# Day1

## Hadoop Introduction

### What is Big Data?

- Huge Data

- Keeps growing

- Structured and Unstructured in nature

- Challenges with traditional RDBMS systems to handle Big Data

--------------------------------------------------------------------------------

### What is Hadoop?

- System to handle Structured and Unstructured data

- Ecosystem of many tools (Ex. HDFS, MapReduce, YARN, Pig, Hive, HBase, Flume, Sqoop, Spark, Impala...)

- Distributed Computing Framework to handle Big Data

- Fault Tolerant

- Open Source

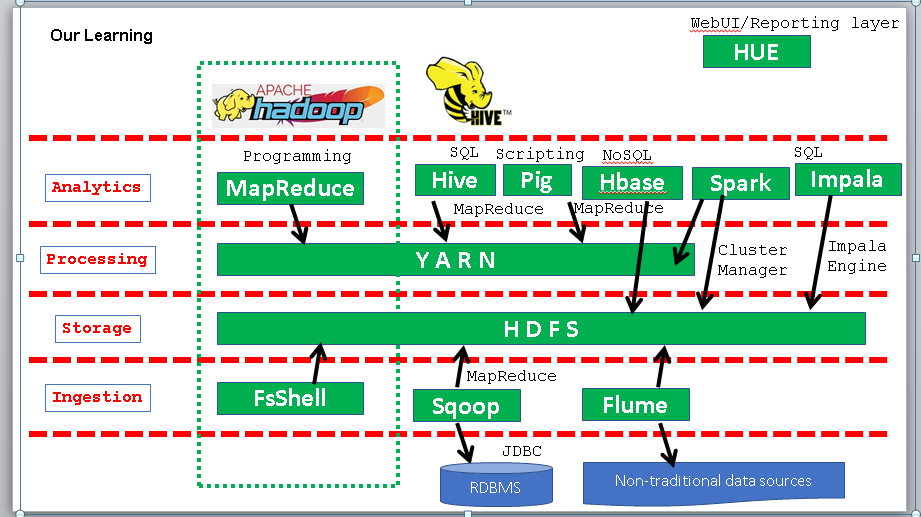
- Helps to gain better insights

- Can run on low cost infrastructure

- Linearly scalable

- Fast processing

### Hadoop Ecosystem



================================== Tea Break ========================================

setting up VM

### Hadoop Roles

Hadoop Developer & Analyst

- Pre-Requisites: Programming skills | Java | Scala | Python and SQL

- Hadoop Tools: MapReduce, Hive, Impala, Pig, HBase, Spark etc

Hadoop Administrator

- Pre-Requisites: Administration skills | Linux Admin | DBA

- Hadoop Tools: Cloudera Manager, Ambari, ZooKeeper, Sentry etc

Data Scientist

- Pre-Requisites: Statistics and R | Python, SQL

- Hadoop Tools: Mahout, Spark ML, SparkR etc

## Hadoop Architecture

fs command - Getting Started.txt

hadoop fs <command> (hadoop 1.0) OR hdfs dfs <command> (hadoop 2.0)

Hadoop = FsShell + HDFS + MapReduce + YARN

FsShell - Lets a user interact with HDFS

HDFS - Distributed File System provided by Hadoop --> Storage (Disks)

MapReduce - Programming paradigm for distributed data

YARN - Cluster Resource Management --> Compute (CPU + Memory)

With Hadoop, we talk about a paradigm of putting our program on data

Each node within the cluster, has storage and also compute

====================================Lunch ============================================

## HDFS

### Overview of HDFS

Hadoop processes that need to be running (Hadoop Daemons)

Hadoop is a master-slave architecture

HDFS

- NameNode (Master)

- DataNode(s) (Slave)

- SecondaryNameNode (Master) --> Out of date, instead we have to setup a StandbyNameNode

YARN

- ResourceManager (Master)

- NodeManager(s) (Slave)

--------------------------------------------------------------------------------

--------------------------------------------------------------------------------

Our discussions will be pertaining to 1 cluster.

