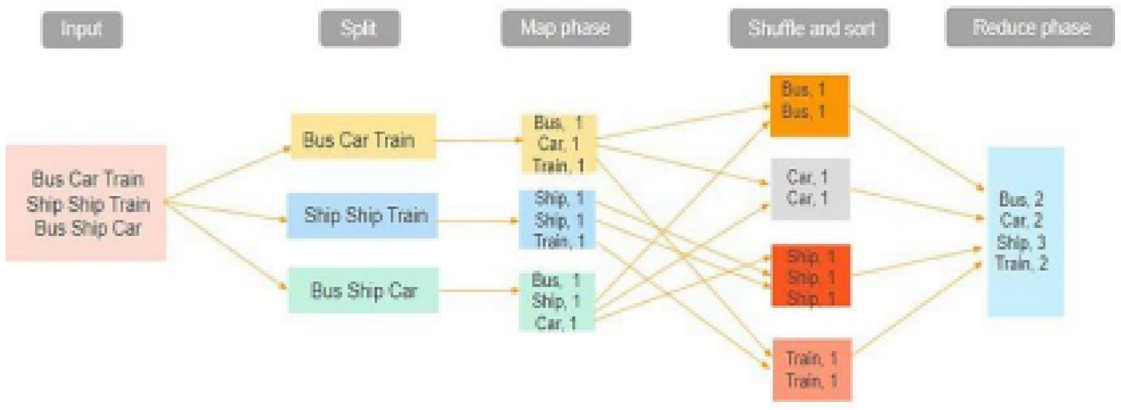
(b) What is Map Reduce and explain How Map Reduce Works?

A MapReduce is a data processing tool which is used to process the data parallelly in a distributed form.

There are two functions: Map and Reduce.

They are sequenced one after the other.

· The Map function takes input from the disk as <key,value> pairs, processes them, and produces another set of intermediate <key,value> pairs as output. · The Reduce function also takes inputs as <key,value> pairs, and produces <key,value> pairs as output.



The types of keys and values differ based on the use case. All inputs and outputs are stored in the HDFS.

While the map is a mandatory step to filter and sort the initial data, the reduce function is optional.

<k1, v1> -> Map() -> list(<k2, v2>)
<k2, list(v2)> -> Reduce() -> list(<k3, v3>)

Mappers and Reducers are the Hadoop servers that run the Map and Reduce functions respectively. It doesn't matter if these are the same or different servers.

Map

The input data is first split into smaller blocks. Each block is then assigned to a mapper for processing.

For example, if a file has 100 records to be processed, 100 mappers can run together to process one record each.

Or maybe 50 mappers can run together to process two records each. The Hadoop framework decides how many mappers to use, based on the size of the data to be processed and the memory block available on each mapper server.

'

Reduce

After all the mappers complete processing, the framework shuffles and sorts the results before passing them on to the reducers.

A reducer cannot start while a mapper is still in progress. All the map output values that have the same key are assigned to a single reducer, which then aggregates the values for that key.

Combine and Partition

There are two intermediate steps between Map and Reduce.

Combine is an optional process.

The combiner is a reducer that runs individually on each mapper server.

It reduces the data on each mapper further to a simplified form before passing it downstream.

This makes shuffling and sorting easier as there is less data to work with.

Partition is the process that translates the <key, value> pairs resulting from mappers to another set of <key, value> pairs to feed into the reducer.

It decides how the data has to be presented to the reducer and also assigns it to a particular reducer.

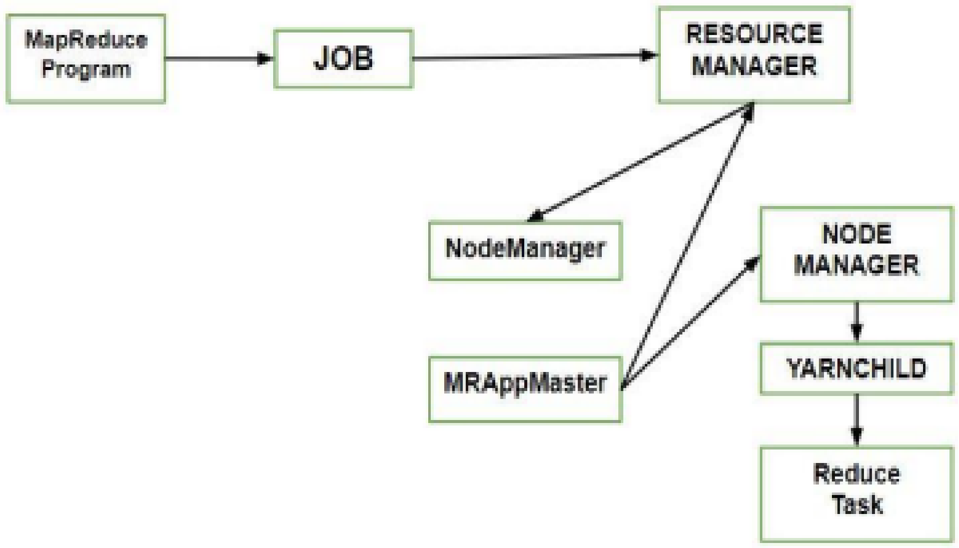
The default partitioner determines the hash value for the key, resulting from the mapper, and assigns a partition based on this hash value.

