Big Data Hadoop Tutorial

Introduction

Getting started with Big Data and Hadoop can be very daunting, full of complicated terminology and configuration headaches. We have developed this tutorial to take away those very same pain points by simplifying the concepts into easy, understandable steps. You will learn how to process large data sets in an efficient way. Ready to demystify Big Data? Click here to view the full **Big Data & Hadoop Course Syllabus**!

Why Students or Freshers Learn Big Data and Hadoop Tutorial for Beginners

Big Data and Hadoop are necessary to be learned by students and freshers because of the huge demand in the modern tech industry, which relates to high earnings.

- **Large Demand for Jobs:** Almost all large companies require experts to monitor and analyze huge data, thus creating a large shortage.

- **High Salaries:** Big Data roles like Data Engineer and Hadoop Developer have highly competitive, above-average starting salaries.
- **Foundational Technology:** Hadoop is the core framework for large-scale processing of data, which provides a very strong foundation for learning advanced tools such as Spark and Cloud platforms.
- **Futures-proof skill:** Your skill is in demand in finance, e-commerce, and healthcare-in short, making your career pretty resilient.

Ready to land your dream job? Click here for the best **Big Data & Hadoop Interview Questions and Answers**!

Step-by-Step Big Data and Hadoop Tutorial for Beginners

Part 1. Introduction to Big Data Hadoop

Big Data: Big Data are extremely large-scale data sets. Typically, we work with data that is MB (Word Doc, Excel) or up to GB (movies, codes); big data is defined as data that is in petabytes.

Big Data Hadoop: Hadoop is an Apache open-source platform used for processing and analyzing massive volumes of data. Facebook, Yahoo, Google, Twitter, LinkedIn, and numerous other companies use it.

Modules of Hadoop

- **HDFS:** Hadoop Distributed File System. It says that blocks of files will be divided up and kept in nodes throughout the distributed architecture.

- **Yarn:** Yet Another Resource Negotiator. It is used for managing the cluster and scheduling jobs.
- **Map Reduce:** It is a framework that aids Java programs in leveraging key-value pairs to compute data in parallel. The '*Map'* process converts data input into a collection of data that can be computed in key-value pairs.
- **Hadoop Common:** Other Hadoop modules use these Java libraries, which are also used to launch Hadoop.

Part 2. Hadoop Installation

To install Hadoop from a *'tar ball'* in a UNIX environment, you require the following:

- Java Installation
- SSH installation
- Hadoop Installation and File Configuration

Java Installation

Step 1: Get Java at

www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html

if it's not already installed. On your computer, the tar file *jdk-7u71-linux-x64.tar.gz* will be downloaded.

Step 2: Use the command below to extract the file. *#tar zxf jdk-7u71-linux-x64.tar.gz*

Step 3: Move the file to */usr/local* and configure the path to enable Java for all UNIX users.

To relocate the JDK to */usr/lib*, switch to the root user at the prompt and enter the following command.

 # mv jdk1.7.0_71 /usr/lib/

To configure the path, add the following instructions to the ~/.bashrc file.

 # export JAVA_HOME=/usr/lib/jdk1.7.0_71

 # export PATH=PATH:$JAVA_HOME/bin

Now that you have typed “java -version” into the prompt, you may verify the installation.

SSH Installation

Passwords are not requested while interacting with the master and slave computers over SSH. Make a Hadoop user on the master and slave systems first.

 # useradd hadoop

 # passwd Hadoop

To map the nodes, open the host file located in each machine’s /etc/ folder and provide the *hostname and IP address*.

 # vi /etc/hosts

Fill in the lines below.

190.12.1.114 hadoop-master

190.12.1.121 hadoop-salve-one

190.12.1.143 hadoop-slave-two

Configure each node with an SSH key so that it may communicate with the others without a password. Instructions for the same are:

 # su hadoop

$ ssh-keygen -t rsa

$ ssh-copy-id -i ~/.ssh/id_rsa.pub tutorialspoint@hadoop-master

$ ssh-copy-id -i ~/.ssh/id_rsa.pub hadoop_tp1@hadoop-slave-1

$ ssh-copy-id -i ~/.ssh/id_rsa.pub hadoop_tp2@hadoop-slave-2

$ chmod 0600 ~/.ssh/authorized_keys

$ exit

Hadoop Installation

Download links for Hadoop are available at

developer.yahoo.com/hadoop/tutorial/module3.html

Extract the Hadoop now, and move it to a different location.

