CentraleSupelec 2019-2020 MSC DSBA / DATA SCIENCES

Big Data Algorithms, Techniques and Platforms

Distributed Databases Hadoop Applications and Ecosystem.

Hugues Talbot & Céline Hudelot, professors.

Big Data
Hadoop Stack

Lecture #1

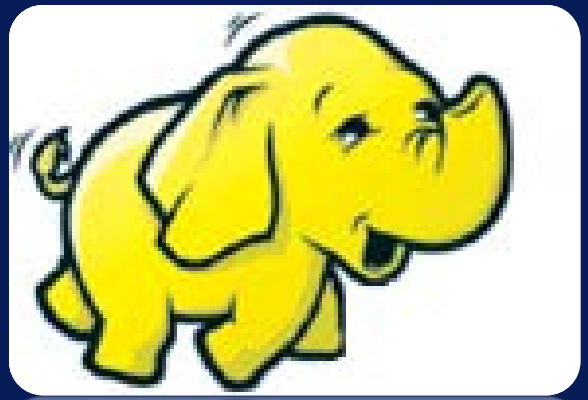
Hadoop Beginnings

What is Hadoop?

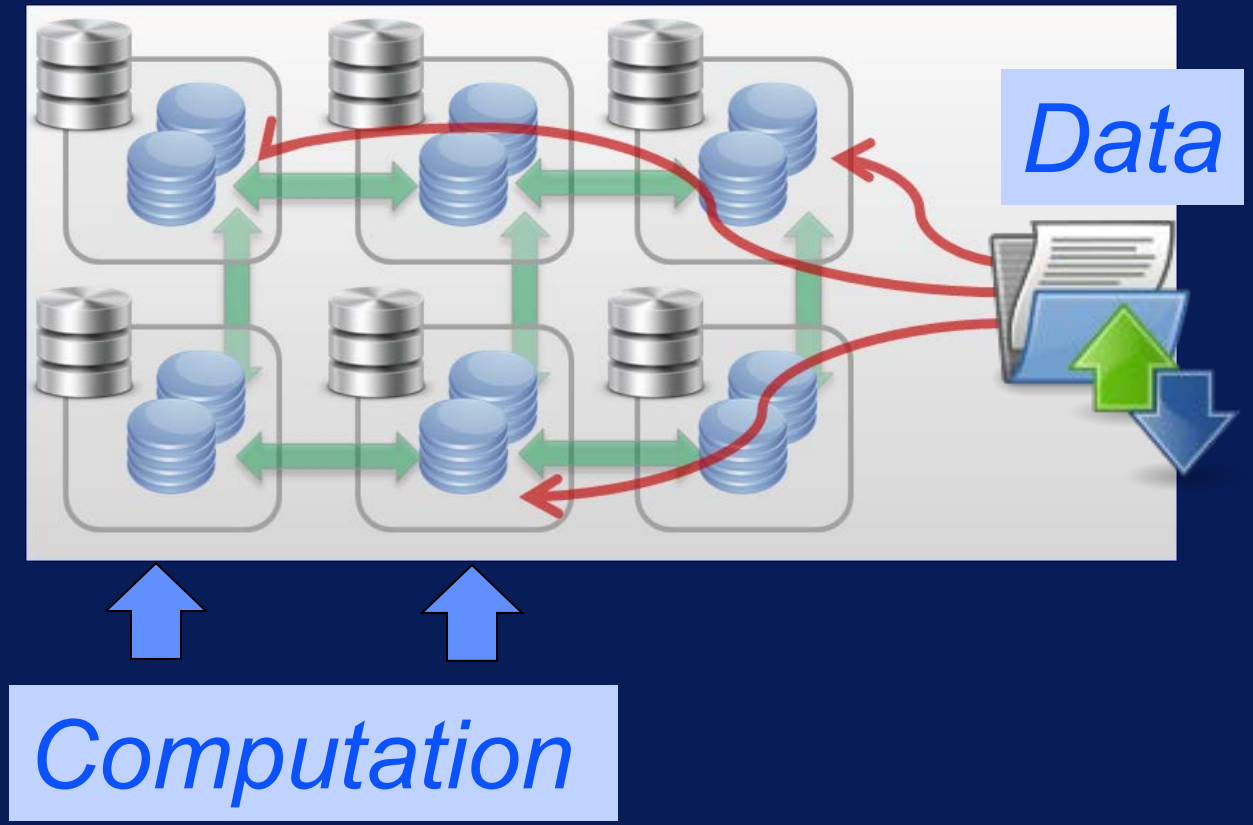
Apache Hadoop is an open source software framework for storage and large scale processing of data-sets on clusters of commodity hardware

Hadoop was created by Doug Cutting and Mike Cafarella in 2005

Named the project after son's toy elephant



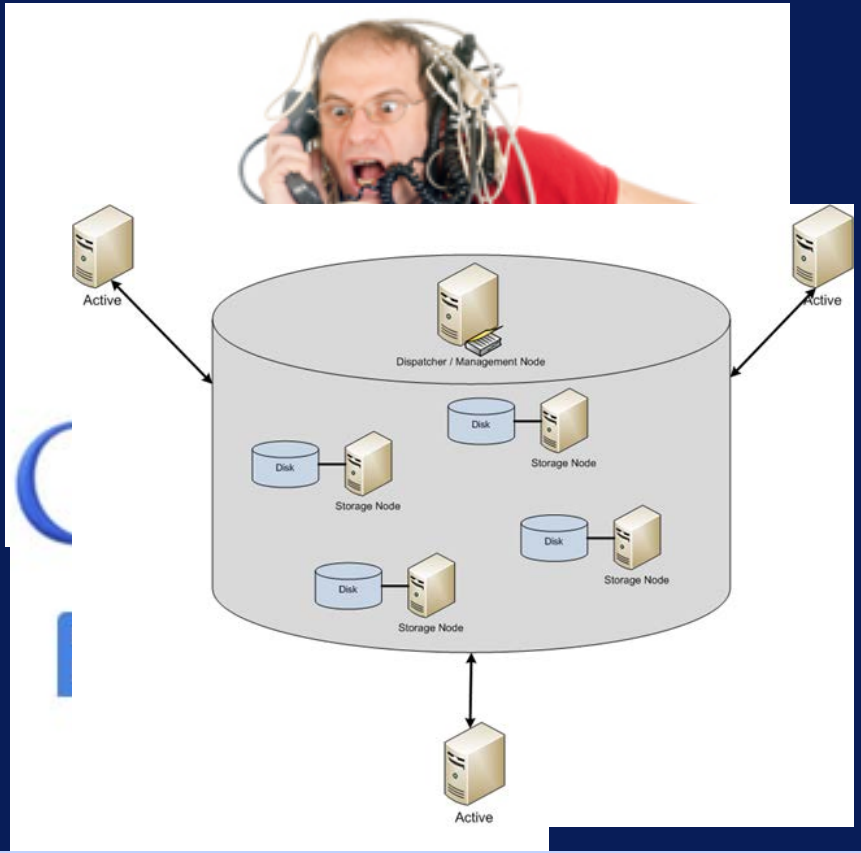
Moving Computation to Data



Scalability at Hadoop’s core!

Reliability!
Reliability!
Reliability!

Reliability!
Reliability!
Reliability!

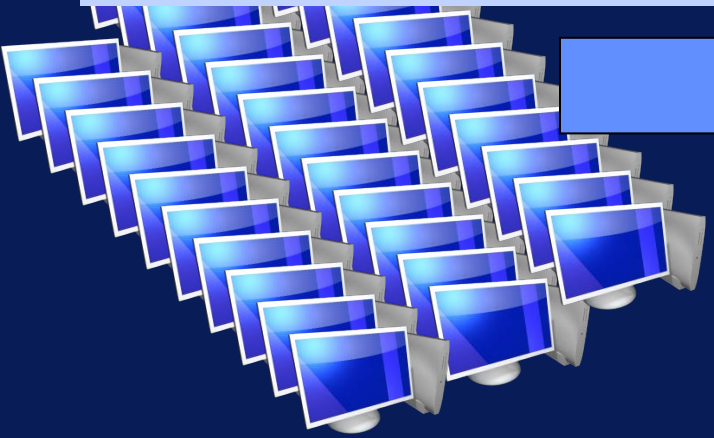


Reliability!
Reliability!
Reliability!

Google File System

Once a year

365 Computers



Once a day

Hourly

New Approach to Data

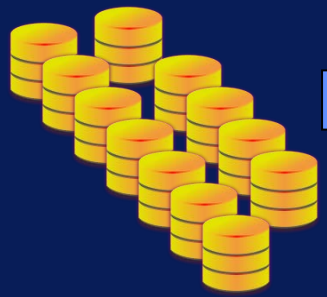
Keep all data

New Kinds of Analysis

Schema-on read style

New Kinds of Analysis

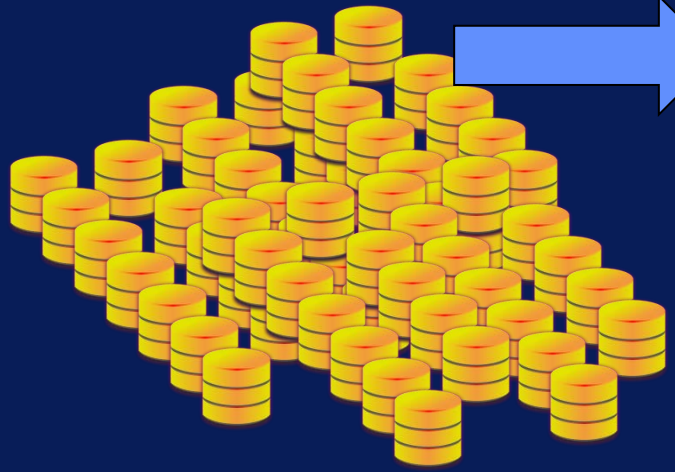
New analysis

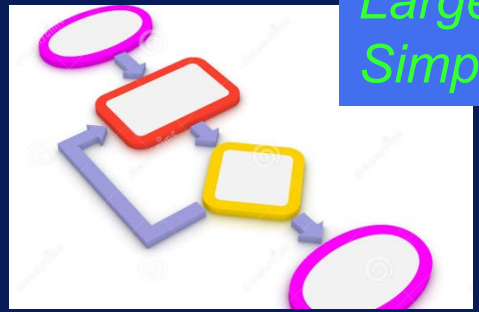


Small Data & Complex Algorithm

Vs.

Large Data & Simple Algorithm





Lecture #2

Apache Framework
Hadoop Modules

Apache Framework Basic Modules

- Hadoop Common
- Hadoop Distributed File System (HDFS)
- Hadoop YARN
- Hadoop MapReduce

Apache Framework Basic Modules

Hadoop Common

Hadoop Distributed File System (HDFS)

Hadoop YARN

Hadoop MapReduce

Apache Framework Basic Modules

Hadoop Common

Hadoop Distributed File System (HDFS)

Hadoop YARN

Hadoop MapReduce

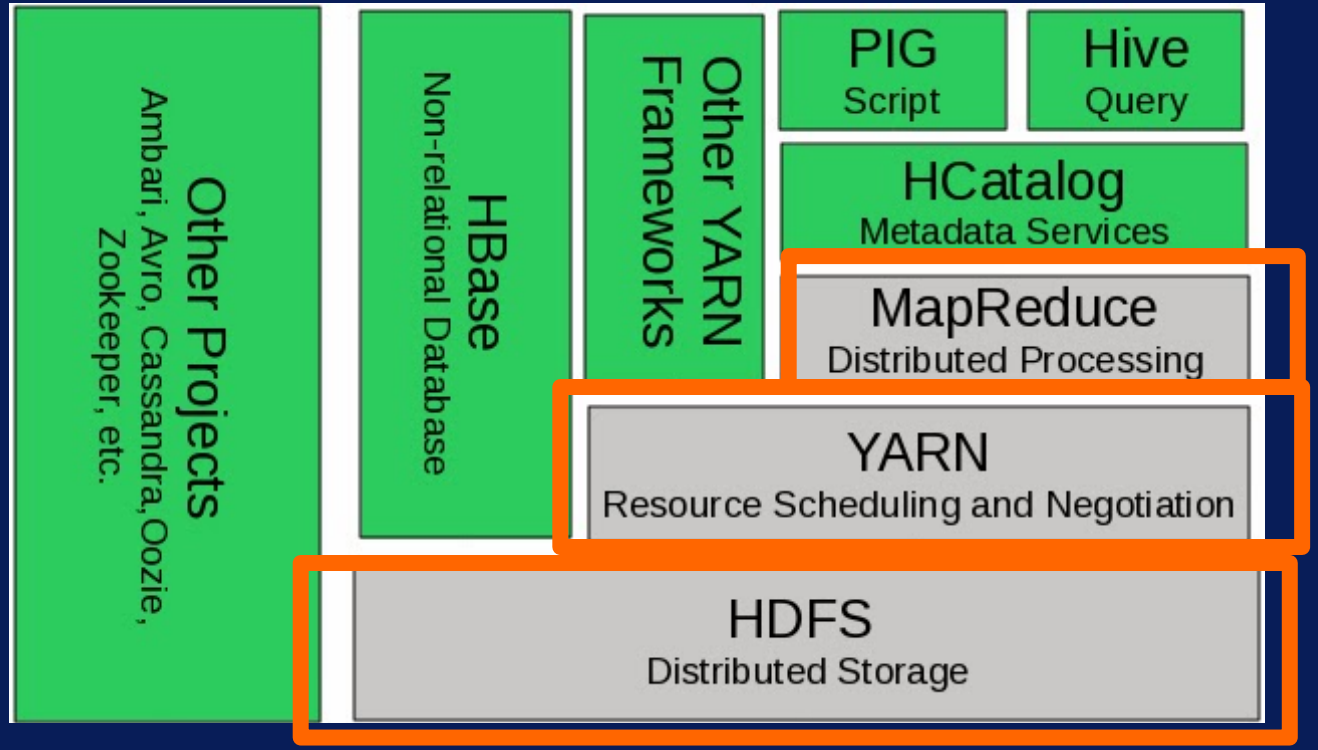
Apache Framework Basic Modules

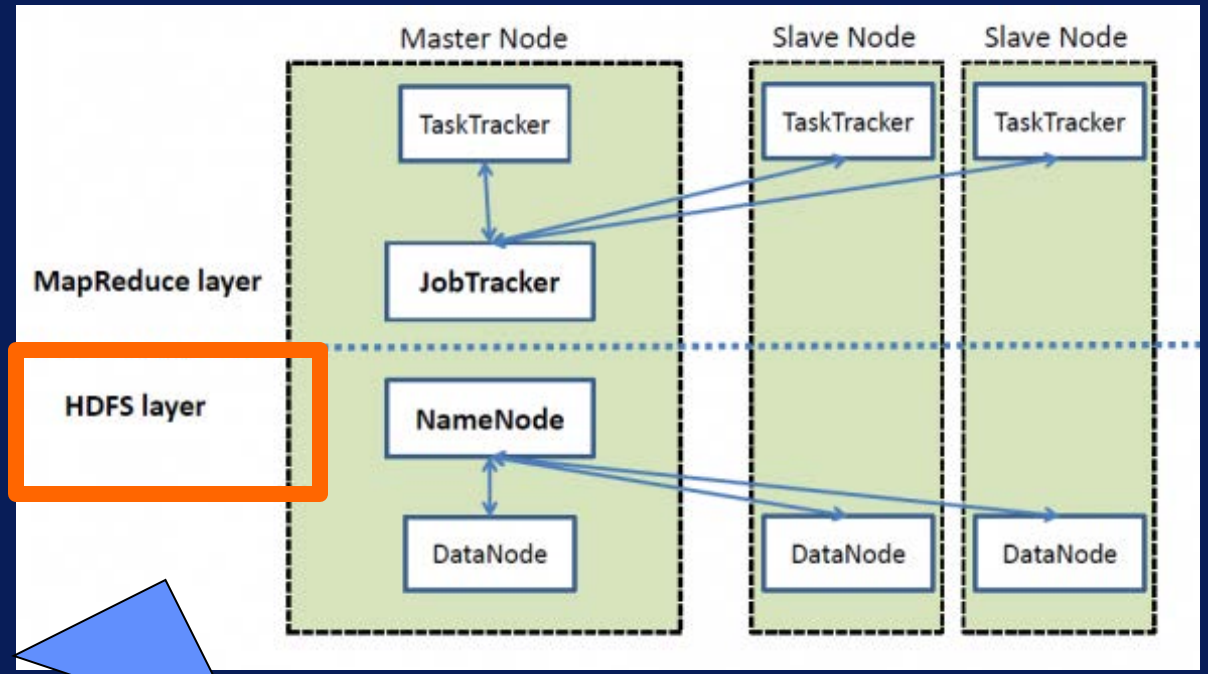
Hadoop Common

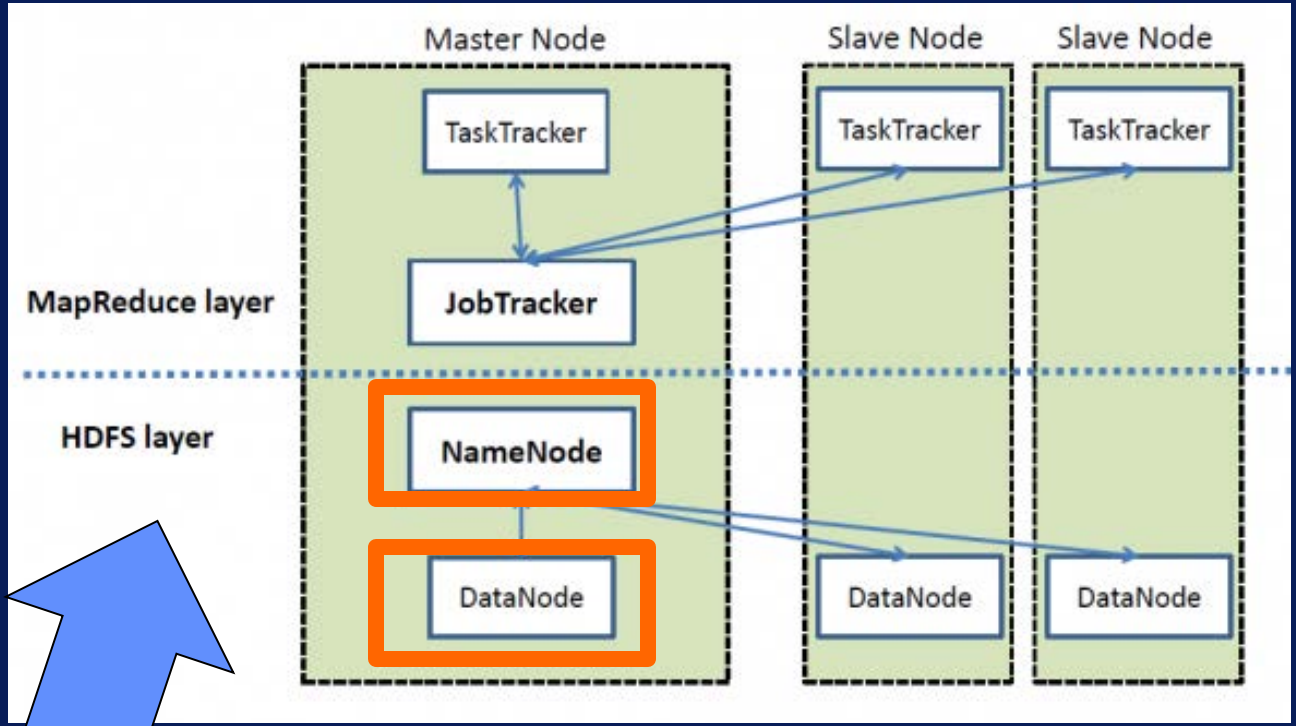
Hadoop Distributed File System (HDFS)

