Exercise #1:

HOW TO ANALYZE MACHINE AND SENSOR DATA

INTRODUCTION

This tutorial describes how to refine data from heating, ventilation, and air conditioning (HVAC) systems using the Hortonworks Data Platform, and how to analyze the refined sensor data to maintain optimal building temperatures.

SENSOR DATA

A sensor is a device that measures a physical quantity and transforms it into a digital signal. Sensors are always on, capturing data at a low cost, and powering the "Internet of Things."

POTENTIAL USES OF SENSOR DATA

Sensors can be used to collect data from many sources, such as:

- To monitor machines or infrastructure such as ventilation equipment, bridges, energy meters, or airplane engines. This data can be used for predictive analytics, to repair or replace these items before they break.
- To monitor natural phenomena such as meteorological patterns, underground pressure during oil extraction, or patient vital statistics during recovery from a medical procedure.

In this tutorial, we will focus on sensor data from building operations. Specifically, we will refine and analyze the data from Heating, Ventilation, Air Conditioning (HVAC) systems in 20 large buildings around the world.

OVERVIEW

To refine and analyze HVAC sensor data, we will:

- Download and extract the sensor data files.
- Load the sensor data into the Hortonworks Sandbox.
- Run two Hive scripts to refine the sensor data.
- Access the refined sensor data with Apache Zeppelin.
- Visualize the sensor data using Apache Zeppelin.

DOWNLOAD AND EXTRACT THE SENSOR DATA FILES

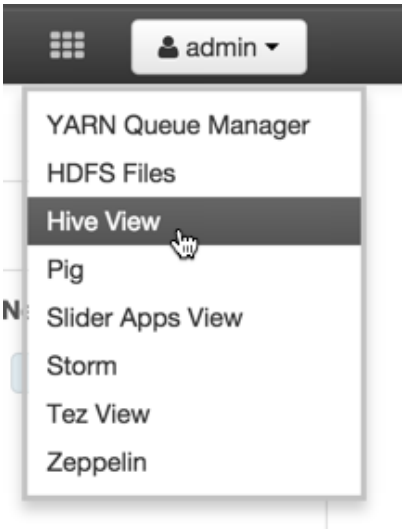
- You can download the sample sensor data contained in a compressed (.zip) folder here:
- SensorFiles.zip
- Save the `SensorFiles.zip` file to your computer, then extract the files. You should see a SensorFiles folder that contains the following files:
- `HVAC.csv` – contains the targeted building temperatures, along with the actual (measured) building temperatures. The building temperature data was obtained using Apache Flume. Flume can be usedas a log aggregator, collecting log data from many diverse sources and moving it to a centralized data store. In this case, Flume was used to capture the sensor log data, which we can now load into the Hadoop Distributed File System (HFDS). For more details on Flume, refer to Tutorial 13: Refining and Visualizing Sentiment Data
- `building.csv` – contains the "building" database table. Apache Sqoop can be used to transfer this type of data from a structured database into HDFS.

LAB 1 – LOAD DATA INTO HIVE

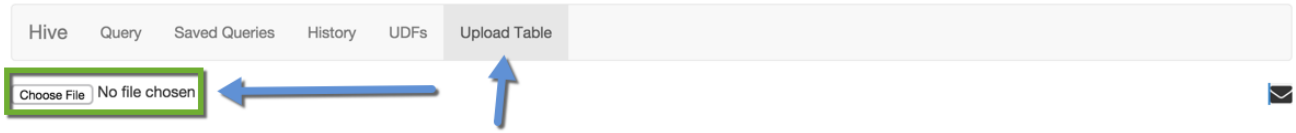
STEP 2: LOAD THE SENSOR DATA INTO THE HORTONWORKS SANDBOX

- Navigate to the ambari login by going to the web address `http://localhost:8080`
- Login with the username `maria_dev` and password `maria_dev`.

Head on over to the Hive view using the dropdown menu in the top-right corner.



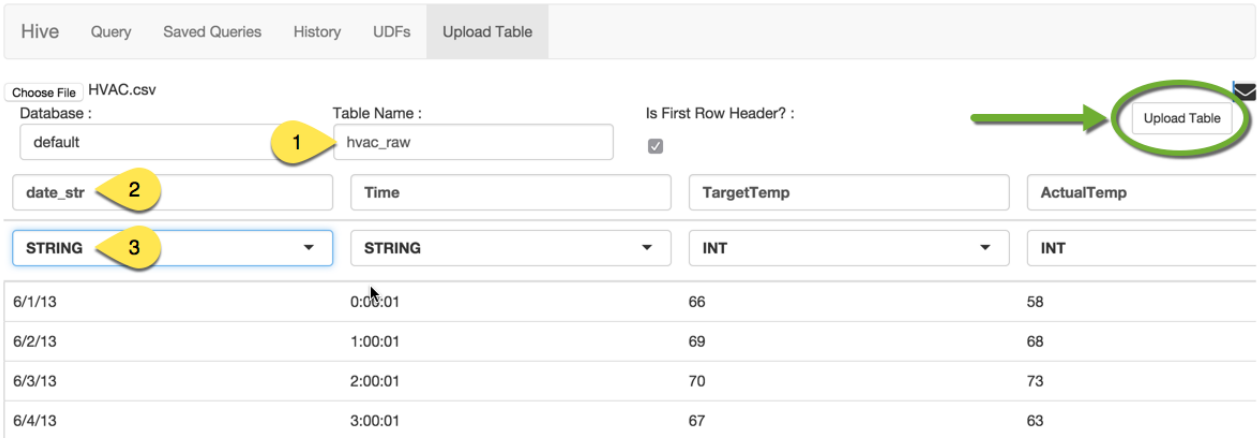
Then use the **Upload Table** tab to upload the two csv files contained within `SensorFiles.zip`



When uploading the two tables we'll need to change a few things.

