AI VIETNAM
All-in-One Course

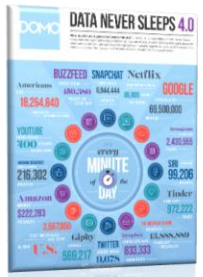
# Big Data

## *Hadoop in Google Colab*


Vinh Dinh Nguyen
PhD in Computer Science

# Big Data
## *From Zero To Mastery*

## Big Data
### *(Zero To Mastery: Big Data Analytics Tools)*

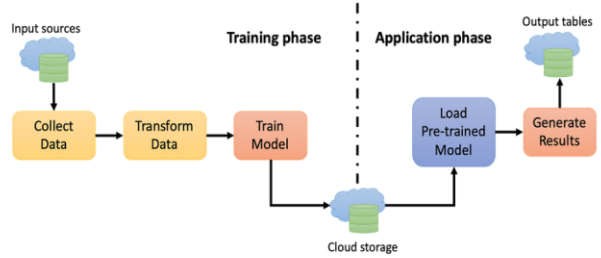





Vinh Dinh Nguyen
PhD in Computer Science

*Year 2023*

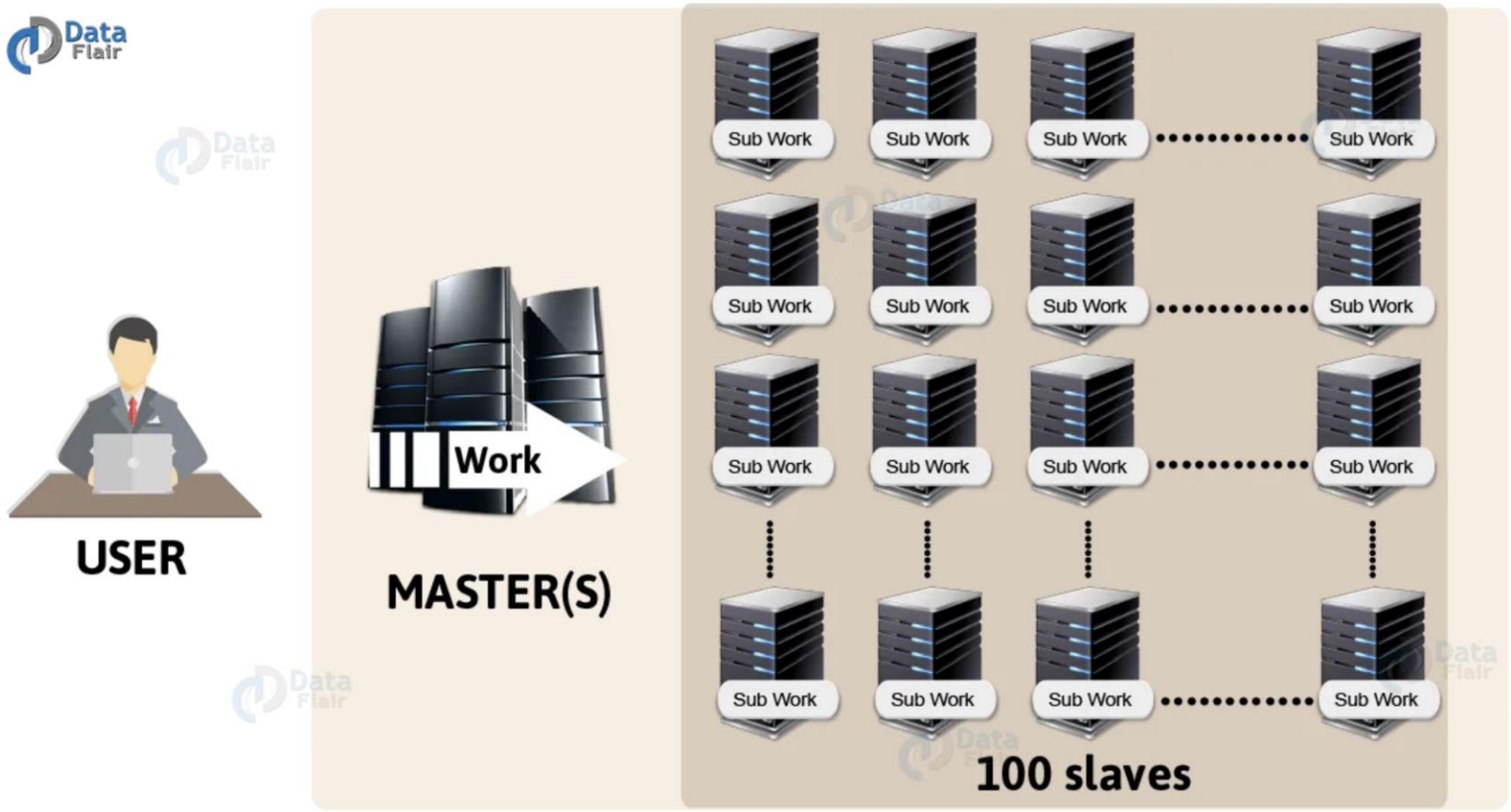
## Big Data
### *(Zero To Mastery: ML Models Using PySpark in Cloud Platform)*

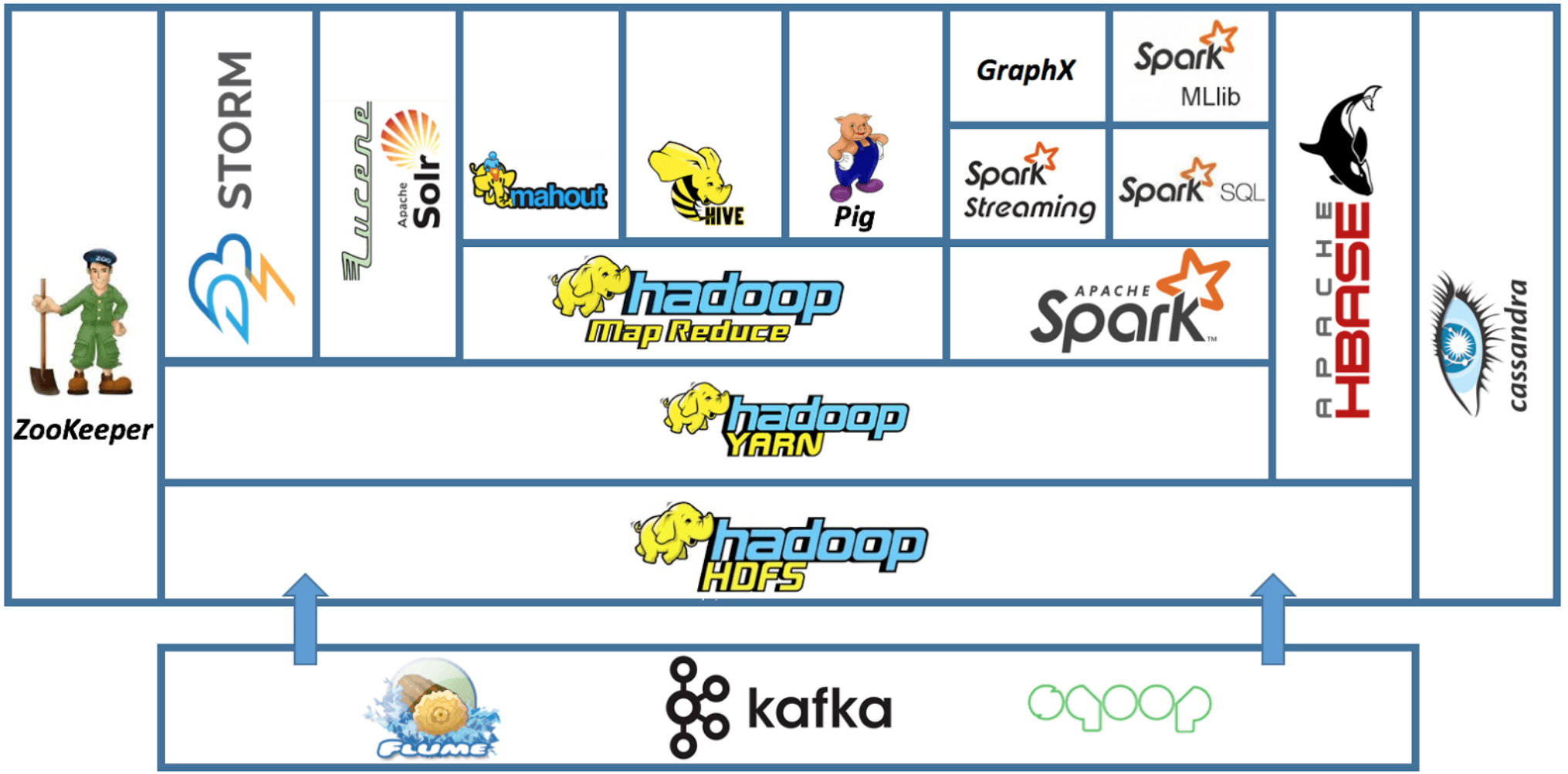

Vinh Dinh Nguyen
PhD in Computer Science

*Year 2023*

# Hadoop Architecture Review



# Hadoop Architecture Review



# Hadoop Architecture Review

Hadoop is an open-source framework which is mainly used for storage purposes and maintaining and analyzing a large amount of data or datasets on the clusters of commodity hardware, which means it is actually a data management tool.

- •Standalone Mode
- •Pseudo-distributed Mode
- •Fully-distributed Mode

This mode is useful for debugging and there isn't any need to configure core-site.xml, hdfs-site.xml, mapred-site.xml, masters & slaves. Stand-alone mode is usually the fastest mode in Hadoop.

# Hadoop Architecture Review

Hadoop is an open-source framework which is mainly used for storage purposes and maintaining and analyzing a large amount of data or datasets on the clusters of commodity hardware, which means it is actually a data management tool.

- Standalone Mode
- Pseudo-distributed Mode
- Fully-distributed Mode

In this mode, each daemon runs on separate java processes. In this mode custom configuration is required (core-site.xml, hdfs-site.xml, mapred-site.xml). Here HDFS is utilized for input and output. This mode of deployment is useful for testing and debugging purposes.

# Hadoop Architecture Review

Hadoop is an open-source framework which is mainly used for storage purposes and maintaining and analyzing a large amount of data or datasets on the clusters of commodity hardware, which means it is actually a data management tool.

- Standalone Mode
- Pseudo-distributed Mode
- Fully-distributed Mode

In this mode typically one machine in the cluster is designated as NameNode and another as Resource Manager exclusively. These are masters. All other nodes act as Data Node and Node Manager. These are the slaves. Configuration parameters and environment need to be specified for Hadoop Daemons.

# Hello World in Hadoop using Hortonworks Sandbox

Hello World program in Hadoop using Hortonworks Sandbox


HCatalog


**Step 1:**
Load Data into HDFS

**Step 2:**
Register data with HCatalog

**Step 3:**
Process data using Hive

**HDFS** is a Hadoop component where data is stored

**HCatalog** provides data in tabular format

**Hive** is a SQL-like language to query the data

# Step 1: Prepare Hadoop Environment

Instead of installing Hadoop from scratch, we will be using sandbox system provided by Hortonworks. The Hortonworks Sandbox is a personal, portable Hadoop environment that comes with a dozen interactive Hadoop tutorials.

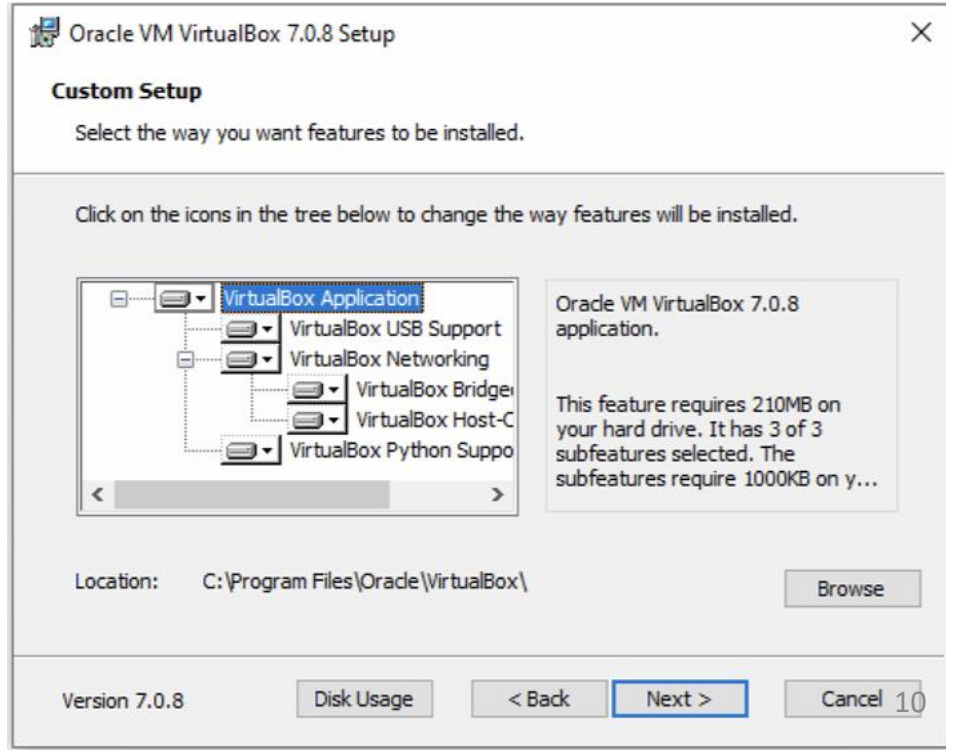
Download and install Oracle Virtual Box


https://www.virtualbox.org/wiki/Downloads

# Step 1: Prepare Hadoop Environment

Instead of installing Hadoop from scratch, we will be using sandbox system provided by Hortonworks. The Hortonworks Sandbox is a personal, portable Hadoop environment that comes with a dozen interactive Hadoop tutorials.

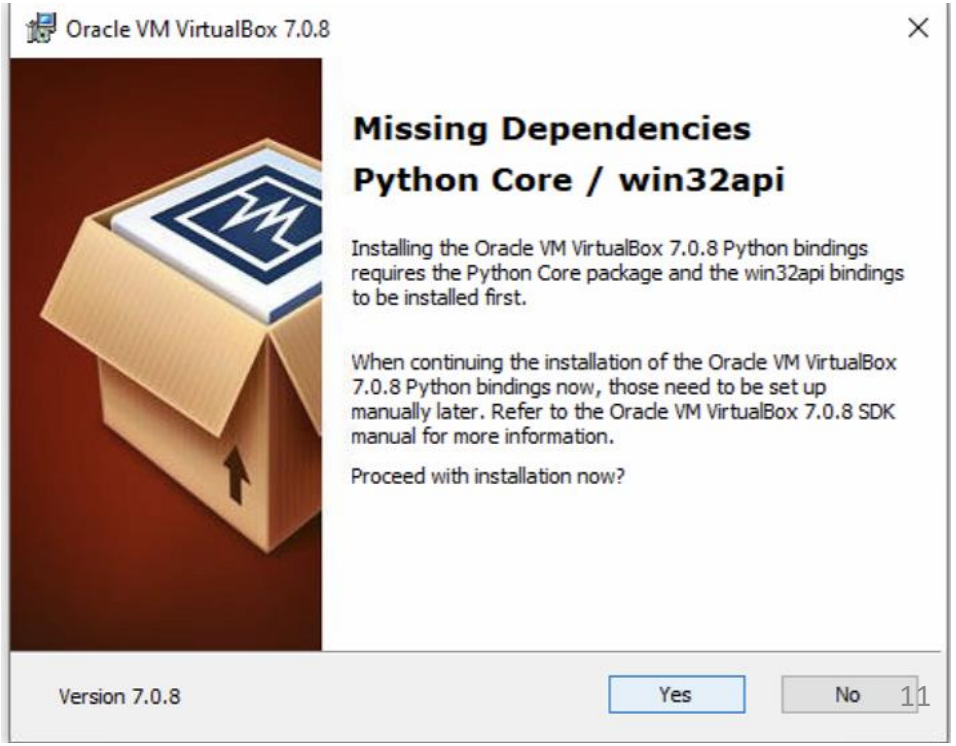
Download and install Oracle Virtual Box

# Step 1: Prepare Hadoop Environment

Instead of installing Hadoop from scratch, we will be using sandbox system provided by Hortonworks. The Hortonworks Sandbox is a personal, portable Hadoop environment that comes with a dozen interactive Hadoop tutorials.

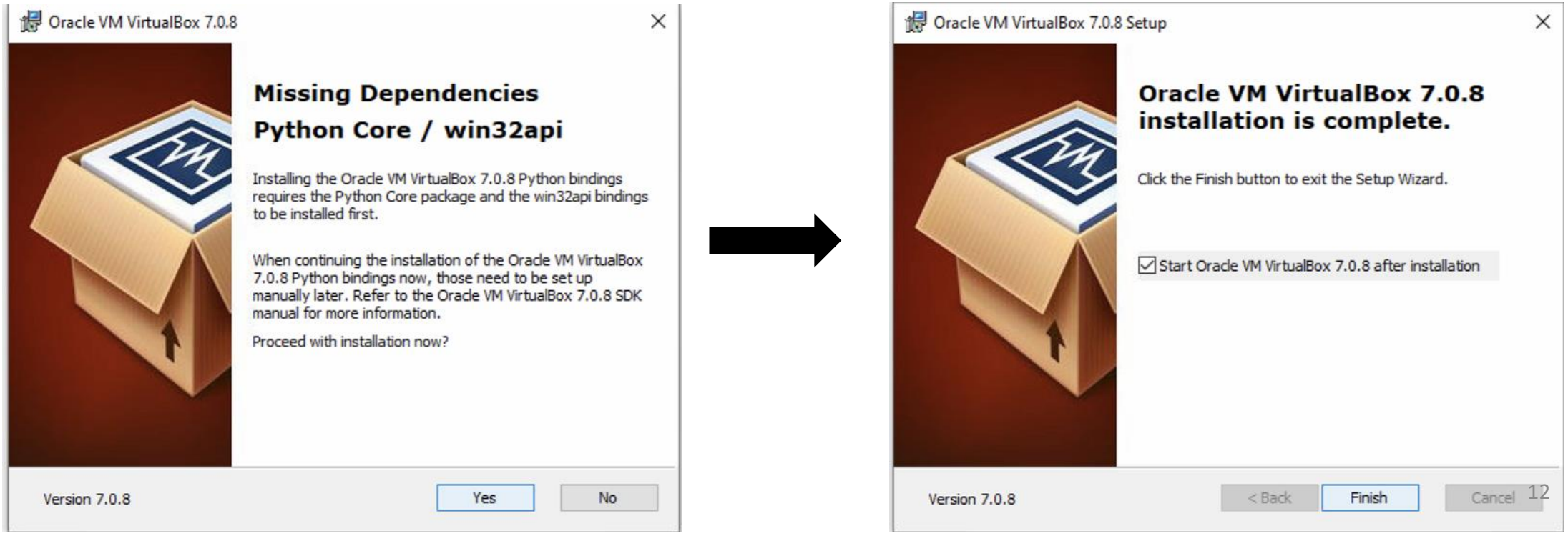
Download and install Oracle Virtual Box

# Step 1: Prepare Hadoop Environment

Instead of installing Hadoop from scratch, we will be using sandbox system provided by Hortonworks. The Hortonworks Sandbox is a personal, portable Hadoop environment that comes with a dozen interactive Hadoop tutorials.

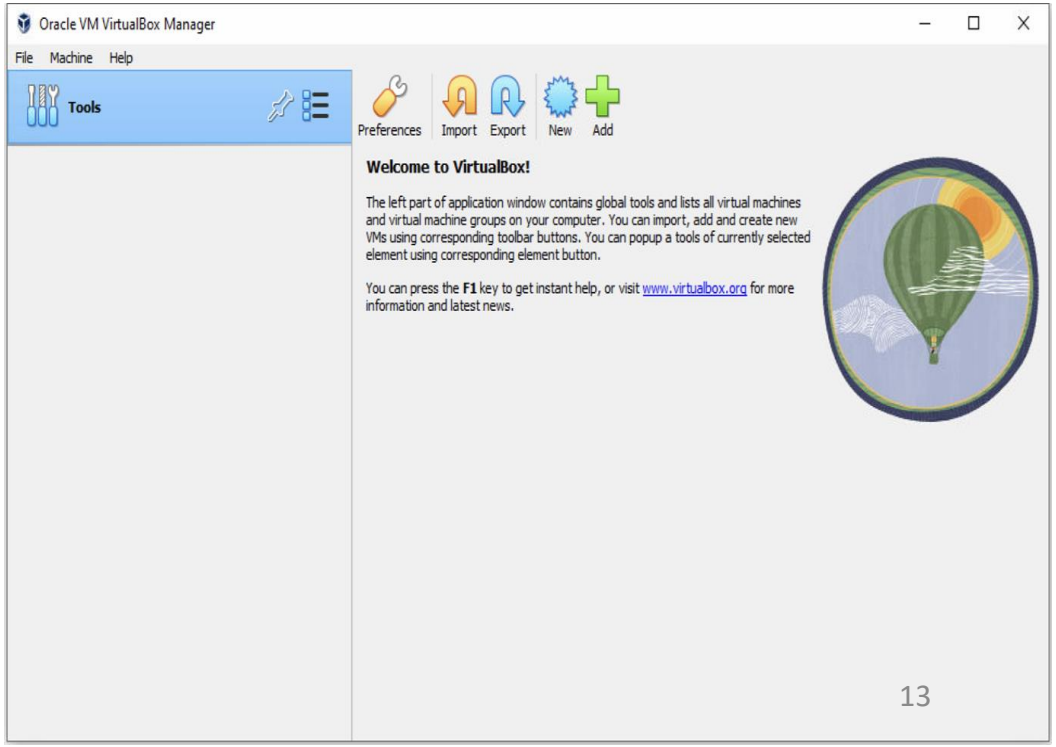
Download and install Oracle Virtual Box



# Step 1: Prepare Hadoop Environment

Instead of installing Hadoop from scratch, we will be using sandbox system provided by Hortonworks. The Hortonworks Sandbox is a personal, portable Hadoop environment that comes with a dozen interactive Hadoop tutorials.