There are as many partitions as there are reducers. So, once the partitioning is complete, the data from each partition is sent to a specific reducer.

(c) Brief Anatomy of a Map Reduce Job run and Failures.

MapReduce can be used to work with a solitary method call: **submit()** on a Job object (you can likewise call **waitForCompletion()**, which presents the activity on the off chance that it hasn't been submitted effectively, at that point sits tight for it to finish).
1. **Client:** Submitting the MapReduce job.

2. **Yarn node manager:** In a cluster, it monitors and launches the compute containers on machines.
3. **Yarn resource manager:** Handles the allocation of computing resources coordination on the cluster.
4. **MapReduce application master** Facilitates the tasks running the MapReduce work.
5. **Distributed Filesystem:** Shares job files with other entities.



To create an internal JobSubmitter instance, use the **submit()** which further calls **submitJobInternal()** on it. Having submitted the job,

waitForCompletion() polls the job's progress after submitting the job once per second.

If the reports have changed since the last report, it further reports the progress to the console.

The job counters are displayed when the job completes successfully. Else the error (that caused the job to fail) is logged to the console.

OR

(c) Describe Map Reduce Types and Formats.

A MapReduce is a data processing tool which is used to process the data parallelly in a distributed form.

MapReduce architecture contains two core components as Daemon services responsible for running mapper and reducer tasks, monitoring, and re-executing the tasks on failure.

In Hadoop 2 onwards Resource Manager and Node Manager are the daemon services. When the job client submits a MapReduce job, these daemons come into action.

They are also responsible for parallel processing and insensitive features of MapReduce jobs.

In Hadoop 2 onwards resource management and job scheduling or monitoring functionalities are segregated by YARN (Yet Another Resource Negotiator) as different daemons.

TYPES OF INPUT FORMAT:

1. FileInputFormat: It is the base class for all file-based InputFormats.

When we start a MapReduce job execution, FileInputFormat provides a path containing files to read.

This InputFormat will read all files and divides these files into one or more InputSplits.

2. TextInputFormat: It is the default InputFormat.

This InputFormat treats each line of each input file as a separate record. It performs no parsing.

TextInputFormat is useful for unformatted data.

3. KeyValueTextInputFormat: It is similar to TextInputFormat.

This InputFormat also treats each line of input as a separate record.

While the difference is that TextInputFormat treats entire line as the value, but the KeyValueTextInputFormat breaks the line itself into key and value .

4. SequenceFileInputFormat: It is an InputFormat which reads sequence files.

Sequence files are binary files. These files also store sequences of binary key-value pairs.

These are block-compressed and provide direct serialization and deserialization of several arbitrary data.

5. N-lineInputFormat: It is another form of TextInputFormat where the keys are byte offset of the line.

And values are contents of the line. So, each mapper receives a variable number of lines of input with TextInputFormat and KeyValueTextInputFormat.

So, if want our mapper to receive a fixed number of lines of input, then we use NLineInputFormat.

6. DBInputFormat: This InputFormat reads data from a relational database, using JDBC.

It also loads small datasets, perhaps for joining with large datasets from HDFS using MultipleInputs.

TYPES OF OUTPUT FORMAT:

1. **TextOutputFormat**: The default OutputFormat is TextOutputFormat. It writes (key, value) pairs on individual lines of text files. Its keys and values can be of any type.

2. **SequenceFileOutputFormat**: This OutputFormat writes sequences files for its output.

SequenceFileInputFormat is also intermediate format use between MapReduce jobs. It serializes arbitrary data types to the file.

3. **SequenceFileAsBinaryOutputFormat**: It is another variant of SequenceFileInputFormat.

It also writes keys and values to sequence file in binary format.

[Q.3]

(a) Explain NoSQL data architecture.

Types of NoSQL Database:

Document-based databases
Key-value stores
Column-oriented databases

Graph-based databases

(Refer Q-3-c SUMMER 2022)

(b) Elaborate Key-value stores, Graph stores, Column family stores & Document stores.

(Refer Q-3-c SUMMER 2022)

(c) Describe analyzing big data with a shared-nothing architecture.

