* Day 1 :
  + Introduction Of Big Data
  + Challenges of Hadoop
    - Hadoop
    - Mongo DB
    - Spark
  + Internals of hadoop
  + Ecosystem
* Day 2:
  + Hardware and Software Requirement
  + Advance Feature of Hadoop
  + Hands on Installation
* Day 3 :
  + PIG : Yahoo
  + HBASE : Facebook
  + Hive : Amazon
  + Integeration
* Day 4 :
  + Project
* Day 5 :
  + Sqoop
  + oozie
  + Flume
  + Chukwa
  + Basic Admin Activity

http://hadooptutorial.info/

* Introduction to big Data :
* Data Source :
  + Mobile data
  + Scientific Data
  + Environment Data
  + Financial Data
  + Industrial Data
  + Agriculture Data
  + Social Media Data
  + Server Logs
* Data which is processed is known as Information
* Data Classification
* 2 parts
  + **Structured Data**
    - Definite Architecture :Stored in DB2, or Excel sheet
  + **Unstructured Data**
    - No specific architecture : document, emails,video , audio
    - 80 % data is unstructured data
* Combination of both Structured and Unstructured or only structured or only unstructured data constitutes **Big Data**
* **Big Data Identification** :
* **Thumb rules** to follow at every big data
  + Understand if actually given data is big data or not
  + Also, know the Flow of application very well
* 3 factors to keep in mind if given data or not :
  + **Data Velocity** :
    - Type of processing:
      * Real Time
      * Batch processing
  + **Data Volume** :
    - huge amount of data
  + **Data Variety** :
    - Type of data we have : Ex SMS and Videos.
* Veracity :
  + Make decision
* Challenges :
  + Data Storage :
    - Economical Data Storing
  + Data Manipulation :
    - Processing
* To overcome this we have Hadoop
* Introduction to Hadoop Environment :
* Apache Hadoop : Project of Google and known as Big Table
  + Open Source
  + License to Apache
  + Framework : Collection of various Infrastructure path that includes :
    - programming language,
    - database,
    - datawarehouse,
    - applications (oozie),
    - data migration,
    - scheduling.. etc
  + used for distributed processing
    - Deploy application on different data centre
  + Commodity computers :
    - low cost computers.
  + Simple Programming Module
    - Use any language :Java, C, C ++ etc to write the module.
* Hadoop Structure
  + hadoop
    - HDFS
      * Name Node
      * Data Node
    - Map Reduce
      * Job Tracker
      * Task Tracker
* Data Storage :
  + HDFS (Hadoop Distributed File System)
* Data Processing :
  + MapReduce : can be achieved using any programming language
* Background Service : for windows and Daemons in Unix Environment
* There are total 5 daemons which constitute both HDFS and Map Reduce
  + Name Node
  + Secondary Name Node : nothing to do with Name Node
  + Data Node
  + Job Tracker
  + Task Tracker
* Architecture :
  + Hadoop Follows Master and Slave kind of architecture
  + It is a multi node structure
  + Name node and Job Tracker : run on Master servers
  + Data node and Task Tracker are optional in master server
  + Where ever Data node is running task tracker should run on same Server
* Modern DATA Architecture
  + Data Source :
    - Existing Source
  + Data System :
    - Repository
    - HAdoop : Pls do not mix Hadoop with Oracle Database
  + Applications :
    - Business Analysis
    - Custom Application
    - Packaged Application
* Use Cases : BANK
* Companies which provide Hadoop Framework :
  + Cloudera
  + Hortworks
  + Map R
  + IBM
  + Microsoft
* HDFS : is similar to Windows and Unix File System
* Hadoop Implementation :
* Operational Store : date used for transaction purpose
* HDFS (staging)
* HDFS (Retention)
* New Data is stored for HDFS
* From HDFS the ETL is done then the result is used for further Analytics.
* Implementation
  + High Volumne Data Flow :
    - Science
    - Industry
    - Legacy
    - System Data
  + Map Reduce Process
    - Create map and Reduce
  + Consume Results
    - Import to RDBMS
* Transaction :
  + A : Atomicity
  + C : Consistency
  + I :Isolation
  + D :Durability
* Reason not to install Hadoop in Windows
  + Windows :
    - Virtual Machine
    - Ecosystem : Ubuntu/ Cent OS
    - and then install Hadoop
* Installation of Hadoop :
  + Apache :
  + Third party
* Ecosystem :
* YARN : Yet Another Resource Negotiator
* HBASE :
  + NO SQL(Not Only SQL) Database ,
  + developed by Amazon
* HIVE :
  + Part of Data Warehouse
  + Programming Language : Hive QL : Similar to SQL
  + Facebook
* Sqoop :
  + SQL + Hadoop
  + Migrate Data to Hadoop and Vice versa
* Oozie :
  + Work Flows
    - Ex Data flow from Oracle to HDFS and then to Pig and then MYSQL
  + 2 Kinds of scheduling
    - Data Bound Scheduling : workflow
    - Time bound Scheduling : executing work flow at particular time
* PIG :
  + Programming Language
  + developed by Yahoo
* Zookeeper :
  + High availability
  + Maintenance of Server which is present hadoop cluster
* Flume :
  + Transfer of data Within Hadoop Environment
* Spark :
  + Execution engine

