cloudera

Microsoft
Azure

Contents

1. About Cloudera

Cloudera is an open-source Apache Hadoop distribution, CDH (Cloudera Distribution Including Apache Hadoop) targets enterprise-class deployments of that technology.

Cloudera provides a scalable, flexible, integrated platform that makes it easy to manage rapidly increasing volumes and varieties of data in your enterprise. Cloudera products and solutions enable you to deploy and manage Apache Hadoop and related projects, manipulate and analyze your data, and keep that data secure and protected.

Cloudera develops a Hadoop platform that integrates the most popular Apache Hadoop open source software within one place. Hadoop is an ecosystem, and setting a cluster manually is a pain. Going through each node, deploying the configuration though the cluster, deploying your services, and restarting them on a wide cluster is a major drawback of distributed system and require lot of automation for administration. Cloudera developed a big data Hadoop distribution that handles installation and updates on a cluster in few clicks.

Cloudera also develop their own projects such as Impala or Kudu that improve hadoop integration and responsiveness in the industry.

1.1. Cloudera Director

Cloudera Director enables reliable self-service for using CDH and Cloudera Enterprise Data Hub in the cloud.

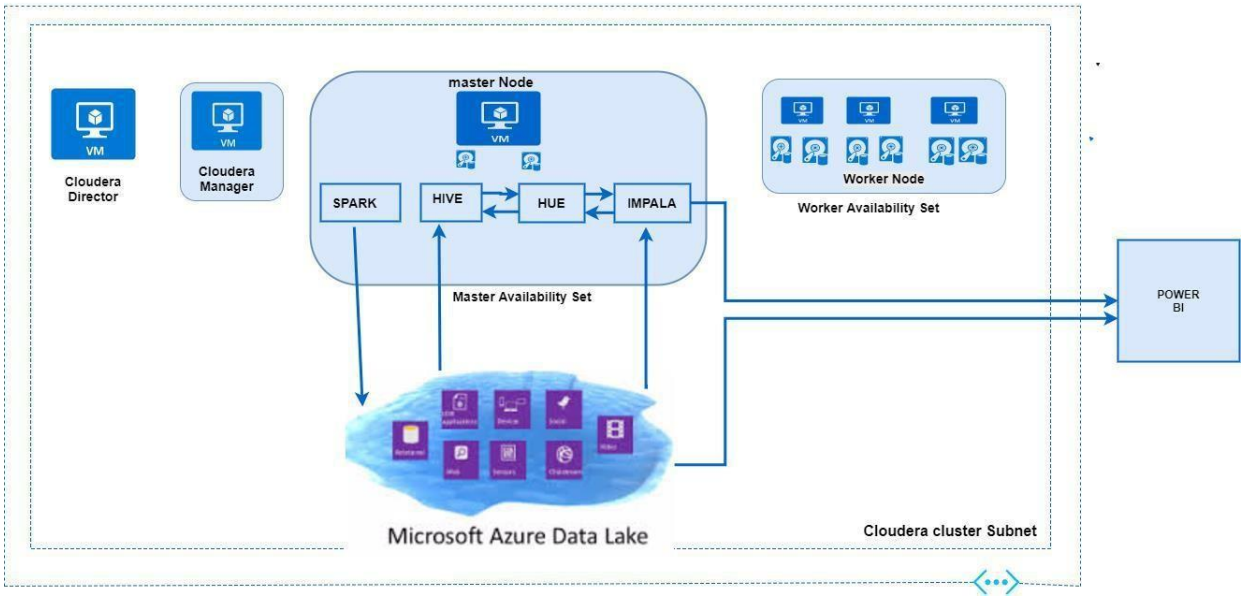
Cloudera Director provides a single-pane-of-glass administration experience for central IT to reduce costs and deliver agility, and for end-users to easily provision and scale clusters. Advanced users can interact with Cloudera Director programmatically through the REST API or the CLI to maximize time-to-value for an enterprise data hub in cloud environments.

Cloudera Director is designed for both long running and transient clusters. With long running clusters, you deploy one or more clusters that you can scale up or down to adjust to demand. With transient clusters, you can launch a cluster, schedule any jobs, and shut the cluster down after the jobs complete.

The Cloudera Director server is designed to run in a centralized setup, managing multiple Cloudera Manager instances and CDH clusters, with multiple users and user accounts. The server works well for launching and managing large numbers of clusters in a production environment.

1.2. Cloudera Manager

Cloudera Manager is a sophisticated application used to deploy, manage, monitor, and diagnose issues with your CDH deployments. Cloudera Manager provides the Admin Console, a web-based user interface that makes administration of your enterprise data simple and straightforward. It also includes the Cloudera Manager API, which you can use to obtain cluster health information and metrics, as well as configure Cloudera Manager.



2. Objective

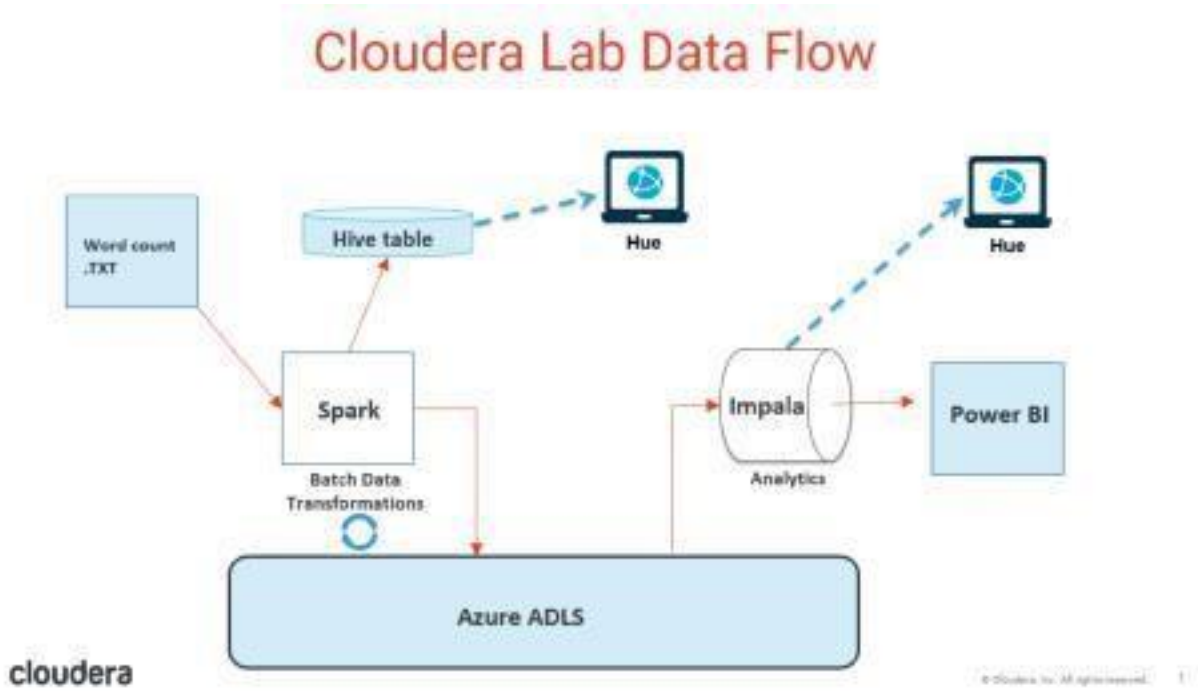
NOTE: As this test drive provides access to the full Cloudera Director platform, deployment can sometimes take up to **45 minutes**. While you wait, please feel free to review to helpful content in this manual and on Cloudera's Azure Marketplace product page, or on the Cloudera website. Please also consider watching the demo video showcased on the test drive launch page on the Azure Marketplace web site.

The test drive provisions Cloudera Director, the environment, Cloudera Manager, and a cluster consisting of 1 master node and 3 worker nodes. The test drive also integrates with Azure Data Lake Store.

The use case scenario for this test drive is to provide users with a test Azure Data Lake Store and:

1. Run the **WordCount** app with Hadoop/Spark on ADLS.
2. Create a Hive table on the output, and query Hive from Hue.
3. Run query using Impala from Hue or Power BI.

The following diagram shows how the data in this test case flows from a .TXT file via Hue to ADLS, processed by Spark.

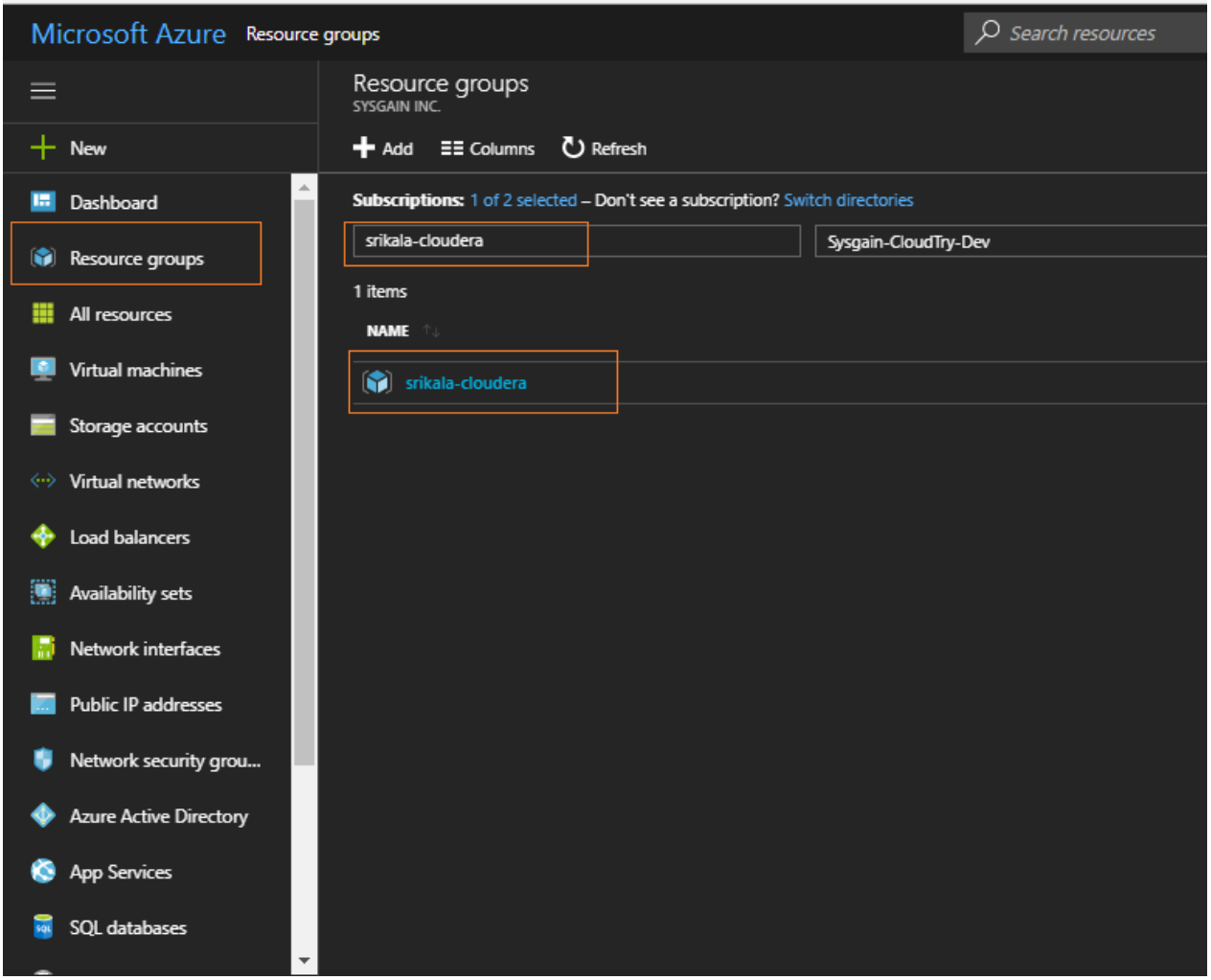


3. Getting Started

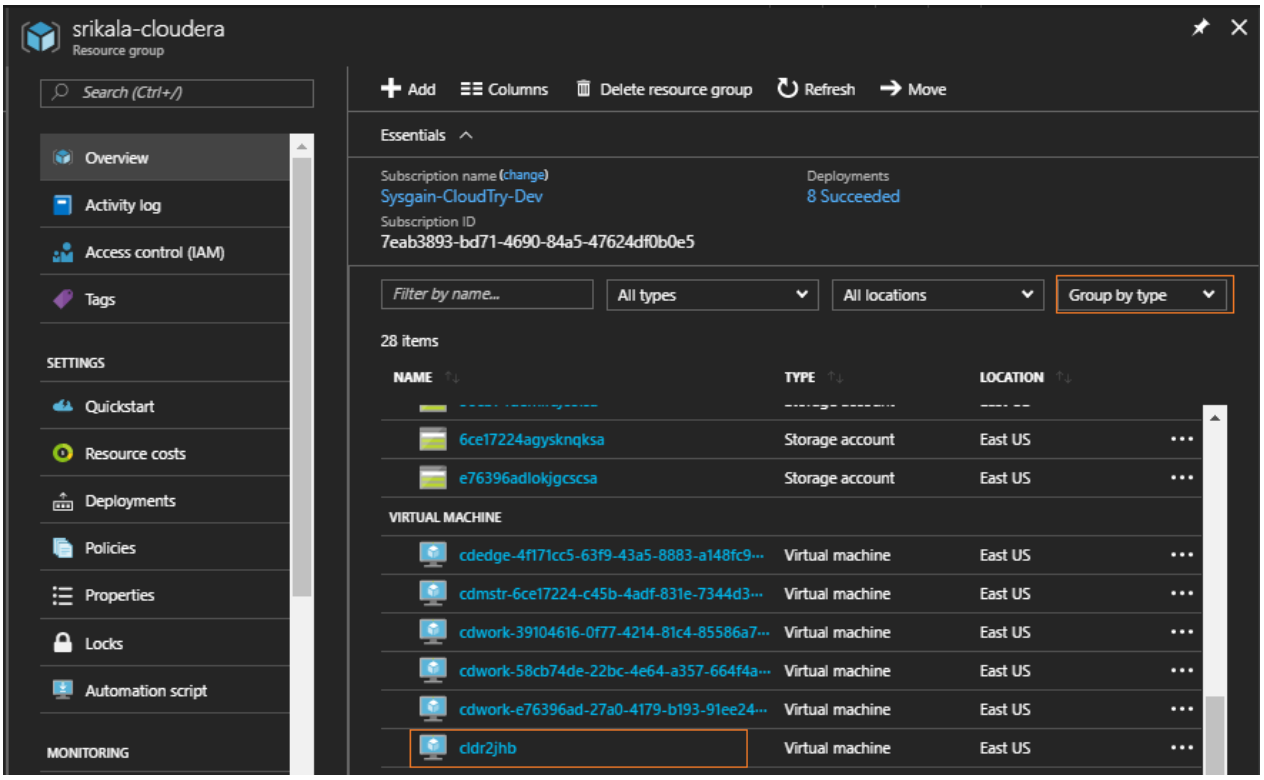
3.1 Accessing Cloudera Backend cluster details

Please login to the Azure portal and go to the Cloudera Director HOL Azure resource group allocated to you. Copy the DNS URLs for the **Cloudera Director**, **Manager** and **Master** nodes.

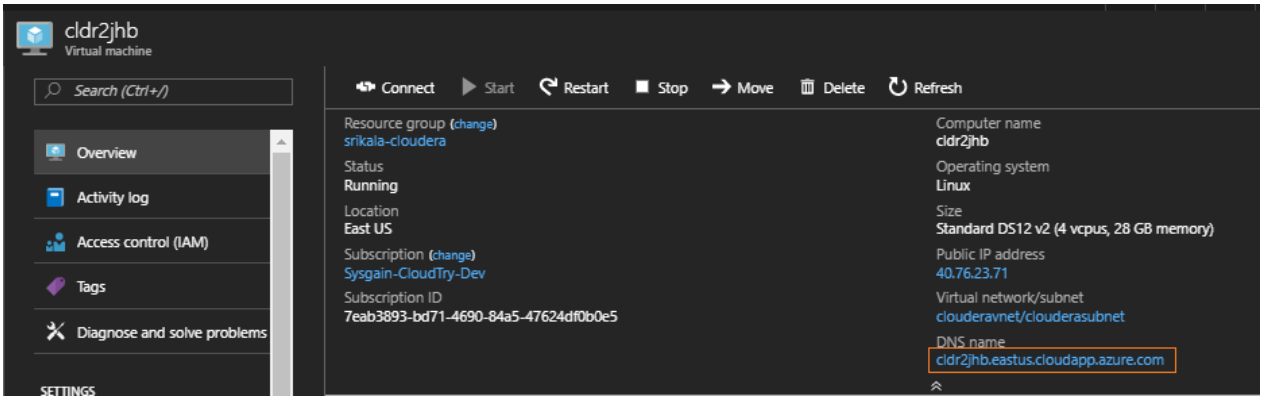
1. Go to the Resource Groups section and search by name for the Resource Group provided to you.