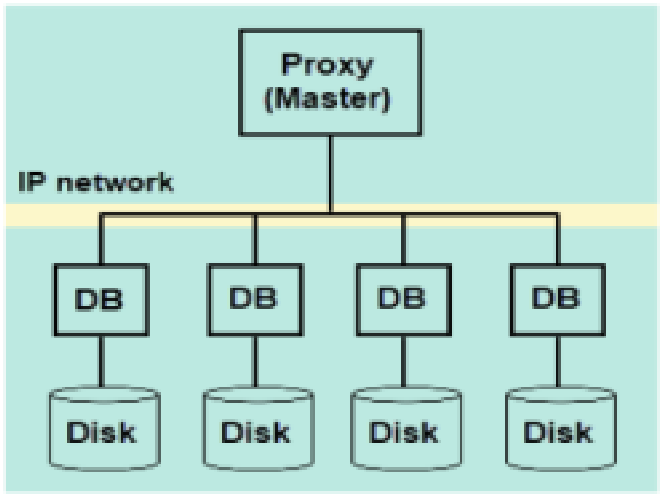
Shared Nothing Architecture (SNA) is a distributed computing architecture that consists of multiple separated nodes that don't share resources.

The nodes are independent and self-sufficient as they have their own disk space and memory.

In such a system, the data set/workload is split into smaller sets (nodes) distributed into different parts of the system.

Each node has its own memory, storage, and independent input/output interfaces. It communicates and synchronizes with other nodes through a high-speed interconnect network.

Such a connection ensures low latency, high bandwidth, as well as high availability (with a backup interconnect available in case the primary fails).



Advantages

There are many advantages of SNA, the main ones being scalability, fault tolerance, and less downtime.

Easier to Scale

⇨ There is no limit when it comes to scaling in the shared-nothing model. Unlimited scalability is one of the best features of this type of architecture.

Eliminates Single Points of Failure

⇨ If one of the nodes in the application fails, it doesn't affect the functionality of others as each node is self-reliant.

Simplifies Upgrades and Prevents Downtime

⇨ There is no need to shut down the system while working on or upgrading individual nodes.

Disadvantages

Cost

⇨ A node consists of its individual processor, memory, and disk.

Decreased Performance

⇨ Scaling up your system can eventually affect the overall performance if the cross communication layer isn't set up correctly.

OR

[Q.3]

(a) Difference between master-slave versus peer-to-peer distribution models.

x

(b) What is a big data NoSQL? Explain in details.

NoSQL is an open -source database. NoSQL databases do not have a schema. They are flexible.

NoSQL databases are divided into categories based on the data model.

The different data models are Key-Value store, Document, Column — input, Graph data models.

These databases can be distributed on different devices and locations. Scalability is also one of the main characteristics.

These databases do not always follow ACID properties.

NoSQL databases follow CAP theorem.

The CAP theorem states that a distributed computer system cannot guarantee Consistency, Availability and Partition tolerance at the same time.

NoSQL databases are BASE(Basically Available Soft state Eventually consistent) systems.

These databases guarantee availability. These databases become consistent over time when they do not receive input.

Examples of NoSQL Databases:

DynamoDB2: Dynamo DB is the NoSQL database of Amazon Web Services.

⇨ It supports both Key-Value store and Document data models.

⇨ It stores data in Tables, Items, attributes, Maps, lists.

Mongodb3: MongoDB is an open source database.

⇨ It uses document model, it stores data in JSON-like documents.

⇨ It is a distributed database. MongoDB Compass is the graphical user interface for

MongoDB. It provides ACID properties at the document level.

⇨ It maintains the copies of the data using replication. It stores the data in Collections, Documents, fields, embedded documents, Arrays.

(c) Which Four ways that NoSQL systems handle big data problems.

Different ways to handle Big Data problems:

1. The queries should be moved to the data rather than moving data to queries:

At the point, when an overall query is needed to be sent by a customer to all hubs/nodes holding information, the more proficient way is to send a query to every hub than moving a huge set of data to a central processor.

The stated statement is a basic rule that assists to see how NoSQL data sets have sensational execution benefits on frameworks that were not developed for queries distribution to hubs.

The entire data is kept inside hub/node in document form which means just the query and result are needed to move over the network, thus keeping big data's queries quick.

2. Hash rings should be used for even distribution of data:

To figure out a reliable approach to allocating a report to a processing hub/node is perhaps the most difficult issue with databases that are distributed.

With a help of an arbitrarily produced 40-character key, the hash rings method helps in even distribution of a large amount of data on numerous servers and this is a decent approach to uniform distribution of network load.

3. For scaling read requests, replication should be used:

In real-time, replication is used by databases for making data's backup copies. Read requests can be scaled horizontally with the help of replication.