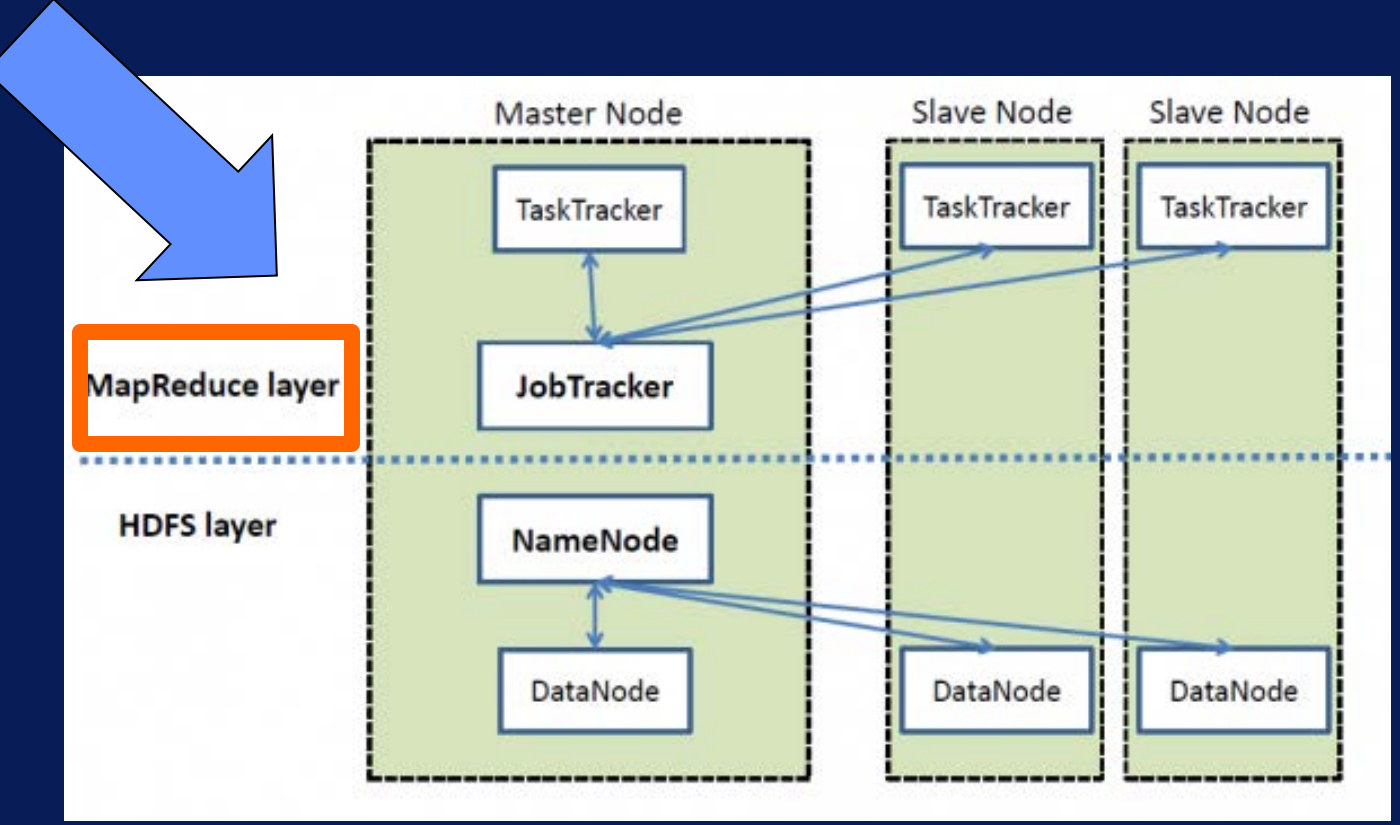
Hadoop YARN

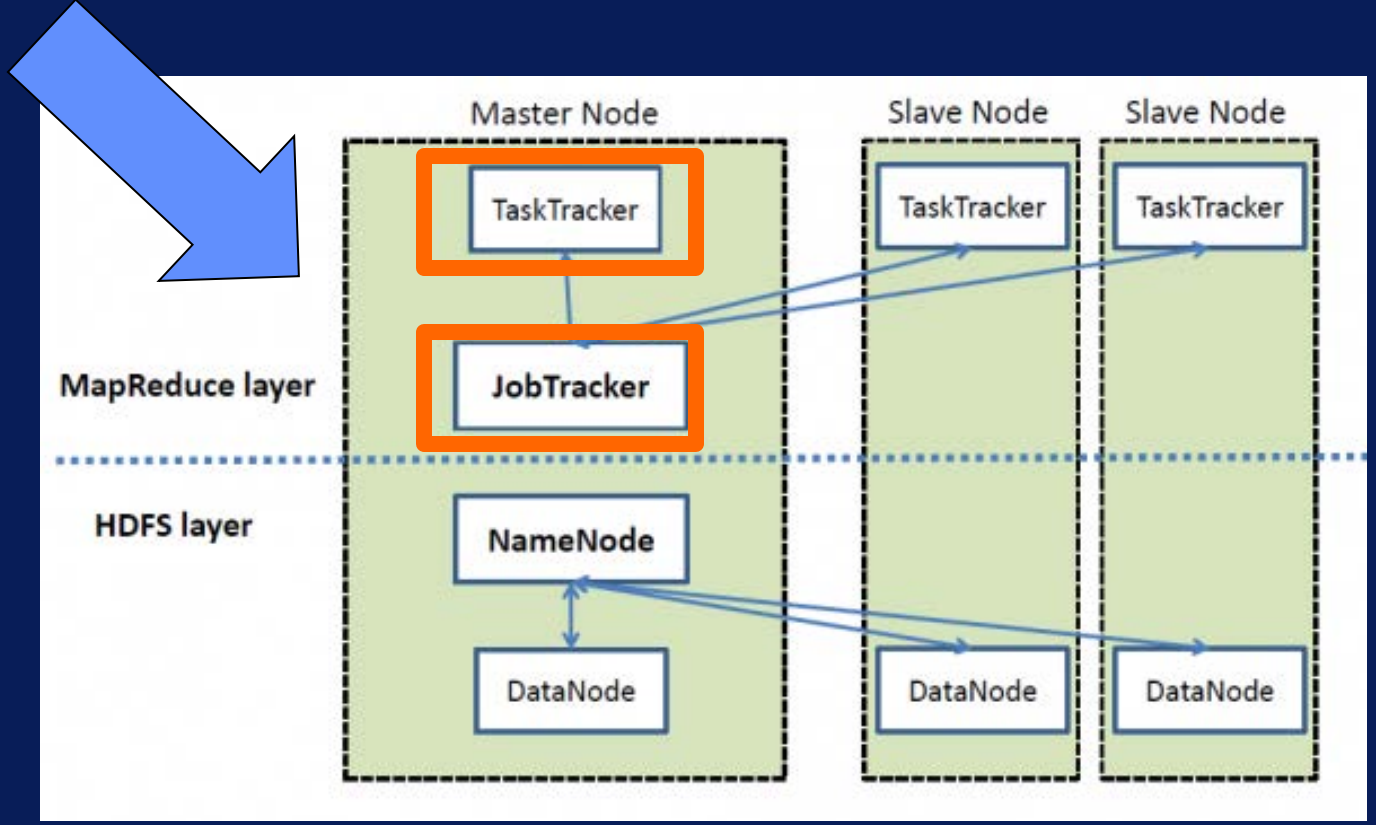
Hadoop MapReduce


Other Projects
Ambari, Avro, Cassandra, Oozie, Zookeeper, etc.
HBase
Non-relational Database
Other YARN Frameworks
PIG
Script
Hive
Query
HCatalog
Metadata Services
MapReduce
Distributed Processing
YARN
Resource Scheduling and Negotiation
HDFS
Distributed Storage


Master Node
Slave Node
Slave Node
TaskTracker
TaskTracker
TaskTracker
MapReduce layer
JobTracker
HDFS layer
NameNode
DataNode
DataNode
DataNode


Master Node
Slave Node
Slave Node
TaskTracker
TaskTracker
TaskTracker
MapReduce layer
JobTracker
HDFS layer
NameNode
DataNode
DataNode
DataNode


Master Node
Slave Node
Slave Node
TaskTracker
TaskTracker
TaskTracker
MapReduce layer
JobTracker
HDFS layer
NameNode
DataNode
DataNode
DataNode


Master Node
Slave Node
Slave Node
TaskTracker
TaskTracker
TaskTracker
MapReduce layer
JobTracker
HDFS layer
NameNode
DataNode
DataNode
DataNode

Lecture #3

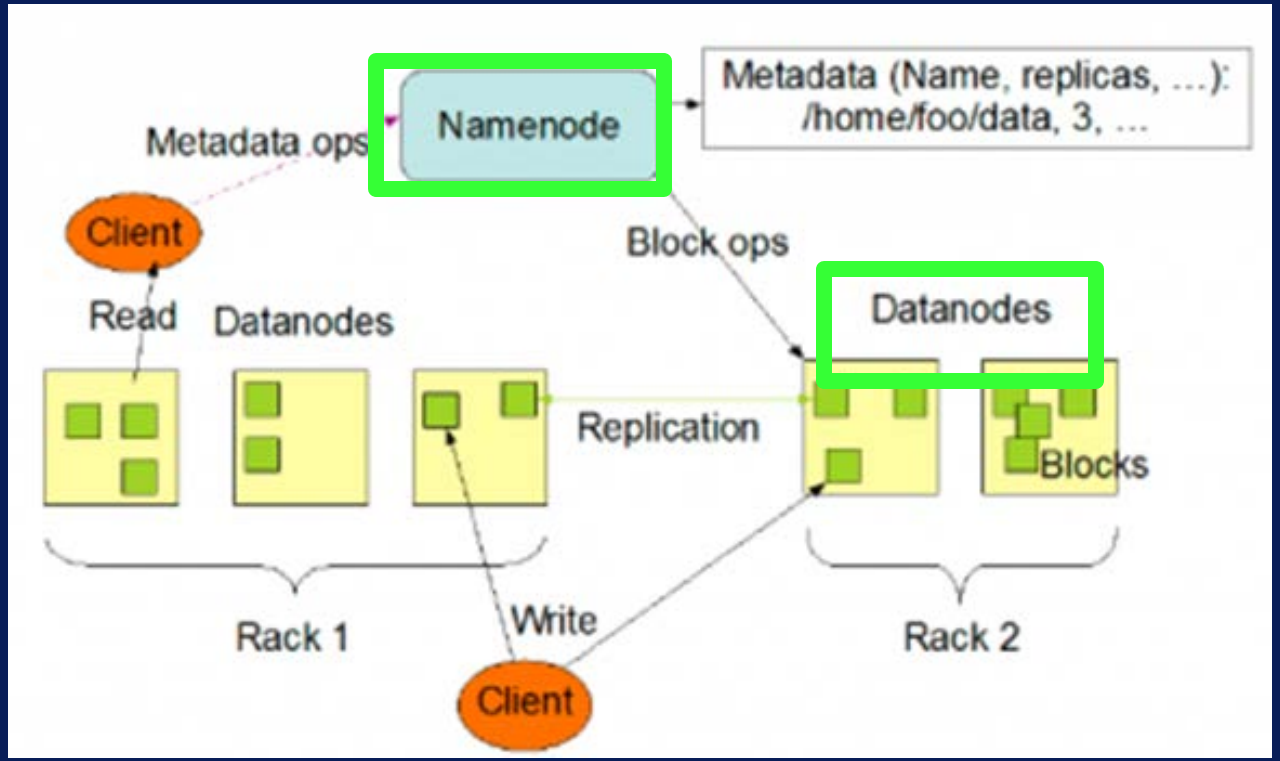
Hadoop Distributed File System (HDFS)

