A.Q1) Hadoop divides the job into two types of tasks:

1) Map tasks (splits & Mapping)
2) Reduce tasks (shuffling, Reducing)

The complete execution process is controlled by two types of entities:

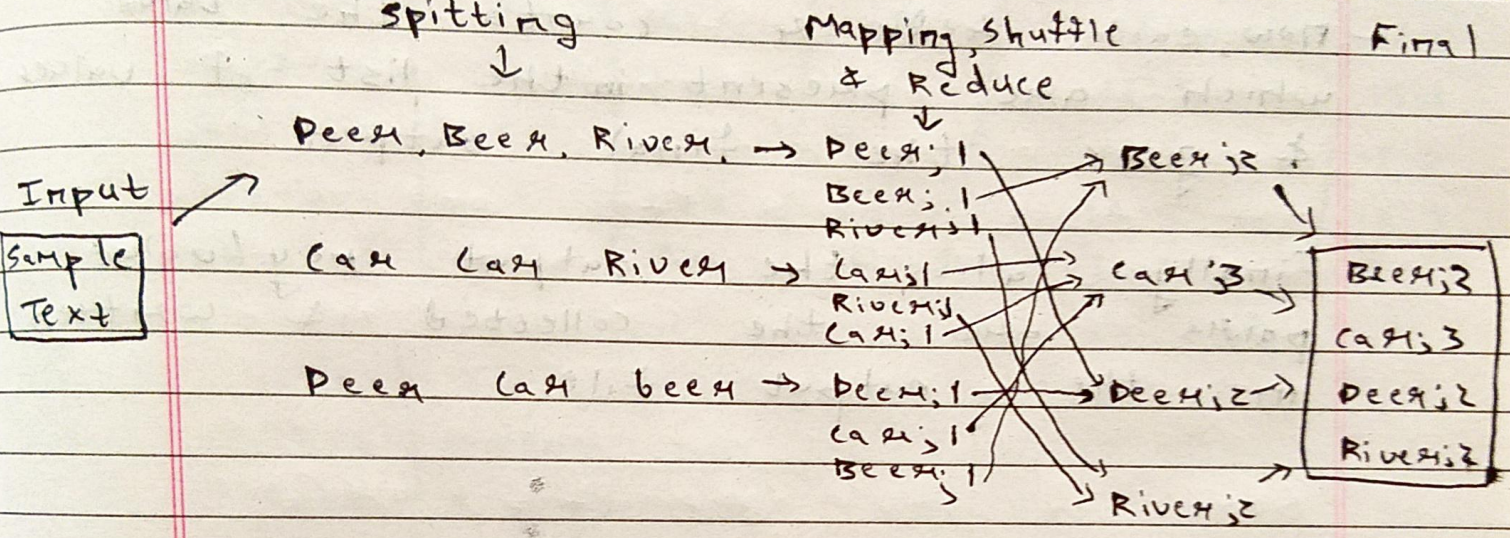
1) Jobtracker - Acts like a master (Responsible for complete execution of submitted job)
2) Multiple Task Trackers - Acts like slaves, each of them performing the job.

For every job submitted for execution in system, there is one jobtracker that resides on Name node & there are multiple tasktrackers which reside on datanode.

-> ~~Wordk~~ Word Count Example:

Suppose one text file contains following data: Deer, Bear, River, Car, ~~River~~ Car, River, Deer, Car & Bear

We have to perform wordcount using MapReduce on this.



- First we divide the input in three splits, this will distribute the work.

- Then we tokenize the work in each of the Mapper.

- Then list the key-value pair, will be created where key the word & value is the occurrence of that word in Mapper.

- After Mapper phase, partition process take place where sorting & shuffling happen so that all the tuples

with some key are sent to the corresponding reducers.

- Now, each reducer counts the value which are present in the list of values & gives the final output
- Finally all the output key/value pairs are the collected & written in the output file.