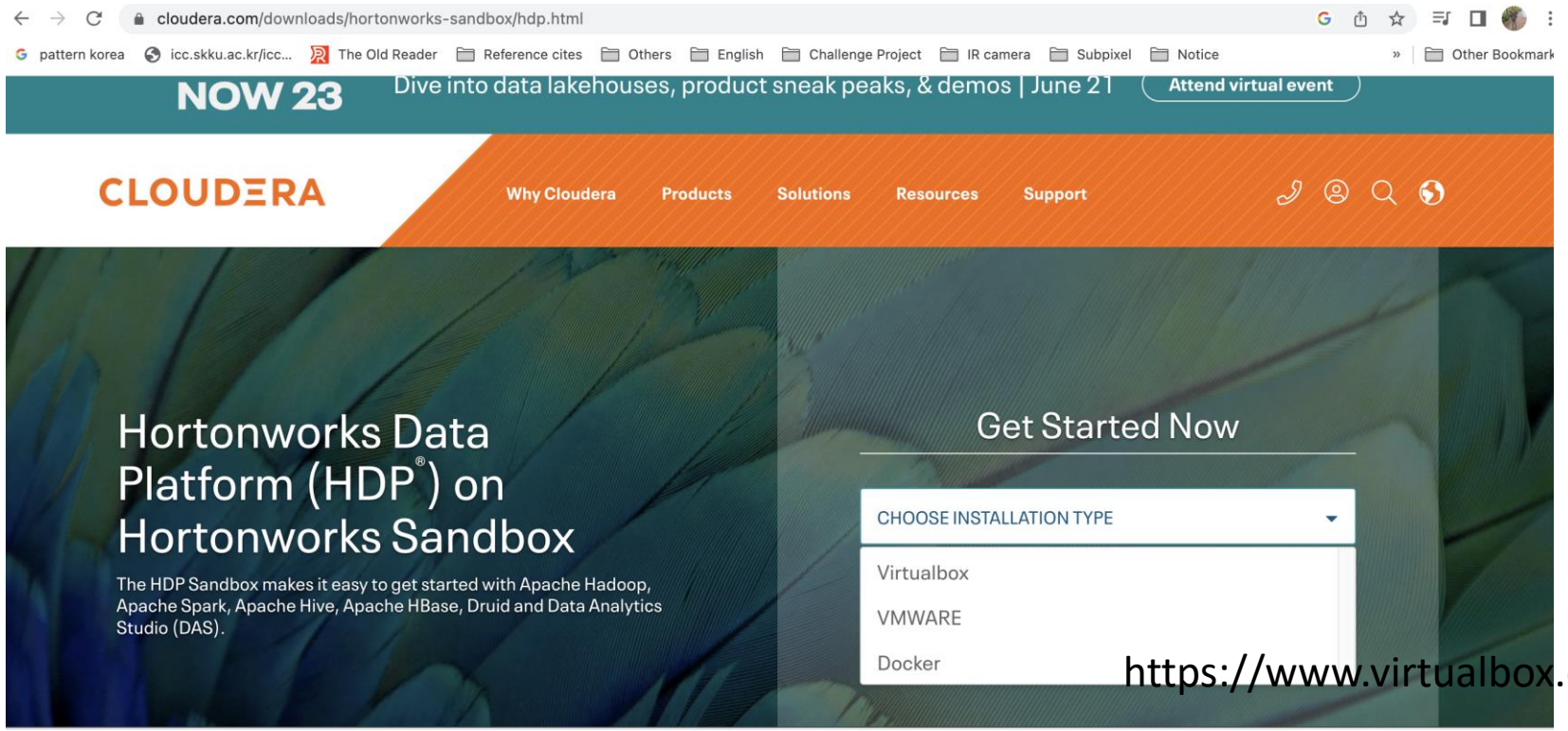
Download and install Oracle Virtual Box

# Step 2: Prepare Hadoop Environment

Instead of installing Hadoop from scratch, we will be using sandbox system provided by Hortonworks. The Hortonworks Sandbox is a personal, portable Hadoop environment that comes with a dozen interactive Hadoop tutorials.

Download Hortonworks sandbox

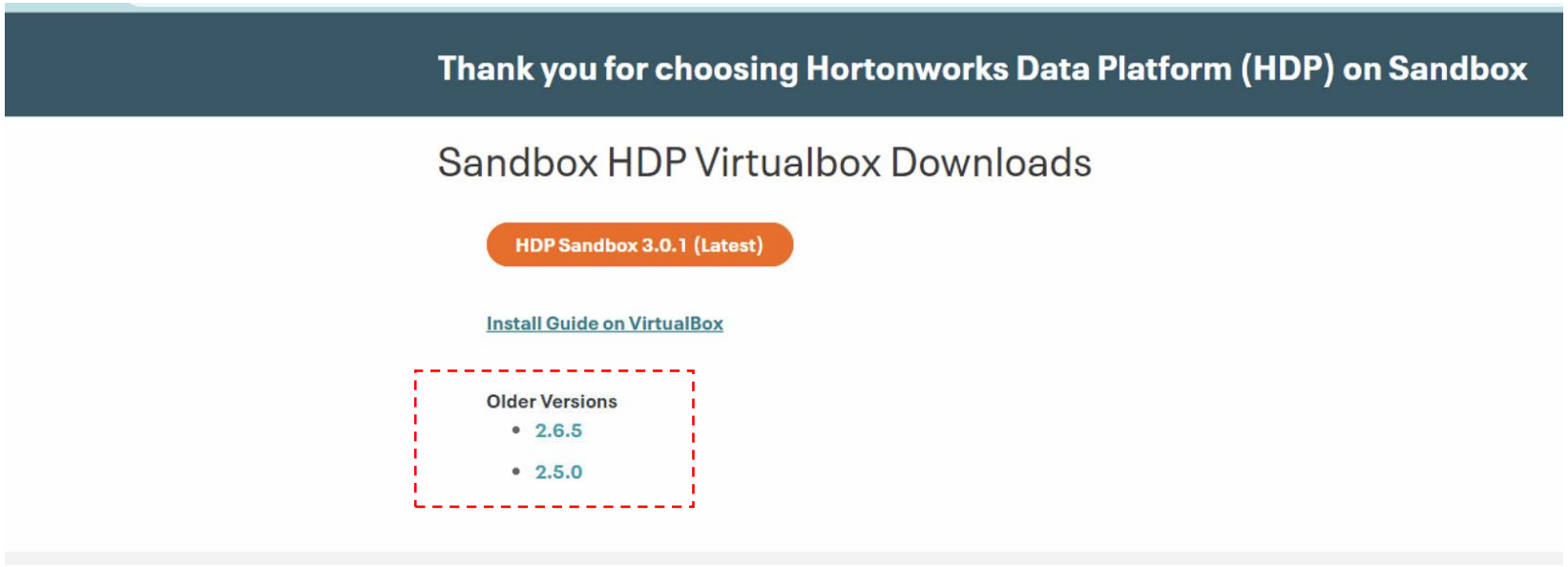



https://www.virtualbox.org/wiki/Downloads

# Step 2: Prepare Hadoop Environment

Instead of installing Hadoop from scratch, we will be using sandbox system provided by Hortonworks. The Hortonworks Sandbox is a personal, portable Hadoop environment that comes with a dozen interactive Hadoop tutorials.

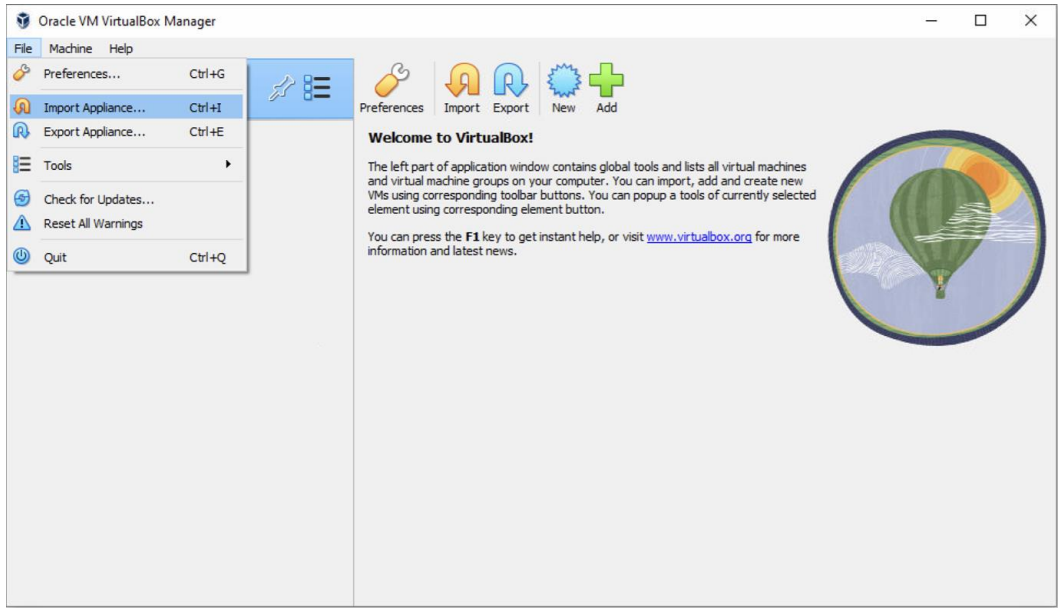
Download Hortonworks sandbox

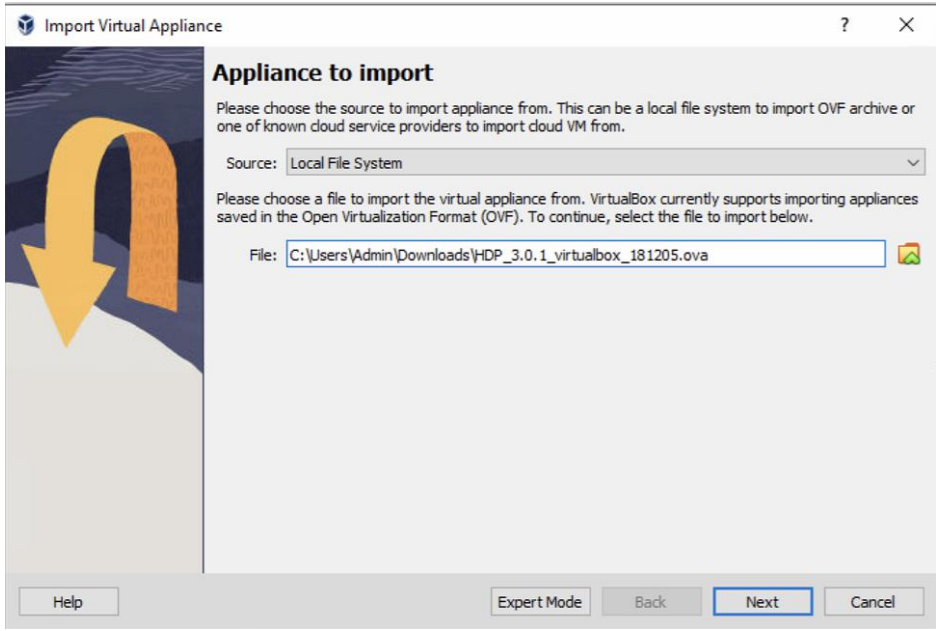



https://www.virtualbox.org/wiki/Downloads

# Step 3: Prepare Hadoop Environment

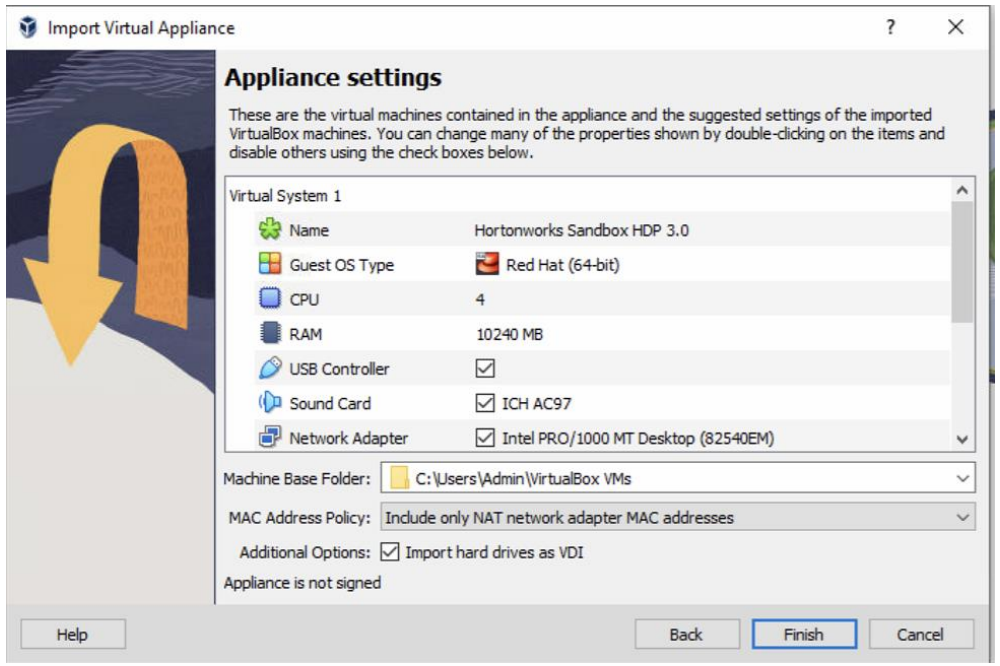
Open Oracle VM VirtualBox: File ->Import Appliance and select the .ova file

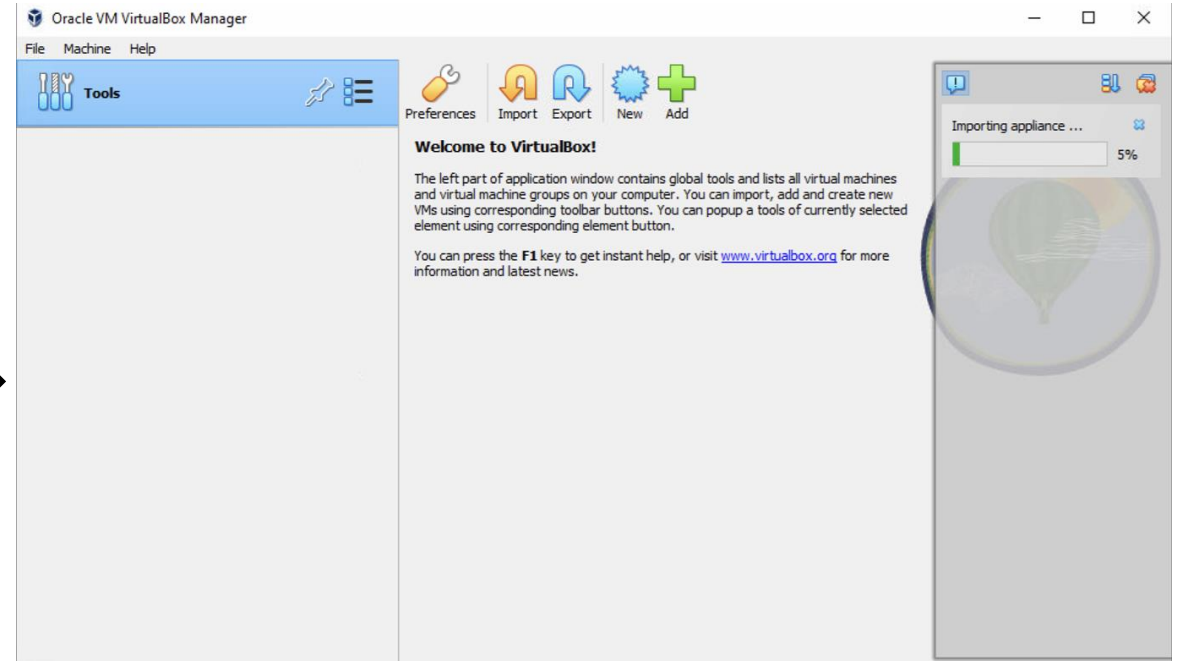


# Step 3: Prepare Hadoop Environment

Open Oracle VM VirtualBox: File ->Import Appliance and select the .ova file

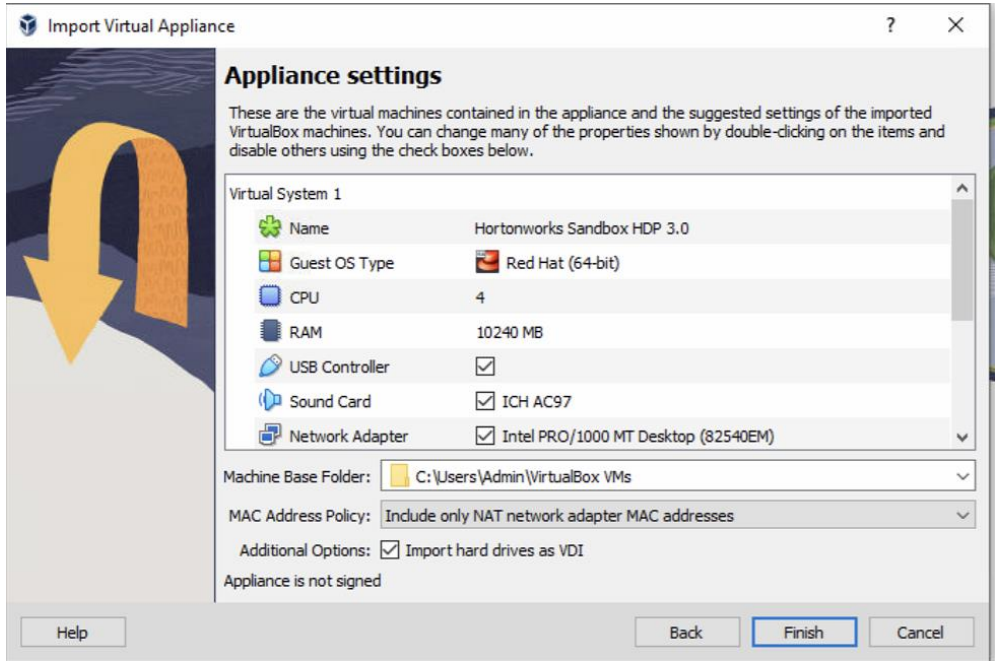


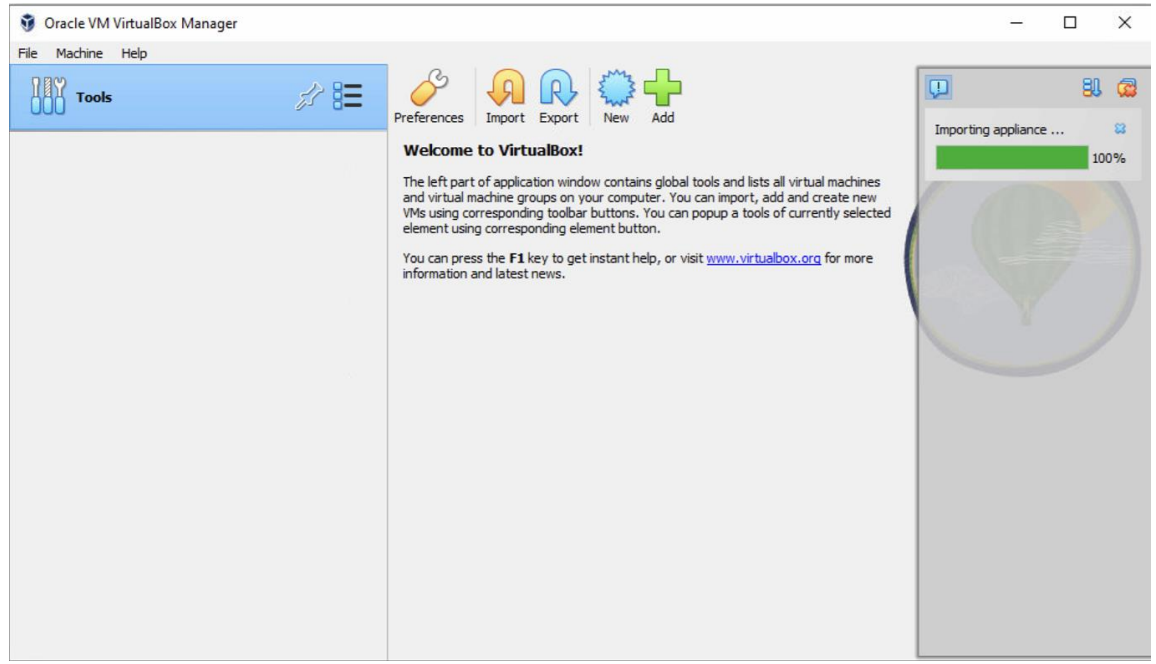




Have a quick look on Appliance Settings which would tell you that what is the guest OS on which you would be running Sandbox.