The strategy of replication functions admirably much of the time.

4. Distribution of queries to nodes should be done by the database: Separation of concerns of evaluation of query from the execution of the query is important for getting more increased performance from queries traversing numerous hubs/nodes.

The query is moved to the data by the NoSQL database instead of data moving to the query.

[Q.4]

(a) How Graph Analytics used in Big Data.

Graph Analytics refers to the analysis performed on the data stored in knowledge graph data.

It’s just like Data Management and Data Analysis. You organize the data in a Graph Database before performing the Graph Analytics.

In Graph Analytics, the queries are executed via the edges connecting the entities. The query execution on a graph database is comparatively faster than a relational database.

You can differentiate entity types like a person, city, etc, by adding colors, weightage, format data, and label them in the way you want for visualizing it.

Types of Graph Analytics

Based on your goal, graph analytics could be used in different ways. Let’s see them briefly below.

Node strength analysis

The significance of a specific node in a network of nodes is determined by node strength analysis. The higher the strength the more important the node to the network.

Edge strength analysis

As the term indicates, the edge significance analysis is all about the weightage of an edge in connecting two nodes. This analysis helps to determine the strength (strong or weak) of an edge between two nodes.

Clustering

Clustering enables grouping objects based on the characteristics they exhibit. Clustering is extremely useful when you want to categorize your graph data in a customized way.