18DCE115

PAGE NO.:
DATE / /

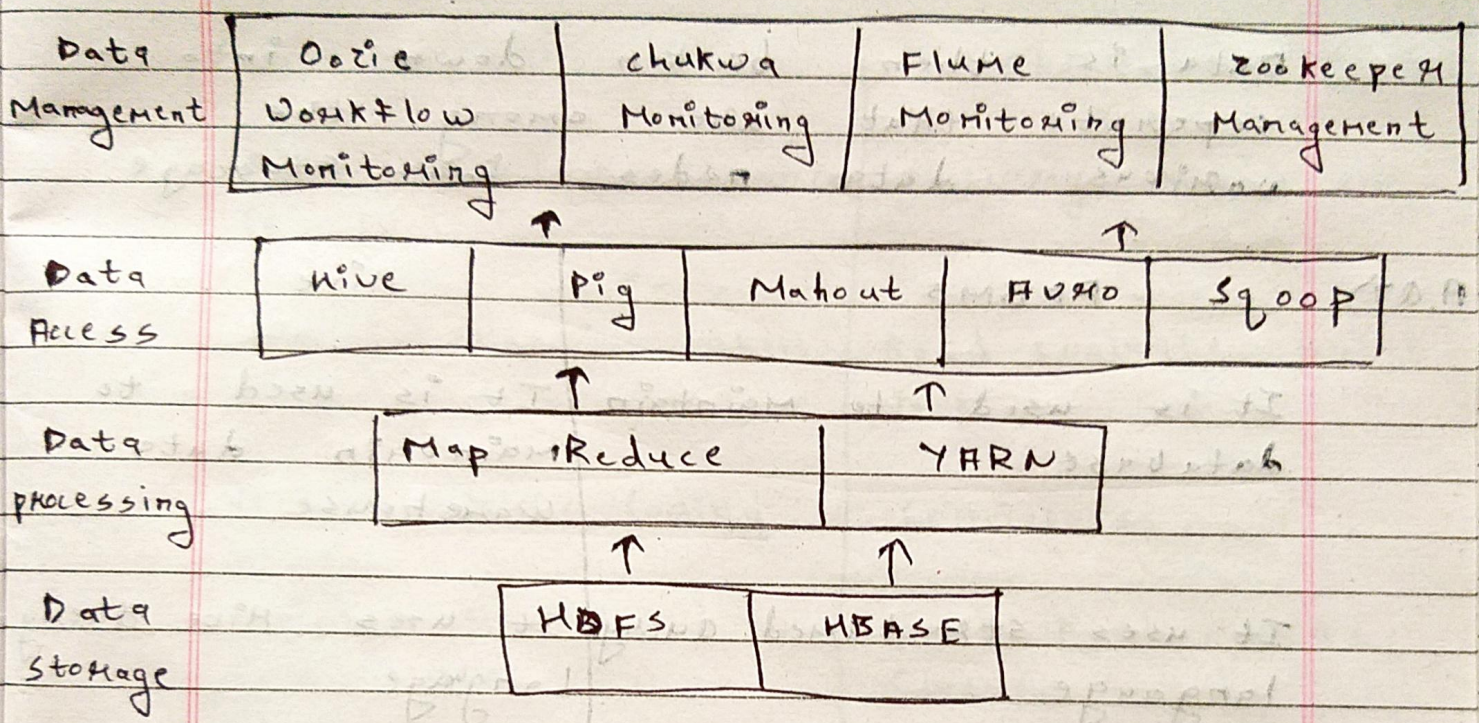
A.Q3) Hadoop ecosystem is a platform or a suite which provides various service to solve the big data problems, It includes Apache projects and various commercial tools & solutions

There are four maior elements of hadoop :- HDFS, MAPReduce, YARN & Hadoop common.

Most of the tools or solutions are used to supplement or support these maior element.

All these tools work collectively to provide services such as absorption, analysis, storage & maintainance of data, etc.

=> Hadoop Ecosystem diagram :-



Q7)

~~RDBMS~~ ~~Hive~~

=> HDFS Work :-

With the HDFS the data is written once on the server & subsequently read and re-used many times thereafter

The way it works is by having a main << Namenode >> & multiple << data nodes >> on a commodity hardware cluster. All the nodes are organised within the same physical rack in the data center.

Data is then broken down into a seperate that are among the various data nodes for storage.

A.Q7)

RDBMS	Hive
It is used to maintain database	It is used to maintain data warehouse.
It uses structured query language	It uses Hive query language.
Tables are sparse	Tables are dense
No partition method is used	sharding method is used for partition.
Normalised data is stored	normalized both type of data is stored
schema is fixed	schema varies

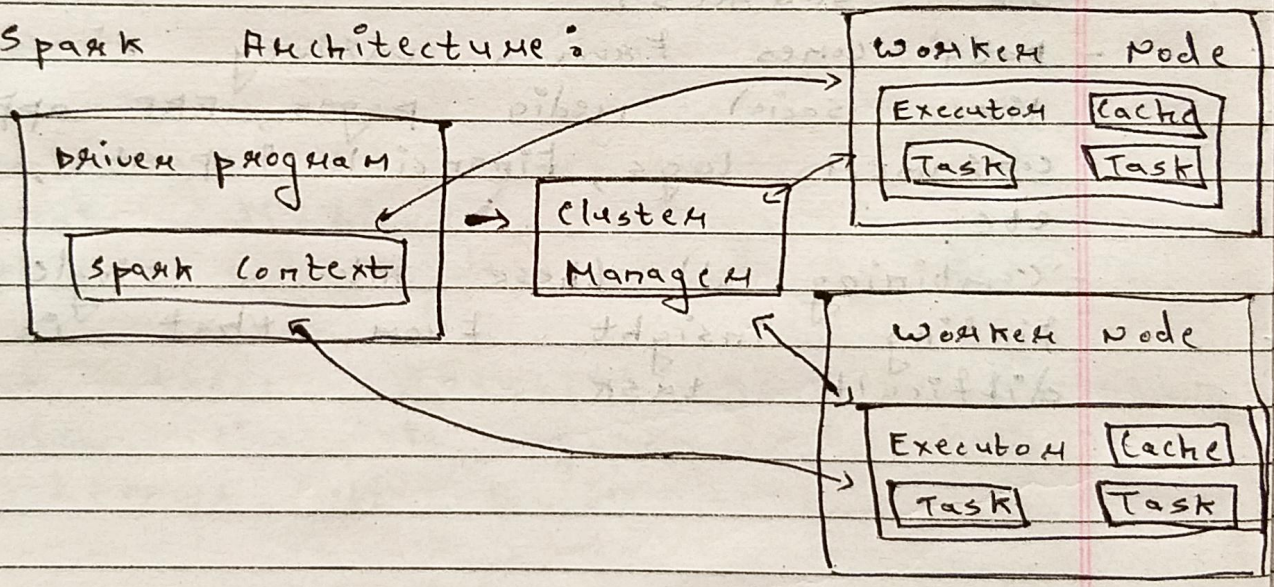
MapReduce	SPARK
Disk based, performance is low	Memory based only
No interactive shell	Read evaluate print loop
Boiler plate coding	Conciseness
Mainly restricted for Java Dev.	Java, python, scala Closure
only for Batch processing	Batch as well as interactive processing
No Graph processing	Graph processing is supported.
Used for writing data into HDFS	Used for fast data processing