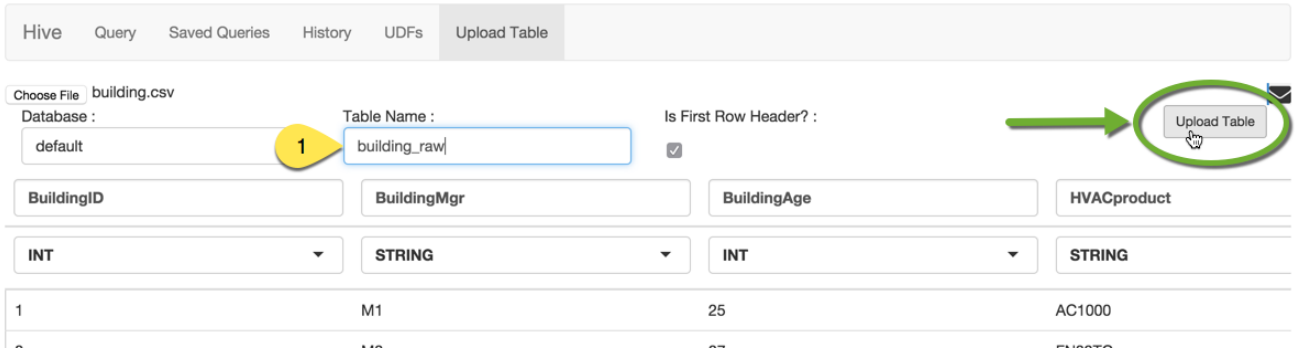
For `HVAC.csv`

- Change the table name to `hvac_raw`
- Change the name of the `Date` column to `date_str`
- Change the type of the `date_str` column from `DATE` to `STRING`



For `buildings.csv`

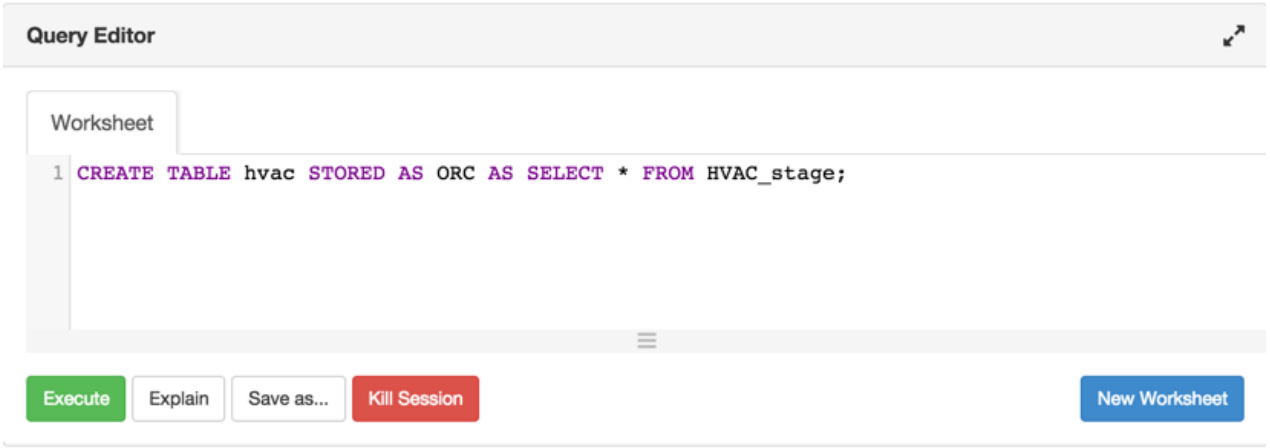
- Change the table name to `building_raw`



- Now that we have both tables loaded in, we want to get better performance in Hive, so we're going to create new tables that utilize the highly efficient ORC file format. This will allow for faster queries when our datasets are much larger.

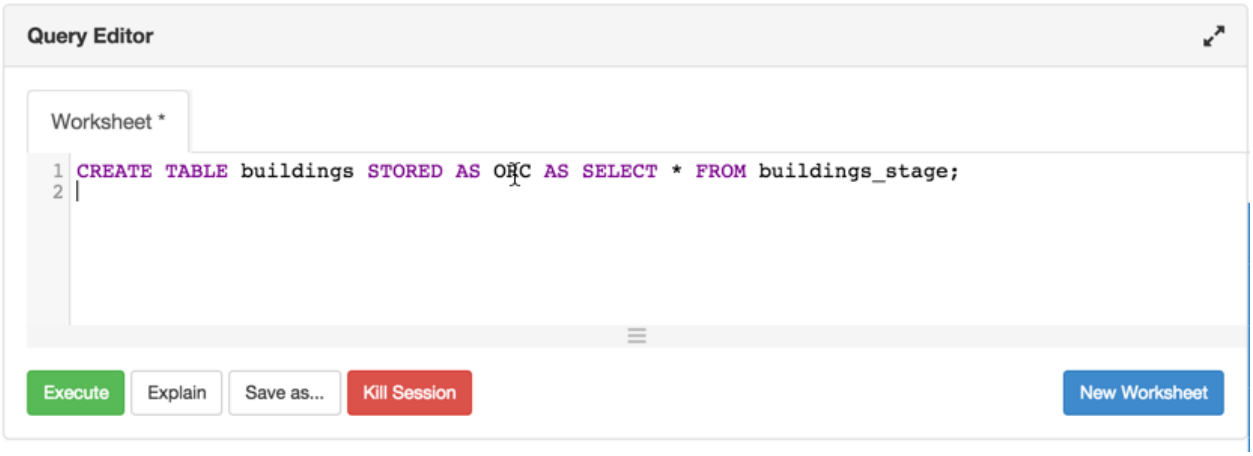
- Execute the following query to create a new table `hvac` that is stored as an ORC file.

```
CREATE TABLE hvac STORED AS ORC AS SELECT * FROM hvac_raw;
```



- Repeat the previous step, except this time we will make a table for `buildings`.

```
CREATE TABLE buildings STORED AS ORC AS SELECT * FROM building_raw;
```



