Environment setup

### 

### Objective - Environment configuration set up to onboard a pipeline from MySQL to Hive using Apache Airflow.

This document outlines the tools needed to be installed before we can run an Airflow pipeline from MySQL to HIVE.

We need -

1. Java 1.8
2. Hadoop Stack [HDFS, MapReduce]
3. MySQL [[How To Install MySQL on Ubuntu](https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-20-04)]
4. Python3 [[How to Install Python 3.9 on Ubuntu 22.04](https://linuxhint.com/install-python-ubuntu-22-04/)]
5. HIVE
6. Airflow [with MySQL and HIVE providers package]

### Java

#### Install JDK on Ubuntu

Before installing the Java Runtime Environment (JRE) and Java Development Kit (JDK) update and upgrade packages in the Ubuntu system.

| $ sudo apt-get update $ sudo apt-get upgrade |
| --- |

The easiest option for installing Java is to use the version packaged with Ubuntu. By default, Ubuntu includes OpenJDK, which is an open-source variant of the JRE and JDK.

Check if Java is already installed:

| $ java -version; javac -version |
| --- |

If Java is not currently installed, then run the following command:

| $ sudo apt install openjdk-8-jdk -y |
| --- |

Verify the installation with:

| $ java -version; javac -version |
| --- |

#### Install OpenSSH on Ubuntu

The ssh command provides a secure encrypted connection between two hosts over an insecure network. This connection can also be used for terminal access, file transfers, and executing commands on the remote machine. Install the OpenSSH server and client using the following command:

| $ sudo apt install openssh-server openssh-client -y |
| --- |

#### Configure passwordless SSH

The user will able to SSH to the localhost without being prompted for a password. Generate Public and Private Key Pairs with the following command:

| $ ssh-keygen -t rsa -P '' -f ~/.ssh/id\_rsa |
| --- |

Copy the public keys from id\_rsa.pub to authorized\_keys.

| $ cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys |
| --- |

Set the permissions for your user with the chmod command:

| $ chmod 0600 ~/.ssh/authorized\_keys |
| --- |

The new user is now able to SSH without needing to enter a password every time. Verify everything is set up correctly by using the **hadoop** user to SSH to **localhost**:

| $ ssh localhost $ exit |
| --- |

After an initial prompt, the user is now able to establish an SSH connection to the localhost seamlessly.

### Download and Install Hadoop on Ubuntu

Visit the [official Apache Hadoop project page](https://hadoop.apache.org/release/2.9.1.html), and select the version of Hadoop you want to implement.

**Note** - Select the latest 2.9.1 and click on binary to download. The steps outlined in this tutorial use the Binary download for Hadoop Version 2.9.1. It up to the user which version it wants to use; installation steps will remain the same.

| $ sudo mv /home/bharat/Desktop/hadoop-2.9.1/ /home/hadoop/ $ ls $ tar xzf hadoop-2.9.1.tar.gz $ mv hadoop-2.9.1 hadoop $ ls |
| --- |

#### Single Node Hadoop Setup

This setup, also called pseudo-distributed mode, allows each Hadoop daemon to run as a single Java process. A Hadoop environment is configured by editing a set of configuration files:

* bashrc
* hadoop-env.sh
* core-site.xml
* hdfs-site.xml
* mapred-site-xml
* Yarn-site.xml

#### [Not all commands but the configs related to this environment setup]

.bashrc

| export PATH="$HOME/gems/bin:$PATH" export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-amd64 export HADOOP\_HOME=/home/pkt/hadoop-2.9.1 export HIVE\_HOME=/home/pkt/apache-hive-3.1.2-bin export PATH=$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$HIVE\_HOME/bin:$HADOOP\_HOME/sbin:$PATH export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_HOME/lib/native" |
| --- |

core-site.xml

| <**configuration**> <**property**>  <**name**>hadoop.tmp.dir</**name**>  <**value**>/home/pkt/tmpdata</**value**> </**property**> <**property**>  <**name**>fs.default.name</**name**>  <**value**>hdfs://localhost:9000</**value**> </**property**> </**configuration**> |
| --- |

hdfs-site.xml

| <**configuration**> <**property**>  <**name**>dfs.replication</**name**>  <**value**>1</**value**> </**property**>  <**property**>  <**name**>dfs.name.dir</**name**>  <**value**>file:///home/pkt/hadoopdata/hdfs/namenode</**value**> </**property**>  <**property**>  <**name**>dfs.data.dir</**name**>  <**value**>file:///home/pkt/hadoopdata/hdfs/datanode</**value**> </**property**> </**configuration**> |
| --- |

mapred-site.xml

| <**configuration**> <**property**>  <**name**>mapreduce.framework.name</**name**>  <**value**>yarn</**value**>  </**property**> </**configuration**> |
| --- |

yarn-site.xml

| <**configuration**>  <**property**>  <**name**>yarn.nodemanager.aux-services</**name**>  <**value**>mapreduce\_shuffle</**value**>  </**property**> </**configuration**> |
| --- |

hadoop-env.sh

| export JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64 |
| --- |

#### Format NameNode

It is important to format the NameNode before starting Hadoop services for the first time. Now format the namenode using the following command, make sure that Storage directory is :

| $ cd ~/hadoop/sbin $ hdfs namenode -formathttp://localhost:9864/ |
| --- |

#### Start Hadoop Cluster

Navigate to the *hadoop/sbin* directory and execute the following commands to start the NameNode and DataNode:

| $ ./start-dfs.sh |
| --- |

Once the **namenode**, **datanodes**, and **secondary** **namenode** are up and running, start the **YARN** resource and nodemanagers by typing:

| $ ./start-yarn.sh |
| --- |

To verify all the Hadoop services/daemons are started successfully you can use the jps command.

| $ jps |
| --- |

**NameNode here-** [**http://localhost:50070**](http://localhost:50070)

**Resource Manager -** [**http://localhost:8042/node**](http://localhost:8042/node)

### HIVE Installation -

### [Installation [ Apache Hive 2 with MySQL ]](https://www.youtube.com/watch?v=COBDBw5GKjw)

One can copy the commands to be executed from - [here](https://github.com/Gowthamsb12/hive/blob/main/hive-site.xml)

### Airflow Installation -

| pip install apache-airflow-providers-apache-hive pip install apache-airflow-providers-mysql pip install "apache-airflow[celery]==2.5.0" --constraint "https://raw.githubusercontent.com/apache/airflow/constraints-2.5.0/constraints-3.7.txt" |
| --- |

Reference Links -

* [Installation from PyPI](https://airflow.apache.org/docs/apache-airflow/stable/installation/installing-from-pypi.html#installation-from-pypi)
* [Quick Start](https://airflow.apache.org/docs/apache-airflow/stable/start.html)
* [How to install Apache Airflow on local machine?: Airflow Tutorial](https://www.youtube.com/watch?v=lKL7DMIfMyc)

### MySQL Python connector -

Documentation here - [mysql-connector-python 8.0.31](https://pypi.org/project/mysql-connector-python/)