1 master / cluster

--------------------------------------------------------------------------------

--------------------------------------------------------------------------------

Note: The system is designed in such a way that the user data never flows through the NameNode

--------------------------------------------------------------------------------

### HDFS Architecture

**NameNode**

- Arbitrator - makes all decisions

- allocates data nodes depending on the replication factor

- Manages the File System metadata

- Metadata is stored as files on the local file system of NN

- Persistence of the File System Metadata

- FsImage

- EditLog (Transaction Log)

**DataNode**

- Handle read / write operations

- Listen to the NN

- DataBlocks are stored on DataNodes

- DataBlocks are stored as files on the local file system of DN

**SecondaryNameNode**

- Not a hot backup for the NameNode

- Checkpoint Node - periodically merges EditLogs with the FsImage

- 1 hour by default or 1 million transactions on the EditLog (whichever is earlier)

Note: NameNode is the Single Point of Failure

**StandbyNameNode**

- To be configured to achieve High Availability for the NameNode

- SecondaryNameNode is removed and StandbyNameNode is configured

- This feature does not come by default

- Requires ZooKeeper for failover and JournalNodes for sync of edits

## Map Reduce

- Programming model for distributed data

- Job

Understanding MapReduce with an example:

Problem Statement: Find out maximum close price / stock symbol

Input Set in HDFS - /Stocks

Mapper Class - map method

Reducer Class - reduce method

A MapReduce program comprises of a mapper class, a reducer class and a main class

Mapper - map method - Write the logic for transformation

Reducer - reduce method - Write the logic for aggregation

Involves 2 phases for a developer

For the framework

- Input Split

- Map

- Shuffle & Sort

- Reduce

- Final Output

**Note: The map intermediate results are stored on the local file system of the node where the map task was running**

--------------------------------------------------------------------------------

No of maps is equal to the number of blocks where data is stored.

**Example**

1000 mode cluster

1 GB files

128 MB block

**8 Map jobs** will be there (1024/128=8,replication copies should not be consideration as it only for fault tolerance )

**By default 1 reduce** if we need to increase the reduce we can program it

### Demo 1 MaxClosePrice

MaxClosePrice(Labs folder)

MaxClosePrice.java, MaxClosePriceMapper.java, MaxClosePriceReducer.java

JAR file mcp.jar

MaxClosePrice:Submit job

$ hadoop jar /home/cloudera/Desktop/Labs/mcp.jar MaxClosePrice /Stocks /OP2**Check the Output:**$ hadoop fs -cat /OP2/part-r-00000

### Demo 2 WordCount

MapReduce WordCount program: Count each word in the input set

/Sample

Welcome to Hadoop

Learning Hadoop is fun

Hadoop Hadoop Hadoop is the buzz

Expected Output?

Hadoop 5

Learning 1

Welcome 1

buzz 1

fun 1

is 2

the 1

to 1

Logic for map - Read each record, convert into words. Mark each word as key and assign a value 1

Logic for reduce - For each key, sum the values within the list

$ yarn jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar wordcount /Sample /OP1$ hadoop fs -cat /OP1/part-r-00000

## HIVE (Day 1)

- Hadoop related project

- Donated by Facebook in 2008

- The Apache Hive facilitates reading, writing, and managing large datasets residing in distributed storage (HDFS) using SQL

- Hive is not an RDBMS

- Hive offers SQL interface for Hadoop datasets

- Structure can be projected onto data already in storage

- Hive tables are stored in a metastore DB (For our VM it is MySQL)

### Demo1 MaxClosePrice

MaxClosePrice-Hive.txt

### Demo 2 WordCount

Hive\_WordCount.txt

### Hive architecture

Hive is schema on read while RDMS are schema on write

Hive query can be used to query unstructured data

# Day 2

## Yarn Architecture

- A MapReduce program under execution is a job (application)

- MapReduce job has tasks

- map tasks

- reduce tasks

- Tasks are the actual units of execution

- Compute resources are required for task execution - CPU+Mem - YARN is responsible for allocation

- YARN Components

- ResourceManager - 1 / cluster

- ApplicationsManager

- Scheduler

- NodeManager - 1 / DN

- Containers - Compute Resources - CPU+Mem (YarnChild) - 1 / task

- Allocated for execution of map / reduce tasks

- taskJVM

- ApplicationMaster - 1 / job - (MRAppMaster)