STEP 3: RUN TWO HIVE SCRIPTS TO REFINE THE SENSOR DATA

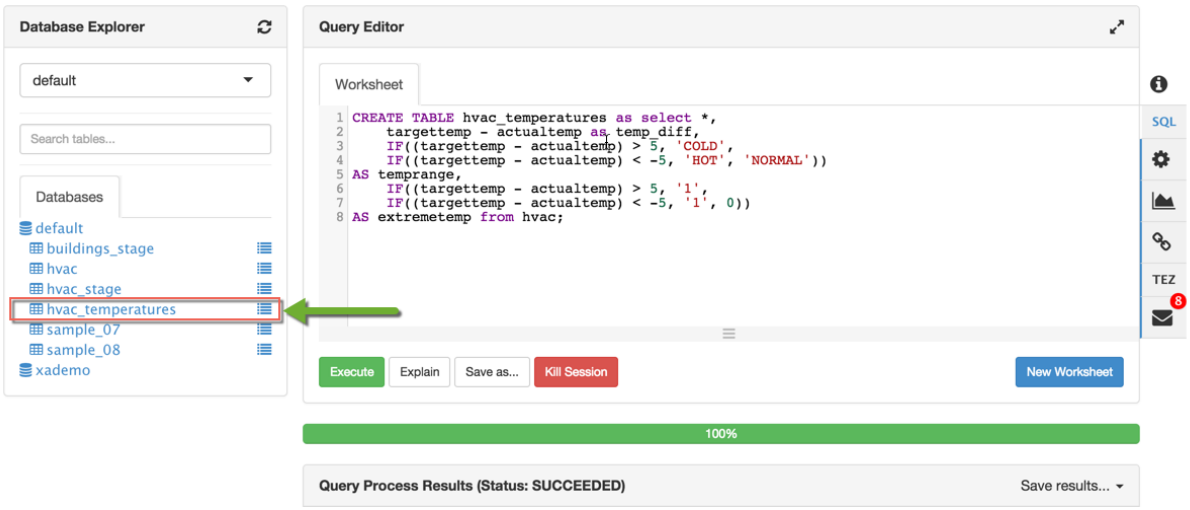
We will now use two Hive scripts to refine the sensor data. We hope to accomplish three goals with this data:

- Reduce heating and cooling expenses.
- Keep indoor temperatures in a comfortable range between 65-70 degrees.
- Identify which HVAC products are reliable, and replace unreliable equipment with those models.

- First, we will identify whether the actual temperature was more than five degrees different from the target temperature.
- Create a new worksheet in the Hive view and paste the following Hive query into your window.

```
CREATE TABLE hvac_temperatures as
select *, targettemp - actualtemp as temp_diff,
IF((targettemp - actualtemp) > 5, 'COLD',
IF((targettemp - actualtemp) < -5, 'HOT', 'NORMAL'))
AS temprange,
IF((targettemp - actualtemp) > 5, '1',
IF((targettemp - actualtemp) < -5, '1', 0))
AS extremetemp from hvac;
```

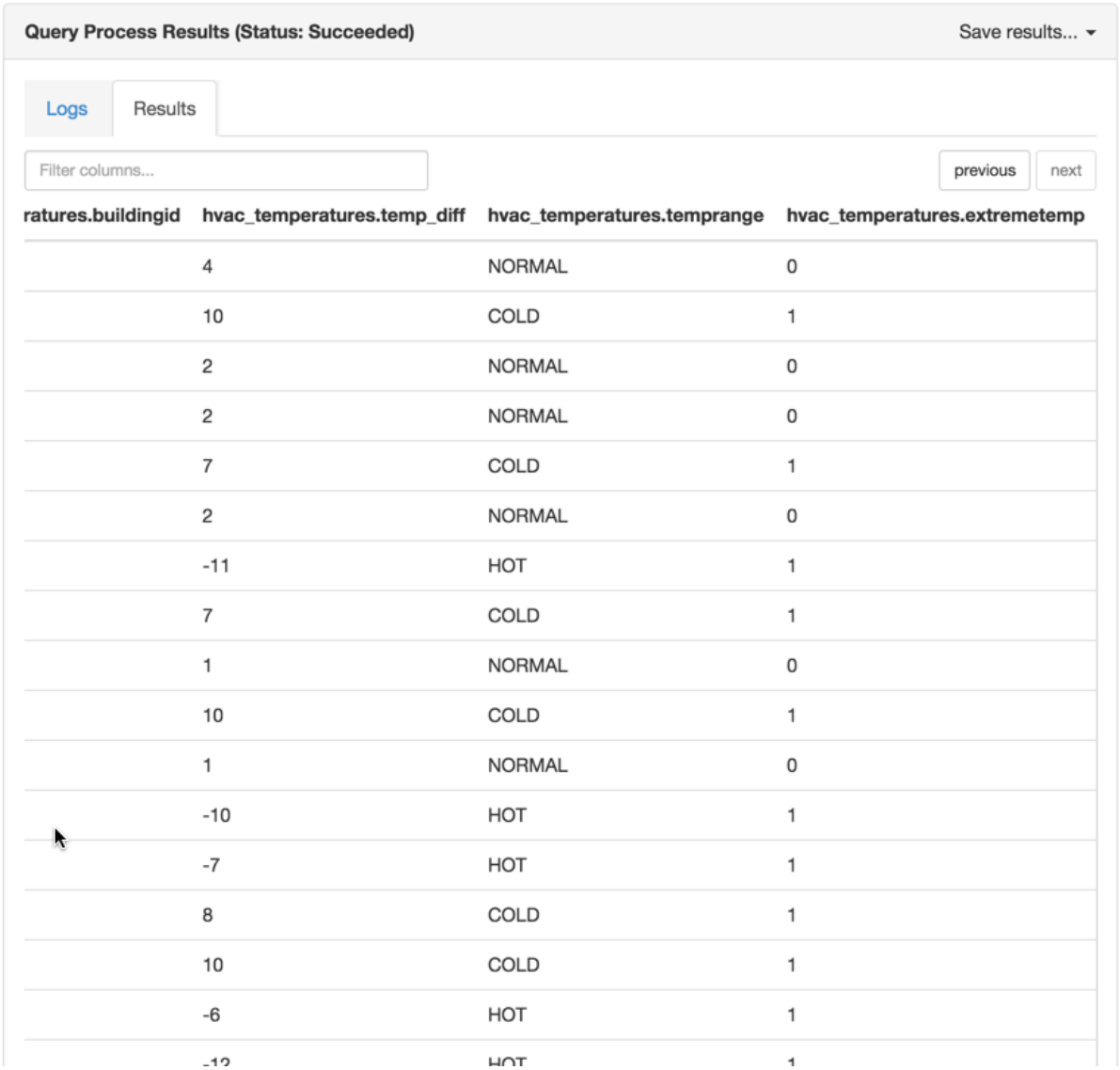
- This query creates a new table `hvac_temperatures` and copies data from the `hvac` table
- After you paste the query use **Execute** to create the new table.