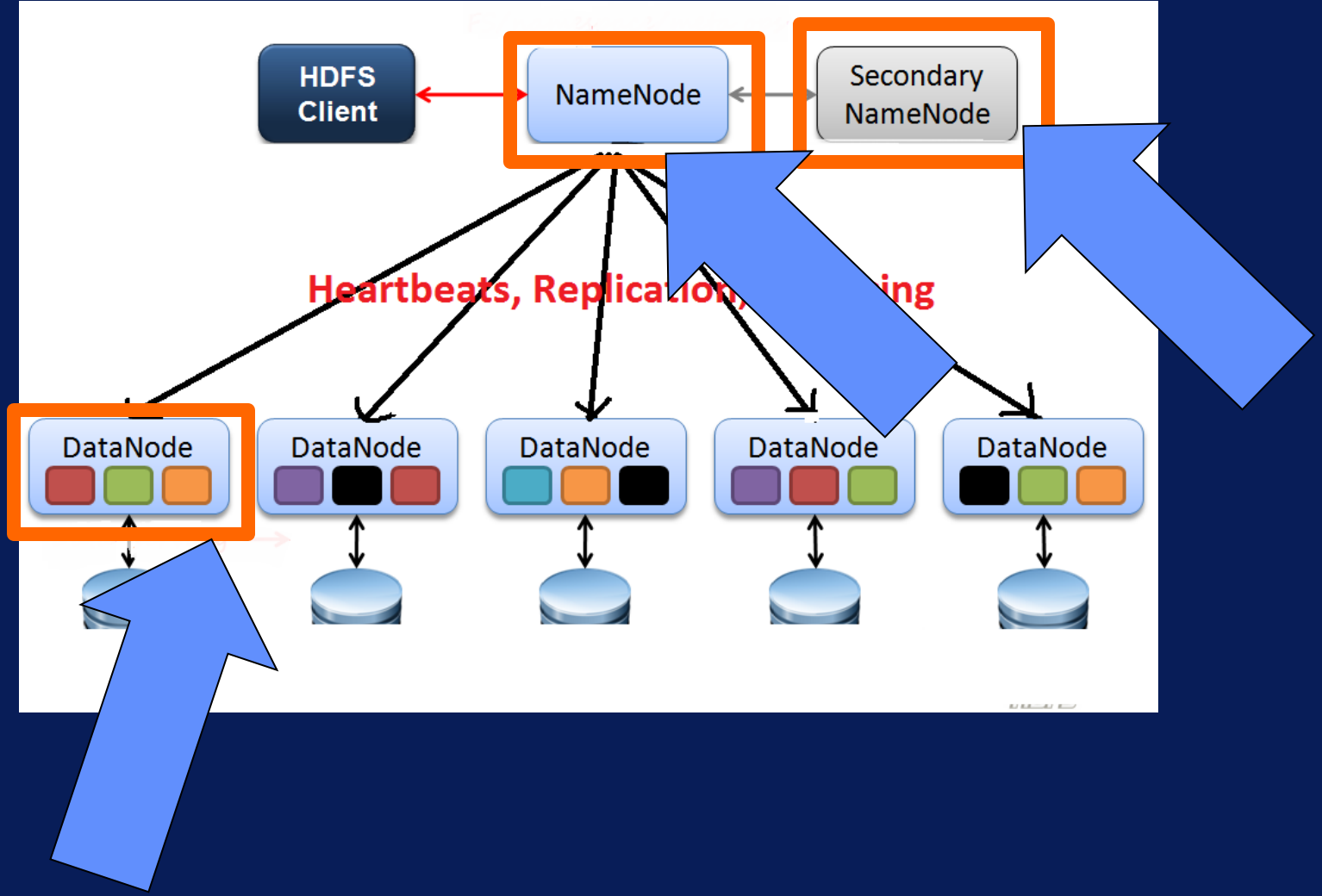
HDFS

Hadoop Distributed File System

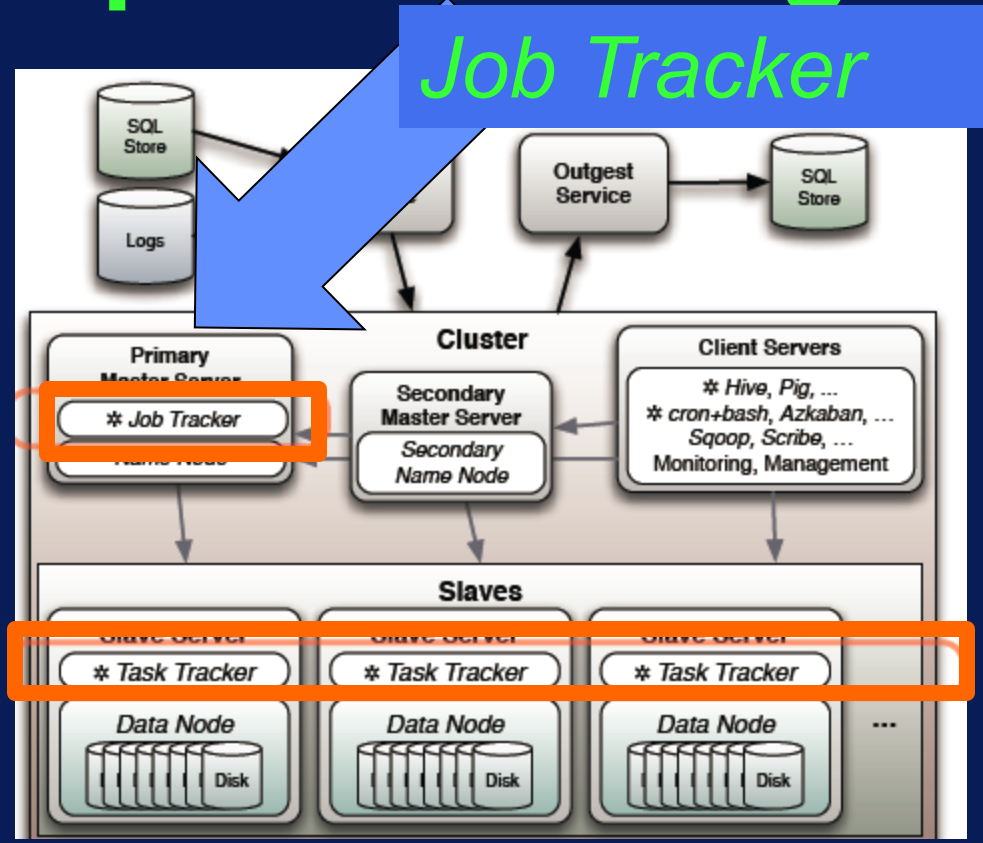
Distributed, scalable, and portable file-system written in Java for the Hadoop framework

HDFS

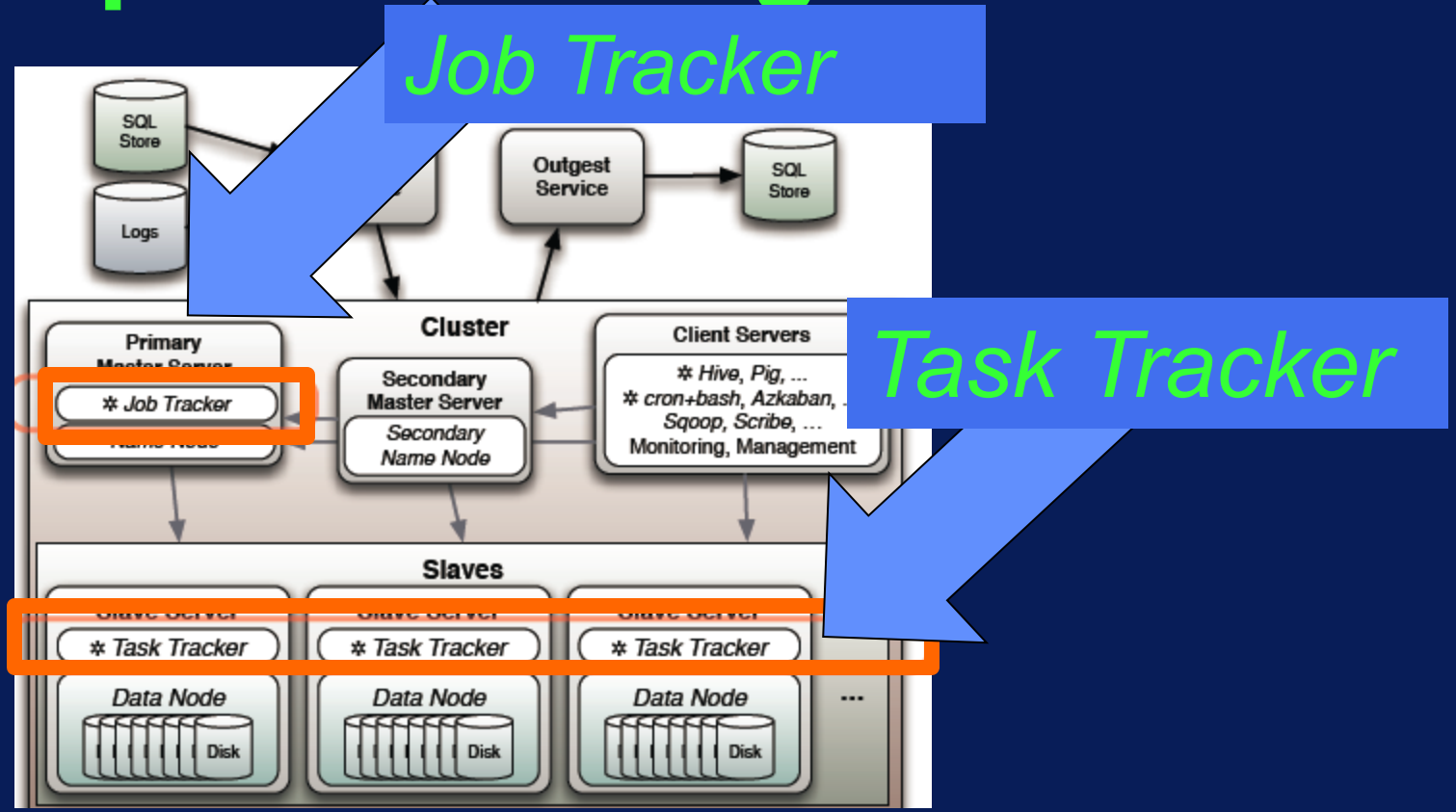

Metadata ops
Namenode
Metadata (Name, replicas, ...):
/home/foo/data, 3, ...
Client
Block ops
Read
Datanodes
Datanodes
Replication
Blocks
Rack 1
Write
Rack 2
Client


HDFS Client
NameNode
Secondary NameNode
Heartbeats, Replication,
ing
DataNode
DataNode
DataNode
DataNode
DataNode

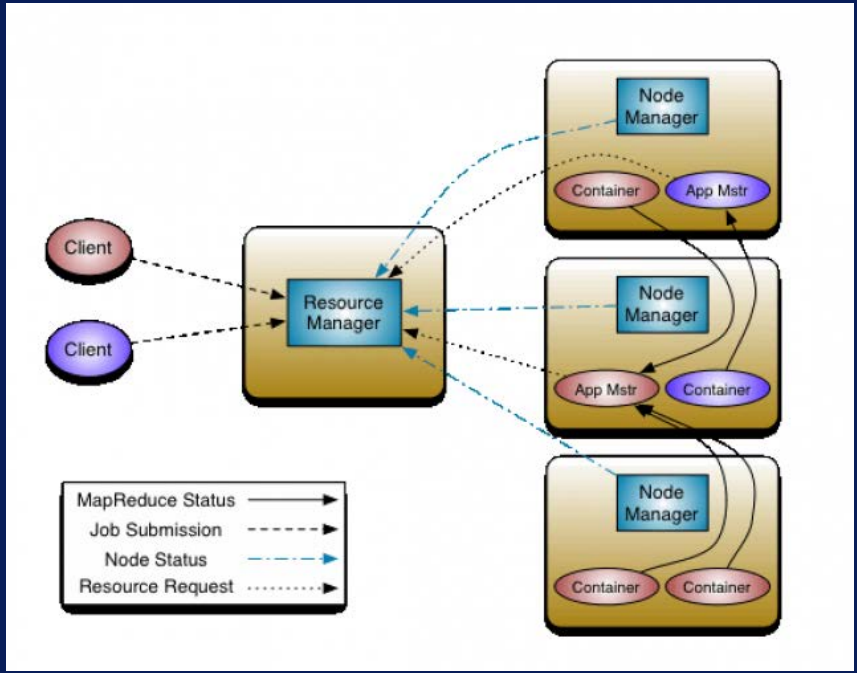
MapReduce Engine

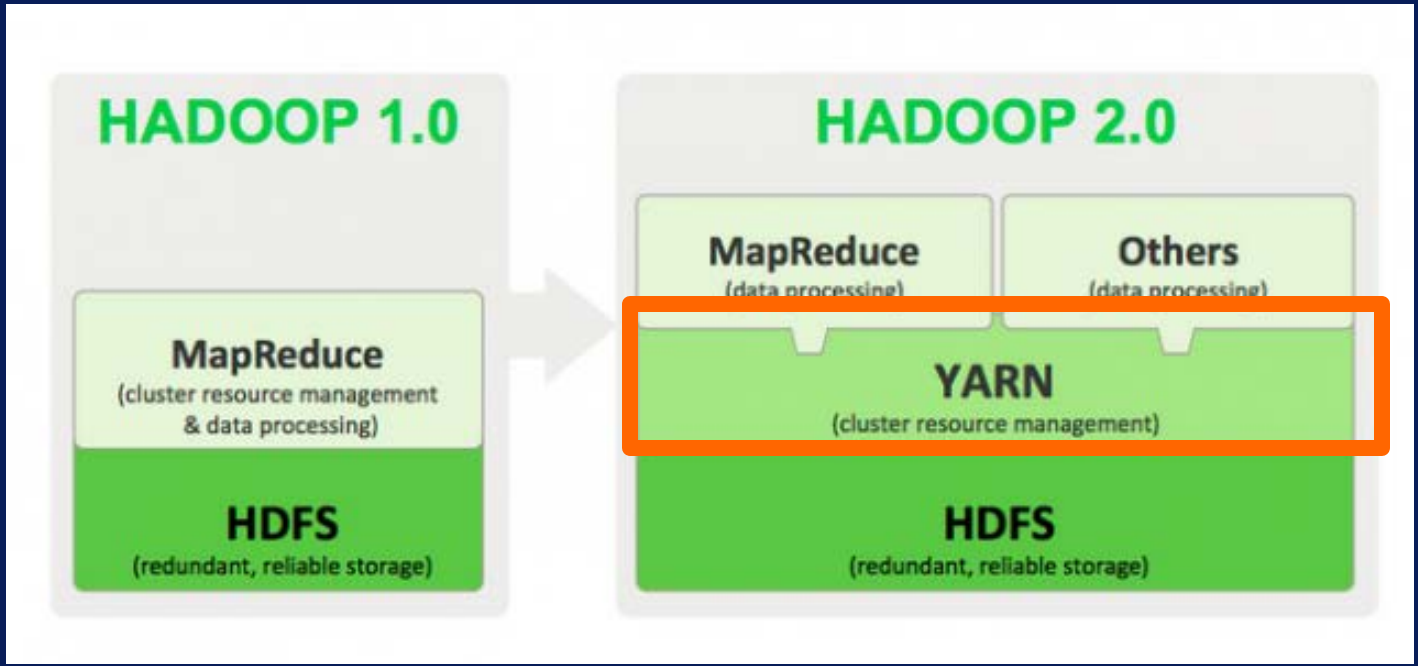


MapReduce Engine



Apache Hadoop NextGen MapReduce (YARN)




HADOOP 1.0
MapReduce
(cluster resource management
& data processing)
HDFS
(redundant, reliable storage)
HADOOP 2.0
MapReduce
(data processing)
Others
(data processing)
YARN
(cluster resource management)
HDFS
(redundant, reliable storage)

What is Yarn?

- **YARN enhances the power of a Hadoop compute cluster**

Scalability

What is Yarn?

- **YARN enhances the power of a Hadoop compute cluster**

Scalability

Improved cluster utilization

What is Yarn?

- **YARN enhances the power of a Hadoop compute cluster**

Scalability

Improved cluster utilization

MapReduce Compatibility

What is Yarn?

- **YARN enhances the power of a Hadoop compute cluster**

Scala

Map

Improved cluster utilization

Supports Other Workloads

Lecture #4

The Hadoop “Zoo”

Apache Hadoop Ecosystem

Ambari
Provisioning, Managing and Monitoring Hadoop Clusters

Sqoop Data Exchange

Flume Log Collector

Zookeeper Coordination

Oozie Workflow

Pig Scripting

Mahout Machine Learning

R Connectors Statistics

Hive SQL Query

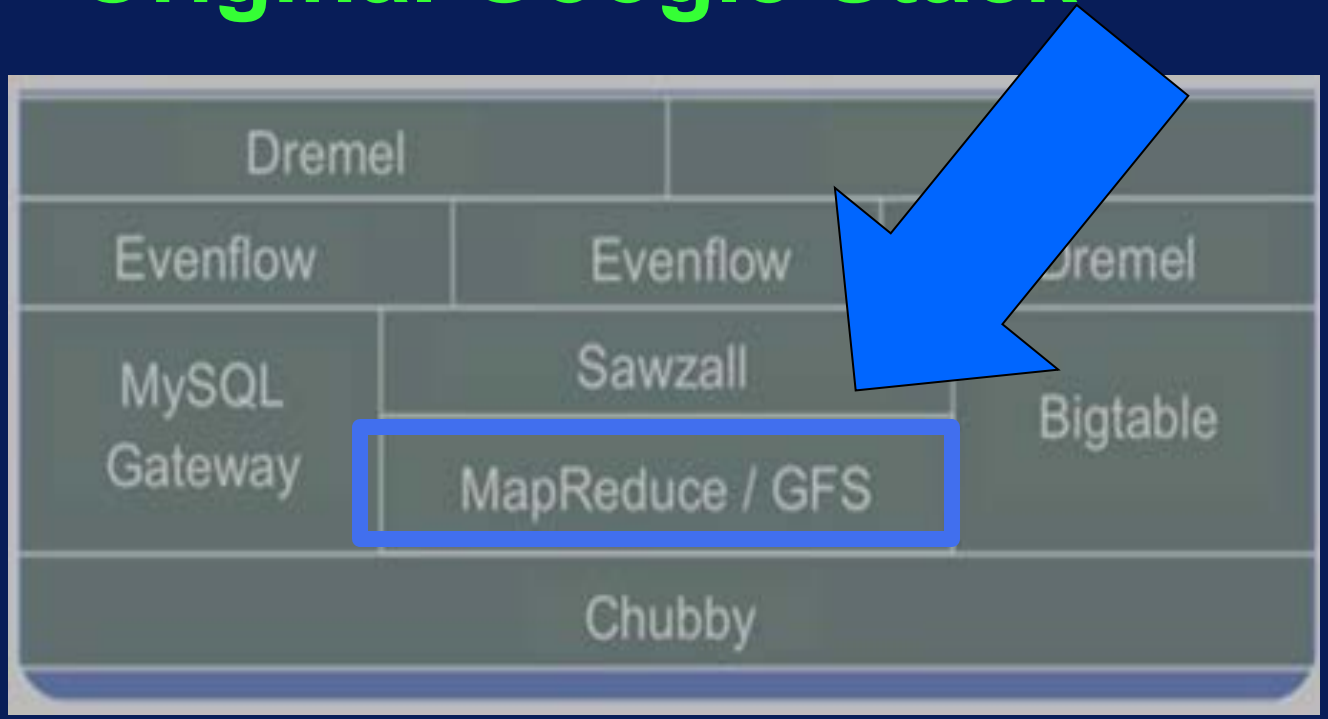
Hbase Columnar Store

YARN Map Reduce v2
Distributed Processing Framework

HDFS
Hadoop Distributed File System

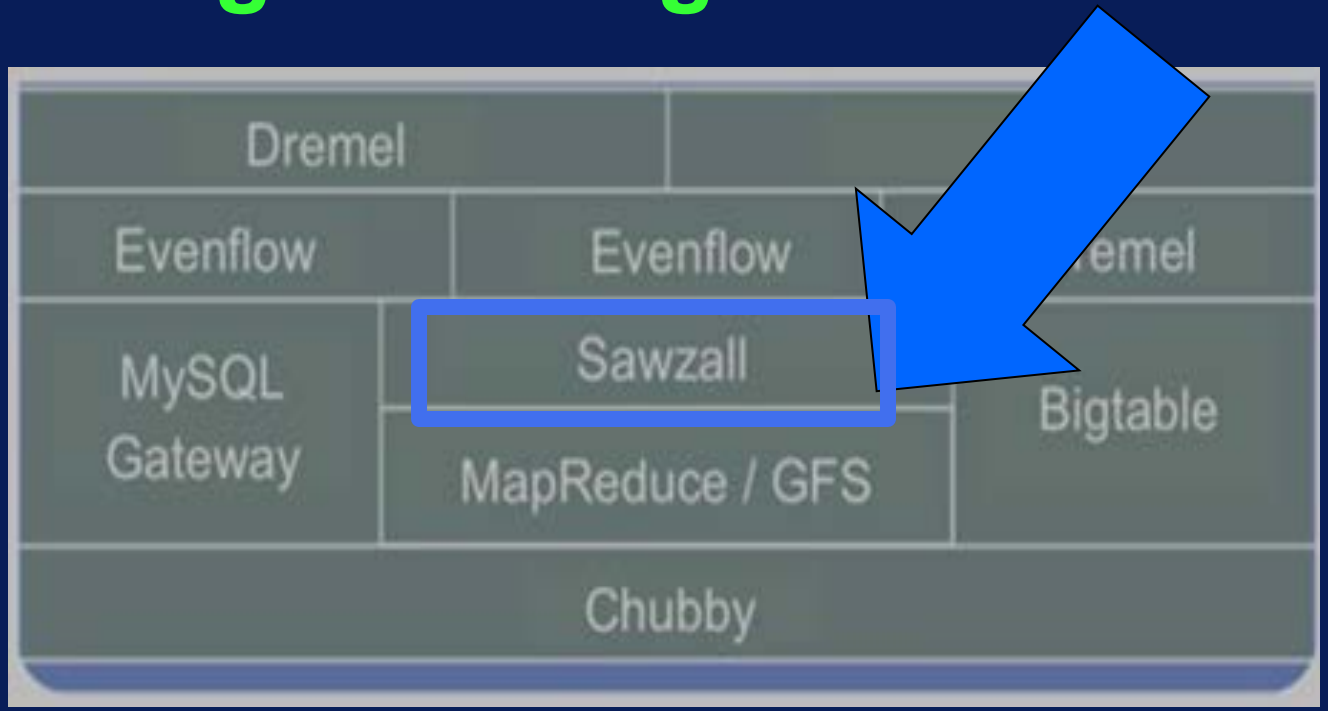
How to figure out the Zoo??

