

Hive Lab

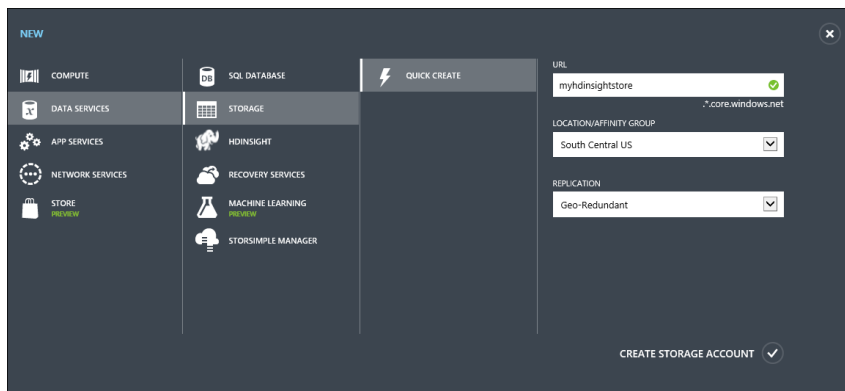
## Contents

Lab 1: Provisioning an HDInsight Hadoop Cluster .....	3
Exercise 1: Create an Azure Storage Account .....	15
Exercise 2: Create an HDInsight Hadoop cluster. ....	16
Lab 2: Hive & Query Console .....	17
Exercise 1: Query Console.....	17
Exercise 2: Exploratory Queries .....	20
Exercise 3: Structured Queries.....	22
Lab 3: Hive with Azure Machine Learning Studio .....	29
Exercise 1: Hive Queries using Azure ML .....	29
Exercise 2: Writing and Creating a Hive Table Using Azure ML .....	35
Lab 4: Loading Hive Tables from Azure Blob.....	44
Exercise 1: Uploading Data to a Blob Storage Using Azure Storage Explorer .....	44
Exercise 2: Create Hive Table using Query Console.....	51
Lab 5: Joining Hive Tables .....	54
Exercise 1: Loading two tables to be joined. ....	54
Exercise 2: HiveQL Join Using Query Console. ....	55
Exercise 3: Join with Aggregation .....	57
Exercise 4: Join in Azure Machine Learning Studio .....	58
Extra Lab: Hive with Powershell.....	59
Exercise 1: Intro to PowerShell .....	59
Exercise 2: Uploading Files to the Blob Storage.....	61
Exercise 3: Importing Datasets as Hive Tables.....	63
Appendix & Alternative Methods .....	66

# Lab 1: Provisioning an HDInsight Hadoop Cluster

## Exercise 1: Create an Azure Storage Account

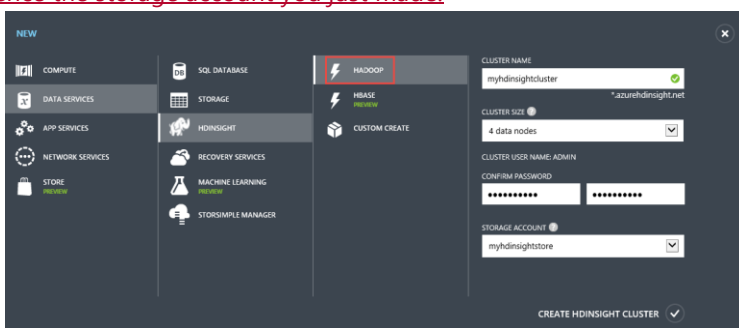
1. Sign into the Azure portal.
  - a. If not sign up for a free trial  
<http://azure.microsoft.com/en-us/pricing/free-trial/>
2. Create a data storage for the Hadoop cluster.
  - a. Click **New>Data Services>Storage>Quick Create**
  - b. The URL will be the name of your storage and serve as a pseudo primary key for your storage name. Assign a unique name to it. It's called a URL because the storage account can be referenced directly via HTTP, ex:  
<https://mystorageaccount.blob.core.windows.net/>
  - c. Select a region that is closest to you, your users, or your Hadoop clusters.



## Exercise 2: Create an HDInsight Hadoop cluster.

Once your storage account has been created (~2 minute process), create an HDInsight Hadoop cluster.

1. Starting from the bottom-left corner of your Azure portal (manage.windowsazure.com), click **New>Data Services>HDInsight>Hadoop**
2. Assign a cluster name.
3. Select anywhere between 1-4 nodes (1-2 recommended).
  - a. More nodes equates to more computers for parallel processing, however, it will result in higher usage fees.
4. Create a password for your Hadoop cluster.
5. Reference the storage account you just made.



- d. The cluster will take ~15 minutes to be up and running.

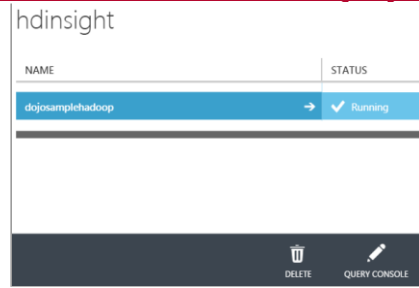
## Lab 2: Hive & Query Console

### Exercise 1: Query Console

1. Select **HDINSIGHT** from the left-hand menu on the Azure Portal.

a. Select the Hadoop cluster from the previous lab.

b. From the bottom menu, click **Query Console**.



c. You will be prompted to log in.

i. Input "admin" for the username, and then input the password you chose for the Hadoop cluster.

d. Select Hive editor at the top.

2. Run the default query by clicking submit.

a. The query will be added to "job session" at the bottom of the screen.

i. This is a queue of jobs.

b. Each query will take a minimum of ~2 minutes do to overhead.

c. HiveQL statements are not case sensitive. So "**select \* from table**", "**SELECT \* FROM table**", and "**SeLeCt \* fRoM table**" will all yield the same result.

d. Note the jobID of the query. The job session queue will only display recent jobs. You can view all jobs in "Job History" at the top.

Job Session					Submit
Query Name	Date	Id	Action	Status	
<a href="#">View Details</a>	1/9/2015, 8:27:49 AM	job_1420819640797_0001		Completed	







e. It's possible to fire off multiple jobs at the same time and wait for them to finish.

Click "view details" on the jobs that are running; it will populate once the job is

finished.

Job Session

Submit

Query Name	Date	Id	Action	Status
<a href="#">View Details</a>	1/9/2015, 2:53:49 PM	job_1420819640797_0018	 	Running
<a href="#">View Details</a>	1/9/2015, 2:53:27 PM	job_1420819640797_0017	 	Running
<a href="#">View Details</a>	1/9/2015, 2:50:29 PM	job_1420819640797_0015		Completed
<a href="#">View Details</a>	1/9/2015, 2:48:36 PM	job_1420819640797_0014		Failed

f. Name queries by filling in **Query Name** above the console. Doing this will replace the "view details" title with the selected query name. Just as before, you can click on the query name to see the details and the output of the query.



Query Name

Select from HiveSampleTable

```
1 Select * from hivesampletable
```

Job Session

Submit

Query Name	Date	Id	Action	Status
Select from HiveSampleTable	1/10/2015, 11:21:52 PM	job_1420945180561_0020	 	Running

## Exercise 2: Exploratory Queries

Hive is almost like SQL. To run multiple queries in the same block, you must end each individual query with a semicolon (;).

### 1. Which Hive tables are present within this Hadoop cluster?

#### Job Query      Job Output

```
show tables;
```

```
hivesampletable
```

### 2. Let's get information about hivesampletable.

#### Job Query

```
describe hivesampletable;
```

#### Job Output

clientid	string
querytime	string
market	string
deviceplatform	string
devicemake	string
devicemodel	string
state	string
country	string
querydwelltime	double
sessionid	bigint
sessionpagevieworder	bigint

#### a. For a full list of class types visit

[http://docs.hortonworks.com/HDPDocuments/HDP1/HDP-1.2.2/ds\\_Hive/language\\_manual/datatypes.html](http://docs.hortonworks.com/HDPDocuments/HDP1/HDP-1.2.2/ds_Hive/language_manual/datatypes.html)

- This type list will be important for adding more data and creating tables
- Horton Works is a business that focuses on the development and support of Hadoop.

