# Hand on Lab 3 - HDI Batch Analysis and PowerBI

Overall time to complete: **30 minutes**

Prerequisites: **Azure Storage Account Creation in Hands on Lab 1**, **HDInsight Cluster Creation in Hands on Lab 2**

At the end of this lab you should have achieved:

* The creation of HIVE tables
* Data transformation using HIVE
* PowerBI visualization showing IoT device data

# 1. Overview

This lab will demonstrate batch analysis with aggregation on a data set. Performing simple sums, counts and averages of data using HIVE. The data will be displayed through a simply generated Power Dashboard on the web.

# 2. Introduction to Hive

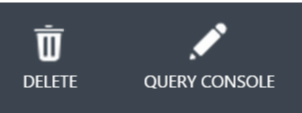
The Apache Hive data warehouse software facilitates querying and managing large datasets residing in distributed storage. Hive provides a mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL. The query is executed as a MapReduce job on a HDInsight cluster.

In the "Hands on Lab 1" we uploaded a dataset to our HDInsight container using azcopy. In this lab we'll consume this dataset. The data is device data from "Smart Home" sensors which contain everything from temperature to light intensity metrics.

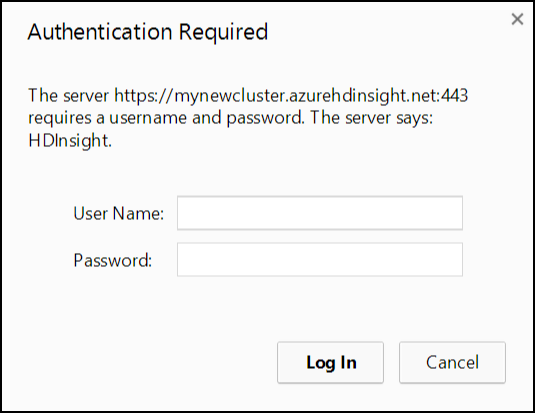
# 3. Using the Hive web interface

The HDInsight cluster provides a web based interface to execute Hive queries.

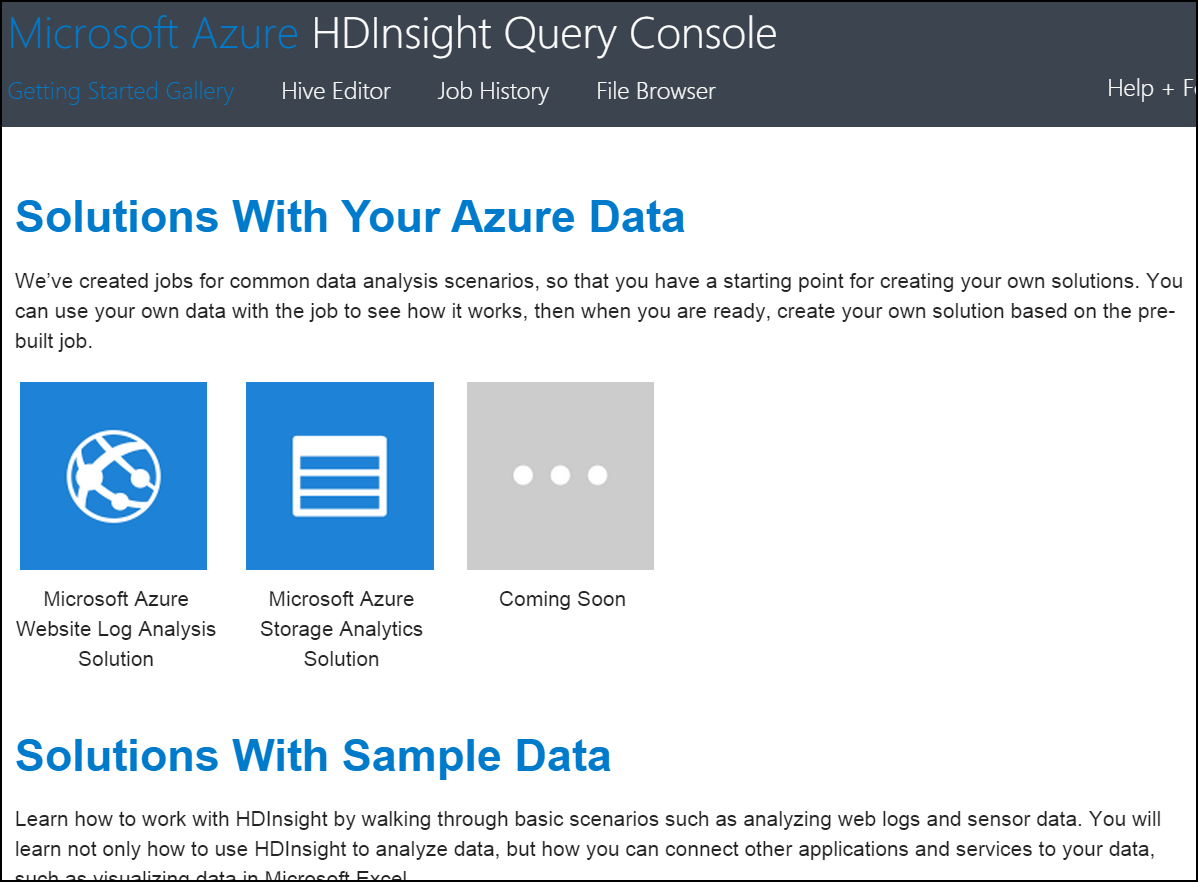
1. Return to the HDInsight cluster in the Azure Management Portal and click on the "Query Console" button at the bottom of the screen.