# Step 3: Prepare Hadoop Environment

Open Oracle VM VirtualBox: File ->Import Appliance and select the .ova file

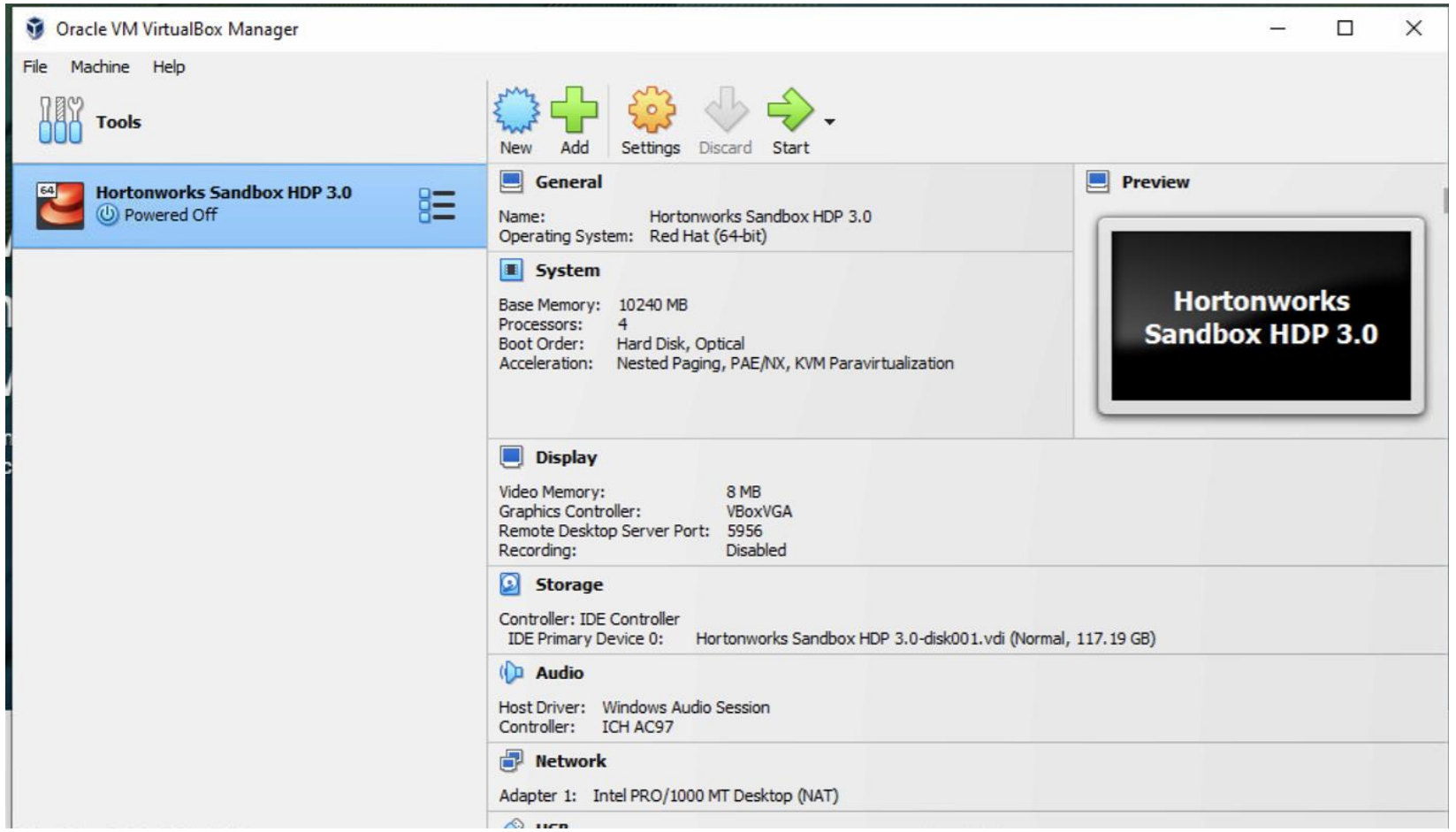



Have a quick look on Appliance Settings which would tell you that what is the guest OS on which you would be running Sandbox.

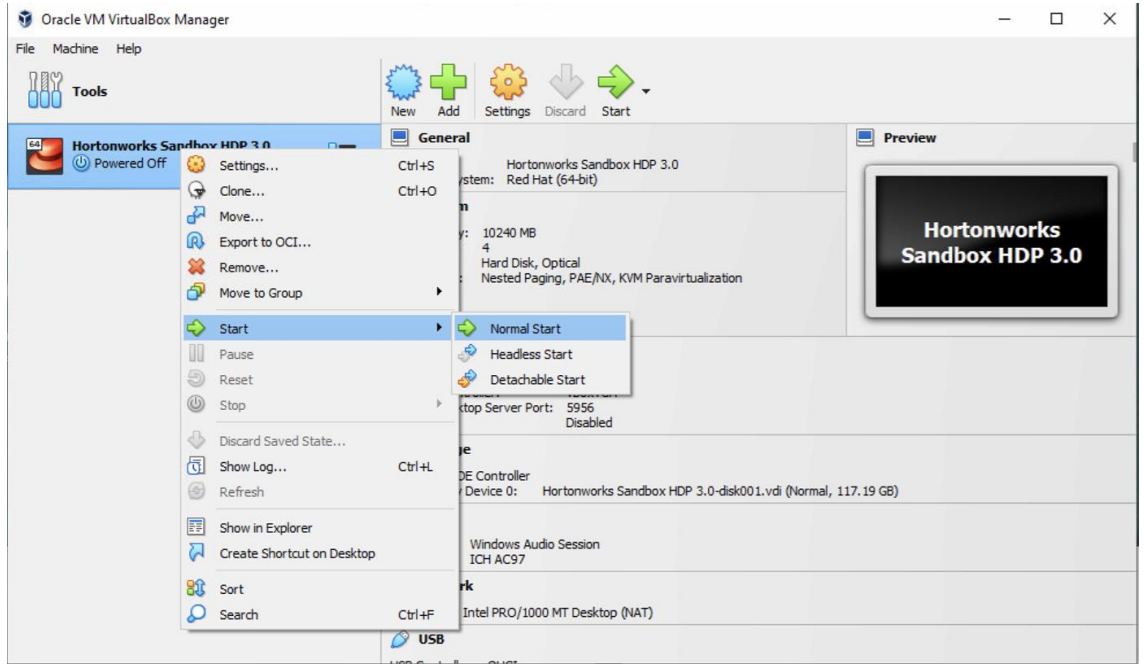
# Step 3: Prepare Hadoop Environment

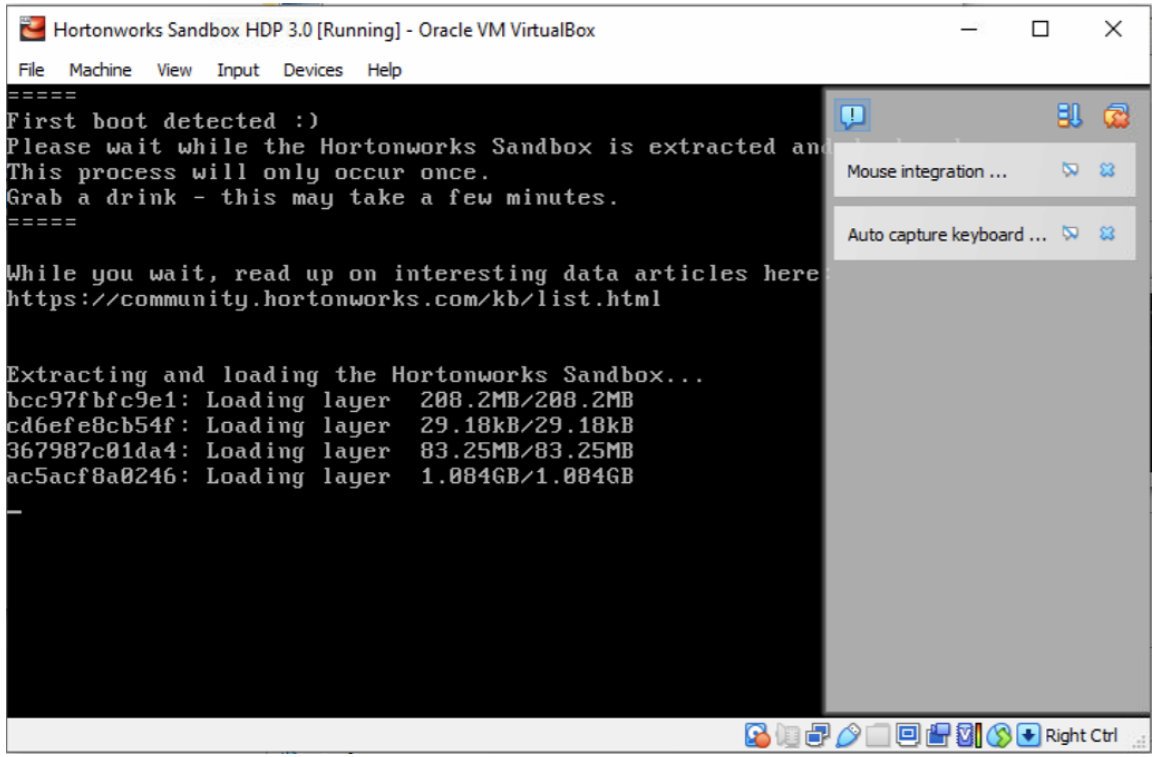
Open Oracle VM VirtualBox: File ->Import Appliance and select the .ova file



# Step 3: Prepare Hadoop Environment

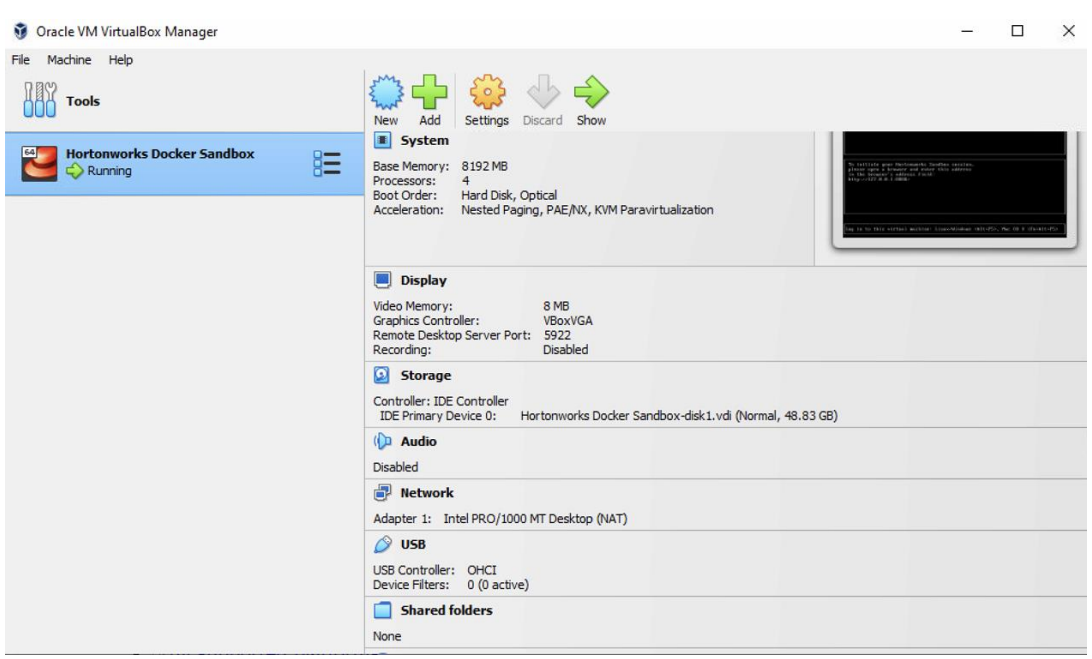
You will notice that virtual machine starts booting up and loading different configuration

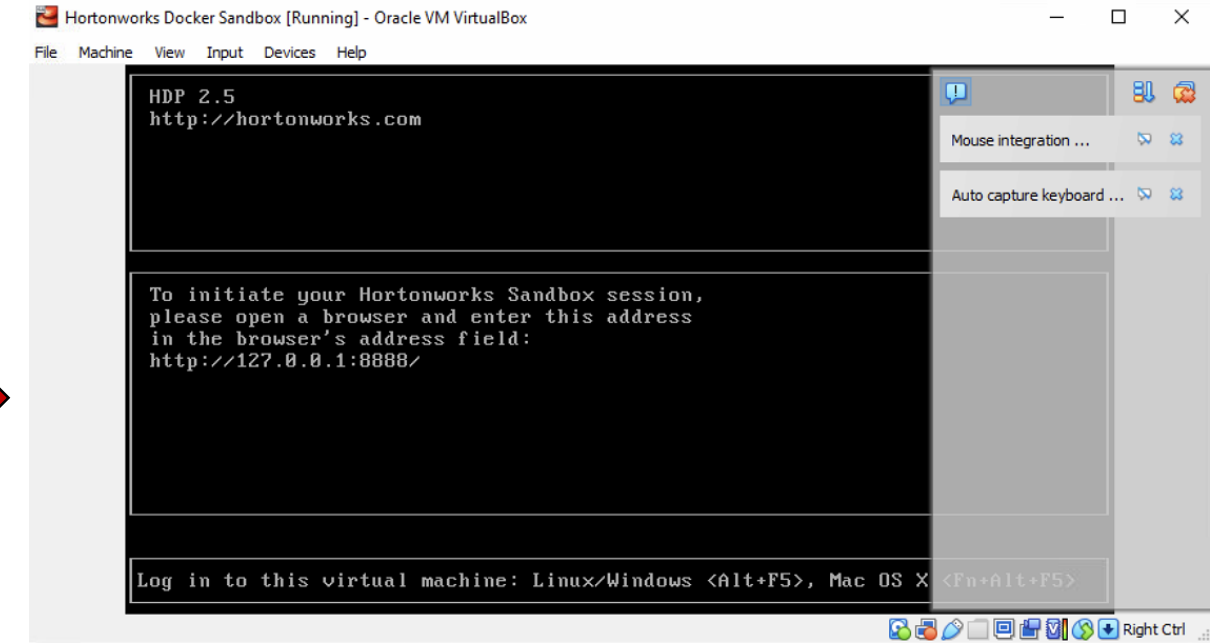


# Step 3: Prepare Hadoop Environment

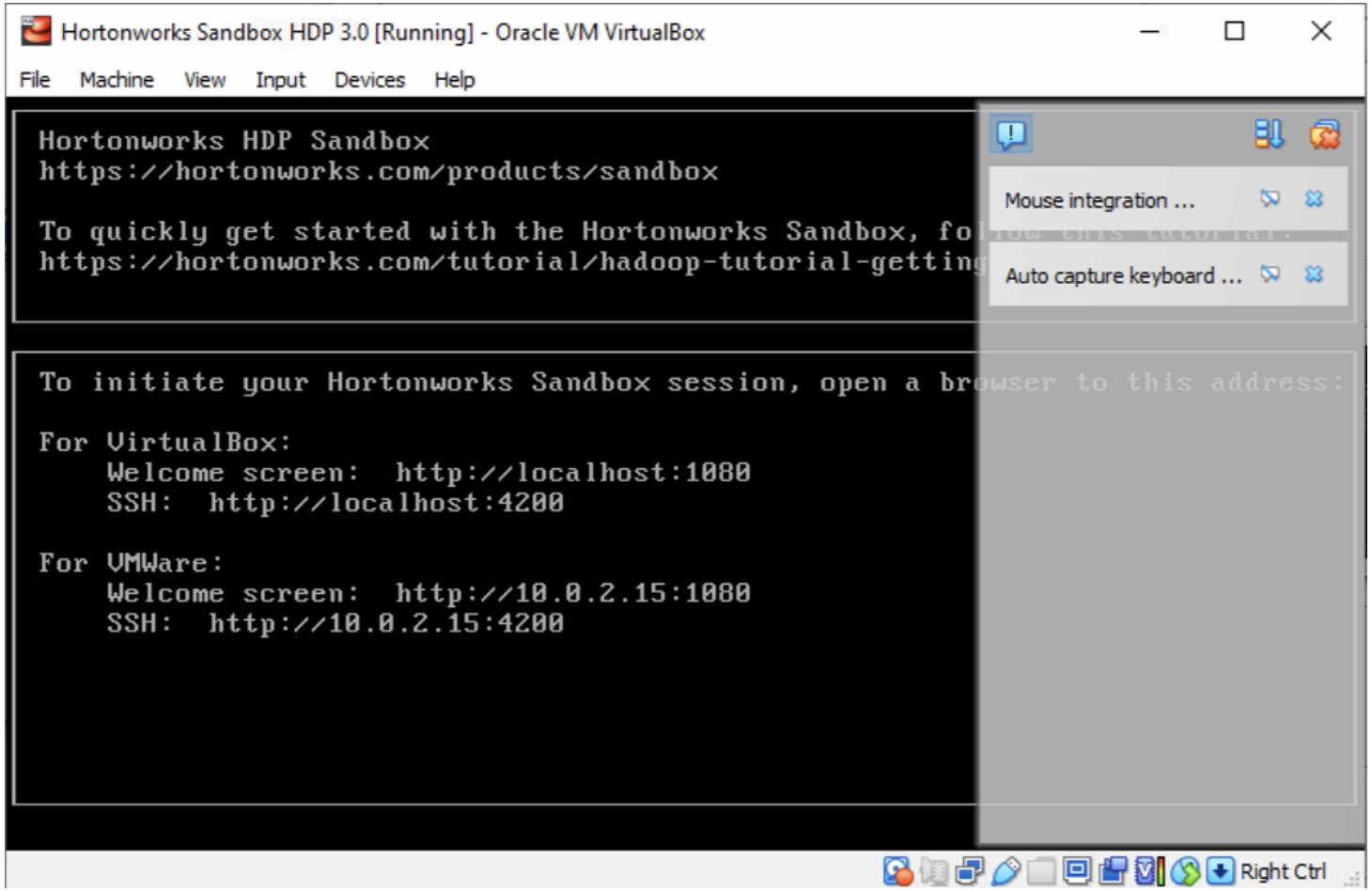
You will notice that virtual machine starts booting up and loading different configuration