$ mkdir /usr/hadoop

$ sudo tar vxzf hadoop-2.2.0.tar.gz ?c /usr/hadoop

Modify who owns the Hadoop folder.

$sudo chown -R hadoop usr/hadoop

Modify the configuration files for Hadoop:

There are all the files in */usr/local/Hadoop/etc/hadoop.*

Step 1: In hadoop-env.sh file add

export JAVA_HOME=/usr/lib/jvm/jdk/jdk1.7.0_71

Step 2: Add the following in core-site.xml in between the configuration tabs:

<configuration>

<property>

<name>fs.default.name</name>

<value>hdfs://hadoop-master:9000</value>

</property>

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

</configuration>

Step 3: After switching between the configuration tabs on *hdfs-site.xmladd,*

<configuration>

<property>

<name>dfs.data.dir</name>

<value>usr/hadoop/dfs/name/data</value>

<final>true</final>

</property>

<property>

<name>dfs.name.dir</name>

<value>usr/hadoop/dfs/name</value>

<final>true</final>

</property>

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

</configuration>

Step 4: Make the necessary changes to Mapred-site.xml as indicated below.

<configuration>

<property>

<name>mapred.job.tracker</name>

<value>hadoop-master:9001</value>

</property>

</configuration>

Step 5: Lastly, make updates to $HOME/.bahsrc.

cd $HOME

vi .bashrc

Append following lines in the end and save and exit

#Hadoop variables

export JAVA_HOME=/usr/lib/jvm/jdk/jdk1.7.0_71

export HADOOP_INSTALL=/usr/hadoop

export PATH=$PATH:$HADOOP_INSTALL/bin

export PATH=$PATH:$HADOOP_INSTALL/sbin

export HADOOP_MAPRED_HOME=$HADOOP_INSTALL

export HADOOP_COMMON_HOME=$HADOOP_INSTALL

export HADOOP_HDFS_HOME=$HADOOP_INSTALL

export YARN_HOME=$HADOOP_INSTALL

Use the following command to install Hadoop on the slave system.

 # su hadoop

$ cd /opt/hadoop

$ scp -r hadoop hadoop-slave-one:/usr/hadoop

$ scp -r hadoop hadoop-slave-two:/usr/Hadoop

Set up the slave and master nodes.

$ vi etc/hadoop/masters

hadoop-master

$ vi etc/hadoop/slaves

hadoop-slave-one

hadoop-slave-two

Following this pattern, launch every deamon and name the node.

 # su hadoop

$ cd /usr/hadoop

$ bin/hadoop namenode -format

$ cd $HADOOP_HOME/sbin

$ start-all.sh

Part 3. Hadoop Architecture

The MapReduce engine, the Hadoop Distributed File System (HDFS), and the file system comprise the Hadoop architecture.

- A Hadoop cluster is made up of several slave nodes and one master node.

- **Master Nodes:** Job Tracker, Task Tracker, NameNode, and DataNode.
- **Slave Nodes:** TaskTracker and DataNode.

Part 4. Hadoop Distributed File System

HDFS has a master/slave architecture This design consists of several DataNodes acting as slaves and a single NameNode acting as the master.

NameNode

The HDFS cluster consists of a single master server. Being a single node, it could lead to a single point of failure.

It streamlines the system's architecture. It does this by opening, renaming, and shutting files to manage the file system namespace.

DataNode

There are several DataNodes in the HDFS cluster. Multiple data blocks are present in every DataNode. The purpose of these data blocks is data storage.

DataNode is in charge of reading and writing requests from clients of the file system. On the NameNode's instruction, it creates, deletes, and replicates blocks.

Job Tracker

Accepting MapReduce jobs from clients and using NameNode to process the data is Job Tracker's responsibility. As a result, NameNode gives Job Tracker metadata.

Task Tracker

It functions as a Job Tracker slave node. It applies the code to the file after receiving the task and code from Job Tracker. Another name for this procedure is a mapper.

Map Reduce Layer

The MapReduce is generated when the client application sends the MapReduce job to Job Tracker. In response, the job tracker forwards the request to the appropriate task trackers.