* 

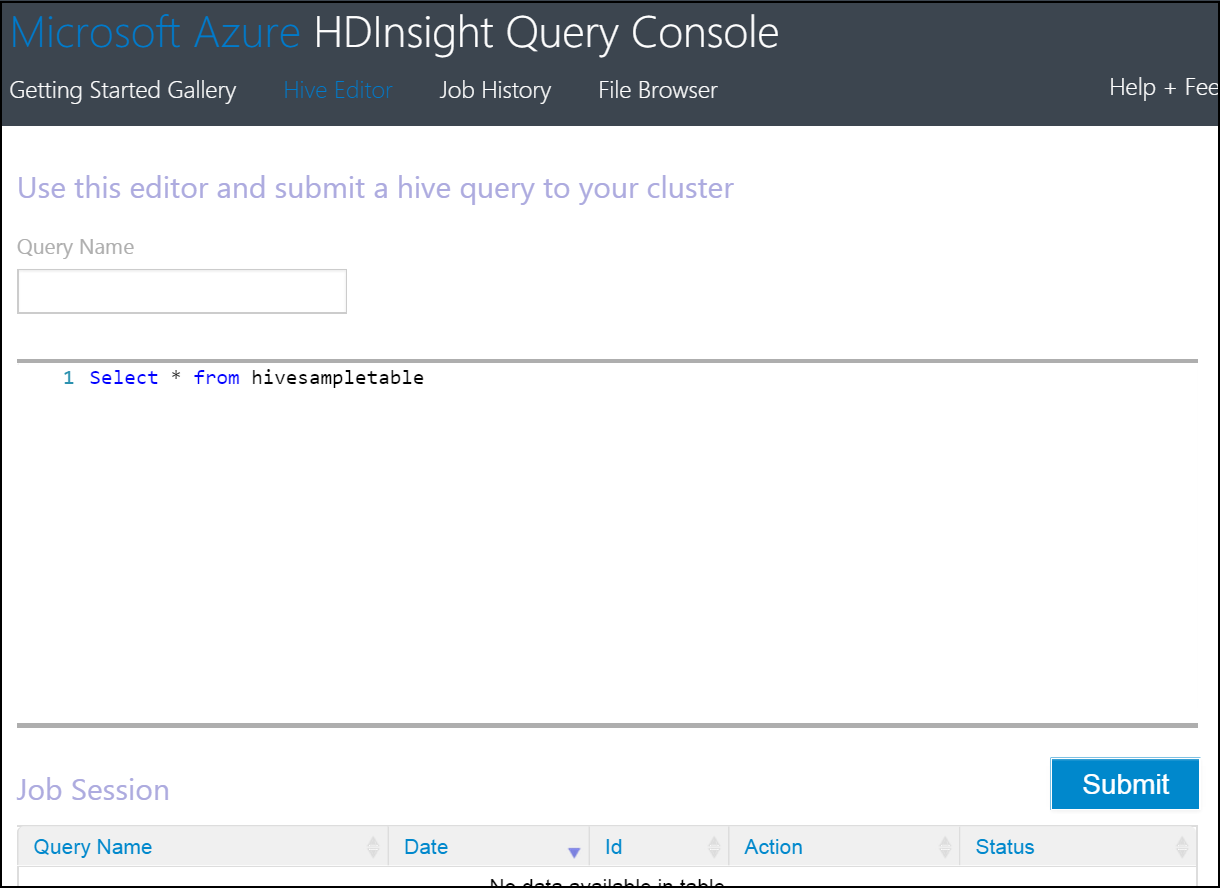
1. Enter the credentials you initially used when creating the HDInsight cluster when presented with the login dialog.

* 

1. The **Getting Started Gallery** will be displayed. Several samples and common solutions are available from the gallery.

* 

1. Click on the Hive Editor link at the top of the screen. Note that a sample query is loaded into the query editor.

* 

# 4. Building HIVE Queries

Hive supports both INTERNAL and EXTERNAL tables.  
- Use the CREATE EXTERNAL TABLE command to create the table without moving the raw data. The raw data is not affected by any operation, including a DROP TABLE or when you drop the Hadoop cluster.  
- The CREATE TABLE statement will create an internally managed table. When you drop this table, the data is also deleted.

Hive processes may include using intermediate tables to transform data. These tables typically do not need to persist when you drop the cluster. These intermediate tables are best suited for internally managed tables. Final processed data is best suited for EXTERNAL tables. This will provide the benefit of dropping a cluster without losing the processed data.

## 4.1 Creating the HIVE source table

The following steps will associate the source data loaded Azure Blob Storage with a Hive table.

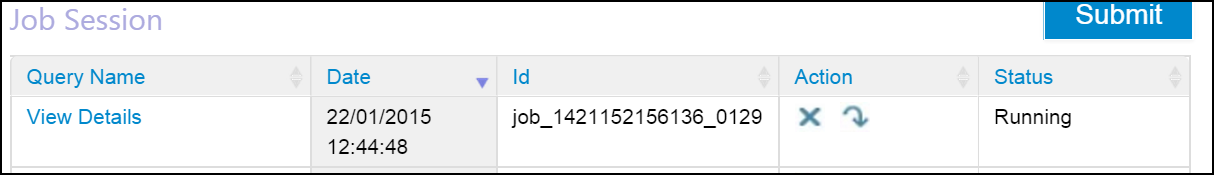
1. Copy the following hive query into the editor window to create an EXTERNAL table over the data loaded in Hands on Lab 1. A copy of this statement is stored on the course virtual machine at \*\*C:\_CreateDeviceReadings.txt\*\*.

* Note that in this case the data file is loaded prior to the schema. This demonstrates the "schema on read" nature of Hive, which means the schema defined in the CREATE TABLE statement is applied when a SELECT statement is issued on the table.
* DROP TABLE IF EXISTS DeviceReadings;  
    
  CREATE EXTERNAL TABLE DeviceReadings (  
   type string, sensorDateTime string, deviceId string, roomNumber int, reading float  
  )  
  ROW FORMAT DELIMITED FIELDS TERMINATED BY '\054'  
   STORED AS TEXTFILE  
   LOCATION 'wasb://data@<storage account name>.blob.core.windows.net/input';
* n.b. Update the last line in query and replace with the storage account you created in Hands on Lab 1.