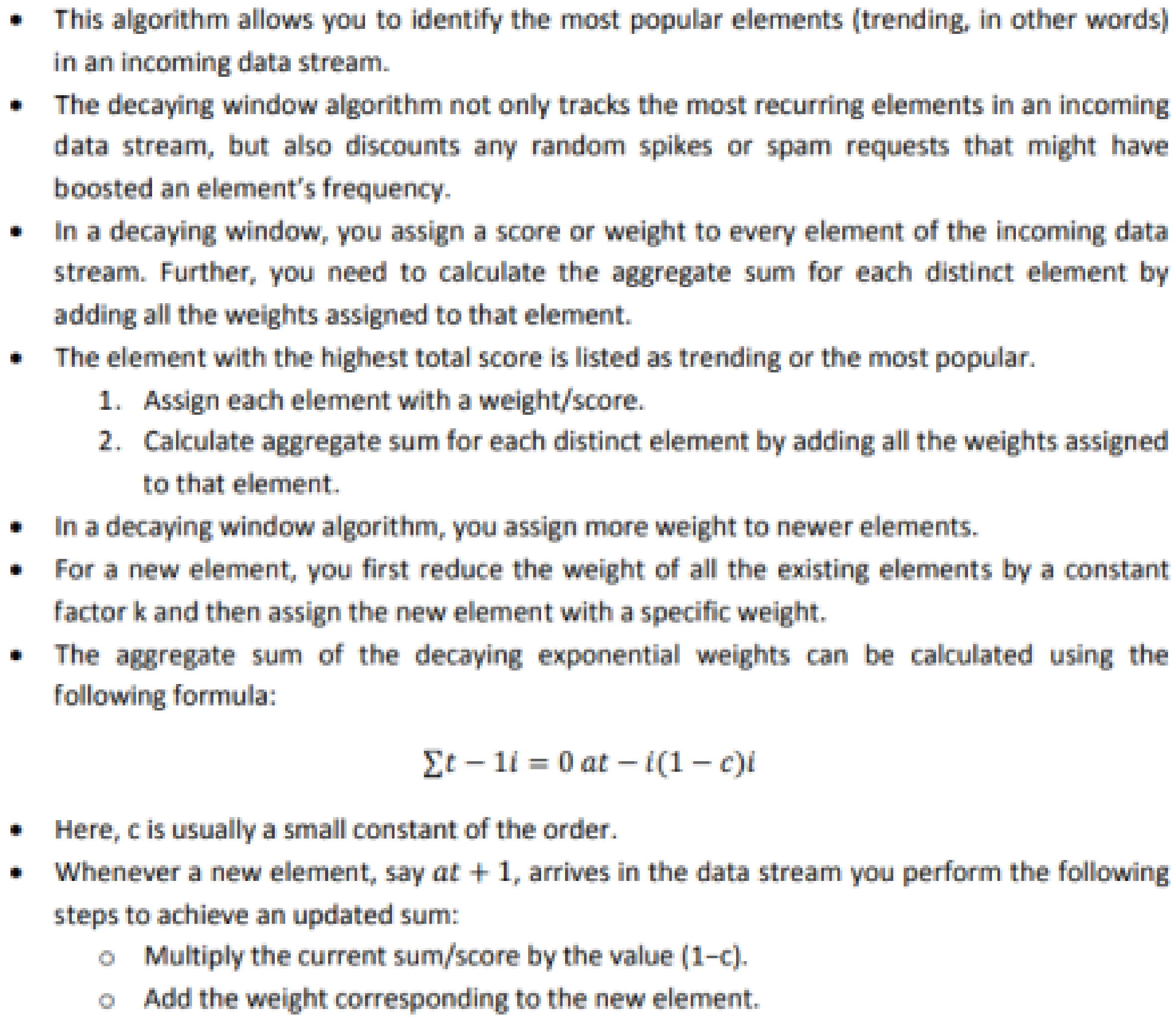
Path analysis

Path analysis involves finding out the shortest and widest path between two nodes. This kind of analysis is used in social network analysis, supply chain optimization.

Predictive graph analysis

Predictive analysis, in a graph database, is the analysis performed on past graph data, to determine the edges or nodes in the future.

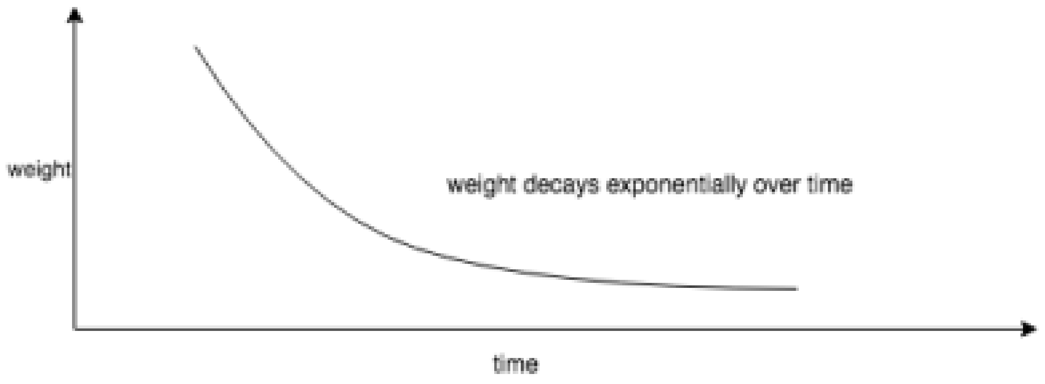
(b) Write a short note on Decaying Window.



- This algorithm allows you to identify the most popular elements (trending, in other words) in an incoming data stream.
- The decaying window algorithm not only tracks the most recurring elements in an incoming data stream, but also discounts any random spikes or spam requests that might have boosted an element's frequency.
- In a decaying window, you assign a score or weight to every element of the incoming data stream. Further, you need to calculate the aggregate sum for each distinct element by adding all the weights assigned to that element.
- The element with the highest total score is listed as trending or the most popular.
    1. Assign each element with a weight/score.
    2. Calculate aggregate sum for each distinct element by adding all the weights assigned to that element.
- In a decaying window algorithm, you assign more weight to newer elements.
- For a new element, you first reduce the weight of all the existing elements by a constant factor k and then assign the new element with a specific weight.
- The aggregate sum of the decaying exponential weights can be calculated using the following formula:

$$\sum t - 1i = 0\; at - i(1 - c)i$$

- Here, c is usually a small constant of the order.
- Whenever a new element, say $at + 1$, arrives in the data stream you perform the following steps to achieve an updated sum:
    - Multiply the current sum/score by the value (1−c).
    - Add the weight corresponding to the new element.



In a data stream consisting of various elements, you maintain a separate sum for each distinct element.

For every incoming element, you multiply the sum of all the existing elements by a value of (1−c). Further, you add the weight of the incoming element to its corresponding aggregate sum.

(c) Explain with example: How to perform Real Time Sentiment Analysis of any product.

Sentiment analysis is an automated process capable of understanding the feelings or opinions that underlie a text.

It is one of the most interesting subfields of NLP, a branch of Artificial Intelligence (AI) that focuses on how machines process human language.

Sentiment analysis studies the subjective information in an expression, that is, the opinions, appraisals, emotions, or attitudes towards a topic, person or entity.

Expressions can be classified as **positive**, **negative**, or **neutral**.

For example:

- “I really like the new design of your website!” → Positive
- “I’m not sure if I like the new design” → Neutral
- “The new design is awful!” → Negative

Real Time Sentiment Analysis of any product.

⇨ Sentiment analysis is a powerful tool, which can be used across industries and teams. Learn about some of the most popular sentiment analysis business applications, below:

- Social media monitoring
- Brand monitoring
- Customer support analysis
- Customer feedback analysis
- Market research

⇨ **Social Media Monitoring :**

⇨ There are more than 3.5 billion active social media users; that’s 45% of the world’s population.