Original Google Stack



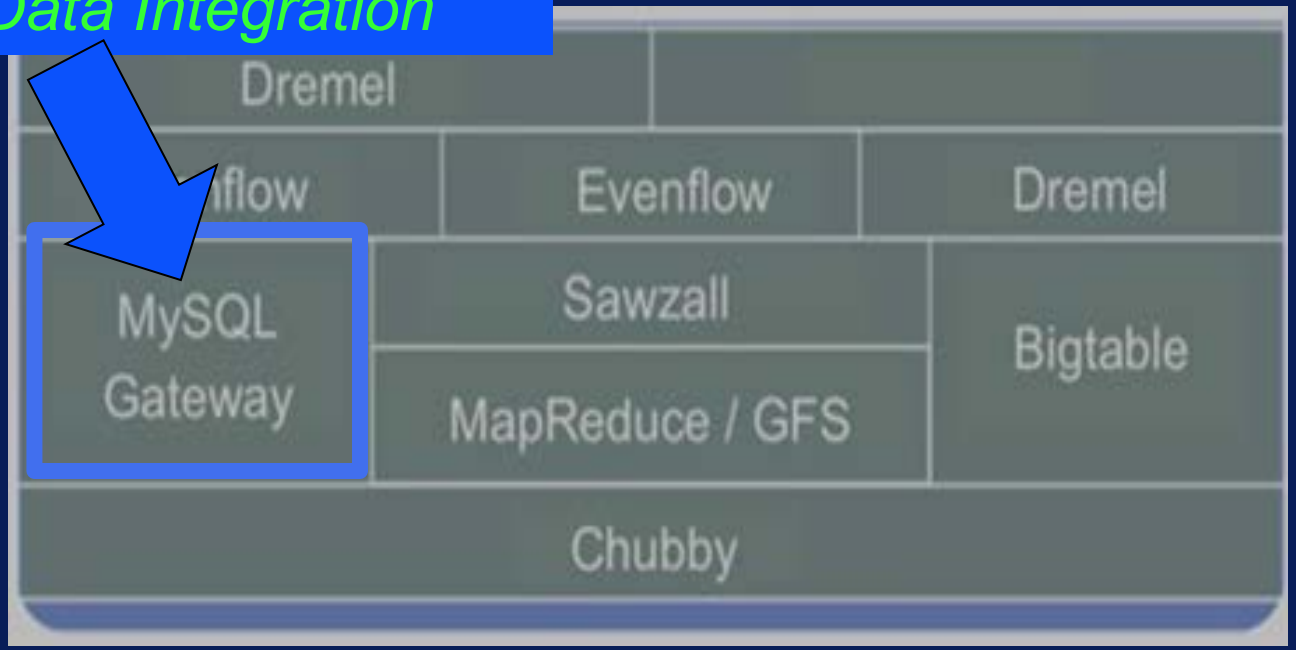
Original Google Stack

Original Google Stack

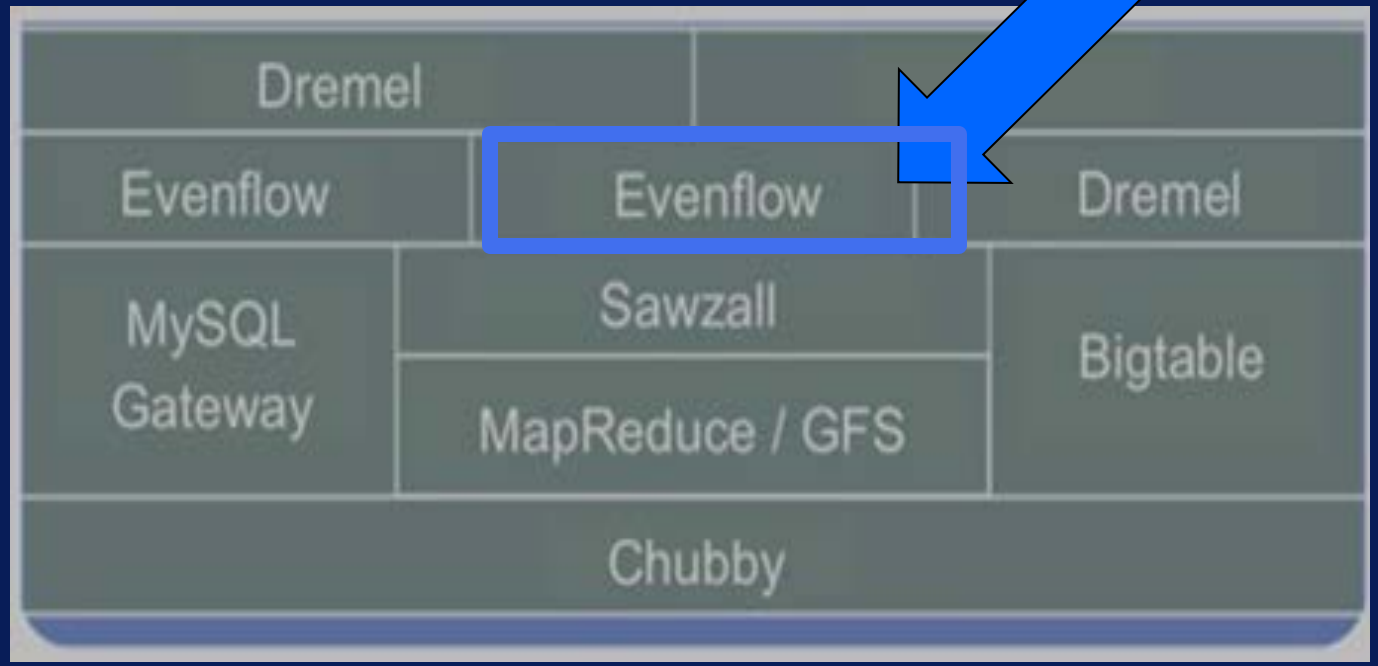


Original Google Stack

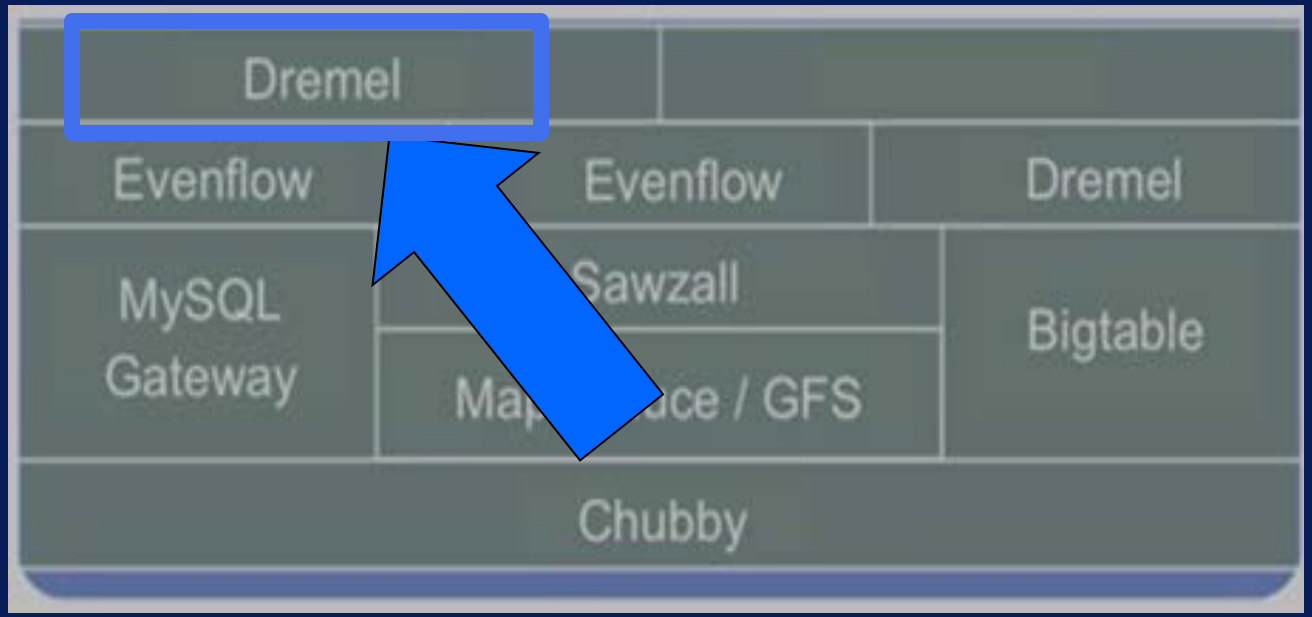
Data Integration



Original Google Stack

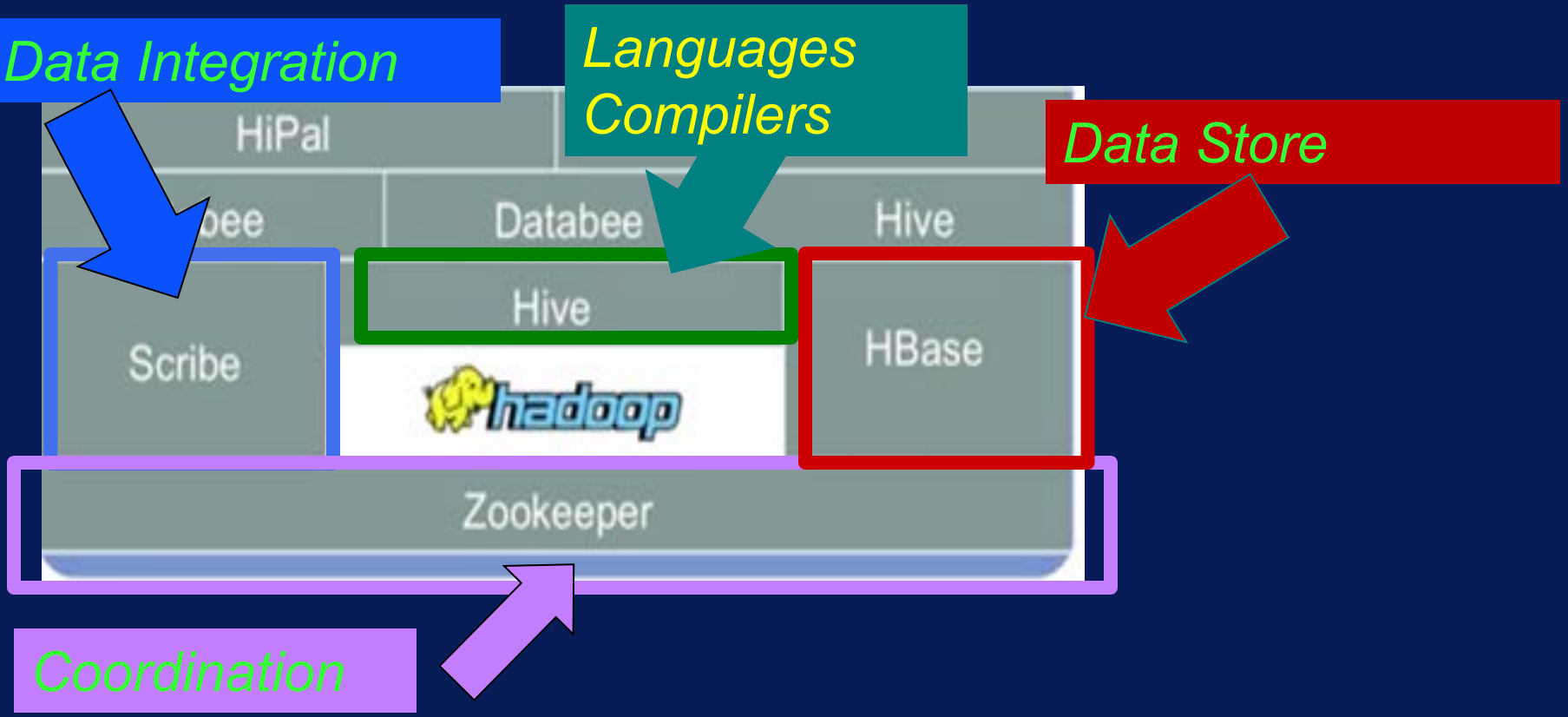


Original Google Stack

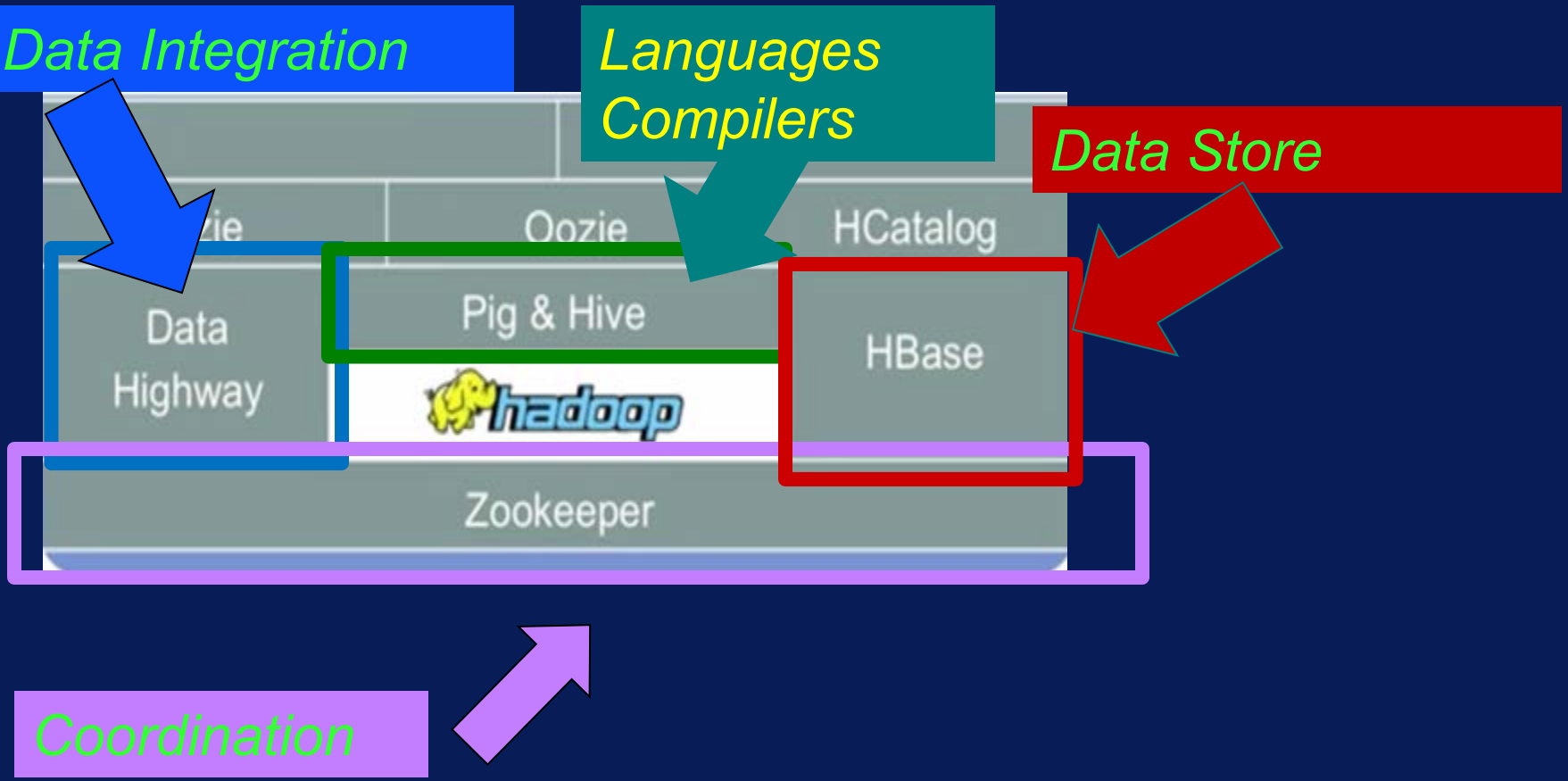


Original Google Stack

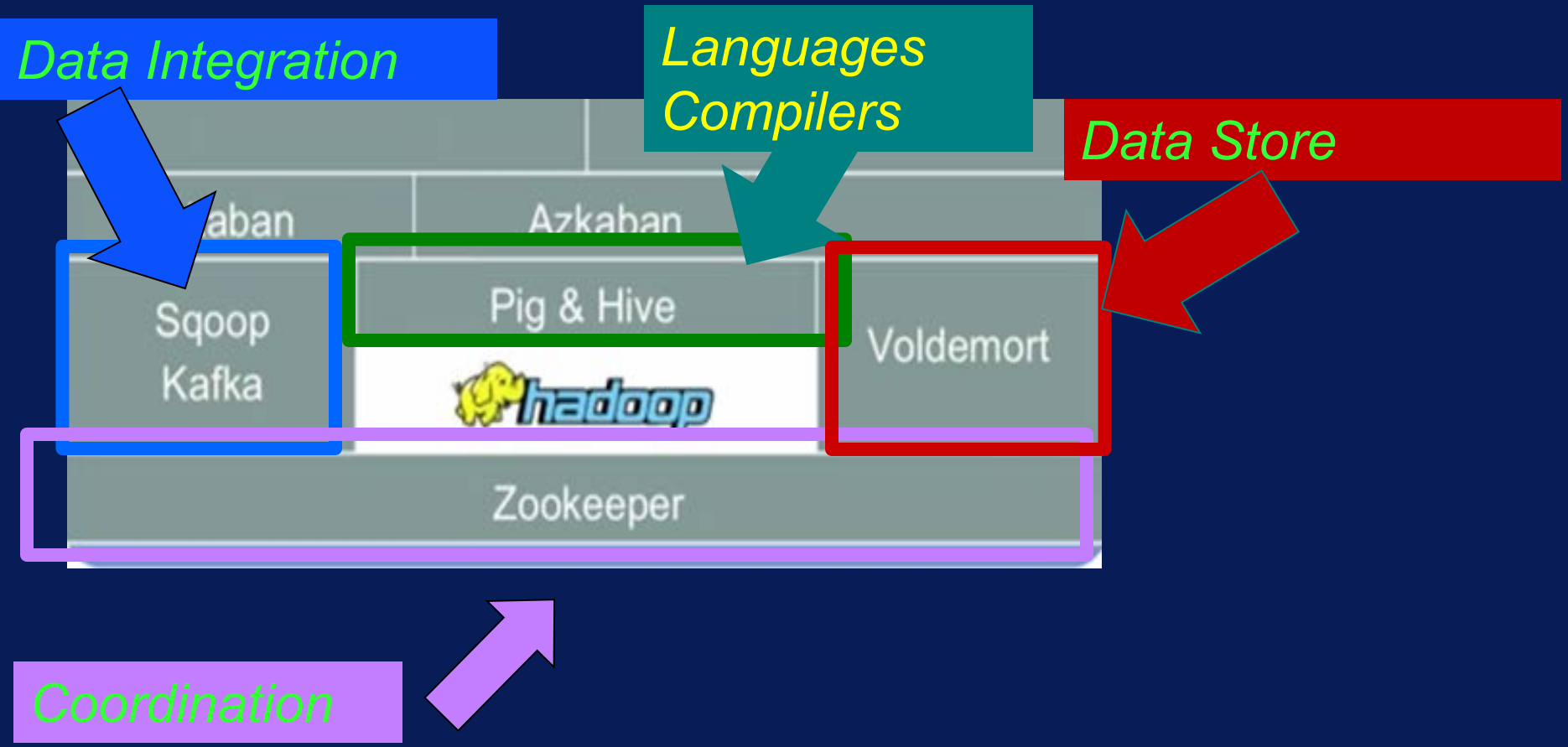
Facebook's Version of the Stack



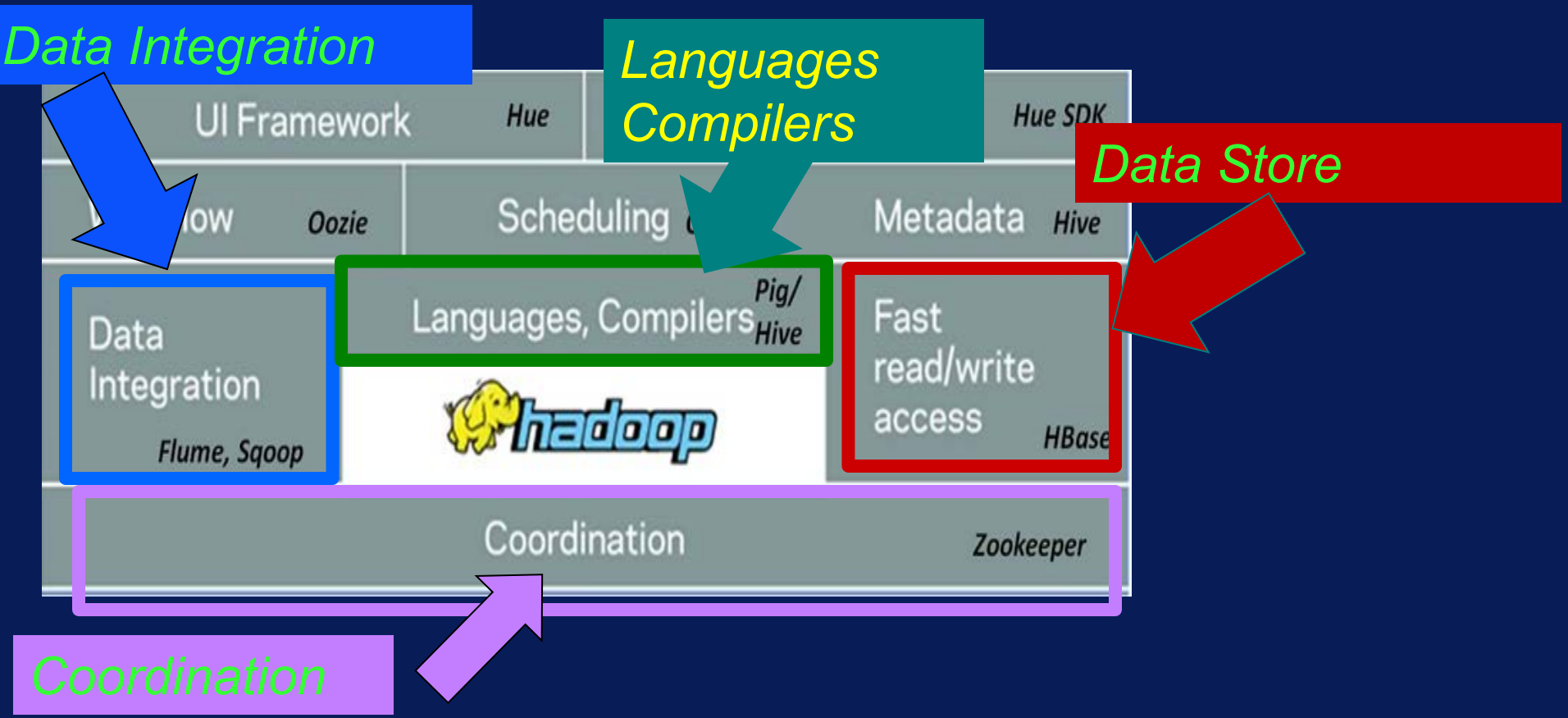
Yahoo's Version of the Stack



LinkedIn's Version of the Stack

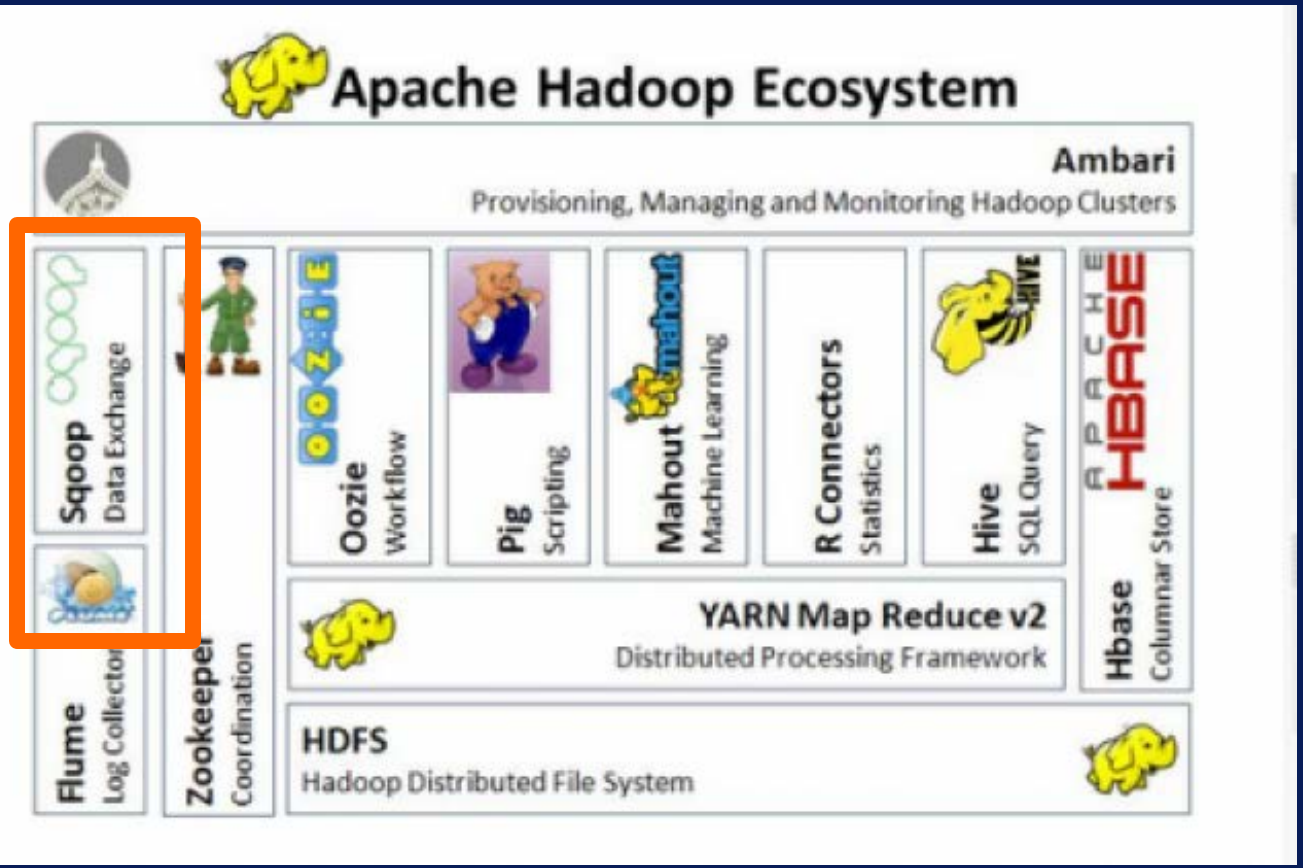