2 kinds of execution engine :

* + - Spark Execution Engine :Real time
      * 80 %-100% is faster than Map Reduce Execution
    - Map Reduce Execution Engine
* 3 ways to use Spark :
  + Pig : STORM
  + Hive :SHARK
  + Scalar (Programming Language)
* HDFS Characteristics :
  + Involves commodity servers
  + supports Parallel reading and processing data
  + Built in Redudancy
  + Automatically manage addition and removal nodes
  + Versatile,resilient and clustered approach to manage files
  + Breaks data into blocks
  + default size of datablock is 128 MB MB
    - 50 MB > 64 MB
    - 100 MB > 128 MB
    - 300 mB >
  + Replication is done to maintain the availability of Data
  + Default Replication Factor =3
  + Compromise of Name node, secondary name node
* Points of Criticality :
  + Replication Factor for business can be 10 or 12
  + Non Buisness : Replication factor is 1
* HDFS Components :
* Name Node :
  + Master of server
  + Master of System
  + single point of investment
  + maintains and manages the blocks which are present on the data node
  + Name node should not be commodity hardware
  + Never contains actual data
  + Custodian of Metadata (data about data)
* Data Node :
  + Custodian of user Data
  + Slaves which are deployed on each machine and provide the actual storage
  + Responsible for serving read and write request for client
* Secondary Name Node :
  + Read data  from RAM and writes the same to Hard Disk
* Writing in Hadoop is Serial
* Reading in Hadoop is Parallel
* Rack Awarness :
* First data block get stored in rack which is closer to client place
* Then, 2 data block gets stored in the next rack but in different server of same racks
* ReplicationL
* Under Replication :
  + No of Blocks < Replication Factor
* Over Replication
  + No of Blocks > Replication Factor
* TO Check Replication Factor
  + hadoop fs –ls

or

hadoop fs –stat

* Configuration Replication Factor
  + hdfs-site.xml

<property>

<name> dfs.replication <name?

<value> 3</value>

<description> Block Replication</description>

<property>

File base replication

hadoop fs –setrep –w 3 /my/file

replication for all files under a directory :

hadoop fs –setrep-w 3 – R /my/dir

Configuring Block Size :