- On the Query Results page, use the slider to scroll to the right. You will notice that two new attributes appear in the `hvac_temperatures` table.

  The data in the **temprange** column indicates whether the actual temperature was:
- **NORMAL –** within 5 degrees of the target temperature.
- **COLD –** more than five degrees colder than the target temperature.
- **HOT –** more than 5 degrees warmer than the target temperature.

  If the temperature is outside of the normal range, `extremetemp` is assigned a value of 1; otherwise its value is 0.

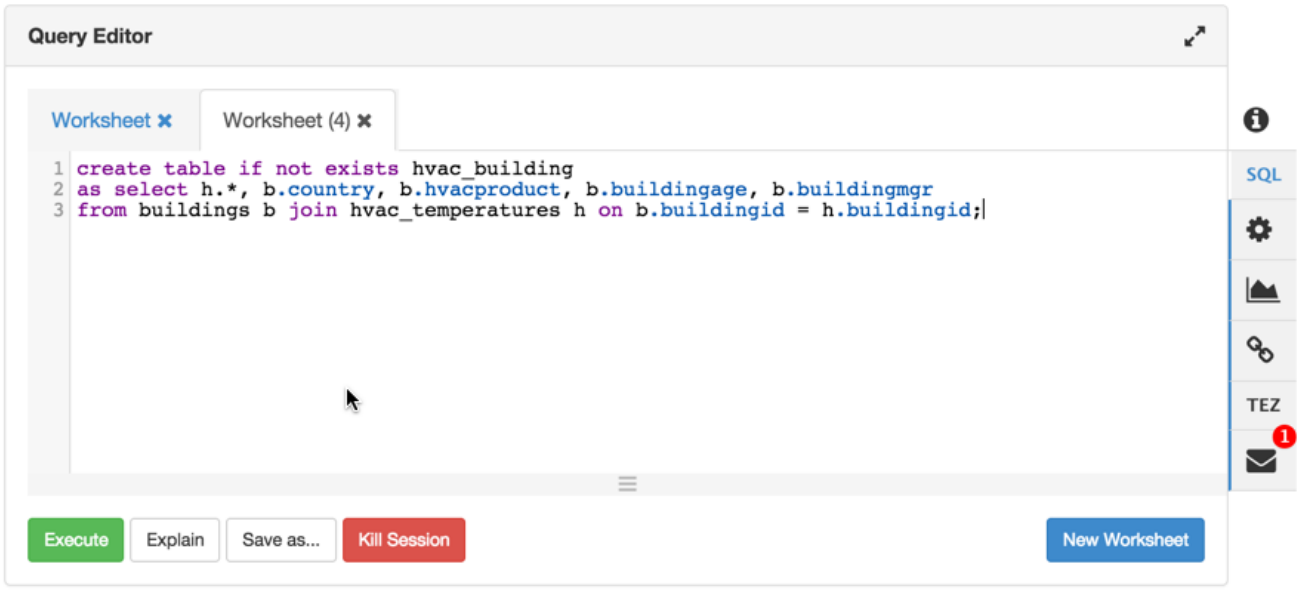


ratures.buildingid	hvac_temperatures.temp_diff	hvac_temperatures.temprange	hvac_temperatures.extremetemp
	4	NORMAL	0
	10	COLD	1
	2	NORMAL	0
	2	NORMAL	0
	7	COLD	1
	2	NORMAL	0
	-11	HOT	1
	7	COLD	1
	1	NORMAL	0
	10	COLD	1
	1	NORMAL	0
	-10	HOT	1
	-7	HOT	1
	8	COLD	1
	10	COLD	1
	-6	HOT	1
	-12	HOT	1

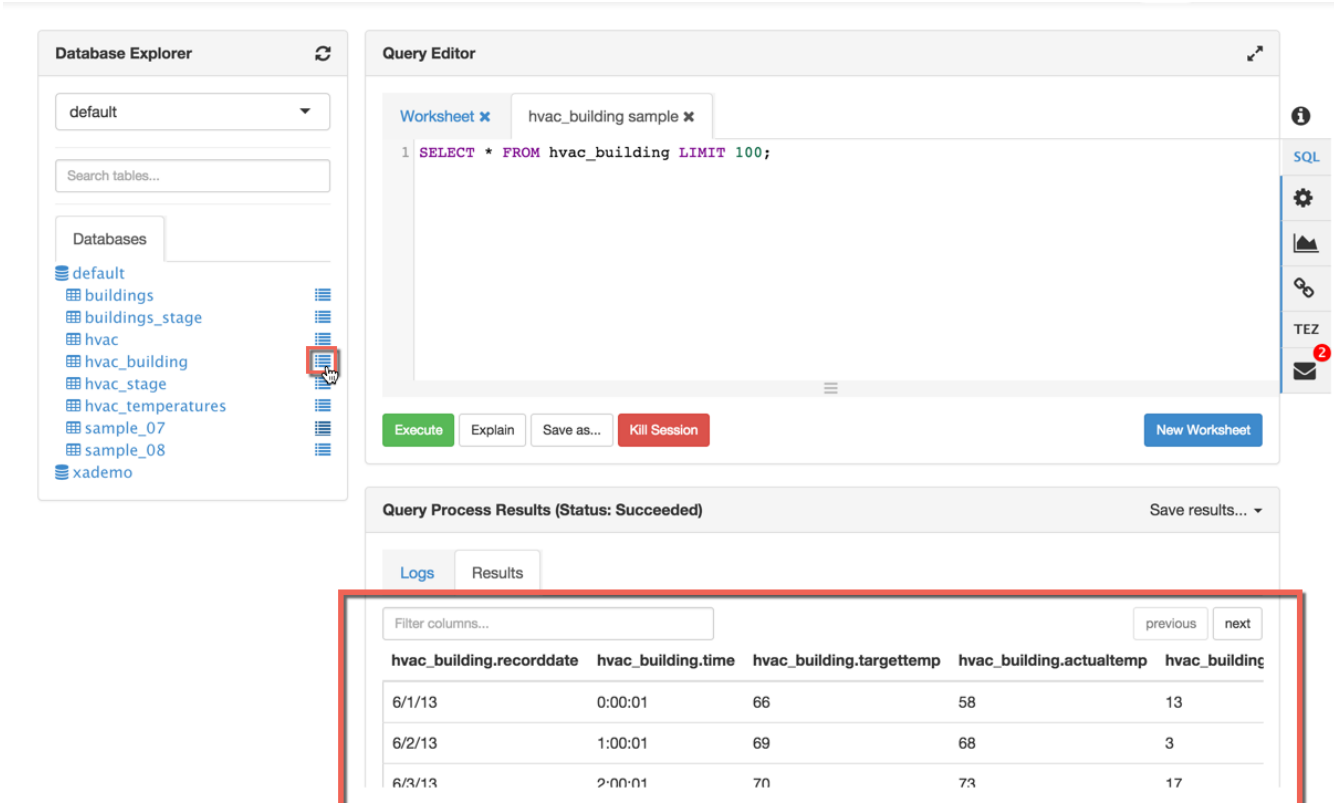
- Next we will combine the **hvac** and **hvac_temperatures** data sets.
  Create a new worksheet in the hive view and use the following query to create a new table `hvac_building` that contains data from the `hvac_temperatures` table and the `buildings` table.

```
create table if not exists hvac_building
as select h.*, b.country, b.hvacproduct, b.buildingage, b.buildingmgr
from buildings b join hvac_temperatures h on b.buildingid = h.buildingid;
```