Cloudera's Version of the Stack



Lecture #5

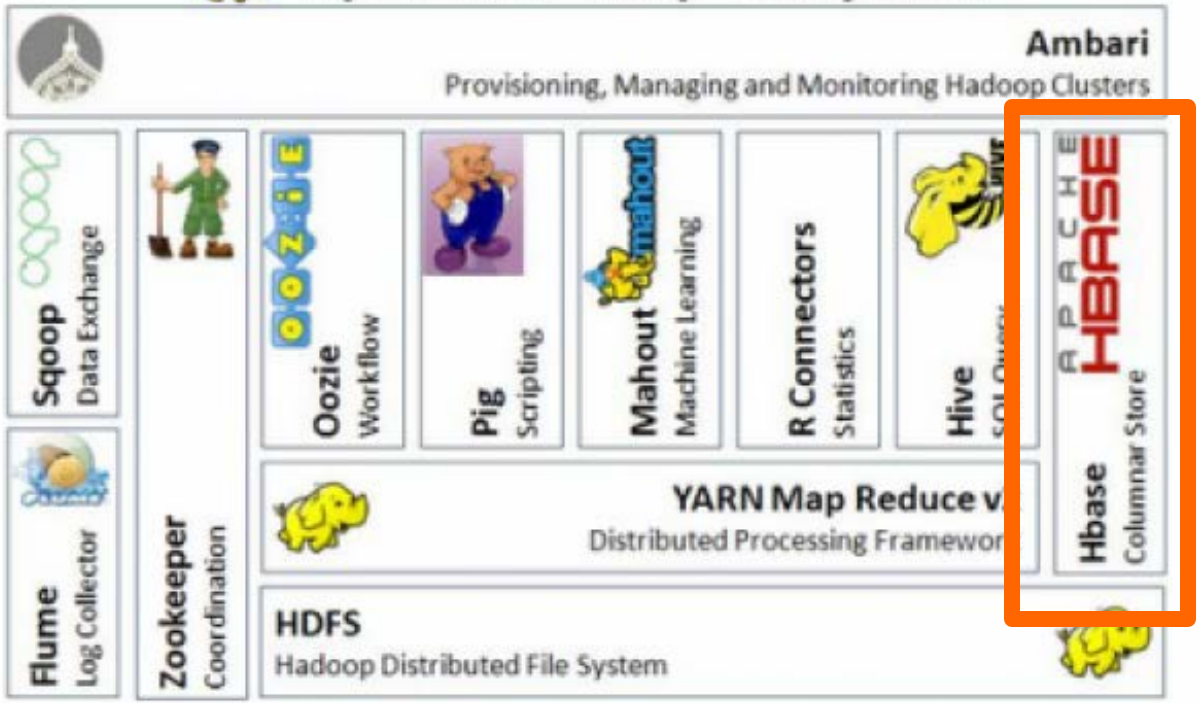
Hadoop Ecosystem Major Components


Apache Hadoop Ecosystem
Ambari
Provisioning, Managing and Monitoring Hadoop Clusters
Sqoop
Data Exchange
Flume
Log Collector
Zookeeper
Coordination
Oozie
Workflow
Pig
Scripting
Mahout
Machine Learning
R Connectors
Statistics
Hive
SQL Query
APACHE
HBASE
Hbase
Columnar Store
YARN Map Reduce v2
Distributed Processing Framework
HDFS
Hadoop Distributed File System

Apache Sqoop

- Tool designed for efficiently transferring bulk data between Apache Hadoop and structured datastores such as relational databases