- The first container for the job

--------------------------------------------------------------------------------

# of map tasks = # of input splits = # of blocks

# of reduce tasks = 1

### Demo2 Yarn process creature for big file

Window 1

jps | sort -k 2

Window 2

Override blocksize from 128 MB to 2 MB

hadoop fs **-D dfs.blocksize=2m** -put Desktop/Labs/SampleBigFile.txt /Sample/

yarn jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar wordcount /Sample /OP4

***Yarn resource manager logs***

**ApplicationMaster**

Attempt Number Start Time Node Logs

**1** Mon Nov 12 21:49:02 PST 2018 quickstart.cloudera:8042 logs

Task Type Total Complete

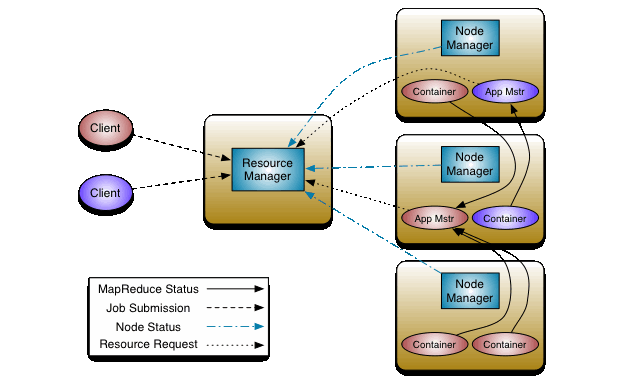
**Map 3 3**

**Reduce 1 1**

Attempt Type Failed Killed Successful

**Maps 0 1 3**

**Reduces 0 0 1**



http://hadoop.apache.org/docs/r2.7.5/hadoop-yarn/hadoop-yarn-site/YARN.html

## HUE

http://quickstart.cloudera:8888/hue

create directories, upload file create able and select query from Hue UI

## SQOOP

Sqoop is a tool to transfer data from RDBMS to Hadoop and vice versa

### Demo 1 retail\_db

*Sqoop.txt*

sqoop import --connect jdbc:mysql://localhost:3306/retail\_db --username root -P --table categories --target-dir /categories -m 1

-m - to specified no. of map task to be used

-p - to prompt for password

Sqoop will create java file, create the jar and submit the jar to yarn for exaction.

cloudera@quickstart ~]$ ls -ltr

-rw-rw-r-- 1 cloudera cloudera 14124 Nov 12 22:44 categories.java

-rw-rw-r-- 1 cloudera cloudera 16344 Nov 12 22:46 orders.java

-rw-rw-r-- 1 cloudera cloudera 22555 Nov 12 22:47 order\_items.java

-rw-rw-r-- 1 cloudera cloudera 27626 Nov 12 22:51 customers.java

## HIVE (Day 2)

### Joins

Joins.txt

### Sub query

not support sub query

### HIVE cheat sheet

hql\_cheat\_sheet.pdf

### Partitioning

Hive\_Partitions.txt

txns.txt

whenever we are not specifying the location hive will create table in /user/hive/warehouse

### UDF

1. create java project in eclipse

LowerA.java

1. add below jar

hive-exec.jar -usr/lib/hive/lib

hadoop-common.jar - usr/lib/hadoop

1. Export jar name udf.jar;
2. add jar to hive

* we can add the jar to hive in config
* we can also add jar to hive current session

hive> add jar file:////home/cloudera/Desktop/Labs/udf.jar;

1. register function - this is one time activity and it will get register in the meta store

hive> create function LowerA as 'hiveUDF.LowerA';

OK

Time taken: 0.111 seconds

1. to get the list of jar

hive> list jar;

