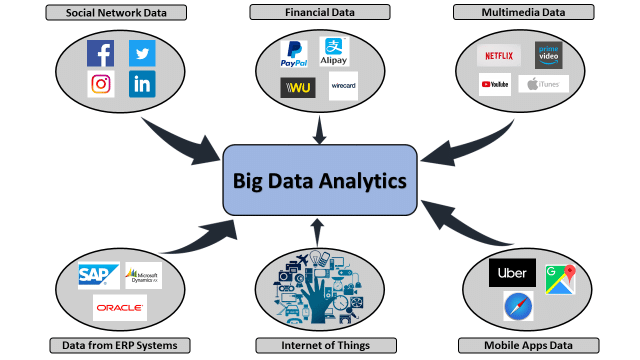
**Assignment 1**

1. What are the Skills and Sources of big data?

Sources of big data:



Other sources:

* Social media
* Web
* Machine logs
* Databases
* Cloud
* Archives
* Iot
* Media
* Government
* Transportation
* Health care

1. What is Big Data Adoption?

Big data adoption **is a process through which businesses find innovative ways to enhance productivity and predict risk to satisfy customers need more efficiently**.

Big data adoption is defined as a process that allows an innovation to alter the infrastructure of an organization. Big data adoption covers advanced information processing techniques and technologies that improve the decision-making process. It gives new opportunities to organizations to leverage information and gain competitive advantage. Big data adoption enhances productivity, predict risk, and satisfy customers more effectively. Adoption is a phase in which technology is chosen by the organization or any business for their usage. Big data adoption provides an opportunity for organizations and industry to surpass the competitors. The adoption of big data might be laborious with a big budget, but the return advantages may develop the path of success in the long run.

1. What is Structured And Unstructured Data

Structures data: Structured data is data that has been predefined and formatted to a set structure before being placed in data storage, which is often referred to as schema-on-write. The best example of structured data is the relational database: the data has been formatted into precisely defined fields, such as credit card numbers or address.

Unstructured data: Unstructured data is data stored in its native format and not processed until used, which is known as schema-on-read. It comes in a myriad of file formats, including email, social media posts, presentations, chats, IoT sensor data, and satellite imagery.

| **Properties** | **Structured data** | **Semi-structured data** | **Unstructured data** |
| --- | --- | --- | --- |
| Technology | It is based on Relational database table | It is based on XML/RDF(Resource Description Framework). | It is based on character and binary data |
| Transaction management | Matured transaction and various concurrency techniques | Transaction is adapted from DBMS not matured | No transaction management and no concurrency |
| Version management | Versioning over tuples,row,tables. | Versioning over tuples or graph is possible | Versioned as a whole |
| Flexibility | It is schema dependent and less flexible | It is more flexible than structured data but less flexible than unstructured data | It is more flexible and there is absence of schema |
| Scalability | It is very difficult to scale DB schema | It’s scaling is simpler than structured data | It is more scalable. |
| Robustness | Very robust | New technology, not very spread | — |
| Query performance | Structured query allow complex joining | Queries over anonymous nodes are possible | Only textual queries are possible |

1. Explain Architecture of Hadoop in Detail.

HDSF component – NameNode and DataNode

NameNode is also called as master node. This node is responsible for maintaining file names, directory names, file system hierarchy, Permissions and ownerships,etc

1. What is HDFS? What are Core Components of HDFS?

The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS has a master/slave architecture.