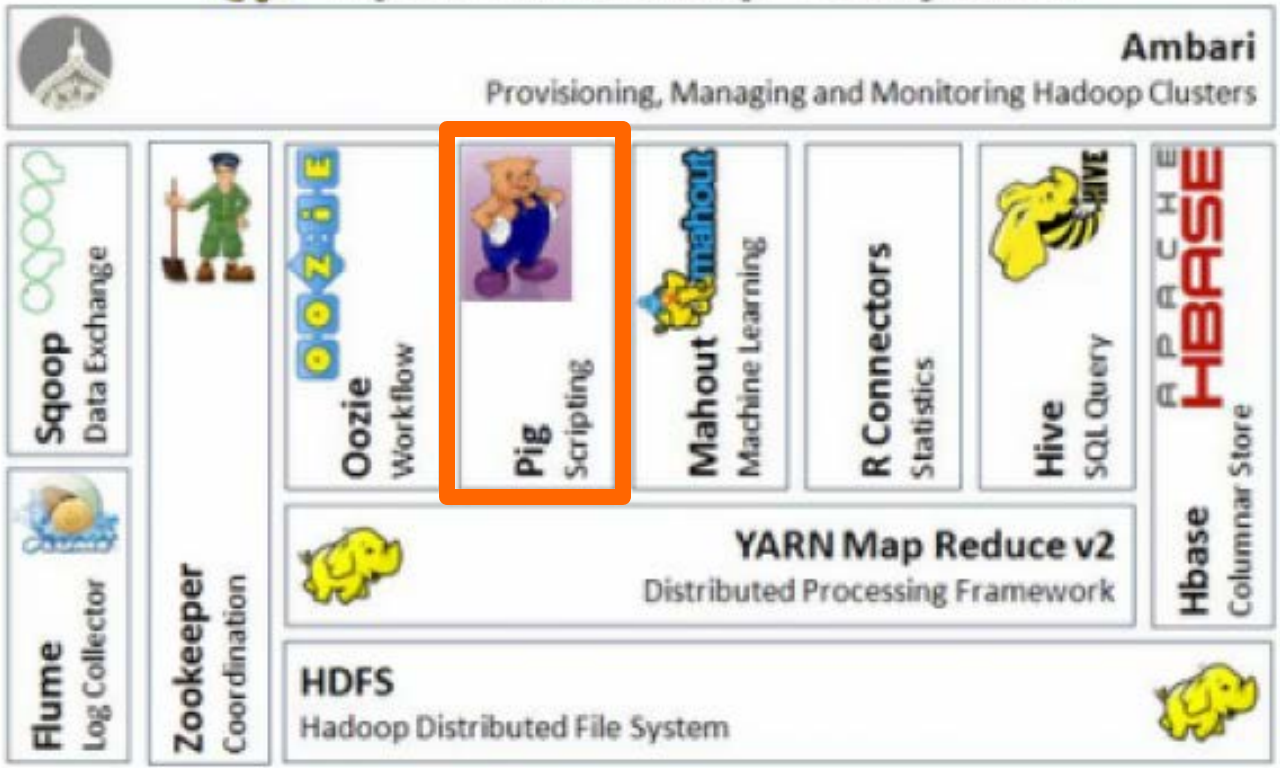
Apache Hadoop Ecosystem



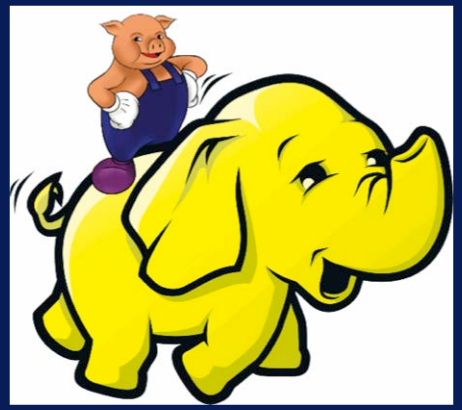
HBASE

- Column-oriented database management system
- Key-value store
- Based on Google Big Table
- Can hold extremely large data
- Dynamic data model
- Not a Relational DBMS

Apache Hadoop Ecosystem

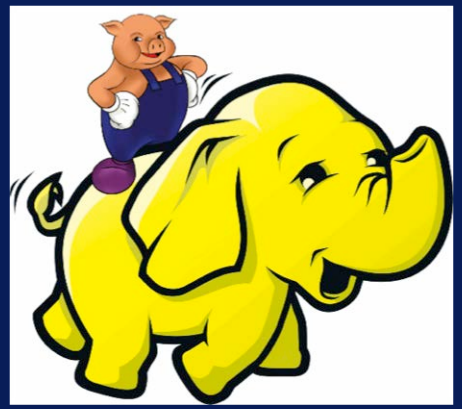


PIG



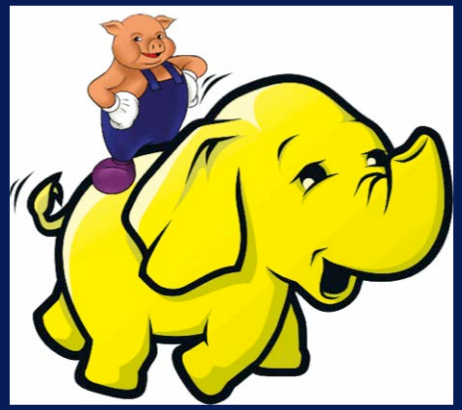
High level programming on top of Hadoop MapReduce

PIG



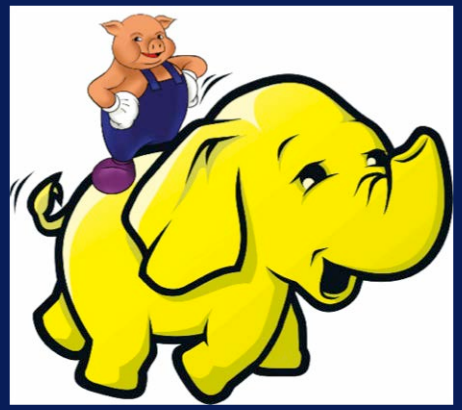
The language: Pig Latin

PIG



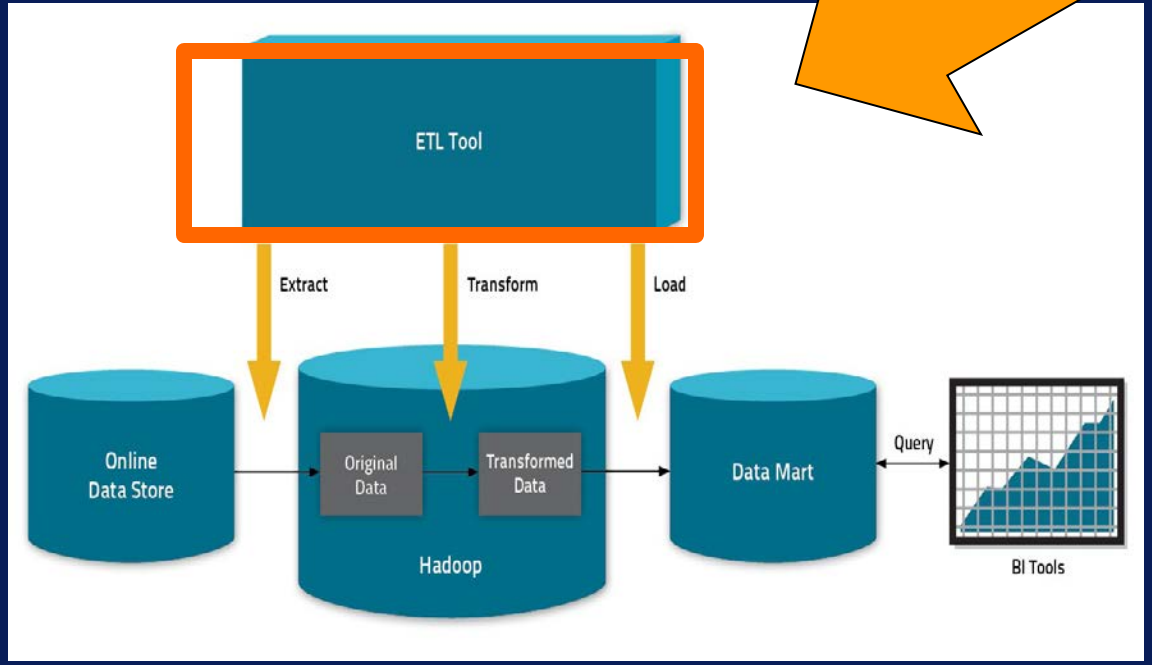
Data analysis problems as data flows

PIG

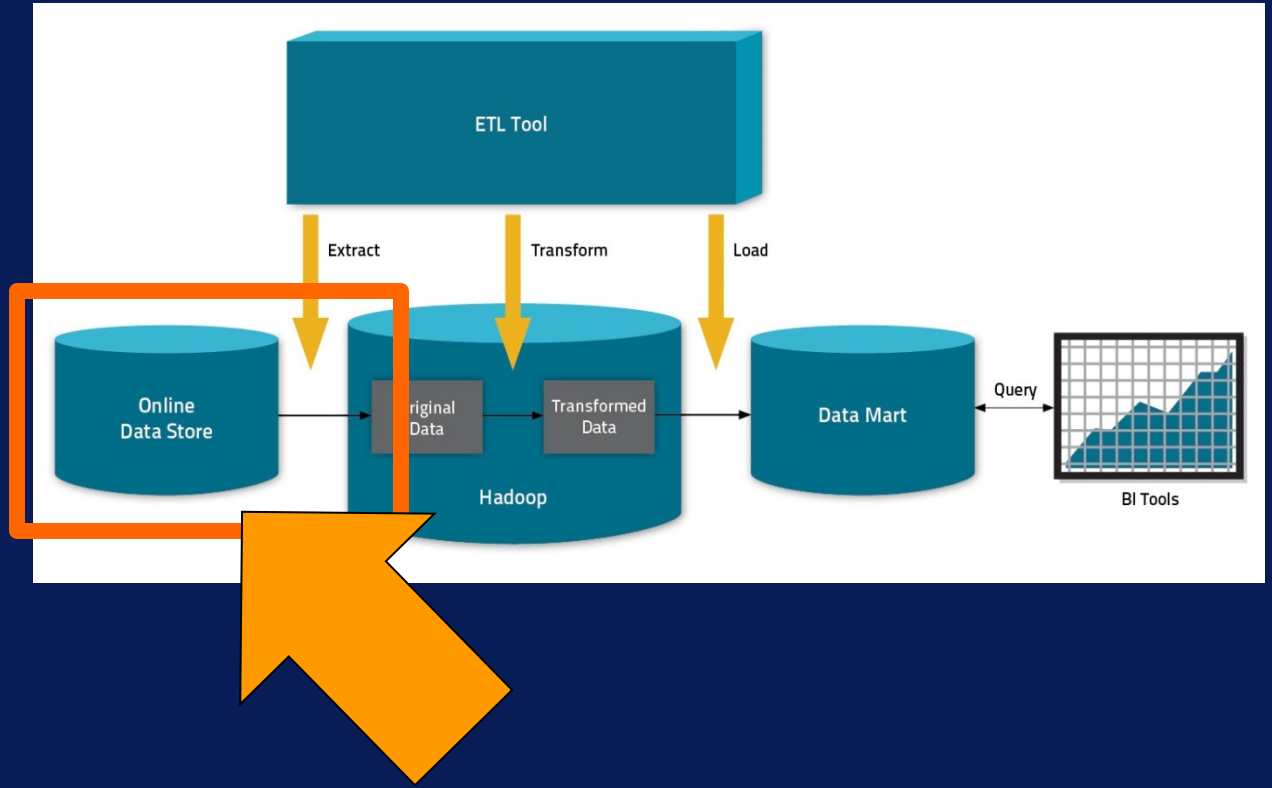


Originally developed at Yahoo 2006

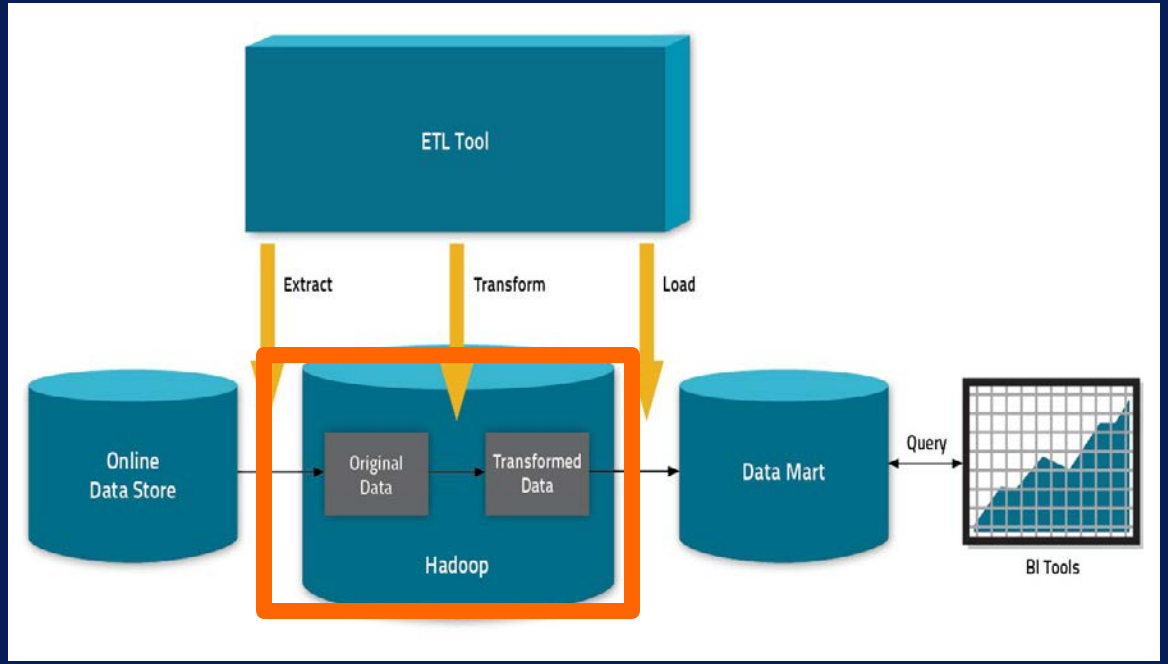
Pig for ETL

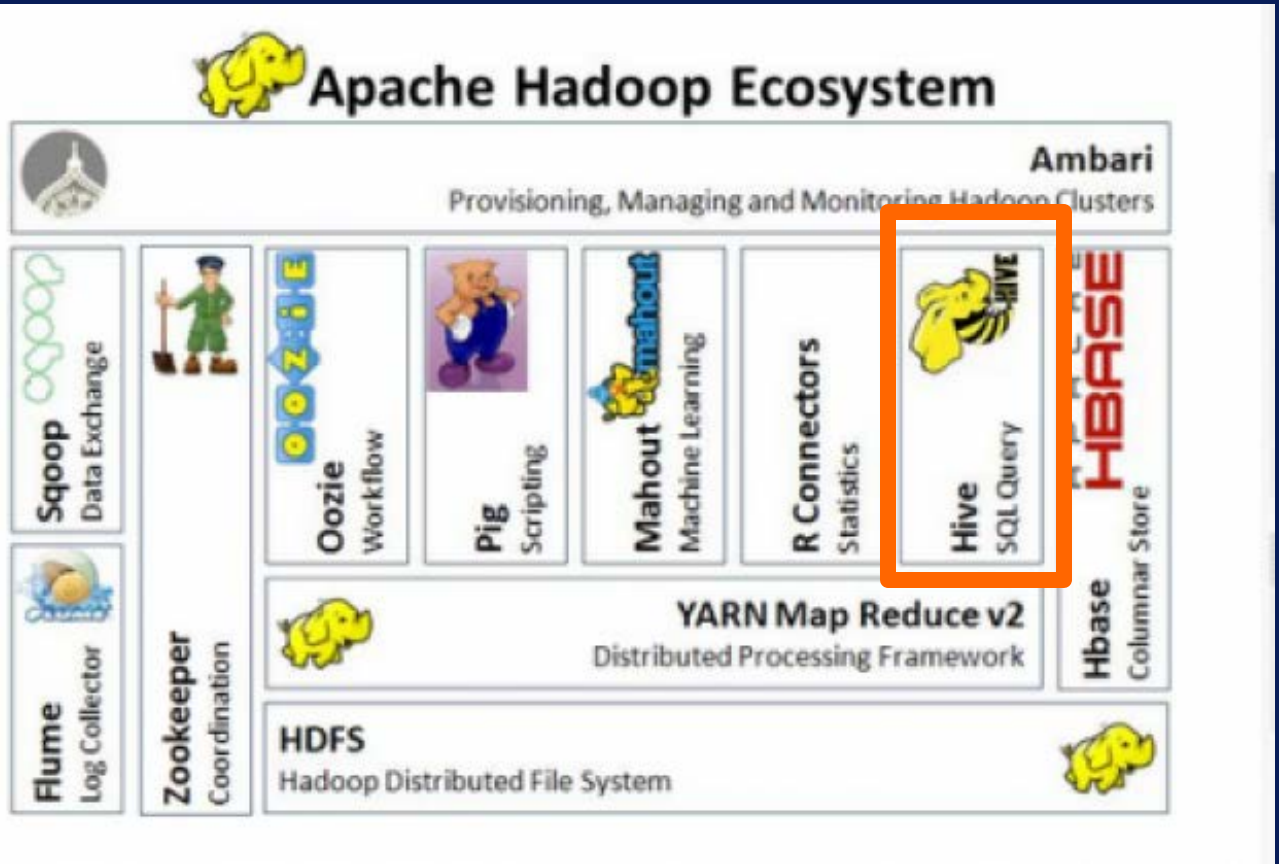

ETL Tool
Extract
Transform
Load
Online
Data Store
Original
Data
Transformed
Data
Hadoop
Data Mart
Query
BI Tools

Pig for ETL


ETL Tool
Extract
Transform
Load
Online
Data Store
riginal
Data
Transformed
Data
Hadoop
Data Mart
Query
BI Tools

Pig for ETL


ETL Tool
Extract
Transform
Load
Online
Data Store
Original
Data
Transformed
Data
Hadoop
Data Mart
Query
BI Tools


Apache Hadoop Ecosystem
Ambari
Provisioning, Managing and Monitoring Hadoop Clusters
Sqoop
Data Exchange
Flume
Log Collector
Zookeeper
Coordination
Oozie
Workflow
Pig
Scripting
Mahout
Machine Learning
R Connectors
Statistics
Hive
SQL Query
Hbase
Columnar Store
YARN Map Reduce v2
Distributed Processing Framework
HDFS
Hadoop Distributed File System

Apache Hive

- Data warehouse software facilitates querying and managing large datasets residing in distributed storage

Apache Hive

SQL-like language!

