

PI World 2019 Lab

Essential Data Analysis Skills for the PI
System User



OSIsoft, LLC
1600 Alvarado Street
San Leandro, CA 94577 USA
Tel: (01) 510-297-5800
Web: <http://www.osisoft.com>

© 2019 by OSIsoft, LLC. All rights reserved.

OSIsoft, the OSIsoft logo and logotype, Analytics, PI ProcessBook, PI DataLink, ProcessPoint, Asset Framework (AF), IT Monitor, MCN Health Monitor, PI System, PI ActiveView, PI ACE, PI AlarmView, PI BatchView, PI Vision, PI Data Services, Event Frames, PI Manual Logger, PI ProfileView, PI WebParts, ProTRAQ, RLINK, RtAnalytics, RtBaseline, RtPortal, RtPM, RtReports and RtWebParts are all trademarks of OSIsoft, LLC. All other trademarks or trade names used herein are the property of their respective owners.

U.S. GOVERNMENT RIGHTS

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the OSIsoft, LLC license agreement and as provided in DFARS 227.7202, DFARS 252.227-7013, FAR 12.212, FAR 52.227, as applicable. OSIsoft, LLC.

Published: March 22, 2019

Table of Contents

Contents

Table of Contents.....	3
PI System Software Components.....	5
Virtual Learning Environment.....	5
Introduction	6
Background	6
Intro to PI AF hierarchy and Data Set	7
PI Integrator for BA View Configuration	9
Intro to Power BI.....	12
Importing a Data Set	15
The Starter File.....	18
Exercise 1 – Power Generation	19
Configuring the Hierarchy.....	20
Weekly Average Power - Matrix	21
Daily Total Active Power – Stacked Column Chart.....	30
Rated Power – Multi-Row Card	33
Active Power by Hour – Line Chart	34
Wind Speed by Hour – Line Chart.....	36
Wind Turbine Location – Map	38
Takeaways.....	39
Exercise 2 – Power Generation Analysis	40
Power Curve – Scatter Chart.....	41
Correlation – R Corr. Plot.....	45
Active Power by Air Temperature – Line Chart	47
Active Power by Wind Turbulence – Line Chart	50
Nacelle Direction – Pie Chart	53
Takeaways.....	56
Exercise 3 – Curtailment Report	57
Importing the Data.....	58
Production Loss by Turbine – Clustered Column Chart	59
Curtailment Statistics and PI Vision Links – Matrix.....	61

Wind Curtailment vs Wind Speed and Hour of Day – Line Chart	71
Operational Efficiencies – Violin Plot.....	73
Takeaways.....	76
Appendix – Python Scripting in Power BI.....	77
Save the Date!.....	86

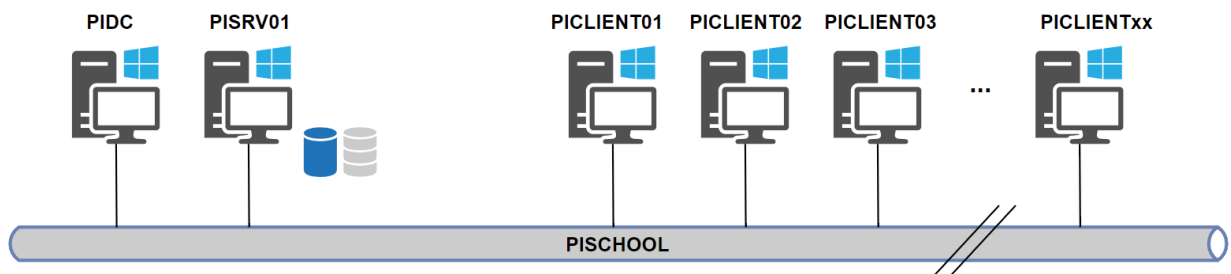
PI System Software Components

The Virtual Machines used for this lab have the following PI System software components installed:

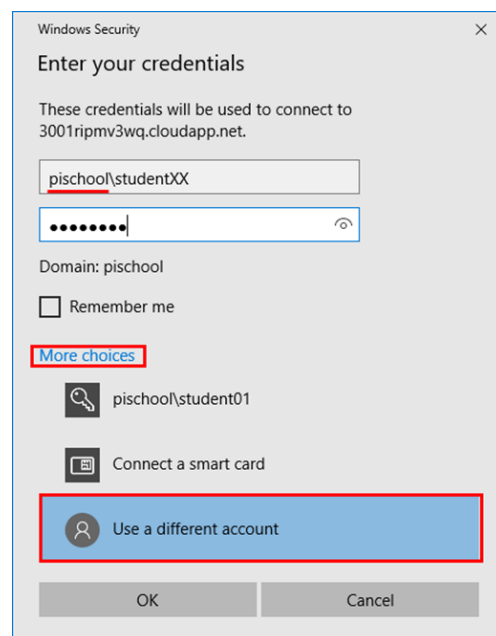
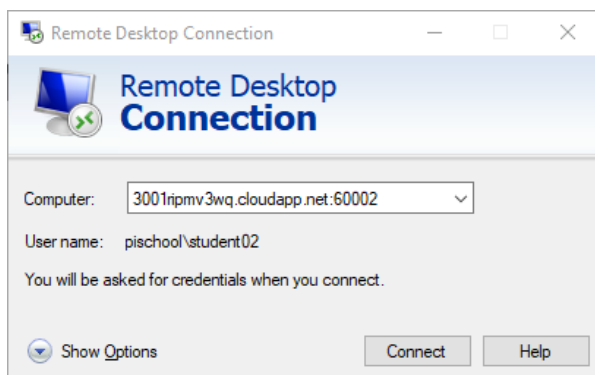
Software	Version
PI Data Archive	2018
PI Asset Framework (PI AF) Services	2018
PI System Explorer (PI AF Client)	2018 SP1
PI Vision	2017 R2 SP1
PI Integrator for Business Analytics	2018
Microsoft Power BI	2.66.5376.1681

Virtual Learning Environment

The machines being used are all in the PISCHOOL domain. The architecture is as follows:



Each user has a client machine that will be used to access the data from the PI Data Archive via Microsoft Power BI, PI System Explorer, and PI Vision. We will be using Remote Desktop to access these servers. Please ensure to select “use a different account” when prompted for credentials, and use the account PISCHOOL\student01:



Introduction

Welcome to Essential Data Analysis Skills for the PI User. In this lab, we'll be using Microsoft Power BI to develop rich reports and dashboards using a pre-published data set. There will be no PI Asset Framework (PI AF) or PI Integrator for BA configuration or setup work. We will just be diving straight into Power BI to quickly develop the skills necessary to discover insights and analyze large, complex data sets.

Background

The data set we will be working with comes from a fictitious wind farm. We will be taking a look at one specific wind farm in a fleet of wind farms to analyze key operational factors and gain insights into how the wind farm operates. The source data comes from a real PI System and has been published in a data-science ready format using PI Integrator for BA. We'll configure an array of Power BI visuals and integrate the results with PI Asset Framework and PI Vision.

The wind farm has **10 wind turbines** each with multiple operational factors which need to be analyzed. We will examine data from these 10 wind turbines over 12 days to see how the wind turbines are performing. Large data sets with hundreds of thousands of rows are difficult to work with in MS Excel and this is where Power BI can provide value.

Intro to PI AF hierarchy and Data Set

There is no need to copy what the instructor is doing on screen for this section, feel free to just sit back and watch.

We will take a few minutes to better understand where the data set came from and relate this lab back to the PI System. We are working with a data set for a fictitious wind generation company. They have built a PI AF Hierarchy for their wind farms and turbines in different geographical areas. In this lab, we will focus on analyzing the wind turbines.

Open PI System Explorer and head to the **Wind Power Generation AF database**. Drill down to Black Mesa Wind Farm, this is the wind farm we will be working with and inspect the available attributes for a specific wind turbine (**GE0X**). We will be using a sub-set of these attributes for all our analysis, in addition to leveraging the AF hierarchy. Note that not all attributes will have data as only relevant attributes pertaining to the lab were added.

The screenshot shows the PI System Explorer interface. On the left, the 'Elements' tree is expanded to 'Black Mesa Wind Farm' > 'GE01'. The right pane shows the 'Attributes' tab for 'GE01' with a table of values.

Name	Value
Active Power	1525.8 kW
Apparent Power	1526.6 kVA
Apparent Power (calc)	Calc Failed
BatchWindBinFilter	WTG01_*
Bearing A Temperature	37.833 °C
Bearing B Temperature	-7.5795 °C
Bearing Shaft Temperature	-19.933 °C
Blade 1, Actual Value	1.1306 °
Blade 1, Set Value	0.68706 °
Blade2, Actual Value	1.6743 °
Blade2, Set Value	0.68706 °
Blade3, Actual Value	1.3457 °
Blade3, Set Value	0.68706 °
Blade 1 Error	0.44355 °
Blade 2 Error	0.98721 °
Blade 3 Error	0.65863 °
Blade Total Error	1.2669 °
Capacity	29.08 %
Circuit Breaker Cut-Ins	4202.7 count
Current L1	1585.8 A

Navigate to the Analyses tab of the Wind Turbine to see the pre-configured analyses. These analyses will help us add additional attributes which may provide added value when creating BA dashboards.

GE02

General Child Elements Attributes Ports **Analyses** Notification Rules Version

Name	Backfilling
Power Factor	!
Production Delta	!
Turbine Curtailment	

Name: Power Factor
Description:
Categories:
Analysis Type: Expression

Add a new variable Evaluate

Name	Expression	Output Attribute
PF	'Active Power'/'Apparent Power (calc)'	Power Factor (calc)



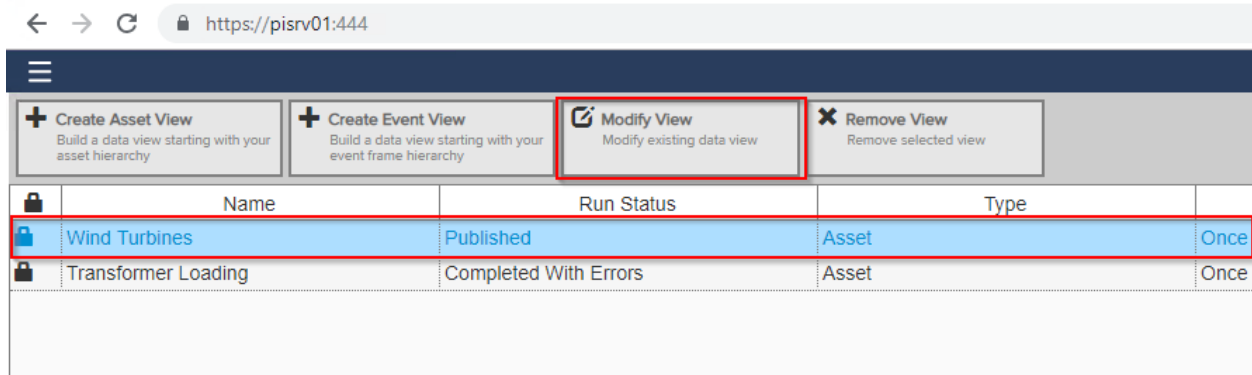
Analyses support a wide variety of functions and are useful for creating measurement which may not exist natively.

Tip

Data from this PI AF hierarchy has been published to a Microsoft SQL Server table using the **PI Integrator for Business Analytics**. This product is designed to make publishing data in a data-science ready format as easy as possible.

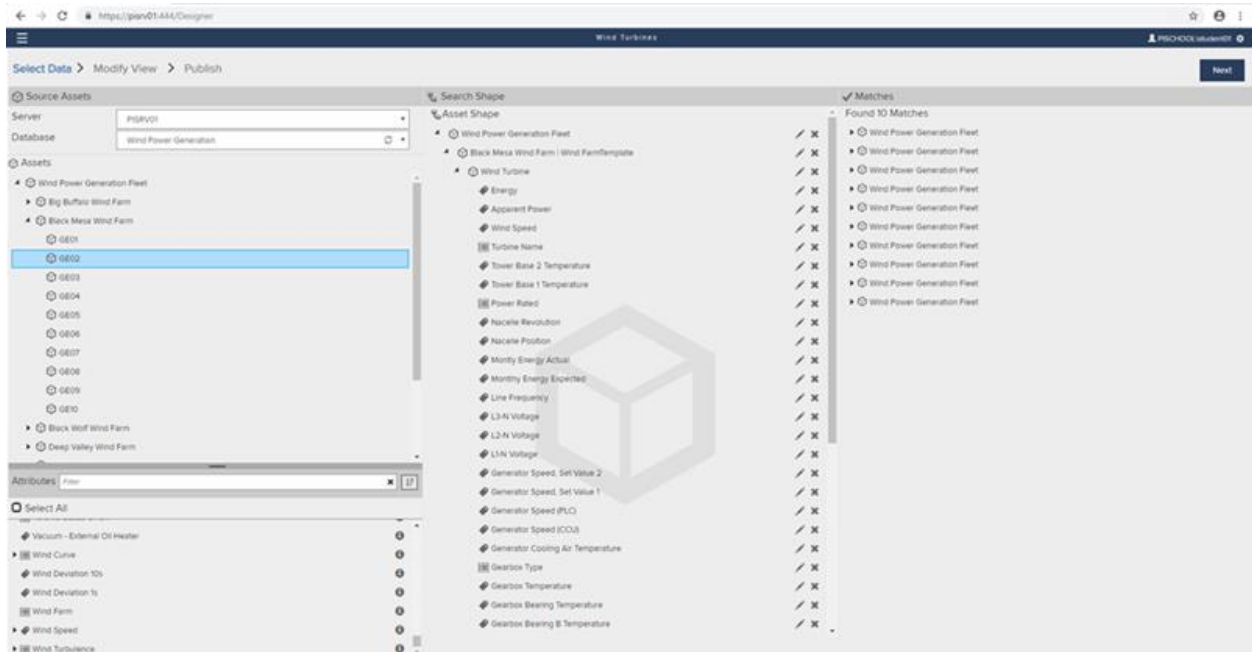
PI Integrator for BA View Configuration

- Step1: Open a web browser and navigate to <https://pisrv01:444/> to access the PI Integrator for BA web UI or open the shortcut on the desktop. Use the same credentials you used to log into the machine when prompted
- Step 2: Select the Wind Turbines View and click Modify View:



- Step 3: Choose **Edit a Copy of this View** and give it a dummy name.

Take a note of the shape configuration. It is leveraging PI AF Templates in order to select a subset of attributes for all wind turbines in the Black Mesa Wind Farm based on the wind turbine template.



- Step4: Leave everything as-is and click next. The next page shows a preview of the publication. Note that we're publishing data every minute for about 12 days:

Wind Power Generation Fleet	TimeStamp	Back Mesa Wind Farm Wind	Wind Turbine	Energy	Apparent Power	Wind Speed	Turbine Name	Tower Base 2 Temperature	Tower Base 1 Temperature
Wind Power Generation Fleet	4/18/2017 6:31:00 AM	Black Mesa Wind Farm	GE04	26,189,520	604,014	6.490	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:32:00 AM	Black Mesa Wind Farm	GE04	26,181,764	498,158	7.465	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:33:00 AM	Black Mesa Wind Farm	GE04	26,171,285	334,333	6.969	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:34:00 AM	Black Mesa Wind Farm	GE04	26,157,490	398,561	7.820	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:35:00 AM	Black Mesa Wind Farm	GE04	26,142,665	456,648	7.595	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:36:00 AM	Black Mesa Wind Farm	GE04	26,127,844	290,825	6.461	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:37:00 AM	Black Mesa Wind Farm	GE04	26,113,020	335,927	6.960	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:38:00 AM	Black Mesa Wind Farm	GE04	26,098,881	390,837	7.459	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:39:00 AM	Black Mesa Wind Farm	GE04	26,085,639	447,841	7.103	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:40:00 AM	Black Mesa Wind Farm	GE04	26,072,396	477,778	8.120	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:41:00 AM	Black Mesa Wind Farm	GE04	26,058,299	296,484	6.269	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:42:00 AM	Black Mesa Wind Farm	GE04	26,043,775	334,458	7.589	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:43:00 AM	Black Mesa Wind Farm	GE04	26,030,467	663,051	8.910	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:44:00 AM	Black Mesa Wind Farm	GE04	26,018,752	518,074	7.680	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:45:00 AM	Black Mesa Wind Farm	GE04	26,005,898	262,363	6.450	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:46:00 AM	Black Mesa Wind Farm	GE04	25,992,479	354,114	7.313	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:47:00 AM	Black Mesa Wind Farm	GE04	25,979,059	387,869	7.358	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:48:00 AM	Black Mesa Wind Farm	GE04	25,970,035	437,887	7.403	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:49:00 AM	Black Mesa Wind Farm	GE04	25,964,949	478,211	8.205	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:50:00 AM	Black Mesa Wind Farm	GE04	25,959,961	482,020	7.309	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:51:00 AM	Black Mesa Wind Farm	GE04	25,949,234	325,115	6.412	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:52:00 AM	Black Mesa Wind Farm	GE04	25,935,672	233,584	5.515	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:53:00 AM	Black Mesa Wind Farm	GE04	25,925,752	334,113	6.828	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:54:00 AM	Black Mesa Wind Farm	GE04	25,916,310	418,494	7.429	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:55:00 AM	Black Mesa Wind Farm	GE04	25,907,268	398,725	6.672	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:56:00 AM	Black Mesa Wind Farm	GE04	25,894,080	398,246	7.337	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:57:00 AM	Black Mesa Wind Farm	GE04	25,889,029	623,718	8.002	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:58:00 AM	Black Mesa Wind Farm	GE04	25,888,954	620,547	7.837	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 6:59:00 AM	Black Mesa Wind Farm	GE04	25,890,092	1,286,199	10.632	WT004	-27.654	-27.654
Wind Power Generation Fleet	4/18/2017 7:00:00 AM	Black Mesa Wind Farm	GE04	25,901,217	1,347,618	10.789	WT004	-27.654	-27.654

- Step 5: Click Next, the publish page is where the publication target is chosen. In this case we are publishing to a Microsoft SQL Server. Alternatively we can publish to a text file via the text option. We have already done this, see the BlackMesaWindFarm.txt which is available in the C:\Class folder. **DON'T CLICK PUBLISH.** It will work but we don't want to add clutter to the SQL Server.

Target Configuration

PISR01

Run Once
 Run on a Schedule

Summary

Shape and Matches

- There are 10 Matching Instances

Timeframe and Interval

- Your Start Time is 4/18/2017 6:31:00 AM
- Your End Time is 4/30/2017 12:00:00 AM
- Your Time Interval gets an interpolated measurement Every 1 minute

Publish

- Step6: At this point close the web browser. It will save automatically.

Introduction

- Step 7: Optionally open SQL Server Management Studio and inspect the **dbo.Wind Turbines** table in the **PI World** database.