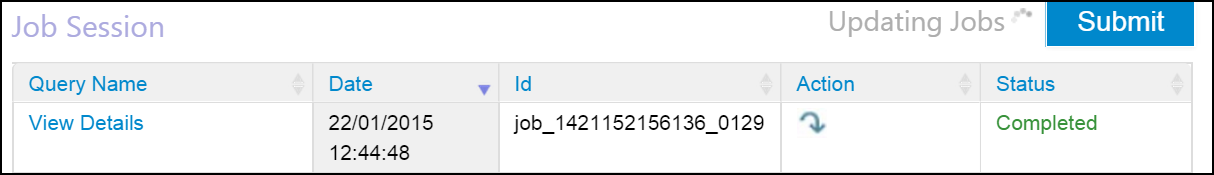
1. Click the Submit button to execute the query.

* batchAnalysisImg4.png

1. After a short delay a new Job will appear in the Job Session section with a status of Running.

* 

1. Wait for the job to complete.

* 

1. Replace the text in the Hive editor window with the following to validate the table was created and the data was loaded correctly.

* SELECT deviceId FROM DeviceReadings;

When the query completes, click on "View Details" in the Job Session. The Job Output should list the deviceId values in the data.csv file.

NOTE: Typically data is captured by calendar day, so, the input folder would be organised as input/yyyy/MM/dd. The table creation script above would then be modified to include a partition strategy for new input folders.

```SQL  
PARTITIONED BY (year string, month string, day string)  
```

## 4.2 Average reading by device type query

1. Copy the following query and replace the text in the editor window to create a table for aggregated data and insert data from the previously created DeviceReadings table. A copy of this statement is stored on the course virtual machine at \*\*C:\_CreateAverageReadingByType.txt\*\*.

* It is common in Hadoop batch processing to iterate through data processing of source data and store denormalized data.
* DROP TABLE IF EXISTS AverageReadingByType;  
    
  CREATE EXTERNAL TABLE AverageReadingByType (type string, reading float)  
   row format delimited   
   fields terminated by '\t'   
   lines terminated by '\n'   
   stored as textfile location 'wasb://data@<storage account name>.blob.core.windows.net/output/averageReadingByType';  
    
    
  INSERT INTO TABLE AverageReadingByType SELECT TYPE, avg(reading)   
   FROM DeviceReadings   
   GROUP BY TYPE;
* n.b. Update the last line in the CREATE TABLE query and replace with the storage account you created in Hands on Lab 1.

1. Click the Submit button to execute the query and wait for the job to complete.
2. After the job has completed, click File Explorer from the top menu of HDInsight Query Console.
3. To view the files generated with the previous Hive statement, click through the following path: **<storage account name> > data > output > averageReadingByType**.
4. Click on the file with a numeric name and select **Save** when prompted. **Open** the file using Notepad. Review the output.

## 4.3 Average reading by minute query

1. Return to the Hive Editor. Copy the following query to the editor window to aggregate an average reading by minute. A copy of this statement is stored on the course virtual machine at \*\*C:\_CreateAverageReadingByMinute.txt\*\*.

* DROP TABLE IF EXISTS AverageReadingByMinute;  
    
  CREATE EXTERNAL TABLE AverageReadingByMinute (type string, sensorDateTime string, roomNumber int, reading float)  
   row format delimited   
   fields terminated by '\t'   
   lines terminated by '\n'   
   stored as textfile location 'wasb://data@<storage account name>.blob.core.windows.net/output/averageReadingByMinute';  
    
  INSERT INTO TABLE AverageReadingByMinute SELECT TYPE, concat(substr(sensorDateTime, 1, 16), ":00.0000000Z"), roomNumber, avg(reading)   
   FROM DeviceReadings   
   WHERE roomNumber IS NOT NULL   
   GROUP BY TYPE, concat(substr(sensorDateTime, 1, 16), ":00.0000000Z"), roomNumber;
* n.b. Update the last line in the CREATE TABLE query and replace with the storage account you created in Hands on Lab 1.

1. Click the Submit button to execute the query and wait for the job to complete.

## 4.4 Maximum device reading

1. Copy the following query to the editor window to create a table with maximum device values. A copy of this statement is stored on the course virtual machine at \*\*C:\_CreateMaximumReading.txt\*\*.

* DROP TABLE IF EXISTS MaximumReading;  
    
  CREATE EXTERNAL TABLE MaximumReading (type string, sensorDateTime string, roomNumber int, maxReading float)  
   row format delimited   
   fields terminated by '\t'   
   lines terminated by '\n'   
   stored as textfile location 'wasb://data@<storage account name>.blob.core.windows.net/output/maximumReading';  
    
  INSERT INTO TABLE MaximumReading   
  SELECT mr.type, min(mr.sensorDateTime), min(mr.roomNumber), mr.reading FROM (  
   SELECT a.type, a.sensorDateTime, a.roomNumber, a.reading from DeviceReadings a  
   JOIN (SELECT type, max(reading) reading FROM DeviceReadings GROUP BY type ) b  
   ON (a.type = b.type and a.reading = b.reading)   
  ) mr   
  GROUP BY mr.type, mr.reading;
* n.b. Update the last line in the CREATE TABLE query and replace with the storage account you created in Hands on Lab 1.

1. Click the Submit button to execute the query and wait for the job to complete.

## 4.5 Minimum device reading

1. Copy the following query to the editor window to create a table to store minimum device reading values. A copy of this statement is stored on the course virtual machine at \*\*C:\_CreateMinimumReading.txt\*\*.

* DROP TABLE IF EXISTS MinimumReading;  
    
  CREATE EXTERNAL TABLE MinimumReading (type string, sensorDateTime string, roomNumber int, maxReading float)  
   row format delimited   
   fields terminated by '\t'   
   lines terminated by '\n'   
   stored as textfile location 'wasb://data@<storage account name>.blob.core.windows.net/output/minimumReading';  
    
  INSERT INTO TABLE MinimumReading   
  SELECT mr.type, min(mr.sensorDateTime), min(mr.roomNumber), mr.reading FROM (  
   SELECT a.type, a.sensorDateTime, a.roomNumber, a.reading from DeviceReadings a  
   JOIN (SELECT type, min(reading) reading FROM DeviceReadings GROUP BY type ) b  
   ON (a.type = b.type and a.reading = b.reading)   
  ) mr   
  GROUP BY mr.type, mr.reading;
* n.b. Update the last line in the CREATE TABLE query and replace with the storage account you created in Hands on Lab 1.