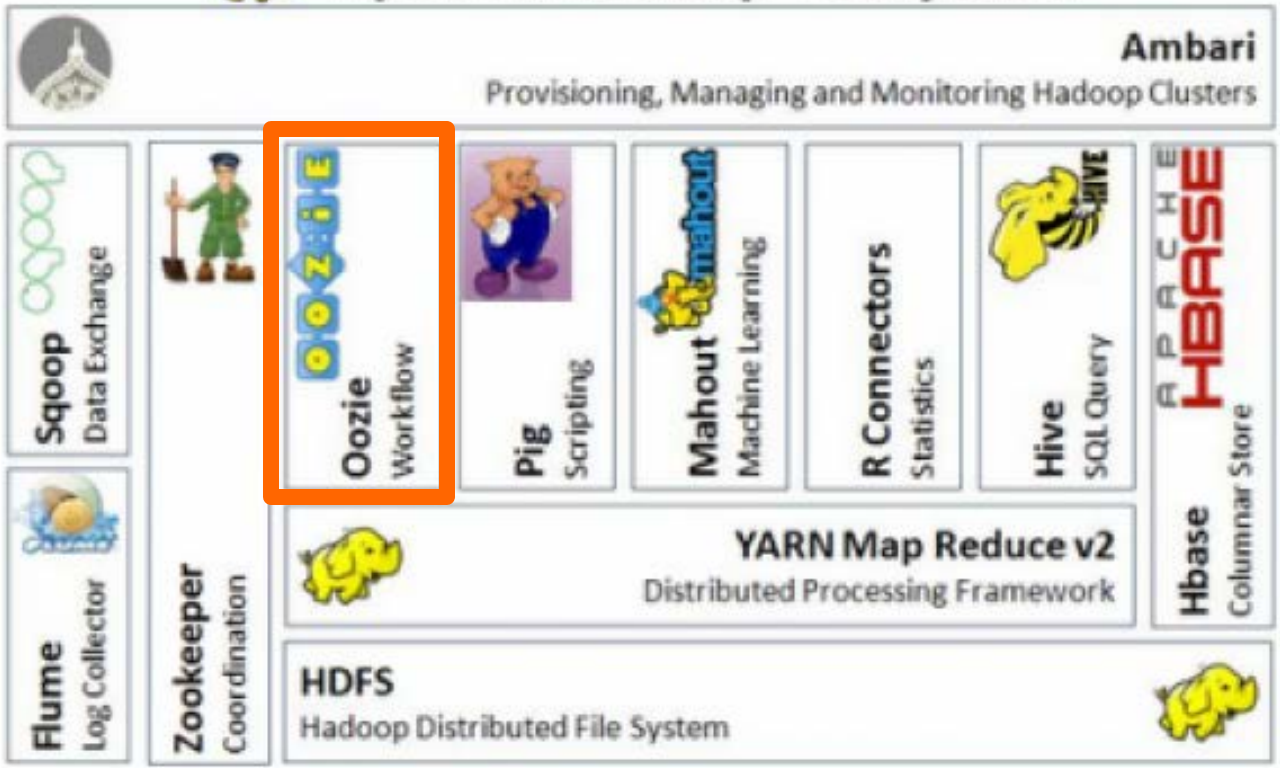
Apache Hive

Facilitates querying and managing large datasets in HDFS

Apache Hive

Mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL

Apache Hadoop Ecosystem



Oozie

Workflow scheduler system to manage Apache Hadoop jobs

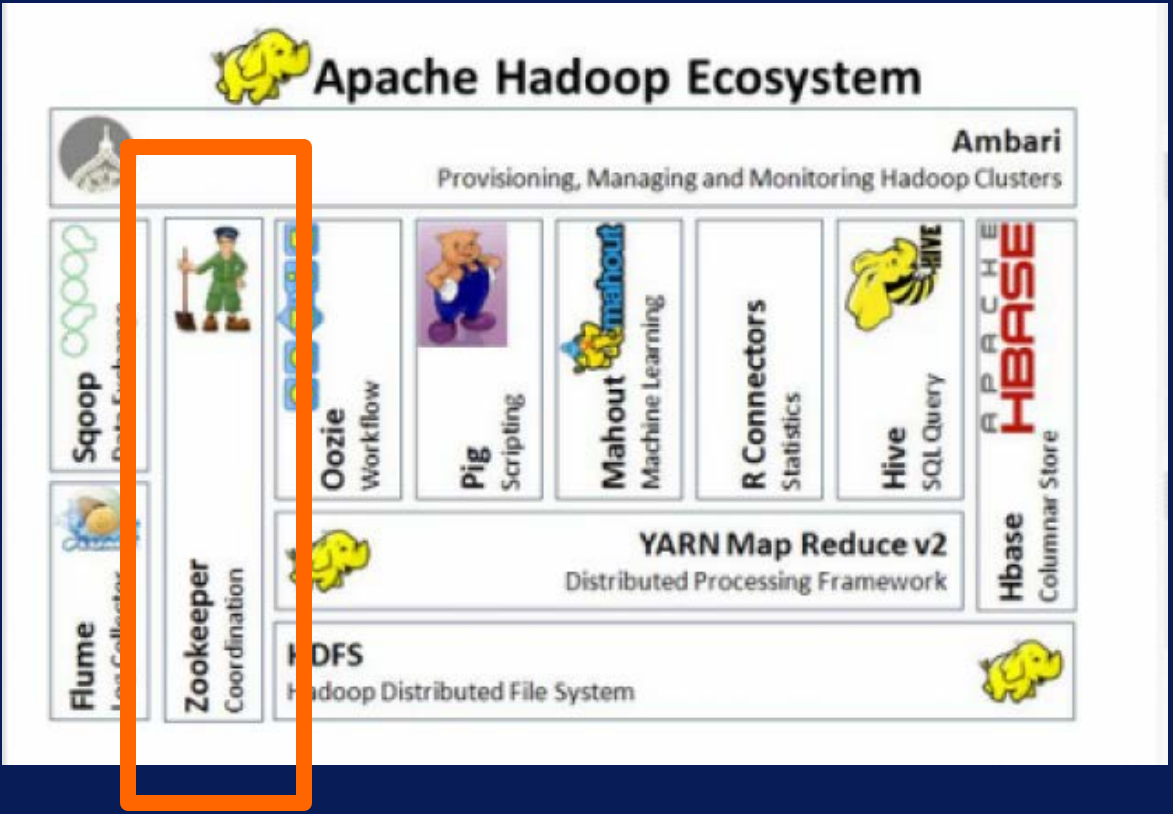
Oozie

Oozie Coordinator jobs!

Oozie

Supports

MapReduce, Pig, Apache Hive, and Sqoop, etc.


Apache Hadoop Ecosystem
Ambari
Provisioning, Managing and Monitoring Hadoop Clusters
Sqoop
Flume
Zookeeper
Coordination
Oozie
Workflow
Pig
Scripting
Mahout
Machine Learning
R Connectors
Statistics
Hive
SQL Query
APACHE
HBASE
Hbase
Columnar Store
YARN Map Reduce v2
Distributed Processing Framework
HDFS
Hadoop Distributed File System

Zookeeper

Provides operational services for a Hadoop cluster group services

Zookeeper

Centralized service for:

maintaining configuration information

naming services

providing distributed synchronization

and providing group services

Zookeeper

Centralized service for:

maintaining configuration information

Zookeeper

Centralized service for:

maintaining configuration information

naming services

Zookeeper

Centralized service for:

maintaining configuration information

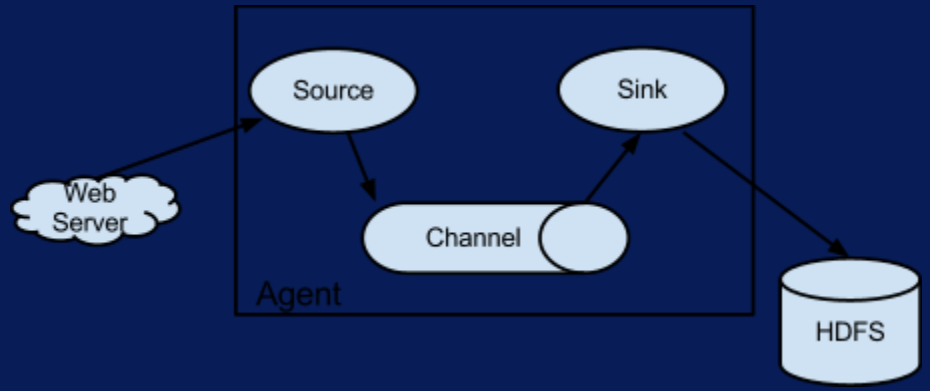
naming services

providing distributed synchronization

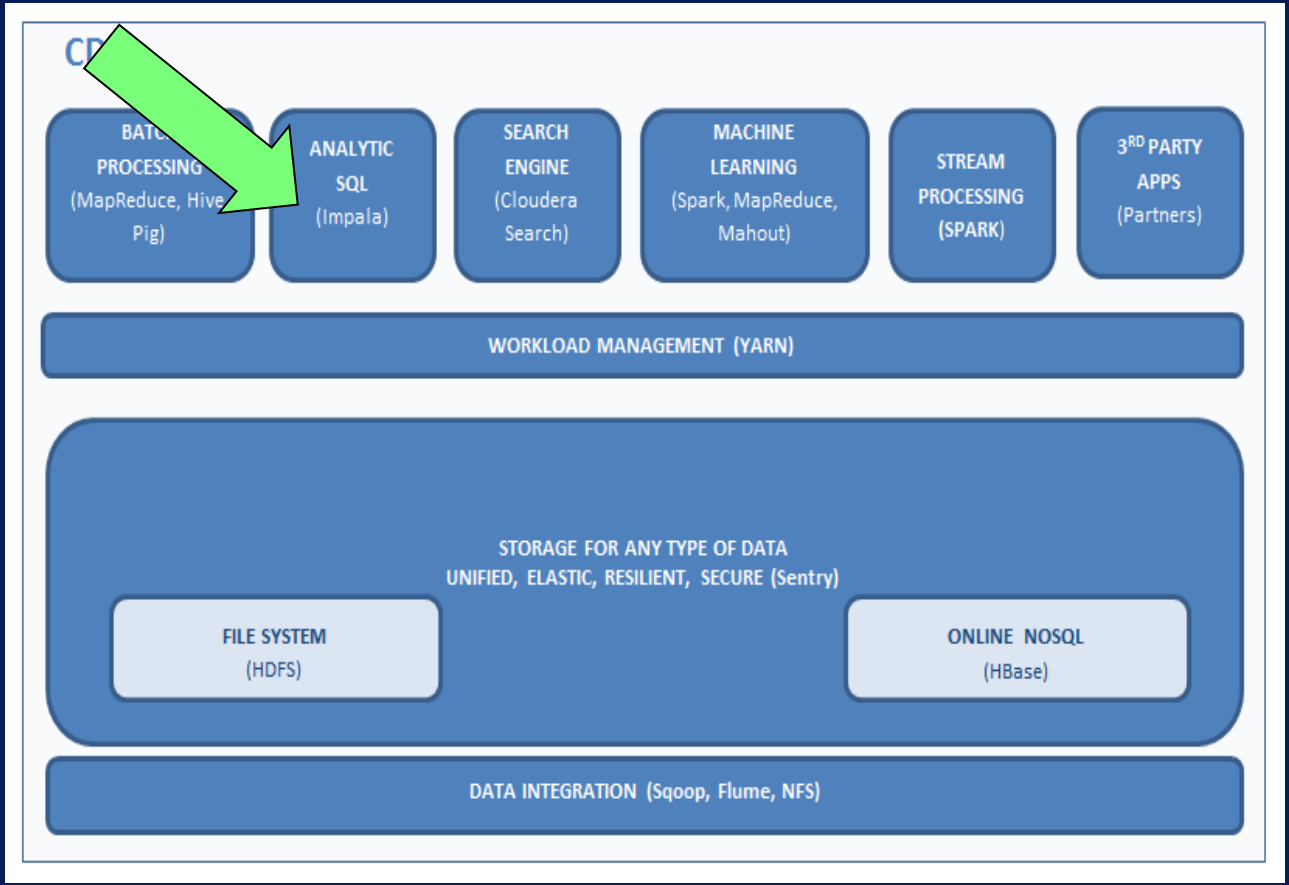
and providing group services

Flume

Distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data



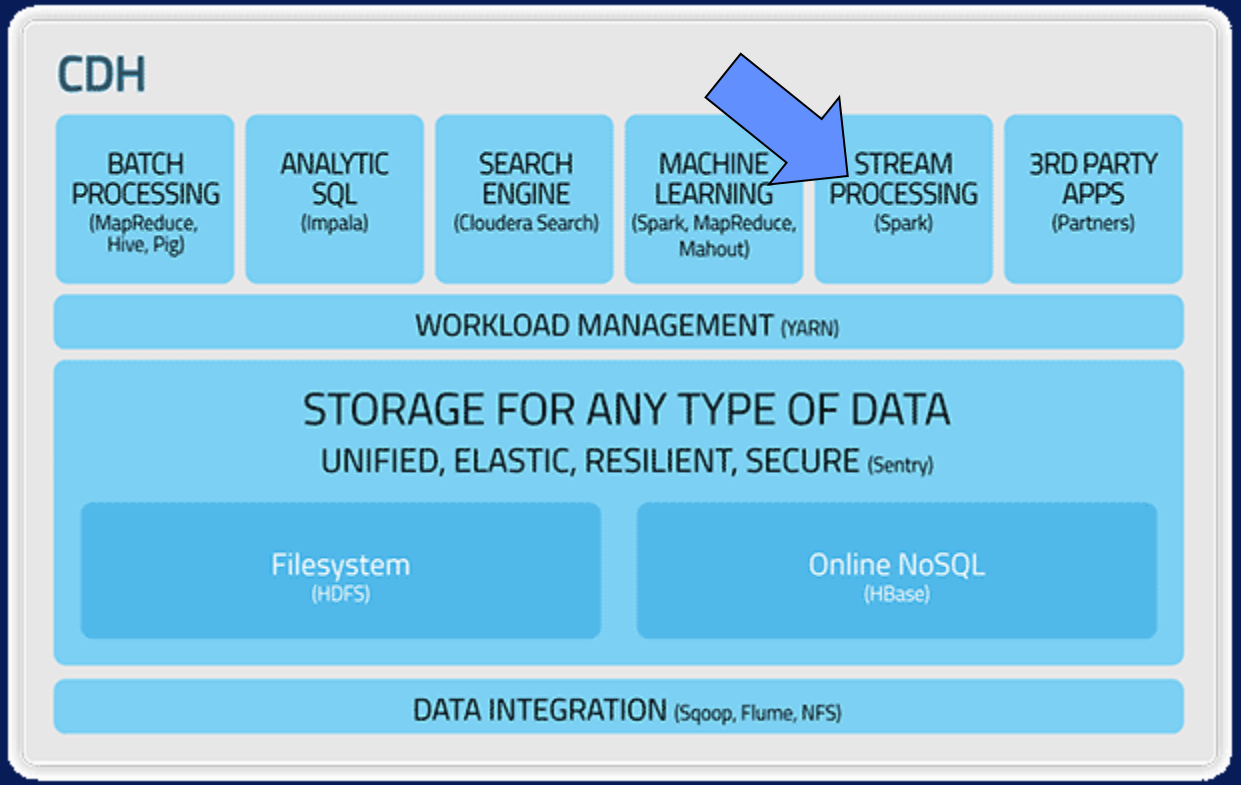
Additional Cloudera Hadoop Components
Impala


CD
BATCH PROCESSING (MapReduce, Hive, Pig)
ANALYTIC SQL (Impala)
SEARCH ENGINE (Cloudera Search)
MACHINE LEARNING (Spark, MapReduce, Mahout)
STREAM PROCESSING (SPARK)
3RD PARTY APPS (Partners)
WORKLOAD MANAGEMENT (YARN)
STORAGE FOR ANY TYPE OF DATA
UNIFIED, ELASTIC, RESILIENT, SECURE (Sentry)
FILE SYSTEM (HDFS)
ONLINE NOSQL (HBase)
DATA INTEGRATION (Sqoop, Flume, NFS)

Impala

- Cloudera's open source massively parallel processing (MPP) SQL query engine Apache Hadoop

Additional Cloudera Hadoop Components
Spark
The New Paradigm


CDH
BATCH PROCESSING (MapReduce, Hive, Pig)
ANALYTIC SQL (Impala)
SEARCH ENGINE (Cloudera Search)
MACHINE LEARNING (Spark, MapReduce, Mahout)
STREAM PROCESSING (Spark)
3RD PARTY APPS (Partners)
WORKLOAD MANAGEMENT (YARN)
STORAGE FOR ANY TYPE OF DATA
UNIFIED, ELASTIC, RESILIENT, SECURE (Sentry)
Filesystem (HDFS)
Online NoSQL (HBase)
DATA INTEGRATION (Sqoop, Flume, NFS)

Spark

Apache Spark™ is a fast and general engine for large-scale data processing

Spark Benefits

Multi-stage in-memory primitives provides performance up to 100 times faster for certain applications

Spark Benefits

Allows user programs to load data into a cluster's memory and query it repeatedly

- Well-suited to machine learning!!!

Up Next

Tour of the Cloudera’s Quick Start VM

Apache Pig

- *Overview of apps, high level languages, services*
- *Databases/Stores*
- *Querying*
- *Machine Learning*
- *Graph Processing*

Databases/Stores

- ***Avro:*** data structures within context of Hadoop MapReduce jobs.
- ***Hbase:*** distributed non-relational database
- ***Cassandra:*** distributed data management system

Querying

- ***Pig :*** Platform for analyzing large data sets in HDFS
- ***Hive :*** Query and manage large datasets
- ***Impala :*** High-performance, low-latency SQL querying of data in Hadoop file formats
- ***Spark :*** General processing engine for streaming, SQL, machine learning and graph processing.