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left shows the 'PIWorld' database expanded, with the 'dbo.Wind Turbines' table selected. The main window shows the table's data, which is as follows:

Id	Wind Power Generation Fleet	TimeStamp	Black Mesa Wind Farm Wind FarmTemplate	Wind Turbine	Energy	Apparent Power	Wind Speed	Turbine Name	Tower Base 2 Temperature
1	Wind Power Generation Fleet	2017-04-18 06:31:00.000	Black Mesa Wind Farm	GE01	26271.0390625	320.207580566406	7.1469073295933	WTG01	-27.6543214588046
2	Wind Power Generation Fleet	2017-04-18 06:32:00.000	Black Mesa Wind Farm	GE01	26253.271484375	399.701354980469	7.26167864935303	WTG01	-27.6543214588046
3	Wind Power Generation Fleet	2017-04-18 06:33:00.000	Black Mesa Wind Farm	GE01	26240.16015625	428.250518798828	8.39625072479248	WTG01	-27.6543214588046
4	Wind Power Generation Fleet	2017-04-18 06:34:00.000	Black Mesa Wind Farm	GE01	26231.59375	338.518218994141	7.44572782516479	WTG01	-27.6543214588046
5	Wind Power Generation Fleet	2017-04-18 06:35:00.000	Black Mesa Wind Farm	GE01	26223.029296875	302.177812204888	7.0754806313214	WTG01	-27.6543214588046
6	Wind Power Generation Fleet	2017-04-18 06:36:00.000	Black Mesa Wind Farm	GE01	26219.283203125	530.576989746094	8.34295463562012	WTG01	-27.6543214588046
7	Wind Power Generation Fleet	2017-04-18 06:37:00.000	Black Mesa Wind Farm	GE01	26213.189483125	602.623291015625	8.81623077392578	WTG01	-27.6543214588046
8	Wind Power Generation Fleet	2017-04-18 06:38:00.000	Black Mesa Wind Farm	GE01	26202.890625	543.946105967031	8.34540557861328	WTG01	-27.6543214588046
9	Wind Power Generation Fleet	2017-04-18 06:39:00.000	Black Mesa Wind Farm	GE01	26194.197265625	510.858306884766	8.32504172093506	WTG01	-27.6543214588046
10	Wind Power Generation Fleet	2017-04-18 06:40:00.000	Black Mesa Wind Farm	GE01	26184.84765625	343.413238825391	7.30402757373065	WTG01	-27.6543214588046
11	Wind Power Generation Fleet	2017-04-18 06:41:00.000	Black Mesa Wind Farm	GE01	26170.671875	295.728179931641	7.22276973724365	WTG01	-27.6543214588046
12	Wind Power Generation Fleet	2017-04-18 06:42:00.000	Black Mesa Wind Farm	GE01	26159.517578125	260.54638671875	6.42301797868821	WTG01	-27.6543214588046
13	Wind Power Generation Fleet	2017-04-18 06:43:00.000	Black Mesa Wind Farm	GE01	26149.5546875	289.887780665234	6.59949159622192	WTG01	-27.6543214588046
14	Wind Power Generation Fleet	2017-04-18 06:44:00.000	Black Mesa Wind Farm	GE01	26136.697296875	244.611200694888	6.77398521377963	WTG01	-27.6543214588046
15	Wind Power Generation Fleet	2017-04-18 06:45:00.000	Black Mesa Wind Farm	GE01	26123.841796875	314.1850265375	7.33705711364746	WTG01	-27.6543214588046
16	Wind Power Generation Fleet	2017-04-18 06:46:00.000	Black Mesa Wind Farm	GE01	26110.984375	312.629191894531	7.26267378234863	WTG01	-27.6543214588046
17	Wind Power Generation Fleet	2017-04-18 06:47:00.000	Black Mesa Wind Farm	GE01	26097.56640625	204.146743774414	5.99133205413818	WTG01	-27.6543214588046
18	Wind Power Generation Fleet	2017-04-18 06:48:00.000	Black Mesa Wind Farm	GE01	26079.59375	439.689544677734	8.3330088882568	WTG01	-27.6543214588046

Intro to Power BI

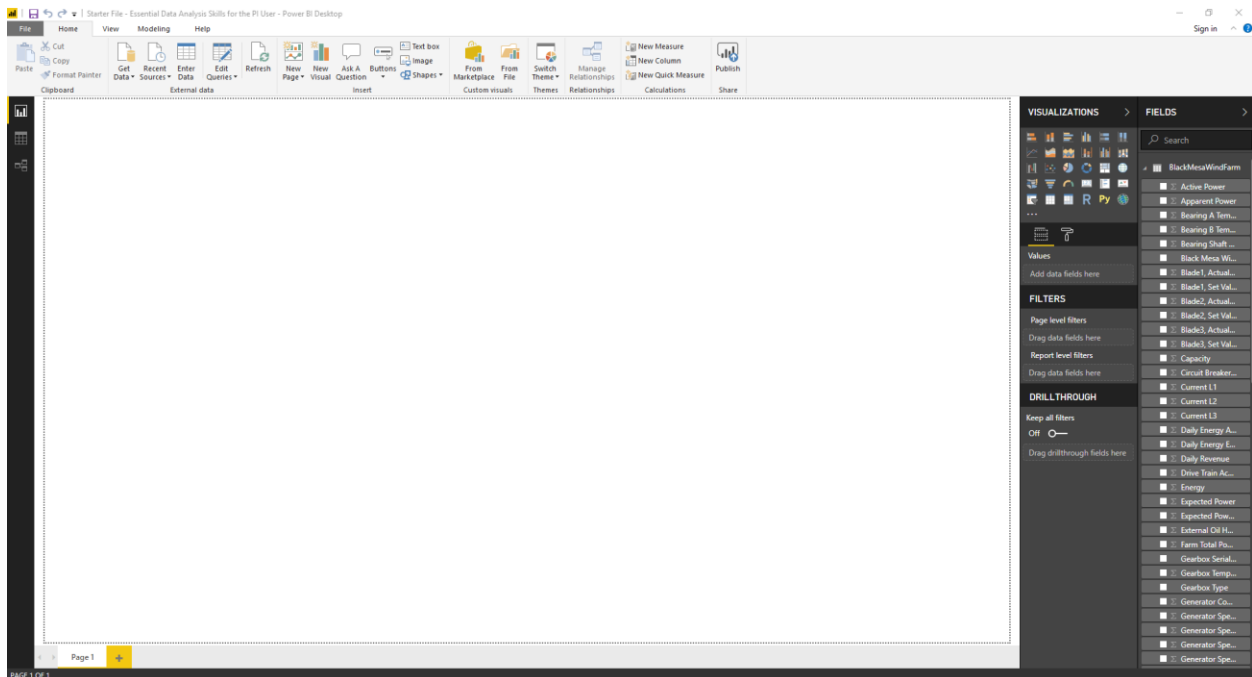
Power BI is a business analytics service and client provided by Microsoft. It provides interactive visualizations with self-service business intelligence capabilities, where end users can create reports and dashboards by themselves, without having to depend on information technology staff or database administrators.

Some of the benefits of Power BI:

- Less work than Excel for more complex analysis and visuals
- Can solve problems that are simply too large for Excel and PI DataLink
- Cheap – [Free download](#) or \$9.99 / month per user for Power BI Pro
- Live reporting and centralized web-based dashboards in Office 365
- Slick visuals including 3rd Party Visuals in [Microsoft AppSource](#)

We will start by getting a feel for Power BI using the starter file.

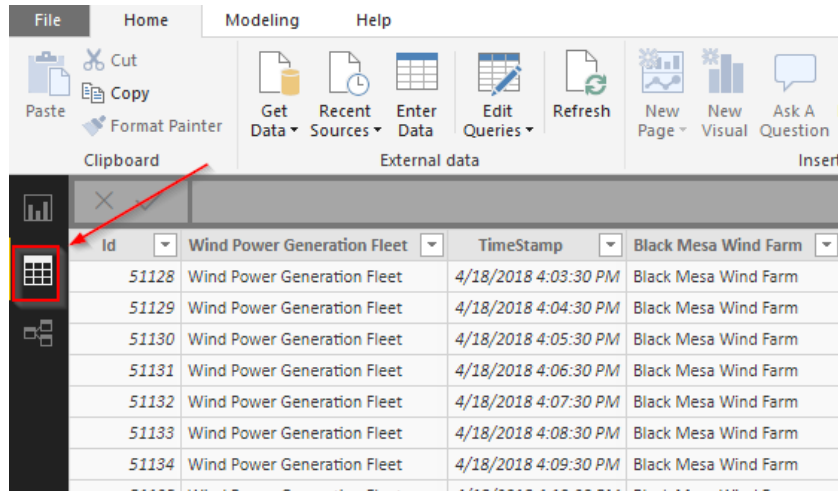
- Step 1: Open **C:\Class\ Starter File - Essential Data Analysis Skills for the PI User.pbix**. You should now see a blank report:



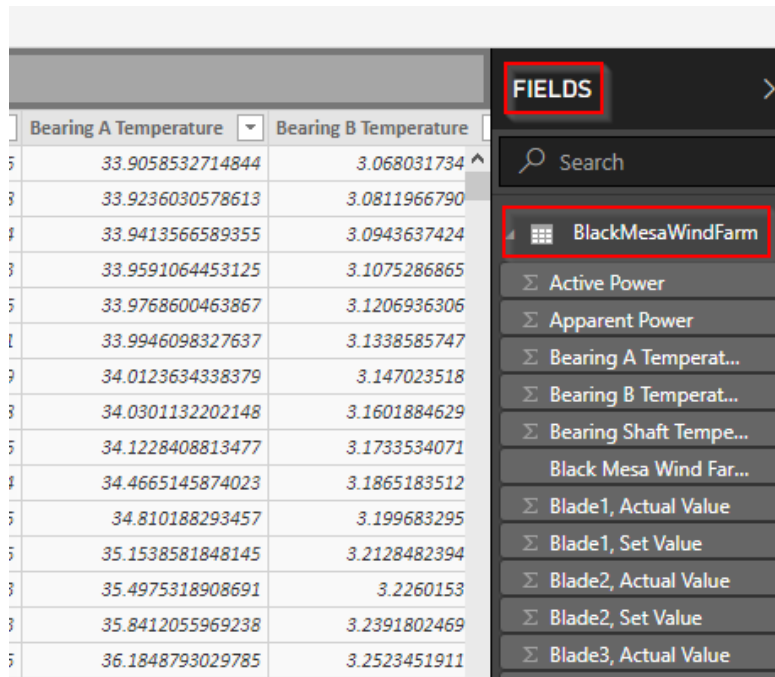
For now we are looking at the **Report View** where the report and visuals are configured. You can switch between the different views on the left-hand side of the report:



- Step 2: Click on the Data Tab to inspect the data set we'll be working with:

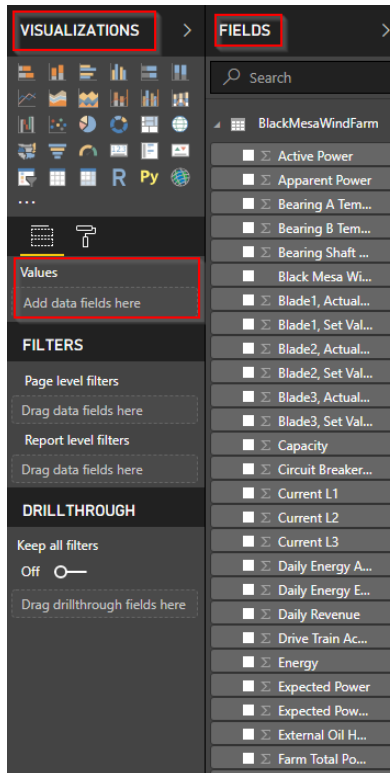


Note that all the columns are available in the **Fields List**, the name of the table is shown above the columns:

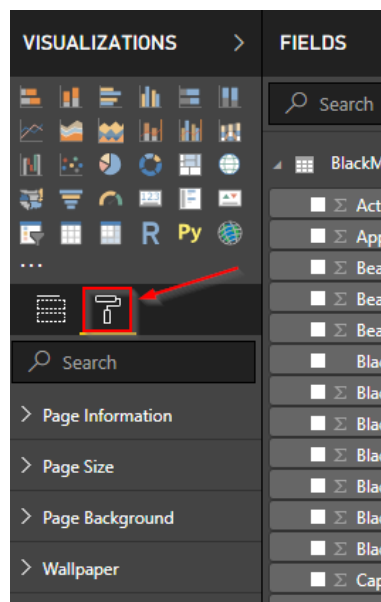


- Step 3: Go back to the Report View and note the **Visualizations Pane** and **Fields Pane**. These sections are where the bulk of the configuration take place.

The visualizations pane contains all of our visuals or symbol which display the data. Columns from the data set are dragged and dropped from the Fields Pane onto the various sections in the Visualizations Pane such as the values field.



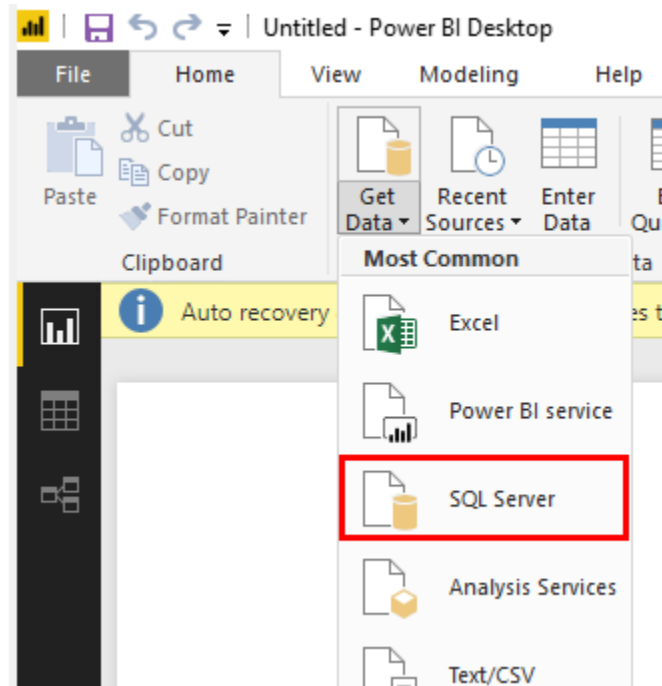
Formatting for our visual is done by selecting the Format tab from the visualizations pane.



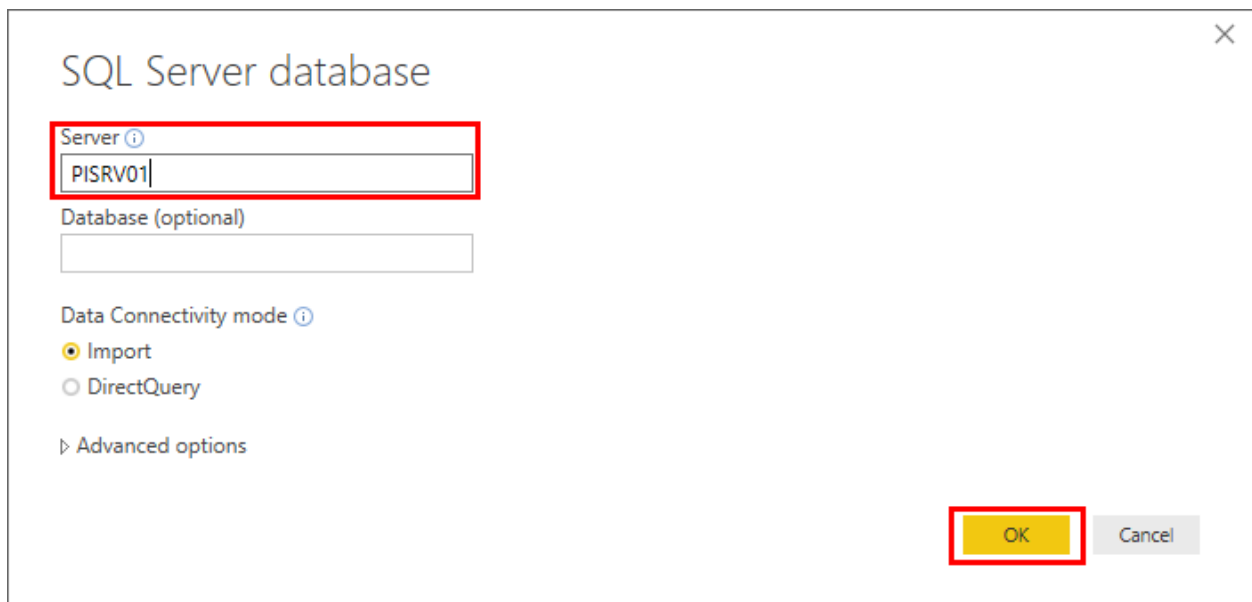
Importing a Data Set

We'll go through the exercise of importing a data set however **we will not complete the final step**, since we are sharing a SQL Server and don't want everyone importing millions of rows at the same time.

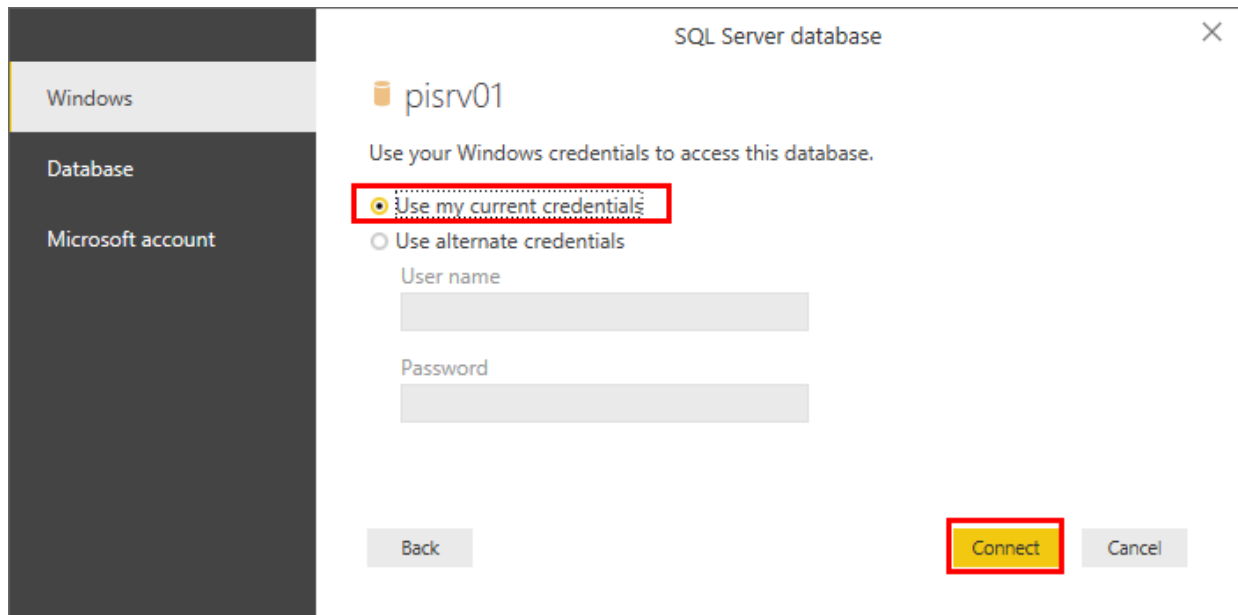
- Step 1: Open a new file.
- Step 2: Select Get Data -> SQL Server Database.



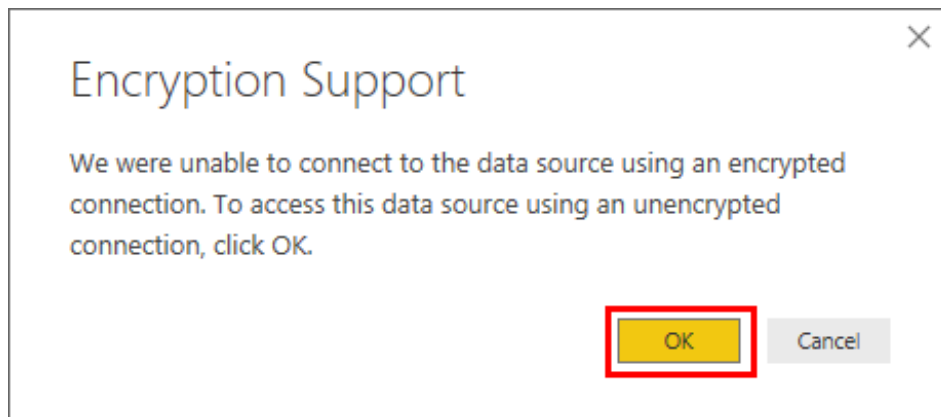
- Step 3: Enter PISRV01 as the server name and click OK.



- Step 4: Leave “Use my current credentials” selected and click Connect:



- Step 5: There will be a warning that the connection is not encrypted, this can be safely ignored, **click OK:**



- Step 6: Select the Transformer Loading Table and briefly inspect the preview. At this point the user would have the option to click Edit and explicitly select columns.

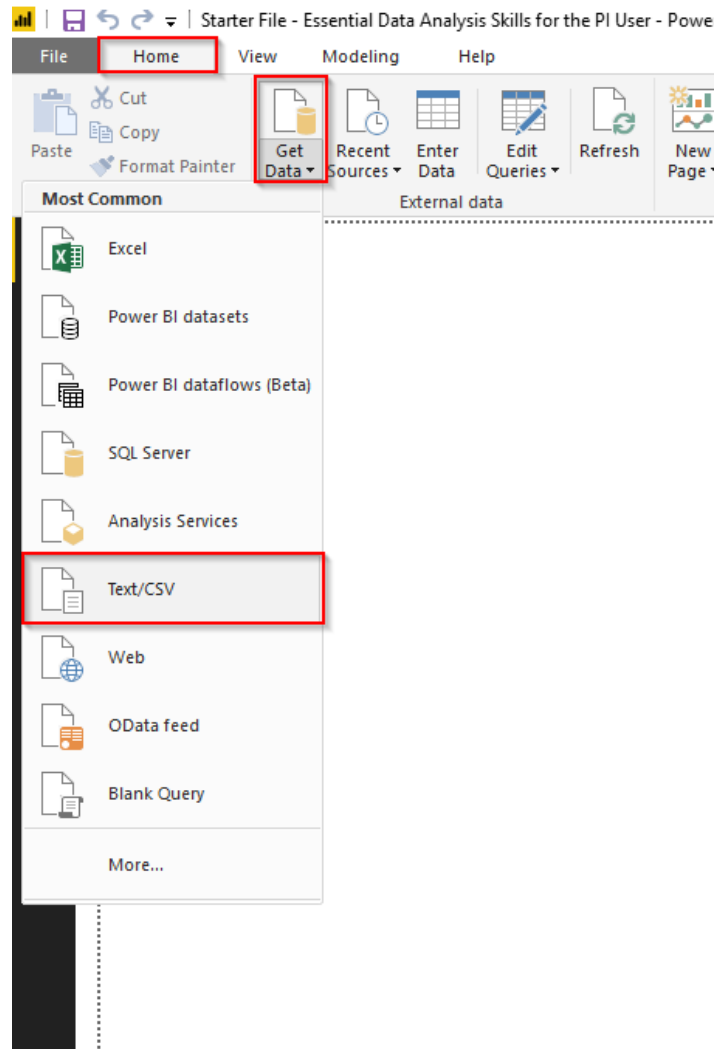
Do NOT click the load button. It will work, but it will take several minutes and we're all sharing a SQL Server. We don't want everyone in the class importing millions of rows at the same time. Instead the following exercises will use a Starter file with the data pre-loaded.

The screenshot shows a software interface with a left-hand 'Navigator' pane and a right-hand preview pane. The Navigator pane shows a tree view of folders: 'pisrv01 [6]', 'PIFD', 'PIIntegratorDB', 'PIVision', 'PIWorld [2]', 'ReportServer', and 'ReportServerTempDB'. Under 'PIWorld [2]', there are two sub-items: 'Transformer Loading' and 'Wind Turbines'. The 'Wind Turbines' item is selected and highlighted with a red box. The preview pane on the right is titled 'Wind Turbines' and displays a table with 10 rows. The columns are 'Id', 'Wind Power Generation Fleet', 'TimeStamp', and 'Black Mesa Wind Farr'. Below the table, there is a message: 'The data in the preview has been truncated due to size limits.' At the bottom of the interface, there are several buttons: 'Select Related Tables', 'DO NOT CLICK', 'Load', 'Edit', and 'Cancel'. The 'DO NOT CLICK' button is highlighted with a red box, and a red arrow points from it to the 'Load' button.