* <property>
* <name>dfs.block.size<name>
* <value>134217728<value>
* <description>Block size<description>
* <property>
* **Introduction to Map Reduce :**
* Derived from two terms
* Map and Reduce
* Map: breaking data + Apply business logic
* Reduce : Data consolidation
* Daemons for Map Reduce :
  + Job Tracker
  + Task Tracker
* Map Reduce Paradigm
* Splitting :
  + default is new line
  + Key Value pair :
    - Based on loation number and value at that position
* Mapper ->
  + Buisness logic is written
* Shuffling : controlled by task tracker
* Reducing
* Final Results
* File Types and Formats :
* Mapper : breaking logic
* Reducer : Consolidation logic
* Driver Class : Application Logic (Files and Frmat)
* NODES :
  + Master Node :
    - Name Node
    - Secondary Name node
    - Job Tracker
  + Slave :
    - Datanode
    - Task Tracker
* Hardware Requirement :
  + Master Node :
    - 4-6 ITB hard Disk
    - 2quad -2.25GHz
    - 64-512 GB of RAM
    - 10gigabit Ethernet
  + Slave Node :
    - 4-6 ITB hard Disk
    - 2quad -2.25GHz
    - 64-512 GB of RAM
    - 10gigabit Ethernet
* JBOD: just a bunch of disks
* Software Requirements :
  + Red Hat Compatible systems
  + SLES systems
  + Debian systems
  + Ubuntu systems
* Package Installation :
* RECAP :
* We have commercial vendors of Hadoop :
  + cloudera
  + Hortonworks
  + Map R
  + Data Stacks
* There is no much difference in all the above architecture except for Data Stack
* **cloudera, Hortonworks and Map R** follow Master slave architecture
* **DataStack**
* Data Stack follow : Ring Topology.
* Data Stack has a database known as Cassandra
* Architecture of DataStack : No master and slave concept.
* It is a Peer to peer process : every server knows about its peer
* Replication factor of DataStack is same as other
* Application can be deployed on any of the server.
* Consider 5 server. In server number 3 we have the application. Now, that a request from client has been in received in server no 5, then server no 5 is the coordinator . The coordinator, request peer for the shortest distance towards the application. Using the shortest path it reaches the application.
* **Advantages:**
  + Simplicity :
  + consistency
  + integeration
  + versioning

**Important paths :**

* + /etc/hadoop :
    - hadoop configuration files .
    - equivalent to conf directory
  + /etc/init.d :
    - init script for each hadoop daemons
  + /usr/lib :
    - C libraries of hadoop
  + /usr/libexec :
    - Miscellaneous files used by various libraries and scripts that come with Hadoop
  + /usr/sbin :
    - Helper shell scripts used by all administrators of hadoop are installed here
  + /usr/share/doc/hadoop :
    - License, NOTICE and readme files
* **Major Configuration Files :**
  + Hadoop-env.sh
    - Regarding all the environment info
  + Core-site.xml :
    - application details
    - ip address and port number
  + Hdfs-site.xml :
    - size of block
    - replication factor
  + Mapred-site.xml:
  + Log4j.properties:
    - Logging message
  + Slave (optional)
  + Capacity-scheduler.xml(optional)
  + Hadoop-policy.xml
  + TaskController.cfg
* Machine : hadoop-cdh-04 / 45.79.161.237
* username :root
* pwd : India@123
* Installation command
* ssh root@ipaddress
* pwd :India@123
* chkconfig iptables off
* chkconfig ip6tables off
* Stop Service:
* service iptables stop
* service ip6tables stop
* important Step :
* Host File contains Ip address and host name of all the service which is participating in the cluster
* vi /etc/hosts
* copy the following
* 45.79.161.237 hadoop-cdh-04
* Download cluster
* wget <http://archive.cloudera.com/cm4/installer/latest/cloudera-manager-installer.bin>
* Change mode of the file downloaded
* **chmod u+x cloudera-manager-installer.bin**
* Execute File