⇨ Every minute users send over 500,000 Tweets and post 510,000 Facebook comments, and a large amount of these messages contain valuable business insights about how customers feel towards products, brands and services.

⇨ **Brand Monitoring :**

⇨ Besides social media, online conversations can take place in blogs, review websites, news websites and forum discussions.

⇨ Product reviews, for instance, have become a crucial step in the buyer’s journey.

⇨ Consumers read at least 10 reviews before buying, and 57% only trust a business if it has a star-rating of 4 or more.

⇨ **Customer Support Analysis :**

⇨ Providing outstanding customer service experiences should be a priority. After all, 96% of consumers say great customer service is a key factor to choose and stay loyal to a brand.

⇨ **Customer Feedback Analysis :**

⇨ Net Promoter Score (NPS) surveys are one of the most popular ways to ask for customers feedback about a product or service.

⇨ Sentiment analysis of NPS surveys allows you to go beyond the numerical scores and groups (Detractors, Promoters, Passives), as well as speed up the process and obtain more consistent results than if you were tagging these results manually.

⇨ **Market Research :**

⇨ Want to collect insights on customer feelings, experiences, and needs relating to a marketing campaign for a new product release Sentiment analysis can help monitor online conversations about a specific marketing campaign, so you can see how it’s performing.

[Q.5]

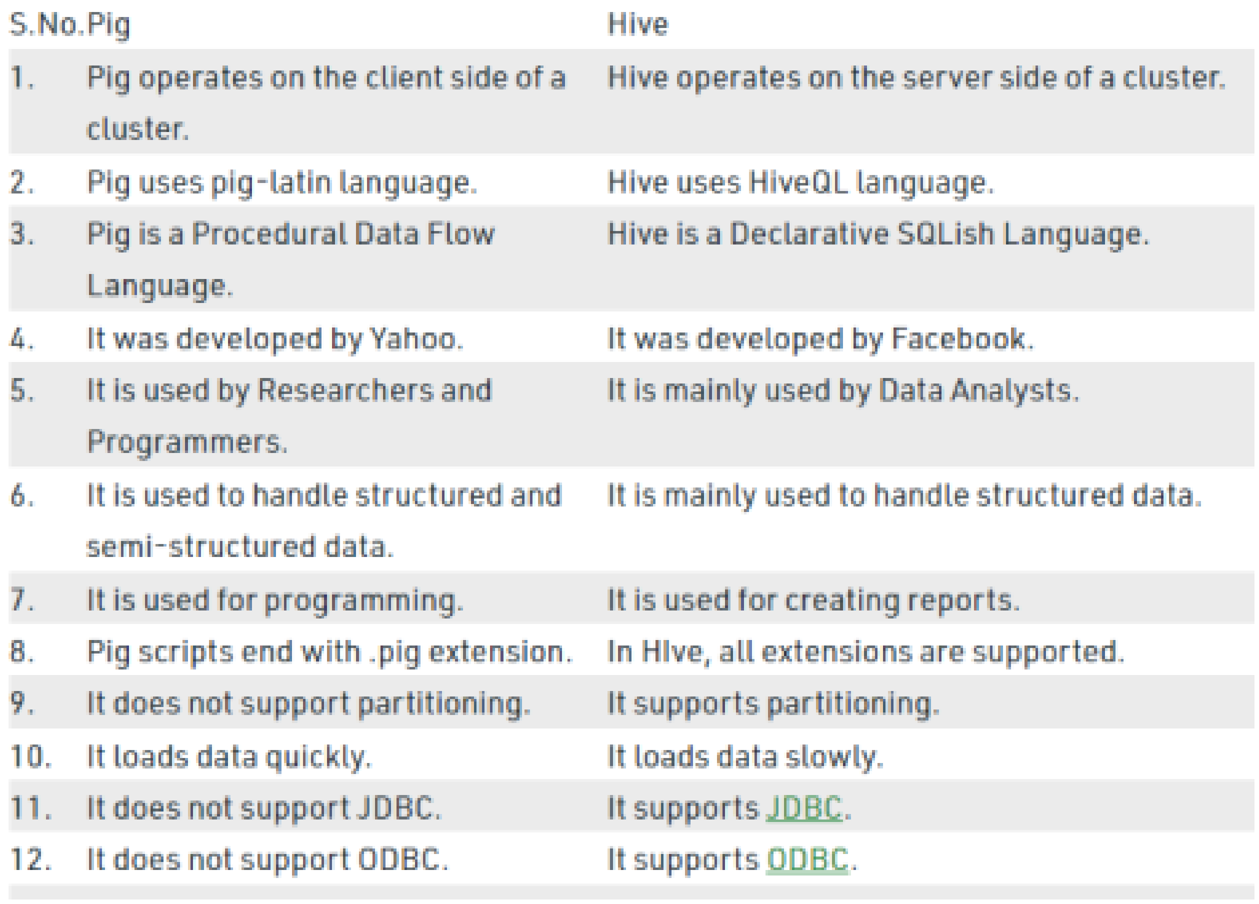
(a) What is the use of Pig and Hive in Big Data?