3. Let's get a preview of our data. Use the "limit" clause at the end of the HiveQL statement. The star tells Hive to grab each rows and limit returns X number of rows at random.

```
1 Select *
2 from hivesampletable
3 limit 100
```

## Job Output

8	18:54:20	en-US	Android Samsung SCH-i500	California	United States	13.9204007	0	0
23	19:19:44	en-US	Android HTC Incredible	Pennsylvania	United States	NULL	0	0
23	19:19:46	en-US	Android HTC Incredible	Pennsylvania	United States	1.4757422	0	1
23	19:19:47	en-US	Android HTC Incredible	Pennsylvania	United States	0.245968	0	2
28	01:37:50	en-US	Android Motorola Droid X Colorado	United States	20.3095339	1	1	
28	00:53:31	en-US	Android Motorola Droid X Colorado	United States	16.2981668	0	0	
28	00:53:50	en-US	Android Motorola Droid X Colorado	United States	1.7715228	0	1	
28	16:44:21	en-US	Android Motorola Droid X Utah	United States	11.6755987	2	1	
28	16:43:41	en-US	Android Motorola Droid X Utah	United States	36.9446897	2	0	

4. Include "set hive.cli.print.header=true;" to see headers.
- Hive does not print headers by default. Hadoop is designed for unstructured data, which does not require headers or prescribed schemas.
  - Notice the ";" at the end. That means it's a separate statement altogether. This is a Hive command rather than a HiveQL statement.

```
1 set hive.cli.print.header=true;
2 Select *
3 from hivesampletable
4 limit 100
```

		hivesampletable.clientid	hivesampletable.querytime	hivesampletable.market	hivesampletable.deviceplatform	hivesampletable.devicemake		
		hivesampletable.devicemodel	hivesampletable.state	hivesampletable.country	hivesampletable.querydweltime	hivesampletable.sessionid		
8	18:54:20	en-US	Android Samsung SCH-i500	California	United States	13.9204007	0	0
23	19:19:44	en-US	Android HTC Incredible	Pennsylvania	United States	NULL	0	0
23	19:19:46	en-US	Android HTC Incredible	Pennsylvania	United States	1.4757422	0	1
23	19:19:47	en-US	Android HTC Incredible	Pennsylvania	United States	0.245968	0	2
28	01:37:50	en-US	Android Motorola Droid X Colorado	United States	20.3095339	1	1	

5. Run the following query to find the number of rows



a. Count(\*) is the command to count all rows.

Job Query	Job Output
<pre>select count(*) from hivesampletable</pre>	59793

### Exercise 3: Structured Queries

1. Run the following query print the first 20 entries where the device maker starts with HTC.

**Microsoft Azure HDInsight**  
Hive Editor   Job History   File Browser

Use this editor to author and submit a hive query to your cluster.

Query Name

```
1 SELECT * FROM hivesampletable
2 WHERE devicemake LIKE "HTC%"
3 LIMIT 20;
```

Job Session

Submit

Title	Date	Id	Action	Status
<a href="#">View Details</a>	8/12/2014 3:32:07 PM	job_1407880989859_0001	✕	Running

2. Run the following query to see the first 20 entries where client dwell time exceeded 20 seconds. These people are probably the more interested customers. Let's isolate them.

### Job Query

```
set hive.cli.print.header=true;
select *
from hivesampletable
where querydwelltime > 20;
```

### Job Output

hivesampletable.clientid	hivesampletable.querytime	hivesampletable.market	hivesampletable.deviceplatform	hivesampletable.devicecode	hivesampletable.state	hivesampletable.country	hivesampletable.querydwelltime	hivesampletable.sessionpagevieworder	
28	01:37:50	en-US	Android Motorola	Droid X Colorado	United States	20.3095339	1	0	
28	16:43:41	en-US	Android Motorola	Droid X Utah	United States	36.9446892	2	0	
28	01:37:19	en-US	Android Motorola	Droid X Colorado	United States	28.9811416	1	0	
30	17:19:36	en-US	RIM OS RIM	9650	Massachusetts	United States	3468.538966	0	2
30	17:17:18	en-US	RIM OS RIM	9650	Massachusetts	United States	66.8533378	0	1
45	21:09:43	en-US	Android Samsung	SCH-I500	Illinois	United States	857.1453275	1	0
62	03:07:36	en-US	Android LG	V5910	California	United States	39.4991038	0	0
77	14:48:39	en-US	Android LG	V5660	Illinois	United States	41.2376733	0	0
89	22:54:19	en-US	Android Samsung	SCH-I500	Texas	United States	77.033459	0	0
89	22:55:38	en-US	Android Samsung	SCH-I500	Texas	United States	46.2687482	0	2
93	00:15:36	en-US	Android LG	V5910	California	United States	20.5633449	0	4
109	18:55:10	en-US	RIM OS RIM	9330	Massachusetts	United States	728.9452825	0	0
186	18:09:48	en-US	Android LG	V5660	Colorado	United States	20.9030322	3	0
212	17:45:32	en-US	Android Samsung	SCH-I500	New York	United States	23.6670157	2	0
343	17:37:54	en-US	Android Samsung	SCH-I500	New York	United States	1003.473103	3	0

3. What was the average dwell time? Run the following query to find out.

```
1 select "average dwell time", avg(querydwelltime)
2 from hivesampletable
3 ;
```

### Job Output

```
average dwell time      26821.62253550747
```

- avg(querydwelltime) returns the average from the entire column
- The average dwell time is equivalent to 4.5 hours. Perhaps there is a reason for such a high number. Let's explore further.

4. Run the following query to find the min, max, variance, and sum of the dwell time.

```
1 select
2     "max", max(querydwelltime),
3     "min", min(querydwelltime),
4     "variance", variance(querydwelltime),
5     "sum", sum(querydwelltime)
6 from hivesampletable
7 ;|
```

### Job Output

max	1.311337648E9	min	0.0	variance	3.5094722745573395E13	sum	1.31420586099479
SE9							

- a. The variance and maximum values are incredibly large numbers. It looks like an outlier is throwing off our summary statistics. Let's grab the top 10 dwell times to check.

5. Run the query below to print the 10 largest dwell time in descending order.

### Job Output

```
1 select clientid, querydwelltime
2 from hivesampletable
3 order by querydwelltime desc
4 limit 10
5 ;
```

93611	1.311337648E9
79817	74095.20764
39995	72913.46986
11870	66059.51017
15774	65864.25682
55113	63043.17924
30867	57675.22212
128986	52097.84944
53432	49528.82018
95273	45412.09978

- a. "order by" ensures total population ordering, which can take a very long time with a big dataset. This is why Hive also supports "sort by", which is far less time consuming. It sorts post query and will be demonstrated in a later exercise.

b. The highest value appears to be an incredibly large outlier. It equates to 1000 years and is almost 1800 times the next value, which is converts to around 22 hours.

6. What is the distribution of our devices? Run the following query to find the total devices for each brand.

### Job Output

```
1 set hive.cli.print.header=true;
2 Select devicemake, count(*)
3 from hivesampletable
4 group by devicemake;
```

devicemake	_c1
ASUS	45
Apple	22731
Archos	1
Casio	996
DELL	54
FujitsuToshibaMobileCommun	2
HTC	2971
Huawei	230

7. The next query renames the queried columns using the "as" clause.

```
1 set hive.cli.print.header=true;
2 Select devicemake as device_make, count(*) as device_total
3 from hivesampletable
4 group by devicemake;
```

a. HiveQL syntax does not use quotes around the new names. This is a subtle difference from SQL.

b. Also note that we're still explicitly stating that we want headers in our queries as a separate statement, even though we are predefining our own metadata.

## Job Output

device_make	device_total
ASUS	45
Apple	22731
Archos	1
Casio	996
DELL	54
FujitsuToshibaMobileCommun	2
HTC	2971
Huawei	230
Kyocera	117
LG	8116

8. Let's get the frequency counts by devices where dwell times where higher than 20 second. This will filter out the devices that aren't generating enough engagement.

a. Apply a "where" clause before the group by.

b. Examine how the totals dropped.

## Job Output

```
1 set hive.cli.print.header=true;
2 select devicemake as Device, count(*) as Total
3 from hivesampletable
4 where querydwelltime > 20
5 group by devicemake
6 ;
```

device	total
ASUS	11
Apple	4517
Casio	141
DELL	5
HTC	511
Huawei	66
Kyocera	58
LG	1432
Microsoft	3
Motorola	297
RIM	985
SAMSUNG	84
Samsung	2429
SonyEricsson	22
Unknown	367

9. To focus on the top brands, let's include those with 100 dwell times greater than 20 seconds. Use the "HAVING" clause since the filter occurs after the grouping.