**./cloudera-manager-installer.bin**

* Once installation is done..
* Go to browser , type the <http://ipaddress:port_number>
* Username/password :admin/admin
* Select cloudera standad version and click continue
* Enter the ipaddress
* Hue : GUI
* <http://ipaddress:8888>
* first time the first user who logs in becomes : administrator
* username : root
* password : India@123
* Introduction to YARN :
  + yet Another resource Negotiator - YARN
  + Flow of Map Reduce :
    - Job tracker has to give job to task tracker.
    - Before giving job the job tracker has to check for the infrastructure availability of Task Tracker
  + When to use YARN :
    - Yarn when it is real time analysis
    - my cluster has more compute capacity than Big Data
  + YARN includes :/ How YARN works
    - 3 services :
      * Global Resource Manager
        + Purpose : is to ensure resource availability.
      * Application master
        + Transfering task from application to certain set of cluster.
      * Node Manager :
        + Runs on All the servers
  + Protocol :
    - Client to RMs : Submit to App Master
    - RM to NM : Start the app master
    - AM to RM :Request/Release containers
    - RM to NM : Start Task in conatainer
* Scalalbility:
  + Horizontal
    - Adding maximum servers
  + Vertical
    - adding more resources
* Command to load file to HDFS:
  + hadoop fs – ls /usr/root
* Execution:
* **Binary Code** : is low level language
* **High level language**
* In between these two we have **compiler or interpreter** in this code.
* Above High Level is the **Hadoop**
* In between **high Level Language** and **hadoop** we have execution engine
* Processing and Analytics
  + Data Alignment
  + Data Rectification
  + Business Value Selection
  + Final output
* PIG :
* Open Source
* High Level Data Flow system for data stored in HDFS
* Provides simple language for queries and data manipulation
* Programming Language :
* Object oriented
* functional programming
* SQL
* Converted to base code
* Pig is functional Programming Language
* work on any kind of data
* Subset of Map Reduce:
* Belongs to map reduce
* PIG Steps
* Load the data :
* Moving particular set of data
* Business logic:
* Set of transformation.
* DUMP :
* Output on screen : DUMP variable
* Store output in some file : STORE filename HDFS\_path
* Pig command leads to PIG terminal
* Components of PIG :
* Pig Latin
* Program written
* Pig Execution Engine:
* Compiler convert PIG code to Map Reduce
* Working of PIG:
* GUI
* Command line interface
* GRUNT
* Store file with .pig extension
* Two kinds of Loader:
* Text Loader
* Pig Loader : To load the content to concerned variable
* Hands on PIG : http://hortonworks.com/hadoop-tutorial/word-counting-with-apache-pig/
* in Hue,
* Upload File :
* Go to File Browser and click on upload
* Select file from local machine
* Click on the upload
* To Get the count
* HUE :
* Go to Pig Editor
* **myinput= load “HDFS\_path/filename” using TextLoader as (myword:chararray) ;**
* (myword:chararray) : optional
* **word = FOREACH myinput GENERATE FLATTER (tokenize(myword));** : aligns word in the straight line
* **grouped = GROUP words BY $0;** : Grouping words according to the column ,
* $0 specifies the column of the group.
* **count = FOREACH grouped generate group, COUNT(word) ;**
* **Store count into ‘hdfs-path’**
* GRUNT :
* In command line type pig
* hadoop fs –ls /user/root
* HIVE :
  + SQL kind of query language
  + Data Warehouse kind of requirement using
    - database
    - tables
    - Partitioning
  + Associated to this directories will be created in HDFS Execution :
    - GUI
    - CLI
    - JDBC/ODBC :Programming Language
* Data Model
* **Table :**
  + Like SQL
  + in /user/hive/demo.db is created
  + Create Table is stored in demo.db folder
  + We can store bulk of data into Hive environment and single data
  + **Load data in path =”Path in HDFS?file\_name” into table \_name**
  + hands On

**Create database DEMO;**

**create table e\_name (name string);**

**load data inpath ‘hdfs path/file\_name’ into table e\_name;**

* **Partition :**
  + where clause
  + select \*from emp where designation=’manager’;
  + I.e, grouping the data according to particular column value.
  + By doing this an folder would be created.
  + It is specified at the time of table creation
  + Always Remember :
    - First create a table with non partition
    - Secondly, create a table with partition
  + Two types of Partitioning
    - Static Partioning
      * Data should be contain column names in Partition which is not used in the where clause.
* **Bucket :**
* Number of buckets
* division happens by hash algorithm