A. Q4) The spark architecture depends upon two abstractions:-

- Resilient distributed dataset (RDD)
- Directed Acyclic Graph (DAG)

The spark follows master-slave architecture. Its cluster consists of a single Master & multiple slaves.

Spark Architecture:



- Driver program is a process that runs the main() function of the application & creates the spark Context object whose purpose is to coordinate the spark applications, running as independent sets of processes on a cluster.

- Cluster Manager role is to allocate resources across applications. The spark is capable enough of running on a large number of clusters.

- Worker node is a slave node, its role is to run the application code in the cluster.

- Executor is a process launched for an application on a worker nod. It runs tasks & keeps data in memory or disk storage across them. It reads & write data to the external sources.

- Task is a unit of work that will be sent to one executor.

A.Q5

Range	Bins	Non events	%	Events	%
0-50	1	197	3.9	20	6.8
51-100	2	450	13.5	34	11.6
101-150	3	492	14.9	39	13.3
151-200	4	597	18.1	51	17.4
201-250	5	609	18.4	54	18.4
251-300	6	582	17.6	55	18.7
301-356	7	386	11.6	41	14
		3313		294	

$$WOE \Rightarrow \ln\left(\frac{\%\text{ of non events}}{\%\text{ of events}}\right)$$

$$IV \Rightarrow \sum (\%\text{ non events} - \%\text{ of events}) * WOE$$

Bins	WOE	IV
1	-0.419	0.12771
2	0.1516	0.288
3	0.1135	0.1816
4	0.0397	0.0277
5	0	0
6	-0.0606	0.0666
7	-0.1880	0.4512

$\sum WOE = -0.086$

$\sum IV = 1.9915$

B.Q) Features of spark:-

- Fault tolerance
- Dynamic in nature
- Lazy evalution
- Real time stream processing
- Reusability
- speed
- Advanced Analytics
- in-Memory computing

Transformation refers to the operation applied on a RDD to create new RDD. Filter, groupBy & map are the examples of transformation.

Actions refers to an operation which also applies on RDD, that instructs spark to perform compitition & send the result back to driver.

- The transformations & actions in apache spark are divided into 4 major categories

1) General
2) Data-structure & IO
3) Mathematical & statistical
4) set Theory & Relational

B. Q4) Kibana offers a flexible platform for visualization; it also gives real time updates/summary of the operating data. Grafana is built for cross platform; it is mostly integrated with Graphite, influxDB, and Elastisearch.

Grafana is mainly designed as UI tool for better interaction with users, it accepts data from multiple ~~login~~ pluging data from various sources. Kibana is designed specially to work with ELK stack.

Kibana is developed using Lucene libraries. For querying, kibana follows Lucene syntax. Grafana on the other hand, uses a query editor, which follows different syntax based on the editor it is associated with as it can be used across platforms.

Grafana supports built-in alerts to the end-users. It can send alerts

to the user's email if it finds any unusual data while Monitoring. Kibana by itself doesn't support alerts, but with the help of plugins, it can be made possible.

B.Q6) Challenges :-

1) Lack of proper understanding of big data :
- companies & organizations are failing to develop big data skills which they need due to rapid developement in the field
- companies need to come up with proper solutions for big data

2) Data growth issues :
- Most difficult challenge is to store this huge data properly & keep its backup.
- As these datasets grows exponentially with time, it gets extremely difficult to handle.

3) Confusion with big data tools :
- companies need to choose the tools accordingly to their use case with utmost care.

4) Lack of data professionals :
- The rate with which data is increasing, it is difficult to grow.

5) Securing data:
- The big challenge is security of this data.
- Hackers are smart so, it is difficult task for any organisation to come up with fully secure data plans.

6) Integrating data from a variety of sources:
- Data comes from a variety of sources like social media pages, ERP app.s consumer logs, financial reports, emails, etc.
- Combining all these into a single & finding insight from that is a difficult task.