Core components of HDFS are: NameNode and DataNodes

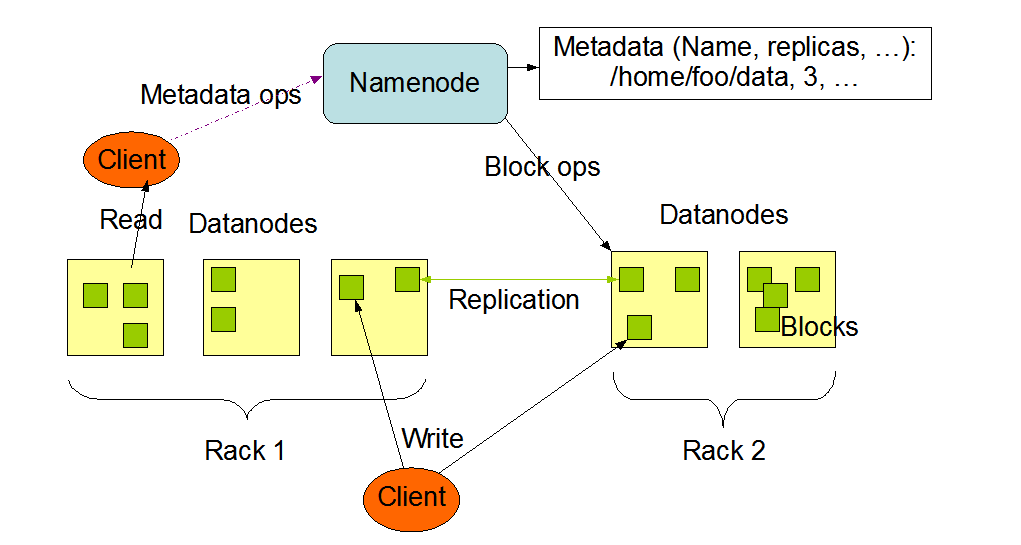
NameNode

* Is the “master” node of HDFS
* Determines and maintains how the chunks of data are distributed across the DataNodes

DataNode

* Stores the chunks of data, and is responsible for replicating the chunks across other DataNodes

1. Explain Architecture of HDFS in Detail.



The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware.

HDFS has a master/slave architecture.

An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on.

**Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes.**

The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system’s clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

HDFS is built using the Java language; any machine that supports Java can run the NameNode or the DataNode software. The NameNode is the arbitrator and repository for all HDFS metadata. The system is designed in such a way that user data never flows through the NameNode.

1. Run the HDFS commands, and add a one liner understanding for each of the command(Practiced on Day 1 and Day 2 ).

Hdfs dfs -mkdir: to create a directory in hdfs

Hdfs dfs -ls: to see all files in hdfs

Hdfs dfs -rm -R: to delete directories

Hdfs dfs -D dfs.blocksize=1048576 -put stocks.csv: to divide the stocks.csv file into blocks of 1MB.

Sudo find / type -f -name <flie\_name> : to find a file

Hdfs fsck /user/cloudera/stocks.csv -files -blocks -location: to find the locations of the blocks created.

Hdfs dfs -put nexon.txt car/tata/suv: to upload a file(nexon.txt) to HDFS from client.

Hdfs dfs -get nexon.txt /vehicle: to get a file(nexon.txt) from hadoop to Client machine

1. Explain Functioning of Name Node,Seconday Name Node,Data Node.

Namenode: NameNode works on the Master System. The primary purpose of Namenode is to manage all the MetaData. Metadata is the list of files stored in HDFS(Hadoop Distributed File System). As we know the data is stored in the form of blocks in a Hadoop cluster. So the DataNode on which or the location at which that block of the file is stored is mentioned in MetaData. All information regarding the logs of the transactions happening in a Hadoop cluster (when or who read/wrote the data) will be stored in MetaData. MetaData is stored in the memory.

Secondary NameNode: Secondary NameNode is used for taking the hourly backup of the data. In case the Hadoop cluster fails, or crashes, the secondary Namenode will take the hourly backup or checkpoints of that data and store this data into a file name fsimage. This file then gets transferred to a new system. A new MetaData is assigned to that new system and a new Master is created with this MetaData, and the cluster is made to run again correctly.   
This is the benefit of Secondary Name Node. Now in Hadoop2, we have High-Availability and Federation features that minimize the importance of this Secondary Name Node in Hadoop2.

DataNode: DataNode works on the Slave system. The NameNode always instructs DataNode for storing the Data. DataNode is a program that runs on the slave system that serves the read/write request from the client. As the data is stored in this DataNode, they should possess high memory to store more Data.

1. Explain 3 V’s in Big Data.

The three V’s of bid data: Volume, Velocity, Variety

Variety: Variety refers to the number of types of data being

Generated. Varieties of data include structured, semi-structured,

and unstructured data arriving from a myriad of sources. Data can be gathered from databases, XML or JSON files, text documents, email, video, audio, stock ticker data, and financial transactions.

Volume: Volume refers to amount of data being generated. Think in terms of gigabyte, terabytes, petabytes.

Velocity: Velocity refers to the rate at which new data is created. Think in terms of megabytes per second and gigabytes per second. Data is streaming in at unprecedented speed and must be dealt with in a timely manner in order to extract maximum value from the data. Sources of this data include logs, social media, RFID tags, sensors, and many more.