Id	Wind Power Generation Fleet	TimeStamp	Black Mesa Wind Farr
1	Wind Power Generation Fleet	4/18/2017 6:31:00 AM	Black Mesa Wind Farr
2	Wind Power Generation Fleet	4/18/2017 6:32:00 AM	Black Mesa Wind Farr
3	Wind Power Generation Fleet	4/18/2017 6:33:00 AM	Black Mesa Wind Farr
4	Wind Power Generation Fleet	4/18/2017 6:34:00 AM	Black Mesa Wind Farr
5	Wind Power Generation Fleet	4/18/2017 6:35:00 AM	Black Mesa Wind Farr
6	Wind Power Generation Fleet	4/18/2017 6:36:00 AM	Black Mesa Wind Farr
7	Wind Power Generation Fleet	4/18/2017 6:37:00 AM	Black Mesa Wind Farr
8	Wind Power Generation Fleet	4/18/2017 6:38:00 AM	Black Mesa Wind Farr
9	Wind Power Generation Fleet	4/18/2017 6:39:00 AM	Black Mesa Wind Farr
10	Wind Power Generation Fleet	4/18/2017 6:40:00 AM	Black Mesa Wind Farr

Alternatively, you can Load a text file, like the one we have provided in the **C:\Class\BlackMesaWindFarm**. This is the file we will be using for our dataset. It is very similar to the data we have imported into the SQL database but has a few additional attributes.

- Step 7: You can import text files via Get Data > Text/CSV.



The Starter File

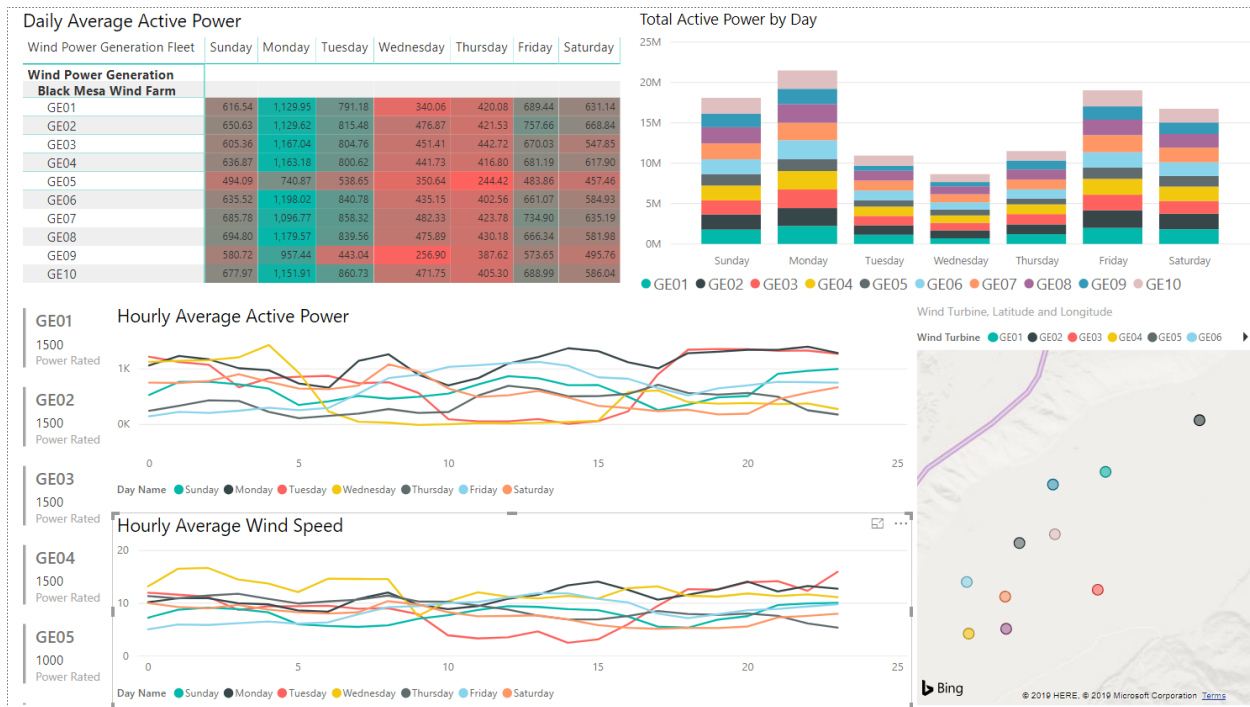
To save time importing the data set, and to make it easy for students to work on the exercises in their own environment, a starter .pbix file has been created with the raw data set (From the BlackMesaWindFarm.txt file) already imported. The file is **C:\Class\Starter File - Essential Data Analysis Skills for the PI User.pbix**. Additionally, all the exercises solution files can be found under the **C:\Class\Solutions** folder as well.

Students may download the free version of Microsoft Power BI and save a local copy of the starter file. **The majority of the exercises will not actually require the use of the VMs.**

Exercise 1 – Power Generation

In this exercise we are going to analyze the power generation of each of our wind turbines in the Black Mesa wind farm. We are going to look at the power generated by each turbine over the course of each day and each hour within the day.

The final display you create for this exercise will look similar to this:

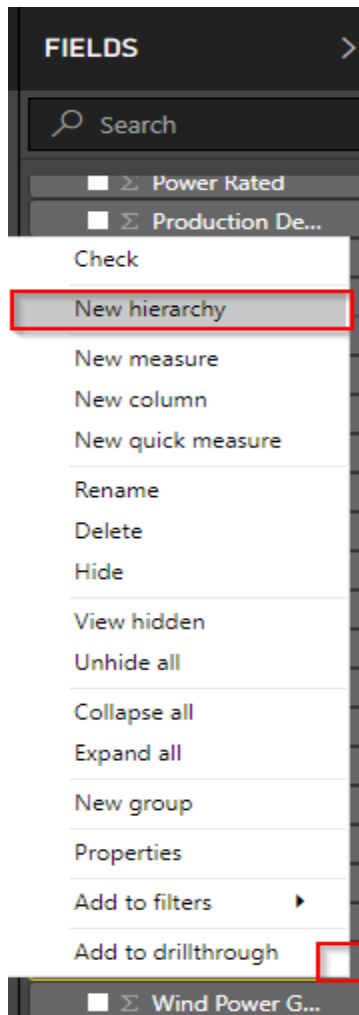


The main attribute we will be looking at is Active Power for each turbine. The resulting report will help us see how power changes as a function of time of day and day of week and establish a correlation with wind speed on a given day. General Steps are:

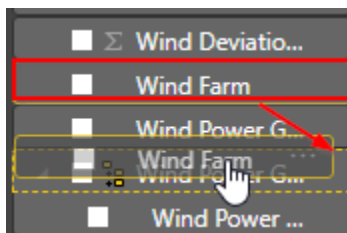
- Configure a **Hierarchy**
- Configure a **Matrix** to show average active power per turbine
- Configure a **Stacked Column Chart** to show the average active power per day of week
- Configure a **Multi-row Card** to show rated power
- Configure a **Line Chart** to show the average active power per hour of day
- Configure a **Line Chart** to show the average wind speed per hour of day
- Configure a **Map** to show wind turbine location

Configuring the Hierarchy

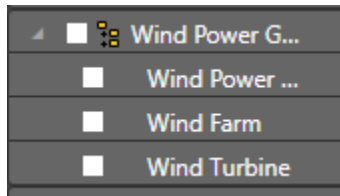
- Step1: We will now create a hierarchy. In the **Fields List**, click the ellipses next to **Wind Power Generation Fleet** and select **New hierarchy**:



- Step 2: Within the fields list, drag and drop the **Wind Farm** field on top of the new **wind power generation fleet hierarchy**:



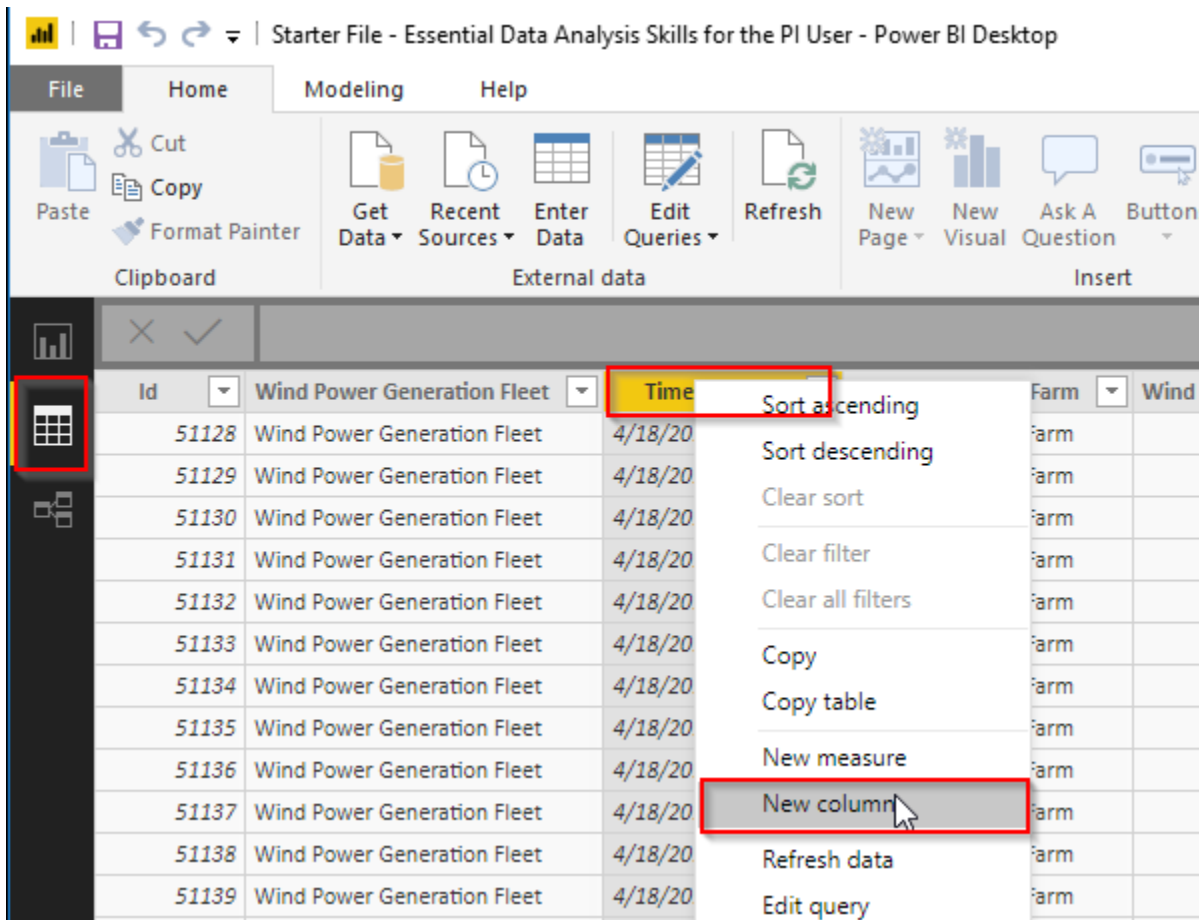
- Step 3: Repeat for **Wind Turbine** so that your hierarchy looks like this.




Weekly Average Power - Matrix

We're going to display the daily average power on a per wind turbine basis using the hierarchy we just created. First, we need to create a new column for our data set so that we can group our timestamps by days of the week.

- Step1: Go to the **data view**, right click on any **column header**, and select **New column**:



Next, we will use a **DAX formula** to format our timestamp column so that only days of the week are show.



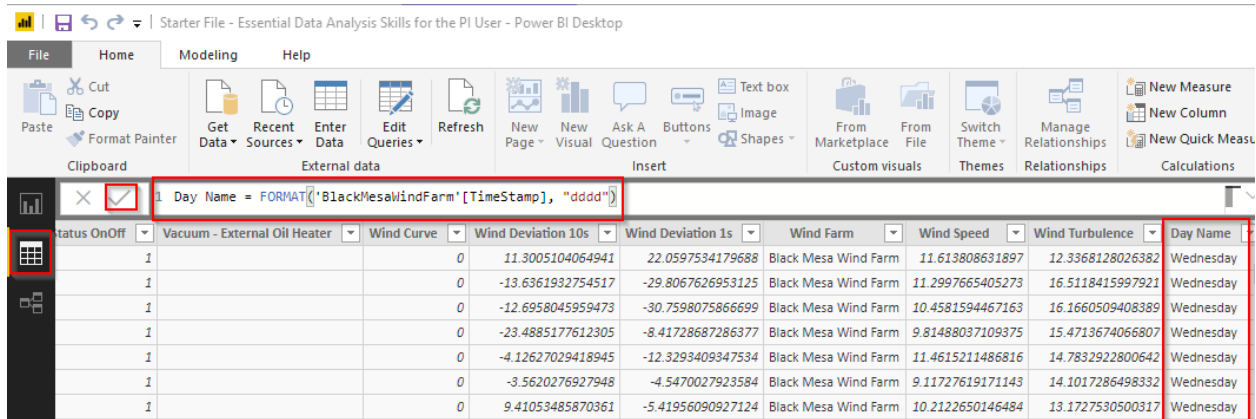
Data Analysis Expressions (DAX) can be used for basic calculations and data analysis in Power BI. More information can be found [Here](#).

Tip

- Step 2: Copy and paste the DAX formula below into your new column:

Day Name = FORMAT('BlackMesaWindFarm'[TimeStamp], "dddd")

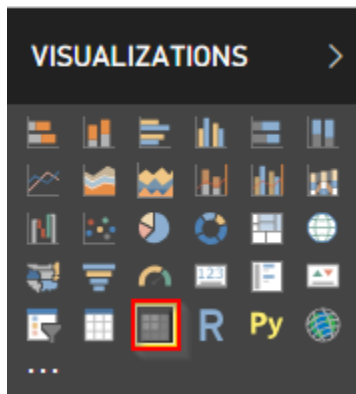
This new column will be called **Day Name** and will reference the timestamp column of our **BlackMesaWindFarm** data set and show the weekday name ("dddd" format):



The screenshot shows the Power BI Desktop interface. The formula bar at the top contains the DAX formula: `Day Name = FORMAT('BlackMesaWindFarm'[TimeStamp], "dddd")`. Below the formula bar, a data table is displayed with the following columns: status OnOff, Vacuum - External Oil Heater, Wind Curve, Wind Deviation 10s, Wind Deviation 1s, Wind Farm, Wind Speed, Wind Turbulence, and Day Name. The 'Day Name' column contains the values 'Wednesday' for all rows.

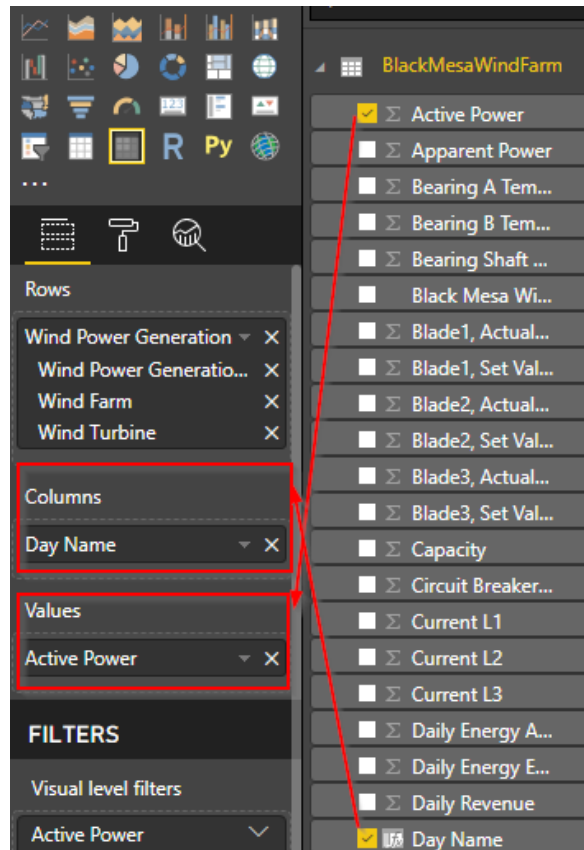
status OnOff	Vacuum - External Oil Heater	Wind Curve	Wind Deviation 10s	Wind Deviation 1s	Wind Farm	Wind Speed	Wind Turbulence	Day Name
1			11.3005104064941	22.0597534179688	Black Mesa Wind Farm	11.613808631897	12.3368128026382	Wednesday
1			-13.6361982754517	-29.8067626953125	Black Mesa Wind Farm	11.2997665405273	16.5118415997921	Wednesday
1			-12.6958045959473	-30.7598075866699	Black Mesa Wind Farm	10.4581594467163	16.1660509408389	Wednesday
1			-23.4885177612305	-8.41728687286377	Black Mesa Wind Farm	9.81488037109375	15.4713674066807	Wednesday
1			-4.12627029418945	-12.3293409347534	Black Mesa Wind Farm	11.4615211486816	14.7832922800642	Wednesday
1			-3.5620276927948	-4.5470027923584	Black Mesa Wind Farm	9.11727619171143	14.1017286498332	Wednesday
1			9.41053485870361	-5.41956090927124	Black Mesa Wind Farm	10.2122650146484	13.1727530500317	Wednesday

- Step 3: Now, navigate back to the report view (📊) and add a **Matrix** to the canvas by clicking the Matrix icon in the Visualization Pane:

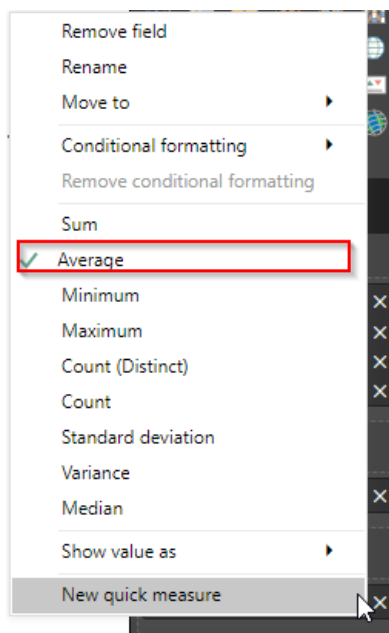



The screenshot shows the 'VISUALIZATIONS' pane in Power BI. The Matrix icon, which is a grid of cells, is highlighted with a red box. Other icons for various chart types like bar, line, and pie charts are also visible.

- Step 4: Drag and drop the **Wind Power Gen Hierarchy** for the Rows, **Day Name** for the columns, and **Active Power** for the Values:



- Step 5: Change the value field to summarize by **Average Active Power**:



- Step 6: Now we will drill down into the hierarchy. Press the **drill down button**  located on the bottom. Once this button is highlighted, any time you click on a name within this symbol, it will drill down to expand the hierarchy instead of slicing and dicing the data. Now click on **Wind Power Generation Fleet** to drill down:

Wind Power Generation Fleet	Friday	Monday	Saturday	Sunday	Thursday	Tuesday
Wind Power Generation Fleet	660.71	1,091.42	580.71	627.83	399.50	759.31
Black Mesa Wind Farm	660.71	1,091.42	580.71	627.83	399.50	759.31
Total	660.71	1,091.42	580.71	627.83	399.50	759.31

- Step 7: Expand all the way to the bottom until the Matrix looks like this:

Wind Power Generation Fleet	Friday	Monday	Saturday	Sunday	Thursday	Tuesday	Wednesday	Total
Wind Power Generation Fleet	660.71	1,091.42	580.71	627.83	399.50	759.31	418.27	626.10
Black Mesa Wind Farm	660.71	1,091.42	580.71	627.83	399.50	759.31	418.27	626.10
GE01	689.44	1,129.95	631.14	616.54	420.08	791.18	340.06	638.73
GE02	757.66	1,129.62	668.84	650.63	421.53	815.48	476.87	681.15
GE03	670.03	1,167.04	547.85	605.36	442.72	804.76	451.41	642.25
GE04	681.19	1,163.18	617.90	636.87	416.80	800.62	441.73	654.98
GE05	483.86	740.87	457.46	494.09	244.42	538.65	350.64	458.76
GE06	661.07	1,198.02	584.93	635.52	402.56	840.78	435.15	650.27
GE07	734.90	1,096.77	635.19	685.78	423.78	858.32	482.33	678.63
GE08	666.34	1,179.57	581.98	694.80	430.18	839.56	475.89	667.81
GE09	573.65	957.44	495.76	580.72	387.62	443.04	256.90	525.01
GE10	688.99	1,151.91	586.04	677.97	405.30	860.73	471.75	663.57
Total	660.71	1,091.42	580.71	627.83	399.50	759.31	418.27	626.10

Exercise 1 – Power Generation

- Step 8: You can navigate back up the layers by clicking the Up Arrow on the bottom right of the visual:

Wind Power Generation Fleet	Friday	Monday	Saturday	Sunday	Thursday	Tuesday	Wednesday	Total
Wind Power Generation Fleet	660.71	1,091.42	580.71	627.83	399.50	759.31	418.27	626.10
Black Mesa Wind Farm	660.71	1,091.42	580.71	627.83	399.50	759.31	418.27	626.10
GE01	689.44	1,129.95	631.14	616.54	420.08	791.18	340.06	638.73
GE02	757.66	1,129.62	668.84	650.63	421.53	815.48	476.87	681.15
GE03	670.03	1,167.04	547.85	605.36	442.72	804.76	451.41	642.25
GE04	681.19	1,163.18	617.90	636.87	416.80	800.62	441.73	654.98
GE05	483.86	740.87	457.46	494.09	244.42	538.65	350.64	458.76
GE06	661.07	1,198.02	584.93	635.52	402.56	840.78	435.15	650.27
GE07	734.90	1,096.77	635.19	685.78	423.78	858.32	482.33	678.63
GE08	666.34	1,179.57	581.98	694.80	430.18	839.56	475.89	667.81
GE09	573.65	957.44	495.76	580.72	387.62	443.04	256.90	525.01
GE10	688.99	1,151.91	586.04	677.97	405.30	860.73	471.75	663.57
Total	660.71	1,091.42	580.71	627.83	399.50	759.31	418.27	626.10

Drill Up

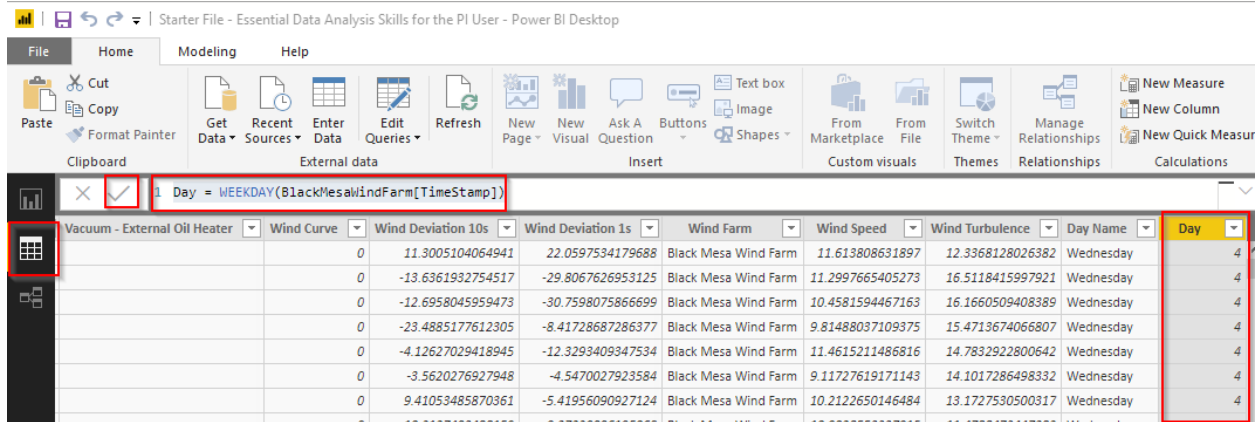
- Step 9: Drill back down to the lowest level and turn off Drill Down mode by clicking on the drill down arrow to deselect it. This will allow us to filter the rest of the report by clicking on Active power and wind turbines rather than drilling.