## Job Output

```
1 set hive.cli.print.header=true;
2 select devicemake as Device, count(*) as Total
3 from hivesampletable
4 where querydwelltime > 20
5 group by devicemake
6 HAVING Total > 100
7 ;
```

device	total
Apple	4517
Casio	141
HTC	511
LG	1432
Motorola	297
RIM	985
Samsung	2429
Unknown	367

10. To finish this lab, let's print the previous query in descending order by using the "sort by" clause. Limit the query to the 7 most engaging brands.

a. The "sort by" clause will ensure that the final group-by list will be sorted instead of the entire dataset, unlike "order by".

```
1 set hive.cli.print.header=true;
2 select
3     devicemake as Device,
4     count(*) as Total
5 from hivesampletable
6 where querydwelltime > 20
7 group by devicemake
8 HAVING Total > 100
9 sort by Total desc
10 limit 7
11 ;
```

## Job Output

device	total
Apple	4517
Samsung	2429
LG	1432
RIM	985
HTC	511
Unknown	367
Motorola	297

## Lab 1: Provisioning an HDInsight Hadoop Cluster

Usually reading and uploading hive tables ~~is a tedious process that~~ requires table creation, metadata creation, data upload, and data parsing onto the HDFS. Azure ML completely automates this process and is currently the only tool to do so.

~~Reading hive tables or loading data into a hive table is usually tricky to do. However with Azure ML, reading and writing to a hive table is easy because Azure will automate the table creation, metadata creation, data upload, and data parsing onto the HDFS. Currently, Azure ML is the best tool for uploading hive tables because it is currently the only tool that is fully managed and automated.~~

~~This lab requires a Hadoop cluster.~~

~~This lab will assume that you already have a Hadoop cluster setup.~~

3.2. \_\_\_\_\_ Populate the required fields ~~of the Reader module~~ using the values below. ~~Open up a second window or tab to retrieve the other information.~~

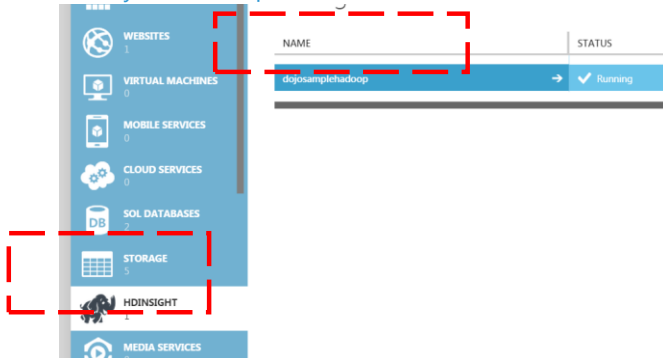
- a. **Data Source:** HiveQuery
- b. **Hive database query:** ~~Use the following query. Below is a sample query statement that you may perform. However, any of the query statements you made in the Hive lab will also work.~~

```
select country, state, count(*) as
records from hivesampletable
group by country, state
order by records
desc limit 5
```

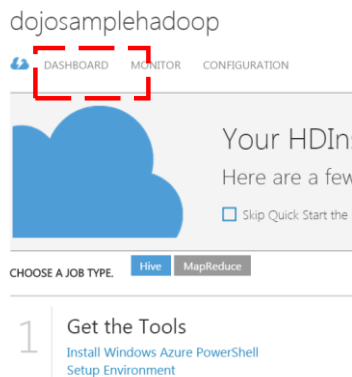
```
1 select country, state, count(*) as records
2 from hivesampletable
3 group by country, state
4 order by records
5 desc limit 5
```

- c. **HCatalog server URI:**
  - i. ~~From your azure management portal (open~~Open this [link](https://manage.windowsazure.com/) in a new window or tab): <https://manage.windowsazure.com/>

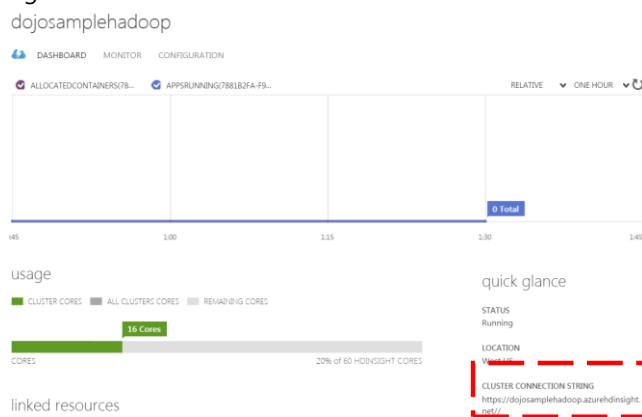
- ii. Click on HDInsight and then on your Hadoop Cluster to enter your Hadoop cluster dashboard.



- iii. Click on "Dashboard."



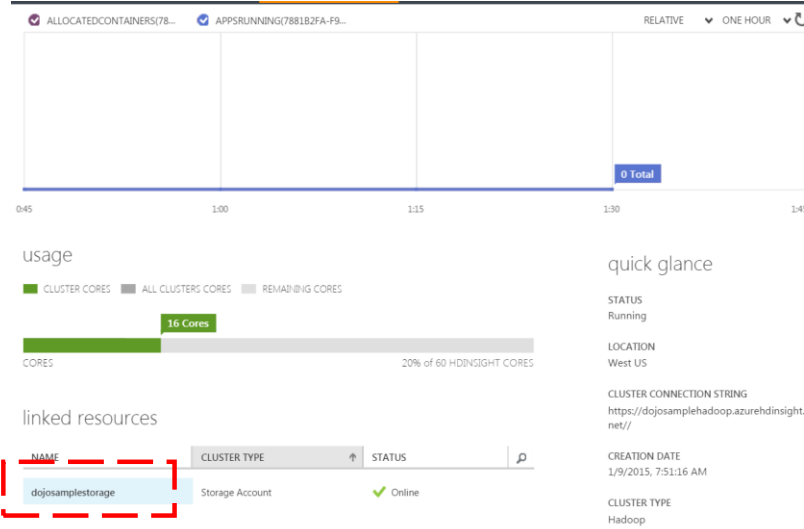
- iv. Retrieve Copy your the cluster connection string located on the bottom right-hand corner of the dashboard screen.



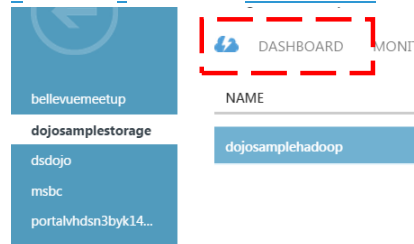


In the above example it is  
<https://dojosamplehadoopcluster.azurehdinsight.net/>

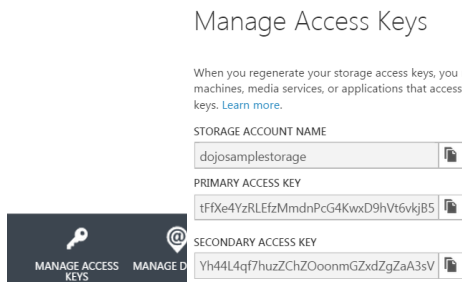
- v. ~~Copy the cluster connection string into your~~Return to -Azure ML ~~and paste the cluster connection string into the reader-Reader module under's~~  
**server URI**~~section.~~
- d. **Hadoop user account name:** admin
- e. **Hadoop user account password:** The password you used ~~when you provisioned creating your for the~~ HDInsight Hadoop cluster.
- f. **Location of output data:** Azure
- g. **Azure storage account name:** ~~To find this detail,~~  
g.i. ~~simply~~ Scroll down on the Hadoop cluster dashboard page. ~~(the same called "linked resources".~~



- h.ii. Click on the cluster ~~and it'll take you to your storage page name. Then~~  
~~Click on "dashboard"~~Dashboard".



iii. Once inside, click on "manage access keys" located in the bottom middle of the page the bottom tool bar. Then, you can click the copy button to the right of the read-only input boxes.



iv. Copy-Paste the storage name into your Azure ML reader the Reader module.

#### h. Azure storage key:

i. You can copy either of the access keys given in the "Manage Access Keys" popup window.

1. Ideally, the primary access key is the key used for important web services that depend on this storage account, such as a Hadoop cluster.
2. The secondary key is the one given out for others, such as clients, to use and can be regenerated often.

#### Manage Access Keys

When you regenerate your storage access keys, you need to update machines, media services, or applications that access this keys. [Learn more.](#)

STORAGE ACCOUNT NAME	dojosamplestorage	
PRIMARY ACCESS KEY	tFXe4YzRLEfzMmdnPcG4KwxD9hVt6vkjBS	
SECONDARY ACCESS KEY	Yh44L4qf7huzZChZOoonmGZxdZgZaA3sV	

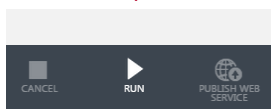
ii. Paste the storage key into the Reader module.

#### i. Azure container name:

i. Click on "containers" and copy the text under name.