- Use **Execute** to run the query that will produce the table with the intended data.



- After you've successfully executed the query, use the database explorer to load a sample of the data from the new `hvac_building` table.



Now that we've constructed the data into a useful format, we can use different reporting tools to analyze the results.

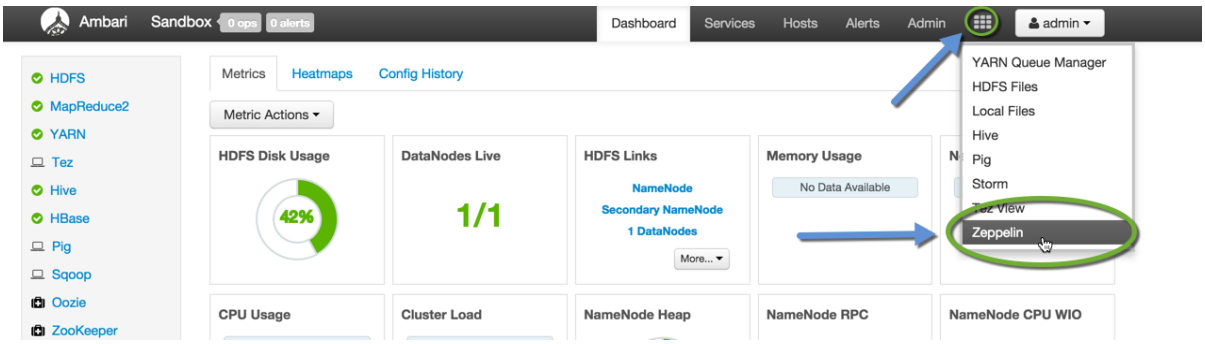
ACCESS THE REFINED SENSOR DATA WITH APACHE ZEPPELIN

Apache Zeppelin makes data reporting easy on Hadoop. It has direct connections to Apache Spark and Hive in your cluster and allows you to create visualizations and analyze your data on the fly.

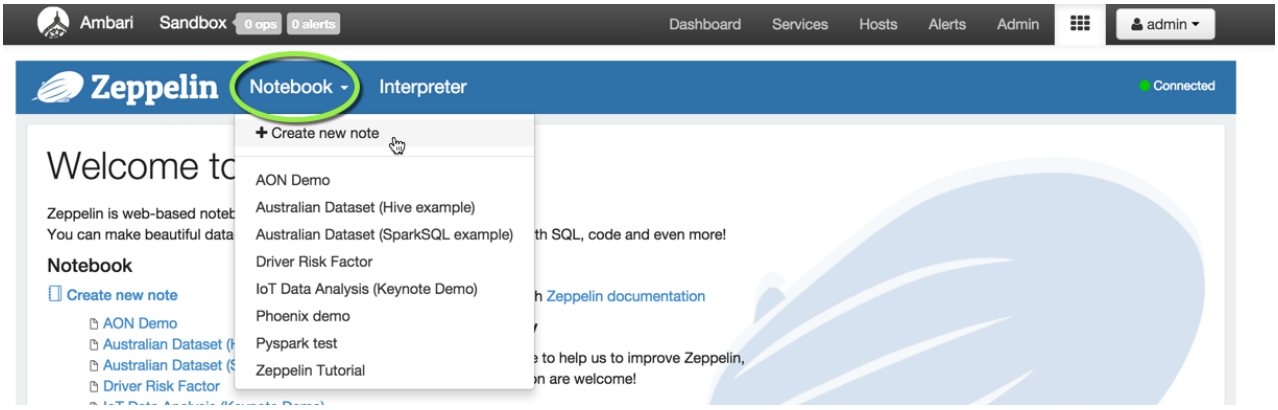
To start you're going to need to open up the Apache Zeppelin view in Ambari.