/usr/lib/hive/lib/hive-contrib.jar

file:////home/cloudera/Desktop/Labs/**udf.jar**

## Impala

* No map reduce
* Run faster than hive

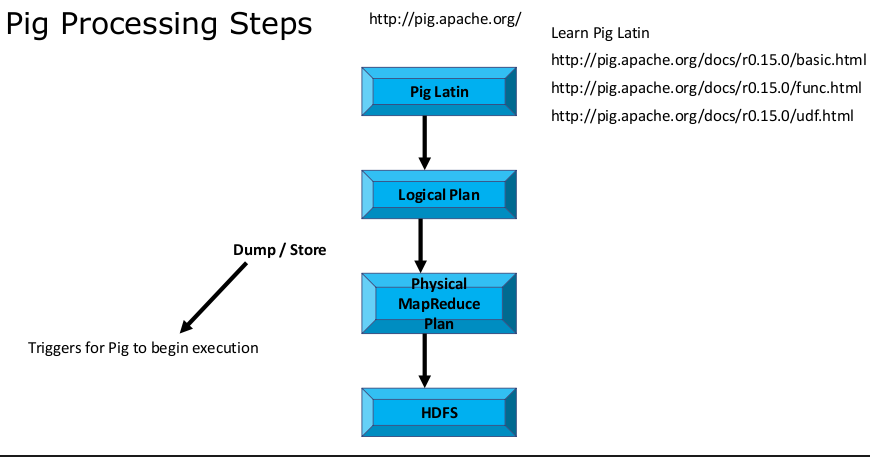
use impala for short lived query . not fault tolerant

use hive for log running query. fault tolerant process

==================================Post Lunch ==========================================

## PIG

Pig is scripting language



### Demo 1 MaxClosePrice

MaxClosePrice-Pig.txt

### Demo 2 MaxClosePrice without creating schema

PigScript.txt

Run the script in non interactive pig mode

[cloudera@quickstart ~]$ pig file:///home/cloudera/Desktop/Labs/PigScript.txt

## SPARK

Spark is a general purpose, unified, large scale data processing engine

- Spark jobs can be written using Scala | Java | Python | R and SQL

- Spark can read data from many data sources like HDFS, Cassandra, S3, HBase, local...

- All analytics using Spark!

- Spark is built in Scala

Spark uses an abstraction called RDD

### What is an RDD?

- Fundamental data strucutures of Spark

- Resilient Distributed Dataset

- RDDs are immutable

- RDD operations are of 2 types

- Transformations (Lazily evaluated)

- Actions (A job is triggered)

- http://spark.apache.org/docs/1.6.0/programming-guide.html#transformations

What is Scala?

- Programming language

- Distributed applications

- Object Oriented

- Functional programming

- Statically typed

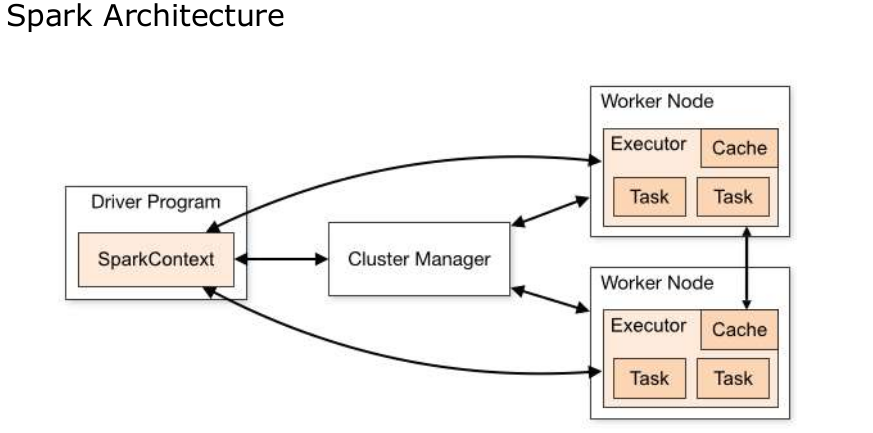
- Scala code runs in a JVM

Scala offers an interactive shell to build our programs faster. This shell is called a REPL