Wind Power Generation Fleet	Friday	Monday	Saturday	Sunday	Thursday	Tuesday	Wednesday	Total
Wind Power Generation Fleet	660.71	1,091.42	580.71	627.83	399.50	759.31	418.27	626.10
Black Mesa Wind Farm	660.71	1,091.42	580.71	627.83	399.50	759.31	418.27	626.10
GE01	689.44	1,129.95	631.14	616.54	420.08	791.18	340.06	638.73
GE02	757.66	1,129.62	668.84	650.63	421.53	815.48	476.87	681.15
GE03	670.03	1,167.04	547.85	605.36	442.72	804.76	451.41	642.25
GE04	681.19	1,163.18	617.90	636.87	416.80	800.62	441.73	654.98
GE05	483.86	740.87	457.46	494.09	244.42	538.65	350.64	458.76
GE06	661.07	1,198.02	584.93	635.52	402.56	840.78	435.15	650.27
GE07	734.90	1,096.77	635.19	685.78	423.78	858.32	482.33	678.63
GE08	666.34	1,179.57	581.98	694.80	430.18	839.56	475.89	667.81
GE09	573.65	957.44	495.76	580.72	387.62	443.04	256.90	525.01
GE10	688.99	1,151.91	586.04	677.97	405.30	860.73	471.75	663.57
Total	660.71	1,091.42	580.71	627.83	399.50	759.31	418.27	626.10

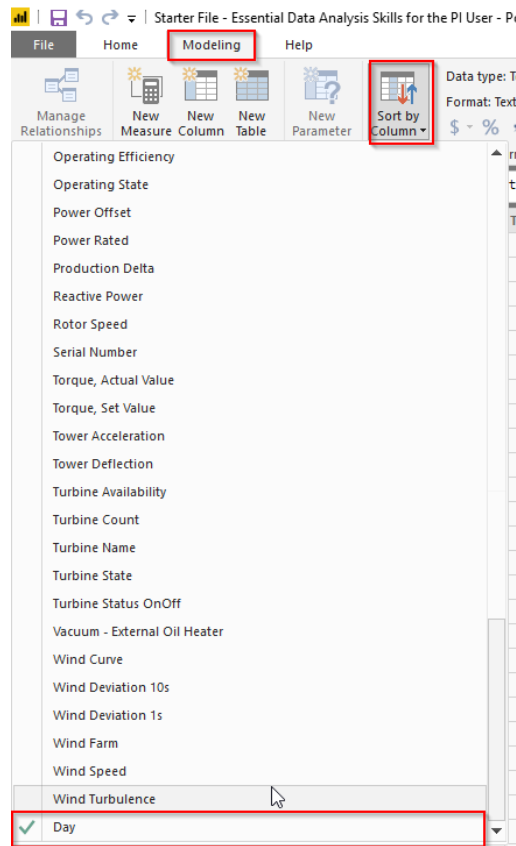
Notice that the days of the week are not ordered. We will create a new column called day to represent the timestamps of each day numerically and order our day name column by this new column.

- Step 10: Navigate back to the **Data View** , right click the column header of any column and select **New Column**, and add the following DAX formula to create a new column called **Day**:

Day = WEEKDAY(BlackMesaWindFarm[TimeStamp])



- Step 11: While still in the **Data View** tab, select **Day Name**, open the **Modeling** Ribbon, and **Sort by Column** -> **Day**:

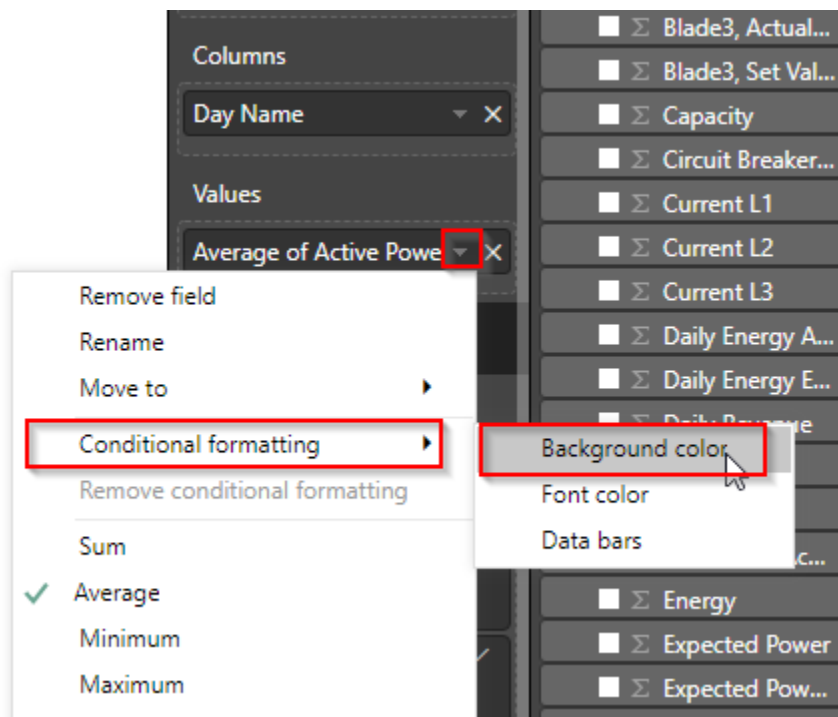


- Step 12: Go back to the **Report View**, the week day headers should now display in chronological order.

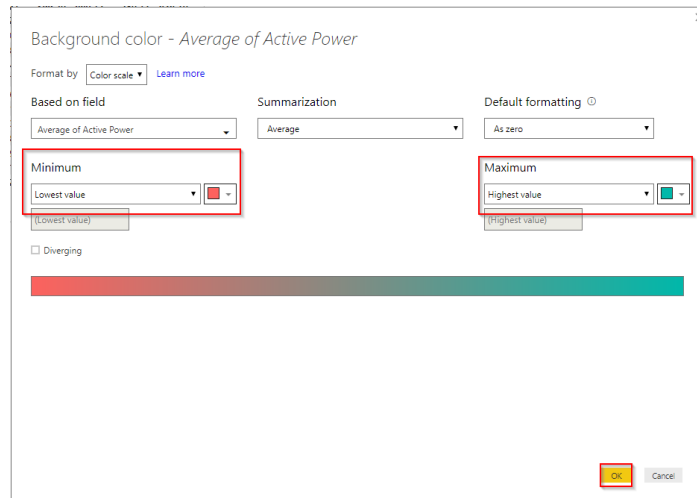
Wind Power Generation Fleet	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Total
Wind Power Generation Fleet	627.83	1,091.42	759.31	418.27	399.50	660.71	580.71	626.10
Black Mesa Wind Farm	627.83	1,091.42	759.31	418.27	399.50	660.71	580.71	626.10
GE01	616.54	1,129.95	791.18	340.06	420.08	689.44	631.14	638.73
GE02	650.63	1,129.62	815.48	476.87	421.53	757.66	668.84	681.15
GE03	605.36	1,167.04	804.76	451.41	442.72	670.03	547.85	642.25
GE04	636.87	1,163.18	800.62	441.73	416.80	681.19	617.90	654.98
GE05	494.09	740.87	538.65	350.64	244.42	483.86	457.46	458.76
GE06	635.52	1,198.02	840.78	435.15	402.56	661.07	584.93	650.27
GE07	685.78	1,096.77	858.32	482.33	423.78	734.90	635.19	678.63
GE08	694.80	1,179.57	839.56	475.89	430.18	666.34	581.98	667.81
GE09	580.72	957.44	443.04	256.90	387.62	573.65	495.76	525.01
GE10	677.97	1,151.91	860.73	471.75	405.30	688.99	586.04	663.57
Total	627.83	1,091.42	759.31	418.27	399.50	660.71	580.71	626.10

Next, we will add conditional formatting to highlight wind turbines with high average active power generation. Conditional formatting is set from the Values field in the Visualizations Pane.

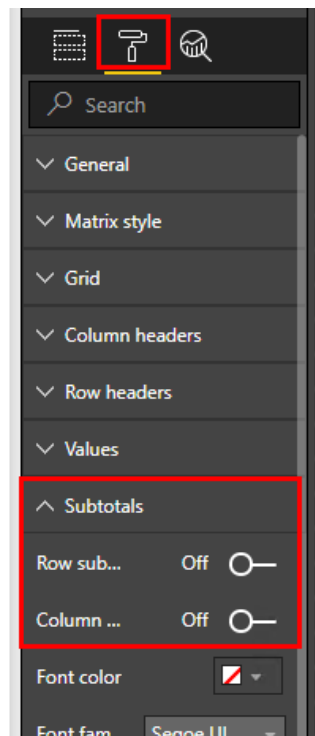
- Step 13: Select the drop down by **Average of Active Power** and click **Conditional Formatting -> Background Color Scales**:



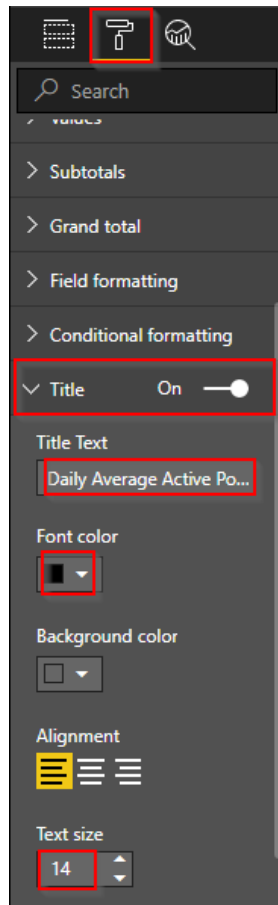
- Step 14: You can change the maximum and minimum colors to your liking. For now, let's make lower values red and higher values green and Click OK.



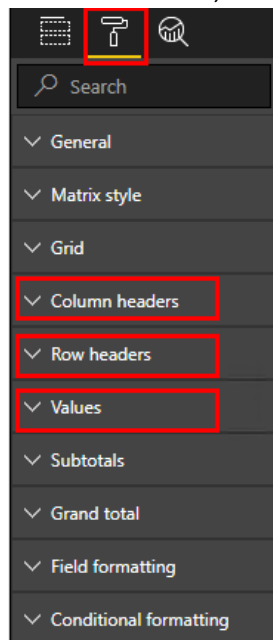
- Step 15: Turn off the **Subtotals** using the **formatting** options:



- Step 16: Add a **Title** to the **Matrix** using the Formatting Options. Call it **Daily Average Active Power**, change the color to black, and increase the font size:



- Step 17: Increase the **font size** of the **Column headers**, **Row headers**, and **Values**:



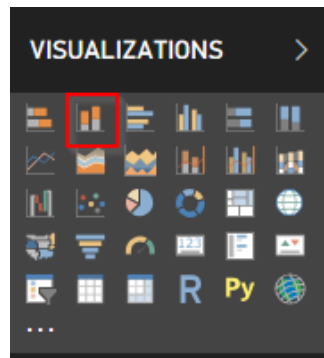
The matrix should look something like this:

Daily Average Active Power		Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Wind Power Generation Fleet								
Black Mesa Wind Farm								
GE01		616.54	1,129.95	791.18	340.06	420.08	689.44	631.14
GE02		650.63	1,129.62	815.48	476.87	421.53	757.66	668.84
GE03		605.36	1,167.04	804.76	451.41	442.72	670.03	547.85
GE04		636.87	1,163.18	800.62	441.73	416.80	681.19	617.90
GE05		494.09	740.87	538.65	350.64	244.42	483.86	457.46
GE06		635.52	1,198.02	840.78	435.15	402.56	661.07	584.93
GE07		685.78	1,096.77	858.32	482.33	423.78	734.90	635.19
GE08		694.80	1,179.57	839.56	475.89	430.18	666.34	581.98
GE09		580.72	957.44	443.04	256.90	387.62	573.65	495.76
GE10		677.97	1,151.91	860.73	471.75	405.30	688.99	586.04

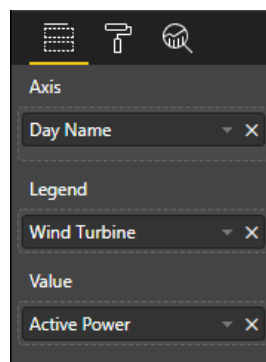
Daily Total Active Power – Stacked Column Chart

Next, we'll configure a new visual to show total active power daily for all the wind turbines in our farm for the 12 days of data we have.

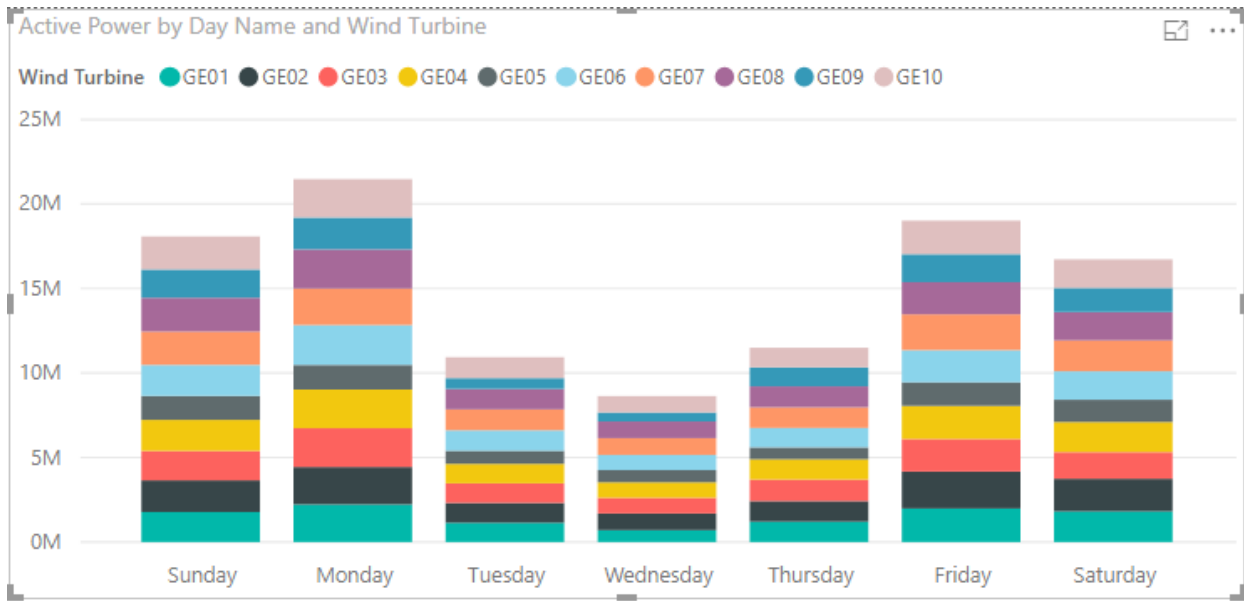
- Step 1: Click the Stacked Column Chart icon in the visualizations pane.



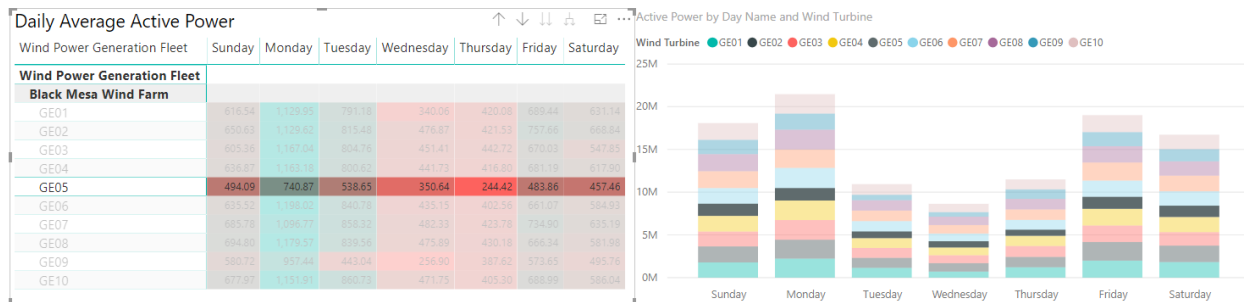
- Step 2: Use **Day Name** for the **Axis**, **Wind Turbine** for the **Legend**, and **Active Power** as the **Value**:



The result visual should look something like this:



- Step 3: Select the wind turbine **GE05** from our **Matrix** and see how the corresponding **clustered column chart** changes:

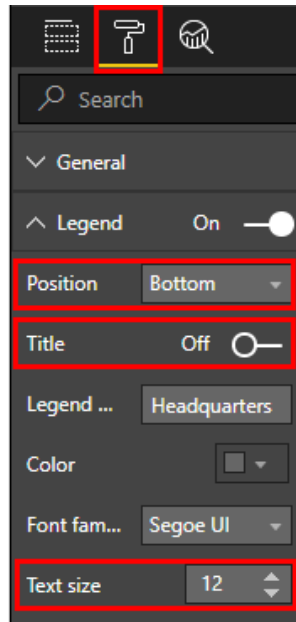


We are **slicing and dicing** the data here so that only the selected parameters are highlighted in all of our visuals.

Note that **Wednesday** seems to produce the least total as well as average active power for all of our turbines. We will look at factors that affect power generation in the next report, but we can extrapolate that since these wind turbines belong to the same farm, they should follow the same trend for active power generation.

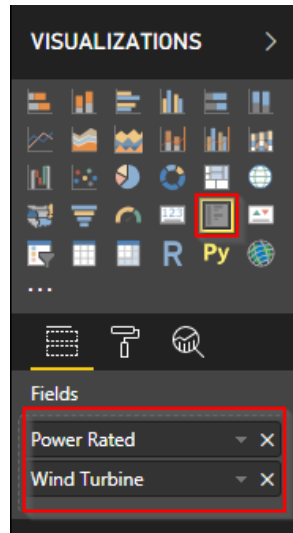
Also note that for **wind turbine GE05** the average active power delivered is significantly lower than the other wind turbines.

- Step 4: **Left Click GE05** again to deselect it and turn off the filtering.
- Step 5: Select the **stacked Column Chart** visual, and adjust the **formatting**:
 - Move the legend to the bottom, remove the legend title, and increase the text size
 - Change the Chart Title to “Total Active Power by Day” with black text, increase the text size

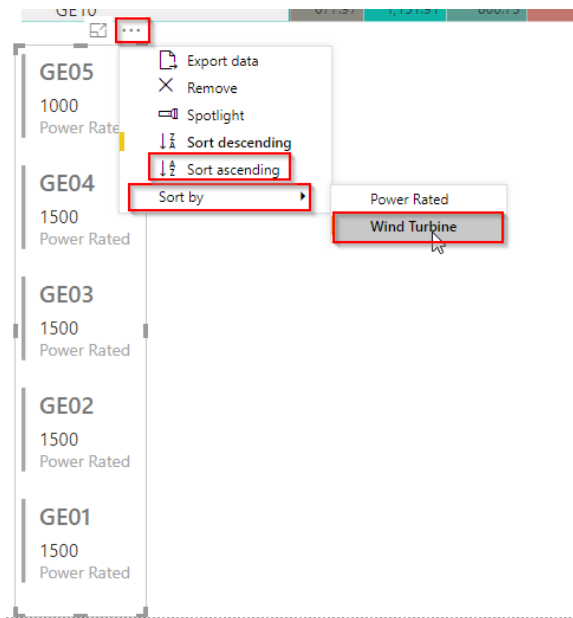


Rated Power – Multi-Row Card

- Step 1: Select the **Multi-row card** symbol from the Visualizations pane and add **Power Rated** and **Wind Turbine** to the Fields section. Make sure to configure **Power Rated** to **Don't Summarize**:



- Step 2: Sort the Card by **wind turbine** and so it is **ascending**:



Discussion: Note the rated power of each of the wind turbines. Does it make sense now that GE05 has the lowest average and total daily active power generation?