Start by navigating back to the Ambari Dashboard at `http://localhost:8080`

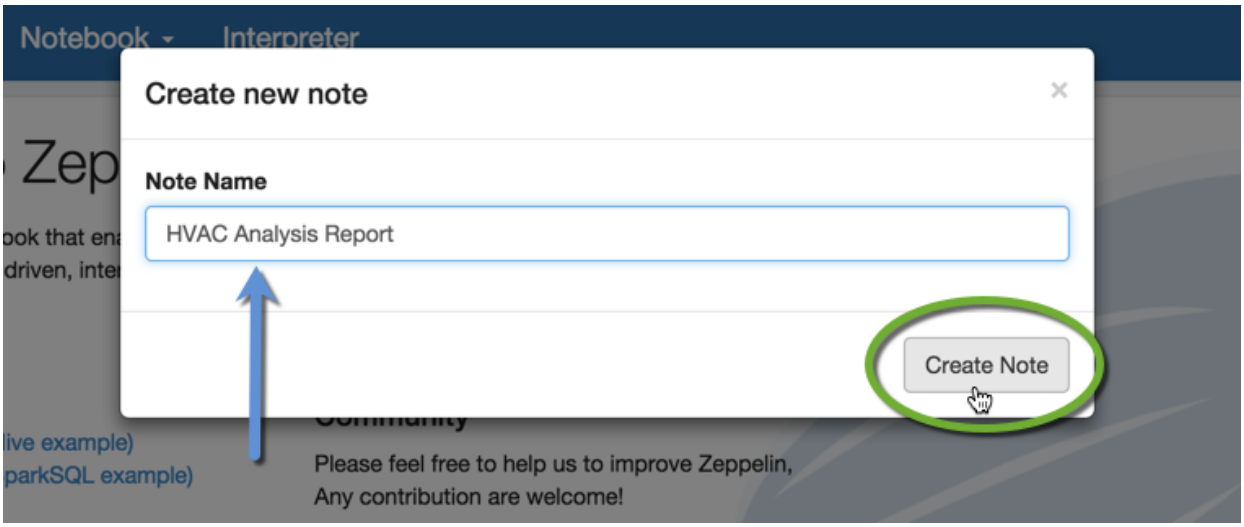
- Use the dropdown menu to open the Zeppelin View.



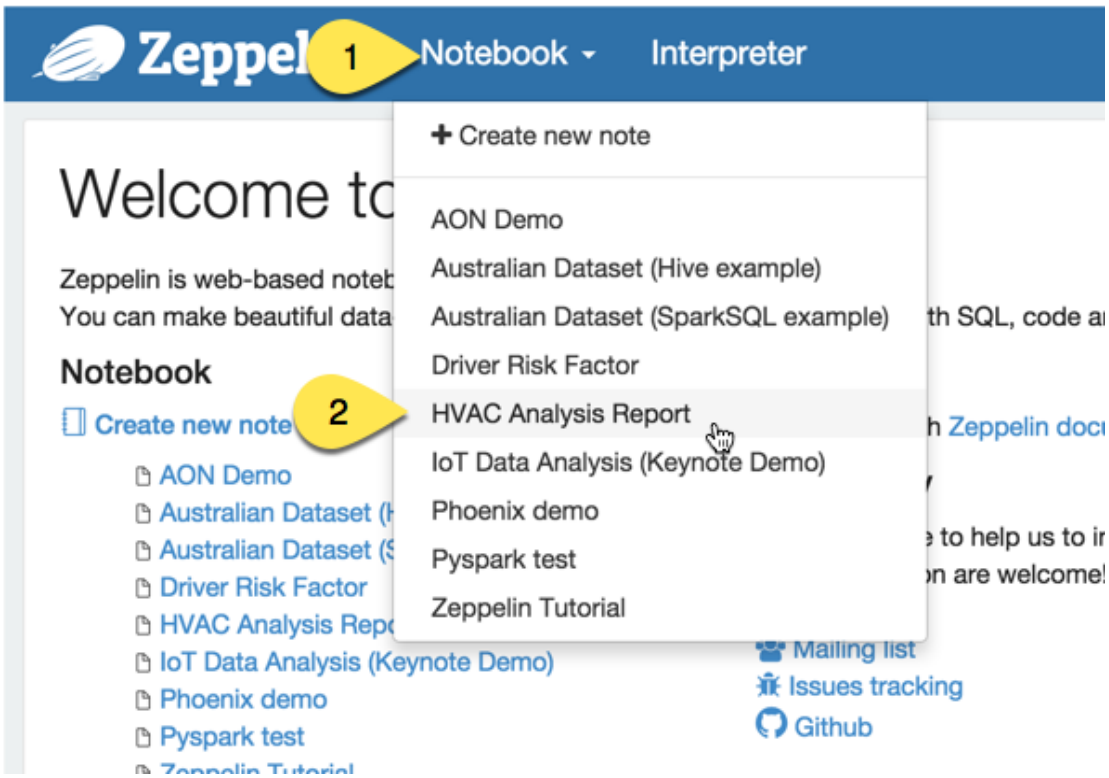
- From here we're going to need to create a new Zeppelin Notebook.
- Notebooks in Zeppelin is how we differentiate reports from one another.
- Hove over **Notebook**. Use the dropdown menu and **Create a new note**.



- Name the note **HVAC Analysis Report** and then **Create Note**.