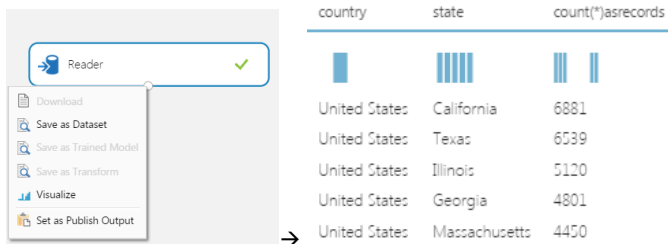
1. It should be the same ~~Your container name should actually be the same~~

dojosamplestorage



6.a. Occasionally on Azure, Hadoop clusters are corrupted on creation. This will result in an "Error 0068" for any valid Hive query. If this happens, create a new Hadoop cluster and start the lab over.

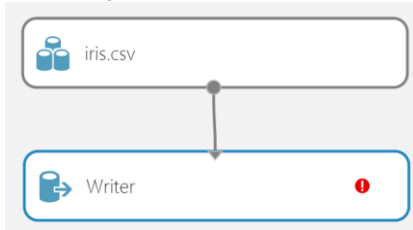
Preview the data by visualizing the output of the reader module. ~~Right click the bottom output from the Hive query.~~



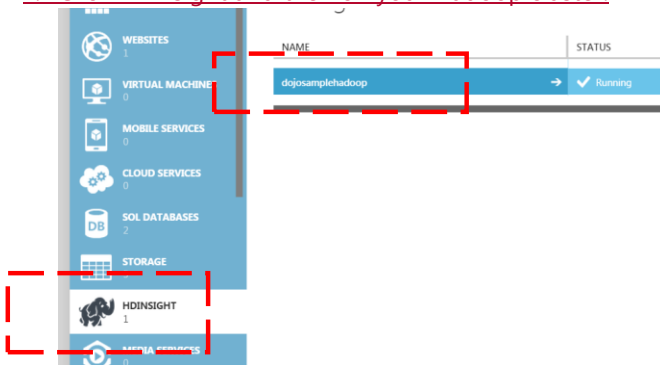
a. If you want to save your the query to Azure ML output, simply clicking select

## Exercise 2: Writing and Creating a Hive Table Using Azure ML

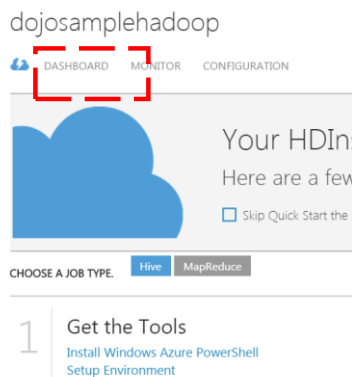
1. Drag in any dataset that you may want to be written to your Hadoop cluster as a Hive Table.
2. Then drag in a writer module and connect the dataset to writer module.



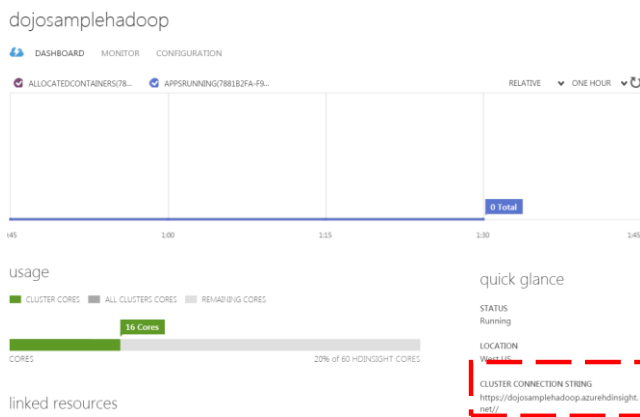
3. Set-Populate the required fields of the Writer module using the values below. the credentials of the writer module:
  - a. Set "Please specify data destination: Set" to "Hive\_Query".
  - b. Hive Table Name: yourTableNameiris
    - This will be the name of your table when uploaded into of the table containing the uploaded data set-hive and will be used in queries, i.e. ,  
FROM yourTableName.iris.
  - c. HCatalog server URI:
    - i. Open this link in a new window or tab: <https://manage.windowsazure.com/>
    - ii. Click HDInsight and then on your Hadoop cluster.



iii. Click on "Dashboard".



iv. Copy the cluster connection string located on the bottom-right corner of the dashboard screen.



v. Return to Azure ML and paste the cluster connection string into the Reader module under **HCatalog server URI**.

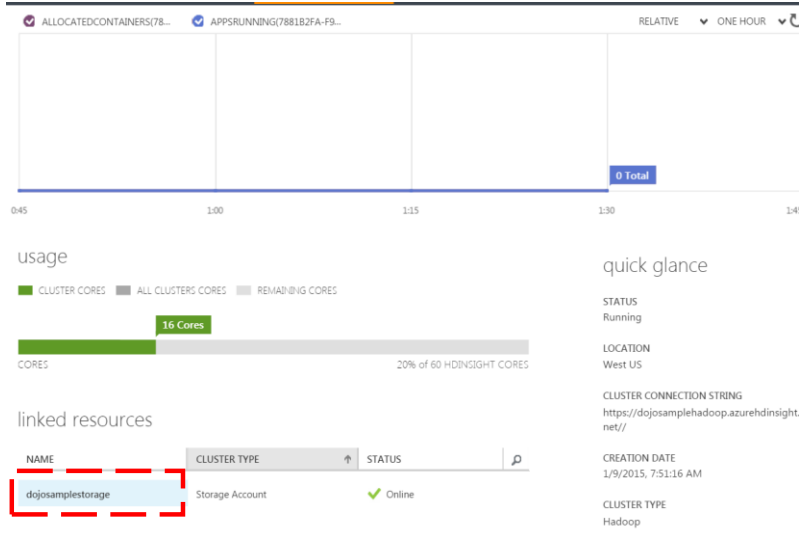
d. **Hadoop user account name:** admin

e. **Hadoop user account password:** The password you used for the HDInsight Hadoop cluster.

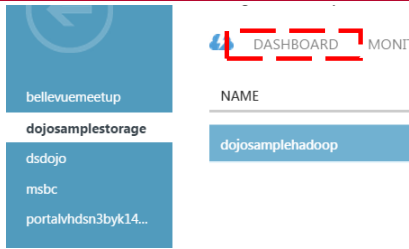
f. **Location of output data:** Azure

g. **Azure storage account name:**

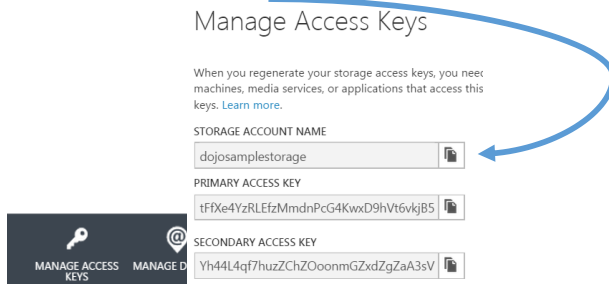
i. Scroll down on the Hadoop cluster dashboard page. It will be under a table called "linked resources".



ii. Click on the cluster name. Then click "Dashboard".



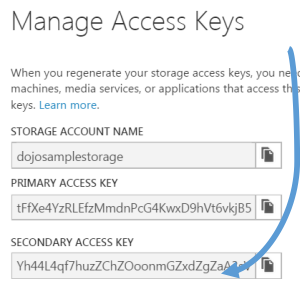
- iii. Click on “manage access keys” located in the bottom tool bar. Then, click the copy button to the right of the input box.



- iv. Paste the storage name into the Writer module.

**h. Azure storage key:**

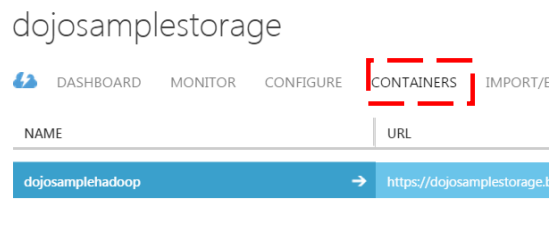
- i. Copy either of the access key given in the “Manage Access Keys” popup.
1. The primary access key is the key used for important web services
  2. The secondary key is for others, such as clients, and can be regenerated often.



- ii. Paste the storage key into the Writer module

**i. Azure Container:**

- i. Click on “containers” and copy the text under name.
1. It should be the same name as your Hadoop cluster.