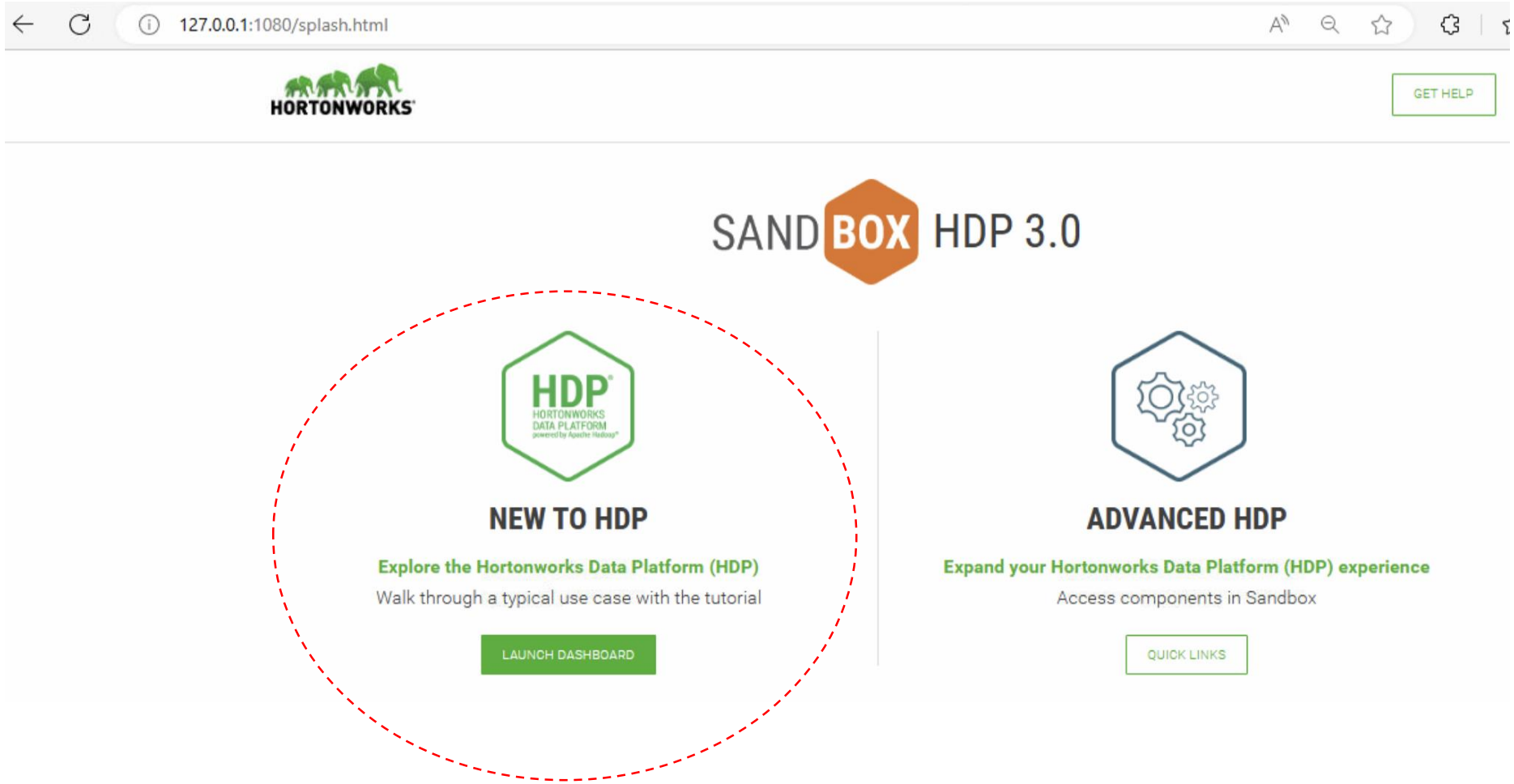
# Step 3: Prepare Hadoop Environment

Done!! We have successfully installed and configured HortonWorks SandBox



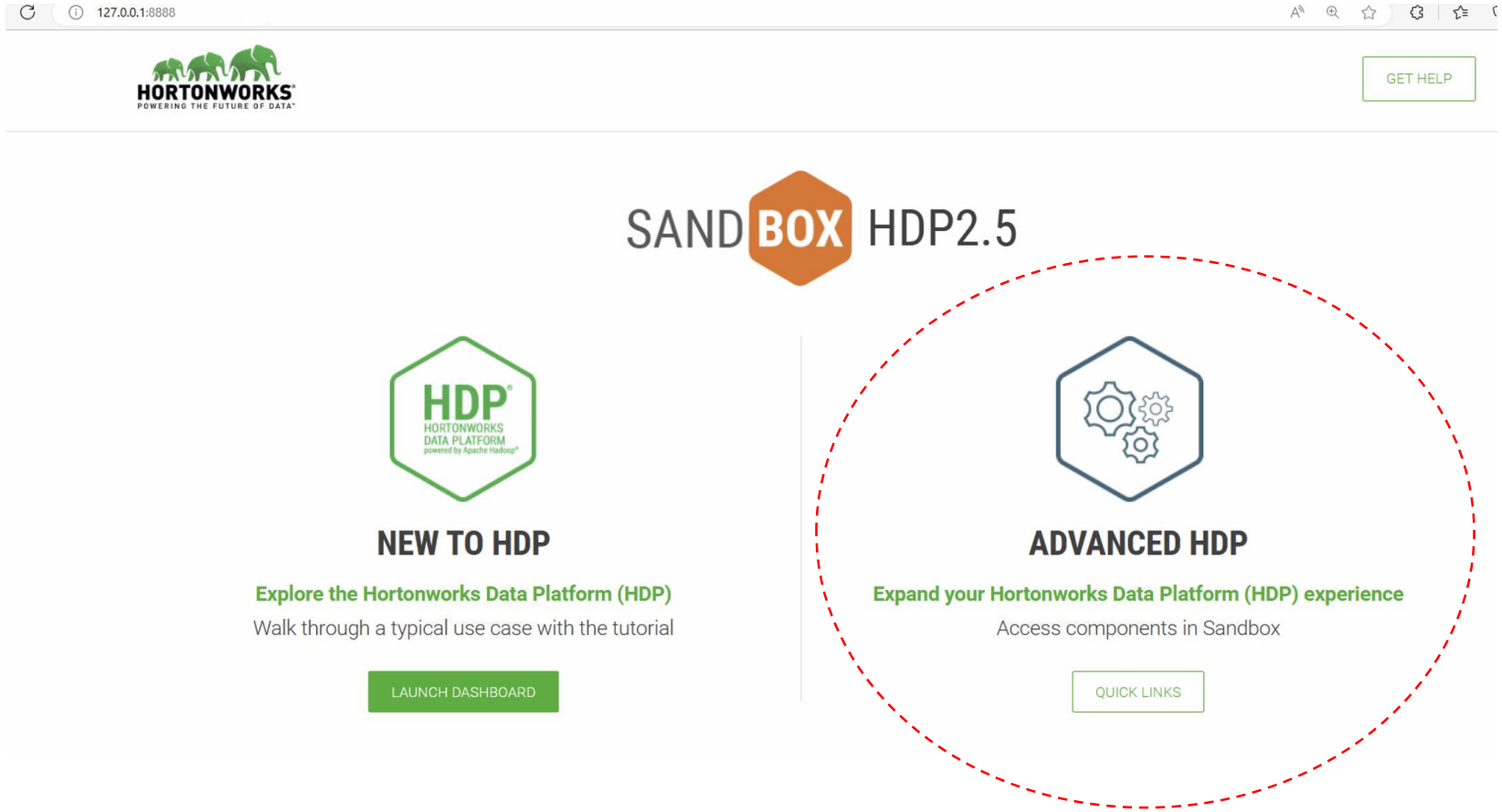
# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox

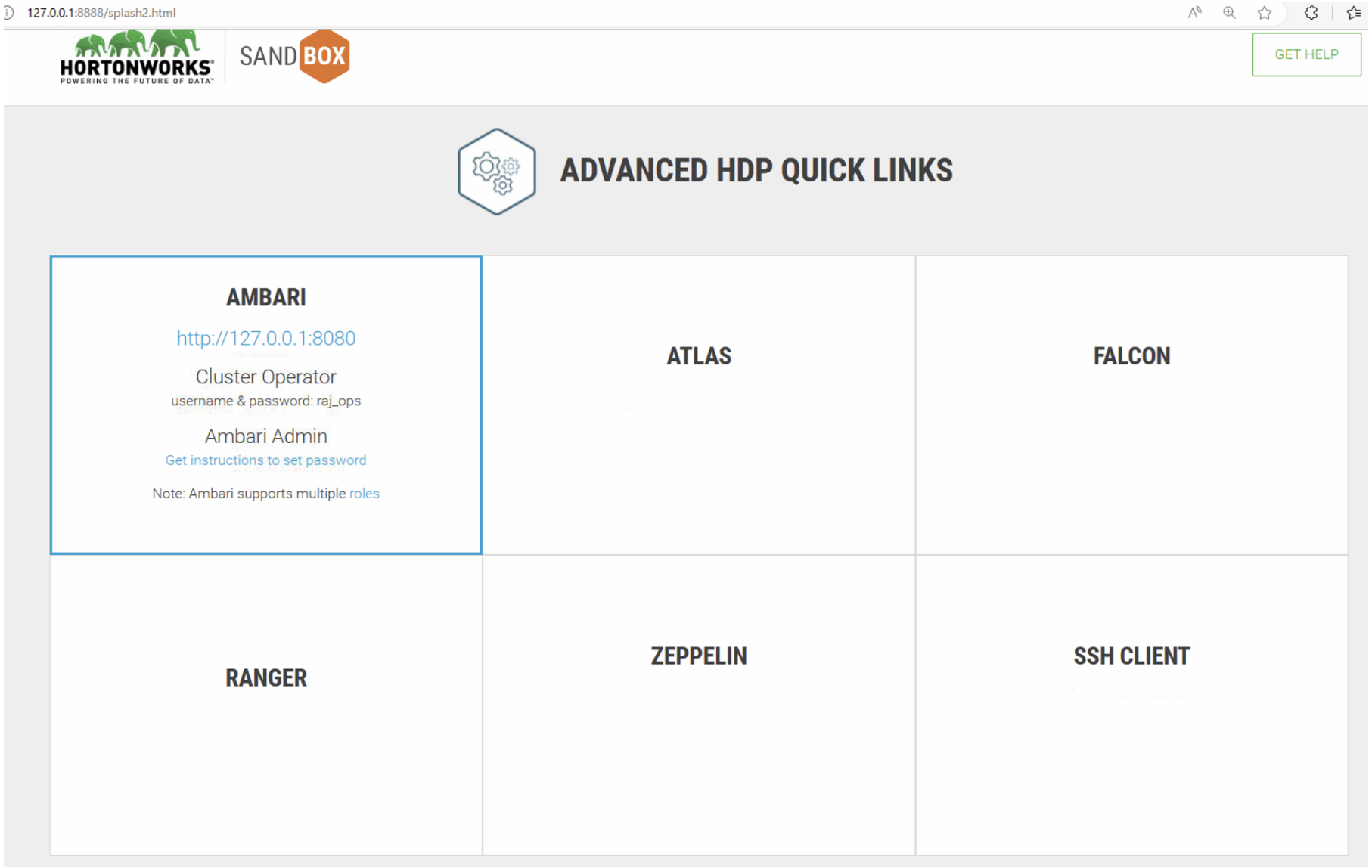


# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox

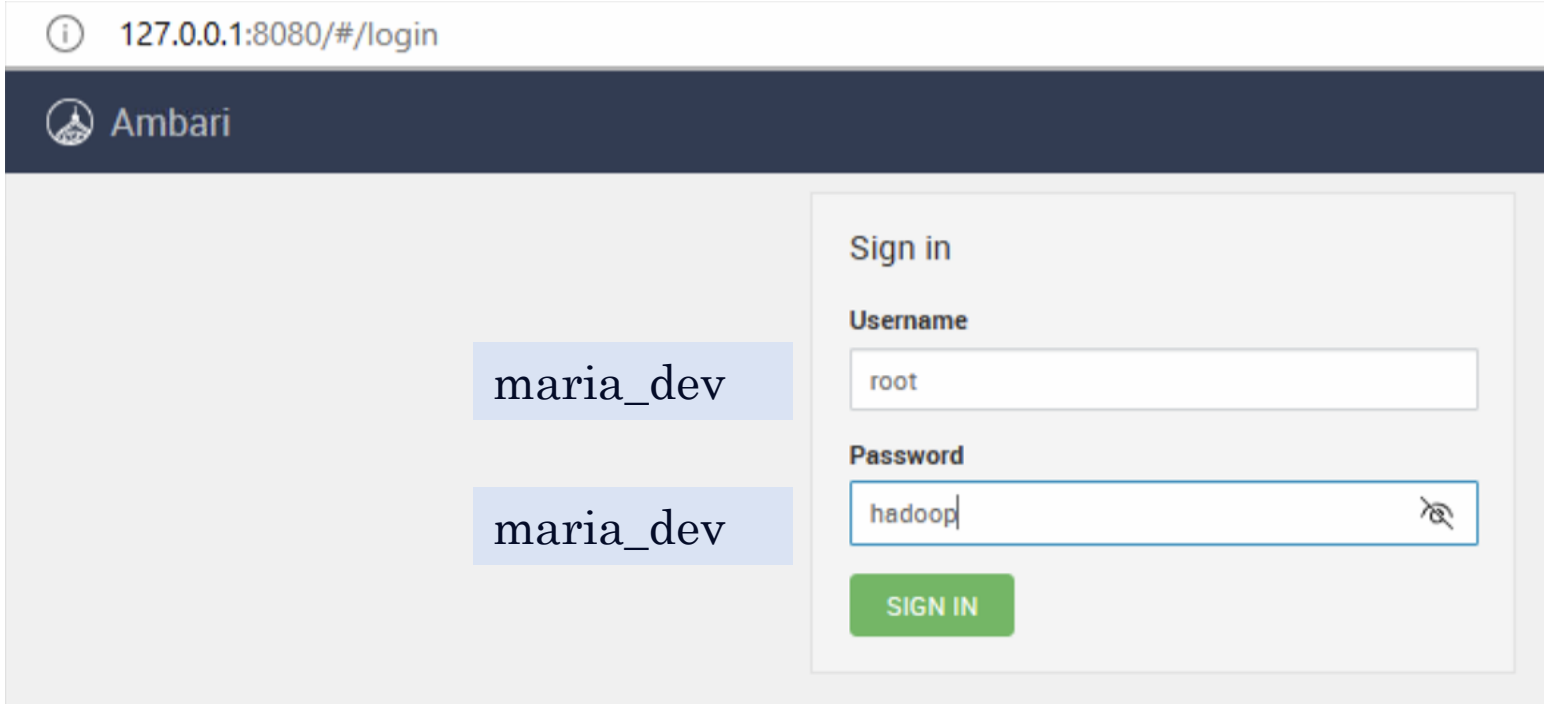


# Step 4: First glance at Hortonworks sandbox



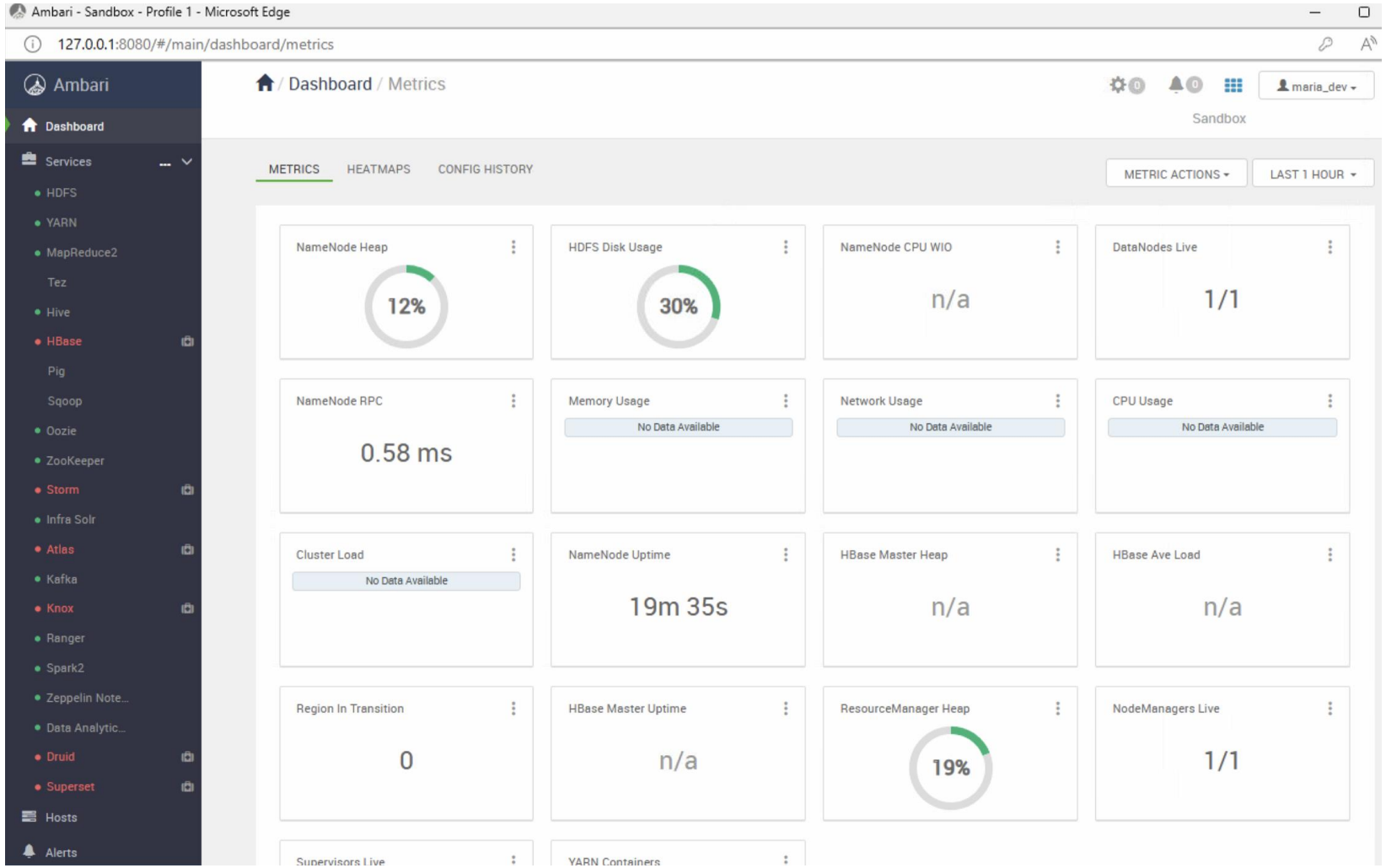
# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox



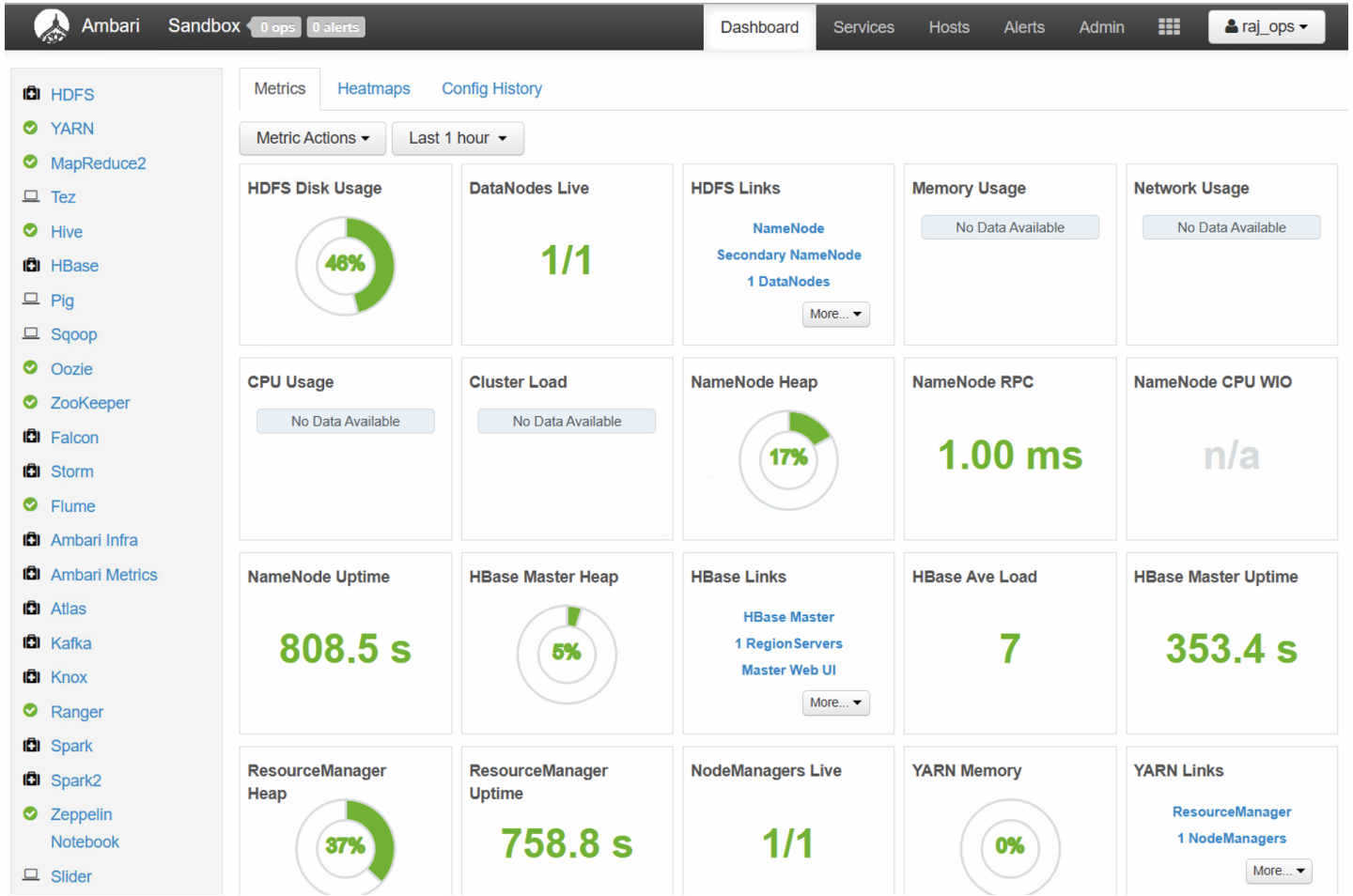
# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox



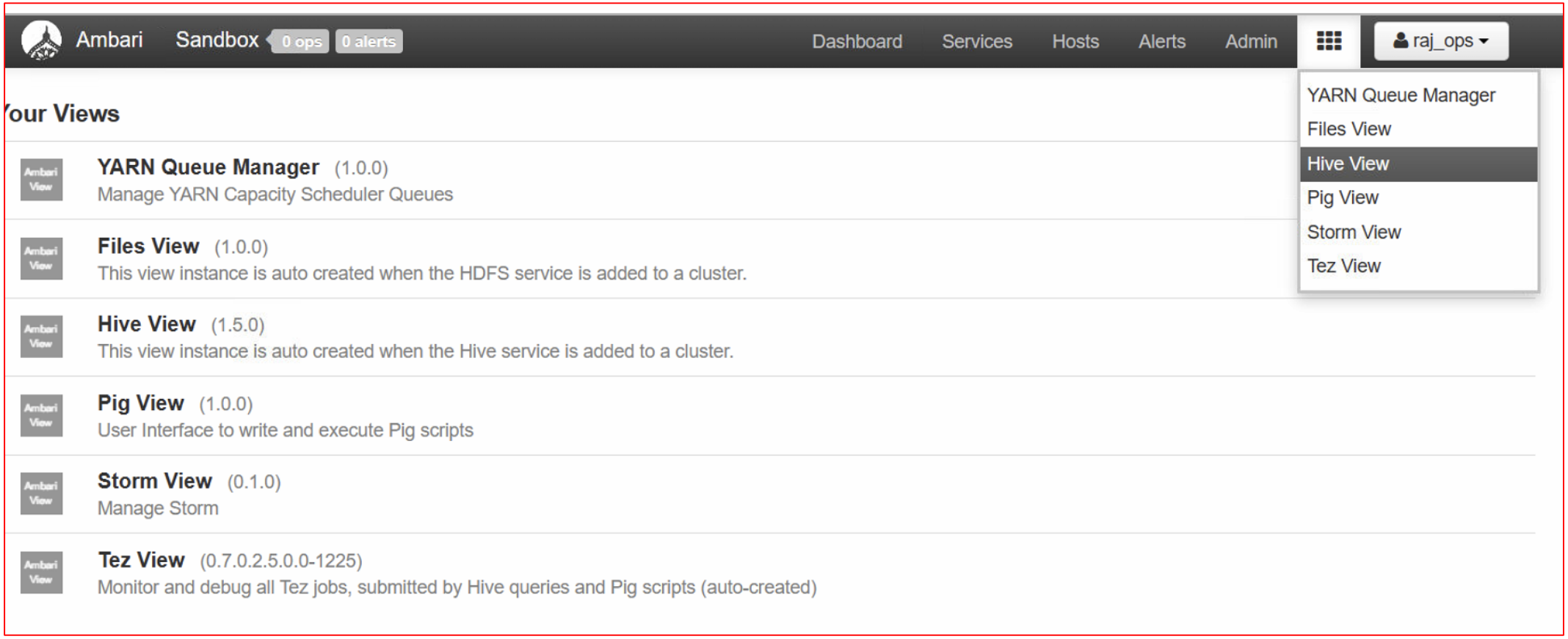
# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox



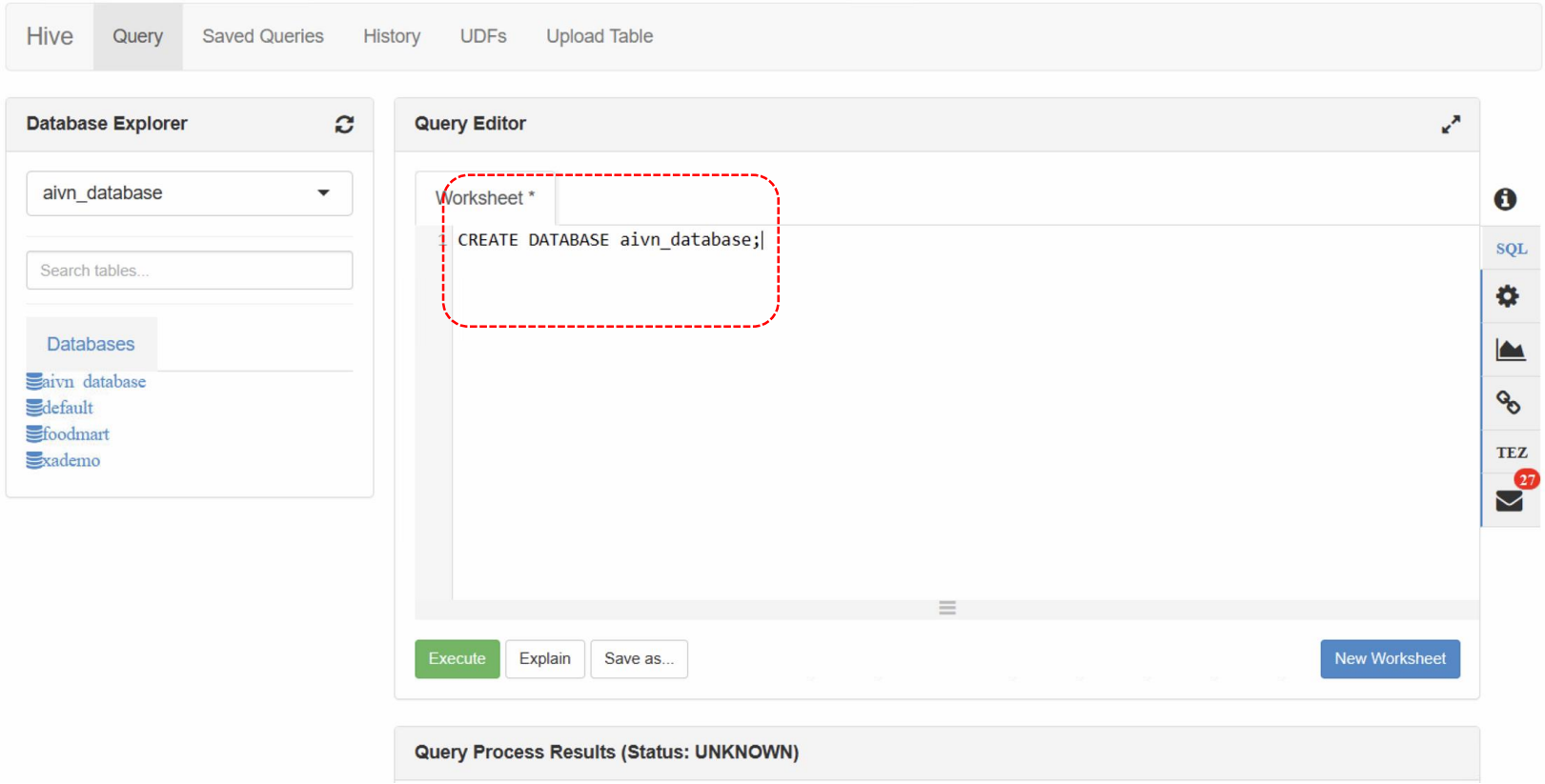
# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox



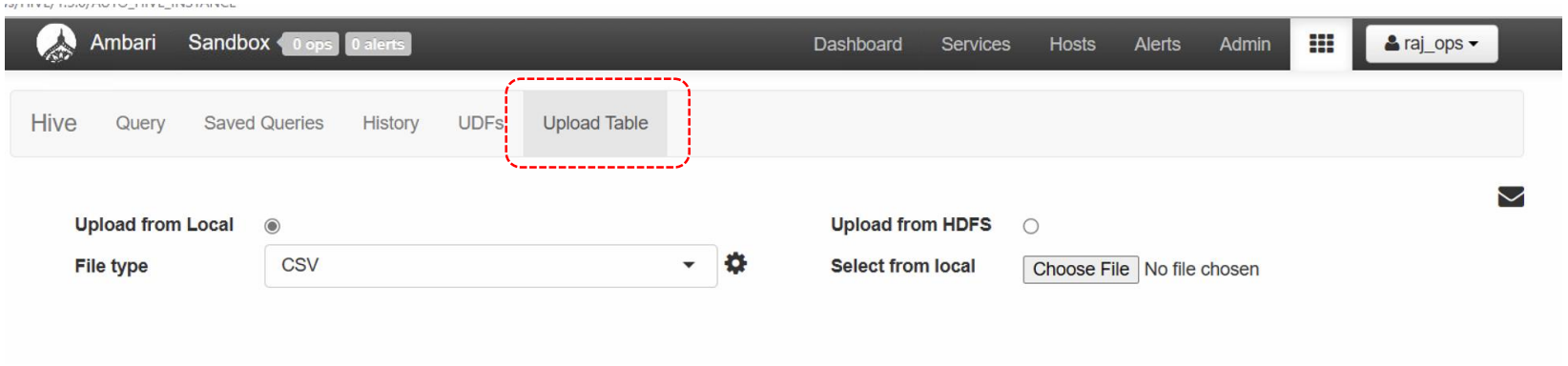
# Step 4: First glance at Hortonworks sandbox

## Create AIVN databaset



# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox

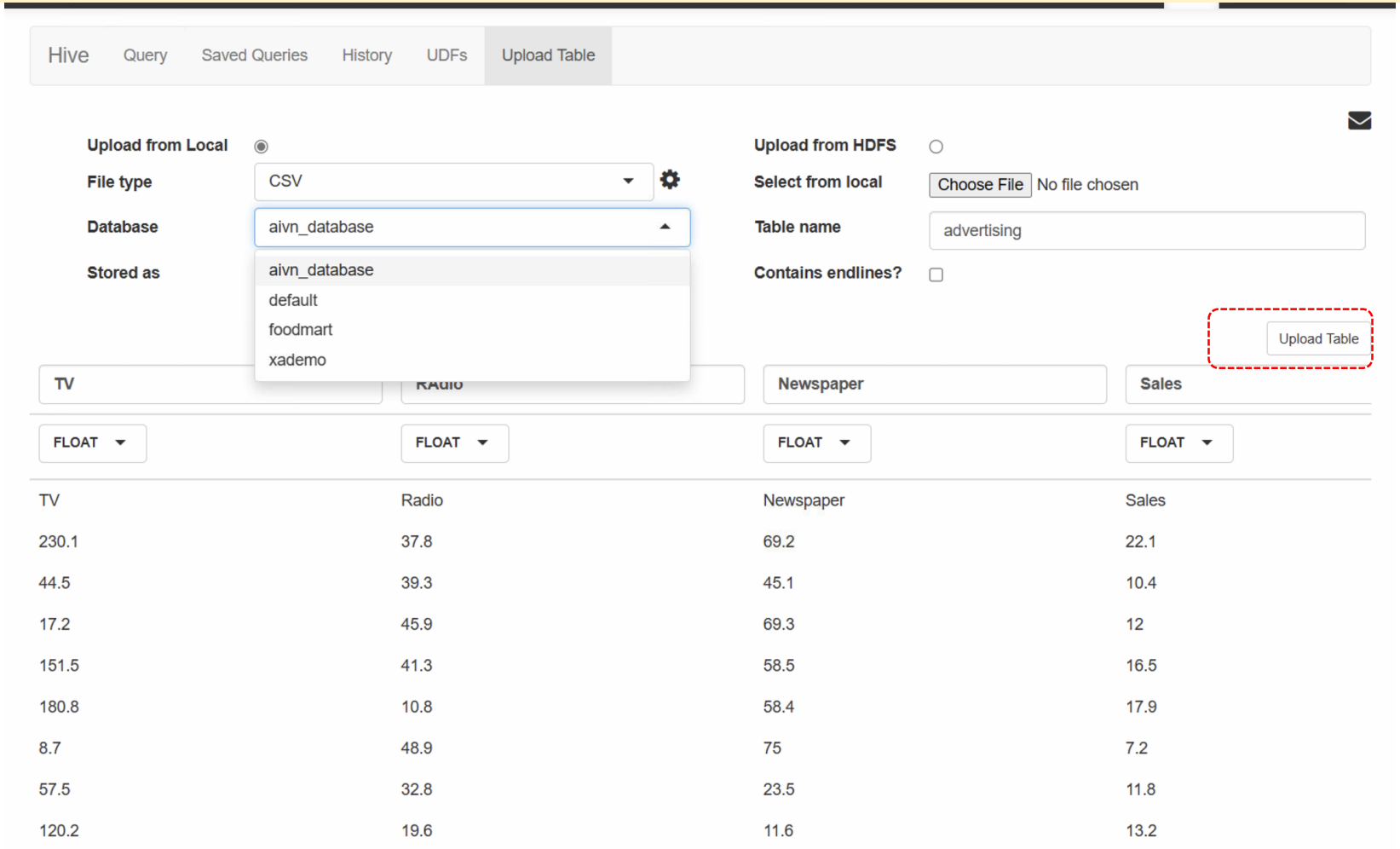


# Advertising Dataset

| TV | Radio | Newspaper | Sales |
|---|---|---|---|
| 230.1 | 37.8 | 69.2 | 22.1 |
| 44.5 | 39.3 | 45.1 | 10.4 |
| 17.2 | 45.9 | 69.3 | 12 |
| 151.5 | 41.3 | 58.5 | 16.5 |
| 180.8 | 10.8 | 58.4 | 17.9 |
| 8.7 | 48.9 | 75 | 7.2 |
| 57.5 | 32.8 | 23.5 | 11.8 |
| 120.2 | 19.6 | 11.6 | 13.2 |
| 8.6 | 2.1 | 1 | 4.8 |
| 199.8 | 2.6 | 21.2 | 15.6 |

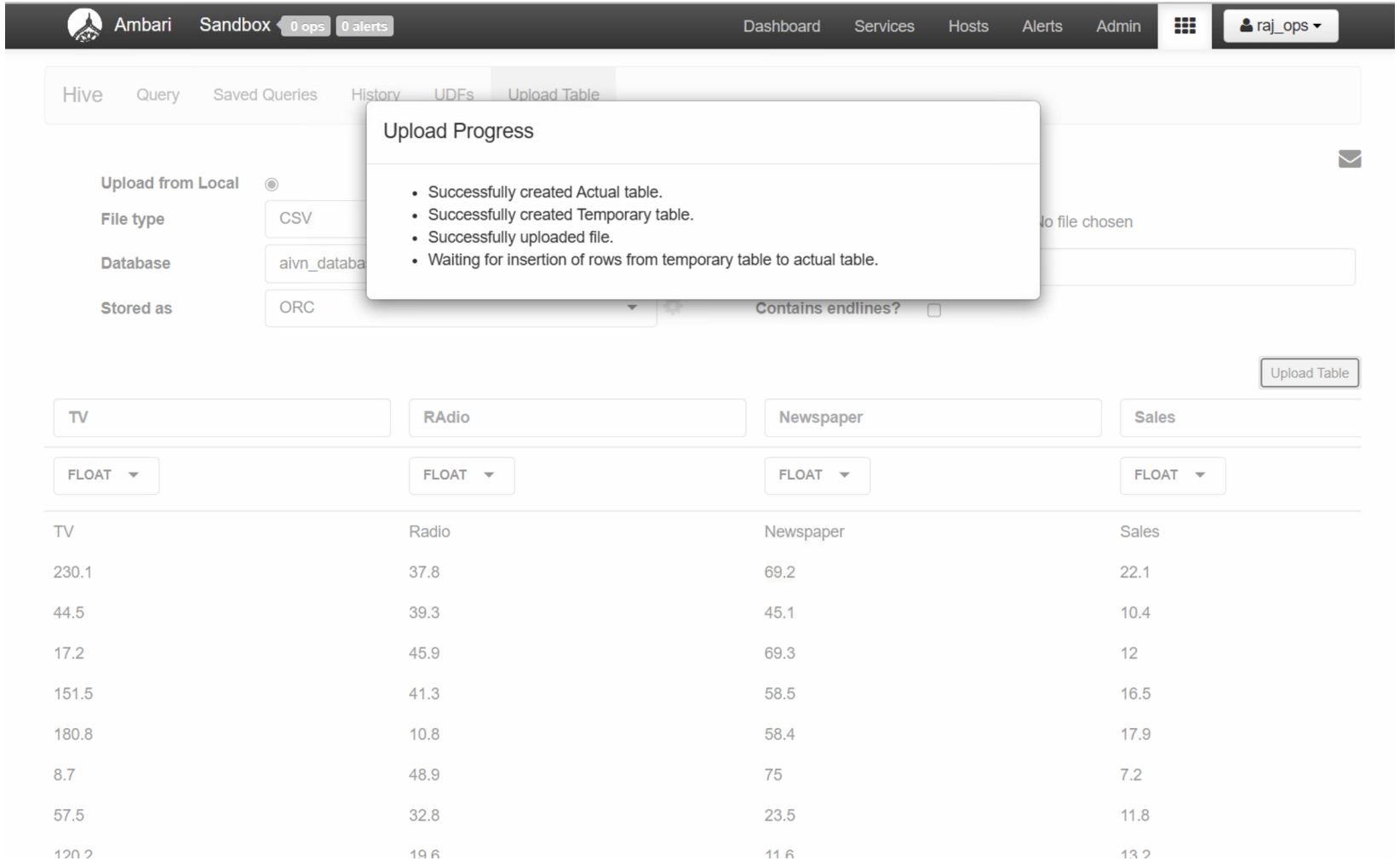
# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox



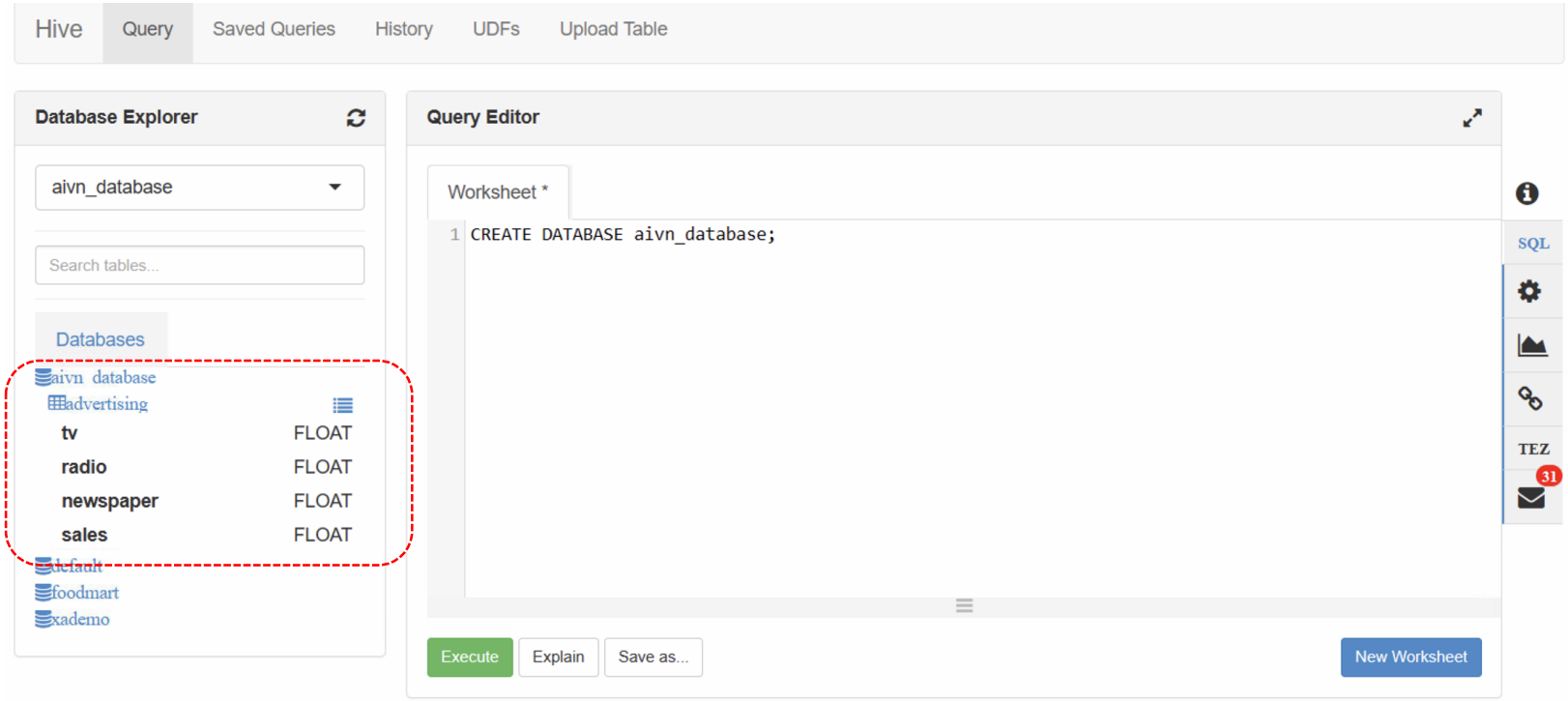
# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox

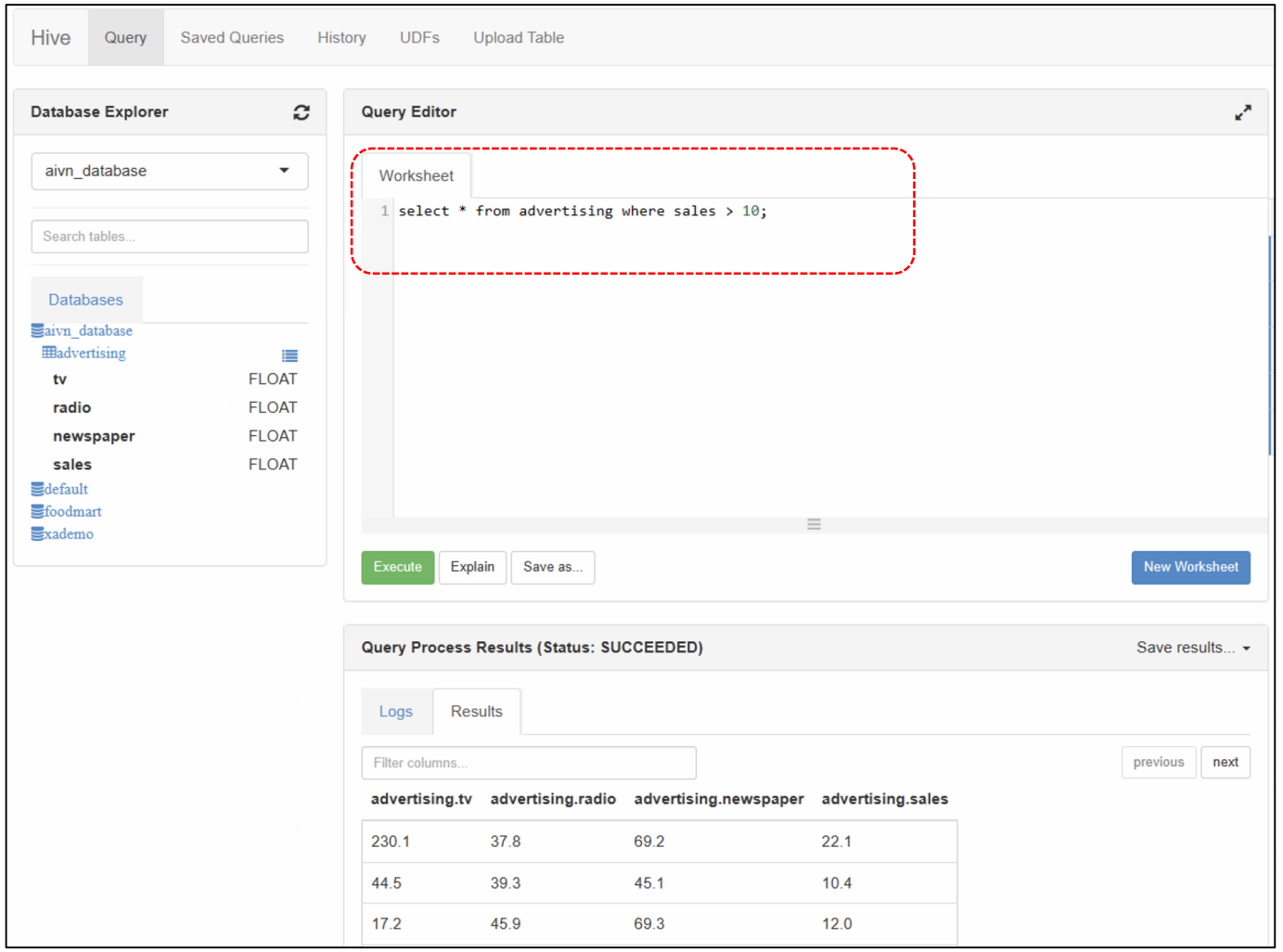


# Step 4: First glance at Hortonworks sandbox

Done!! We have successfully installed and configured HortonWorks SandBox

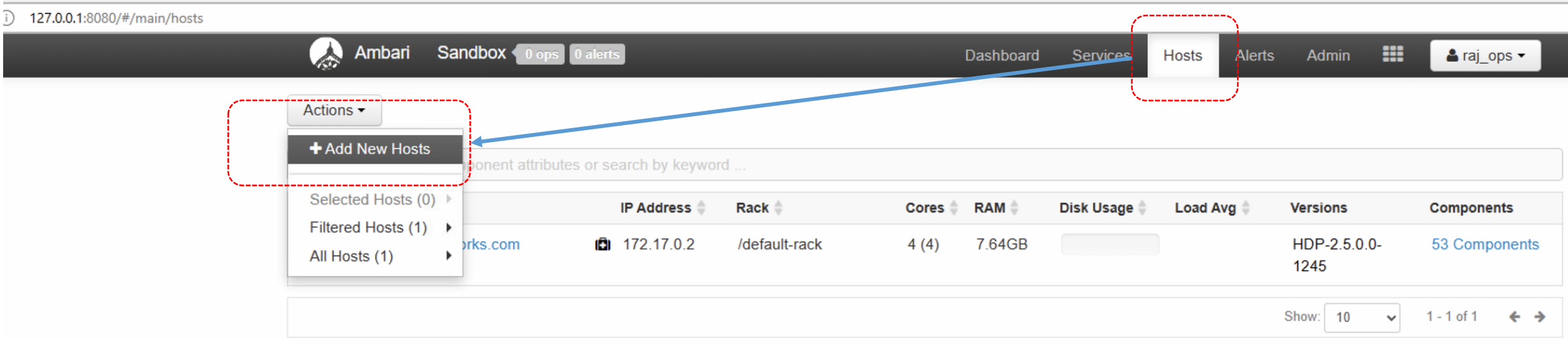


# Step 4: First glance at Hortonworks sandbox

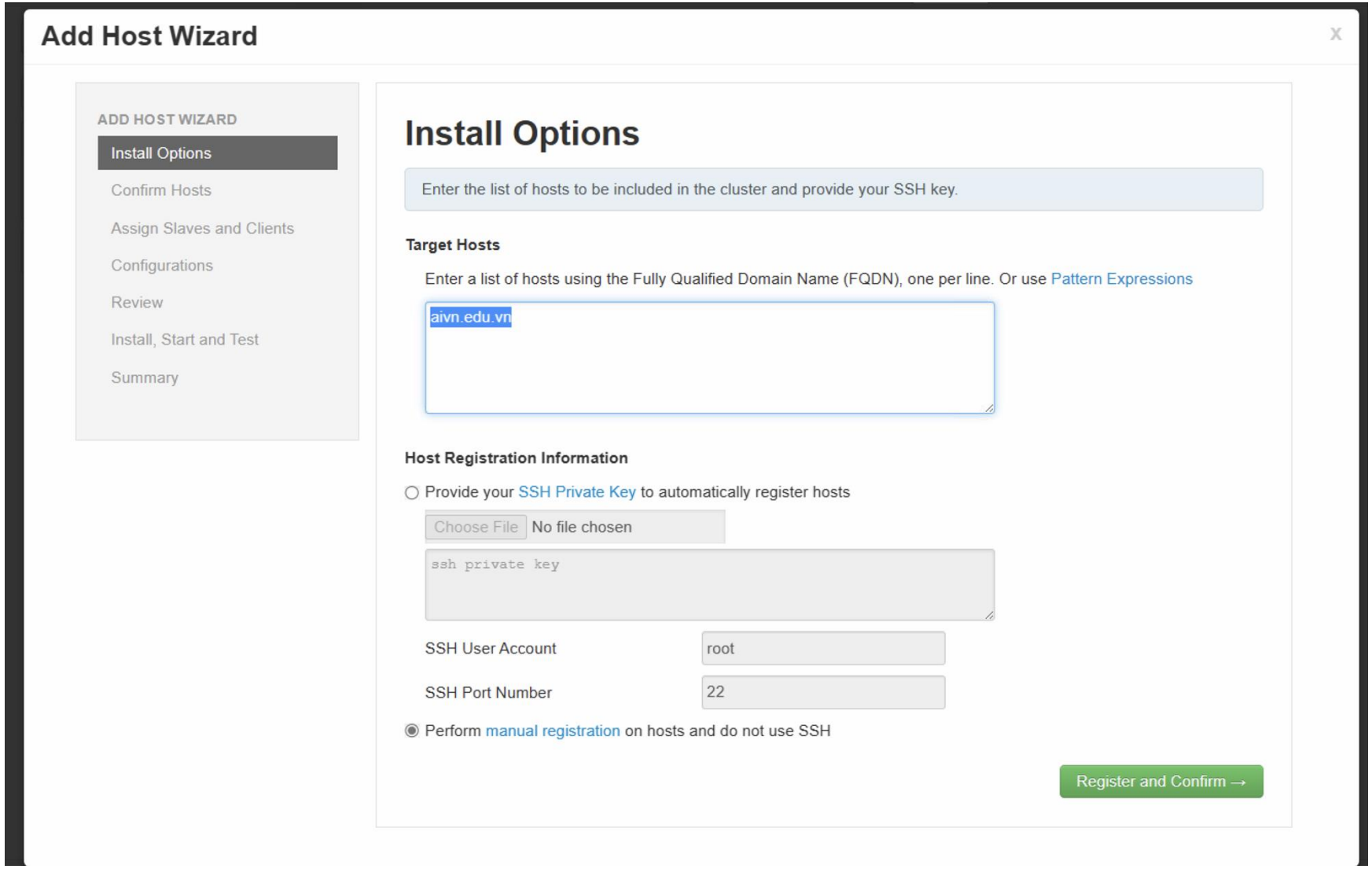


AI VIETNAM
All-in-One Course

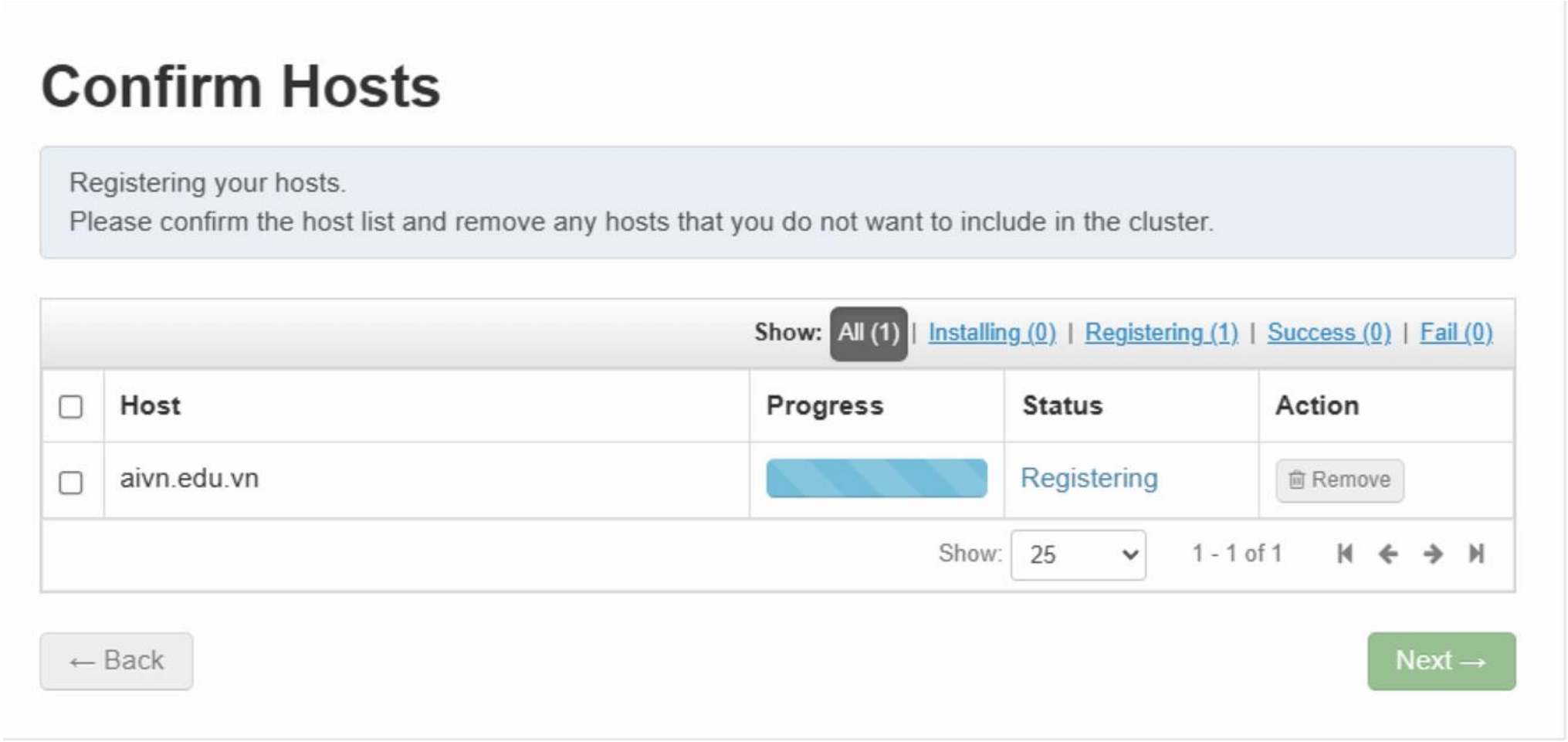
# Adding new DataNode to the cluster using Ambari



# Adding new DataNode to the cluster using Ambari



# Adding new DataNode to the cluster using Ambari



# Hadoop on CoLab: Step by Step

- •Installing Java 8
- •Installing Secure Shell Server (SSHD)
- •Installing Hadoop 3.2.3
- Running Hadoop in standalone mode
- Running Hadoop in Pseudo-distributed mode

# Installing Java 8

```
#Kiểm tra version java hiện tại
!java -version
```

```
openjdk version "11.0.20" 2023-07-18
OpenJDK Runtime Environment (build 11.0.20+8-post-Ubuntu-1ubuntu122.04)
OpenJDK 64-Bit Server VM (build 11.0.20+8-post-Ubuntu-1ubuntu122.04, mixed mode,
```

```
#Tiến hành cài đặt JDK 8
!apt-get install openjdk-8-jdk-headless -
qq > /dev/null
```

```
# Cấu hình sử dụng java 11
!update-alternatives --config java
```

```
There are 2 choices for the alternative java (providing /usr/bin/java).

  Selection    Path                                            Priority   Status
------------------------------------------------------------
  0            /usr/lib/jvm/java-11-openjdk-amd64/bin/java      1111      auto mode
  1            /usr/lib/jvm/java-11-openjdk-amd64/bin/java      1111      manual mode
* 2            /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java   1081      manual mode

Press <enter> to keep the current choice[*], or type selection number: 2
```

```
# cấu hình javac mặc định (chọn số 2)
!update-alternatives --config javac
```

```
There are 2 choices for the alternative javac (providing /usr/bin/javac).

  Selection    Path                                       Priority   Status
------------------------------------------------------------
* 0            /usr/lib/jvm/java-11-openjdk-amd64/bin/javac   1111      auto mode
  1            /usr/lib/jvm/java-11-openjdk-amd64/bin/javac   1111      manual mode
  2            /usr/lib/jvm/java-8-openjdk-amd64/bin/javac    1081      manual mode

Press <enter> to keep the current choice[*], or type selection number: 2
update-alternatives: using /usr/lib/jvm/java-8-openjdk-amd64/bin/javac to provide /usr/bin/javac (javac)
```

# Installing Secure Shell Server (SSHD)

```
#Installing openssh-server
!apt-get install openssh-server -qq >
/dev/null
```

```
#Starting the server
!service ssh start
```

```
# xem cấu hình port
!grep Port /etc/ssh/sshd_config
```

```
#Creating a new rsa key pair with empty
password
!ssh-keygen -t rsa -P "" -f ~/.ssh/id_rsa
```

```
Generating public/private rsa key pair.
Created directory '/root/.ssh'.
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:dz8zNm5GNTSWNEEWHQE3gNjnTzNF2QnuVCi7zUAihNs root@ccb14c0b25d0
The key's randomart image is:
+---[RSA 3072]----+
|      o.  o .+B#%|
|     . . o =.o+B*|
|      o . o =oo o|
|     . E   oo. =.|
|        S . *.o.+|
|         . o +.. |
|             .B  |
|             oo= |
|             o.  |
+----[SHA256]-----+
```

# Installing Hadoop 3.2.3

```
#Downloading Hadoop 3.2.3
!wget -q
https://archive.apache.org/dist/hadoop/comm
on/hadoop-3.2.3/hadoop-3.2.3.tar.gz
```

```
#Untarring the file
!sudo tar -xzf hadoop-3.2.3.tar.gz
#Removing the tar file
!rm hadoop-3.2.3.tar.gz
```

```
#Copying the hadoop files to
user/local
!cp -r hadoop-3.2.3/ /usr/local/
#-r copy directories recursively
```

```
#Exploring hadoop-3.2.3/etc/hadoop directory
!ls /usr/local/hadoop-3.2.3/etc/hadoop
#we can see various configuration files of
hadoop
```

```
capacity-scheduler.xml            kms-log4j.properties
configuration.xsl                 kms-site.xml
container-executor.cfg            log4j.properties
core-site.xml                     mapred-env.cmd
hadoop-env.cmd                    mapred-env.sh
hadoop-env.sh                     mapred-queues.xml.template
hadoop-metrics2.properties        mapred-site.xml
hadoop-policy.xml                 shellprofile.d
hadoop-user-functions.sh.example  ssl-client.xml.example
hdfs-site.xml                     ssl-server.xml.example
httpfs-env.sh                     user_ec_policies.xml.template
httpfs-log4j.properties           workers
httpfs-signature.secret           yarn-env.cmd
httpfs-site.xml                   yarn-env.sh
kms-acls.xml                      yarnservice-log4j.properties
kms-env.sh                        yarn-site.xml
```

# Running Hadoop in standalone mode

```
#Exploring mapreduce tools
!ls
$HADOOP_HOME/share/hadoop/mapreduce/*.jar
```

```
#Exploring the examples of programs
available
!$HADOOP_HOME/bin/hadoop jar
$HADOOP_HOME/share/hadoop/mapreduce/hadoop-
mapreduce-examples-3.2.3.jar
```

```
#Usage of the wordcount MapReduce program
!$HADOOP_HOME/bin/hadoop jar
$HADOOP_HOME/share/hadoop/mapreduce/hadoop-
mapreduce-examples-3.2.3.jar wordcount
```

```
/usr/local/hadoop-3.2.3/share/hadoop/mapreduce/hadoop-mapreduce-client-app-3.2.3.jar
/usr/local/hadoop-3.2.3/share/hadoop/mapreduce/hadoop-mapreduce-client-common-3.2.3.jar
/usr/local/hadoop-3.2.3/share/hadoop/mapreduce/hadoop-mapreduce-client-core-3.2.3.jar
/usr/local/hadoop-3.2.3/share/hadoop/mapreduce/hadoop-mapreduce-client-hs-3.2.3.jar
```

```
An example program must be given as the first argument.
Valid program names are:
  aggregatewordcount: An Aggregate based map/reduce program that counts t
  aggregatewordhist: An Aggregate based map/reduce program that computes
  bbp: A map/reduce program that uses Bailey-Borwein-Plouffe to compute e
  dbcount: An example job that count the pageview counts from a database.
  distbbp: A map/reduce program that uses a BBP-type formula to compute e
  grep: A map/reduce program that counts the matches of a regex in the in
  join: A job that effects a join over sorted, equally partitioned datase
  multifilewc: A job that counts words from several files.
  pentomino: A map/reduce tile laying program to find solutions to pentom
  pi: A map/reduce program that estimates Pi using a quasi-Monte Carlo me
  randomtextwriter: A map/reduce program that writes 10GB of random textu
  randomwriter: A map/reduce program that writes 10GB of random data per
  secondarysort: An example defining a secondary sort to the reduce.
  sort: A map/reduce program that sorts the data written by the random wr
  sudoku: A sudoku solver.
  teragen: Generate data for the terasort
  terasort: Run the terasort
  teravalidate: Checking results of terasort
  wordcount: A map/reduce program that counts the words in the input file
  wordmean: A map/reduce program that counts the average length of the wo
  wordmedian: A map/reduce program that counts the median length of the w
  wordstandarddeviation: A map/reduce program that counts the standard de
```

# Running Hadoop in standalone mode

```
#Running MapReduce program wordcount
#the output directory will be created
automatically
!$HADOOP_HOME/bin/hadoop jar
$HADOOP_HOME/share/hadoop/mapreduce/hadoop-
mapreduce-examples-3.2.3.jar wordcount
/content/101.txt /content/output
```

```
2023-08-04 12:21:33,098 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2023-08-04 12:21:33,229 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2023-08-04 12:21:33,229 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2023-08-04 12:21:33,423 INFO input.FileInputFormat: Total input files to process : 1
2023-08-04 12:21:33,474 INFO mapreduce.JobSubmitter: number of splits:1
2023-08-04 12:21:33,674 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1552352151_0001
2023-08-04 12:21:33,674 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-08-04 12:21:33,923 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2023-08-04 12:21:33,924 INFO mapreduce.Job: Running job: job_local1552352151_0001
```

# Running Hadoop in Pseudo-distributed mode

```
#Adding required property to core-site.xlm file
!sed -i '/<configuration>/a\
<property>\n\
<name>fs.defaultFS</name>\n\
<value>hdfs://localhost:9000</value>\n\
</property>' \
$HADOOP_HOME/etc/hadoop/core-site.xml
```

```
<configuration>
<property>
     <name>fs.defaultFS</name>
     <value>hdfs://localhost:9000</value>
   </property>
</configuration>
```

```
#Adding required property to hdfs-site.xml file
#Since we are running Hadoop in only one machine,
a replication factor greater than 1 does not make
sense
!sed -i '/<configuration>/a\
<property>\n\
<name>dfs.replication</name>\n\
<value>1</value>\n\
</property>' \
$HADOOP_HOME/etc/hadoop/hdfs-site.xml
```

```
<configuration>
<property>
     <name>dfs.replication</name>
     <value>1</value>
   </property>

</configuration>
```

# Running Hadoop in Pseudo-distributed mode

```
#Adding required properties to mapred-site.xml file
!sed -i '/<configuration>/a\
<property>\n\
<name>mapreduce.framework.name</name>\n\
<value>yarn</value>\n\
</property>\n\
<property>\n\
<name>mapreduce.application.classpath</name>\n\
<value>$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*:$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/li
b/*</value>\n\
</property>' \
$HADOOP_HOME/etc/hadoop/mapred-site.xml
```

```
<configuration>
<property>
     <name>mapreduce.framework.name</name>
     <value>yarn</value>
   </property>
   <property>
     <name>mapreduce.application.classpath</name>
     <value>$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*:$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/lib/*</value>
   </property>

</configuration>
```

# Running Hadoop in Pseudo-distributed mode

```
#Adding required properties to yarn-site.xml file
!sed -i '/<configuration>/a\
<property>\n\
<description>The hostname of the RM.</description>\n\
<name>yarn.resourcemanager.hostname</name>\n\
<value>localhost</value>\n\
</property>\n\
<property>\n\
<name>yarn.nodemanager.aux-services</name>\n\
<value>mapreduce_shuffle</value>\n\
</property>\n\
<property>\n\
<name>yarn.nodemanager.env-whitelist</name>\n\
<value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PREPEND_DISTCACHE,HADOOP
_YARN_HOME,HADOOP_
</property>' \
$HADOOP_HOME/etc/h
```