The TaskTracker occasionally times out or fails. That portion of the work is rescheduled in such a scenario.

Part 5. HDFS Basic File Operations

Step 1: Transferring data from the local file system to HDFS

Create an HDFS folder first so that data from the local file system can be stored there.

$ hadoop fs -mkdir /user/test

Copy the file *"data.txt"* from a file stored in the local folder /usr/home/Desktop to the *HDFS folder /user/test*

$ hadoop fs -copyFromLocal /usr/home/Desktop/data.txt /user/test

Show the contents of the HDFS folder with the command

$ Hadoop fs -ls /user/test

Step 2: Transfer data from HDFS to the local file system with the command

$ hadoop fs -copyToLocal /user/test/data.txt /usr/bin/data_copy.txt

Step 3: Verify if the files are identical by comparing them.

$ md5 /usr/bin/data_copy.txt /usr/home/Desktop/data.txt

Recursive Deleting

hadoop fs -rmr <arg>

Example: hadoop fs -rmr /user/sonoo/

Part 6. HDFC Other Commands

The commands make use of the following.

- “<path>” denotes the name of any file or directory.
- “<path>…” denotes a file or directory name or names.
- “<file>” can refer to any filename.
- In a directed operation, the path designations are “<dest>” and “<src>”.
- “<localSrc>” and “<localDest>” are paths on the local file system, similar to those above.

put <localSrc><dest>: It copies the file or directory from the local file system, denoted with localSrc, to dest in the DFS.

copyFromLocal <localSrc><dest>: Similar to -put

copyFromLocal <localSrc><dest>: Similar to -put

moveFromLocal <localSrc><dest>: It copies the file or directory to dest in HDFS from the local file system that localSrc has identified, then, upon success, deletes the local copy.

get [-crc] <src><localDest>: The file or directory is moved locally from HDFS, denoted with src, to the local file system path, denoted as localDest.

cat <filen-ame>: It shows the contents of the filename on the standard output.

moveToLocal <src><localDest>: Similar to -get, except it removes the HDFS copy upon success.

setrep [-R] [-w] rep <path>: It sets the file names indicated by the path to the rep's target replication factor. (Over time, the replication factor itself will approach the target.)

touchz <path>: It creates a file at the path with a timestamp of the present moment. fails if there is already a file in the path unless the file has a zero size.

test -[ezd] <path>: Returns 0 otherwise, 1 if the path is a directory, has zero length, or both.

stat [format] <path>: It prints the path information. File size in blocks (%b), filename (%n), block size (%o), replication (%r), and modification date (%y, %Y) are all accepted in the format, which is a string.

Learn more with our **Big Data Hadoop Challenges and Solutions**.

Real Time Examples for Big Data and Hadoop Tutorial for Learners

The power of Big Data and Hadoop can easily be comprehended from the way they solve real-world problems involving massive, complex, fast-moving data.

E-commerce: Personalized Product Recommendations

This big data analyzing of billions of user actions, purchases, views, and ratings is done by companies like Amazon and Netflix.

- **Problem:** With millions of items, how does a company know which ones to show to a particular customer?
- **Hadoop's role:** It stores enormous historic records in HDFS, like clickstreams, purchase history, and search queries. The MapReduce processing framework, sometimes combined with faster tools like Apache Spark, executes machine learning algorithms that expose complex patterns across all users.

In the result, the system recommends things that you are most likely to purchase: “Customers who bought this also bought.”. This drives a huge percentage of their sales.

Financial Services: Real-time Fraud Detection

Banks and credit card companies need to review millions of transactions every day in real-time to weed out and block fraudulent activities.

- **Problem:** Traditional systems are too slow to check a transaction against millions of historical fraud patterns in real time.
- **Hadoop’s Role:** Hadoop provides a fault-tolerant, scalable repository for all transactional data and historical fraud profiles. The ecosystem, often using Spark Streaming, allows the real-time processing of new transactions. If a card is used in two different countries within a minute, this anomaly can be flagged instantly by the system.

Result: A suspicious transaction is blocked in milliseconds, saving billions of dollars in losses and protecting customer accounts.

Telecom/IoT: Predictive Maintenance

Airlines, manufacturers, and utility companies use data from thousands of sensors-what some call the Internet of Things or IoT-to predict when equipment will fail.