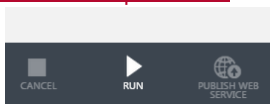
**—HCatalog server URI:**

4. This is a sample of ~~what it should look like when you're done, the completed~~  
Writer module.

The screenshot shows a workflow in the Data Science Canvas. At the top, a box labeled 'iris.csv' with a file icon is connected by an arrow to a 'Writer' module below it. The 'Writer' module has a green checkmark, indicating it is configured correctly. To the right of the canvas is a configuration panel for the 'Writer' module. The panel contains the following fields:

- Please specify data destin...: Hive Query (dropdown menu)
- Hive table name: iris (text input)
- HCatalog server URI: https://dojohadoop.azure (text input)
- Hadoop user account ...: admin (text input)
- Hadoop user account ...: (password input field with dots)
- Location of output data: Azure (dropdown menu)
- Azure storage account...: dojoattendeestorage (text input)
- Azure storage key: (password input field with dots)
- Azure container name: dojohadoop (text input)

4. Run the experiment.~~Run the experiment to execute the import and parse.~~



4.



## Lab 4: Loading Hive Tables from Azure Blob

Hive tables are not permanently persistent entities since Hadoop is meant to be temporary. As a result each Hadoop cluster must be referenced to a Blob Storage for data persistence. That way when the cluster is killed, the data will live on in the Blob Storage. As a result the referenced Blob storage is treated like a local directory for the Hadoop cluster. This lab will explore the Blob/HDInsight relationship by reading a dataset from the storage.

### Exercise 1: Uploading Data to a Blob Storage Using Azure Storage Explorer

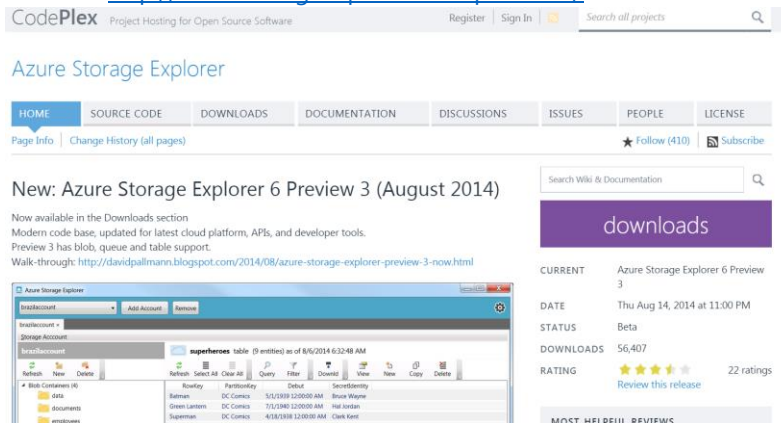
In order to read data from the blob storage, it first must be uploaded to the blob. To do this we will utilize an application called Azure Storage Explorer.

#### 1. Get the data

- ~~We will be working with the Iris dataset. Specifically the 3 class dataset.~~ Visit the following link. ~~Download and view the data to get the three-class iris data set.~~ <https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data>
- Save the dataset to your desktop as a CSV file.

#### 2. ~~Obtaining and~~ Installing Azure Storage Explorer

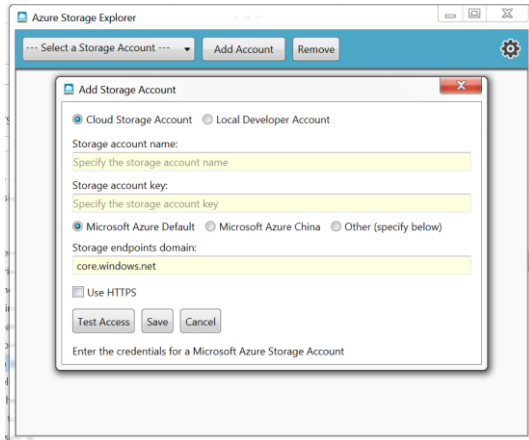
- ~~Download~~ Azure Storage Explorer ~~is a free tool that can be downloaded from~~ ~~CodePlex: Azure Storage Explorer.~~ Full Link: <http://azurestorageexplorer.codeplex.com/>



~~—Unzip and i~~ Install Azure Storage Explorer

- Log onto your ~~referenced~~ blob storage.
  - Launch Azure Storage Explorer

b. Click on "add account"

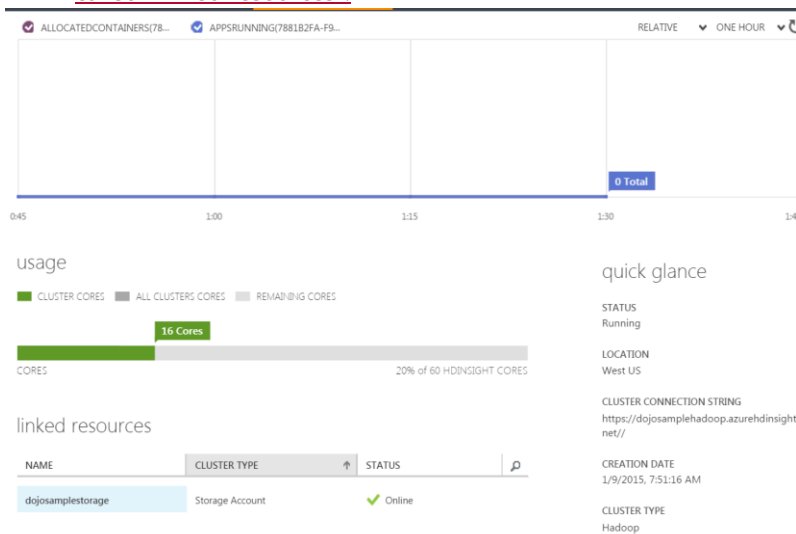


4. Fill in the credentials:

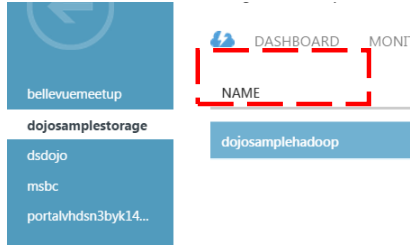
a. Select **Cloud Storage Account**

b. **Azure storage account name:**

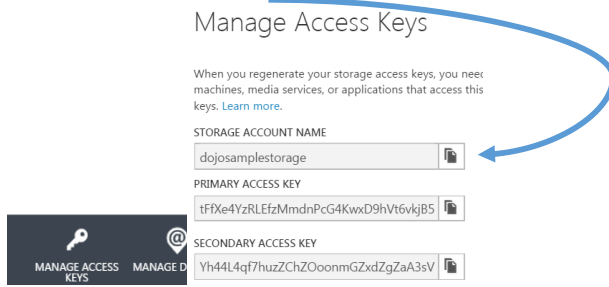
i. Scroll down on the Hadoop cluster dashboard page. It will be under a table called "linked resources".



ii. Click on the cluster name. Then click "Dashboard".



iii. Click on "manage access keys" located in the bottom tool bar. Then, click the copy button to the right of the input box.



iv. Paste the storage name into Azure Storage Explorer.

### c. **Azure storage key:**

i. Copy either of the access key given in the "Manage Access Keys" popup.

1. The primary access key is the key used for important web services

2. The secondary key is for others, such as clients, and can be regenerated often.

### Manage Access Keys

When you regenerate your storage access keys, you need to update machines, media services, or applications that access the keys. [Learn more.](#)

STORAGE ACCOUNT NAME  
dojosamplestorage

PRIMARY ACCESS KEY  
tFXe4YzRLEfzMmdnPcG4KwxD9hVt6vkjBS

SECONDARY ACCESS KEY  
Yh44L4qf7huzZChZOoonmGZxdZgZaA3sV

ii. Paste the Azure storage key into Azure Storage Explorer.

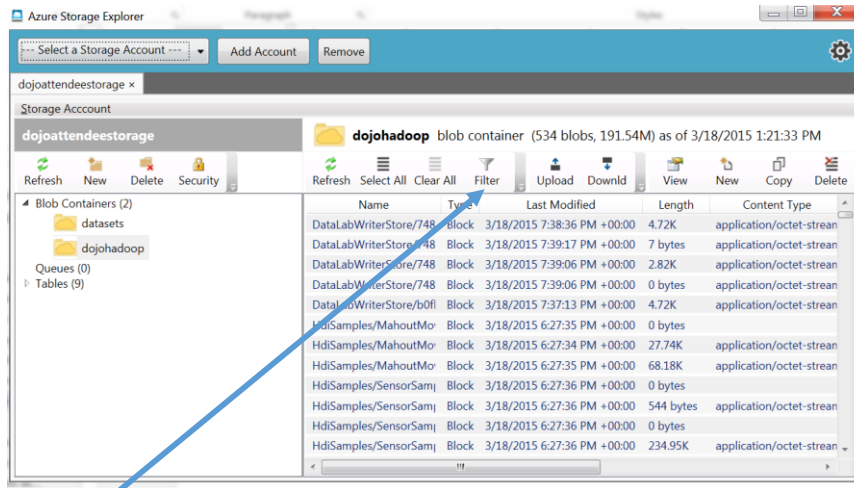
### ~~Storage Account Name:~~

e.e. Save the connection by hitting "save".