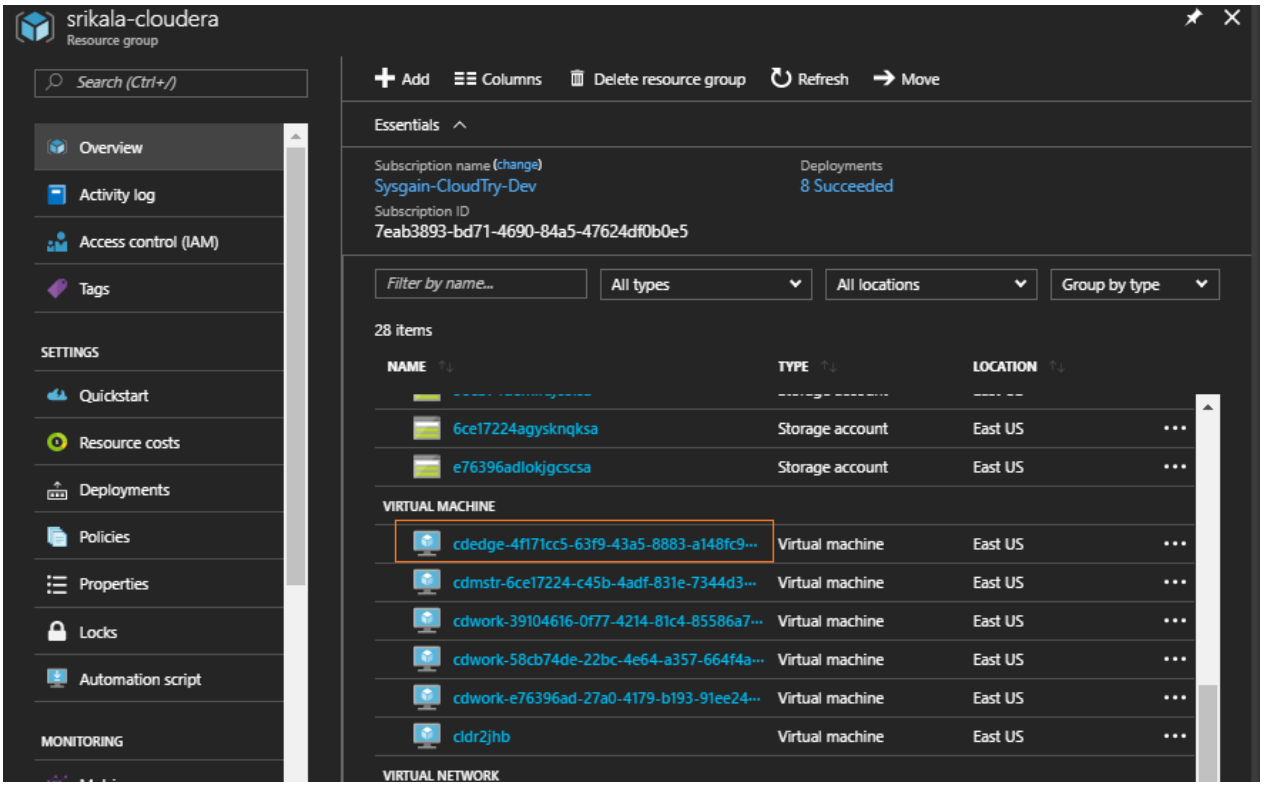
2. Go to the virtual machine starting with "**cldr**" for the **Cloudera Director DNS Name.**



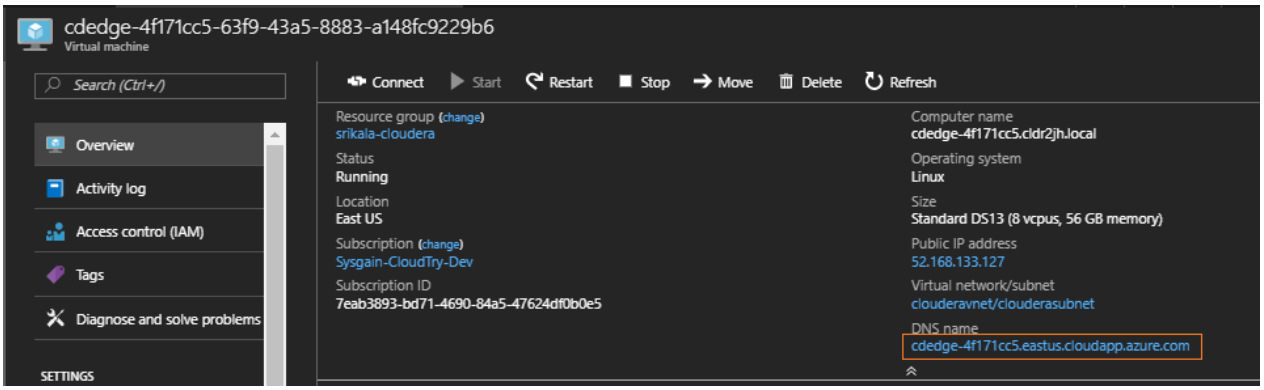
Click on the Cloudera Director virtual machine to get the DNS name. (See below)



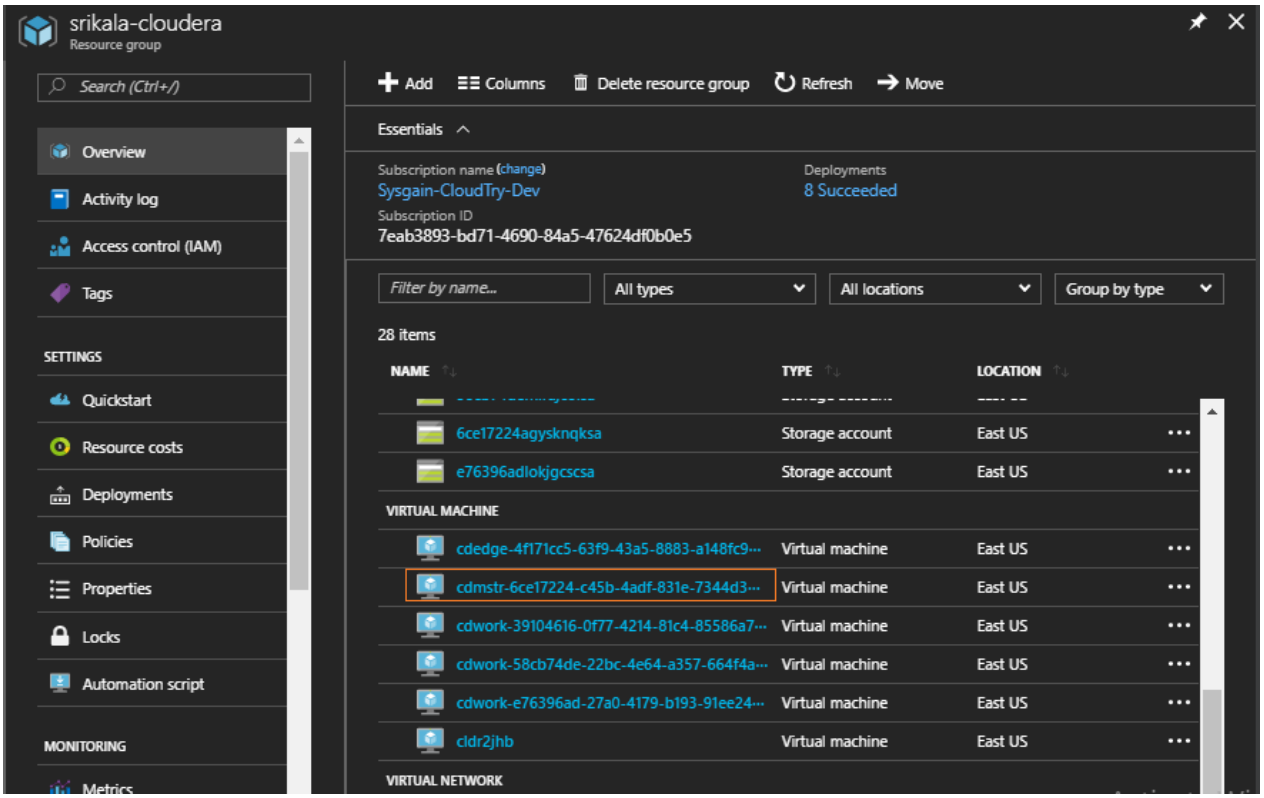
3. Go to the virtual machine starting with "**cdedge**" for the Cloudera Manager DNS name.



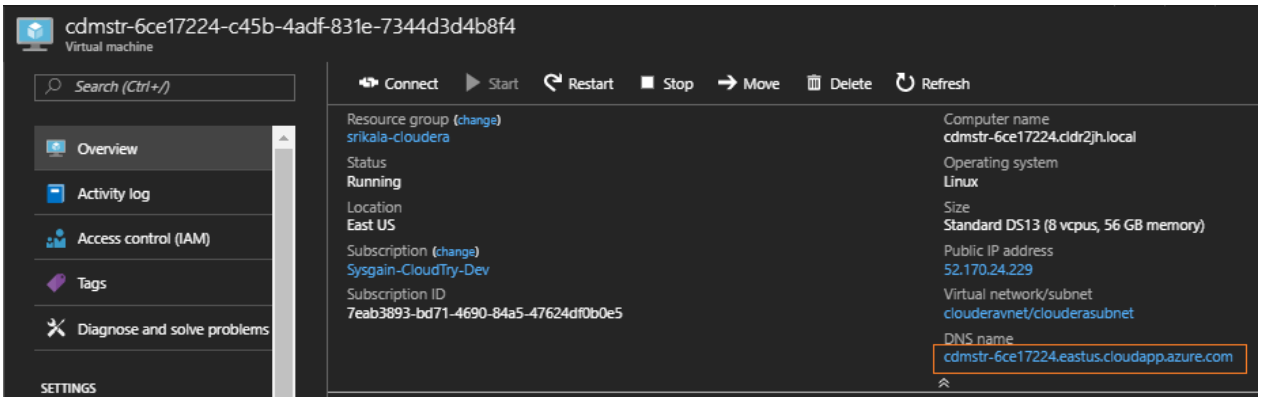
Click on the Cloudera Manager virtual machine to get the DNS name. (See below)



4. Go to the virtual machine starting with "**cdmstr**" for the Cloudera Master DNS name.



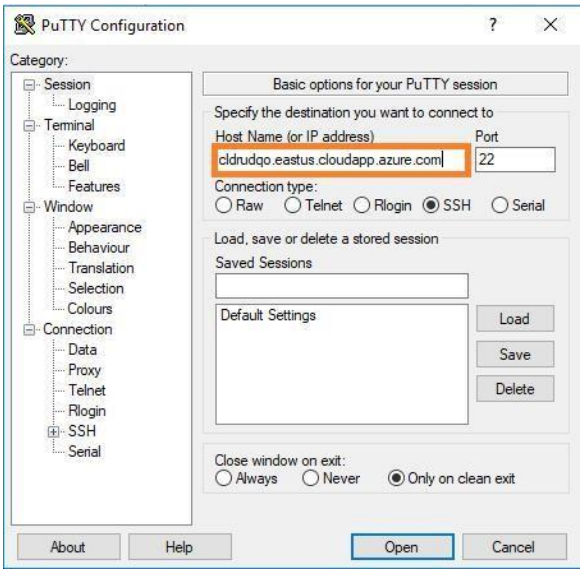
Click on the Cloudera Master virtual machine to get the DNS name. (See below)



You must also access the Cloudera backend cluster details to get the Node Details. This is explained below.

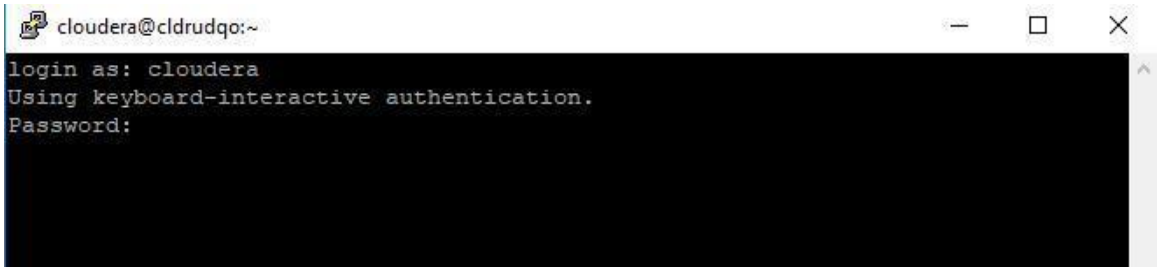
1. Log in to the Cloudera Director VM using the **Cloudera Director FQDN** address gathered from the previous steps, and use an SSH tool like PuTTY (or Terminal on Mac), which we'll refer to in this walkthrough. ([Download PuTTY here](#))

E.g. cldrhyic.eastus.cloudapp.azure.com



2. Once connected, login to the Cloudera Director VM using the **Director Username** and then the **Director Password** from the provided test drive access credentials.

   (**Note**: Passwords are hidden when typed or pasted in Linux terminals)

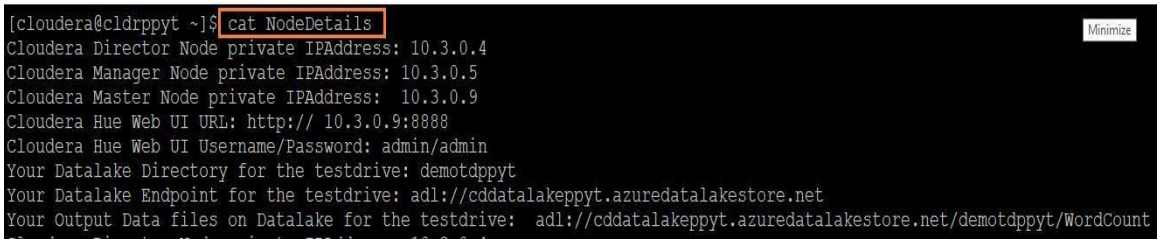


3. All the Cloudera Backend cluster details are present in the **NodeDetails** file. **Copy the NodeDetails into a text file or Word document for reference**, these details will be used later.

   To open the NodeDetails file use the following command.

```
cat NodeDetails
```

The NodeDetails file contains Node and URI details used by the Cloudera test drive environment. These are gathered using a script which pulls required data using the API calls.



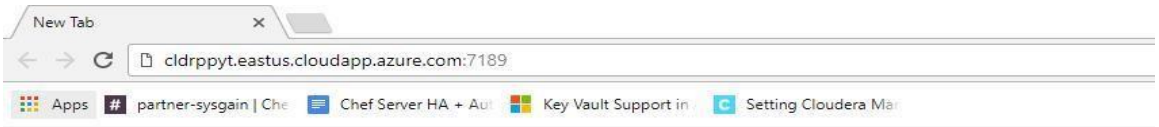
3.2. Accessing Cloudera Manager from Cloudera Director Web UI

After deploying a cluster, you can manage it using Cloudera Manager.

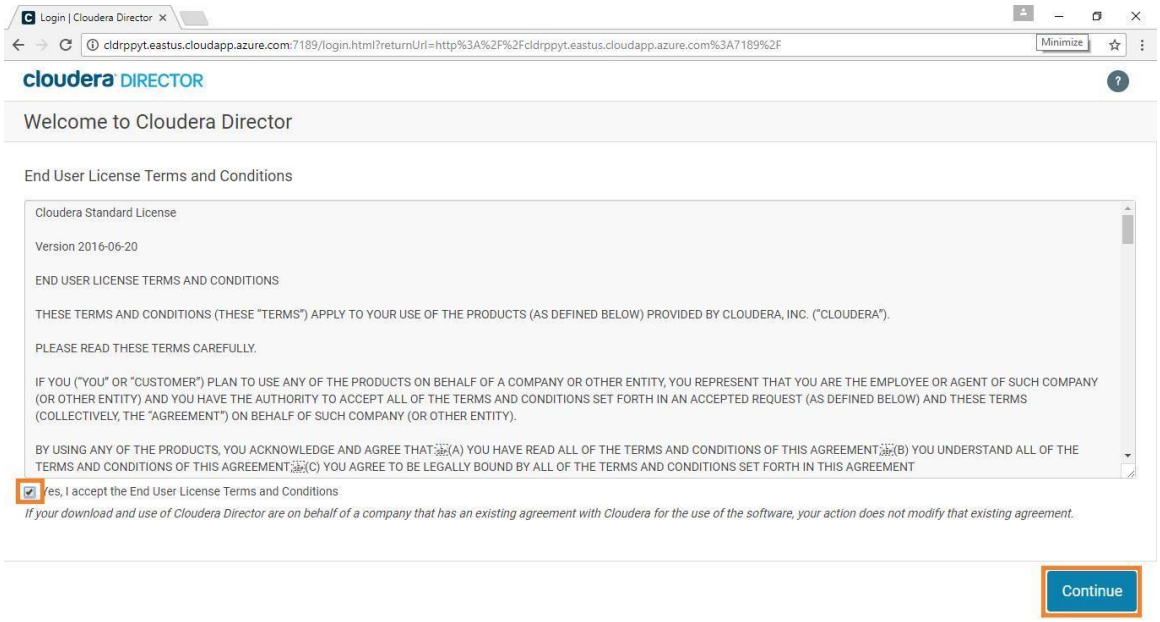
12

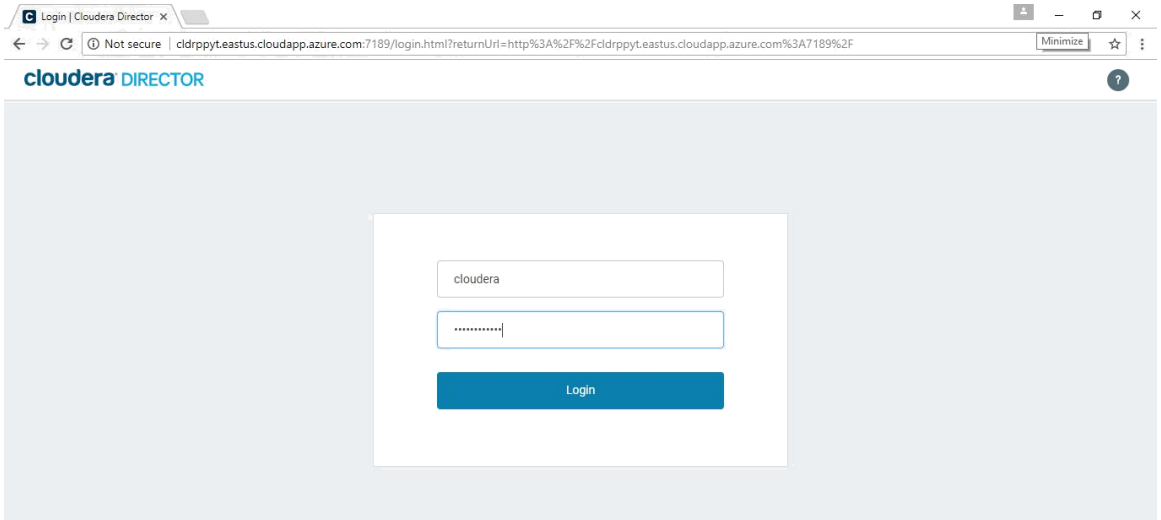
1. Access the Cloudera Director Web UI using the **Cloudera Director Access URL** provided in the Access Information. Enter it into a web browser.