- **Problem:** Manual monitoring of a jet engine or a power turbine is impossible; failure is costly and dangerous.
- **Hadoop’s Role:** Sensor data – vibration, temperature, and pressure – streams into the Hadoop cluster uninterruptedly. The cluster stores this terabytes-scale time-series data and then runs models for the detection of small, leading indicators of wear and tear.

Result: This enables companies to schedule maintenance before a catastrophic failure, reducing downtime significantly, improving safety, while saving millions in unplanned repair costs.

Ready to apply these concepts? Explore our list of **Big Data and Hadoop Project Ideas** to start building your portfolio!

FAQs About Big Data and Hadoop Tutorial for Beginners

1.What is Hadoop and Big Data?

Big Data is generally defined as extremely large, diverse data sets that cannot be managed using traditional tools. Hadoop is the Apache open-source framework that offers distributed storage, HDFS, and parallel processing, MapReduce/YARN over clusters of commodity hardware, to handle and analyze this huge amount of data efficiently.

2.What are the 4 main components of Hadoop?

The four core parts of the Apache Hadoop framework include:

HDFS stands for Hadoop Distributed File System. It enables distributed data storage to be scalable and fault-tolerant.

YARN: Yet Another Resource Negotiator – manages cluster resources and schedules jobs.

MapReduce is a programming model for processing vast quantities of data in parallel across a cluster of computers.

Hadoop Common: This provides the common utilities and libraries required by other modules.

3.What is meant by big data?

Big Data refers to datasets that are too large and complex to be captured, managed, and processed within a tolerable elapsed time by using conventional database software tools. It is typically characterized by the 3 Vs: high Volume (scale), high Velocity or

speed of generation/analysis, and high Variety pertaining to structured, unstructured, and semi-structured formats.

4.What are the 4 types of big data analytics?

The main types of big data analytics, in order from simple to complex, include:

Descriptive Analytics: What happened? Examples include historical reports.
Diagnostic Analytics: Why did it happen? (e.g., drilling down into causes).
Predictive Analytics: What will happen? Examples include forecasting and risk assessment.
Prescriptive Analytics: What should we do? Examples include optimal actions for any circumstance.

5.What is the full form of Hadoop?

The name Hadoop is not a standard, official acronym. It was literally named by its creator, Doug Cutting, after his son's toy elephant. Although sometimes referred to by a backronym such as "High Availability Distributed Object-Oriented Platform," this is not its true meaning.

6.What are the 4 types of data analytics?

Descriptive is used to summarize data from the past to comprehend what took place. Diagnostic involves digging into data to understand 'why it happened.' Predictive analytics does just that, using statistical models to predict what will take place. Prescriptive analytics defines the best course of action to achieve a goal.

7.Why is Hadoop used?

Hadoop is used because it offers a cost-effective and scalable solution for storing and processing massive volumes of diverse data. Its distributed nature provides high fault tolerance and allows for parallel processing, making it ideal for tasks such as data warehousing, complex analytics, and machine learning on Big Data.

8.What skills are needed for Hadoop?

Central to this in the working of Hadoop would be the deep knowledge of its core parts: HDFS and YARN, together with the ecosystem tools like Hive, Pig, and Spark. Additionally, technical expertise in a programming language, such as Java, Python, or

Scala, is required, together with SQL for data querying and Linux command-line operations. Explore **Hadoop salary for freshers**.

9.What type of database is Hadoop?

Technically, Hadoop itself is not a database but a distributed file system-HDFS-and a processing framework. However, it serves as the basis for Big Data NoSQL databases like Apache HBase, which runs atop HDFS. Hadoop serves as a data lake or storage layer, not a transactional RDBMS.

10.Is SQL used in Hadoop?

Yes, SQL is implemented widely in the Hadoop ecosystem. Since native processing in Hadoop (MapReduce) is non-SQL, Apache Hive and Impala allow users to query data stored in HDFS with familiar SQL syntax, making Big Data analytics accessible.

Conclusion

You have learnt the basics of Big Data and the Hadoop ecosystem. Now you know how this powerful technology solves the problems of Volume, Velocity, and Variety: with scalable storage, HDFS, and parallel processing. This is a very important beginning for your successful career. Want to go past the basics and learn how to use the entire ecosystem, including Spark, Hive, and more? Enroll in our full **Big Data and Hadoop course in Chennai** to get certified today!