1. Pig :

Pig is used for the analysis of a large amount of data. It is abstract over MapReduce.
Pig is used to perform all kinds of data manipulation operations in Hadoop. It provides the Pig-Latin language to write the code that contains many inbuilt functions like join, filter, etc.
The two parts of the Apache Pig are Pig-Latin and Pig-Engine. Pig Engine is used to convert all these scripts into a specific map and reduce tasks.
Pig abstraction is at a higher level. It contains less line of code as compared to MapReduce.

2. Hive :
Hive is built on the top of Hadoop and is used to process structured data in Hadoop. Hive was developed by Facebook.
It provides various types of querying language which is frequently known as Hive Query Language.
Apache Hive is a data warehouse and which provides an SQL-like interface between the user and the Hadoop distributed file system (HDFS) which integrates Hadoop.

⇨ **<u>Only for Refer : pig vs hive</u>**



S.No.	Pig	Hive
1.	Pig operates on the client side of a cluster.	Hive operates on the server side of a cluster.
2.	Pig uses pig-latin language.	Hive uses HiveQL language.
3.	Pig is a Procedural Data Flow Language.	Hive is a Declarative SQLish Language.
4.	It was developed by Yahoo.	It was developed by Facebook.
5.	It is used by Researchers and Programmers.	It is mainly used by Data Analysts.
6.	It is used to handle structured and semi-structured data.	It is mainly used to handle structured data.
7.	It is used for programming.	It is used for creating reports.
8.	Pig scripts end with .pig extension.	In HIve, all extensions are supported.
9.	It does not support partitioning.	It supports partitioning.
10.	It loads data quickly.	It loads data slowly.
11.	It does not support JDBC.	It supports JDBC.
12.	It does not support ODBC.	It supports ODBC.

(b) Describe data processing operators in Pig.

The Apache Pig Operators is a high-level procedural language for querying large data sets using Hadoop and the Map-Reduce Platform.

A Pig Latin statement is an operator that takes a relation as input and produces another relation as output.

These operators are the main tools for Pig Latin provides to operate on the data.
They allow you to transform it by sorting, grouping, joining, projecting, and filtering.
The Apache Pig operators can be classified as :

Relational Operators :

- ⇨ **LOAD**: The LOAD operator is used to loading data from the file system or HDFS storage into a Pig relation.
- ⇨ **FOREACH:** This operator generates data transformations based on columns of data. It is used to add or remove fields from a relation.
- ⇨ **FILTER:** This operator selects tuples from a relation based on a condition.
- ⇨ **JOIN:** JOIN operator is used to performing an inner, equijoin join of two or more relations based on common field values

⇨ **ORDER BY:** Order By is used to sort a relation based on one or more fields in either ascending or descending order using ASC and DESC keywords. ⇨ **GROUP:** The GROUP operator groups together the tuples with the same group key (key field).

⇨ **COGROUP:** COGROUP is the same as the GROUP operator.

Diagnostic Operator :

⇨ **<u>DUMP</u>:** The DUMP operator is used to run Pig Latin statements and display the results on the screen.

⇨ **<u>DESCRIBE</u>:** Use the DESCRIBE operator to review the schema of a particular relation. **<u>ILLUSTRATE</u>: I**LLUSTRATE operator is used to review how data is transformed through a sequence of Pig Latin statements.

⇨ **<u>EXPLAIN</u>:** The EXPLAIN operator is used to display the logical, physical, and MapReduce execution plans of a relation.

(c) Describe HBase and ZooKeeper in details.

Describe HBase :

Hbase is an open source and sorted map data built on Hadoop. It is column oriented and horizontally scalable.

It is based on Google's Big Table.It has set of tables which keep data in key value format.

Hbase is well suited for sparse data sets which are very common in big data use cases. Hbase provides APIs enabling development in practically any programming language.

It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in the Hadoop File System.

⇨ Features of Hbase :

- Horizontally scalable: You can add any number of columns anytime.
- Automatic Failover: Automatic failover is a resource that allows a system administrator to automatically switch data handling to a standby system in the event of system compromise
- Integrations with Map/Reduce framework: Al the commands and java codes

internally implement Map/ Reduce to do the task and it is built over Hadoop Distributed File System.

- sparse, distributed, persistent, multidimensional sorted map, which is indexed by rowkey, column key,and timestamp.