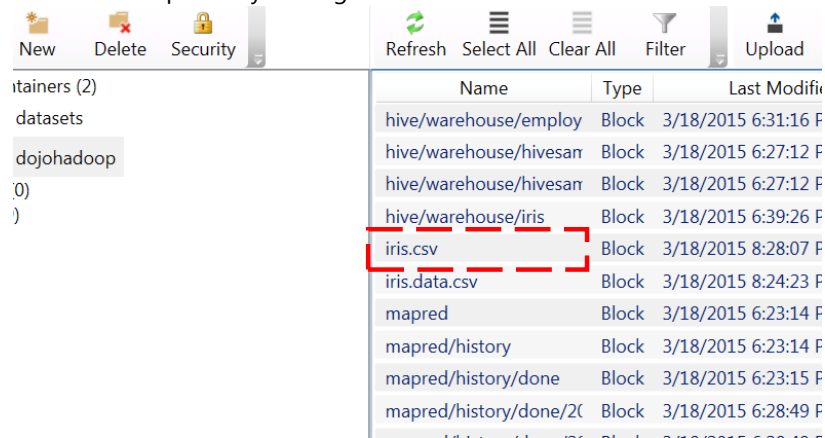
p.i. A tab ~~should containing open uthep and log you into the~~ blob storage

#### 4.5. Uploading File

- a. Click on the blob container that the Hadoop cluster is referencing.
  - i. ~~In most cases the name of the container is the same name as your~~
  - ii. Notice how it is completely populated with files related to the Hadoop cluster.



- b. Hit the upload button and upload your iris csv file.
- c. Confirm the upload by finding the file.



## Exercise 2: Create A Hive Table using Query Console

Before ~~we can load~~loading in the data ~~into Hive that we have just uploaded to the~~from the blob, we must create a new blank hive table with the schemas defined.

1. Create a new table with a defined schema.
  - a. Log ~~back~~into your Hadoop cluster's query console. ~~It~~then go to your Hive Editor.
  - b. We will now create a new table which our data will be uploaded into. This table will be blank, but the schema and storage will be provisioned. This will give our data structure when we populate it into Hadoop as a Hive table.
  - c. Preview the Iris dataset from your local file.

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
```
  - i. From the preview, we want the schema to accept FLOAT, FLOAT, FLOAT, FLOAT, STRING.

- d. To find out what each column represents, visit:  
<https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.names>
  - i. Go to #7 in the file, under “attribute information”
  - ii. This is what we will name our columns.
- e. Syntax: **create table myTableName(mySchema)**. Copy the code below to create the table for the iris dataset ~~to be loaded into~~.

```
CREATE TABLE IF NOT EXISTS iris (
  SepalLengthCM float,
  SepalWidthCM float,
  PetalLengthCM float,
  PetalWidthCM float,
  IrisType String
)
COMMENT 'Iris Dataset from UCI'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

```
#creates hive table for iris data.
CREATE TABLE IF NOT EXISTS iris (
  SepalLengthCM float,
  SepalWidthCM float,
  PetalLengthCM float,
  PetalWidthCM float,
  IrisType String
)
COMMENT 'Iris Dataset from UCI'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

2. Execute “show tables” again to confirm if the table was created. Although the table shows up, it currently has no fields. We can now load data into this table.
3. Load data into the table from your Azure Blob storage.

Query Name

populate iris

```
1 load data inpath '/iris.csv/'
2 into table iris;
```

- a. **LOAD DATA INPATH**, notice how the file path is simply /iris.csv. We did not have to specify which storage or which container because we already uploaded

the file to the default location that the Hadoop cluster reads from.

## Job Query

```
select * from iris;
```

## Job Output

5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
5.0	3.6	1.4	0.2
5.4	3.9	1.7	0.4
4.6	3.4	1.4	0.3
5.0	3.4	1.5	0.2

2. Double Check
- a. Scroll to the bottom of the output. Notice how the last row is “null null null null”. It means that the original CSV we loaded in had an extra line that was empty at the end. Sadly there is actually no way to delete this row in HiveQL. You should not think about Hive as a regular RDBMS. Hive is better suited for batch processing over very large sets of immutable data. There is no operation supported for the deletion or update of a particular record ~~or particular set of records.~~

6.5	3.0	5.2	2.0	Iris-virginica
6.2	3.4	5.4	2.3	Iris-virginica
5.9	3.0	5.1	1.8	Iris-virginica
NULL	NULL	NULL	NULL	NULL

## Lab 5: Joining Hive Tables

~~Like normal SQL, Hive~~ hive lets you join hive tables. Though keep in mind that Hive tables are designed for data warehouses, where datasets are often de-normalized so that joins are not necessary. Ideally your data was already captured in a way so that hive joins are not necessary.

### Exercise 1: Loading two tables to be joined.

We will be uploading two example datasets as Hive tables onto HDFS using Azure ML.

1. Navigate to a new Azure ML experiment.
2. For this lab, drag in the following two datasets: Restaurant ratings and Restaurant customer data.
3. Drag in a writer module and populate the Hive credentials into the writer module. Full directions on how to write datasets to Hadoop as a Hive table can be referenced from **Lab 3, Exercise 2**.
4. After the writer module's credentials have been populated, copy and paste the writer module. Be sure to set "Hive Table Name" differently for their respective datasets. Hive Table Name should also be a single string without spaces or special characters or else the tables will not be written to Hive HDFS. Spaces can be denoted with an underscore "table\_name".

The screenshot displays the Azure ML interface. On the left, two dataset boxes labeled "Restaurant ratings" and "Restaurant customer data" are shown. Arrows point from these datasets to two "Writer" modules, each marked with a green checkmark. On the right, the "Writer" configuration panel is open, showing the following fields:

- Please specify data destination: Hive Query
- Hive table name: restaurant\_customers
- HCatalog server URI: https://dojocluster.azurehdinsight.net
- Hadoop user account name: admin
- Hadoop user account password: [REDACTED]
- Location of output data: Azure
- Azure storage account name: dojocattendestorage
- Azure storage key: [REDACTED]
- Azure container name: dojocluster

At the bottom of the panel, it shows "START TIME 3/18/2015 6:33:30 PM".



Exercise 2: HiveQL Join Using Query Console.

- 1. Navigate to the Hadoop cluster’s query console.
- 2. Confirm the existence of the new tables by performing a “show tables;” query.

Job Query

Job Output

show tables;

hivesampletable  
restaurant\_customers  
restaurant\_ratings

- 3. Preview both tables with the following statement:

```
set hive.cli.print.header=true;  
  
select * from restaurant_customers limit 3;  
  
select * from restaurant_ratings limit 3;
```

```
1 set hive.cli.print.header=true;  
2 select * from restaurant_customers limit 3;  
3 select * from restaurant_ratings limit 3;
```

- a. Notice how we are running 3 separate queries in a single hive job. Our results will be concatenated together, so look carefully when you-viewing the output.

Job Output

restaurant_customers.userid	restaurant_customers.latitude	restaurant_customers.longitude	restaurant_customers.smoker	restaurant_customer
s.drink_level	restaurant_customers.dress_preference	restaurant_customers.ambience	restaurant_customers.transport	restaurant_customers.marita
l_status	restaurant_customers.hijos	restaurant_customers.birth_year	restaurant_customers.interest	restaurant_customers.personality
restaurant_customers.religion	restaurant_customers.activity	restaurant_customers.color	restaurant_customers.weight	restaurant_customer
s.budget	restaurant_customers.height			
"UI001"	22.139997	-100.978803	false	"abstemious"
"variety"	"thrifty-protector"	"none"	"student"	"black" 69
"UI002"	22.150087	-100.983325	false	"abstemious"
"technology"	"hunter-ostentatious"	"Catholic"	"student"	"red" 40
"UI003"	22.119847	-100.946527	false	"social drinker"
t"	1989	"none"	"hard-worker"	"Catholic"
restaurant_ratings.userid	restaurant_ratings.placeid	restaurant_ratings.rating		
"UI077"	135085	2		
"UI077"	135038	2		
"UI077"	132825	2		

- b. Notice that both tables have a userID. This is what we will join onwith.

4. ~~Let's both tables together on userid.~~ Input the following code ~~to join the tables on userid below~~