1. What is Apache hadoop Fault Tolerance?

It is also fault tolerant. Hadoop services become fault tolerant through redundancy. For example, the Hadoop distributed file system, called HDFS, automatically replicates data blocks to three separate machines, assuming that your cluster has at least three machines in it. Many other Hadoop services are replicated too in order to avoid any single points of failure.

1. What is Distributed Processing?

Hadoop also uses distributed storage and processing Large datasets are automatically split into smaller chunks, called blocks, and distributed across the cluster machines Not only that, but each machine processes its local block of data. This means that processing is distributed too, potentially across hundreds of CPUs and hundreds

of gigabytes of memory

1. Hadoop Vs Relational Database?

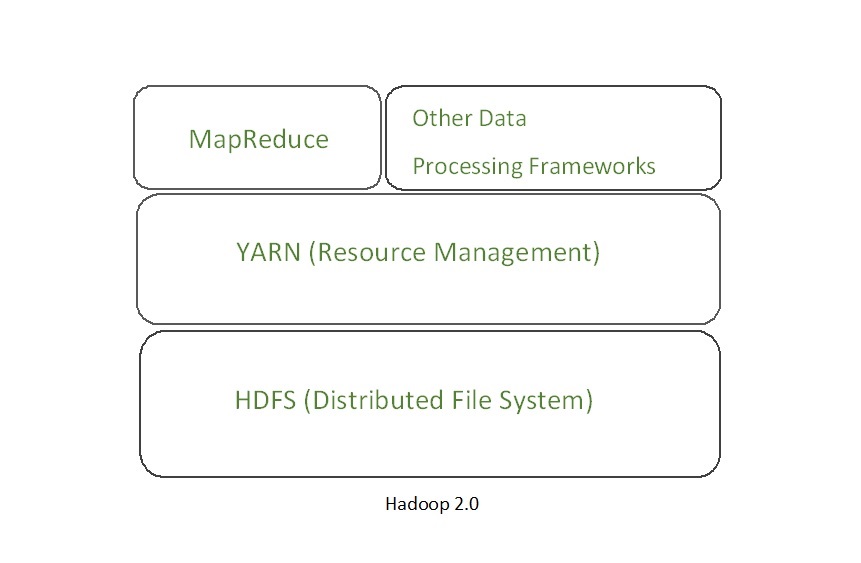
**RDMS (Relational Database Management System):** RDBMS is an information management system, which is based on a data model.In RDBMS tables are used for information storage. Each row of the table represents a record and column represents an attribute of data. Organization of data and their manipulation processes are different in RDBMS from other databases. RDBMS ensures ACID (atomicity, consistency, integrity, durability) properties required for designing a database. The purpose of RDBMS is to store, manage, and retrieve data as quickly and reliably as possible.

**Hadoop:** It is an open-source software framework used for storing data and running applications on a group of commodity hardware. It has large storage capacity and high processing power. It can manage multiple concurrent processes at the same time. It is used in predictive analysis, data mining and machine learning. It can handle both structured and unstructured form of data. It is more flexible in storing, processing, and managing data than traditional RDBMS. Unlike traditional systems, Hadoop enables multiple analytical processes on the same data at the same time. It supports scalability very flexibly.

1. What is YARN?

YARN stands for “***Yet Another Resource Negotiator***“. It was introduced in Hadoop 2.0 to remove the bottleneck on Job Tracker which was present in Hadoop 1.0. YARN was described as a “*Redesigned Resource Manager*” at the time of its launching, but it has now evolved to be known as large-scale distributed operating system used for Big Data processing.

 YARN architecture basically separates resource management layer from the processing layer. In Hadoop 1.0 version, the responsibility of Job tracker is split between the resource manager and application manager.



YARN also allows different data processing engines like graph processing, interactive processing, stream processing as well as batch processing to run and process data stored in HDFS (Hadoop Distributed File System) thus making the system much more efficient. Through its various components, it can dynamically allocate various resources and schedule the application processing. For large volume data processing, it is quite necessary to manage the available resources properly so that every application can leverage them.

**YARN Features:** YARN gained popularity because of the following features- 

* **Scalability:** The scheduler in Resource manager of YARN architecture allows Hadoop to extend and manage thousands of nodes and clusters.
* **Compatibility:** YARN supports the existing map-reduce applications without disruptions thus making it compatible with Hadoop 1.0 as well.
* **Cluster Utilization:**Since YARN supports Dynamic utilization of cluster in Hadoop, which enables optimized Cluster Utilization.
* **Multi-tenancy:** It allows multiple engine access thus giving organizations a benefit of multi-tenancy.