   Eg**: cldrhyic.eastus.cloudapp.azure.com:7189**



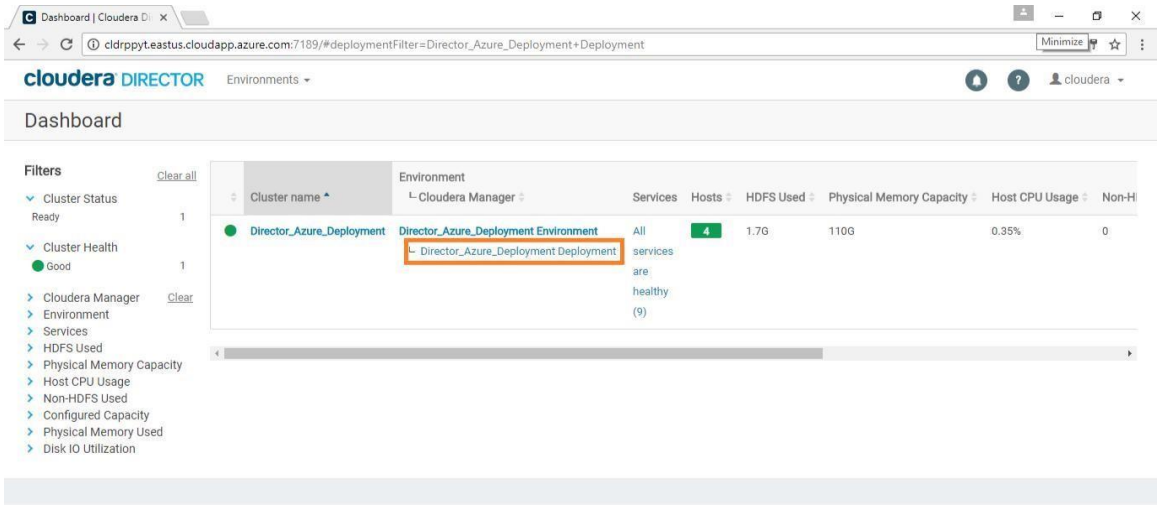
2. **Accept the End User License Terms and Conditions** and click on **Continue**.



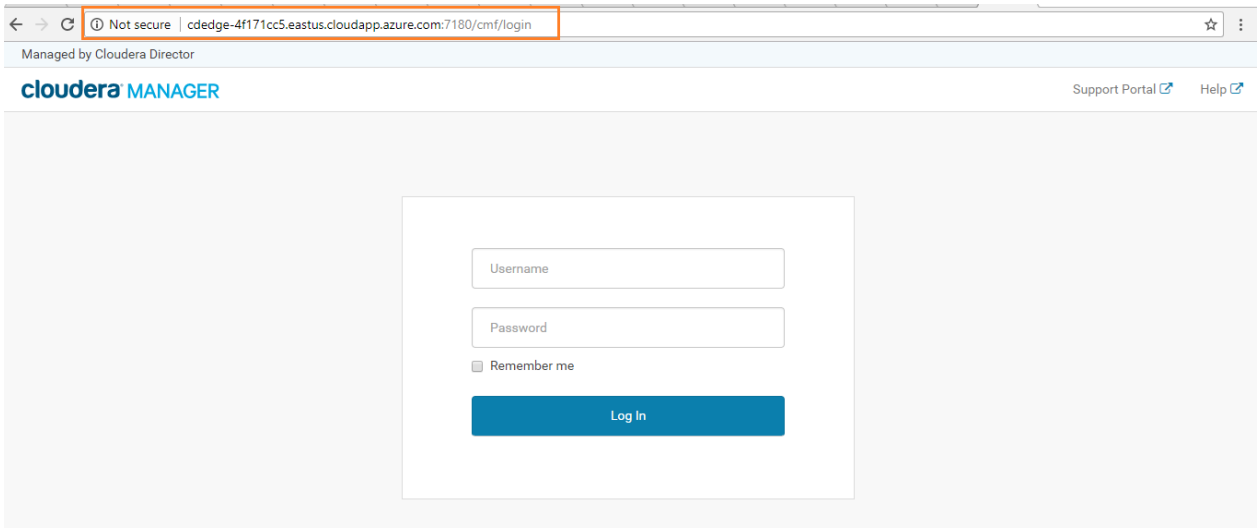


3. Login to the Cloudera Director web console using **CD-WEB UI Username** and **Password** from the Access Information.

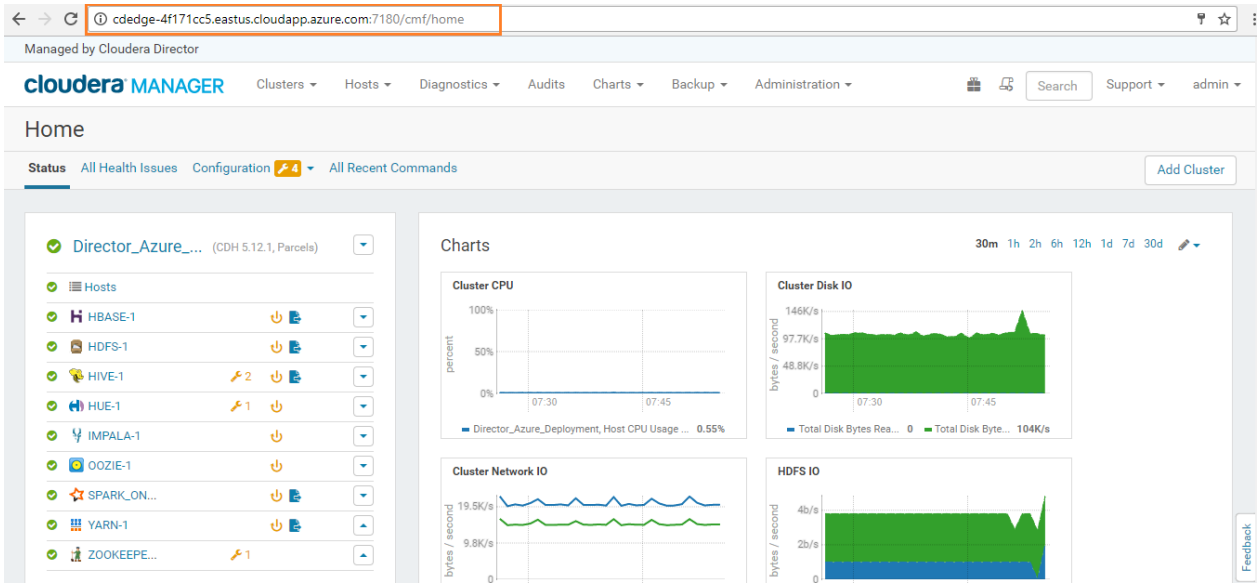
4. The Cloudera Director console should open. Click on the **Cloudera Manager** link from the **Cloudera Director** Dashboard, as shown below.



5. Use the Cloudera Manager FQDN address, along with the **port** number, and paste it in new browser tab.
   EX: cdedge-4f171cc5.eastus.cloudapp.azure.com:7180

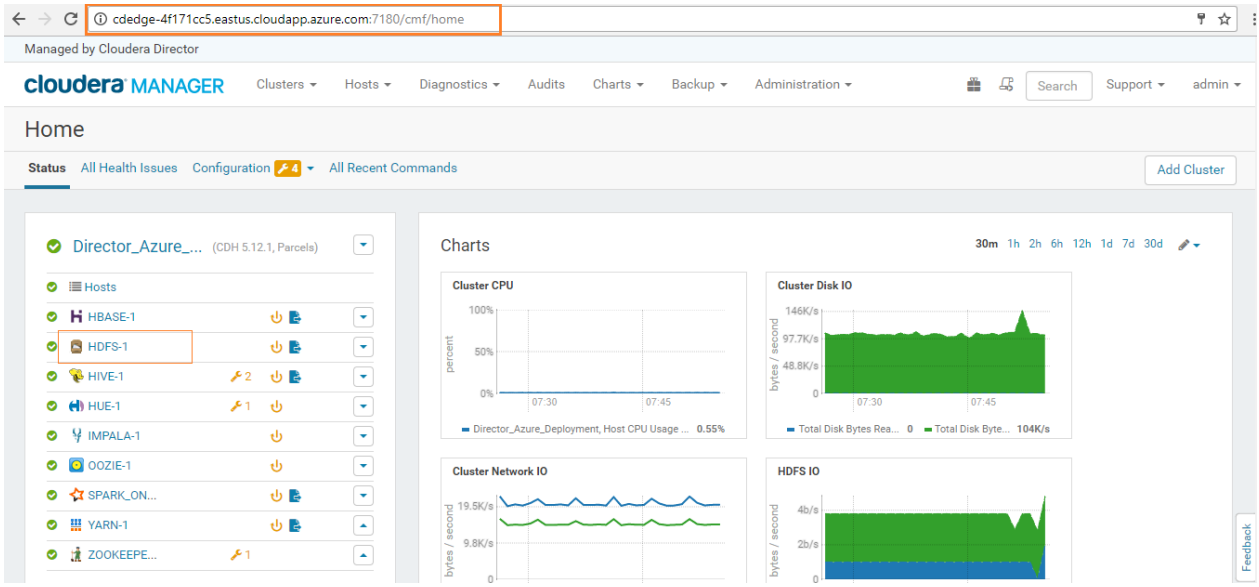


6. Login to the Cloudera Manager Console using **CM-WEB UI Username** and **CM-WEB UI Password** from the Access Information.

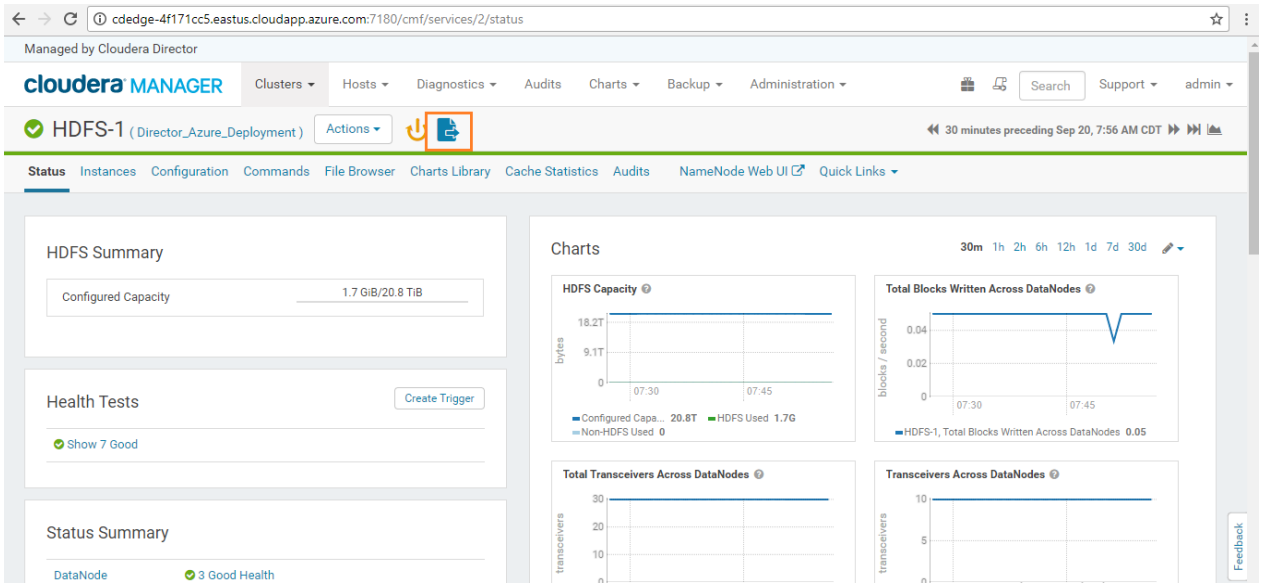


<u>Note:</u> The next step is to Restart Stale Services. We must do this to get the Azure Service Principle updated to the configuration file *site-core.xml*, which is required to integrate with Azure Data Lake Store.

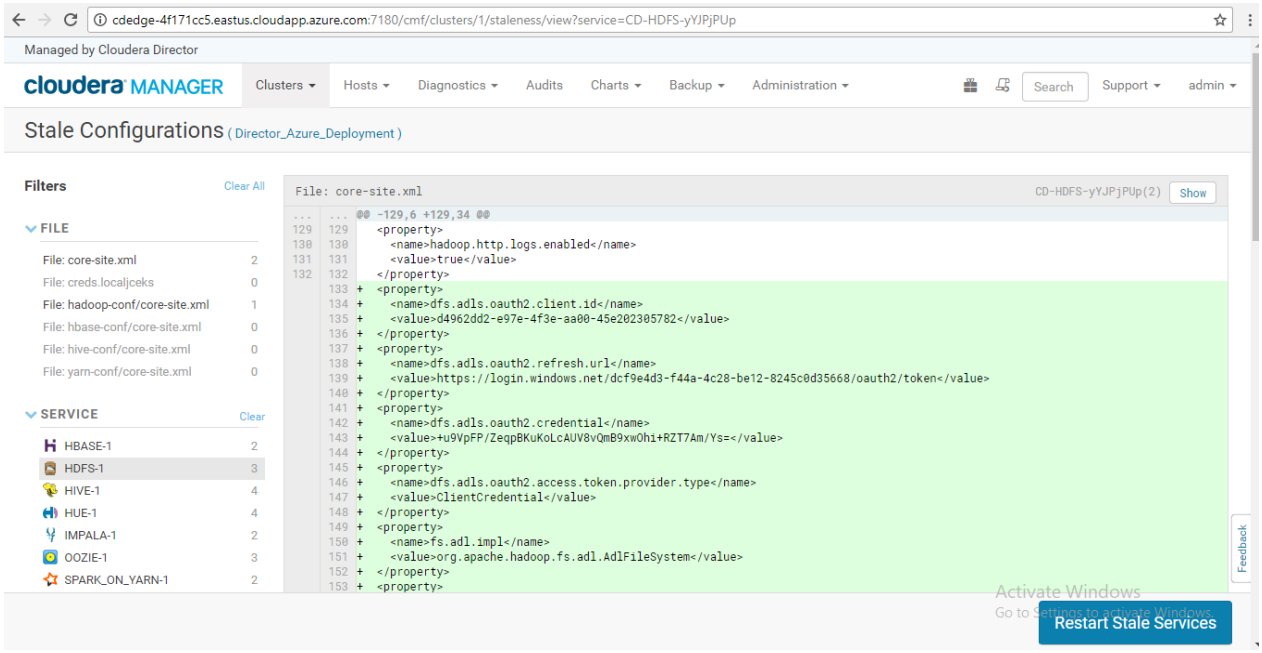
7. In Cloudera Manager, click on the **HDFS-1** service to **Restart Stale Services.**



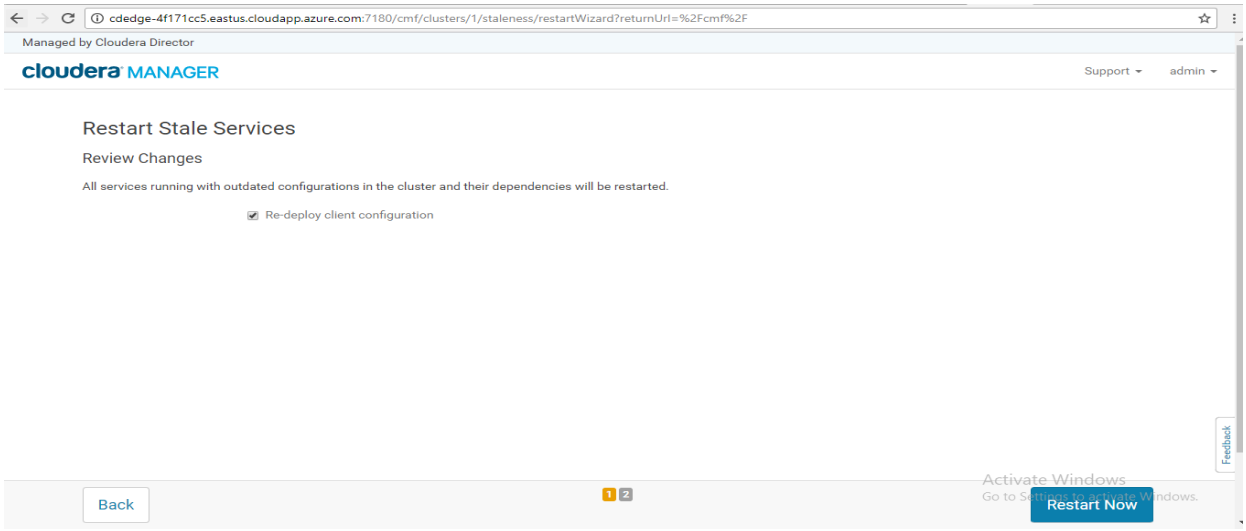
8. Click on the **Restart Stale Services** icon as shown in the below screenshot.