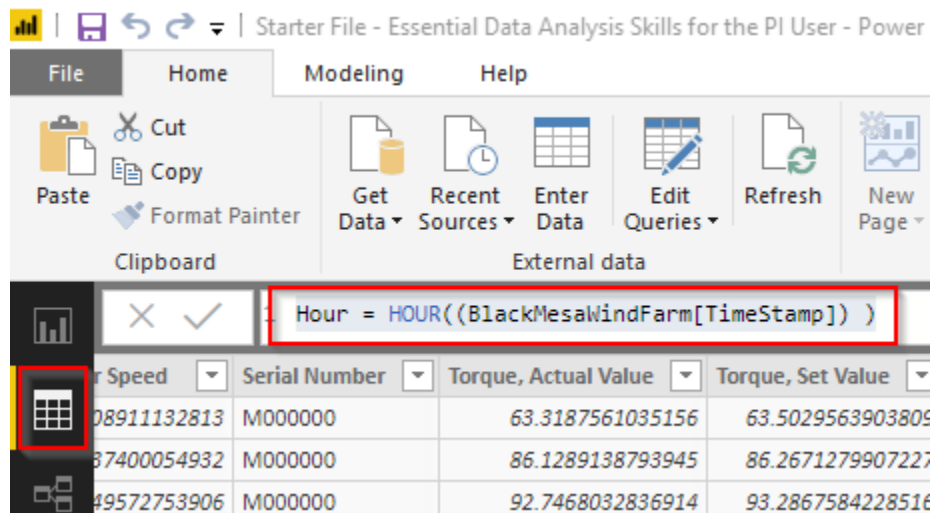
Active Power by Hour – Line Chart

We also want to see how the power generation for our wind turbines change over the course of the day. We will display this information using a Line Chart.

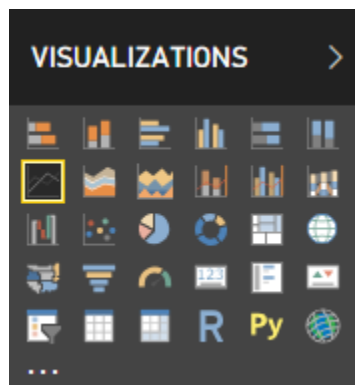
We will need to create a new column yet again to convert and group our timestamp column by the numeric hour of the day.

- Step 1: Create a **New Column** from the **Data View** tab again by right clicking the header of any column -> New Column and add the following Dax Formula:

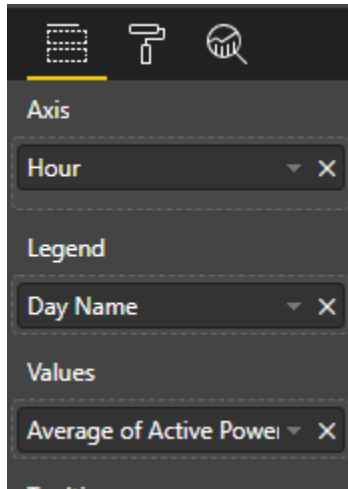
Hour = HOUR((BlackMesaWindFarm[TimeStamp])))



- Step 2: Now it's time to configure the visual. Click some blank space, and add a **Line Chart**:

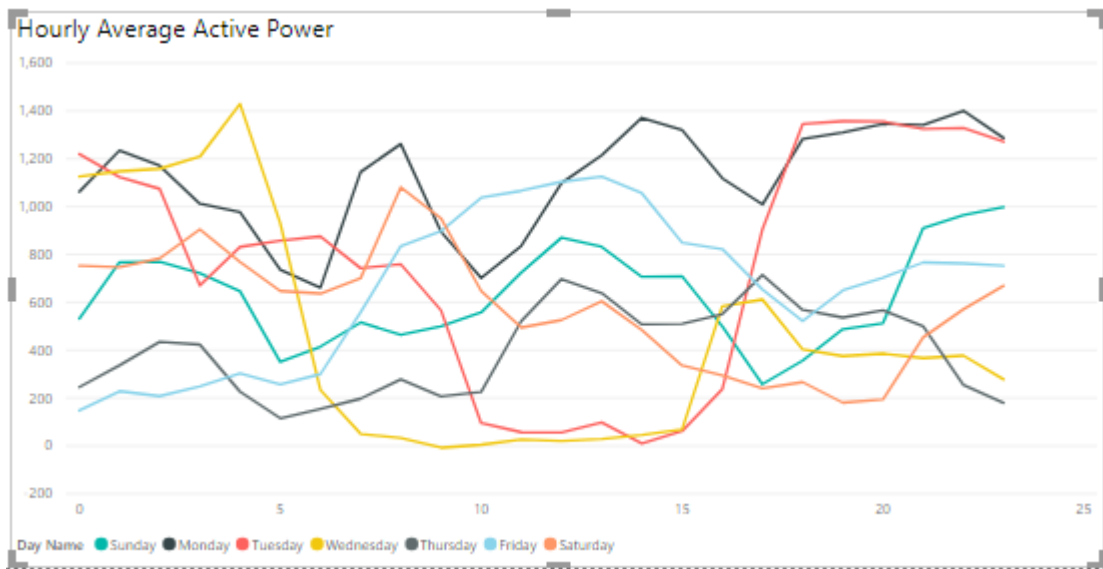


- Step 3: Use **Hour** as the **Axis**, **Day Name** as the **Legend**, and **Average of Active Power** as the **values** (summarize Active Power as an Average):



- Step 4: Adjust the **formatting** of the **Line Chart**:
 - Move the Legend to the bottom
 - Change the title to “Hourly Average Active Power”, text color black, and increase the text size of the title to match the other visuals
 - Optionally change the trend colors

The line chart should look something like this:

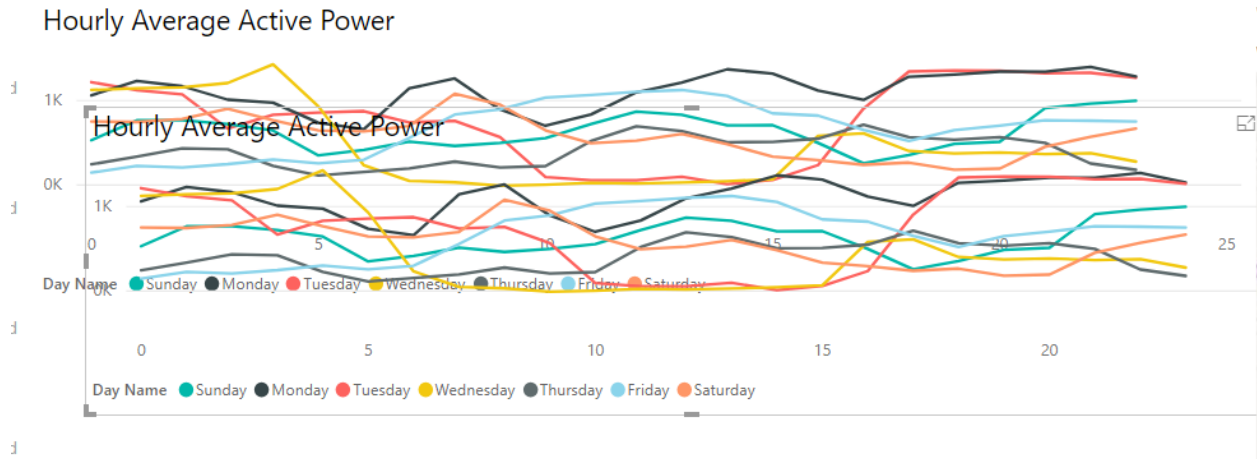


Wind Speed by Hour – Line Chart

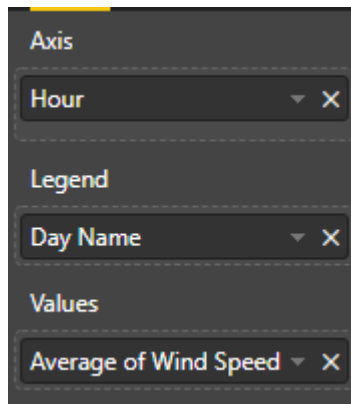
We also want to see how the wind speed change over the course of the day. We'll display this information using a Line Chart right below our Hourly Average Active Power Line Chart.

Since we want the new Line Chart to have the same formatting as our previous symbol, we can simply copy and paste the above symbol and change the values field to average of wind speed.

- Step 1: Select the **Hourly Average Active Power Line Chart** and **Ctrl-C** to copy it and **Ctrl-V** to paste it below. It will copy and paste the symbol like below.



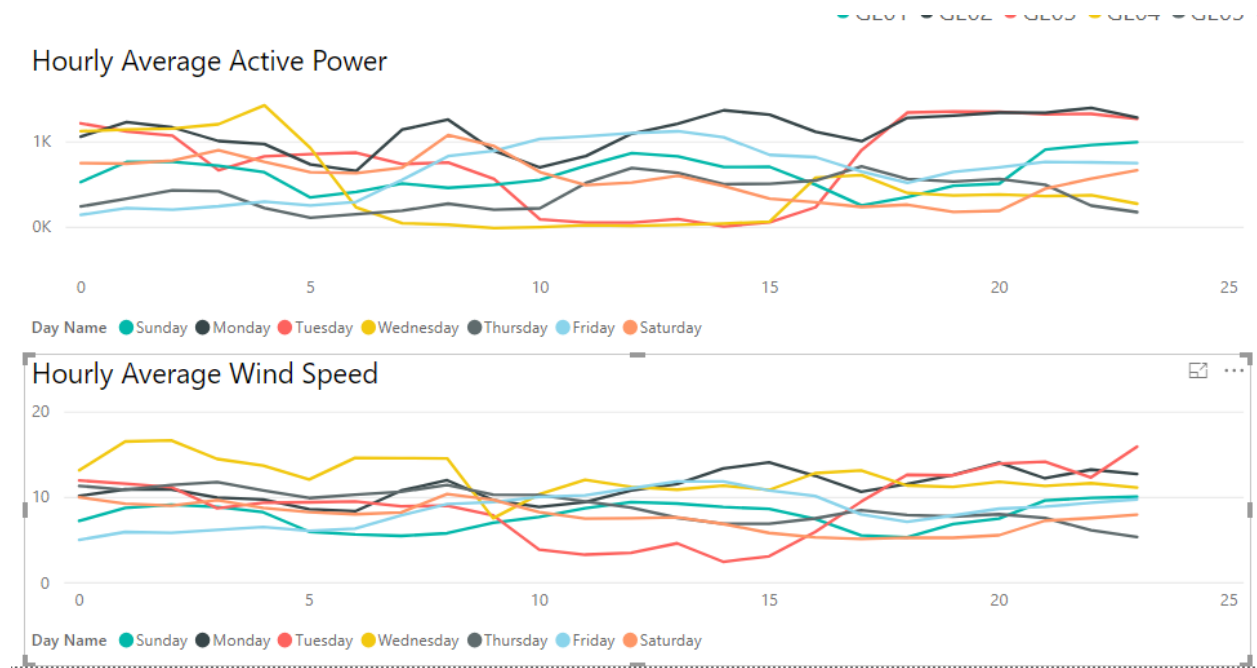
- Step 2: Drag the symbol down and change the **Values** field to **Average of Wind Speed**.



- Step 3: Adjust the **formatting** of the **Line Chart**:
 - Change the title to "Hourly Average Wind Speed"
 - Align the chart so that it is below the Hourly Average Active Power Line Chart

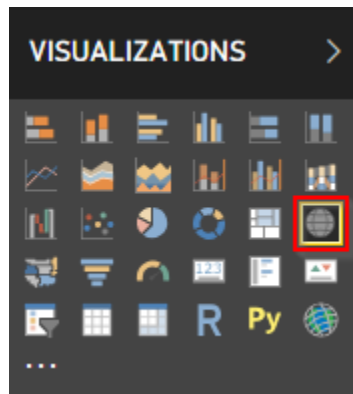
Exercise 1 – Power Generation

The line charts should look something like this:

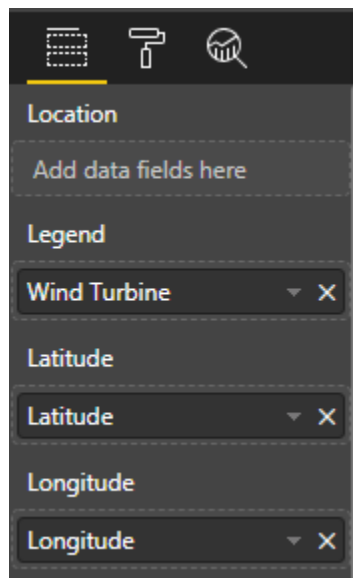


Wind Turbine Location – Map

- Step 1: Click on an empty white space in the Report view and select the **Map** Symbol:

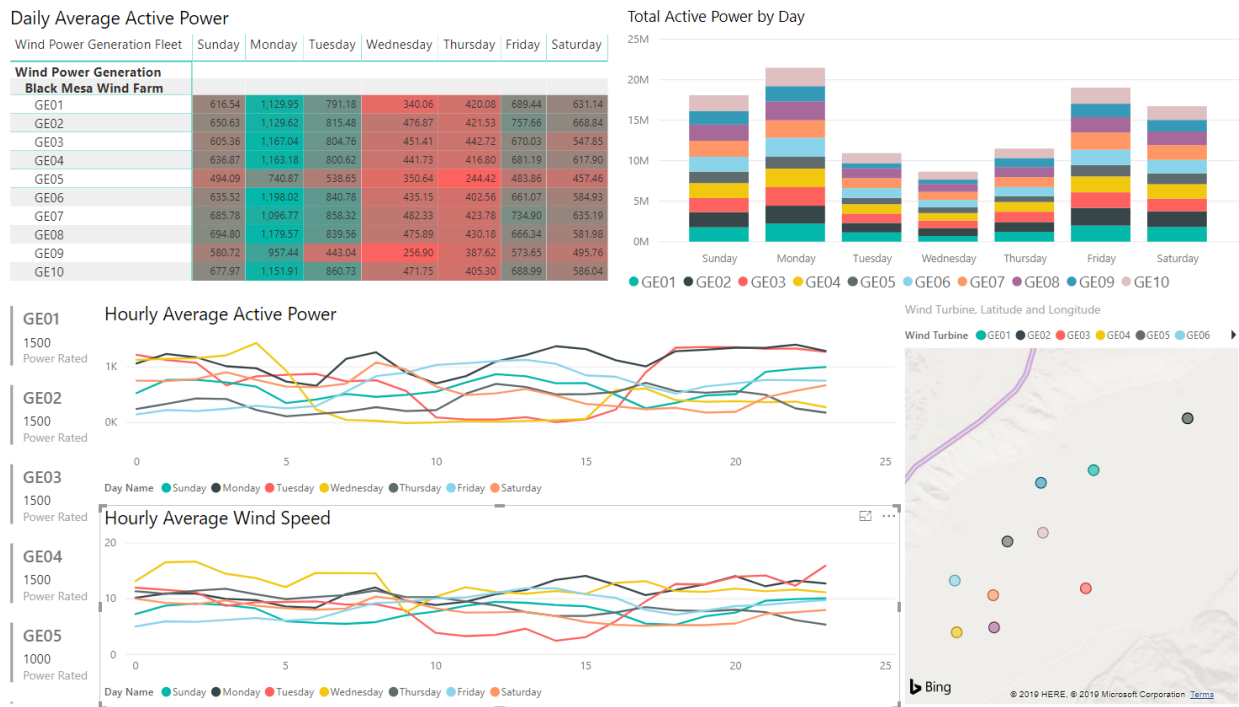


- Step 2: Add **Wind Turbine** to the **Legend**, **Latitude** to the **Latitude field**, and **Longitude** to the **Longitude field**, making sure to select **Don't Summarize** for the latitude and longitude fields or you will receive an error:



Exercise 1 – Power Generation

The report should now look similar to the example outlined at the start of the exercise:



You can rearrange and format the symbols to your liking. Optionally you can add a title and save the report and transfer it to your machine.

Takeaways

You should now feel comfortable with the following Power BI features:

- Hierarchies
- Formatting symbols
- DAX formulas
- Adding additional columns to your data sets
- Different types of summations of attributes
- Slicing and Dicing of data

Discussion: Slicing and Dice the data to look at each day of the week. Does the active power line chart closely follow the wind speed line chart for each of the days?

Discussion: Are there days and times of the day when the correlation does not hold? Are there additional factors at play which could affect correlation? We will explore this in the next exercise.

Exercise 2 – Power Generation Analysis

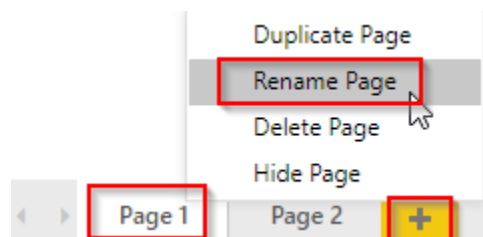
In this exercise, we will analyze factors which affect wind power generation with respect to our wind farm. The final display you create for this exercise will look similar to this:



The goal is to assess how power generation changes as a function of environmental factors. General steps:

- Configure a **Scatter Chart** for the wind turbine power curve
- Configure a **Group** to create bins for air temperature and wind turbulence
- Configure a **R CorrPlot** to analyze correlation between active power and environmental factors
- Configure **Line Charts** to Active power by air temperature and wind turbulence
- Create a **New Table** to convert turbine position to compass directions
- Configure a **Pie Chart** to show Turbine directions

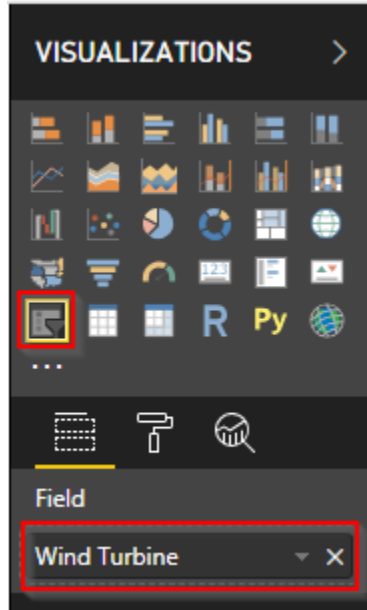
Before we start, rename Page 1 to **Power Generation** and add a new page for exercise 2. Rename the Page to Power Generation Analysis.



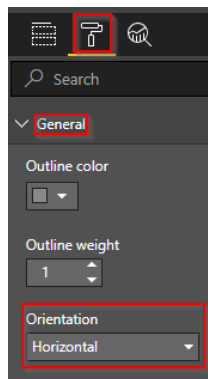
Power Curve – Scatter Chart

First let's add a slicer to filter our report by Wind Turbine.

- Step1: Select the **Slicer** symbol and add it to the report. Add **Wind turbine** to the **Field** section.



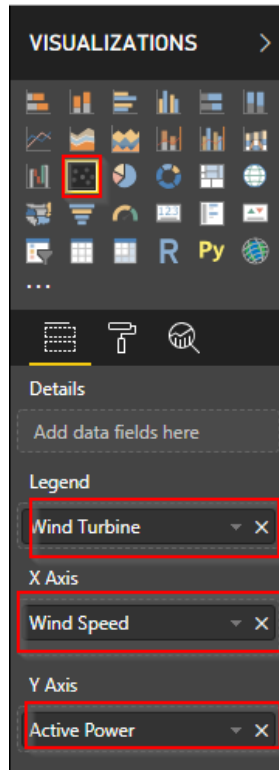
- Step 2: Format the slicer so that the **orientation** is **horizontal** and **stretches** across the **top of our report**.