1. Click the Submit button to execute the query and wait for the job to complete.
2. Close the Hive Editor.

# 5. Drop the Cluster

The nature of HDInsight allows you to create clusters on demand, and only pay for the time needed to process data. Once we completed the above steps, the HDInsight cluster may be dropped. Data in the storage accounts will still be available for analysis, as demonstrated later in the lab.

1. Return to the HDInsight cluster in the Azure Management Portal.
2. Click **Delete** at the bottom of the console. Click **Yes** when prompted to verify the delete action.

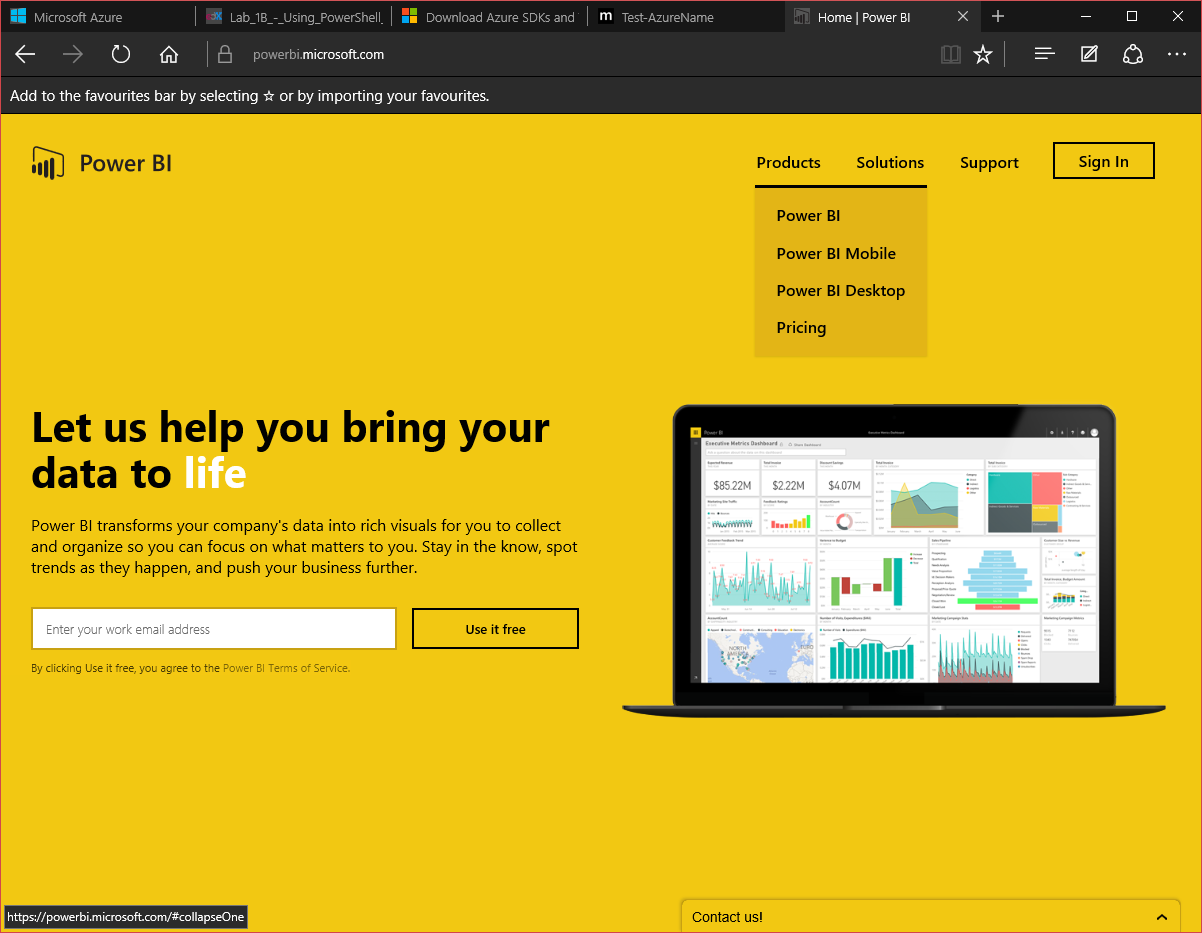
At the end of this section you will have created five new Hive tables, and output data to the storage account. The next sections will demonstrate using Microsoft Power BI to connect to the data, shape the data, and create visualizations.

# 6. HDInsight Integration with Microsoft Power BI through PowerBI Desktop

The Microsoft BI tools that have been available in Excel and SharePoint such as Power BI, Power Pivot etc. have now been incorporated into a stand-alone cloud service, Power BI and a separate desktop designer Power BI Desktop.

With Power BI you can: there is a complete self service capability to extract transform and load data into into a powerful analytical engine on top of which you can create rich reports and interactive dashboards for less experienced users to explore.

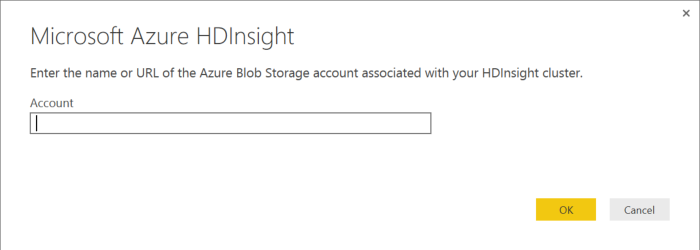
Power BI desktop can be freely downloaded from <https://powerbi.microsoft.com/> by selecting Product Power BI desktop..