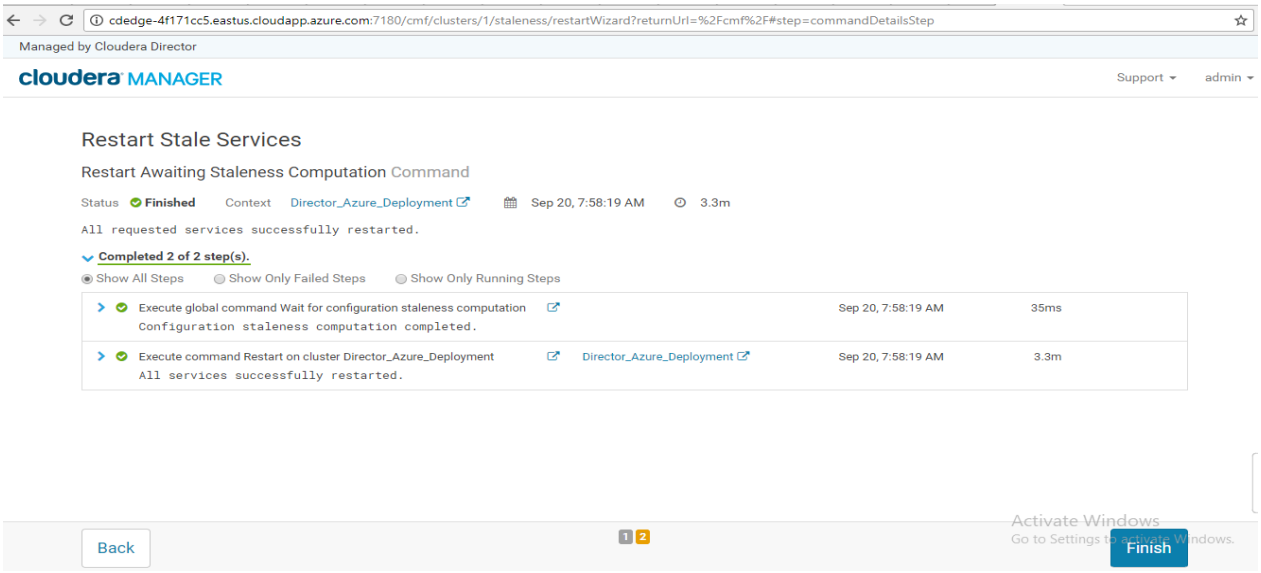
9. Click on the **Restart Stale Services** button so the cluster can read the new configuration information.



10. Click on the **Restart Now** button**.**



11. Wait until all requested services are restarted. Once all the services are restarted, click on the **Finish** button.



12. Now we have the **Cloudera Director** ready, with **Cloudera Manager** and **Cluster** (1 master and 3 workers).

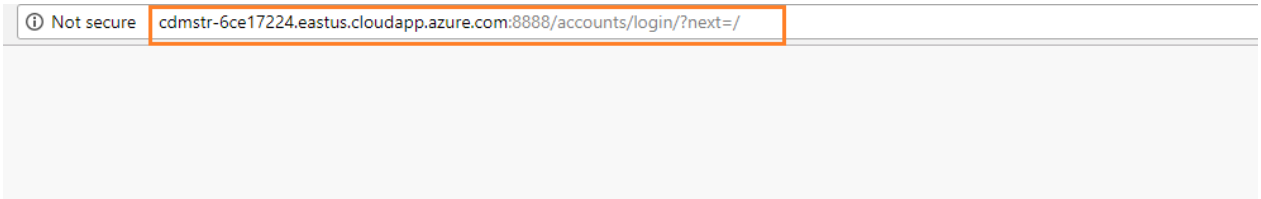
<u>Note:</u> Please visit section **5.1** in the ***Reference*** section later in this guide for additional details and help for any error messages you may encounter.

3.3. Hue

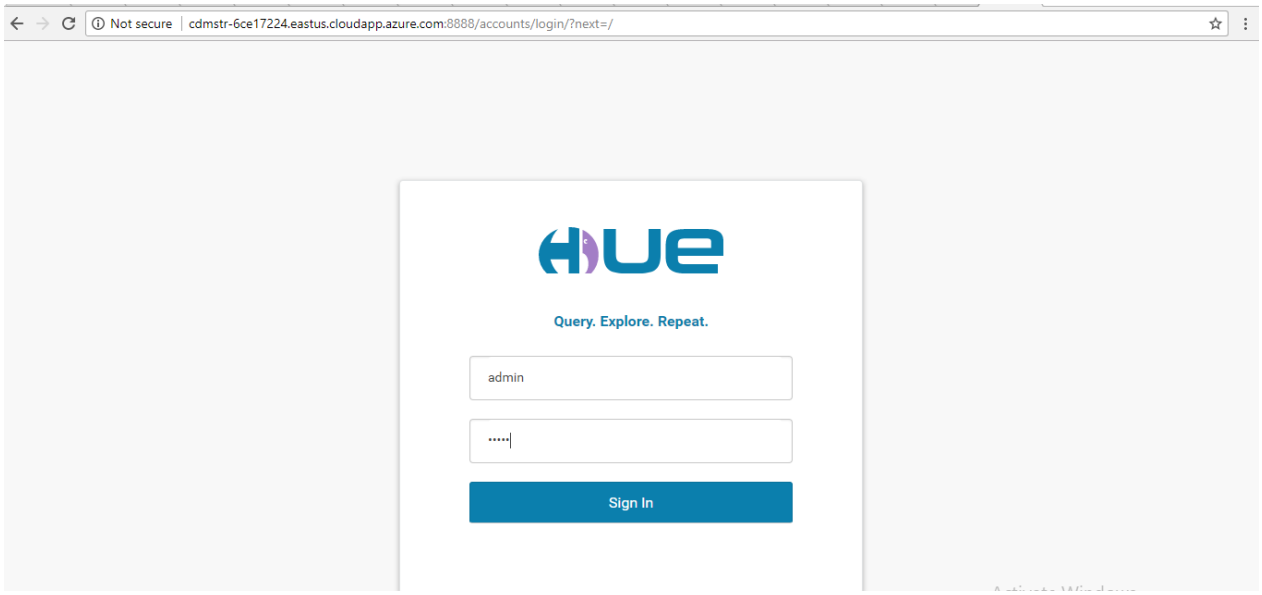
Hue is a set of web applications that enable you to interact with a CDH cluster. Hue applications let you browse HDFS and manage a Hive metastore. They also let you run Hive and Cloudera Impala queries, HBase and Sqoop commands, Pig scripts, MapReduce jobs, and Oozie workflows.

1. Copy the **Cloudera Hue Web URL** using the cloudera master DNS server url with port 8888 as shown in below example and paste it in browser – which opens the Hue console.

Example: http://cdmstr-6ce17224.eastus.cloudapp.azure.com:8888

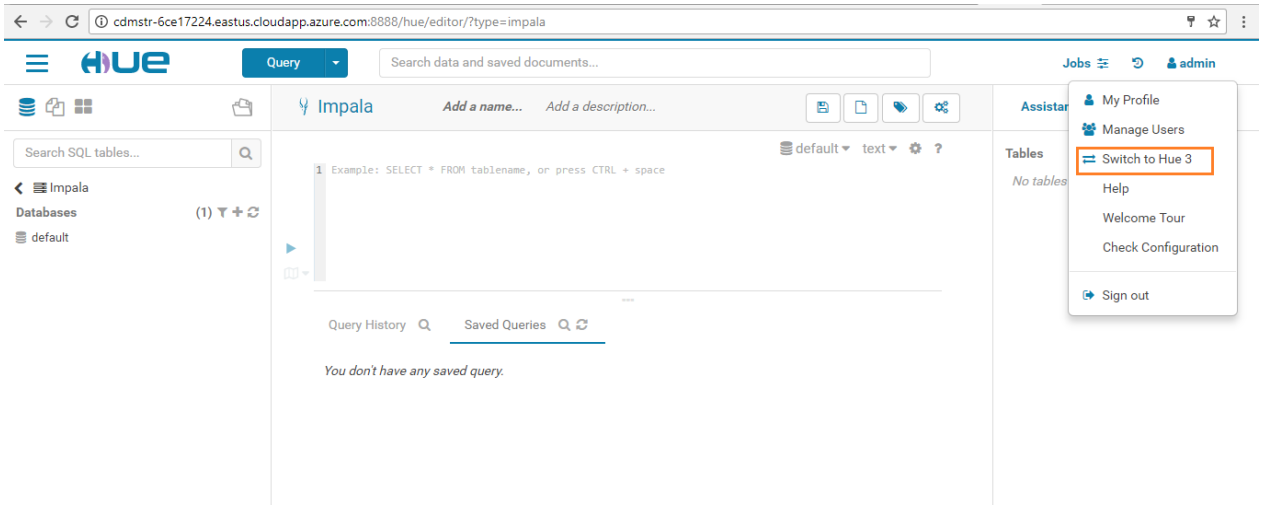


2. Create a Hue Account by giving **Cloudera Hue Web UI Username/Password** from the **NodeDetails** file. **(Username/Password: admin/admin)**



3. You will login into the Hue dashboard. On the right side of the page, click on the **HDFS browser** icon, as shown in the below screenshot.

Note: CDH 5.12 has a new Hue UI. We recommend switching to Hue 3 from the admin tab (see screenshot below).

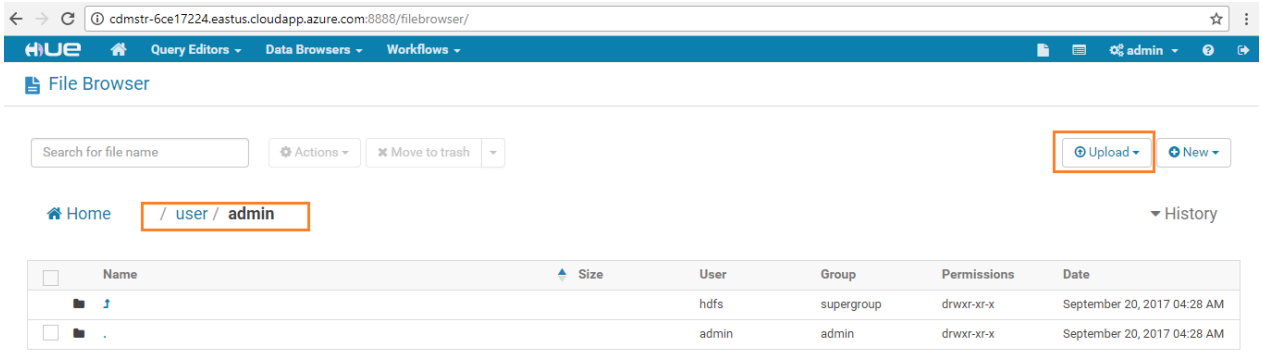


4. Copy the data of **inputfile** from the below link. Give any name to the file (Eg: '**data**' or '**input**'), then save it in **.txt** format.

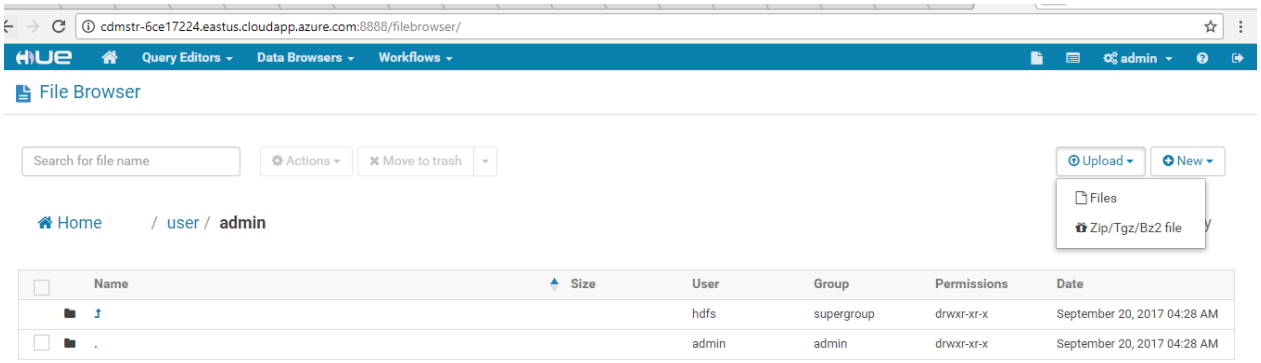
   https://aztdrepo.blob.core.windows.net/clouderadirector/inputfile.txt

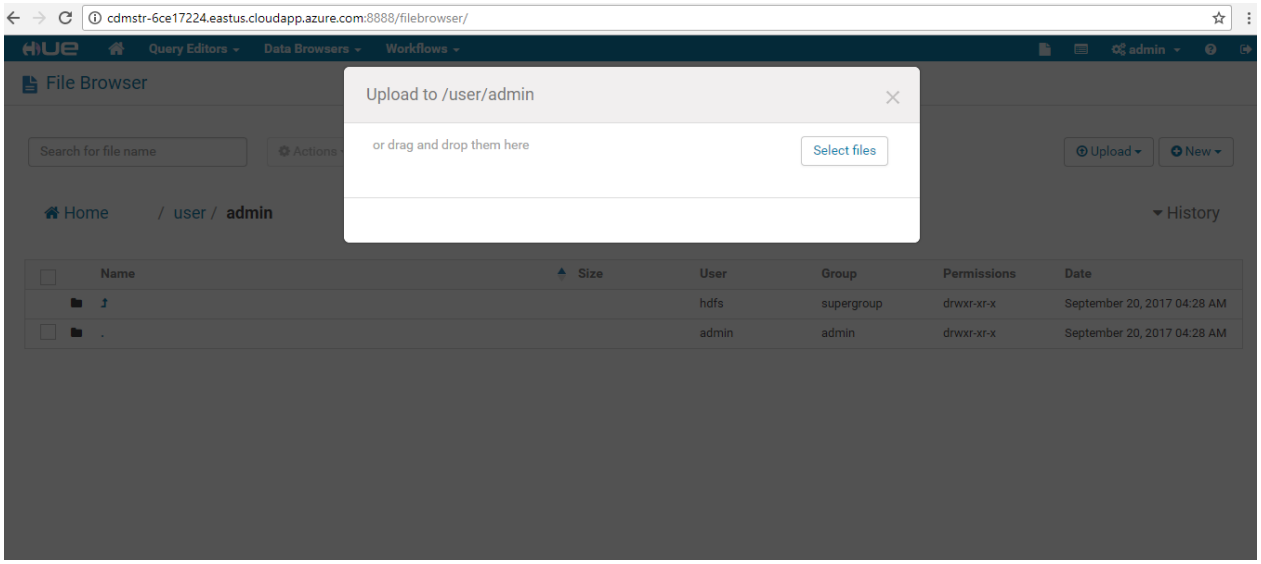
   Once ready, click on **Upload** on the Hue file browser page (see below).

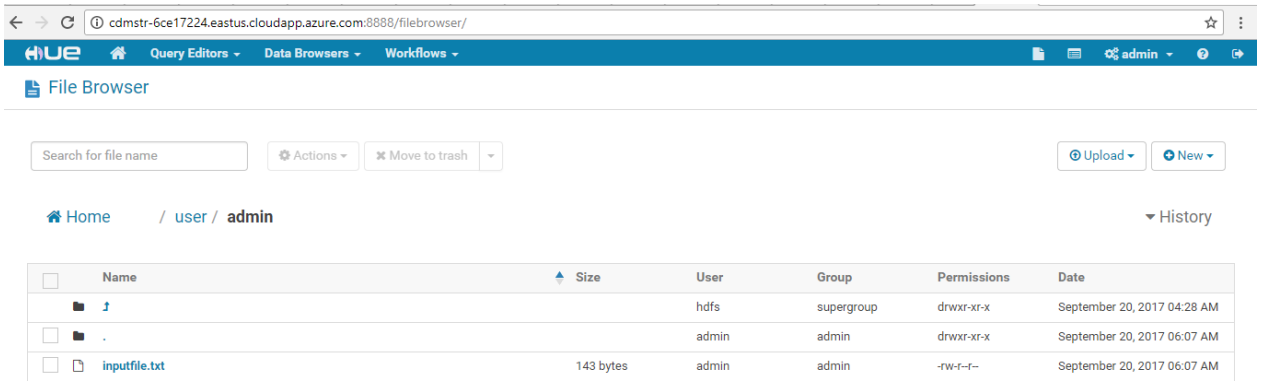
Note: Please ensure the inputfile is uploaded to the path **/user/admin** (see below):



5. Select the saved **.txt** file to upload it.







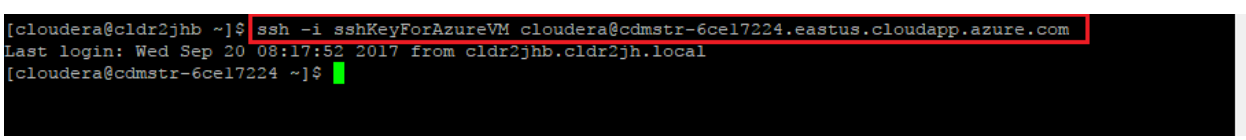
6. The .txt file is now uploaded to Hue. The Spark application will use this data as input and provide the output to ADLS.