REPL - Read, Evaluate, Print and Loop

Python and R also offer REPL

### Architecture



### Scala programming Basics

$ spark-shell

scala> 10 + 2

scala> res0 \* 2

scala> val str = "Hello World"

scala> str = "Hi"

scala> val str = "Hello Spark"

scala> var str1 = "Hi"

scala> str1 = "Hello"

scala> str1 = 100

scala> val x = 100; val y = 90; x - y;

scala> if (x > 100) 1 else 0

scala> for(i <- 1 to 5) print(i\*10)

scala> for(i <- 1 to 5) println(i\*10)

scala> for (i <- 1 to 5; j <- 10 to 30 by 10) println(i+j)

scala> for (i <- 1 to 3; j <- 1 to 3; if i==j; x = 10\*i+j) println(x)

scala> str

scala> str.head

scala> str.tail

scala> str.tail.head

scala> str(0)

scala> str(1)

scala> str.foreach(c => println(c))

scala> val PI = 3.14

scala> def area(radius: Int) = PI \* radius \* radius

scala> area(10)

scala> :quit

https://www.tutorialspoint.com/scala/

### REPL Mode

#### Demo 1 Word count(Scala)

Spark\_WordCount.txt

Logic for WordCount

- Read each record and convert into words

- Mark each word as key and assign a value 1 - <k, v> --> paired RDD

- Group by key

- For each key, sum the values within the list

#### Demo 2 MaxClosePrice

MaxClosePrice-SparkScala.txt

#### Demo 3 Word count(Python)

Spark\_WordCount.txt

# Day 3

## RDD Operations

- Transformations

- Actions

## RDD Dependency

- Narrow

- Wide

## DAG

## Batch Mode

### Demo 1 Batch Mode ( python)

wc.py

[cloudera@quickstart ~]$ spark-submit file:///home/cloudera/Desktop/Labs/wc.py

### Demo 2 OrderRDD ( python)

OrdersRDD\_PySpark.txt

### Demo 3 log analysis (Scala)

sample.log

spark-shell

scala> val logRDD = sc.textFile("file:///home/cloudera/Desktop/Labs/sample.log")

logRDD: org.apache.spark.rdd.RDD[String] = file:///home/cloudera/Desktop/Labs/sample.log MapPartitionsRDD[1] at textFile at <console>:27

scala> val logLevelCountRDD = logRDD.map(m => m.split(" ")(3).replace("[","").replace("]","")).map(n => (n,1)).reduceByKey(\_+\_)

logLevelCountRDD: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <console>:29

scala> logLevelCountRDD.collect

res0: Array[(String, Int)] = Array((ERROR,3), (DEBUG,434), (INFO,96), (WARN,4), (TRACE,816), (FATAL,1))