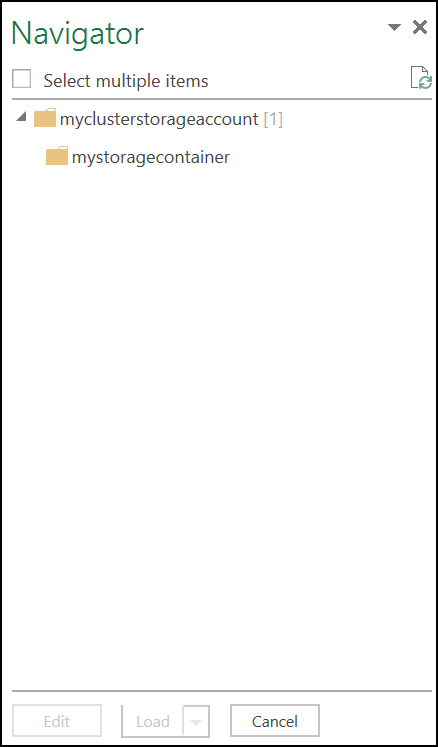
However to use the cloud service and publish Power BI models you’ll need to sign into the Power BI site with an **organizational account** – i.e. not a personal e-mail account like live.com, or google.com.

The following section will introduce a simple scenario of connecting to a Hadoop data source in HDInsight. This section will require the name of your blob storage account and the key associated with the account.

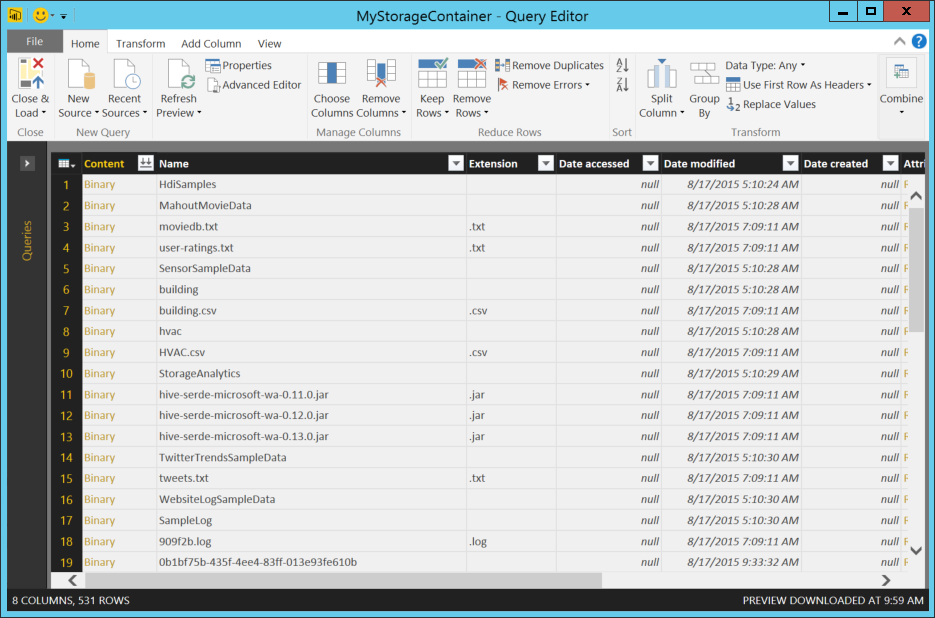
1. Open Power BI desktop.
2. Select Get Data in the ribbon at the top of the screen.
3. Note the various options in the ribbon. Select the **"From Azure"** dropdown select **"From Microsoft Azure HDInsight"** option..
4. Enter the name of your **blob storage account** associated with the HDInsight cluster.

* 

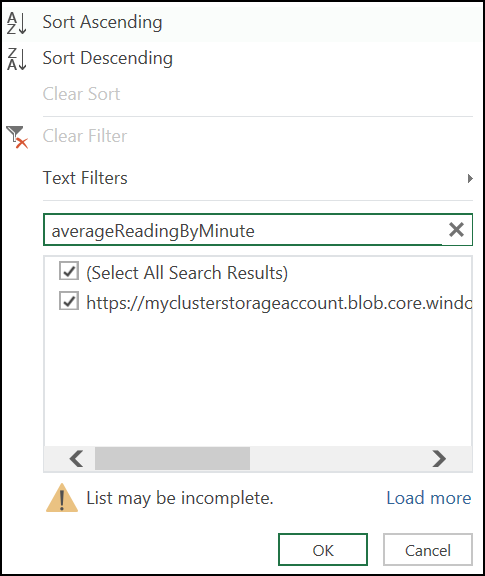
1. Enter your storage account key which should be saved to the desktop via notepad (from Hands on Lab 1) and click **Save**.
2. The Navigator pane is displayed to the right of the workbook. Double click the **data** container name.

* 

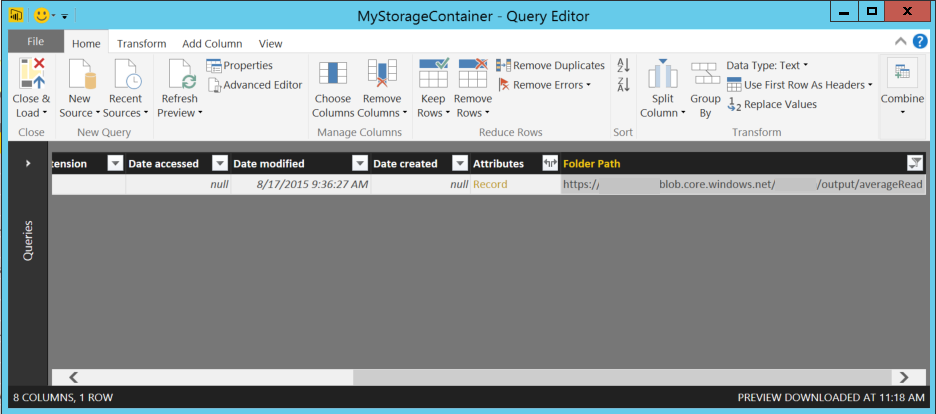
1. The Query Editor window will be load. The blob files in the storage container are displayed.