3.4. Apache Spark (Run Spark App)

Spark is the open standard for flexible in-memory data processing that enables batch, realtime, and advanced analytics on the Apache Hadoop platform.
To use it properly, it is also a good idea to install "dos2unix". dos2unix is a program that converts DOS to UNIX text file format, ensuring everything will run in a Linux environment.

1. Login to the **Master VM** by typing in the below command in the open terminal session from before (copy/paste may not work):

```
ssh -i sshKeyForAzureVM cloudera@<Master Node FQDN>
```



2. **Download** the following script file using the below command**.**

The script contains the spark app (**WordCount**). The application counts the number of occurrences of each letter in words which have more characters than a given threshold.

wget https://raw.githubusercontent.com/sysgain/cloudera-spectra-vip/master/scripts/ClouderaSparkSetup.sh

```
cloudera@cdmstr-6ce17224:~
[cloudera@cdmstr-6ce17224 ~]$ exit
logout
Connection to 10.3.0.9 closed.
[cloudera@cldr2jhb ~]$ ssh -i sshKeyForAzureVM cloudera@cdmstr-6ce17224.eastus.cloudapp.azure.com
Last login: Wed Sep 20 08:17:52 2017 from cldr2jhb.cldr2jh.local
[cloudera@cdmstr-6ce17224 ~]$ wget https://raw.githubusercontent.com/sysgain/clouderatd/master/scripts/ClouderaS
parkSetup.sh
--2017-09-20 08:39:21--  https://raw.githubusercontent.com/sysgain/clouderatd/master/scripts/ClouderaSparkSetup.
sh
Resolving raw.githubusercontent.com... 151.101.32.133
Connecting to raw.githubusercontent.com|151.101.32.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 815 [text/plain]
Saving to: "ClouderaSparkSetup.sh"

100%[===================================================================>] 815         --.-K/s   in 0s

2017-09-20 08:39:21 (162 MB/s) - "ClouderaSparkSetup.sh" saved [815/815]

[cloudera@cdmstr-6ce17224 ~]$
```

3. To install **dos2unix,** run the following command:

```
sudo yum install -y dos2unix
```

```
[cloudera@cdmstr-9990c974 ~]$ sudo yum install -y dos2unix
Loaded plugins: fastestmirror, security
Setting up Install Process
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package dos2unix.x86_64 0:3.1-37.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved
```

4. To give permissions to **ClouderaSparkSetup.sh** file, run the following commands:

```
dos2unix /home/cloudera/ClouderaSparkSetup.sh
chmod 755 /home/cloudera/ClouderaSparkSetup.sh
```

```
cloudera@cdmstr-6ce17224:~
Installed:
  dos2unix.x86_64 0:3.1-37.el6

Complete!
[cloudera@cdmstr-6ce17224 ~]$ dos2unix /home/cloudera/ClouderaSparkSetup.sh
dos2unix: converting file /home/cloudera/ClouderaSparkSetup.sh to UNIX format ...
[cloudera@cdmstr-6ce17224 ~]$ chmod 755 /home/cloudera/ClouderaSparkSetup.sh
[cloudera@cdmstr-6ce17224 ~]$ sh ClouderaSparkSetup.sh demotd2jhb cdmstr-6ce17224.eastus.cloudapp.azure.com inputfile.txt adl://cddatalake
2jhb.azuredatalakestore.net
--2017-09-20 08:42:52--  https://aztdrepo.blob.core.windows.net/clouderadirector/wordcount.jar
Resolving aztdrepo.blob.core.windows.net... 52.238.56.168
Connecting to aztdrepo.blob.core.windows.net|52.238.56.168|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6371588 (6.1M) [application/octet-stream]
Saving to: "/home/cloudera/wordcount.jar"

100%[===================================================================================================>] 6,371,588   5.05M/s   in 1.2s

2017-09-20 08:42:54 (5.05 MB/s) - "/home/cloudera/wordcount.jar" saved [6371588/6371588]

17/09/20 08:42:55 INFO spark.SparkContext: Running Spark version 1.6.0
17/09/20 08:42:56 INFO spark.SecurityManager: Changing view acls to: cloudera
17/09/20 08:42:56 INFO spark.SecurityManager: Changing modify acls to: cloudera
17/09/20 08:42:56 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set
(cloudera); users with modify permissions: Set(cloudera)
```

5. Run the following command to execute the **ClouderaSparkSetup.sh** script**:**

```
sh ClouderaSparkSetup.sh <Datalake Directory> <Master Node FQDN>
<inputfile.txt> <Datalake Endpoint for the testdrive>
```

Note: Replace the above values from **NodeDetails** and give the Name of the input file that you have just uploaded in Hue in the place of ***<inputfile.txt>.***

Example:
sh ClouderaSparkSetup.sh demotdah6k cdmstr-6ce17224.eastus.cloudapp.azure.com inputfile.txt adl://cddatalakeah6k.azuredatalakestore.net


```
cloudera@cdmstr-6ce17224:~
Installed:
  dos2unix.x86_64 0:3.1-37.el6

Complete!
[cloudera@cdmstr-6ce17224 ~]$ dos2unix /home/cloudera/ClouderaSparkSetup.sh
dos2unix: converting file /home/cloudera/ClouderaSparkSetup.sh to UNIX format ...
[cloudera@cdmstr-6ce17224 ~]$ chmod 755 /home/cloudera/ClouderaSparkSetup.sh
[cloudera@cdmstr-6ce17224 ~]$ sh ClouderaSparkSetup.sh demotd2jhb cdmstr-6ce17224.eastus.cloudapp.azure.com inputfile.txt adl://cddatalake
2jhb.azuredatalakestore.net
--2017-09-20 08:42:52--  https://aztdrepo.blob.core.windows.net/clouderadirector/wordcount.jar
Resolving aztdrepo.blob.core.windows.net... 52.238.56.168
Connecting to aztdrepo.blob.core.windows.net|52.238.56.168|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6371588 (6.1M) [application/octet-stream]
Saving to: "/home/cloudera/wordcount.jar"

100%[===================================================================================================>] 6,371,588   5.05M/s   in 1.2s

2017-09-20 08:42:54 (5.05 MB/s) - "/home/cloudera/wordcount.jar" saved [6371588/6371588]

17/09/20 08:42:55 INFO spark.SparkContext: Running Spark version 1.6.0
17/09/20 08:42:56 INFO spark.SecurityManager: Changing view acls to: cloudera
17/09/20 08:42:56 INFO spark.SecurityManager: Changing modify acls to: cloudera
17/09/20 08:42:56 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set
(cloudera); users with modify permissions: Set(cloudera)
```


6. By executing the above script, the data has been stored to ADLS using Spark application.

Note: Please visit section **5.2** in the ***Reference*** section for additional details and help for any error messages you may encounter.

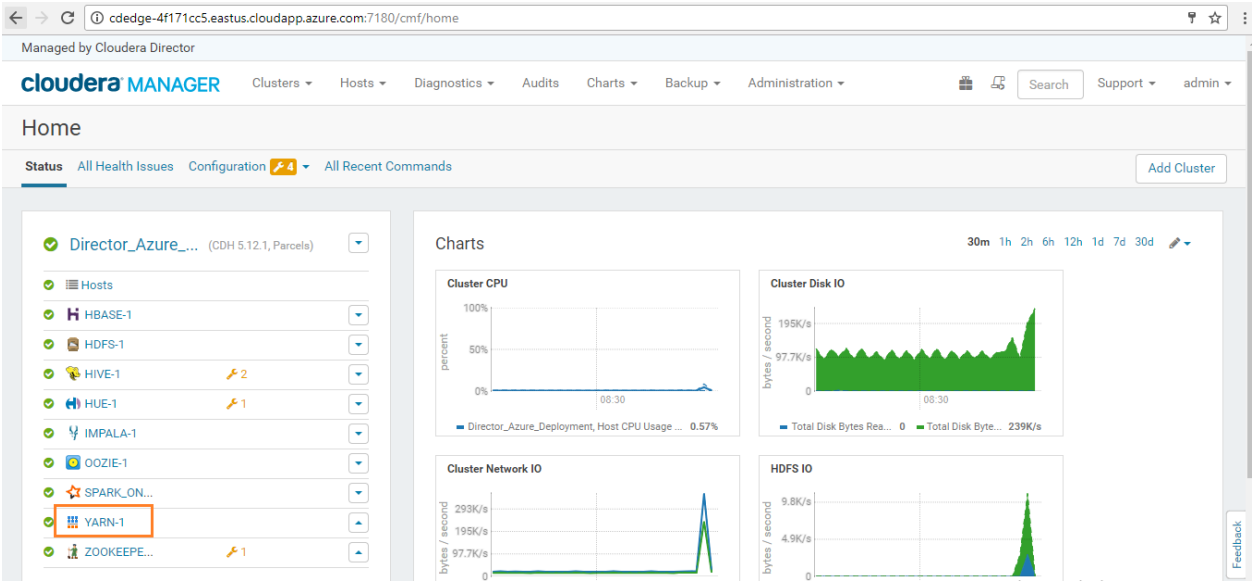
3.5. Viewing Jobs in UI

Next, navigate to the Yarn/Spark UI to see the WordCount Spark job.

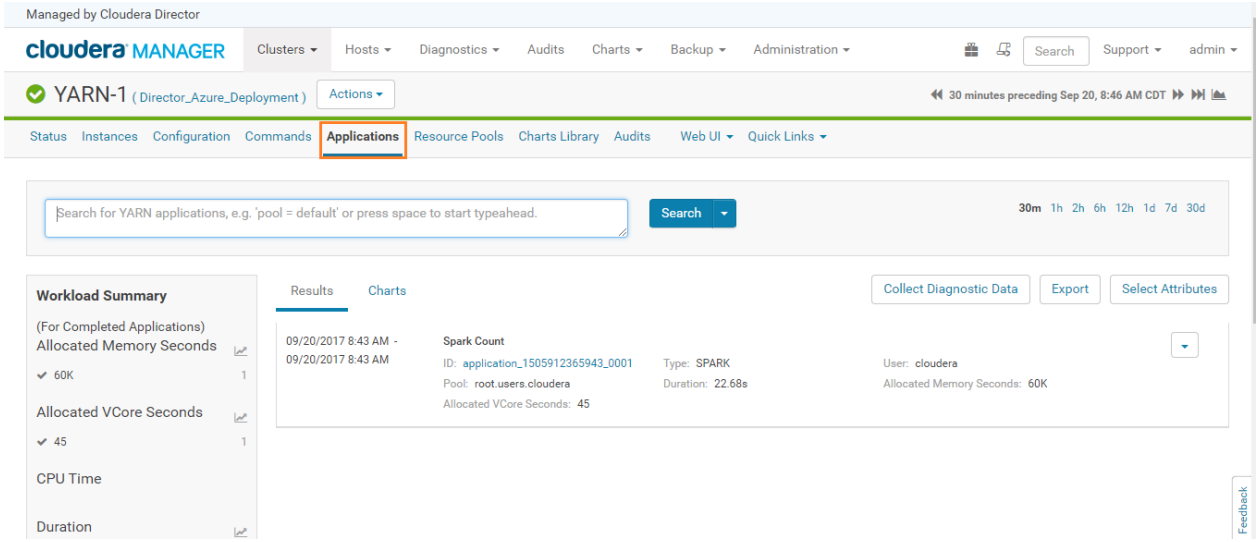
1. Go to http://***<Manager Node FQDN***>:7180/cmf/home

   Example: ***http://cdedge-4f171cc5.eastus.cloudapp.azure.com:7180***

2. Click on **YARN-1**.



3. Click on the **Applications** tab in the top navigation menu to view the available jobs.



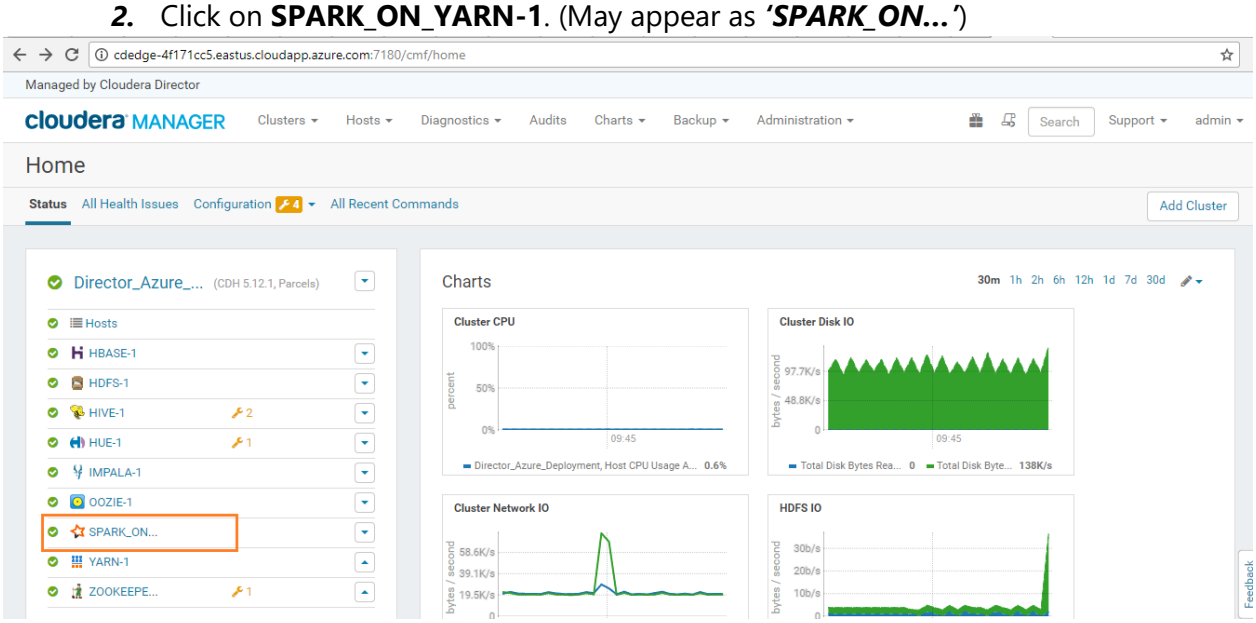
Each job has Summary and Detail information. A job Summary includes the following attributes: **start & end timestamps**, **query name** (if the job is part of a Hive query), **queue**, **job type**, **job ID**, and **user**.

4. You can also see the available applications by navigating to the Spark UI:

1. Go to http://<***Manager Node private FQDN***>:7180/cmf/home

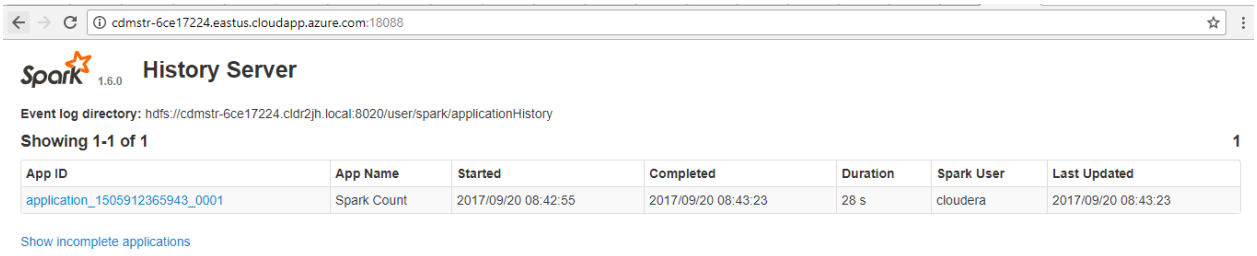
Example: ***http://cdedge-4f171cc5.eastus.cloudapp.azure.com:7180***

2. Click on **SPARK_ON_YARN-1**. (May appear as ***'SPARK_ON...'***)



3. Navigate to the **History Server WEB UI** by going to http://***<Master FQDN***>:18088

Example: ***http:// cdedge-4f171cc5.eastus.cloudapp.azure.com:18088/***



Note: Please visit section **5.2** in the ***Reference*** section for additional details and help for any error messages you may encounter.

3.6. Hive

Apache Hive is a data warehouse software project built on top of Apache Hadoop for providing data summarization, query, and analysis. Hive gives a SQL-like interface to query data stored in various databases and file systems that integrate with Hadoop.