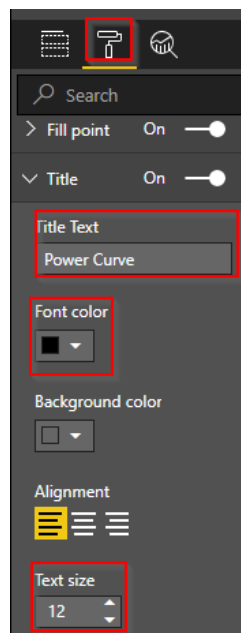
GE01	GE02	GE03	GE04	GE05	GE06	GE07	GE08	GE09	GE10
------	------	------	------	------	------	------	------	------	------

Power Curve Completion Matrix

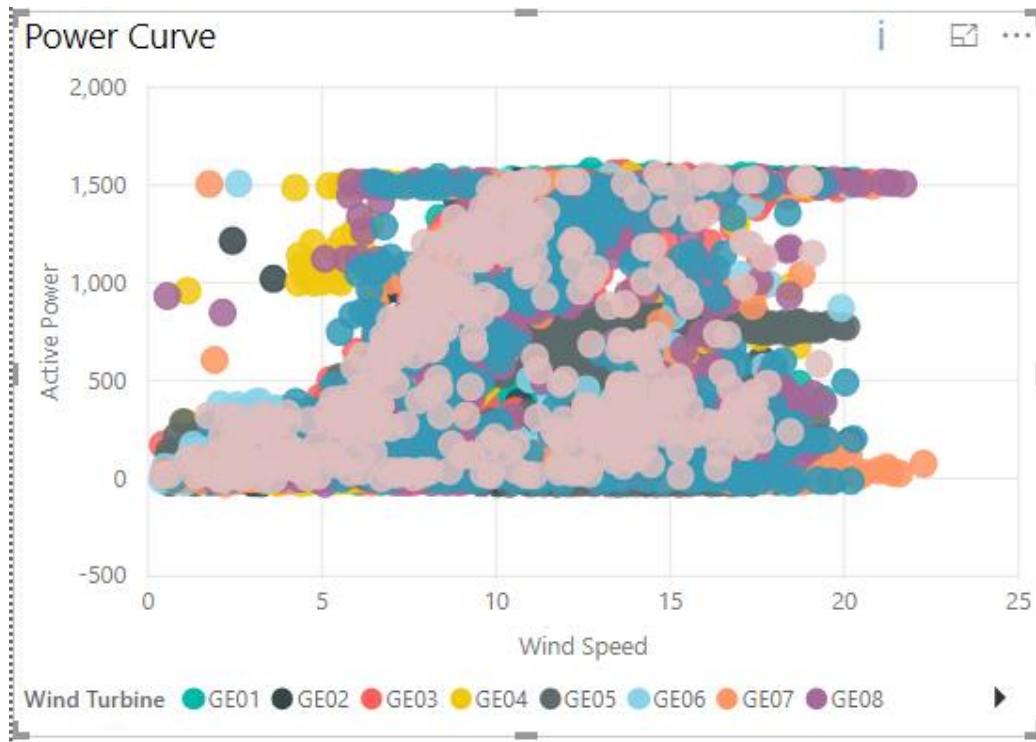
- Step 3: Select an empty part of the report and add a **Scatter Chart** to the report. Add **Wind turbine** as the **Legend**, **wind speed** as the **X Axis** (Don't Summarize), and **Active Power** as the **Y Axis** (Don't Summarize):



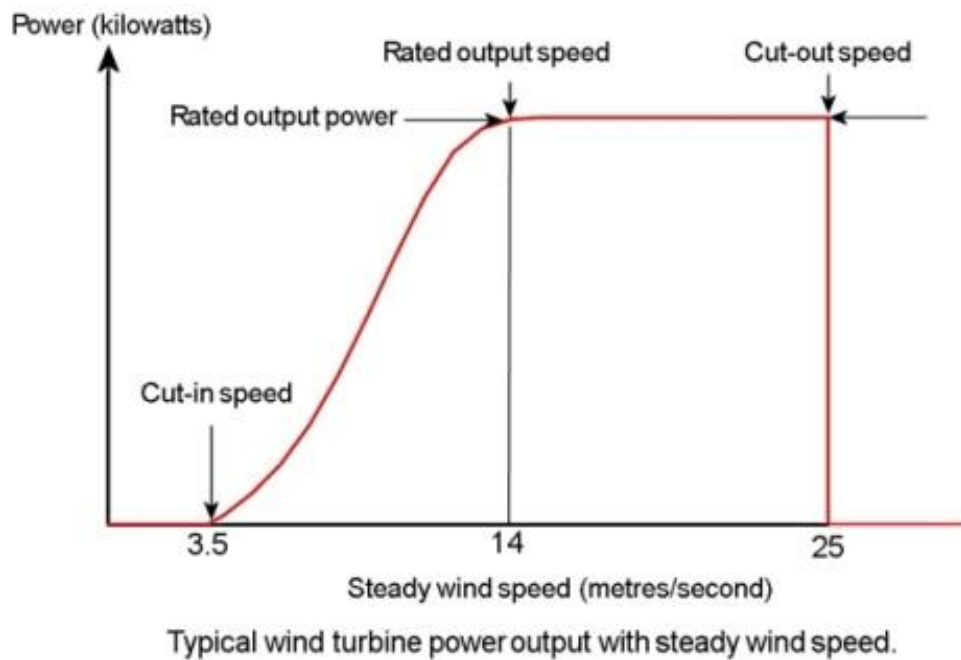
- Step 4: Format the **Scatter chart** so that the **legend** is on the **bottom** and the **Title** is “**Power Curve**”.



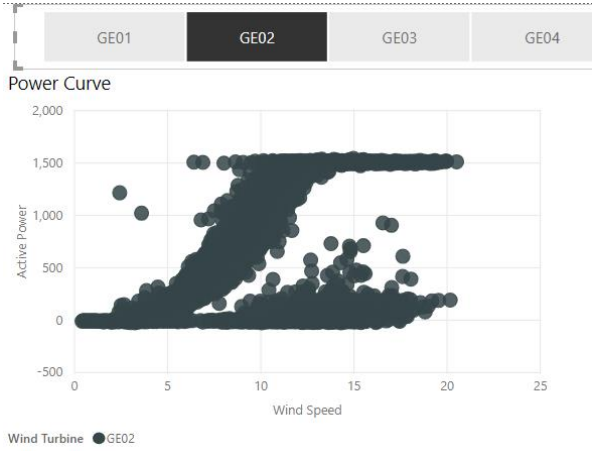
The Power Curve should look something like this:



Power Curves of wind turbines indicate how large the power generation of a given wind turbine would be at a given wind speed. The typical power curve looks something like this:



Note that there is a fair bit of noise which makes it difficult to see the power curve. We will talk about data manipulation in the appendix via python as a way to filter out that noise. For now, filter the report by selecting GE02 from the slicer on the top and you should see a clearer power curve for turbine 2.



This can be useful for analyzing actual vs. expected power curves and determine if a wind turbine is behaving ideally.

Correlation – R Corr. Plot

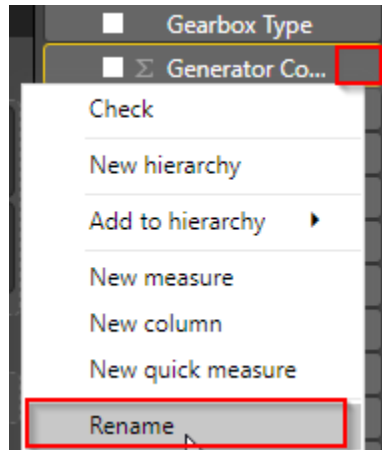
We will now add a custom symbol by taking advantage of R integration with Power BI.

The R Corr. Plot is provided by the community and more details about it can be found here:

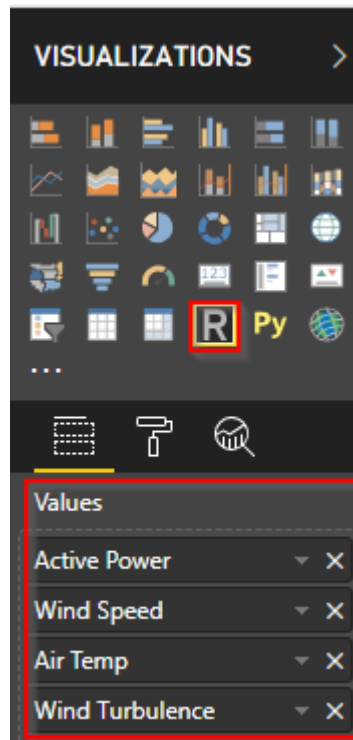
<https://community.powerbi.com/t5/R-Script-Showcase/Correlation-Plot/td-p/58462>

We have already installed the R Engine and R scripts needed for the plot on the VMs.

- Step 1: Note that we will be using Generator Cooling Air Temperature to determine ambient air temperature, so let's rename that attribute to **Air Temp**:



- Step 2: Select R script visual from the visualization pane and add **Active Power**, **Wind Speed**, **Wind turbulence**, and **Air Temp** to the **Values** field. Make sure to select Don't summarize for all the values.



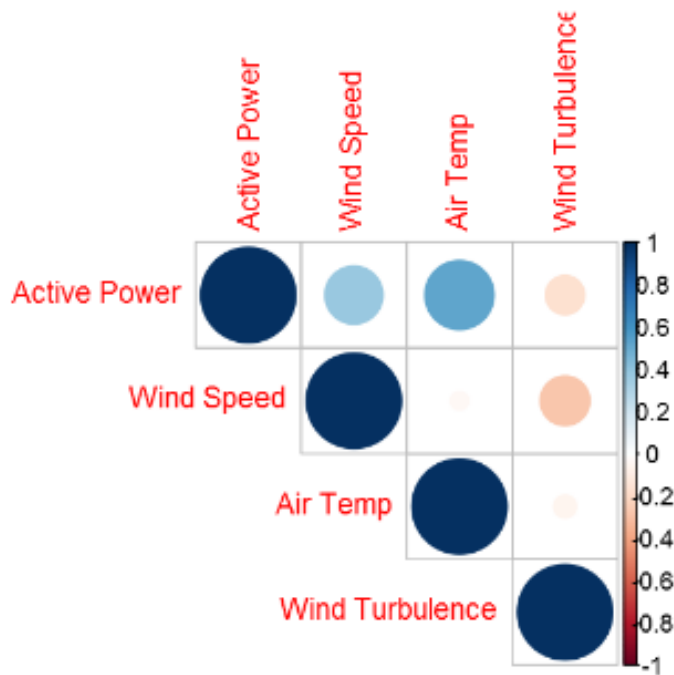
- Step 3: Open **Rcorrplotscript.txt** from the classes folder and copy and paste the text into the **R script editor**:

```

R script editor
⚠ Duplicate rows will be removed from the data.
      mar = defMar, tl.col = tl.col, tl.cex=tl.cex,
      number.digits=number.digits, number.cex=number.cex, addCoef.col=addCoef.col)
}else{ #empty correlation plot
  plot.new()
  pbiWarning<-paste(pbiWarning, "Not enough input dimensions", sep="\n")
}
#add warning as subtitle
if(showWarnings)
  title(main = NULL, sub = pbiWarning, outer = FALSE, col.sub = "gray60", cex.sub = 0.75)

```

- Step 4: Name the Corrplot **“Correlation Matrix”** and increase the text size. Your Corrplot should look something like this:

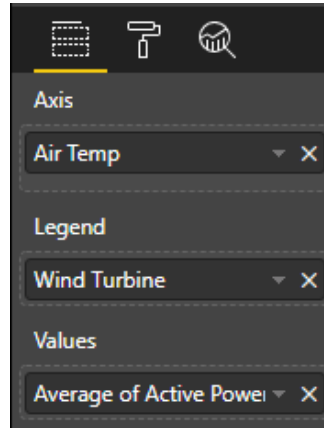


You can add more variables to the Corrplot to determine other relationships.

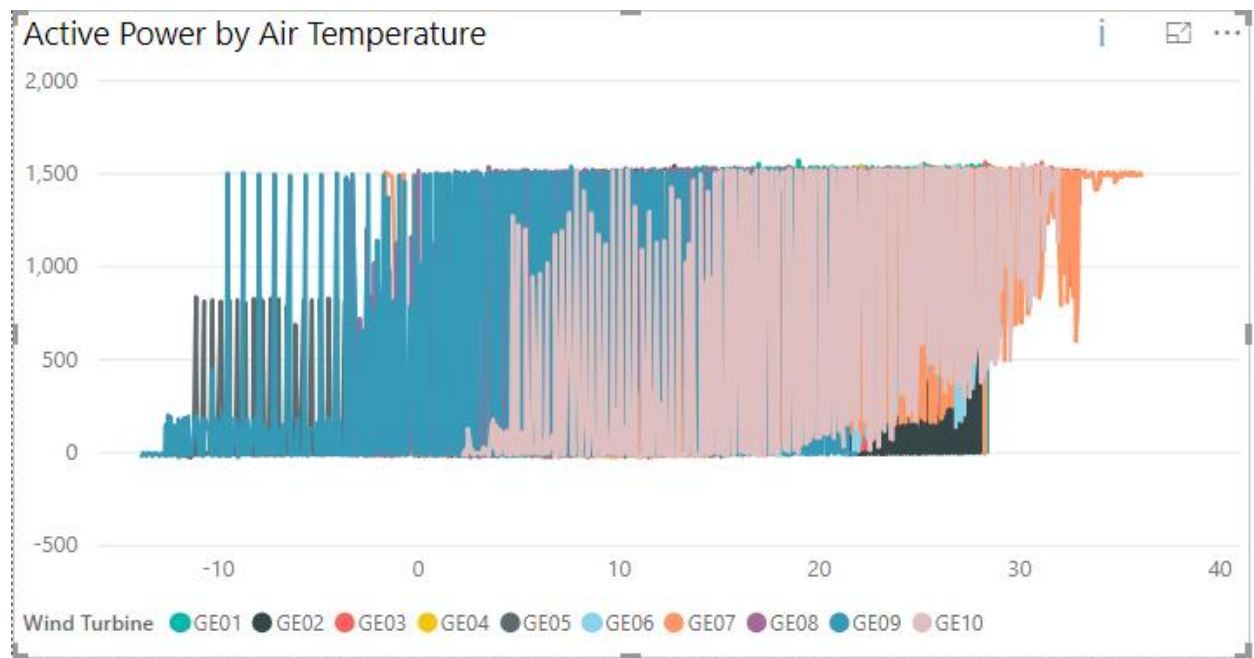
Discussion: What conclusions can you draw from this Corrplot?

Active Power by Air Temperature – Line Chart

- Step 1: Insert a Line Chart into our report.
- Step 2: Use **Air Temp** as the **Axis**, **Wind Turbine** as the **Legend**, and **Average of Active Power** as the **Value**:



The line chart will look something like this:



This is not very helpful to display as we were expecting more of a line for each of our turbines as the temperature increases. Since each turbine will have a distinct value for active power and a different value for the air temperature for each row in our dataset, we see that all of the active power and air temperature values are plotted by the line chart and it looks more like a scatter plot.

What we really want is the average of active power and each discrete air temperature, for example at every 1-degree Fahrenheit. We can solve this by grouping all the continuous values of air temperature into discrete 1-degree groups and plot that instead.

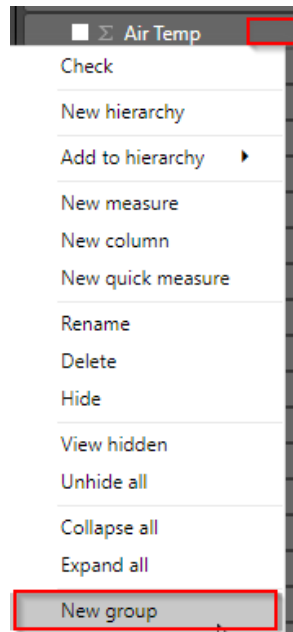


Tip

Power BI allows for the creating of groups which will group continuous into discrete groups of the size of our choosing. More information on groups can be found [Here](#).

Let create a new group to group our reading for Air Temperature into 1-degree intervals.

- Step 3: Click on the ellipses next to **Air Temp** and select **New Group**:



- Step 4: Name the new group **“Air Temp (1)”** and change the **Bin Size to 1**. This will group all readings from our rows into 1 -degree groups allowing for easier visualization. For example, Air temperatures like 2, 2.3, 2.4, 2.8 will now all be grouped into the 2 bins of Air Temp.

Groups

Name: Field:

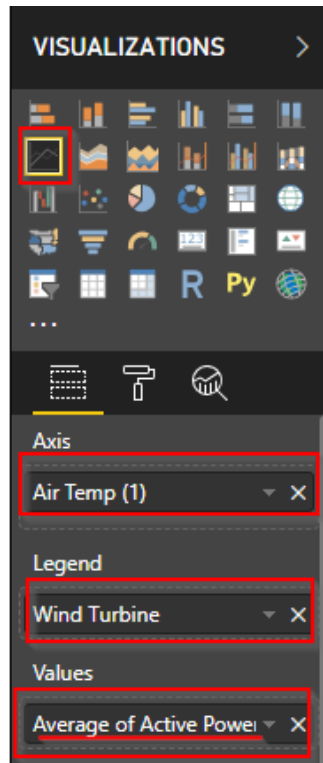
Group type: Min value:

Bin Type: Max value:

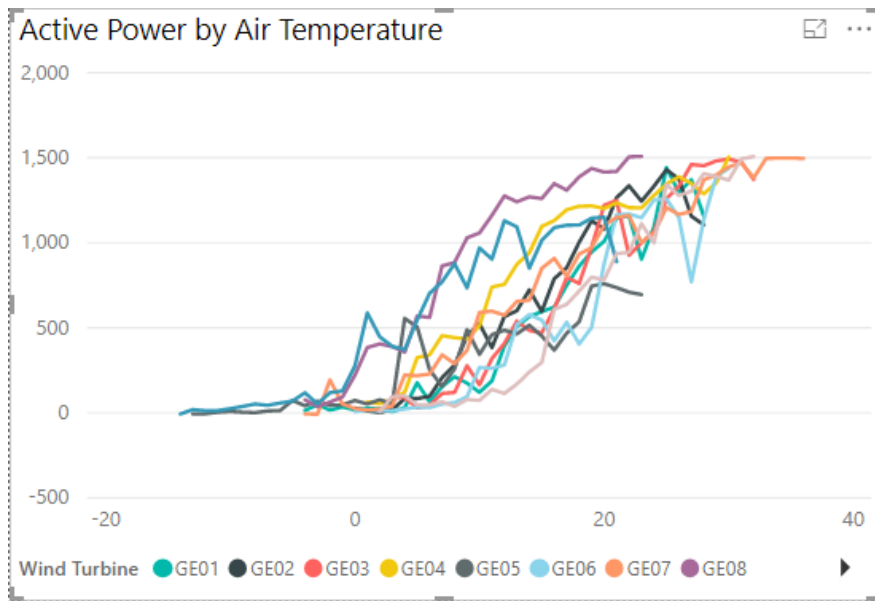
Binning splits numeric or date/time data into equally sized groups. The default bin size is calculated based on your data.

Bin size:

- Step 5: Adjust the Line plot to report with the **Axis** as **Air Temp(1)** instead of the non-binned Air Temp.



- Step 6: Format the symbol by moving the **Legend** on the **bottom** and changing the **Title** to **“Active Power by Air Temperature”** and making it larger:



You can see that this give a nice line graph without trying to draw a line for each data point of Air Temp. We can confirm that increasing Air temperature does correlate with an increasing Active Power.

Active Power by Wind Turbulence – Line Chart

Wind Turbulence is a measure of wind speed changes, and we are calculating this variable in AF via the following formula:

$$(\text{Wind Speed 10 Min StDev} / \text{Wind Speed 10 Min Avg}) * 100$$

We can navigate to the AF from PI System Explorer to see how this formula is defined:

The screenshot displays the PI System Explorer interface for asset 'GE02'. A 'Formula Configuration' dialog box is open for the 'Wind Turbulence' attribute. The dialog shows the following configuration:

- Name:** Wind Turbulence
- Description:** (empty)
- Properties:** <None>
- Categories:** Met
- Default UOM:** percent

The 'Formula Configuration' dialog includes a 'Parameters' section with:

- A=Wind Speed 10 Min StDev
- B=Wind Speed 10 Min Avg

The 'Equations' section contains the formula: $(A/B)*100$.

The 'Result' section shows:

- UOM: <Default> (%)
- Minimum: (empty)
- Maximum: (empty)
- Stipped:

The background table shows the following data for the 'Wind Turbulence' attribute:

Name	Value
Power Rated	1500 kW
Production Delta	-32.677 kW
Reactive Power	PI Point not found 'GE02_Q'.
Rotor Speed	
Serial Number	
Torque, A	
Torque, S	
Tower Acc	
Tower Bas	
Tower Bas	
Tower Def	
Turbine Av	
Turbine Co	
Turbine Na	
Turbine St	
Turbine St	
Vacuum - f	
Wind Curv	
Wind Devi	
Wind Devi	
Wind Farm	
Wind Speed	12.827 m/s
Wind Speed 10 Min Avg	PI Point not found '\\PISRV01\GE02_...
Wind Speed 10 Min StDev	PI Point not found '\\PISRV01\GE02_...
Wind Turbulence	Data was not available for attribute '...

Note that we have not brought over Wind Speed 10 min Avg and StDev into this specific PI Server, but they exist on the source PI Server.



Best Practice

Use Asset Analytics to create and manage complex calculations and write the outputs directly to PI Points to avoid having to write multiple DAX formula for each dataset and sacrificing standardization and manageability.

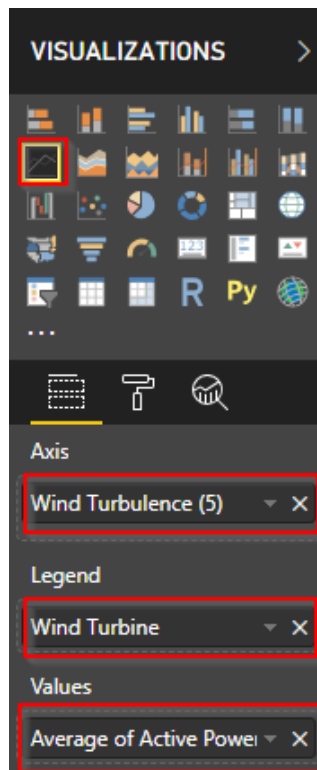
- Step 1: Create a **New Group** for **Wind Turbulence** and make the **Bin Size 5** and change the name to **“Wind Turbulence (5)”**:

The screenshot shows a 'Groups' dialog box with the following fields and values:

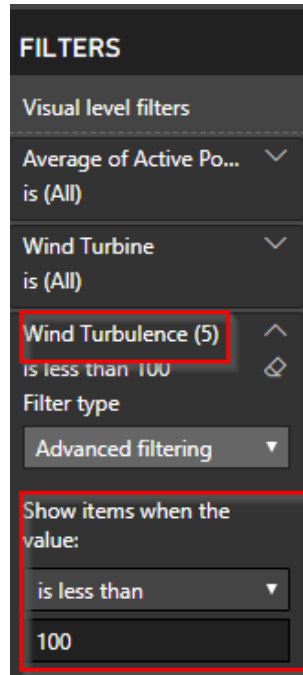
Name	Wind Turbulence (5)	Field	Wind Turbulence
Group type	Bin	Min value	1.41285016653179
Bin Type	Size of bins	Max value	1640.14168931959
Bin size	5		

Below the fields, there is a note: "Binning splits numeric or date/time data into equally sized groups. The default bin size is calculated based on your data." Below this note is a "Reset to default" button. At the bottom right, there are "OK" and "Cancel" buttons.