Describe ZooKeeper :

Apache ZooKeeper is a service used by a cluster (group of nodes) to coordinate between themselves and maintain shared data with robust synchronization techniques.

ZooKeeper is itself a distributed application providing services for writing a distributed application.

The common services provided by ZooKeeper are as follows −

- **Naming service** − Identifying the nodes in a cluster by name. It is similar to DNS, but for nodes.
- **Configuration management** − Latest and up-to-date configuration information of the system for a joining node.
- **Cluster management** − Joining / leaving of a node in a cluster and node status at real time.
- **Leader election** − Electing a node as leader for coordination purpose. · **Locking and synchronization service** − Locking the data while modifying it. This mechanism helps you in automatic fail recovery while connecting other distributed applications like Apache HBase.
- **Highly reliable data registry** − Availability of data even when one or a few nodes are down.

OR

[Q.5]

(a) Explain HIVE services.

Hive is an ETL and Data warehousing tool developed on top of Hadoop Distributed File System (HDFS).

Hive makes job easy for performing operations like

- Data encapsulation
- Ad-hoc queries
- Analysis of huge datasets

In Hive, tables and databases are created first and then data is loaded into these tables.

Hive as data warehouse designed for managing and querying only structured data that is stored in tables.

While dealing with structured data, Map Reduce doesn’t have optimization and usability features like UDFs but Hive framework does.

Query optimization refers to an effective way of query execution in terms of performance.

Hive’s SQL-inspired language separates the user from the complexity of Map Reduce programming.

It reuses familiar concepts from the relational database world, such as tables, rows, columns and schema, etc. for ease of learning.

Hadoop’s programming works on flat files. So, Hive can use directory structures to “partition” data to improve performance on certain queries.

A new and important component of Hive i.e. Metastore used for storing schema information. This Metastore typically resides in a relational database. We can interact with Hive using methods like.

(b) Write application of writing Spark.

Every SBT project needs a *build.sbt* file.
There are many things that a complicated project build might require you to configure in the *build.sbt* file.

However, there are three main things that you must Include in your build file.

1. Project Metadata
2. Dependencies
3. Repository Resolvers

(c) Describe any application that you know related to enhance particular business using big data and explain how it is important as a business prospective.

x

****

Seat No.: ________ Enrolment No.___________

GUJARAT TECHNOLOGICAL UNIVERSITY

BE - SEMESTER–VII (NEW) EXAMINATION – WINTER 2021

Subject Code:3170722 **Date:23/12/2021**

Subject Name:Big Data Analytics

Time:10:30 AM TO 01:00 PM **Total Marks: 70**

Instructions:

1. **Attempt all questions.**
2. **Make suitable assumptions wherever necessary.**
3. **Figures to the right indicate full marks.**
4. **Simple and non-programmable scientific calculators are allowed.**

			MARKS
Q.1	**(a)**	Explain types of Big Data	**03**
	(b)	Describe Traditional vs. Big Data business approach.	**04**
	(c)	What is Big Data? Explain Challenges of Conventional System.	**07**
Q.2	**(a)**	Explain basic Components of Analyzing the Data with Hadoop.	**03**
	(b)	What is Map Reduce and explain How Map Reduce Works?	**04**
	(c)	Brief Anatomy of a Map Reduce Job run and Failures.	**07**
		OR	
	(c)	Describe Map Reduce Types and Formats.	**07**
Q.3	**(a)**	Explain NoSQL data architecture.	**03**
	(b)	Elaborate Key-value stores, Graph stores, Column family stores & Document stores.	**04**
	(c)	Describe analyzing big data with a shared-nothing architecture.	**07**
		OR	
Q.3	**(a)**	Difference between master-slave versus peer-to-peer distribution models.	**03**
	(b)	What is a big data NoSQL? Explain in details.	**04**
	(c)	Which Four ways that NoSQL systems handle big data problems.	**07**
Q.4	**(a)**	What is Stream Computing and Sampling Data in a Stream.	**03**
	(b)	Write the application of RTAP.	**04**
	(c)	Explain Stream Data Model and Architecture.	**07**
		OR	
Q.4	**(a)**	How Graph Analytics used in Big Data.	**03**
	(b)	Write a short note on Decaying Window.	**04**
	(c)	Explain with example: How to perform Real Time Sentiment Analysis of any product.	**07**
Q.5	**(a)**	What is the use of Pig and Hive in Big Data?	**03**
	(b)	Describe data processing operators in Pig.	**04**
	(c)	Describe HBase and ZooKeeper in details.	**07**
		OR	
Q.5	**(a)**	Explain HIVE services.	**03**
	(b)	Write application of writing Spark.	**04**
	(c)	Describe any application that you know related to enhance particular business using big data and explain how it is important as a business prospective.	**07**

****