## RDD Operation

SparkRDDOperations.pdf

## Zepl to run pspark code online

https://www.zepl.com/

## SPARKSQL

### Demo 1 with python

Spark\_DF.txt

## FLUME

flume is data ingest tool. to get data from nontraditional data source

BigData\_Hadoop.pdf flume section

### Demo 1

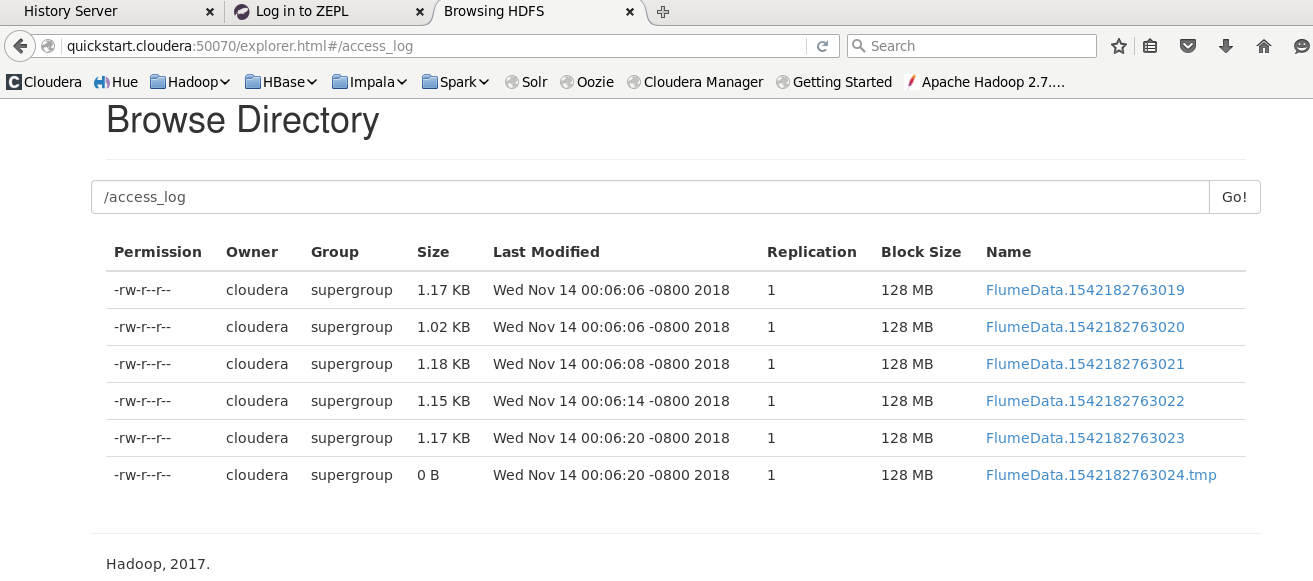
Terminal 1 – Run start log

[cloudera@quickstart ~]$ start\_logs

[cloudera@quickstart ~]$ stop\_logs // to stop log generation to stop file generation after 2-3 min we trigger flume command on terminal 2

Terminal 2 – Run Flume agent

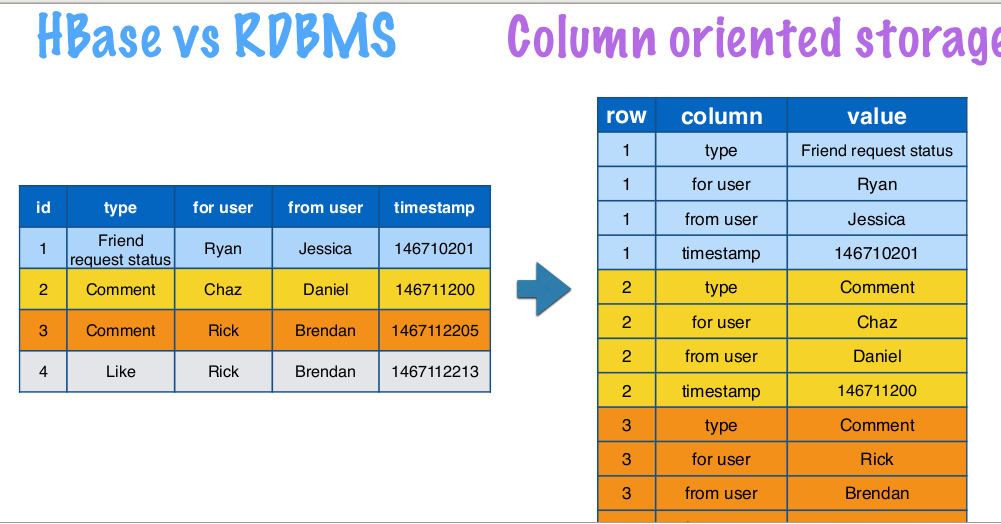
• flume-ng agent -n tail -c conf -f /home/cloudera/Desktop/Labs/tail.conf



## HBASE

Bulk loading is done through Java API

### HBASE vs RDMS



HBaseIntro.pdf

### HBASE Architecture

HBase-Architecture.pdf

### Hbase process

* HRegionServer
* QuorumPeerMain
* HMaster

### Demo

HBaseShell-Ex1to7.pdf

## POC

http://quickstart.cloudera/#/tutorial/ingest\_structured\_data

### Exercise 1

hive shell

drop table order\_items;

drop table orders;

drop table customers;

linux shell

[cloudera@quickstart ~]$ sqoop import-all-tables \

-m 1 \

--connect jdbc:mysql://quickstart:3306/retail\_db \

--username=retail\_dba \

--password=cloudera \

--compression-codec=snappy \

--as-parquetfile \

--warehouse-dir=/user/hive/warehouse \

--hive-import

### Exercise 2

## Big Picture

cloudera-for-iot (2).pdf

## where it is used by customers

https://www.cloudera.com/more/customers.html

can get more info on MapR horton work etc.

# Further study topics

* Avro
* Parquet
* Hadoop 3 features
* Spark whitepaper https://www.usenix.org/legacy/event/hotcloud10/tech/full\_papers/Zaharia.pdf
* Sede http://blog.cloudera.com/blog/2012/12/how-to-use-a-serde-in-apache-hive/