* 

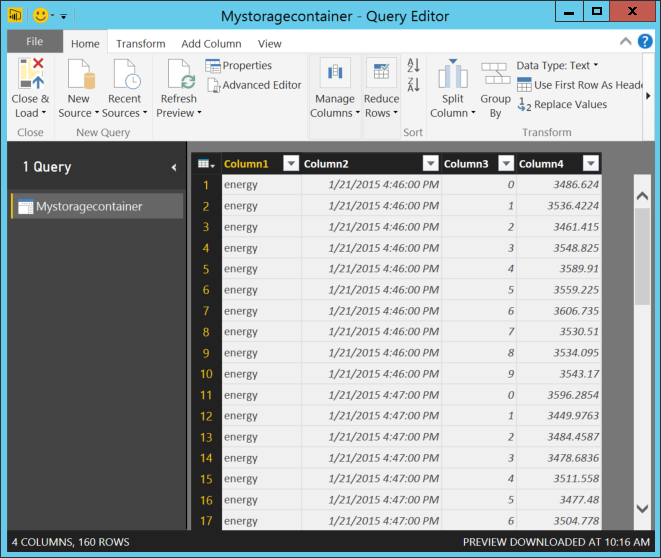
1. To extract the data from the HDInsight job we need to find the reference the output from our HIVE jobs. Click on the batchAnalysisImg12.png on the "Folder Path" column at the far right of the Query Editor window. Enter averageReadingByMinute in the text filter box.

* 

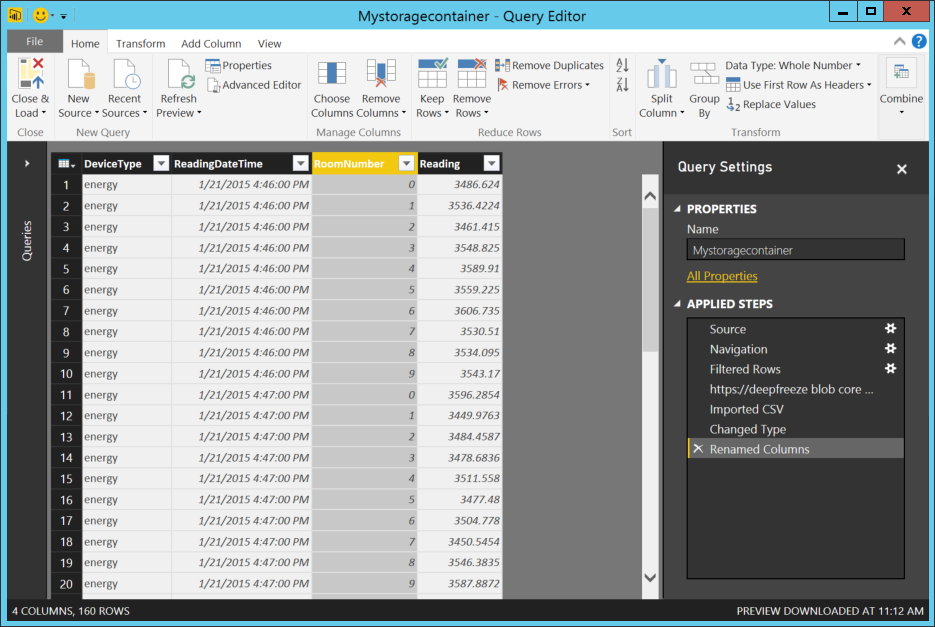
1. The content of the previous window will be updated and only one row will be shown..



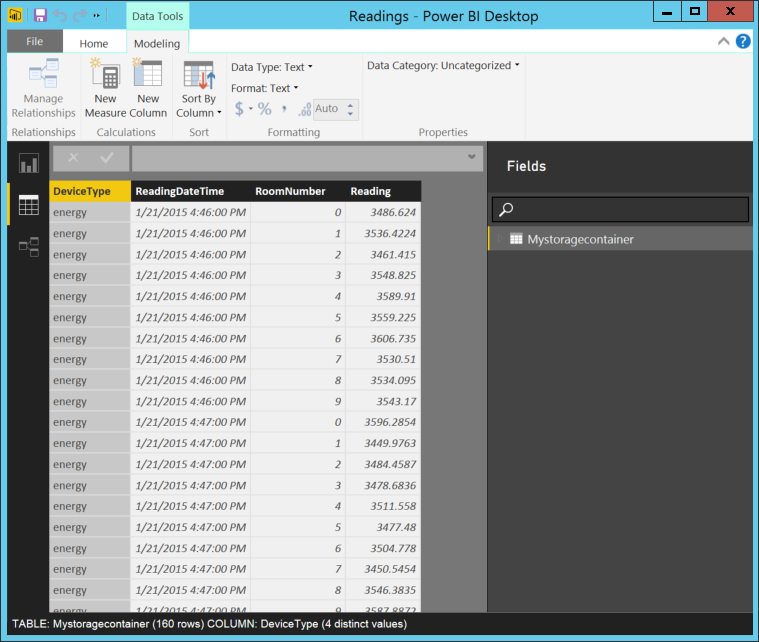
1. Select the batchAnalysisImg15.png link in the first column to import the csv.

* 

1. Notice the options in the ribbon. Power BI Desktop supports simple activites like adding and removing columns, and more advanced options like replacing values, aggregating columns, and pivoting/unpivoting data.
2. Rename columns the columns with the following names. Double-click each column header to rename.
   * DeviceType
   * ReadingDateTime
   * RoomNumber
   * Reading