Machine Learning, Graph Processing

- ***Giraph:*** Iterative graph processing using Hadoop framework
- ***Mahout:*** Framework for machine learning applications using Hadoop, Spark
- ***Spark:*** General processing engine for streaming, SQL, machine learning and graph processing.

Apache Pig

- *Pig components – PigLatin, and infrastructure layer*
- *Typical Pig use cases*
- *Run Pig with Hadoop integration.*

Apache Pig

- *Platform for data processing*
- *Pig Latin: High level language*
- *Pig execution environment: Local, MapReduce, Tez*
- *In built operators and functions*
- *Extensible*

Pig Usage Areas

- *Extract, Transform, Load (ETL) operations*
- *Manipulating, analyzing "raw" data*
- ***Widely used, extensive list at:*** *https://cwiki.apache.org/confluence/display/PIG/PoweredBy*

Pig Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ hdfs dfs -put /etc/passwd /user/cloudera/
[cloudera@quickstart ~]$ pig -x mapreduce
```

- ***Load passwd file and work with data.***
- ***Step 1:***
  hdfs dfs -put /etc/passwd /user/cloudera

(Note: this is a single line)

- ***Step 2:***
  pig -x mapreduce

Pig Example

cloudera@quickstart:~

File Edit View Search Terminal Help

```
grunt> A = load '/user/cloudera/passwd' using PigStorage(':');
grunt> B = foreach A generate $0, $4, $5 ;
grunt> dump B;
```

- ***Puts you in "grunt" shell.***
- ***clear***
- ***Load the file:***
  load '/user/cloudera/passwd' using PigStorage(':');
- ***Pick subset of values:***
  B = foreach A generate $0, $4, $5;
  dump B;

Pig Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
ne.mapReduceLayer.MapReduceLauncher - HadoopJobId: job_1443986695067_0002
2015-10-04 13:36:58,424 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Processing aliases A,B
2015-10-04 13:36:58,424 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - detailed locations: M: A[1,4],B[2,4] C:  R
:
2015-10-04 13:36:58,424 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - More information at: http://localhost:5003
0/jobdetails.jsp?jobid=job_1443986695067_0002
2015-10-04 13:36:58,480 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - 0% complete
2015-10-04 13:37:12,355 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - 50% complete
2015-10-04 13:37:14,056 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
2015-10-04 13:37:14,126 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - 100% complete
2015-10-04 13:37:14,128 [main] INFO  org.apache.pig.tools.pigstats.SimplePigStat
s - Script Statistics:

HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.4.2  0.12.0-cdh5.4.2 cloudera        2015-10-04 13:36:53     2015-10-
04 13:37:14     UNKNOWN
```

Backend Hadoop job info.

Pig Example

cloudera@quickstart:~

File Edit View Search Terminal Help

```
(root,root,/root)
(bin,bin,/bin)
(daemon,daemon,/sbin)
(adm,adm,/var/adm)
(lp,lp,/var/spool/lpd)
(sync,sync,/sbin)
(shutdown,shutdown,/sbin)
(halt,halt,/sbin)
(mail,mail,/var/spool/mail)
(uucp,uucp,/var/spool/uucp)
(operator,operator,/root)
(games,games,/usr/games)
(gopher,gopher,/var/gopher)
(ftp,FTP User,/var/ftp)
(nobody,Nobody,/)
(dbus,System message bus,/)
(vcsa,virtual console memory owner,/dev)
(abrt,,/etc/abrt)
(haldaemon,HAL daemon,/)
(ntp,,/etc/ntp)
(saslauth,"Saslauthd user",/var/empty/saslauth)
(postfix,,/var/spool/postfix)
(sshd,Privilege-separated SSH,/var/empty/sshd)
(tcpdump,,/)
```

- ***Outputs username, full name, and home directory path.***

Pig Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help

grunt> store B into 'userinfo.out';
2015-10-04 13:39:26,711 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2015-10-04 13:39:26,711 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
2015-10-04 13:39:26,740 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: UNKNOWN
2015-10-04 13:39:26,741 [main] INFO  org.apache.pig.newplan.logical.optimizer.Lo
gicalPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPrune, DuplicateFor
EachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptim
izer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimiz
er, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter],
RULES_DISABLED=[FilterLogicExpressionSimplifier, PartitionFilterOptimizer]}
2015-10-04 13:39:26,742 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.textoutputformat.separator is deprecated. Instead, use mapreduce.
output.textoutputformat.separator
2015-10-04 13:39:26,744 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? fal
se
2015-10-04 13:39:26,745 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
2015-10-04 13:39:26,745 [main] INFO  org.apache.pig.backend.hadoop.executionengi
```

- ***Can store this processed data in HDFS***
- ***Command:***
  store B into 'userinfo.out';

Pig Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
grunt> quit
[cloudera@quickstart ~]$ clear

[cloudera@quickstart ~]$ hdfs dfs -ls /user/cloudera
Found 2 items
-rw-r--r--   1 cloudera cloudera       2561 2015-10-04 12:41 /user/cloudera/pass
wd
drwxr-xr-x   - cloudera cloudera          0 2015-10-04 13:39 /user/cloudera/user
info.out
[cloudera@quickstart ~]$ hdfs dfs -ls /user/cloudera/userinfo.out
Found 2 items
-rw-r--r--   1 cloudera cloudera          0 2015-10-04 13:39 /user/cloudera/user
info.out/_SUCCESS
-rw-r--r--   1 cloudera cloudera       1459 2015-10-04 13:39 /user/cloudera/user
info.out/part-m-00000
[cloudera@quickstart ~]$
```

- ***Verify the new data is in HDFS.***

Summary

- *Used interactive shell for Pig example*
- *Can also run using scripts*
- *Also as embedded programs in a host language (Java for example).*

Apache Hive

- *Query and manage data using HiveQL*
- *Run interactively using beeline.*
- *Other run mechanisms*

Apache Hive

- *Data warehouse software*
- *HiveQL: SQL like language to structure, and query data*
- *Execution environment: MapReduce, Tez, Spark*
- *Data in HDFS, HBase*
- *Custom mappers/reducers*

Hive Usage Areas

- *Data mining, analytics*
- *Machine learning*
- *Ad hoc analysis*
- ***Widely used, extensive list at:***
  https://cwiki.apache.org/confluence/display/Hive/PoweredBy

Hive Example

- *Revisit /etc/passwd file example from Pig lesson video.*
- *Start by loading file into HDFS:*
  hdfs dfs -put /etc/passwd /tmp/
- *Run beeline to access interactively:*
  beeline -u jdbc:hive2://

Hive Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ hdfs dfs -put /etc/passwd /tmp/
[cloudera@quickstart ~]$ hdfs dfs -ls /tmp/
Found 3 items
drwxr-xr-x   - hdfs     supergroup          0 2015-06-09 03:36 /tmp/hadoop-yarn
drwx-wx-wx   - hive     supergroup          0 2015-10-04 15:25 /tmp/hive
-rw-r--r--   1 cloudera supergroup       2561 2015-10-04 15:27 /tmp/passwd
[cloudera@quickstart ~]$ beeline -u jdbc:hive2://
scan complete in 4ms
Connecting to jdbc:hive2://
Added [/usr/lib/hive/lib/hive-contrib.jar] to class path
Added resources: [/usr/lib/hive/lib/hive-contrib.jar]
Connected to: Apache Hive (version 1.1.0-cdh5.4.2)
Driver: Hive JDBC (version 1.1.0-cdh5.4.2)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 1.1.0-cdh5.4.2 by Apache Hive
0: jdbc:hive2://>
0: jdbc:hive2://>
```

- *Copy passwd file to HDFS*
- *Running interactively usiing beeline.*

Hive Example: Command list

```
CREATE TABLE userinfo ( uname STRING, pswd STRING, uid INT, gid INT, fullname STRING, hdir STRING, shell STRING ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ':' STORED AS TEXTFILE;

LOAD DATA INPATH '/tmp/passwd' OVERWRITE INTO TABLE userinfo;

SELECT uname, fullname, hdir FROM userinfo ORDER BY uname ;
```

Hive Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ beeline -u jdbc:hive2://
scan complete in 2ms
Connecting to jdbc:hive2://
Added [/usr/lib/hive/lib/hive-contrib.jar] to class path
Added resources: [/usr/lib/hive/lib/hive-contrib.jar]
Connected to: Apache Hive (version 1.1.0-cdh5.4.2)
Driver: Hive JDBC (version 1.1.0-cdh5.4.2)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 1.1.0-cdh5.4.2 by Apache Hive
0: jdbc:hive2://> CREATE TABLE userinfo ( uname STRING, pswd STRING, uid INT, gi
d INT, fullname STRING, hdir STRING, shell STRING ) ROW FORMAT DELIMITED FIELDS
TERMINATED BY ':' STORED AS TEXTFILE;
OK
No rows affected (1.077 seconds)
0: jdbc:hive2://>
0: jdbc:hive2://>
```

- ***Run the Create table command***

Hive Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
Connected to: Apache Hive (version 1.1.0-cdh5.4.2)
Driver: Hive JDBC (version 1.1.0-cdh5.4.2)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 1.1.0-cdh5.4.2 by Apache Hive
0: jdbc:hive2://> CREATE TABLE userinfo ( uname STRING, pswd STRING, uid INT, gi
d INT, fullname STRING, hdir STRING, shell STRING ) ROW FORMAT DELIMITED FIELDS
TERMINATED BY ':' STORED AS TEXTFILE;
OK
No rows affected (1.077 seconds)
0: jdbc:hive2://>
0: jdbc:hive2://> LOAD DATA INPATH '/tmp/passwd' OVERWRITE INTO TABLE userinfo;

Loading data to table default.userinfo
15/10/04 15:41:23 [HiveServer2-Background-Pool: Thread-26]: ERROR hdfs.KeyProvid
erCache: Could not find uri with key [dfs.encryption.key.provider.uri] to create
 a keyProvider !!
chgrp: changing ownership of 'hdfs://quickstart.cloudera:8020/user/hive/warehous
e/userinfo/passwd': User does not belong to hive
Table default.userinfo stats: [numFiles=1, numRows=0, totalSize=2561, rawDataSiz
e=0]
OK
No rows affected (0.491 seconds)
0: jdbc:hive2://>
0: jdbc:hive2://>
```

- *Load passwd file from HDFS*

Hive Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
0: jdbc:hive2://> SELECT uname, fullname, hdir FROM userinfo ORDER BY uname ;
Query ID = cloudera_20151004154141_aa932488-439e-4378-8a93-3adcc30676ef
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
15/10/04 15:41:57 [HiveServer2-Background-Pool: Thread-34]: WARN mapreduce.JobSu
bmitter: Hadoop command-line option parsing not performed. Implement the Tool in
terface and execute your application with ToolRunner to remedy this.
Starting Job = job_1443997416755_0001, Tracking URL = http://quickstart.cloudera
:8088/proxy/application_1443997416755_0001/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1443997416755_0001
WARN  : Hadoop command-line option parsing not performed. Implement the Tool int
erface and execute your application with ToolRunner to remedy this.
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
15/10/04 15:42:09 [HiveServer2-Background-Pool: Thread-34]: WARN mapreduce.Count
ers: Group org.apache.hadoop.mapred.Task$Counter is deprecated. Use org.apache.h
adoop.mapreduce.TaskCounter instead
2015-10-04 15:42:09,249 Stage-1 map = 0%,  reduce = 0%
```

- *Select info - this launches the Hadoop job and outputs once its complete.*

Hive Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 1.97 sec   HDFS Read: 8828 HD
FS Write: 1459 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 970 msec
OK
+----------------+------------------------------+-----------------------------
-+--+
|     uname      |            fullname          |             hdir
 |
+----------------+------------------------------+-----------------------------
-+--+
| abrt           |                              | /etc/abrt
 |
| adm            | adm                          | /var/adm
 |
| avahi-autoipd  | Avahi IPv4LL Stack           | /var/lib/avahi-autoipd
 |
| bin            | bin                          | /bin
 |
| cloudera       |                              | /home/cloudera
 |
| cloudera-scm   | Cloudera Manager             | /var/lib/cloudera-scm-server
 |
| daemon         | daemon                       | /sbin
```

- ***Completed MapReduce jobs; output shows username, fullname, and home directory.***

Summary

- ***Used beeline for interactive Hive example***
- ***Can also use***
  - *Hive command line interface (CLI)*
  - *Hcatalog*
  - *WebHcat.*

Apache HBase

- *Hbase features*
- *Run interactively using HBase shell.*
- *List other access mechanisms*

Apache HBase

- *Scalable data store*
- *Non-relational distributed database*
- *Runs on top of HDFS*
- *Compression*
- *In-memory operations: MemStore, BlockCache*

HBase Features

- *Consistency*
- *High Availability*
- *Automatic Sharding*

Hbase Features

- *Replication*
- *Security*
- *SQL like access (Hive, Spark, Impala)*

HBase Example

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ hbase shell
2015-10-04 18:37:51,715 INFO  [main] Configuration.deprecation: hadoop.native.li
b is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.0.0-cdh5.4.2, rUnknown, Tue May 19 17:07:29 PDT 2015

hbase(main):001:0> create 'userinfotable',{NAME=>'username'},{NAME=>'fullname'},
{NAME=>'homedir'}
0 row(s) in 0.5190 seconds

=> Hbase::Table - userinfotable
hbase(main):002:0>
```

- ***Start HBase shell:***
  hbase shell

HBase Example

- *Create Table: create ‘usertableinfo’,{NAME=>’username’},{NAME=>’fullname’},{NAME=>’hom*

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ hbase shell
2015-10-04 18:37:51,715 INFO  [main] Configuration.deprecation: hadoop.native.li
b is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.0.0-cdh5.4.2, rUnknown, Tue May 19 17:07:29 PDT 2015

hbase(main):001:0> create 'userinfotable',{NAME=>'username'},{NAME=>'fullname'},
{NAME=>'homedir'}
0 row(s) in 0.5190 seconds

=> Hbase::Table - userinfotable
hbase(main):002:0>
```

HBase Example

- *Add data: put ‘userinfotable’,’r1’,’username’,’vcsa’*

cloudera@quickstart:~

File Edit View Search Terminal Help

```
hbase(main):001:0> create 'userinfotable',{NAME=>'username'},{NAME=>'fullname'},
{NAME=>'homedir'}
0 row(s) in 0.5190 seconds

=> Hbase::Table - userinfotable
hbase(main):002:0> put 'userinfotable','r1','username','vcsa'
0 row(s) in 0.1360 seconds

hbase(main):003:0> put 'userinfotable','r2','username','sasuser'
0 row(s) in 0.0050 seconds

hbase(main):004:0> put 'userinfotable','r3','username','postfix'
0 row(s) in 0.0040 seconds

hbase(main):005:0> put 'userinfotable','r1','fullname','VirtualMachine Admin'
0 row(s) in 0.0040 seconds

hbase(main):006:0> put 'userinfotable','r2','fullname','SAS Admin'
0 row(s) in 0.0040 seconds

hbase(main):007:0> put 'userinfotable','r3','fullname','Postfix User'
0 row(s) in 0.0060 seconds

hbase(main):008:0>
```

HBase Example

- *Scan table after data entry: scan ‘userinfotable’*

cloudera@quickstart:~

File Edit View Search Terminal Help

```
hbase(main):026:0> scan 'userinfotable'
ROW                    COLUMN+CELL
 r1                    column=fullname:, timestamp=1444009196964, value=VirtualMa
                       chine Admin
 r1                    column=homedir:, timestamp=1444009268956, value=/home/vcsa
 r1                    column=username:, timestamp=1444009138897, value=vcsa
 r2                    column=fullname:, timestamp=1444009206313, value=SAS Admin
 r2                    column=homedir:, timestamp=1444009278709, value=/var/sasus
                       er
 r2                    column=username:, timestamp=1444009151766, value=sasuser
 r3                    column=fullname:, timestamp=1444009233257, value=Postfix U
                       ser
 r3                    column=homedir:, timestamp=1444009289281, value=/user/post
                       fix
 r3                    column=username:, timestamp=1444009162921, value=postfix
3 row(s) in 0.0290 seconds

hbase(main):027:0>
hbase(main):028:0*
hbase(main):029:0*
hbase(main):030:0*
hbase(main):031:0*
hbase(main):032:0*
hbase(main):033:0*
```

HBase Example

- *Select info from all rows corresponding to column ‘fullname’.*

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ hbase shell
2015-10-04 19:07:50,916 INFO  [main] Configuration.deprecation: hadoop.native.li
b is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.0.0-cdh5.4.2, rUnknown, Tue May 19 17:07:29 PDT 2015

hbase(main):001:0> scan 'userinfotable',{COLUMNS=>'fullname'}
ROW                    COLUMN+CELL
 r1                    column=fullname:, timestamp=1444009196964, value=VirtualMa
                       chine Admin
 r2                    column=fullname:, timestamp=1444009206313, value=SAS Admin
 r3                    column=fullname:, timestamp=1444009233257, value=Postfix U
                       ser
3 row(s) in 0.2780 seconds

hbase(main):002:0>
```

Summary

- ***We used: Apache HBase Shell***
- ***Other options:***
  - *HBase, MapReduce*
  - *HBase API*
  - *HBase External API*