```
1 set hive.cli.print.header=true;
2 select *
3 from restaurant_customers as customer
4 join restaurant_ratings as rating
5 on customer.userid = rating.userid
6 limit 100;
7
```

#### Job Output

customer.userid	customer.latitude	customer.longitude	customer.smoker	customer.drink_level	customer.dress_preference	customer.amb				
ience	customer.transport	customer.marital_status	customer.hijos	customer.birth_year	customer.interest	customer.personality	cust			
omer.religion	customer.activity	customer_color	customer.weight	customer.budget	customer.height	rating.userid	rating.placeid	rating.ratin		
"U1001"	22.139997	-100.978803	false	"abstemious"	"informal"	"family"	"on foot"	"single"	"Independent"	1989
"variety"	"thrifty-protector"	"none"	"student"	"black" 69	"medium"	1.77	"U1001"	132830	1	
"U1001"	22.139997	-100.978803	false	"abstemious"	"informal"	"family"	"on foot"	"single"	"Independent"	1989
"variety"	"thrifty-protector"	"none"	"student"	"black" 69	"medium"	1.77	"U1001"	132825	2	
"U1001"	22.139997	-100.978803	false	"abstemious"	"informal"	"family"	"on foot"	"single"	"Independent"	1989
"variety"	"thrifty-protector"	"none"	"student"	"black" 69	"medium"	1.77	"U1001"	135085	0	
"U1001"	22.139997	-100.978803	false	"abstemious"	"informal"	"family"	"on foot"	"single"	"Independent"	1989
"variety"	"thrifty-protector"	"none"	"student"	"black" 69	"medium"	1.77	"U1001"	135040	1	
"U1001"	22.139997	-100.978803	false	"abstemious"	"informal"	"family"	"on foot"	"single"	"Independent"	1989
"variety"	"thrifty-protector"	"none"	"student"	"black" 69	"medium"	1.77	"U1001"	135039	1	
"U1001"	22.139997	-100.978803	false	"abstemious"	"informal"	"family"	"on foot"	"single"	"Independent"	1989
"variety"	"thrifty-protector"	"none"	"student"	"black" 69	"medium"	1.77	"U1001"	135045	1	
"U1001"	22.139997	-100.978803	false	"abstemious"	"informal"	"family"	"on foot"	"single"	"Independent"	1989

- a. Notice that we ~~proxy the used the "AS" clause to create simple aliases for the tables-table name~~ restaurant\_customers as simply 'customer' and restaurant\_ratings into simply 'rating'. This makes the headers- ~~more readable~~ come out nicer. Also notice that we ~~We also~~ limited the query to only display the first 100. ~~This is a good practice to for previewing data the data, unless you really want to receive all of the data. Keep in mind hive tables tend to be on the larger side.~~

### Exercise 3: Join with Aggregation

1. Let's do a full count of all 3 tables: the customer table, the ratings table, and the combined join table.

```
1 select "customers", count(*)
2 from restaurant_customers;
3
4 select "rating", count(*)
5 from restaurant_ratings;
6
7 select "Joined", count(*)
8 from restaurant_customers as customer
9 join restaurant_ratings as rating
10     on customer.userid = rating.userid;
```

#### Job Output

customers	138
customers	1161
Joined	1161

2. Some things to note:
  - a. We are performing 3 queries in the same job.
  - b. We purposely left out the headers so the tables would ~~just~~ vertically concatenate for us.
  - c. ~~Also notice that we~~We are explicitly addeding the word "Joined" to the output of the last query to give the table symmetry.

Exercise 4: Join [Hive Tables](#) in Azure Machine Learning Studio

This dataset would be much easier to read if we were to populate it into Azure ML instead of the query console.

- 1. Create a new experiment in Azure ML.
- 2. Drag in a reader module.
- 3. For full directions on where to find the writer module's HiveQuery credentials, please refer to **Lab 3, Exercise 1**.
- 4. Insert the join query into the "hive database query" textbox.

Hive database query

```
1 select *
2 from restaurant_customers as customer
3 join restaurant_ratings as rating
4   on customer.userid = rating.userid;
```

rows  
1161

columns  
22

view as

Col1	Col2	Col3	Col4	Col5	Col6	Col7	Col8	Col9	Col10	Col11	Col12
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot	single	independent	1989	vari
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot	single	independent	1989	vari
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot	single	independent	1989	vari
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot	single	independent	1989	vari
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot	single	independent	1989	vari
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot	single	independent	1989	vari
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot	single	independent	1989	vari
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot	single	independent	1989	vari
U1001	22.139997	-100.978803	false	abstemious	informal	family	on foot	single	independent	1989	vari
U1002	22.150087	-100.983325	false	abstemious	informal	family	public	single	independent	1990	tech

## Extra Lab: Hive ~~with~~ with Azure ~~Powershell~~ PowerShell

In this lab, you will find out how to use Hive beyond the query console. This lab will assume that you have completed the HDInsight & Hive lab.

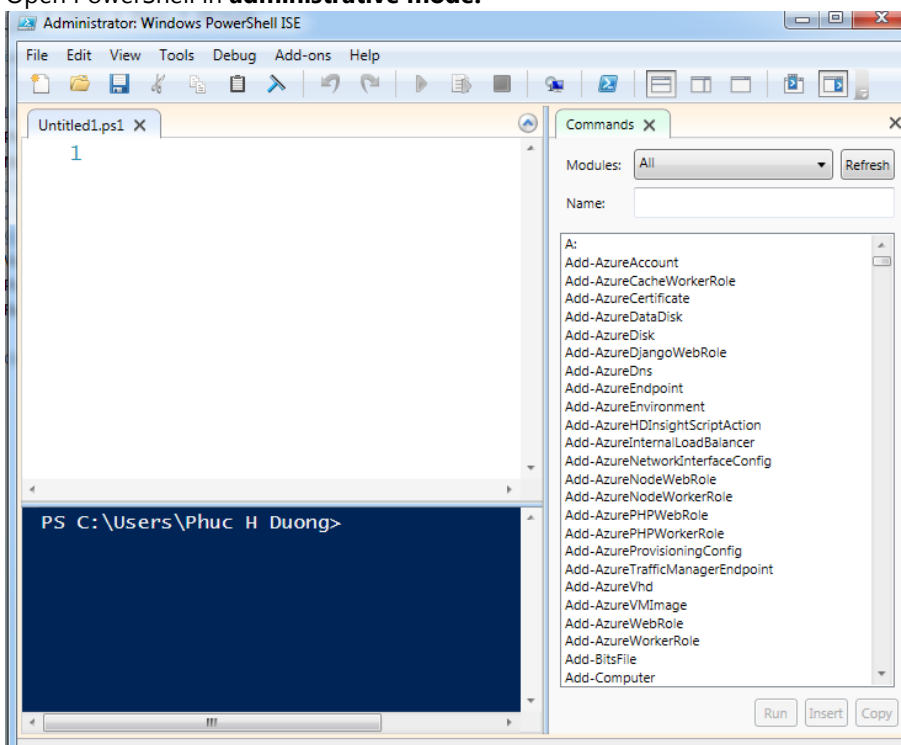
### Exercise 1: Intro to PowerShell

You can't go too long in the Azure ecosystem without encountering PowerShell with Azure SDK (systems development kit). In this exercise, we will upload a dataset to the Azure Blob Storage that our Hadoop cluster is referencing, so that our Hadoop cluster can start reading it.

1. Install PowerShell if you do not have it already. Search in your task bar to see if it exists.

We will be using **PowerShell ISE**.

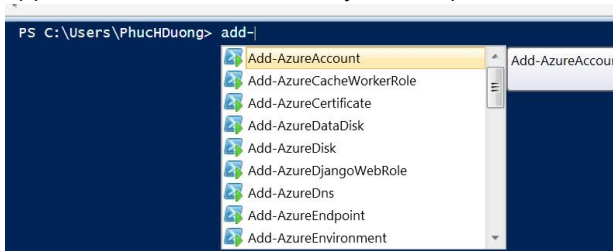
- a. Open PowerShell in **administrative mode**.



- b. Your PowerShell should look like the above picture. It has a top level for scripts and a bottom level (dark blue) for command line.

3. Checking if Azure PowerShell SDK is installed.

- a. Type in **"Add-"** into the PowerShell command line. If intellisense does not pop up, you need to install Azure PowerShell SDK under step 1-b-i. If intellisense does not appear after the install, restart your computer.



- b. Download Azure PowerShell SDK (if you don't have it installed):

<http://azure.microsoft.com/en-us/downloads/>

- c. Under Windows 'PowerShell'

4. Load your Azure Subscription

- a. Method: automatic authentication
- b. Type in **Add-AzureAccount**. It will open a web browser to log into your Azure account with a 12 hour instance. If this step errors, please refer to "Appendix & Alternative Methods" for an alternative method (last exercise of this lab).

## Exercise 2: Uploading Files to the Blob Storage

1. Get the data
  - a. We will be working with the Iris dataset. Specifically the 3 class dataset. Visit the following link. Download and view the data <https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data>
  - b. Save the dataset to your desktop as a CSV file.
2. Below is a sample script. Paste it into the script section of your PowerShell ISE window. We will need to change a few details.