1. Click **Close & Load** from the Home ribbon. A new sheet will be created (AverageReadingByMinute) in the workbook.

* 

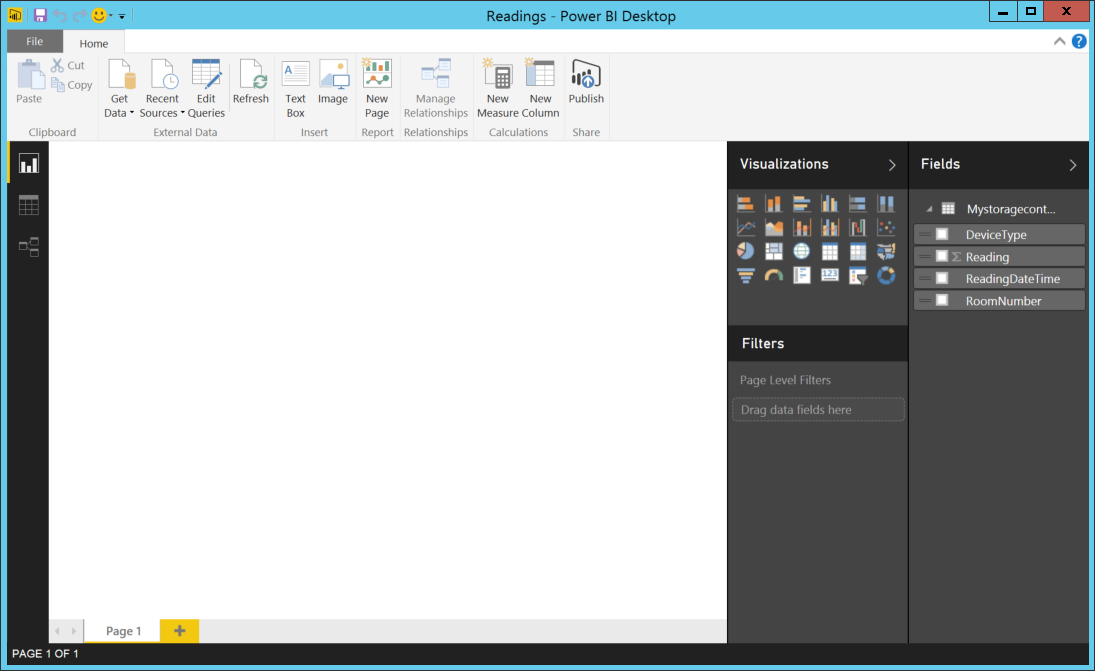
1. Change the RoomNumber column to data Type Text, so that it is not treated as a measure – we would never want to do calculations on the Room Number it is there as a description and needs to be treated as such.
2. Save the model to the desktop.

# 7. Visualizing the HIVE output with Microsoft Power BI through Power View

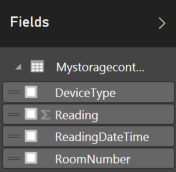
In this lab you will explore the visualization capabilities of Power View.

Power View is an Excel and web-based data exploration and report authoring tool that enables everyone to create compelling, interactive, and rich data visualizations.

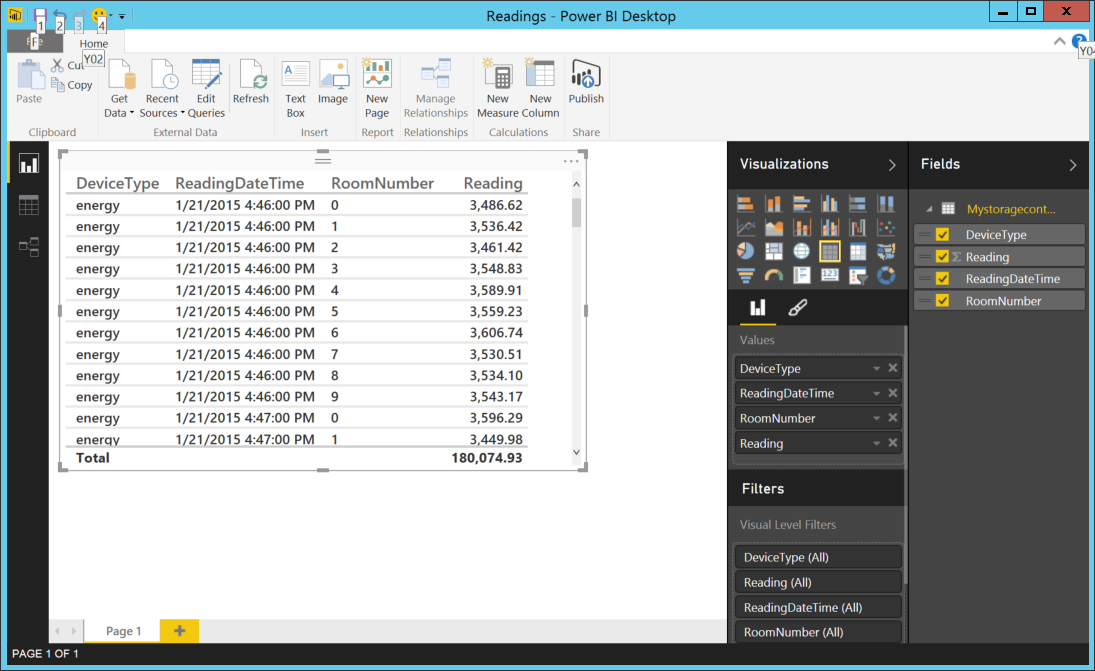
1. Return to the Power BI Desktop model created in the previous section and select the Dashboard icon 



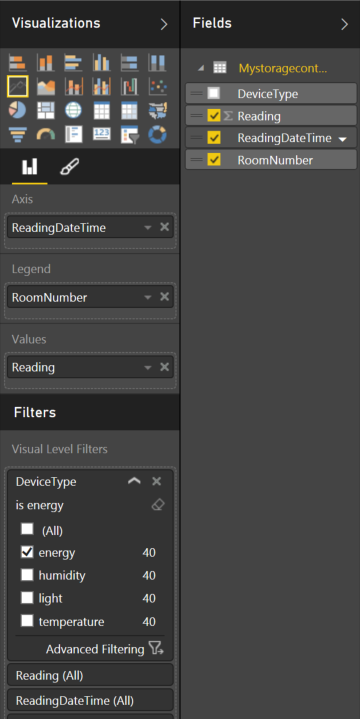
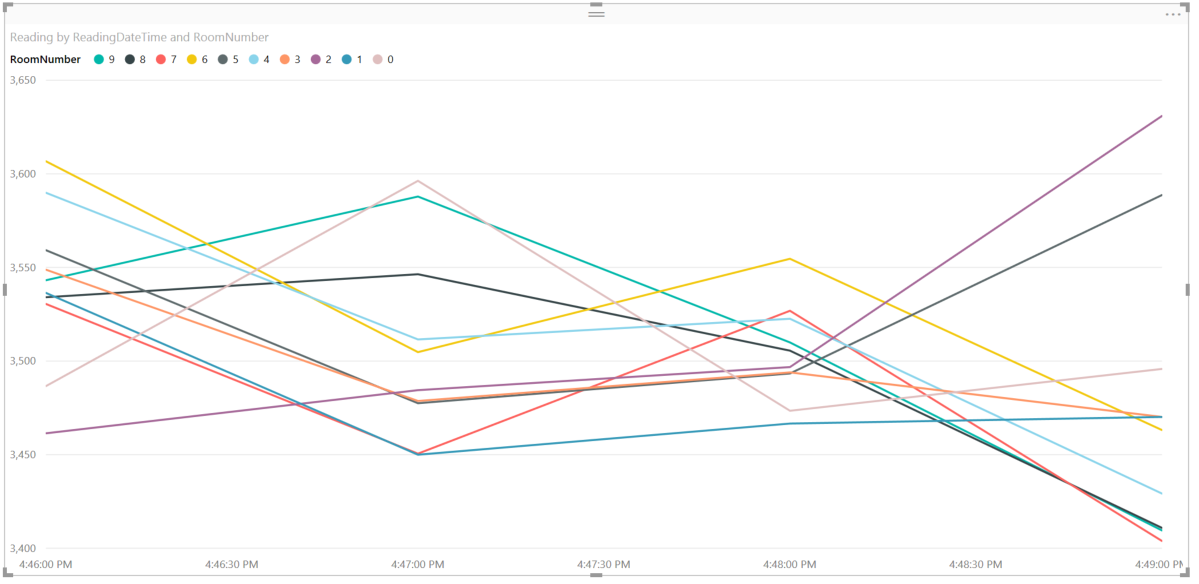
1. To create a graphical representation of the data we can create a new table. Note the **"Fields"** pane on the right hand side which has pre-selected all the fields from the table result of the previous section.