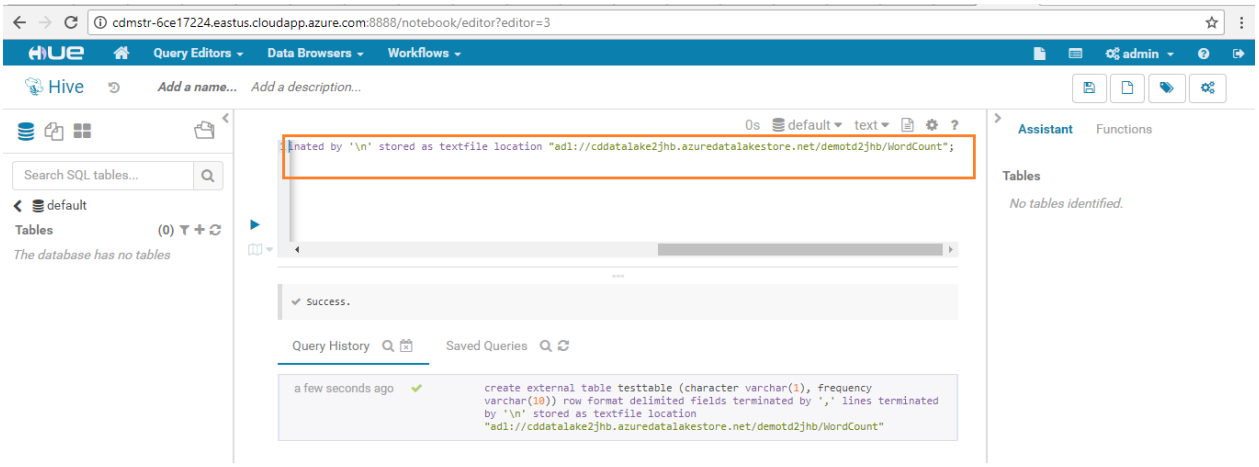
Now we will create a Hive table from the output of the Spark application stored on ADLS and run a Hive query from Hue.

1. Navigate to the **Query Editors** drop-down menu in the Hue WEB UI and click on **Hive**.

2. In the default database, execute the below query:

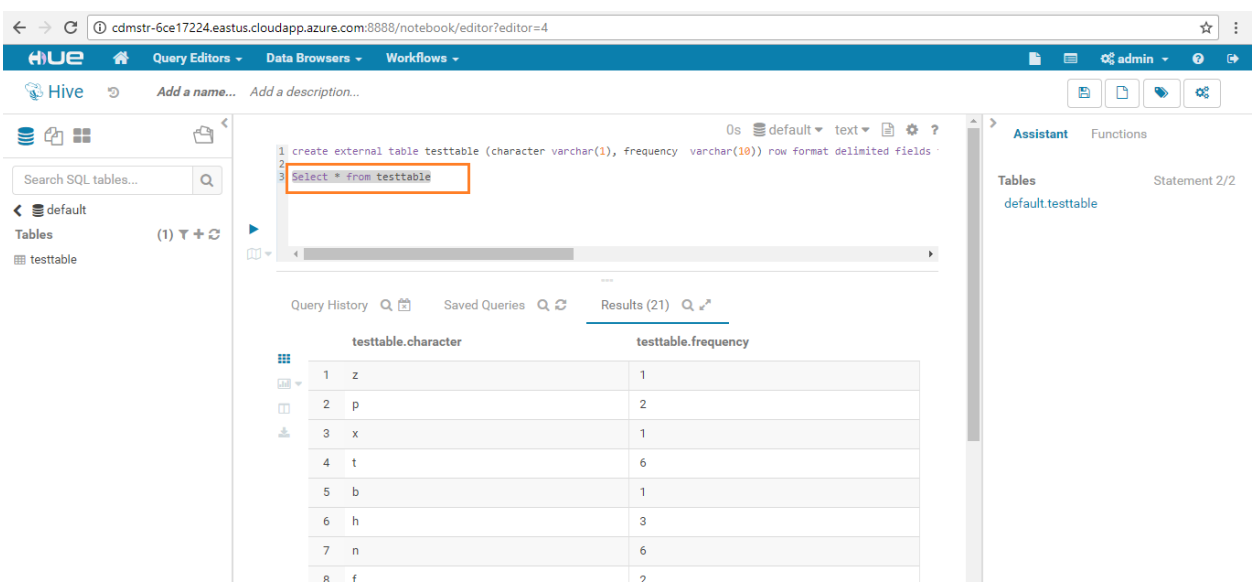
```
create external table <tablename> (character varchar(1), frequency
varchar(10)) row format delimited fields terminated by ',' lines
terminated by '\n' stored as textfile location "<Output Data files on
Datalake for the testdrive>";
```

<u>Note</u>: Add any name for ***<tablename>*** and replace the <***Output Data files on Datalake for the testdrive***> placeholder with the corresponding data from the **NodeDetails** file.



3. View the table by giving the query:

```
Select * from <tablename>
```

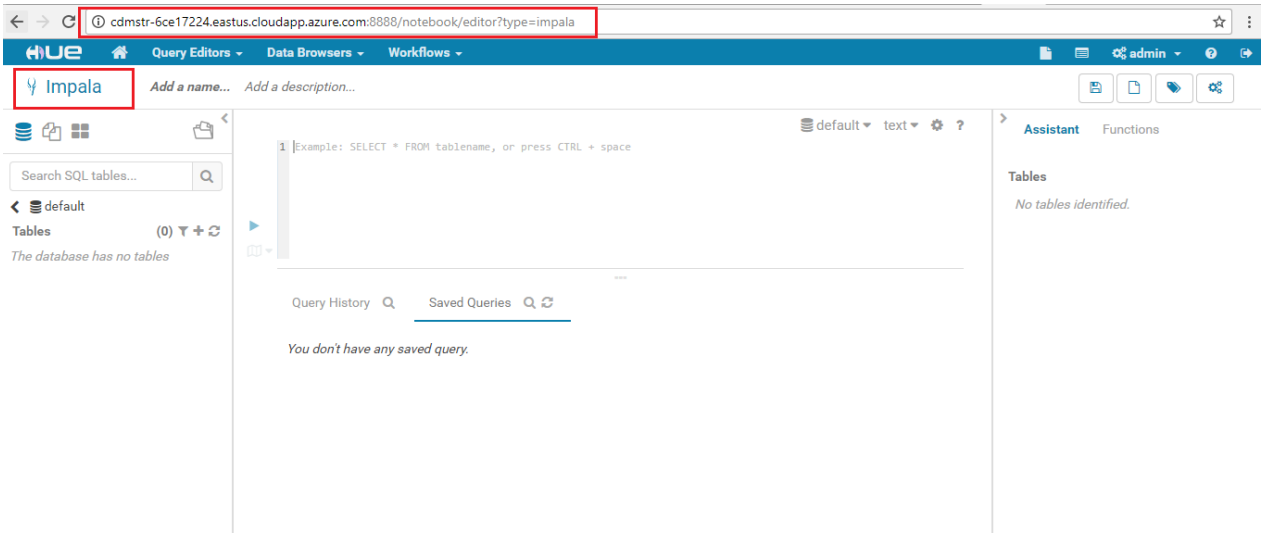


3.7. Impala

Impala is an open source, massively parallel processing query engine on top of clustered systems like Apache Hadoop. It is an interactive SQL like query engine that runs on top of Hadoop Distributed File System (HDFS). It integrates with HIVE metastore to share the table information between both the components.

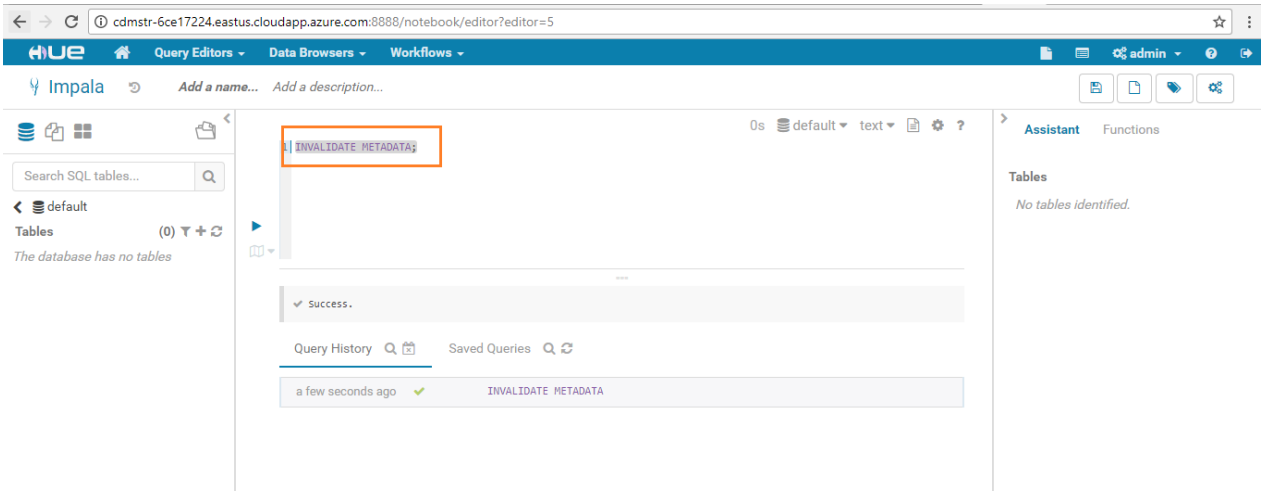
1. **Note**: Impala now integrates with ADLS from version CDH 5.12.

2. Navigate to the **Query Editor** drop-down menu and click on **Impala**.



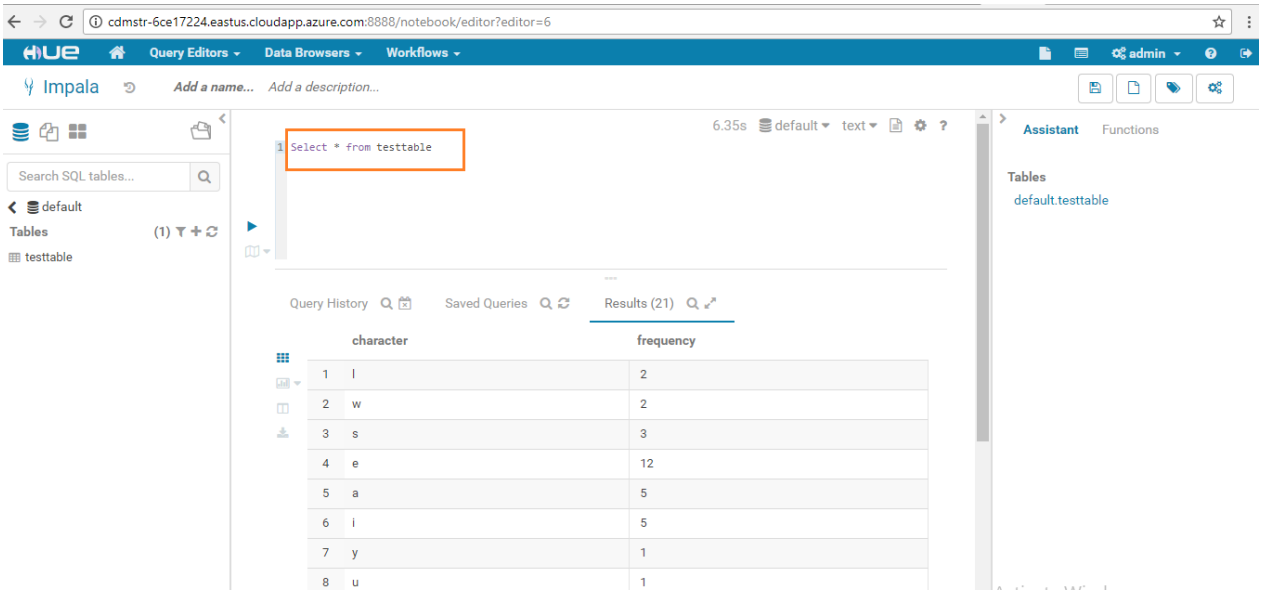
3. Execute the below query in the default database to sync the data from Hive to Impala:

```
INVALIDATE METADATA;
```



4. View the table by giving the query:

```
Select * from <tablename>
```



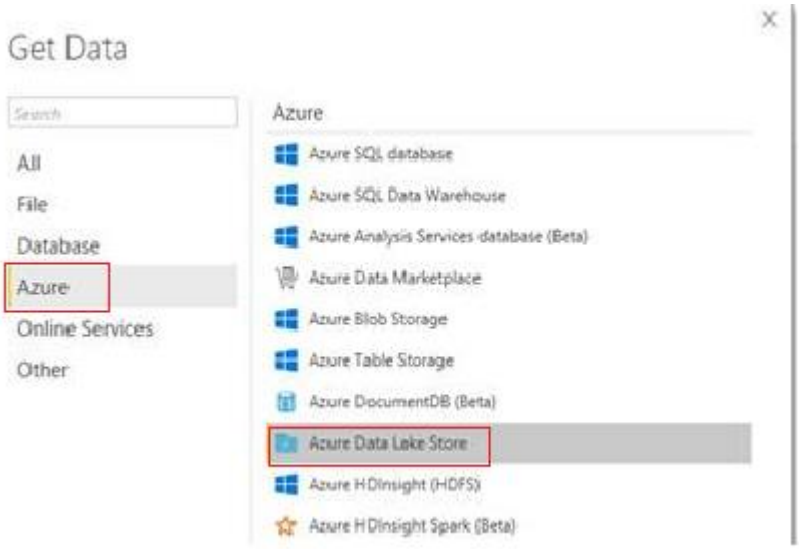
5. You have now successfully run the Impala query using Hue!

4. Power BI integration with Data Lake Store and Impala (Optional)

4.1 Integrating with Data Lake Store

1. Launch **Power BI Desktop** on your computer.

2. From the Home ribbon, click Get Data, and then click More. In the Get Data dialog box, click Azure, click Azure Data Lake Store, and then click Connect.



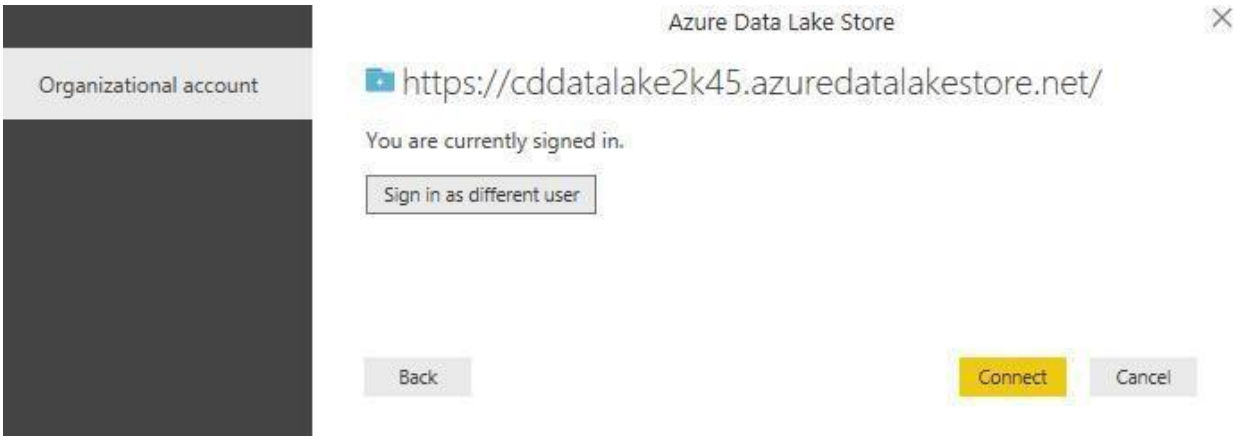
3. In the Microsoft Azure Data Lake Store dialog box, provide the **URL to your Data Lake Store account**, and then click **OK**.

<u>Note</u>: Get the ***URL - Datalake Endpoint*** from the NodeDetails file. (Refer to section **4.1**)

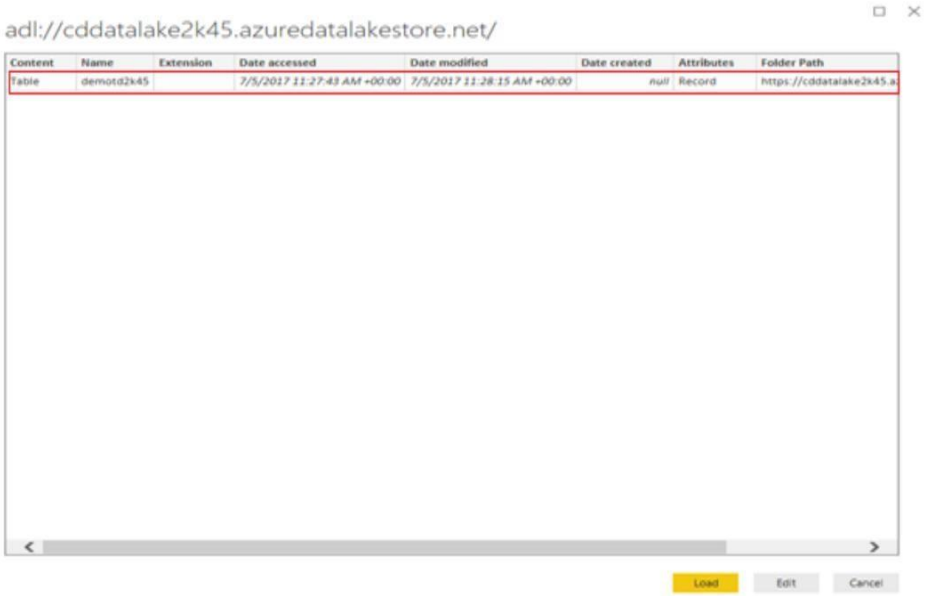
.