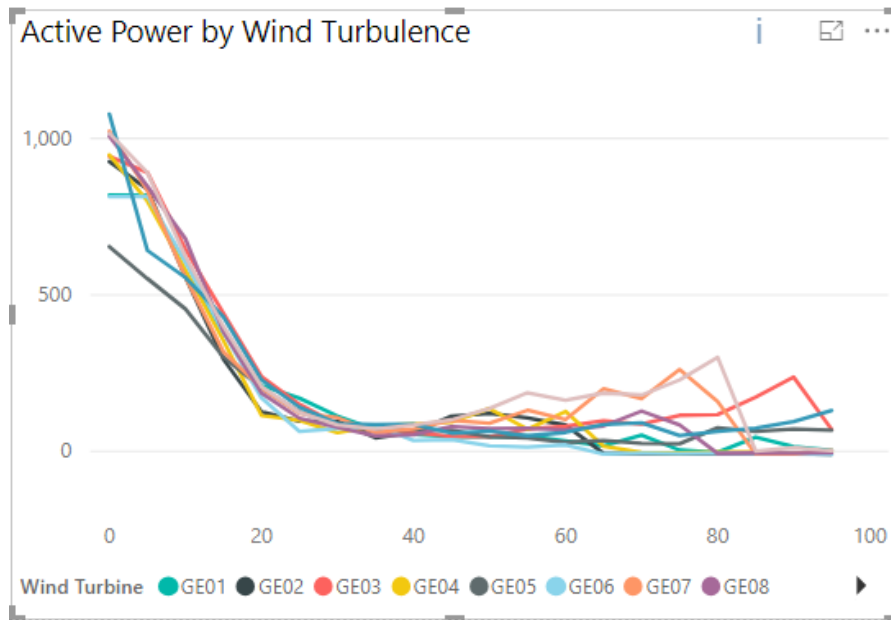
- Step 2: Add a new **line chart** to the report, with **Wind Turbulence (5)** as the **Axis**, **Wind Turbine** as the **Legend**, and **Average of Active Power** as the **Values**:



- Step 3: Also **Filter** the data in the visualization pane by expanding wind turbulence (5) and only show items when the **value is less than 100**:



Format and resize the symbol so that the chart looks like this:



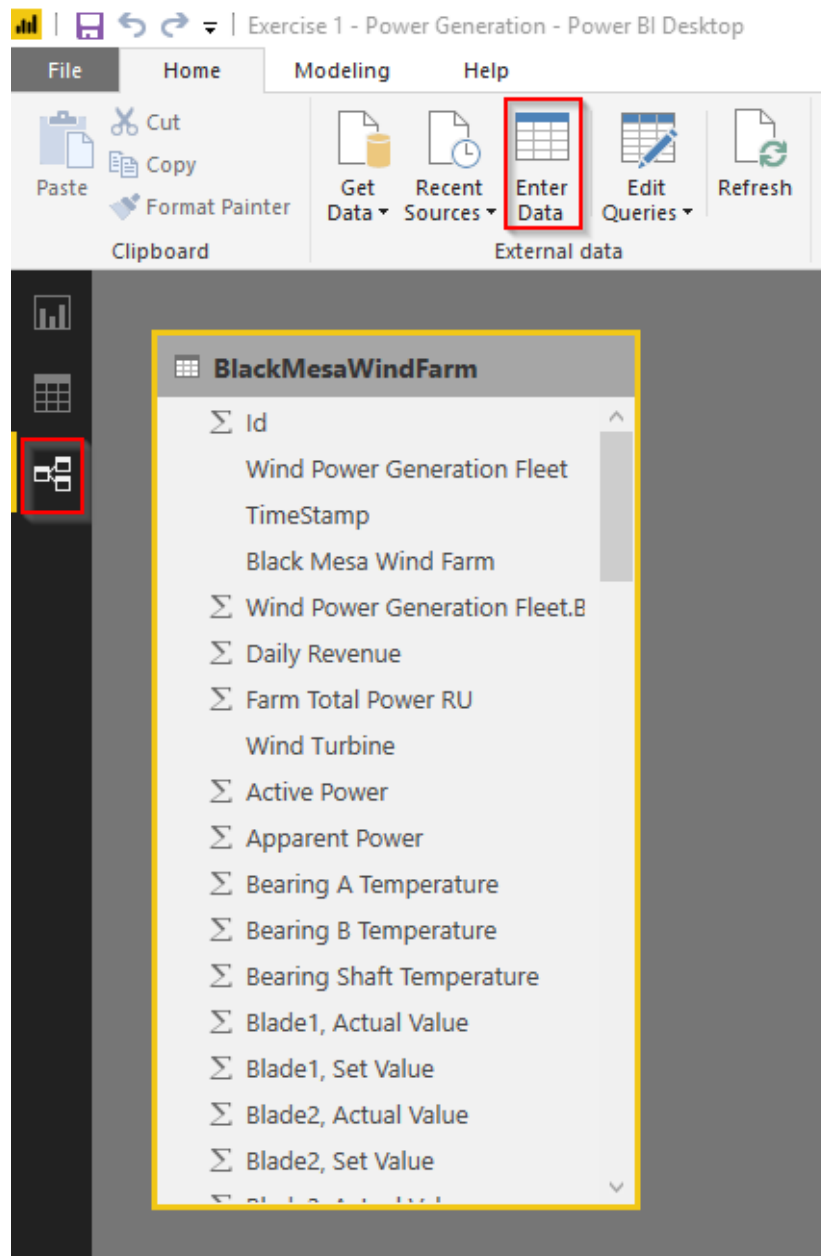
Discussion: What conclusions can we draw from this graph? Is greater wind turbulence preferable for wind power generation?

Nacelle Direction – Pie Chart

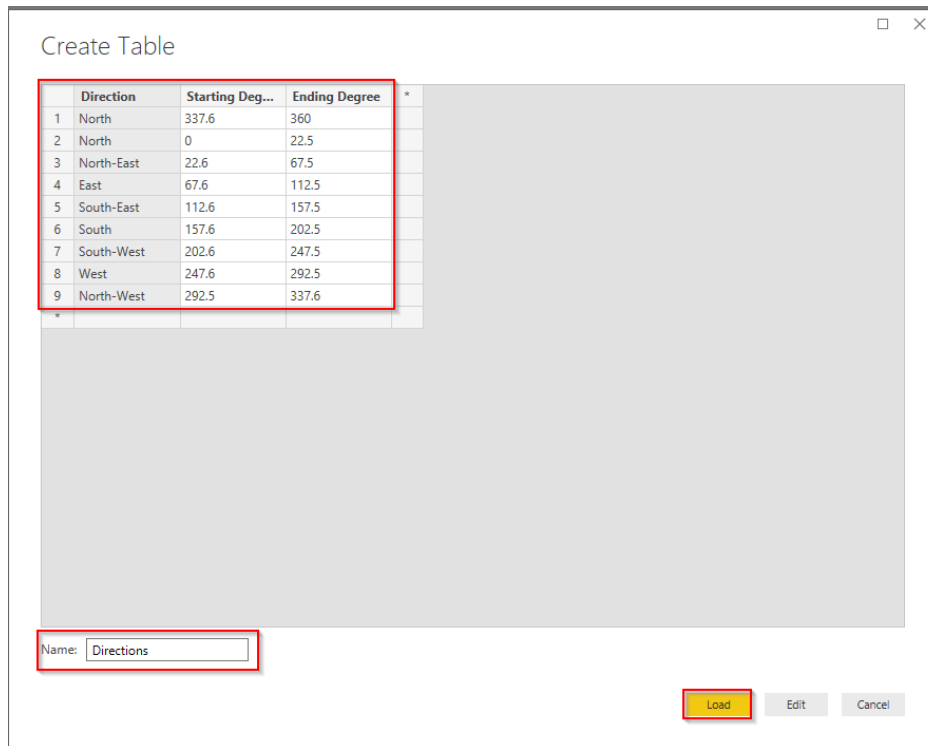
An interesting measure for power generation might be the direction the wind turbine is facing as well as wind direction. We do not have wind direction data in our data set but we can analyze whether the turbines tend to generate more power for a specific direction.

Turbine direction is best depicted by nacelle direction, but this reading is given in degree with 0 degree specifying North. We will need to convert the degrees to compass headings.

- Step 1: Let's create a **New Table** by going to the **Relationships** pane and selecting **Enter Data**:



- Step 2: Fill in the table as shown, naming it Directions. Alternatively, a text file called **Directions.txt** is located in the **Class folder** that you can copy and paste:



This table lists the degree ranges of each direction.

- Step 3: Create a new column in the BlackMesaWindFarm table and use the following DAX for the column:

Direction = CALCULATE(VALUES(Directions[Direction]), FILTER(Directions, BlackMesaWindFarm[Nacelle Position]>= Directions[Starting Degree] && BlackMesaWindFarm[Nacelle Position] < Directions[Ending Degree]))

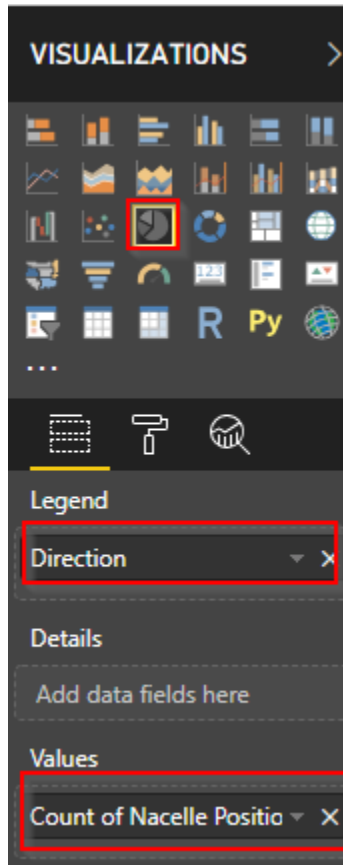
Torque, Set Value	Tower Acceleration	Tower Deflection	Turbine Availability	Turbine Count	Turbine Name	Turbine State	Turbine Status On/Off	Vacuum - External Oil Heater	Wind Curve	Wind Deviation 10s	Wind Deviation 1s	W6
63.5029563903809	88.4314041197695	2552	100	1	WTG04	Load Operation	1		0	11.3005104064941	22.0597534179688	Black N
86.2671279907227	93.514564541602	2552	100	1	WTG04	Load Operation	1		0	-13.6361932754517	-29.8067626853125	Black N
93.2867584228516	98.6845779418945	2552	100	1	WTG04	Load Operation	1		0	-12.6958045959473	-30.7598075866699	Black N

This will populate the new column with values from the Directions table from the Direction column where the nacelle position column of our BlackMesaWindFarm table is within the starting and ending degrees on the Directions table.

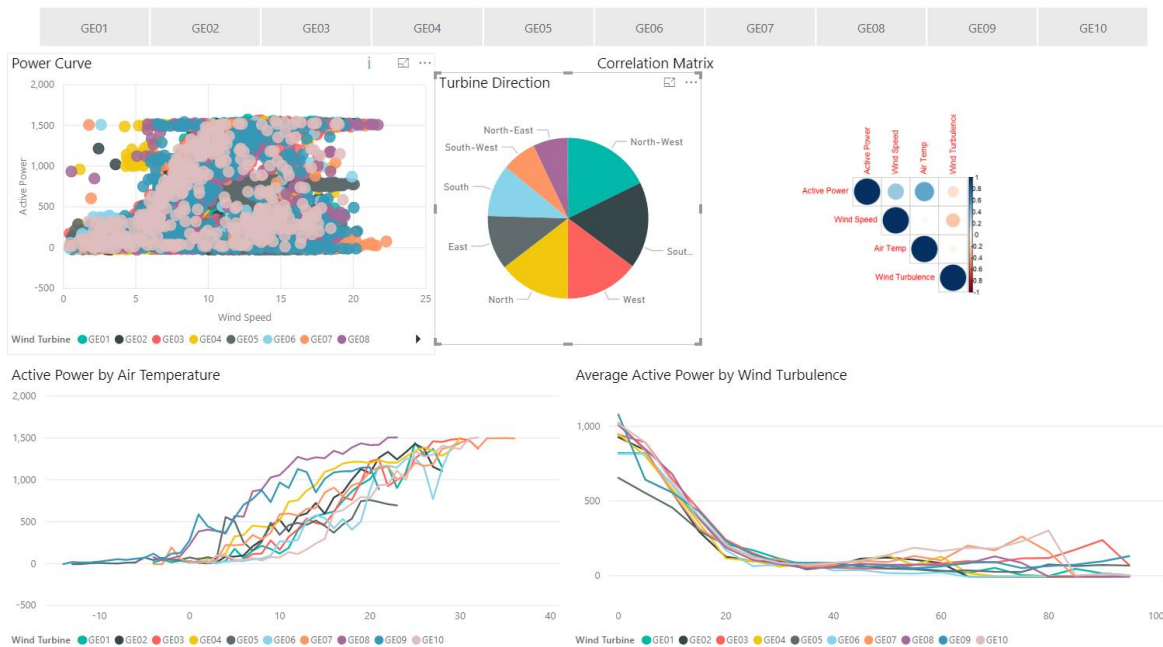
This gives us a way to merge the 2 tables together without defining an explicit relationship.

Exercise 2 – Power Generation Analysis

- Step 4: Add a **Pie Chart** to our report and select our **new column Direction** for the **Legend** and **Nacelle Position** for the **values**. Change Nacelle Position to Count of Nacelle Position:



The report should now look similar to the example outlined at the start of the exercise:

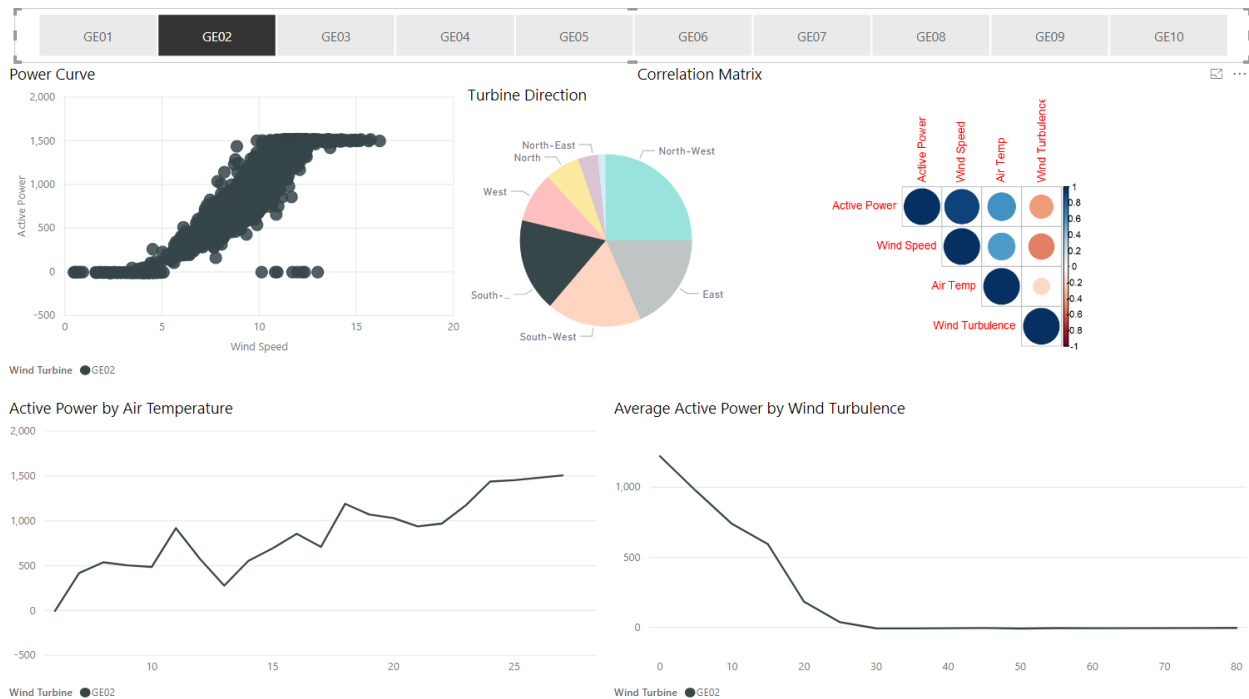


Takeaways

You should now feel comfortable with the following Power BI features:

- Slicers
- Adding custom symbols (R Corr. Plot)
- Creating groups to bin continuous measurements
- Adding new tables
- Linking multiple tables via DAX formulas

We can also splice the report in interesting ways to determine more relationships. For example, selecting wind turbine 2 (GE02) from the slicer and South-East for the turbine direction from the pie-chart shows a much more traditional power for wind turbines:



We can see that generally when the wind was coming from South-East the turbine was behaving according to its power curve.

Wind Direction may affect power generation just like wind temperature and turbulence. These are all factors to consider outside of wind speed and we can see why wind speed might not always correlate to an increase in power generation like we mentioned in Exercise 1.

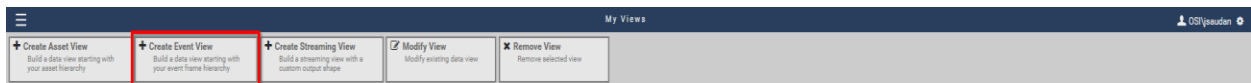
Go ahead and try different combinations to see if you can make any more conclusions.

Exercise 3 – Curtailment Report

In this exercise we’re going to analyze the effect of curtailment on the power generation for our wind farm, as well as explore which turbines experience the most curtailment and what factors contribute to this curtailment.

In AF, we have configured event frames to capture periods of curtailment. These event frames are triggered when the operational status of the turbine is “Feathering Position” and are closed when the turbine leaves this status. In the context of wind turbines, being in the feathering position means that the turbine blades are turned parallel to the wind direction, to prevent the blades from spinning excessively.

With the PI Integrator for Business Analytics, we can export the attributes, calculations, and metadata from these event frames. You would do this by selecting “Create Event View” at the main menu for the integrator:

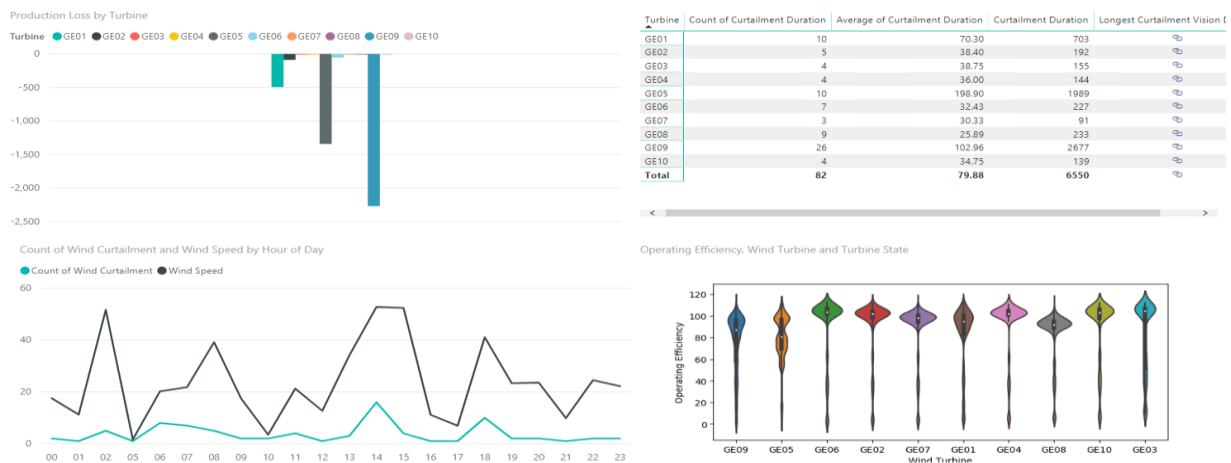


For this exercise, the integrator view has already been published for you. The view in this exercise contains all of the Curtailment event frames for the same timeframe as the previous two exercises, along with all of the attributes included in the event frame template. General steps:

- Configure a **Clustered Column Chart** to discover which turbines suffer the most from curtailments
- Create a **Matrix** to show statistical details about our curtailments
- Create a **New Table** to find the longest curtailment per turbine
- Utilize **PI Vision** by creating dynamic links to PI Vision displays from the report
- Write **DAX Formulas** to find out what times we most frequently experience curtailments
- Leverage the power of **Python** by creating a visualization from a python package to gauge operational efficiency

The final product for this exercise should look something like the screenshot below:

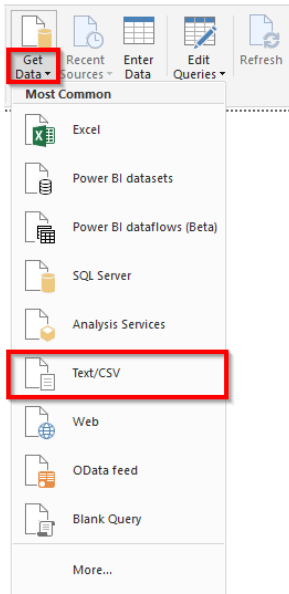
Curtailment Report



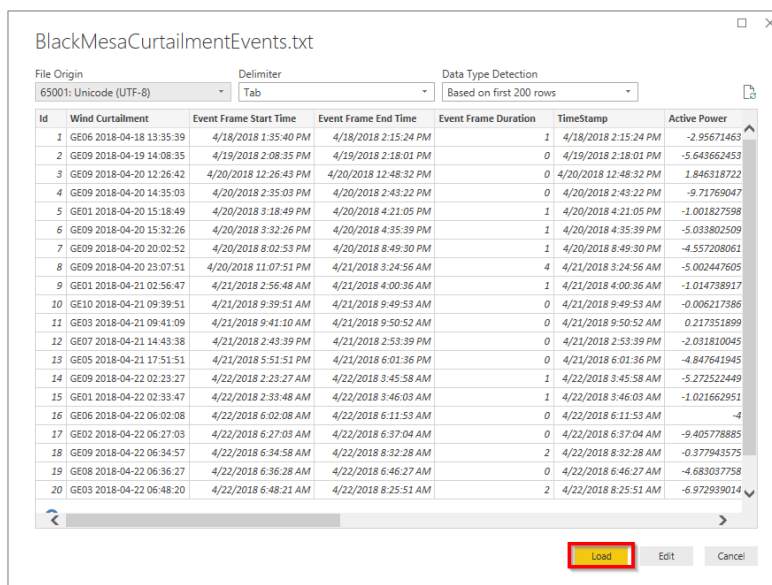
Importing the Data

Let's import our data into Power BI.