* 

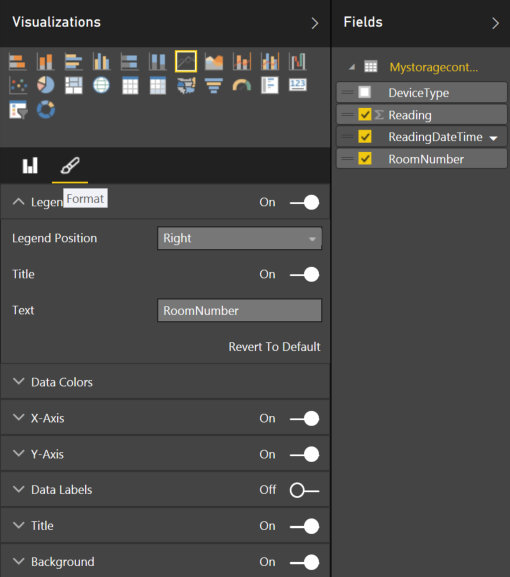
1. Note that only Reading shows up with the ∑ icon to indicate it is a measure.
2. The large blank pane beneath the ribbon is the dashboard design surface. This is a work space that supports an interactive design and data exploration experience and allows us to add multiple object which can all be kept in synch with one another. Select the table  icon and then from the fields pane, check DeviceType, ReadingDateTime, RoomNumber and Reading to get a grid of the data..

* 

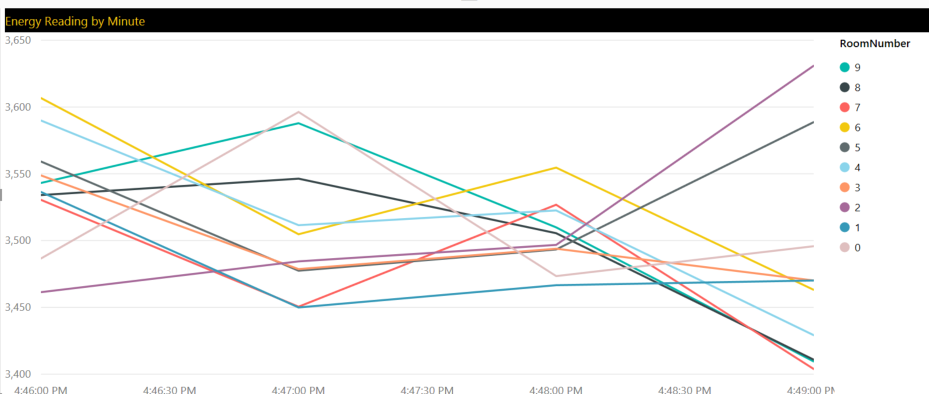
1. We can now change this to a simple line chart by checking the line chart icon Select the table in the report designer pane. At the top Design ribbon, click the line chart icon .

* Now we can change the properties of the chart to just show the energy readings (Device Type= Energy) over time (ReadingDateTime) by Room Number by setting the visualizations options as follows:
* 
* by dragging the fileds required to where they are shown. The end result is a chart like this..
* 

1. If we click on the paintbrush icon  in the Visualizations pane we can change the properties of the chart..

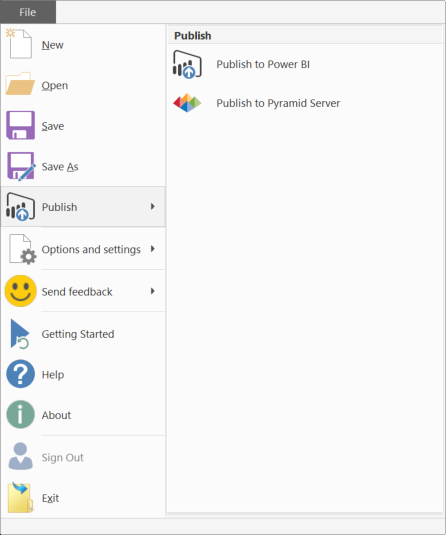


Move the legend to the right, and expand the title to change the title and background and foreground colour to give a chart like this..



# Using Microsoft Power BI Online

# (Optional if you have wish to sign up or have Power BI already)

* If you wish to sign up for a free trial of Power BI using an organizational account it is then a simple matter to upload your dashboard to the Power BI service from file publish to Power BI. Notice because Power BI is an extensible, it’s also possible to publish models to a Pyramid Analytics Pyramid Server..
* 

The report we created looks exactly the same but is now hosted on the Power BI server and is available through any modern browser ..