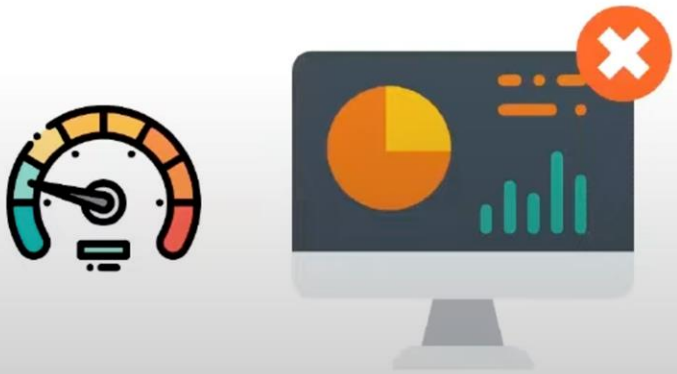
```
<configuration>
<property>
     <description>The hostname of the RM.</description>
     <name>yarn.resourcemanager.hostname</name>
     <value>localhost</value>
   </property>
   <property>
     <name>yarn.nodemanager.aux-services</name>
     <value>mapreduce_shuffle</value>
   </property>
   <property>
     <name>yarn.nodemanager.env-whitelist</name>
     <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PREPEND_DISTCACHE,HADOOP_YARN_
   </property>

<!-- Site specific YARN configuration properties -->

</configuration>
```

# Hadoop vs. Spark

## Performance

hadoop

Hadoop is generally slow as it performs operations on the disk and cannot deliver near real-time analytics from the data



No real-time analytics

APACHE Spark

Spark runs 100 times faster in-memory, and 10 times faster on disk. If Spark runs on YARN with other resources demanding services, there could be major degradation

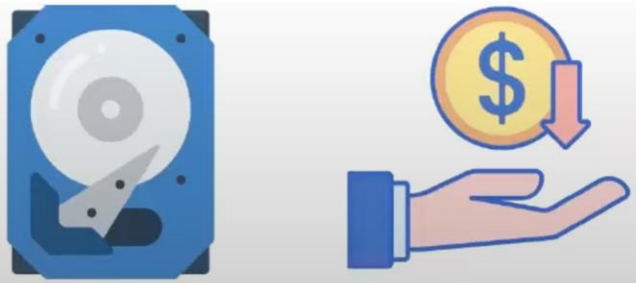
Faster in-memory processing

# Hadoop vs. Spark

## Cost

hadoop

Hadoop is less expensive as it is an open-source software. It requires more memory on disk which is relatively an inexpensive commodity



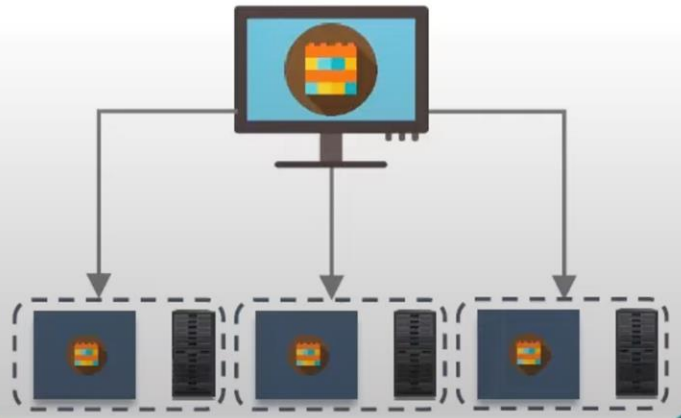
APACHE Spark

Spark is open-source but requires a lot of RAM to run in-memory. This increases the cluster size and its cost

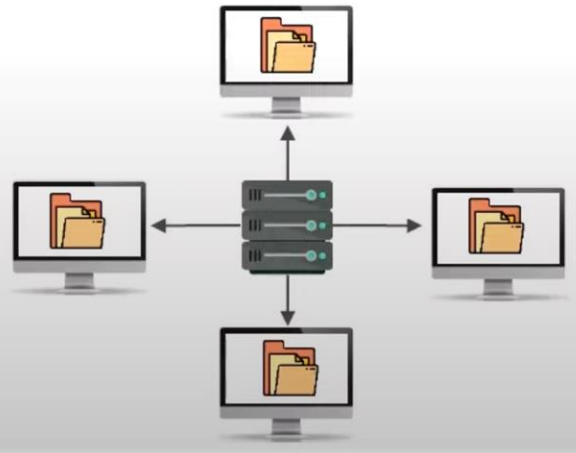
# Hadoop vs. Spark

## Fault Tolerance

Hadoop is highly fault-tolerant because it was designed to replicate data across many nodes. Each file is split into blocks and replicated numerous times across many machines



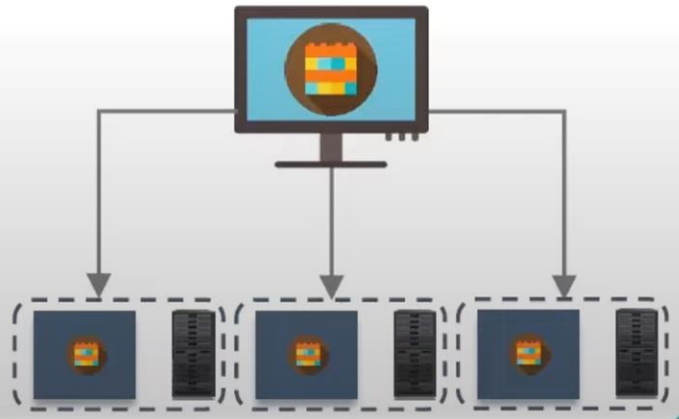
Spark uses Resilient Distributed Datasets (RDDs), which are fault-tolerant collections of elements that can be operated on in parallel



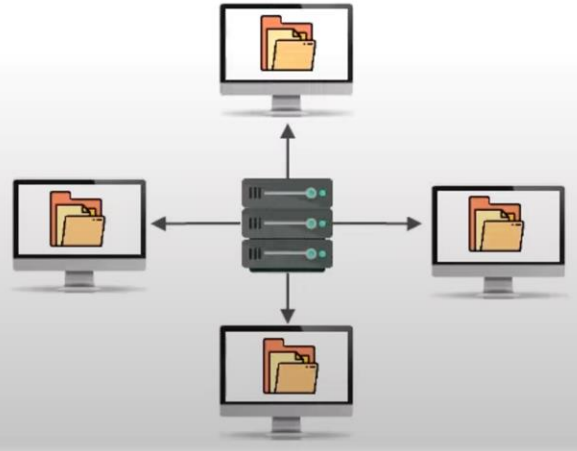
# Hadoop vs. Spark

## Fault Tolerance


Hadoop is highly fault-tolerant because it was designed to replicate data across many nodes. Each file is split into blocks and replicated numerous times across many machines



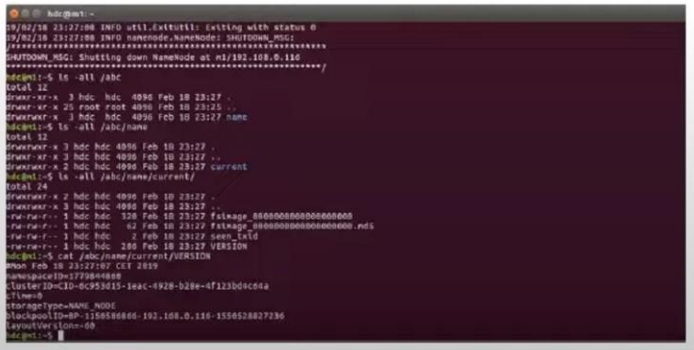

Spark uses Resilient Distributed Datasets (RDDs), which are fault-tolerant collections of elements that can be operated on in parallel



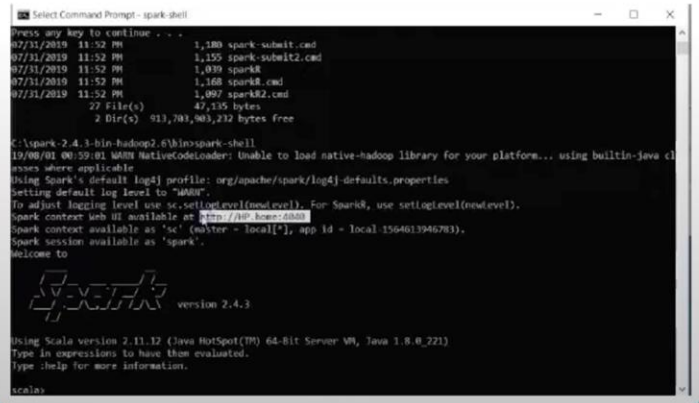
# Hadoop vs. Spark

## Ease of Use

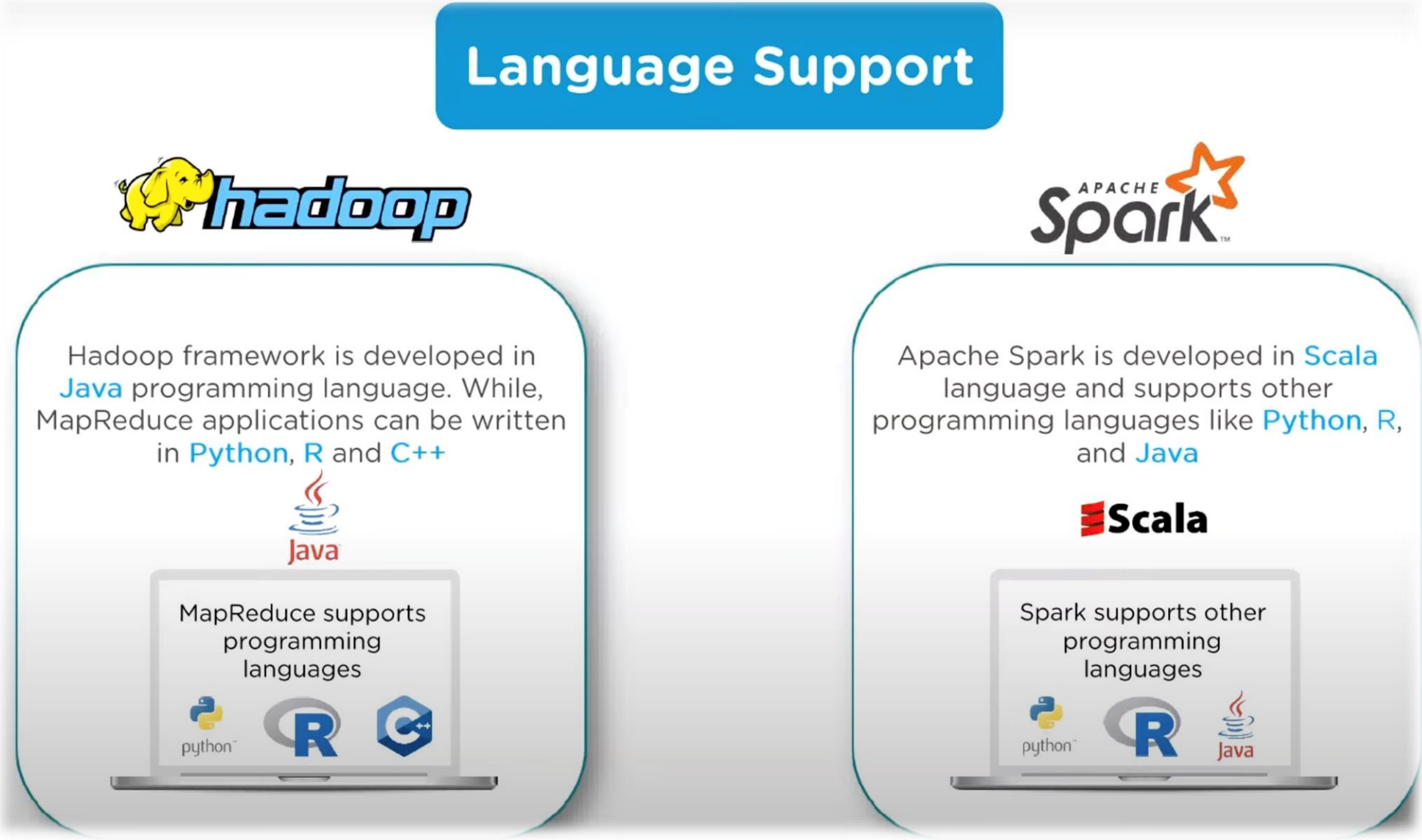

Hadoop's MapReduce has **no interactive mode** and is complex. It needs to handle low-level APIs to process the data, which requires lots of coding




Spark supports **user-friendly APIs** for different languages. It has an **interactive mode** and provides intermediate feedback for queries and actions



# Hadoop vs. Spark

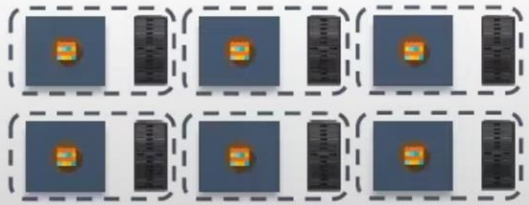


# Hadoop vs. Spark

## Scalability


Hadoop is highly scalable as we can add n number of nodes in the cluster. Yahoo reportedly used a 42,000 node Hadoop cluster


The largest known Spark cluster has 8,000 nodes. But as big data grows, it's expected that cluster sizes will increase to maintain throughput expectations.



# Hadoop vs. Spark

## Security


Hadoop supports Kerberos and LDAP for authentication. It also supports access control lists (ACLs) and a traditional file permissions model


Spark's security is a bit sparse as it supports authentication via passwords. If you run Spark on HDFS, it can use HDFS ACLs and file-level permissions. Additionally, Spark can run on YARN, giving it the capability of using Kerberos authentication.


# Hadoop vs. Spark

## Machine Learning

hadoop

Hadoop uses Mahout for processing data and building models. Also, Samsara, a Scala-backed DSL language can be used for in-memory algebraic operations and allows users to write their own algorithms

mahout

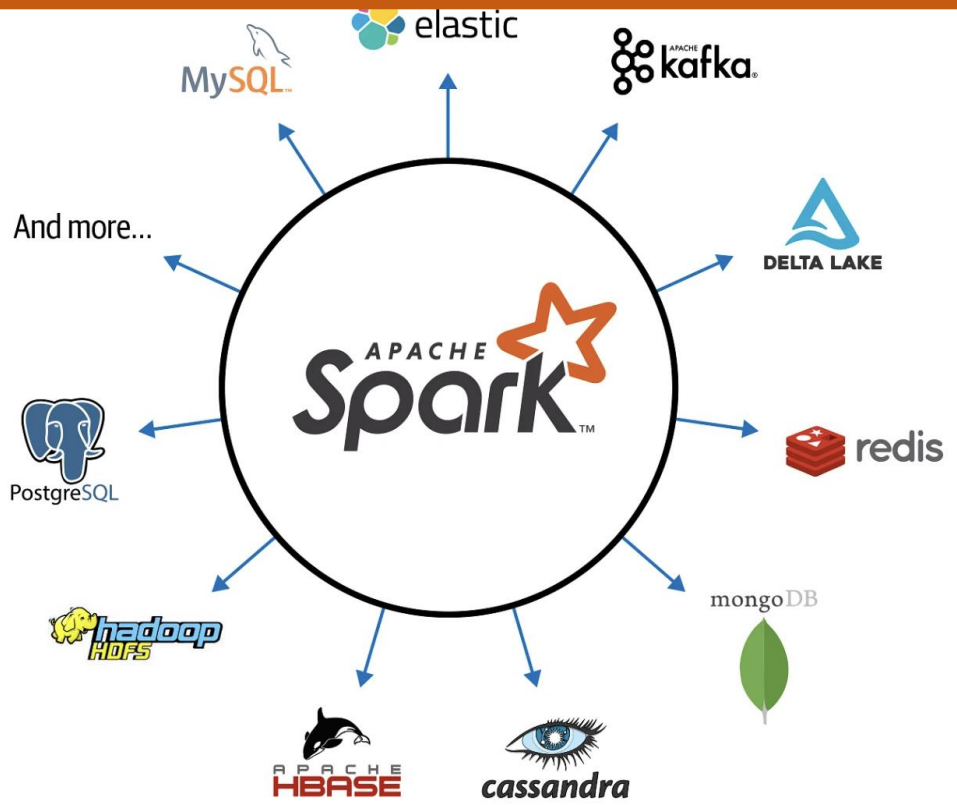
samsara

APACHE Spark

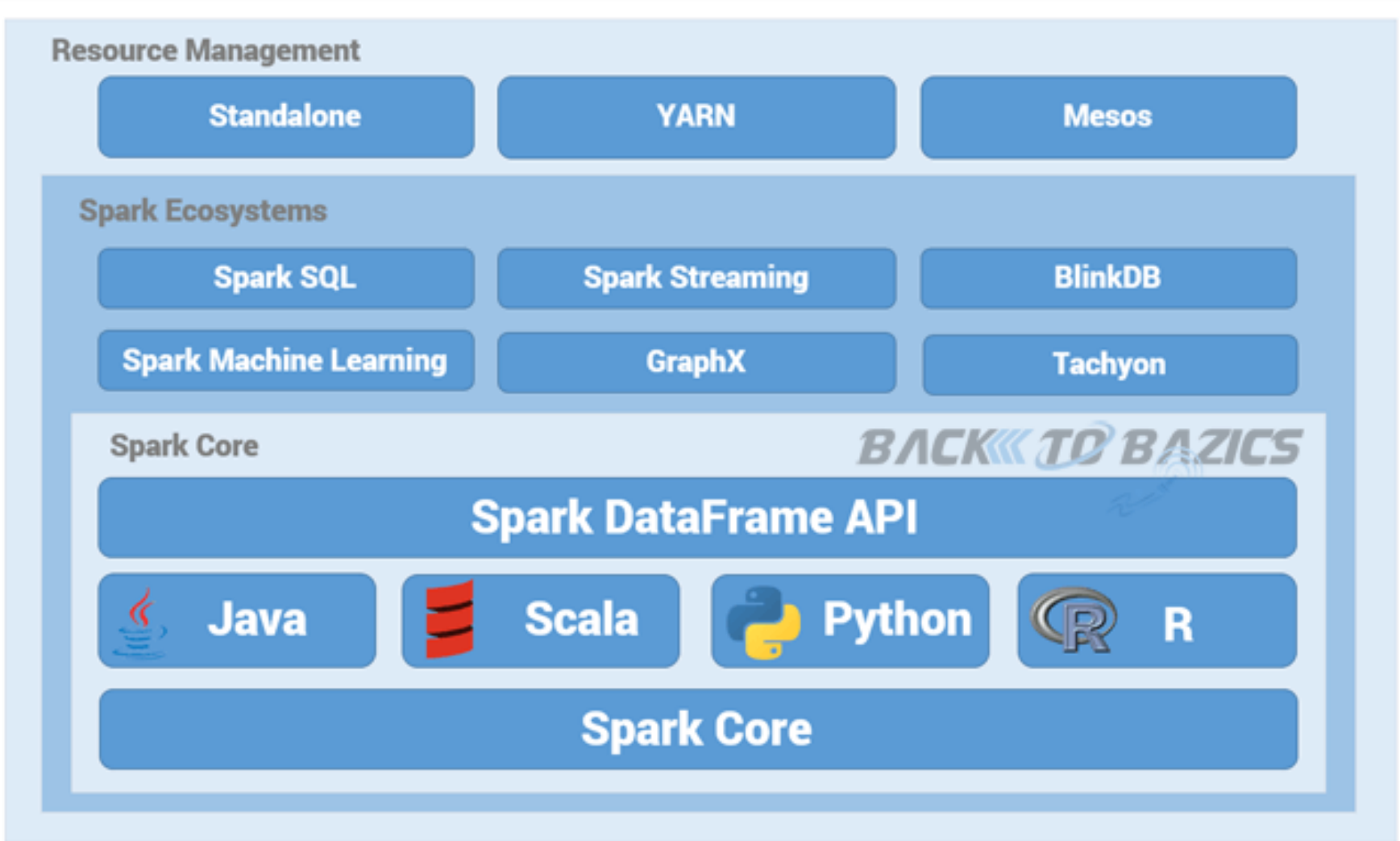
Spark has a built-in machine learning library that can be used for classification, and regression. It can also build machine-learning pipelines with hyperparameter tuning