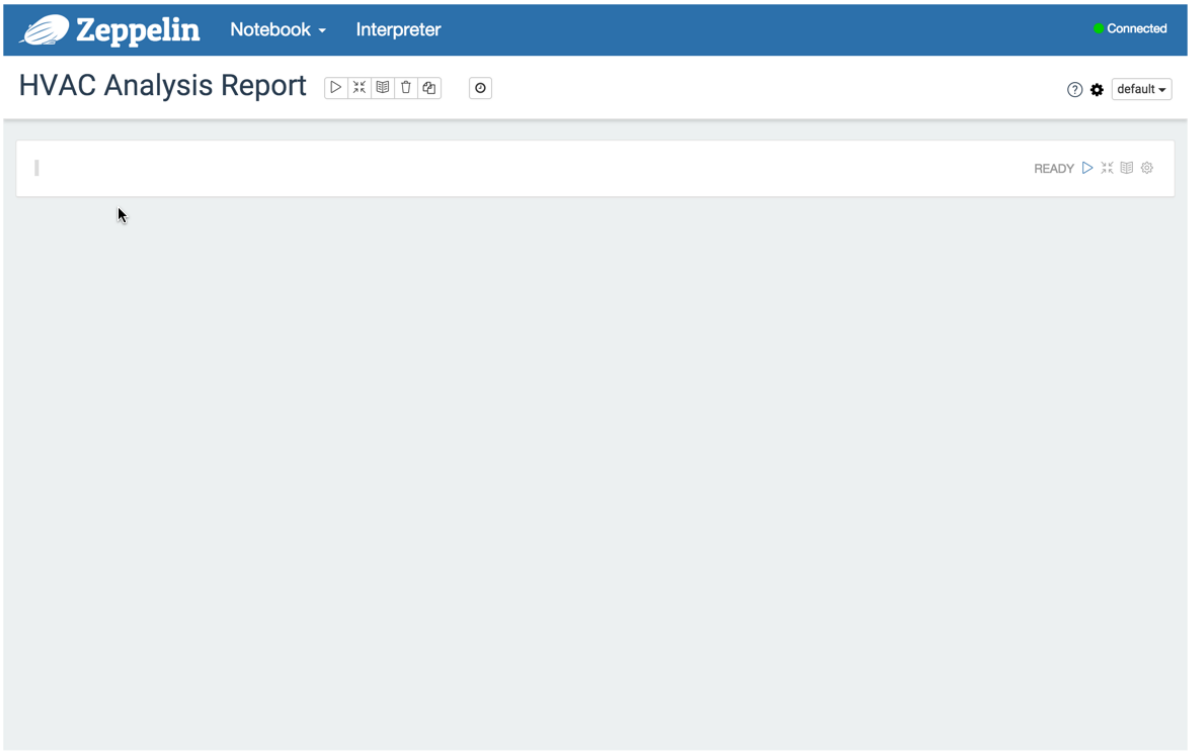


- Head back to the Zeppelin homepage.
- Use the **Notebook** dropdown menu to open the new notebook **HVAC Analysis Report**.



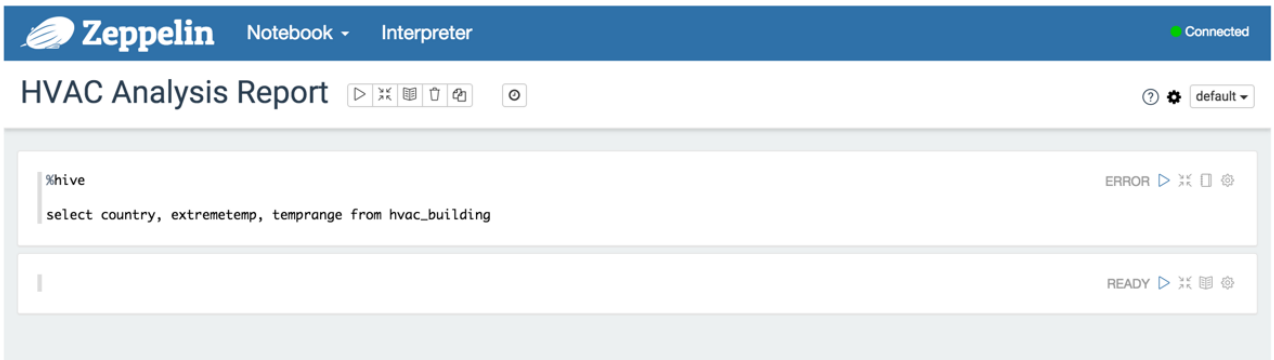
- Zeppelin integrates with Hadoop by using things called *interpreters*.
- In this tutorial we'll be working with the Hive interpreter to run Hive queries in Zeppelin, then visualize the results from our Hive queries directly in Zeppelin.

- To specify the Hive interpreter for this note, we need to put `%hive` at the top of the note. Everything afterwards will be interpreted as a Hive query.

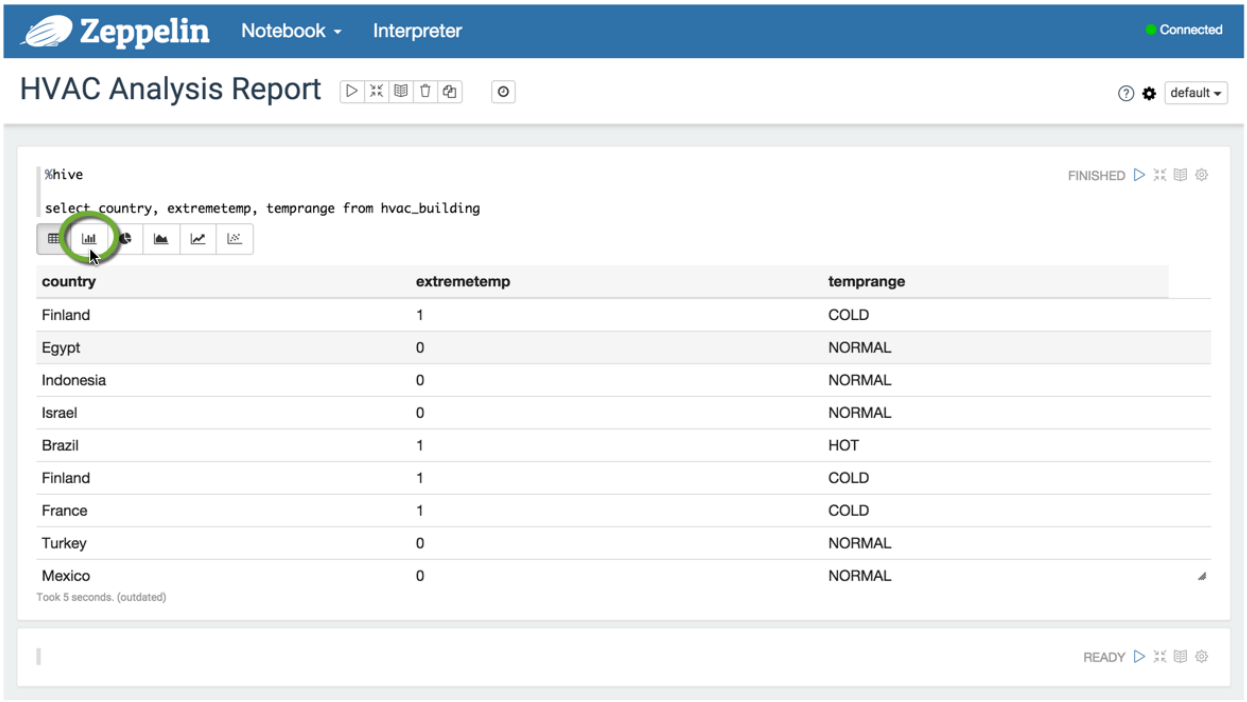


- Type the following query into the note, then run it by clicking the **Run** arrow or by using the shortcut **Shift+Enter**.