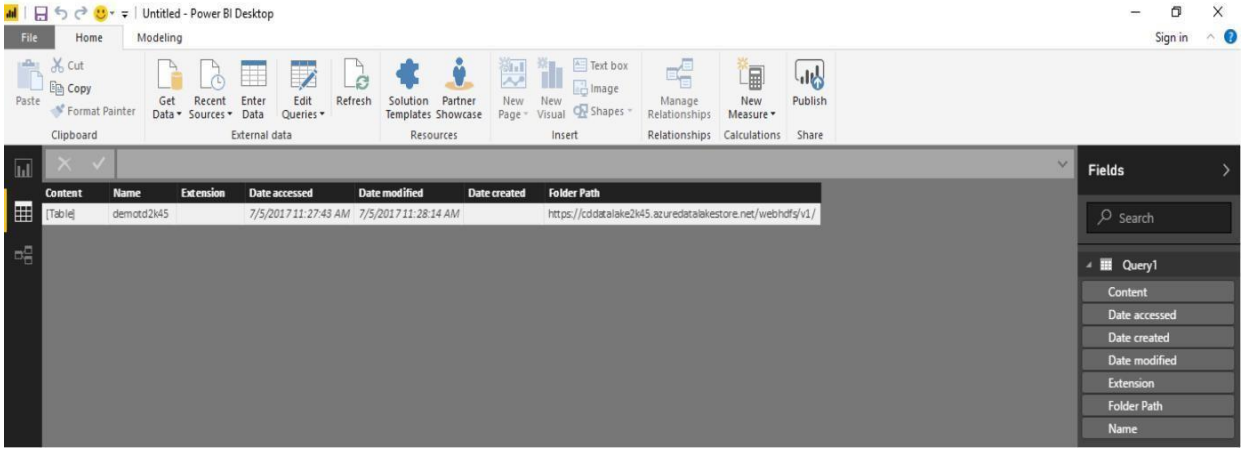
4. In the next dialog box, click **Sign in** to sign into Data Lake Store account. You will be redirected to your organization's sign in page. **Follow the prompts** to sign into the account.

After you have successfully signed in, click **Connect**.

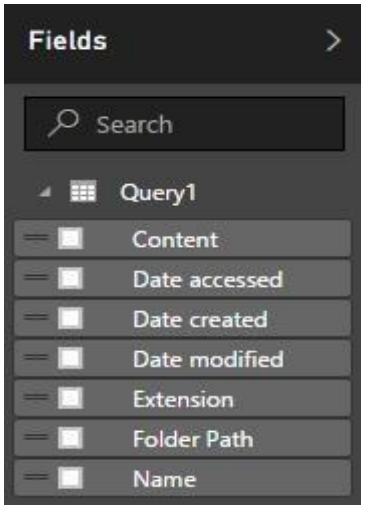


5. The next dialog box shows the file that you uploaded to your Data Lake Store account. **Verify** the info and then click **Load**.





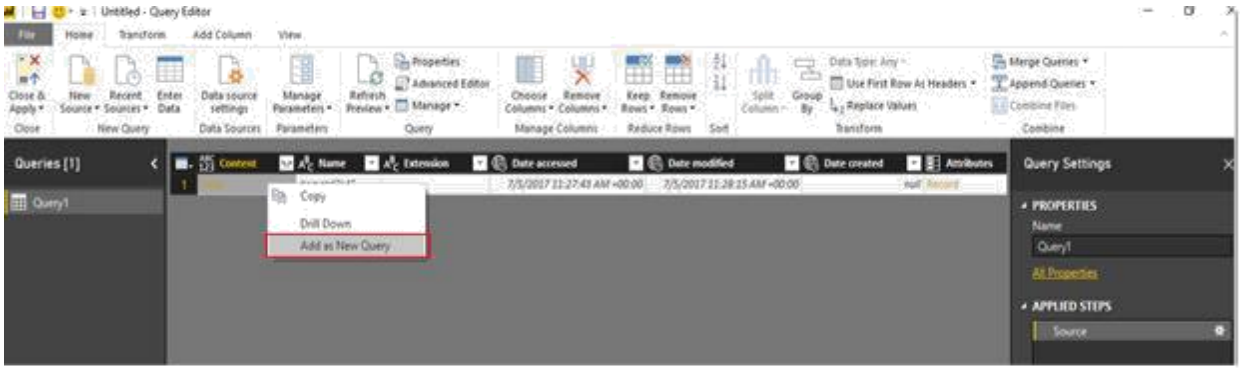
6. After the data has been successfully loaded into Power BI, you will see the available fields in the **Fields** tab.



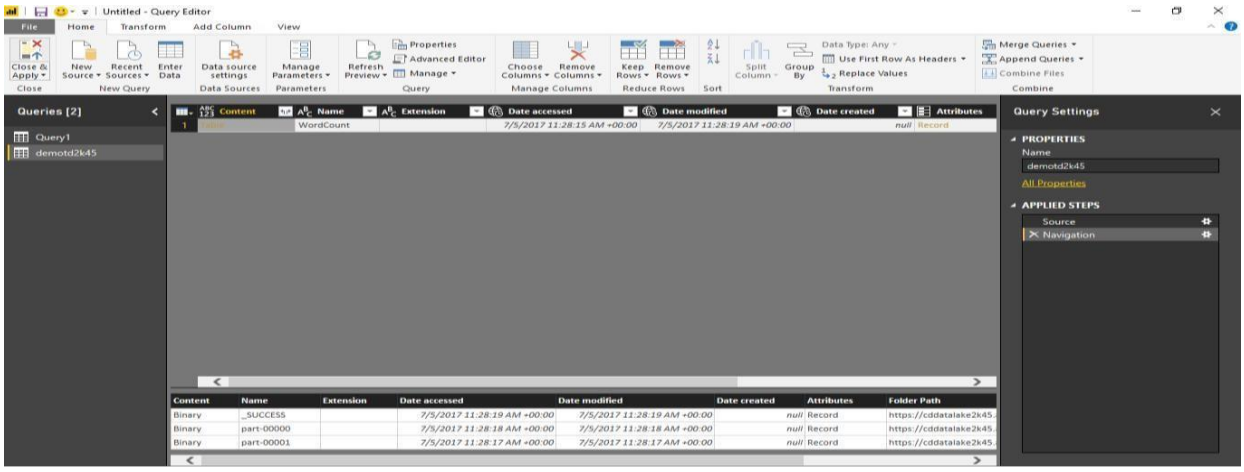
7. However, to visualize and analyze the data, you might prefer the data be available as per your requirements. To do so, follow the steps below:

8. Select **Edit Query** from the top menu bar:

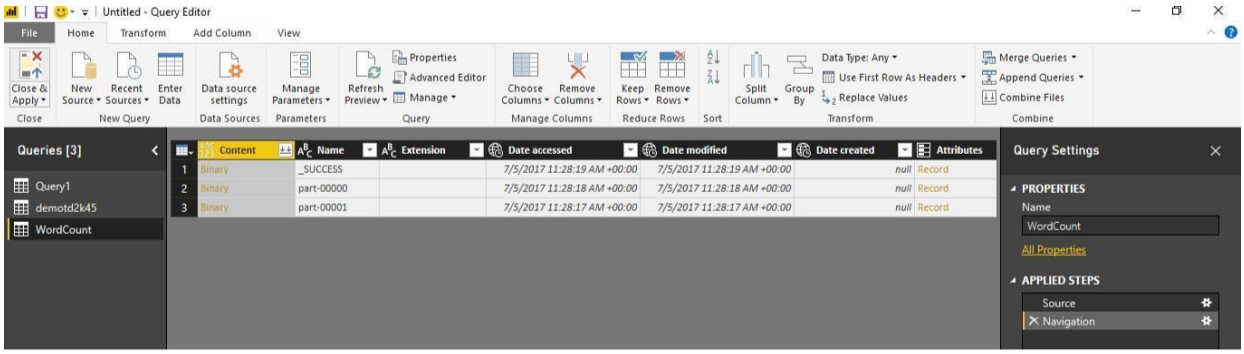
Under the content column, right click on **Table** and select **Add as New Query**, you will see a new query added in the queries column:



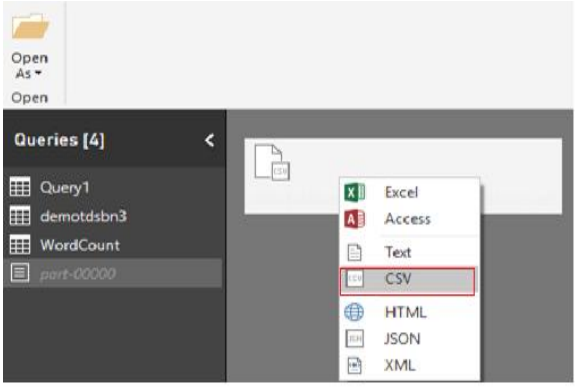
9. Once again, **right click** and select **Add as New Query** to convert the table content to binary form.

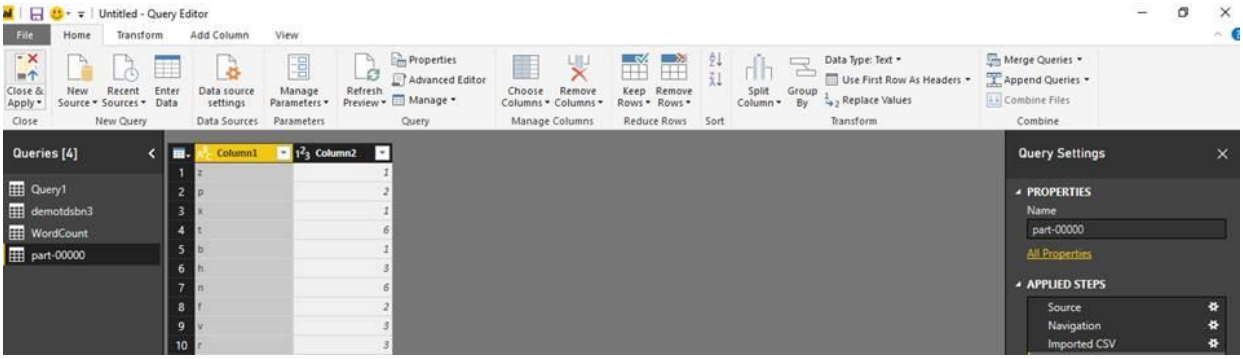


10. **Right click** and **create a new query** to get the data from the table as shown below:



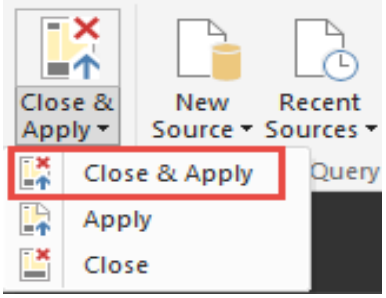
11. You will see a file icon that represents the file that you uploaded. **Right-click** the file, and click **CSV**.



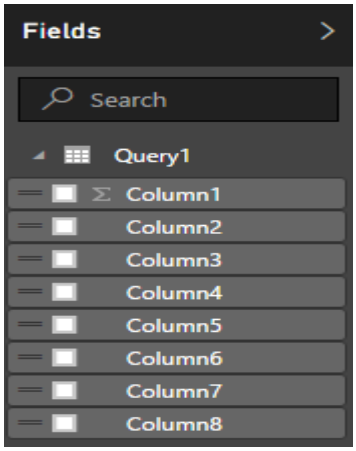


12. Your data is now available in a format that you can use to create visualizations.

13. From the **Home** ribbon, click **Close and Apply**, and then click **Close and Apply**.

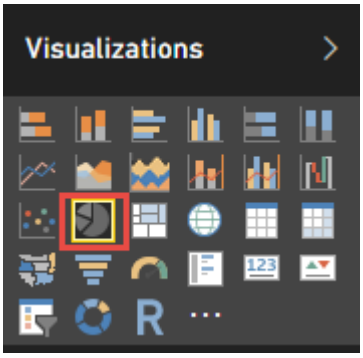


14. Once the query is updated, the **Fields** tab will show the new fields available for visualization.

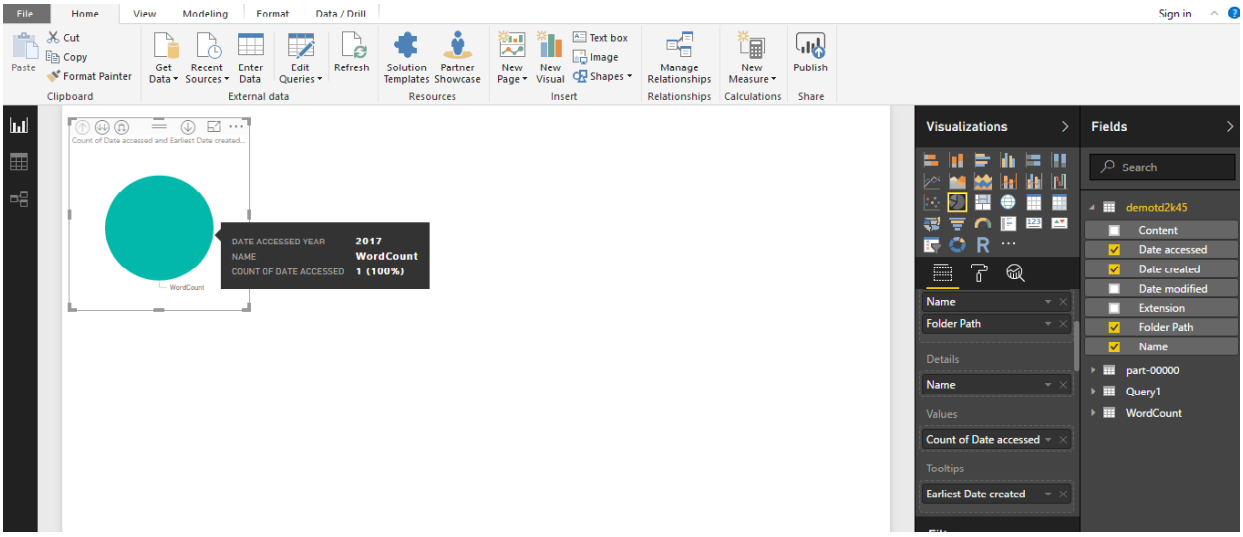


15. You can create a pie chart to represent your data. To do so, make the following selections:

a) From the **Visualizations** tab, click the symbol for a **pie chart** (see below).



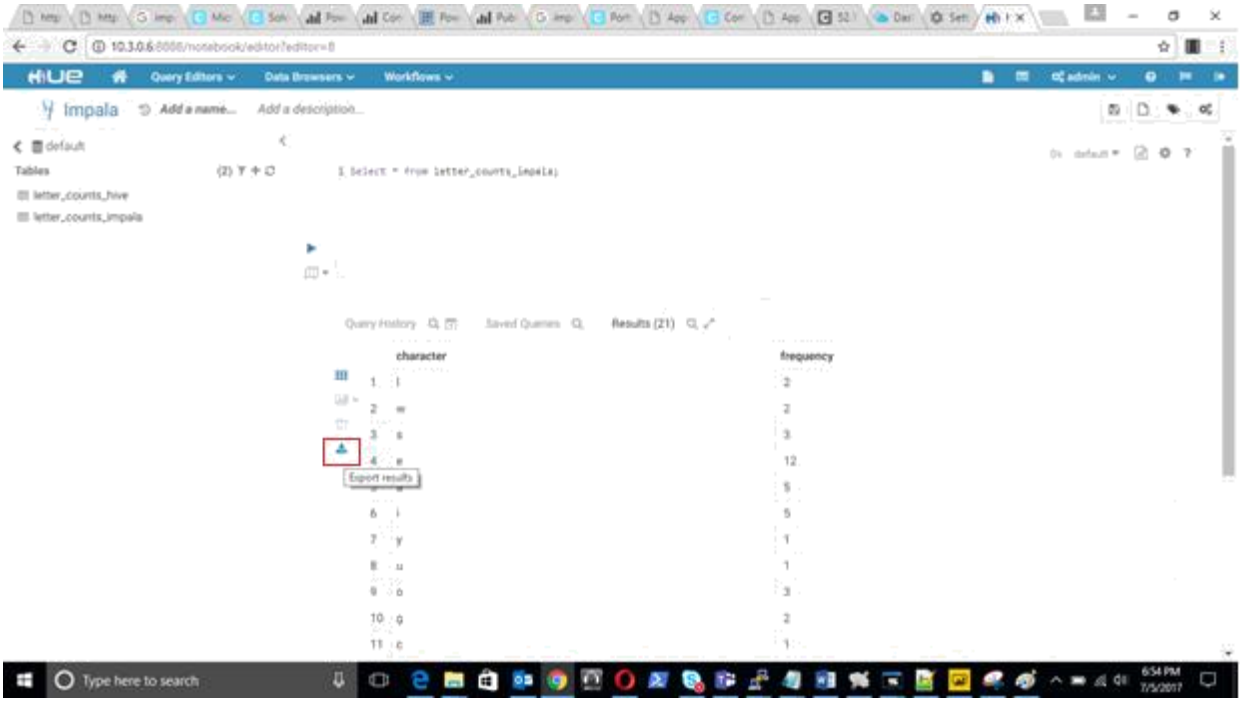
b) Drag the columns that you want to use and represent in your pie-chart from the **Fields** tab to **Visualizations** tab, as shown below:



16. From the **file** menu, click **Save** to save the visualization as a Power BI Desktop file.

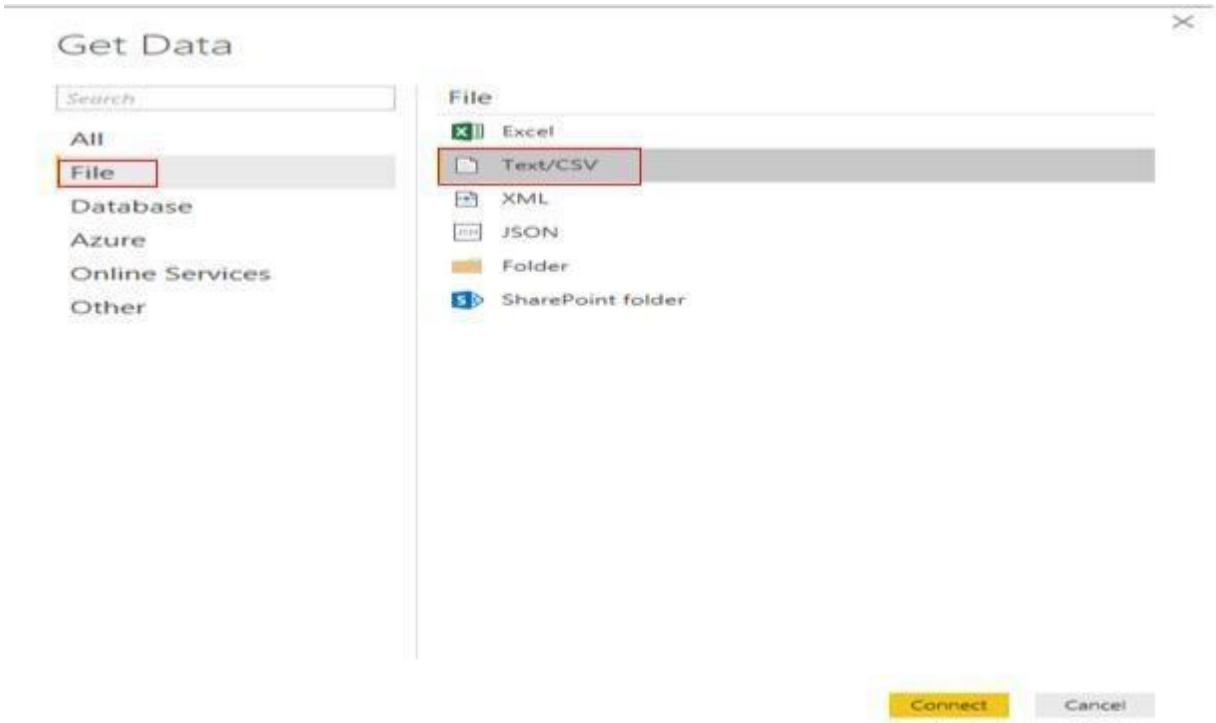
4.2 Integrating with Impala

1. Go to **point 7** of section **4.7**, where you ran a query from the table created using the output from ADLS copied to local HDFS.

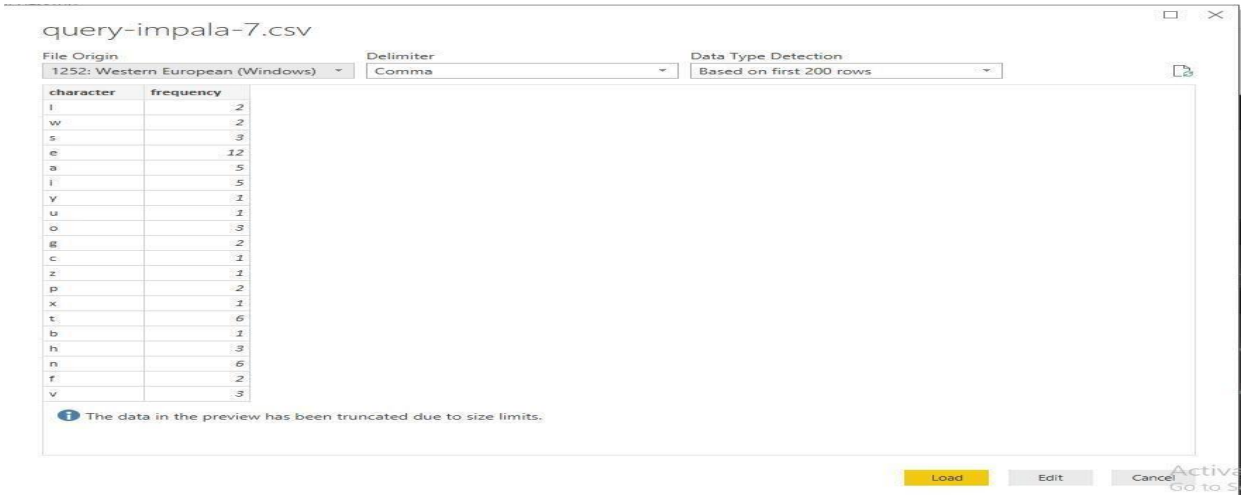


2. Click the **Export Results** button in the Hue Impala UI, as seen in the above screenshot, to download the output as a **CSV** file.

3. From the **Home** ribbon in Power BI, click **Get Data**, and then click **More**. In the **Get Data** dialog box, click **File,** click **Text/CSV,** and then click **Connect.**



4. Select the **CSV** file exported from Impala in **Step 2** and click on **Open**.



5. Click on **Load.**

6. Select the **Data** button to visualize the content.

character	frequency
l	2
w	2
s	3
e	12
a	5
i	5
y	1
u	1
o	3
g	2
c	1
z	1
p	2
x	1
t	6
b	1
h	3
n	6
f	2
v	3
r	3

You have successfully visualized the content exported from impala using power BI.

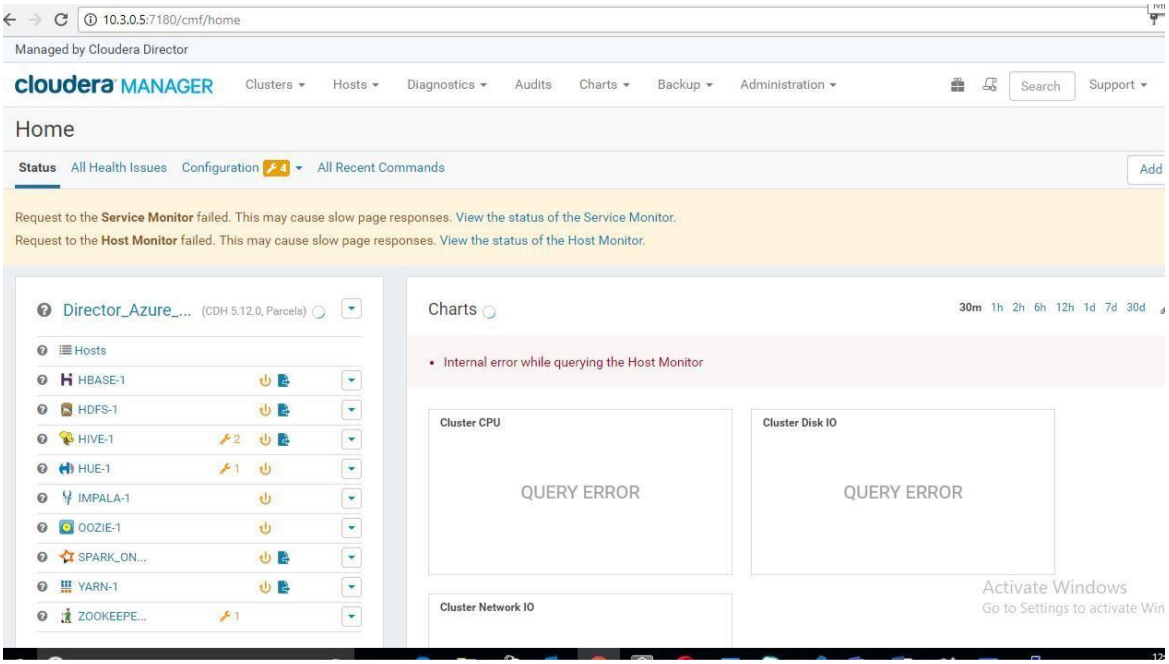
5. Reference

5.1 Restart Cloudera Management Service

You may need to restart Cloudera Management Service for the below errors:

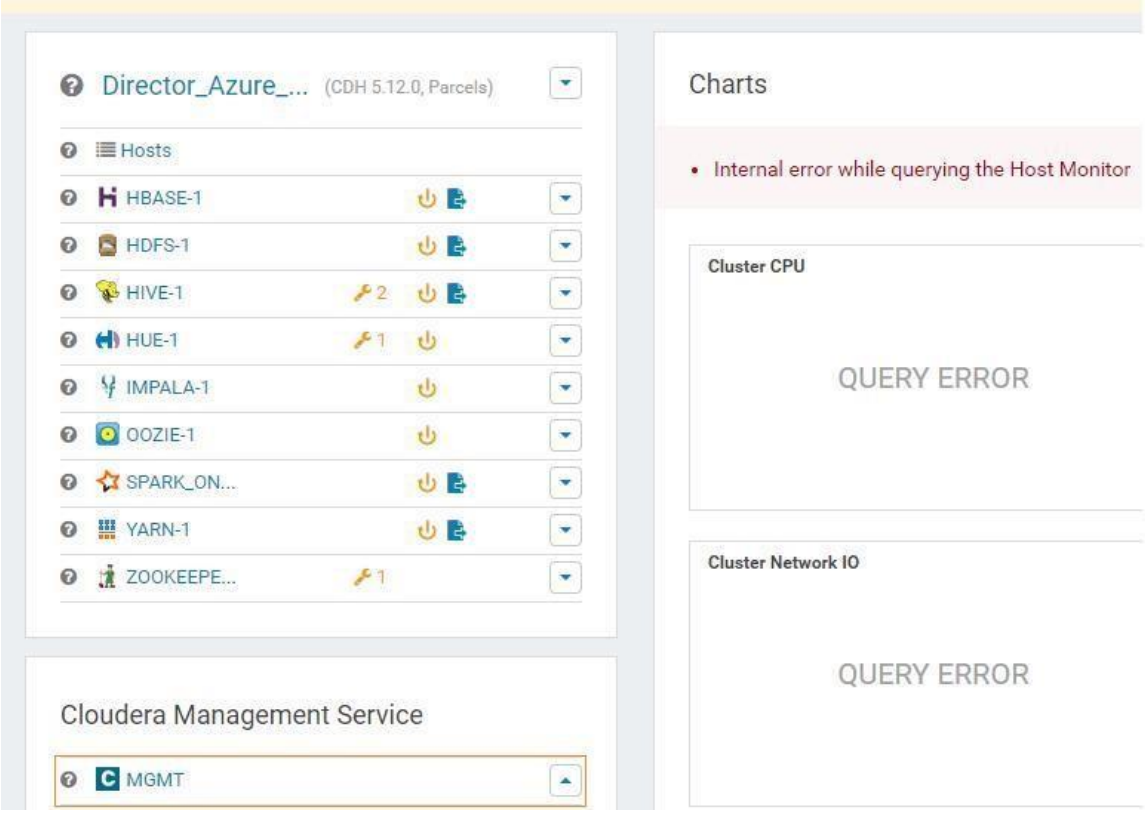
Error:

- Request to the Service Monitor failed. This may cause slow page responses. View the status of the Service Monitor.
- Request to the Host Monitor failed. This may cause slow page responses. View the status of the Host Monitor.

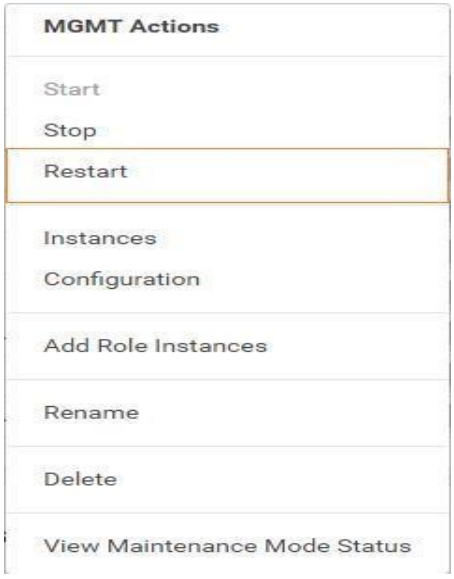


1. Go to http://<***Manager Node FQDN***>:7180/cmf/home.

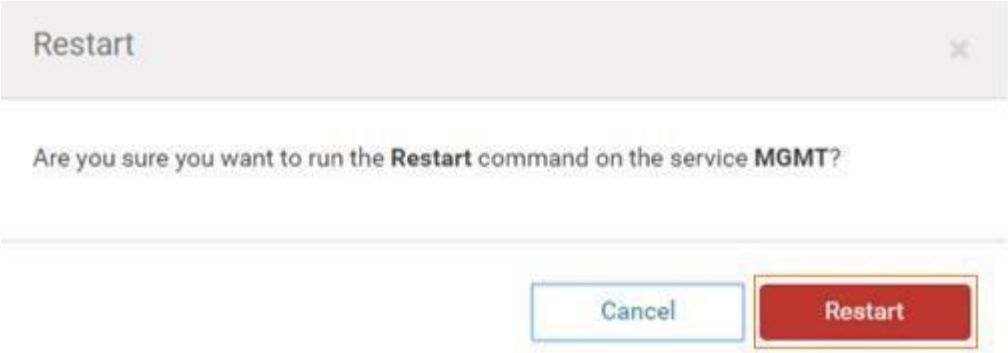
2. Go to **Cloudera Management Service** and select **MGMT.**



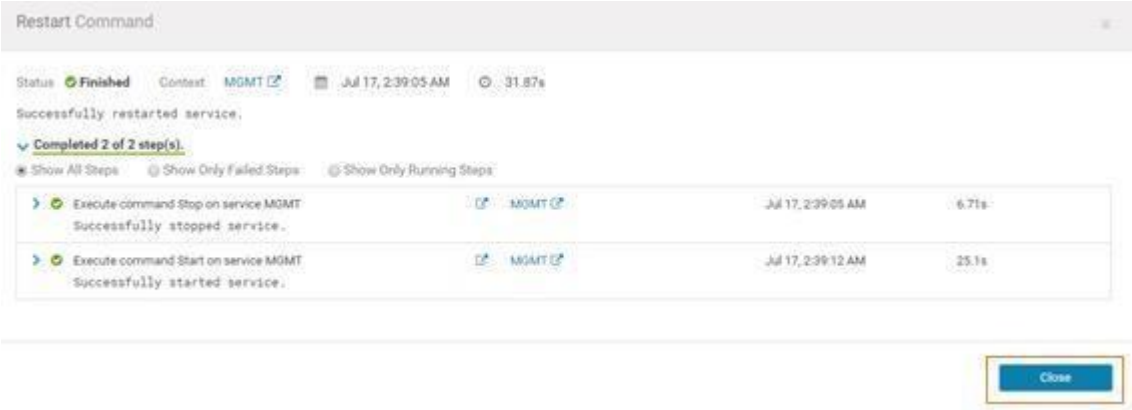
3. Click on the drop down menu and select **Restart.**



4. Confirm by clicking the **Restart** button.



5. Click on **Close** to complete the process.



<u>Note:</u> If you performed this restart in response to errors, please now re-run section **4.3** after performing the above steps.

5.2 Error Messages While Running the Spark Job

1. You may see a few errors popping up while executing the Spark job that can safely be ignored, such as the ones below.

<u>Note:</u> The permissions get properly set in the .sh file.

```
sh ClouderaSparkSetup.sh demotdweti 10.3.0.6 mkdir: Permission denied: user=cloudera,
access=WRITE, inode="/":hdfs:supergroup:drwxr-xr-x --
2017-07-11 16:55:54-- https://aztdrepo.blob.core.windows.net/clouderadirector/wordcount.jar
Resolving aztdrepo.blob.core.windows.net... 52.238.56.168
Connecting to aztdrepo.blob.core.windows.net|52.238.56.168|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6371588 (6.1M) [application/octet-stream]
Saving to: "/home/cloudera/wordcount.jar"
```

2. Searching for Cloudera Navigator – this error can safely be ignored.

```
INFO scheduler.DAGScheduler: Job 1 finished: saveAsTextFile at SparkWordCount.scala:32, took
1.811055 s
INFO spark.SparkContext: Invoking stop() from shutdown hook
ERROR scheduler.LiveListenerBus: Listener ClouderaNavigatorListener threw an exception
java.io.FileNotFoundException: Lineage is enabled but lineage directory
/var/log/spark/lineage doesn't exist
at

com.cloudera.spark.lineage.ClouderaNavigatorListener.checkLineageEnabled(ClouderaNavigatorLis
tener.scala:122) at com.cloudera.spark.lineage.
```

Note: You may refer to the **Spark** section of the **Cloudera release notes** for further details (link below).

https://www.cloudera.com/documentation/enterprise/releasenotes/topics/cn_rn_known_issues.html