APACHE Spark
MLlib

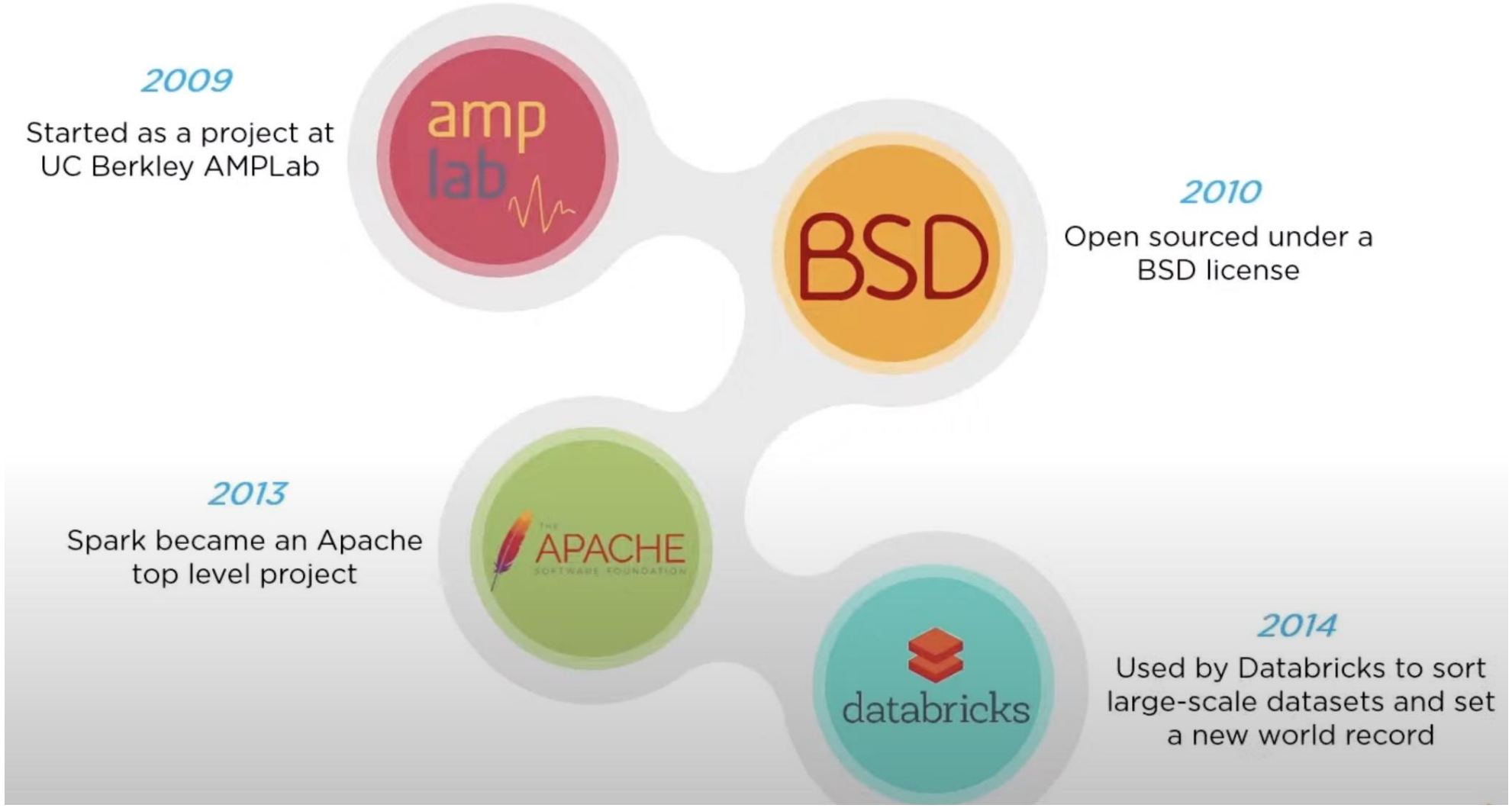
# Data Storage Technologies



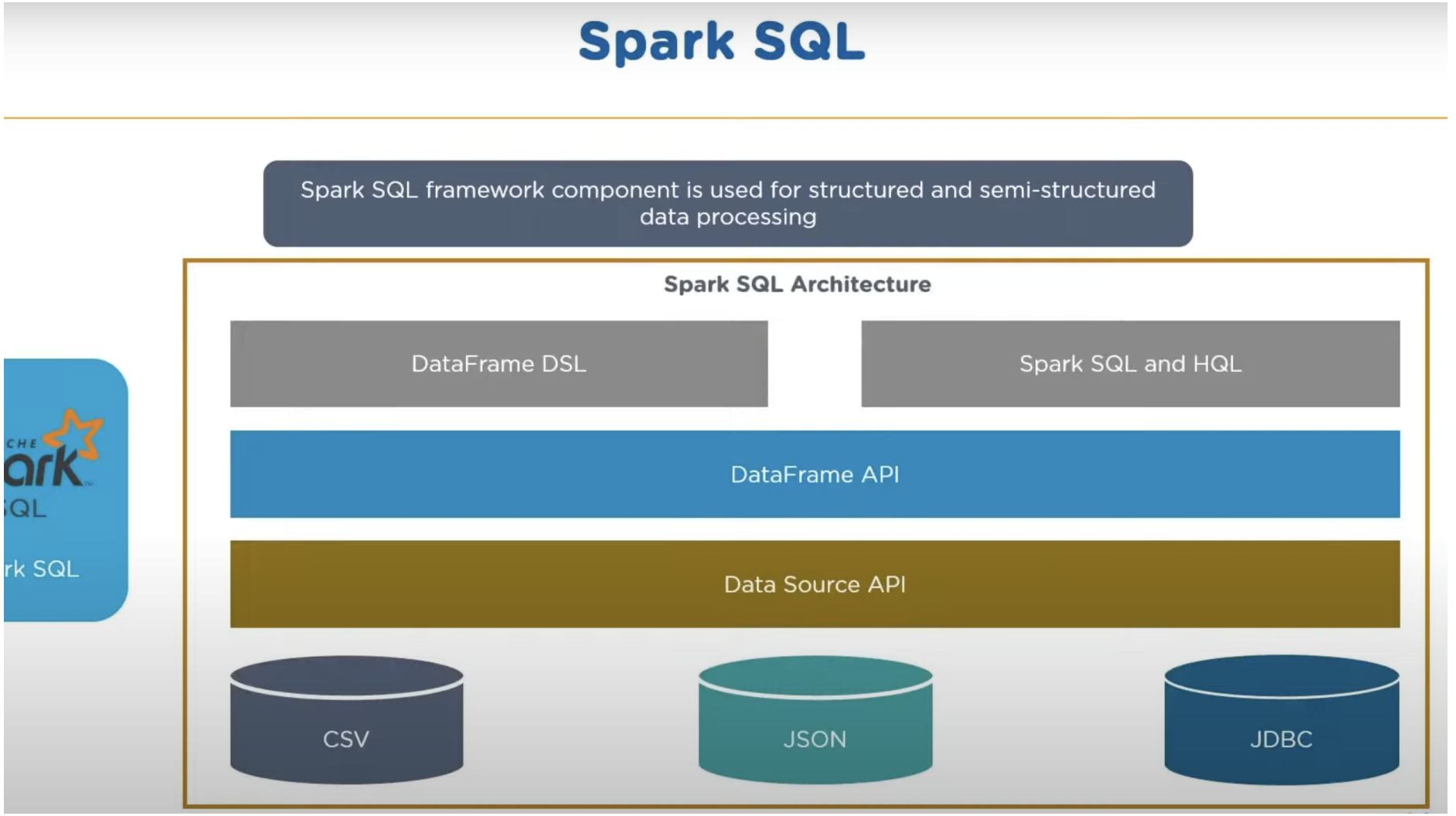

Whereas Hadoop reads and writes files to HDFS, Spark processes data in RAM using a concept known as an RDD, Resilient Distributed Dataset.


Resource Management
Standalone
YARN
Mesos
Spark Ecosystems
Spark SQL
Spark Streaming
BlinkDB
Spark Machine Learning
GraphX
Tachyon
Spark Core
BACK TO BAZICS
Spark DataFrame API
Java
Scala
Python
R
Spark Core

# History of Apache Spark

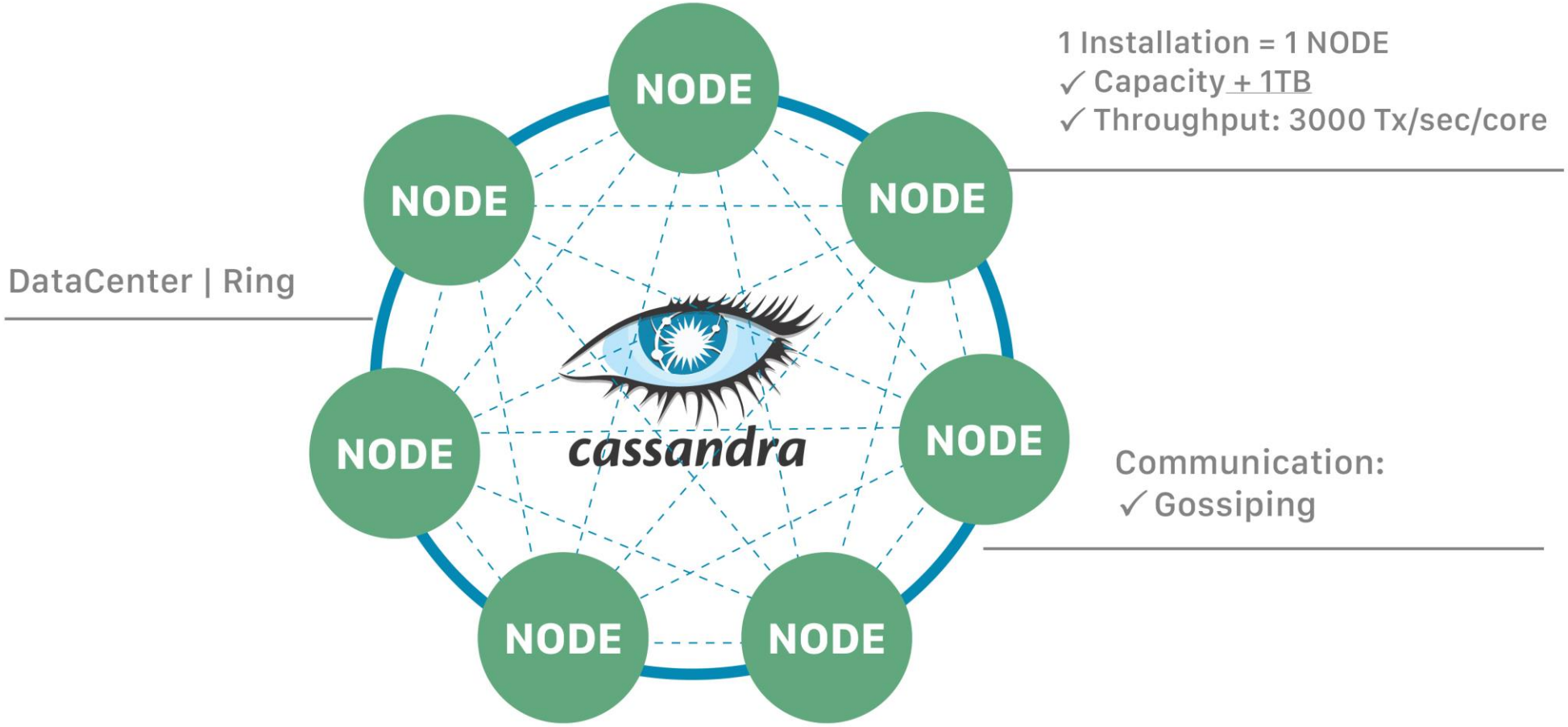


# Spark SQL


Spark SQL
Spark SQL framework component is used for structured and semi-structured data processing
Spark SQL Architecture
DataFrame DSL
Spark SQL and HQL
DataFrame API
Data Source API
CSV
JSON
JDBC

# Data Storage Technologies

## ApacheCassandra™= NoSQL Distributed Database



# Summary


COLLECTION
STORAGE
NETWORK
RESEARCH
BIG DATA
CLOUD
TECHNOLOGY
ANALYSIS
VISUALIZATION
EXABITE
VOLUME

# Summary

***Why is Hadoop used ?***

- it can handle a large amount of data quickly
- all the steps mentioned above are automatic
- it is fully open source
- it is compatible with all platforms since written in Java
- we can add servers and remove some dynamically
- it handles failure cases

# Summary

***When to use Hadoop?***

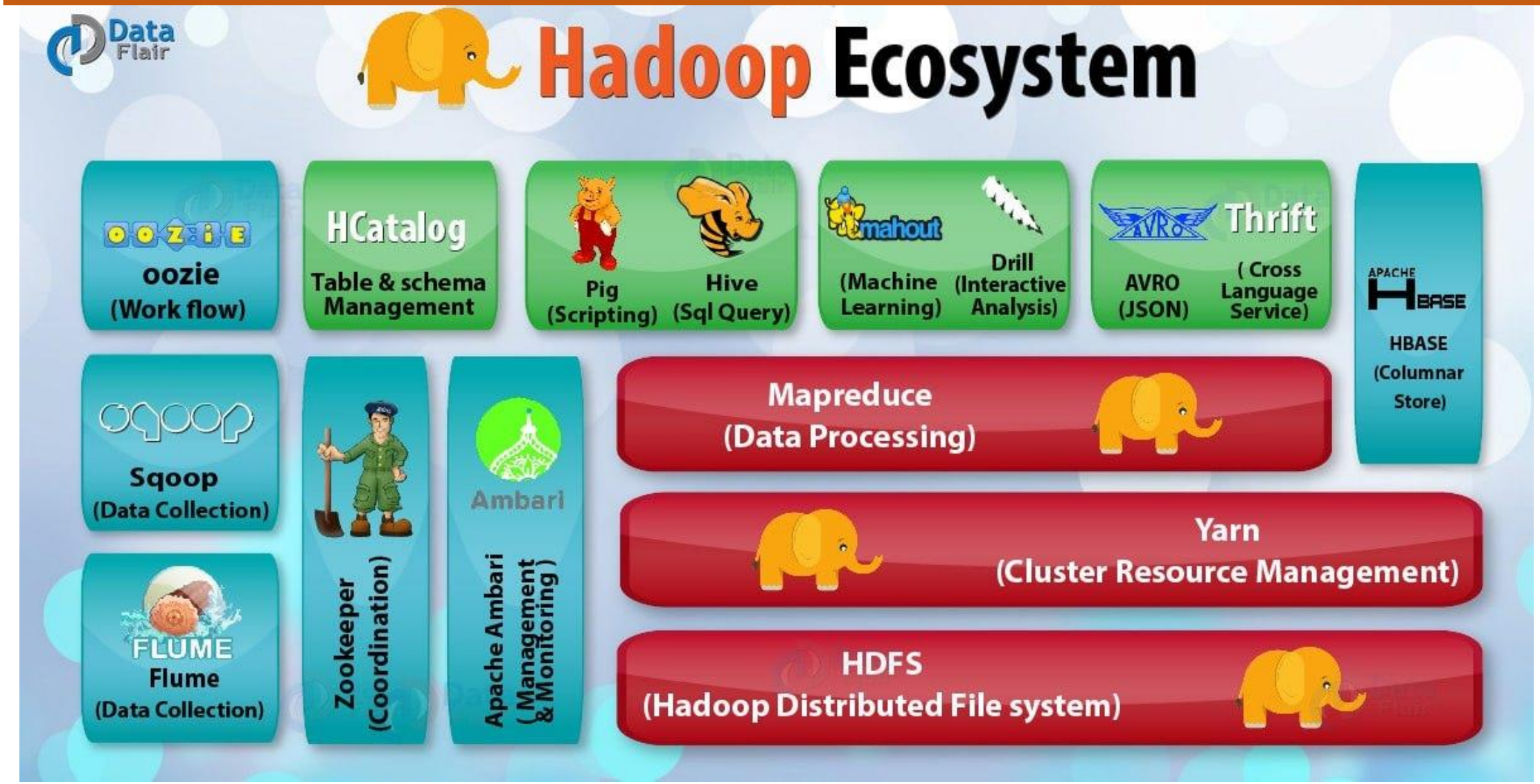
- ❑ your local machine can’t handle the computation
- ❑ the computation can be parallelized
- ❑ you can’t use Spark
- ❑ you have no other choice

# Summary

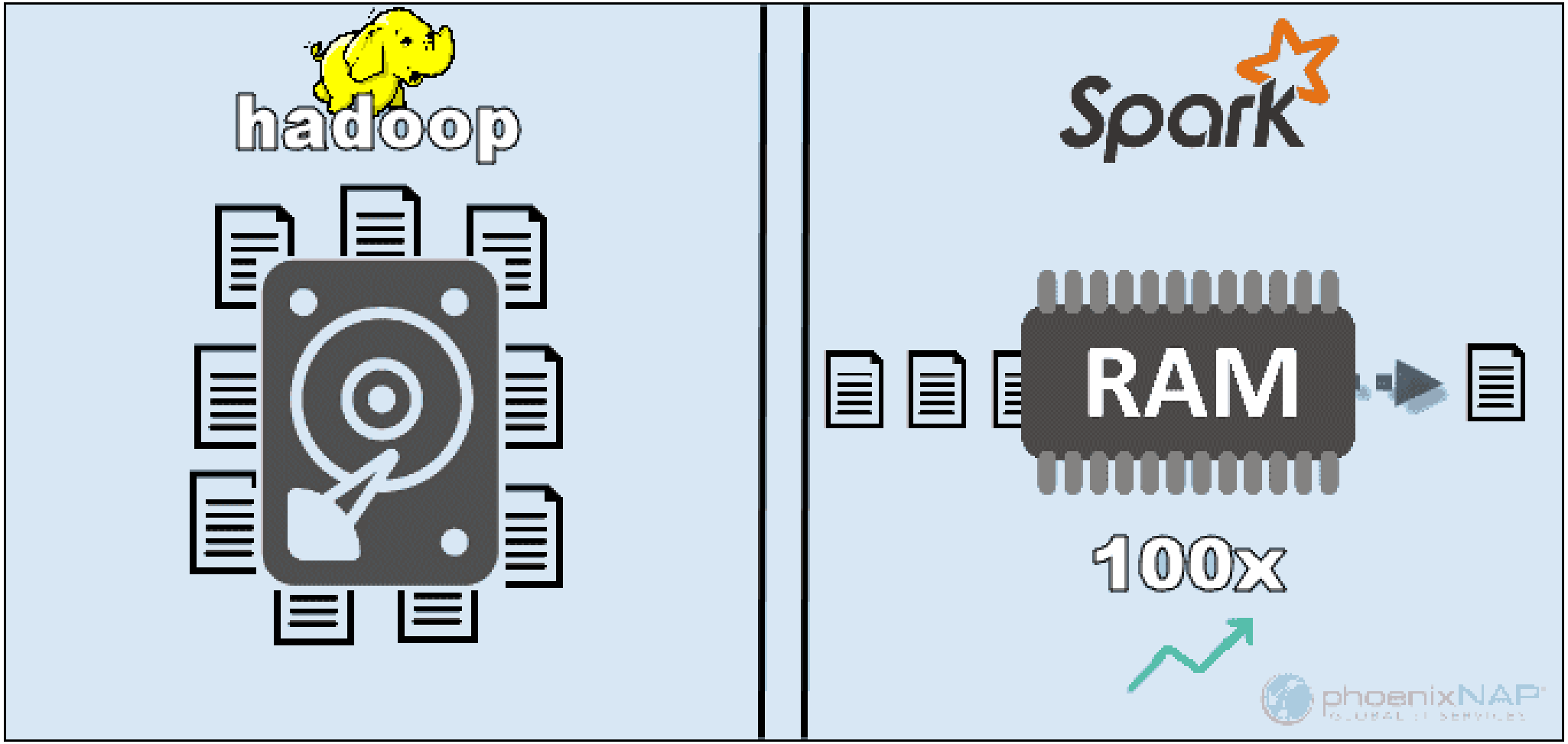
***What are the limits of Hadoop ?***

- ❑ The shuffle part is really expensive in terms of computations
- ❑ The files should be transferred to HDFS, and this is expensive
- ❑ The community around Hadoop is not active anymore
- ❑ Spark is almost systematically chosen

# Summary


Data Flair
Hadoop Ecosystem
oozie
(Work flow)
HCatalog
Table & schema Management
Pig
(Scripting)
Hive
(Sql Query)
mahout
(Machine Learning)
Drill
(Interactive Analysis)
AVRO
(JSON)
Thrift
( Cross Language Service)
APACHE HBASE
HBASE
(Columnar Store)
Sqoop
(Data Collection)
Zookeeper
(Coordination)
Ambari
Apache Ambari
(Management & Monitoring)
Mapreduce
(Data Processing)
FLUME
Flume
(Data Collection)
Yarn
(Cluster Resource Management)
HDFS
(Hadoop Distributed File system)

# Summary

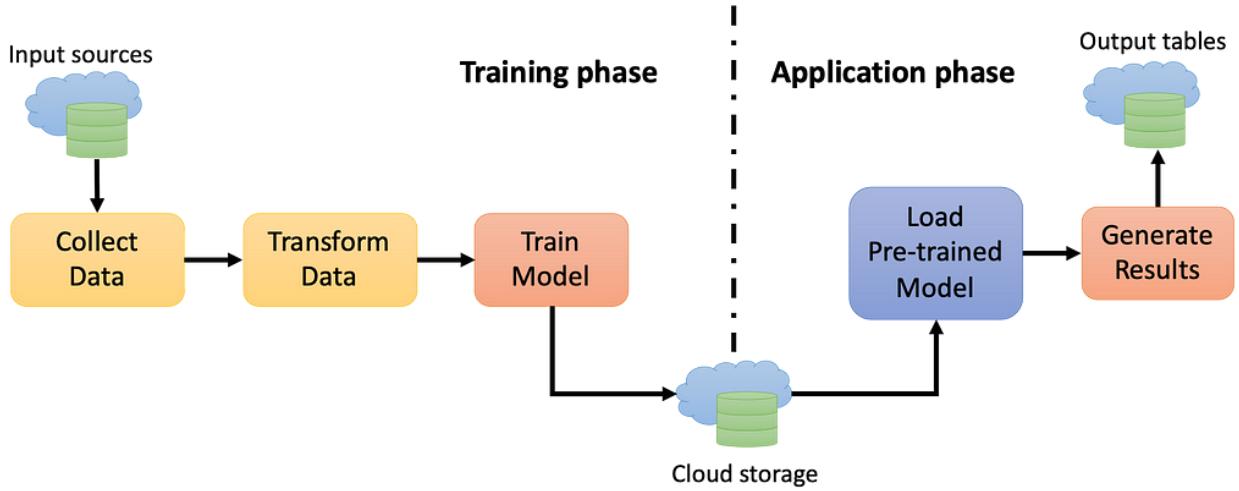


# NEXT TOPIC

AI VIETNAM
All-in-One Course

# Big Data 2

## *(Zero To Mastery: ML Models Using PySpark in Cloud Platform)*


Vinh Dinh Nguyen
PhD in Computer Science

Year 2023