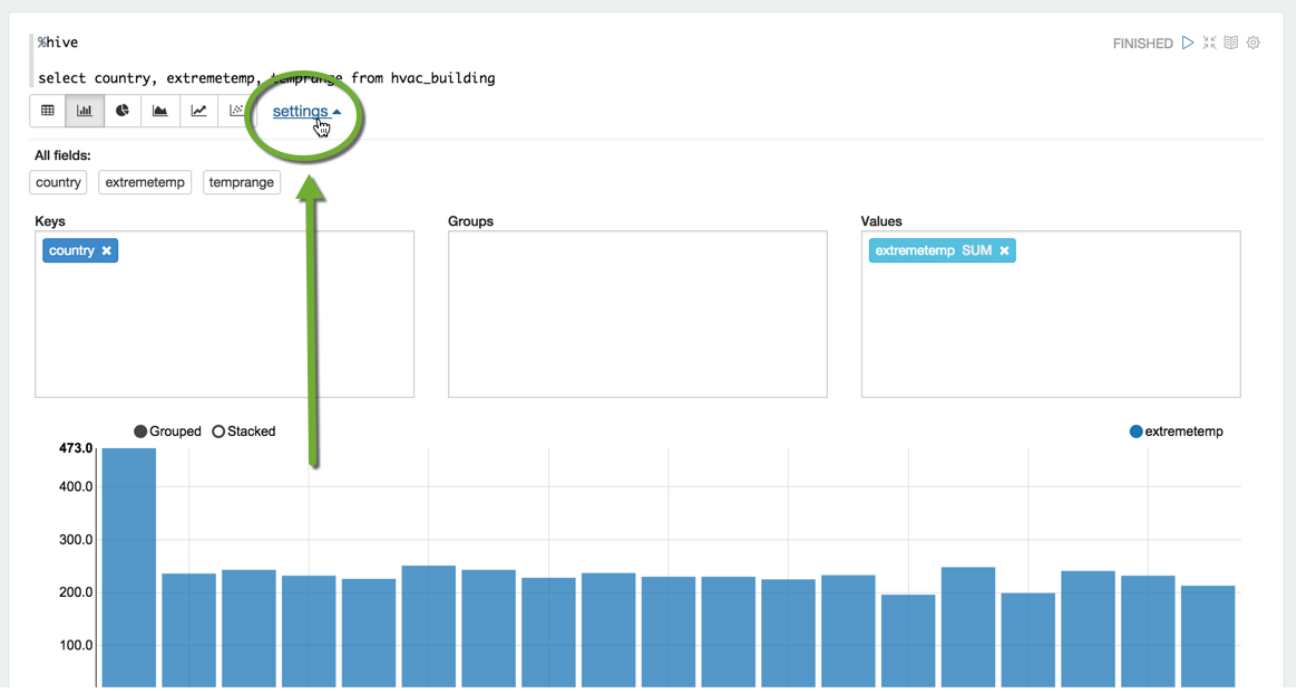
```
%hive
select country, extremetemp, temprange from hvac_building
```



- After running the previous query we can view a chart of the data by clicking the chart button located just under the query.



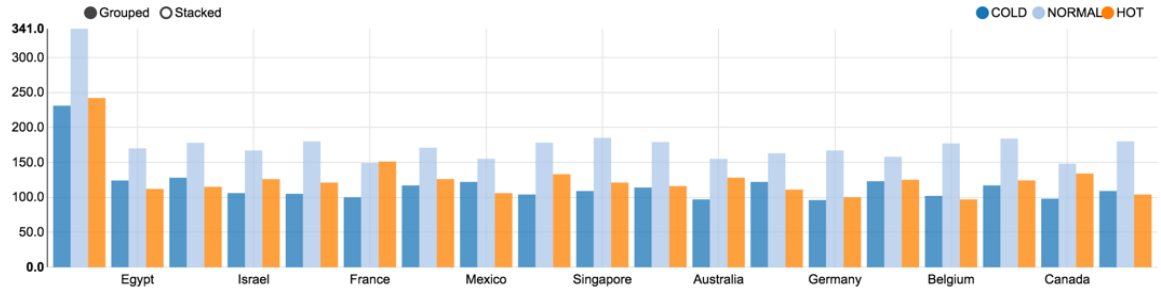
- Click **settings** to open up more advanced settings for creating the chart. Here you can experiment with different values and columns to create different types of charts.



- Arrange the fields according to the following image.
- Drag the field `temprange` into the **groups** box.
- Click **SUM** on `extremetemp` and change it to **COUNT**.
- Make sure that `country` is the only field under **Keys**.

Awesome! You've just created your first chart using Apache Zeppelin.

- From this chart we can see which countries have the most extreme temperature and how many **NORMAL** events there are compared to **HOT** and **COLD**.
- It could be possible to figure out which buildings might need HVAC upgrades, and which do not.



- Let's try creating one more note to visualize which types of HVAC systems result in the least amount of `extremetemp` readings.
- Paste the following query into the blank Zeppelin note following the chart we made previously.

```
%hive
select hvacproduct, extremetemp from hvac_building
```

- Now use **Shift+Enter** to run the note.

- Arrange the fields according to the following image so we can recreate the chart below.
- Make sure that `hvacproduct` is in the **Keys** box.
- Make sure that `extremetemp` is in the **Values** box and that it is set to **COUNT**.

- Now we can see which HVAC units result in the most `extremetemp` readings. Thus we can make a more informed decision when purchasing new HVAC systems.

  Apache Zeppelin gives you the power to connect right to your Hadoop cluster to quickly obtain results from the data inside of Hadoop without having to export data to any other sources.

  It's also important to note that Zeppelin contains many, many interpreters that can be utilized to obtain data in a variety of ways.

  One of the default interpreters included with Zeppelin is for Apache Spark. With the popularity of Apache Spark rising, you can simply write Spark scripts to execute directly on Apache Zeppelin to obtain results from your data in a matter of seconds.