```
$subscriptionName = "mySubscriptionName"
$storageAccountName = "dojosamplestorage"
$containerName = "dojosamplehadoopcluster"
$clusterName = "dojosamplehadoopcluster"

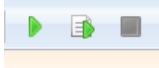
#uploads to https://dojosamplestorage.blob.core.windows.net/dojosamplecluster/iris.csv
$fileName = "C:\Users\Bruce Wayne\Desktop\iris-multi.csv"
$blobName = "iris.csv"

# Get the storage account key
Select-AzureSubscription $subscriptionName
$storageaccountkey = get-azuresoragekey $storageAccountName | %{$_.Primary}

# Create the storage context object
$destContext = New-AzureStorageContext -StorageAccountName $storageAccountName -
StorageAccountKey $storageaccountkey

# Copy the file from local workstation to the Blob container
Set-AzureStorageBlobContent -File $fileName -Container $containerName -Blob $blobName -
context $destContext
```

- a. **\$subscriptionName** to get your subscription name, type in **Get-AzureSubscription**, (assuming you've already done **Add-AzureAccount** and logged in). It will be listed under subscription name.
  - b. **\$storageAccountName** is the storage account name of the Azure Storage account that your Hadoop cluster is referencing, the same one we made in exercise 1 of this lab.
  - c. **\$clusterName** and **\$containerName** should have the same name.
  - d. **\$filename** needs to be changed to the full file path to the Iris dataset that you saved.
  - e. **\$blobname** is what you want the file to be called once it is uploaded to the Azure Storage Blob.
3. Run the script
    - a. Once the changes have been made, hit the run button:



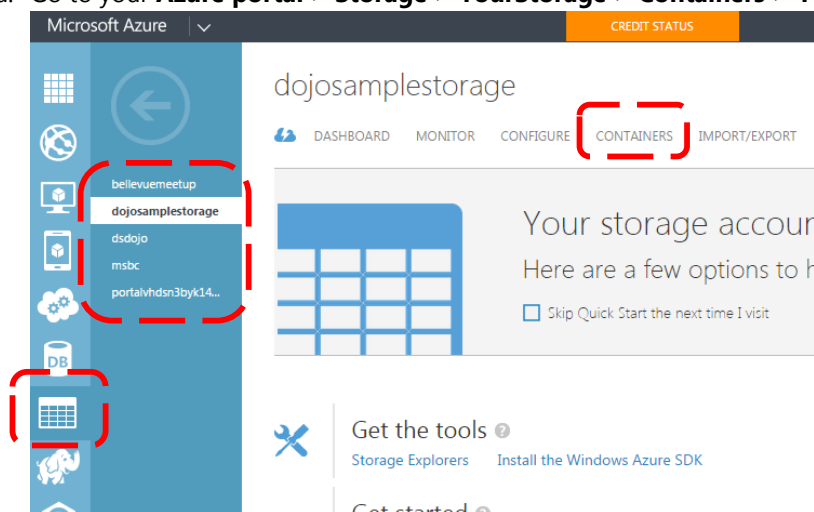
- b. If the script fails to run due to limited permissions:
  1. Input **Set-ExecutionPolicy RemoteSigned** into the command line console.
  2. State 'yes' to the pop-up.
- c. You should get a similar response in your PowerShell console after the script completes.

```
UploadDataset.ps1 X
11
12 $subscriptionName = "Microsoft"
13 $storageAccountName = "dojosamplestorage"
14 $containerName = "dojosamplehadoop"
15 $clusterName = "dojosamplehadoop"
16
17 # Uploads to https://dojosamplestorage.blob.core.windows.net/dojosamplecluster/iris.csv
18 $fileName = "c:\Users\PHUCHDUONG\Documents\Data Mining\HDI Insight Map Reduce Example\iris.data.csv"
19 $blobName = "iris.csv"
20
21 # Get the storage account key
22 $select-AzureSubscription $subscriptionName
23 $storageAccountKey = get-azuresubscriptionkey $storageAccountName | %{$_.Primary}
24
25 # Create the storage context object
26 $destContext = New-AzureStorageContext -StorageAccountName $storageAccountName -StorageAccountKey $storageAccountKey
27
28 # Copy the file from local workstation to the Blob container
29 Set-AzureStorageBlobContent -File $fileName -Container $containerName -Blob $blobName -context $destContext
```

```
Environment : AzureCloud
SupportedModes : AzureServiceManagement
DefaultAccount :
Accounts :
IsDefault : False
IsCurrent : False
CurrentStorageAccountName :

ICloudBlob : Microsoft.WindowsAzure.Storage.Blob.CloudBlockBlob
BlobType : BlockBlob
Length : 4551
ContentType : application/octet-stream
LastModified : 1/10/2015 11:26:58 AM +00:00
SnapshotTime :
ContinuationToken :
Context : Microsoft.WindowsAzure.Commands.Common.Storage.AzureStorageContext
Name : iris.csv
```

4. Confirm the file's existence on the blob.
- a. Go to your **Azure portal** > **Storage** > **YourStorage** > **Containers** > **YourContainer**



- b. Locate your file.



### Exercise 3: Importing Datasets as Hive Tables

4. Create a new table with a defined schema.
  - a. Log back into your Hadoop cluster's query console, then go to your Hive Editor.
  - b. We will now create a new table which our data will be uploaded into. This table will be blank, but the schema and storage will be provisioned. This will give our data structure when we populate it into Hadoop as a Hive table.
  - c. Preview the Iris dataset from your local file.

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
```
  - i. From the preview, we want the schema to accept DOUBLE, DOUBLE, DOUBLE, DOUBLE, STRING.

- d. To find out what each column represents, visit:  
<https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.names>
- i. Go to #7 in the file, under “attribute information”
  - ii. This is what we will name our columns.
- e. Syntax: **create table myTableName(mySchema)**

```
CREATE TABLE IF NOT EXISTS iris(  
    sepal_length_in_cm double,  
    sepal_width_in_cm double,  
    petal_length_in_cm double,  
    petal_width_in_cm double)  
ROW FORMAT DELIMITED  
FIELDS TERMINATED BY ',';  
  
show tables;
```

## Job Output

```
hivesampletable  
iris
```

- 5. Execute “show tables” again to confirm if the table was created. Although the table shows up, it currently has no fields. We can now load data into this table.
- 6. Load data into the table from your Azure Blob storage.

## Job Query

```
LOAD DATA INPATH '/iris.csv/'  
INTO TABLE iris;
```

**LOAD DATA INPATH**, notice how the file path is simply /iris.csv. We did not have to specify which storage or which container because we already uploaded the file to the default location that the Hadoop cluster reads from.

## Job Query

```
select * from iris;
```

## Job Output

5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
5.0	3.6	1.4	0.2
5.4	3.9	1.7	0.4
4.6	3.4	1.4	0.3
5.0	3.4	1.5	0.2

## Appendix & Alternative Methods

1. Manual Azure Authentication In Azure PowerShell: **(skip this step if automatic authentication works)**
2. Sometimes **Add-AzureAccount** grabs the wrong subscription ID or subscription name. This can result from complications in your Azure account such as multiple subscriptions under different party names. To remedy this, we must manually authenticate and then reference the Azure account name directly.
3. Type in the following command: **Get-AzurePublishSettingsFile**, which will initiate a file download.
4. Download the settings file. Rename it something short: "mySubscription.publishsettings"
5. Open up the file in notepad, sublime, notepad++, or any other IDE.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <PublishData>
3   <PublishProfile
4     SchemaVersion="2.0"
5     PublishMethod="AzureServiceManagementAPI">
6     <Subscription
7       ServiceManagementUrl="https://management.
8       Id="b4c1bc-3827fae9d-1fa8bf-a0d37237-b3f"
9       Name="BruceWayne"
10      ManagementCertificate="ADrntTa0qxAlEtynqN
11    <Subscription
12      ServiceManagementUrl="https://management.
13      Id="e-cc68b10ce5d644755ae-c--105cf8fa2b8"
14      Name="ClarkKent"
15      ManagementCertificate="jRXdk7o50iXQEhMoFc
16    </PublishProfile>
17  </PublishData>
```

- a. Note the name of the subscription you want to use.
  - i. "BruceWayne" is our example.
- b. Append all PowershellScripts

```
$subscriptionSettingsFile = "C:\Users\Bruce Wayne\Desktop\BizSpark.publishsettings"
Import-AzurePublishSettingsFile $subscriptionSettingsFile
Get-AzureSubscription
Select-AzureSubscription -SubscriptionName "Bruce Wayne"
```