- Step 1: To do that, click the **Get Data** button on the Home tab, and then select the source you want to get data from. In our case, we've exported the data as a text file, so select **Text/CSV**:



- Step 2: Select the **"BlackMesaCurtailmentEvents"** file. Power BI will then attempt to connect to this file, and once it does, it will display a preview of the data in the file.
- Step 3: If the breakdown of data into their respective columns looks correct, click Load to load the data into the workspace:



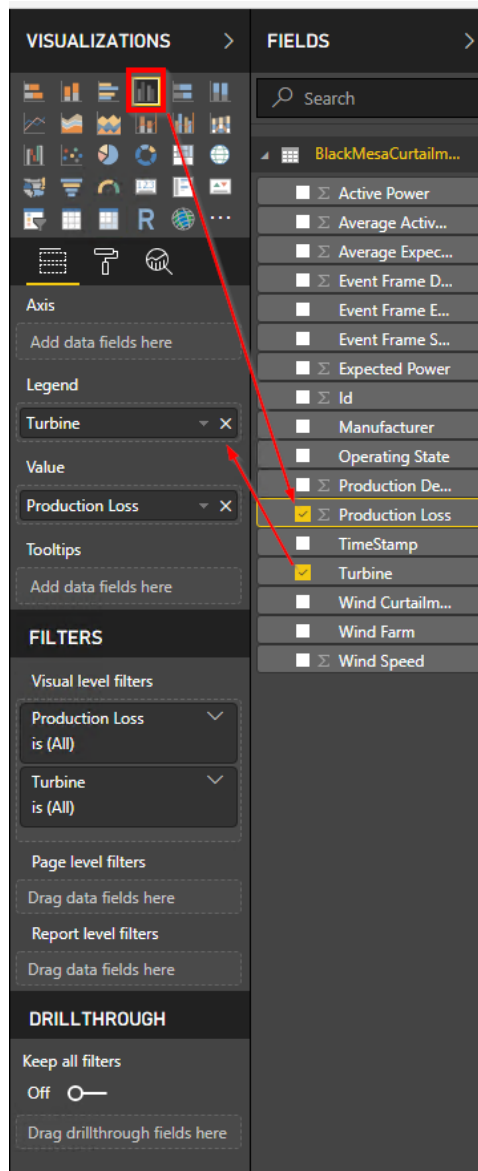
Id	Wind Curtailment	Event Frame Start Time	Event Frame End Time	Event Frame Duration	TimeStamp	Active Power
1	GE06 2018-04-18 13:35:39	4/18/2018 1:35:40 PM	4/18/2018 2:15:24 PM	1	4/18/2018 2:15:24 PM	-2.95671463
2	GE09 2018-04-19 14:08:35	4/19/2018 2:08:35 PM	4/19/2018 2:18:01 PM	0	4/19/2018 2:18:01 PM	-5.643662453
3	GE09 2018-04-20 12:26:42	4/20/2018 12:26:43 PM	4/20/2018 12:48:32 PM	0	4/20/2018 12:48:32 PM	1.846318722
4	GE09 2018-04-20 14:35:03	4/20/2018 2:35:03 PM	4/20/2018 2:43:22 PM	0	4/20/2018 2:43:22 PM	-9.71769047
5	GE01 2018-04-20 15:18:49	4/20/2018 3:18:49 PM	4/20/2018 4:21:05 PM	1	4/20/2018 4:21:05 PM	-1.001827598
6	GE09 2018-04-20 15:32:26	4/20/2018 3:32:26 PM	4/20/2018 4:35:39 PM	1	4/20/2018 4:35:39 PM	-5.033802509
7	GE09 2018-04-20 20:02:52	4/20/2018 8:02:53 PM	4/20/2018 8:49:30 PM	1	4/20/2018 8:49:30 PM	-4.557208061
8	GE09 2018-04-20 23:07:51	4/20/2018 11:07:51 PM	4/21/2018 3:24:56 AM	4	4/21/2018 3:24:56 AM	-5.002447605
9	GE01 2018-04-21 02:56:47	4/21/2018 2:56:48 AM	4/21/2018 4:00:36 AM	1	4/21/2018 4:00:36 AM	-1.014738917
10	GE10 2018-04-21 09:39:51	4/21/2018 9:39:51 AM	4/21/2018 9:49:53 AM	0	4/21/2018 9:49:53 AM	-0.006217386
11	GE03 2018-04-21 09:41:09	4/21/2018 9:41:10 AM	4/21/2018 9:50:52 AM	0	4/21/2018 9:50:52 AM	0.217351899
12	GE07 2018-04-21 14:43:38	4/21/2018 2:43:39 PM	4/21/2018 2:53:39 PM	0	4/21/2018 2:53:39 PM	-2.031810045
13	GE05 2018-04-21 17:51:51	4/21/2018 5:51:51 PM	4/21/2018 6:01:36 PM	0	4/21/2018 6:01:36 PM	-4.847641945
14	GE09 2018-04-22 02:23:27	4/22/2018 2:23:27 AM	4/22/2018 3:45:58 AM	1	4/22/2018 3:45:58 AM	-5.272522449
15	GE01 2018-04-22 02:33:47	4/22/2018 2:33:48 AM	4/22/2018 3:46:03 AM	1	4/22/2018 3:46:03 AM	-1.021662951
16	GE06 2018-04-22 06:02:08	4/22/2018 6:02:08 AM	4/22/2018 6:11:53 AM	0	4/22/2018 6:11:53 AM	-4
17	GE02 2018-04-22 06:27:03	4/22/2018 6:27:03 AM	4/22/2018 6:37:04 AM	0	4/22/2018 6:37:04 AM	-9.405778885
18	GE09 2018-04-22 06:34:57	4/22/2018 6:34:58 AM	4/22/2018 8:32:28 AM	2	4/22/2018 8:32:28 AM	-0.377943575
19	GE08 2018-04-22 06:36:27	4/22/2018 6:36:28 AM	4/22/2018 6:46:27 AM	0	4/22/2018 6:46:27 AM	-4.683037758
20	GE03 2018-04-22 06:48:20	4/22/2018 6:48:21 AM	4/22/2018 8:25:51 AM	2	4/22/2018 8:25:51 AM	-6.972939014

- Step 4: Once the data has been imported, you should see the columns for the data set populated into the **Fields** tab on the right-hand side of your workspace. From here, we can proceed to work with our dataset.

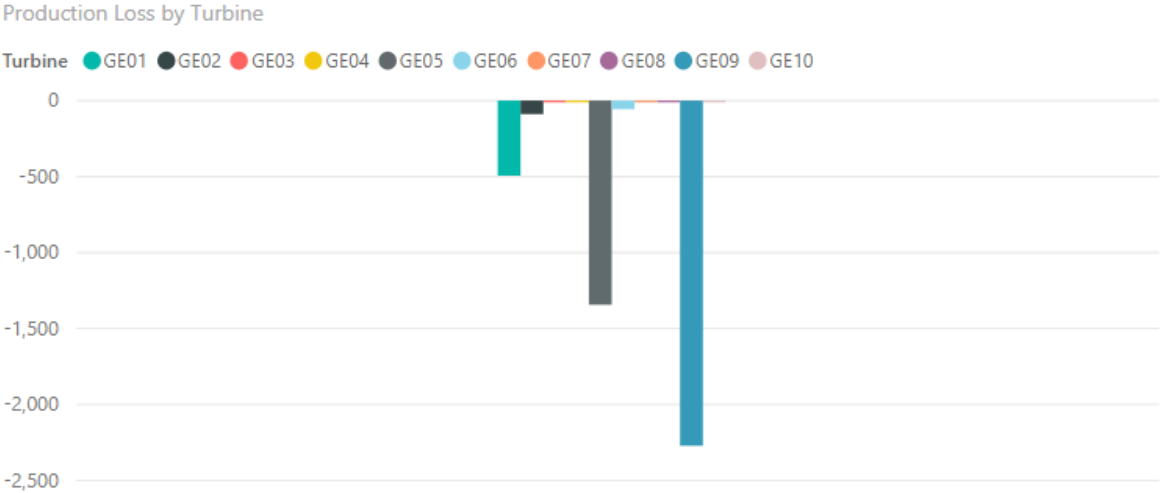
Production Loss by Turbine – Clustered Column Chart

The first thing we want to know about our curtailments is which of our turbines have experienced the greatest loss in energy production as a result of these curtailments. A great way to view this data is in the form of a **clustered column chart**.

- Step 1: Start by selecting the visualization symbol that we want to use. In this case, select the **Clustered Column** symbol (📊).
- Step 2: select value that we want to view – **Production Loss** – from the Fields section of the workspace.
- Step 3: Add the **Turbine** field into the **Legend** section.



The result will look like the graph below, which allows us to quickly and easily see which of our turbines have suffered from the greatest production loss due to curtailment:




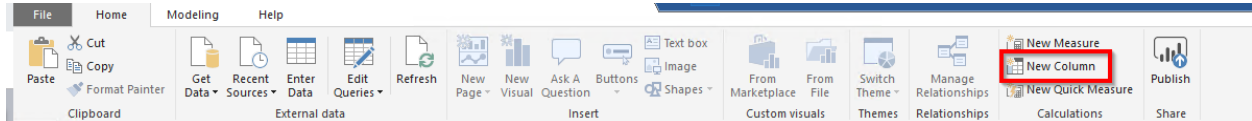
From this graph, we can see that turbine GE09 has suffered the greatest amount of production loss due to curtailments. But, one graph doesn't tell the whole story. We need to investigate these curtailments further to get a better idea of the trends behind these production losses.

Curtailment Statistics and PI Vision Links – Matrix

Our next task is going to be to get insights into the number and duration statistics of our curtailments.


Before we begin, we need to calculate the duration for each curtailment event. We can do this easily by making a new column in our data table and utilizing the **DATEDIFF()** function.

- Step 1: Click the **Data** tab  on the left side of your workspace, which will bring you to the data table we imported before. Here, on the **Home** tab in the **Calculations** section, select **New Column**.



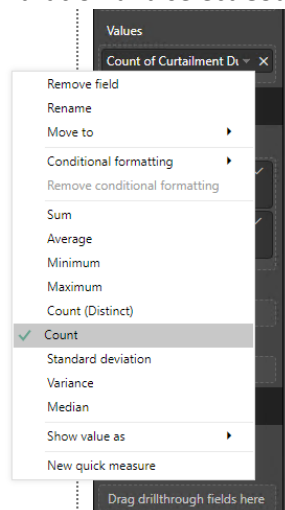
- Step 2: In the formula bar, enter in the following formula to generate a column which calculates the number of minutes each curtailment event has lasted:

Curtailment Duration = DATEDIFF([Event Frame Start Time],[Event Frame End Time],MINUTE)

- Step 3: Once this column has been populated, we can then go back to the **Report** tab on the left side of the workspace to create our matrix symbol.
- Step 4: Click a blank area of the report, then click the **Matrix** symbol . We want our matrix to contain one turbine per row, with the curtailment statistics taking up the columns. Thus, we will start by dragging the **Turbine** field into the **Rows** section of the visualization pane. Then, drag the **Curtailment Duration** field into the **Values** section.

Power BI allows us to do quick, ad hoc calculations on our fields when we bring them into a symbol. We will use this to highlight various statistics around our curtailments.

- Step5: To change the calculation applied to our value field, simply select the drop-down symbol on the field and choose a calculation. First, we will get the count of the curtailments, so click the drop down next to **Curtailment Duration** and select **Count**:



Our matrix now looks like this:

Turbine	Count of Curtailment Duration
GE01	10
GE02	5
GE03	4
GE04	4
GE05	10
GE06	7
GE07	3
GE08	9
GE09	26
GE10	4
Total	82

- Step 6: We then want to add columns for the average duration of the curtailments and the total duration of curtailment. We can follow the same sets of steps as above to add value fields for the **Average** and **Sum of Curtailment Duration**. After this, your matrix should look like this:

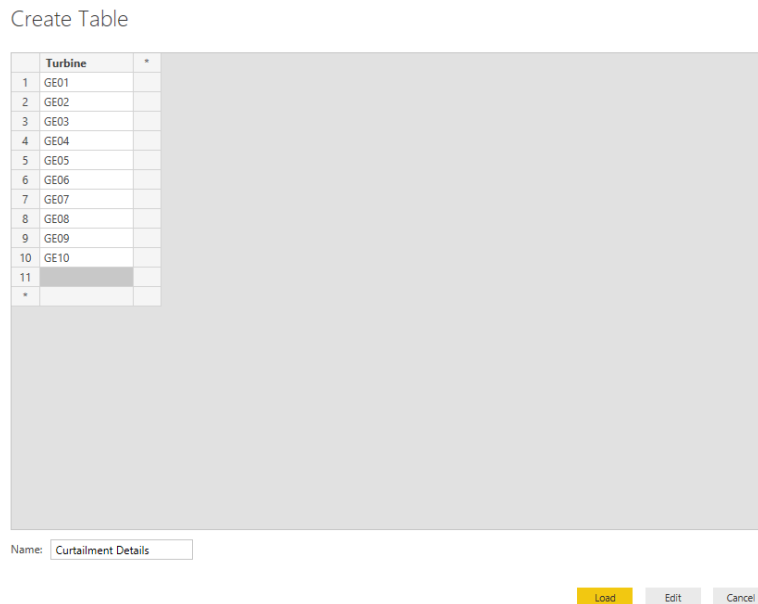
Turbine	Count of Curtailment Duration	Average of Curtailment Duration	Curtailment Duration
GE01	10	70.30	703
GE02	5	38.40	192
GE03	4	38.75	155
GE04	4	36.00	144
GE05	10	198.90	1989
GE06	7	32.43	227
GE07	3	30.33	91
GE08	9	25.89	233
GE09	26	102.96	2677
GE10	4	34.75	139
Total	82	79.88	6550

This table highlights that GE09 has experienced the most curtailments by far, and the curtailments for the turbine tend to be pretty long. However, the curtailments for GE05 are much longer in duration, which might indicate more significant issues.

While our table can tell us quite a bit, we may want to see more about what was going on during the periods of curtailment. In particular, we can leverage the power of PI Vision and event frames to see the values for our PI tags throughout our curtailments. Our next step is to build in a link to a PI Vision display highlighting the longest curtailment period for each turbine.

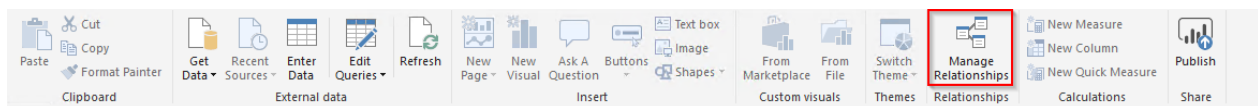
We will start this process by creating a related table for our main data table, to hold the information for the individual curtailments with the longest duration. This table will contain one row per turbine, along with the metadata information for the longest curtailment event frame for each. We will then construct a link to PI Vision from the information in this table.

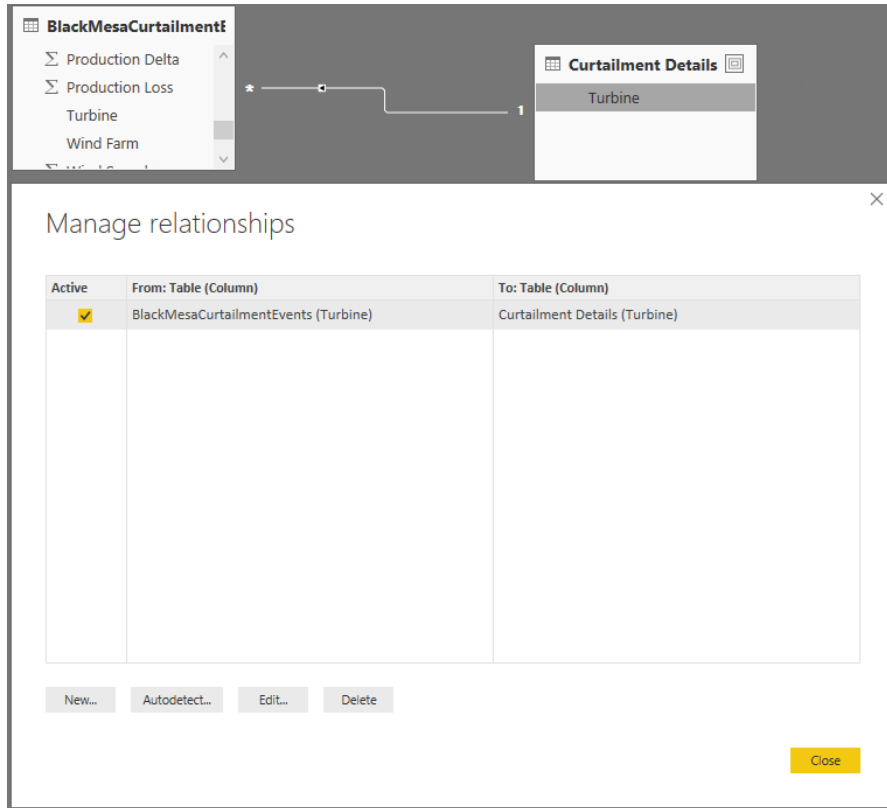
- Step 7: Go to the data tab. Then, in the External Data section of the home bar, select Enter Data. In the “Create Table” window that appears, change the name of Column1 to **Turbine**, and then in the rows beneath, enter the names of the 10 turbines we’re examining (GE01-GE10). Then name the table **Curtailment Details**. After this, your table should look like the picture below:



Based on the shared **Turbine** column name in both tables, Power BI will automatically create a relationship between the tables.

- Step 8: We can confirm that Power BI has done this by opening the **Manage Relationships** menu, and confirming that there is an active relationship present:





Our next step is to build out the Curtailment Details table with our curtailment information. The columns we're eventually going to have are:

- Longest Curtailment Duration
- Longest Curtailment Start
- Longest Curtailment End
- Curtailment Start Vision Formatted
- Curtailment End Vision Formatted
- Vision URL

The table will eventually look like the following:

Turbine	Longest Curtailment Duration	Longest Curtailment Start	Longest Curtailment End	Curtailment Start Vision Formatted	Curtailment End Vision Formatted	Vision URL
GE01	215	4/25/2018 12:13:32 AM	4/25/2018 3:48:17 AM	2018-04-25T00:13:32Z	2018-04-25T03:48:17Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE02	101	4/23/2018 3:28:03 PM	4/23/2018 5:09:16 PM	2018-04-23T15:28:03Z	2018-04-23T17:09:17Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE03	97	4/22/2018 6:48:20 AM	4/22/2018 8:25:51 AM	2018-04-22T06:48:21Z	2018-04-22T08:25:51Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE04	91	4/22/2018 6:58:50 AM	4/22/2018 8:29:21 AM	2018-04-22T06:58:51Z	2018-04-22T08:29:21Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE05	1741	4/25/2018 7:56:05 AM	4/26/2018 12:57:36 PM	2018-04-25T07:56:05Z	2018-04-26T12:57:36Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE06	85	4/22/2018 7:15:07 AM	4/22/2018 8:40:38 AM	2018-04-22T07:15:08Z	2018-04-22T08:40:38Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE07	66	4/22/2018 7:14:08 AM	4/22/2018 8:20:37 AM	2018-04-22T07:14:08Z	2018-04-22T08:20:37Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE08	82	4/22/2018 7:09:42 AM	4/22/2018 8:31:27 AM	2018-04-22T07:09:42Z	2018-04-22T08:31:27Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE09	914	4/25/2018 3:28:27 PM	4/26/2018 6:42:42 AM	2018-04-25T15:28:27Z	2018-04-26T06:42:43Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE10	78	4/22/2018 7:10:27 AM	4/22/2018 8:28:27 AM	2018-04-22T07:10:27Z	2018-04-22T08:28:27Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?

To find the longest curtailment duration for each turbine, we will need to leverage the **MAXX()** function in DAX.

- Step 9: First, we need to create a new column. From the Modeling tab, select New Column. Then, the function we will use is:

```
Longest Curtailment Duration =  
MAXX(RELATEDTABLE(BlackMesaCurtailmentEvents),BlackMesaCurtailmentEvents[Curtailment Duration])
```

This will find the maximum from the curtailment duration column in the **BlackMesaCurtailmentEvents** table, per turbine.



Tip

You can copy and paste the column definitions from this book directly to your Power BI tables (assuming your table and column names are identical).

- Step 10: Next, we need to find the start and end time of this longest curtailment duration. This is a little bit more involved, as we'll have to use a combination of the **CALCULATE()** and **FIRSTNONBLANK()** functions. The column definition we will use is below:

```
Longest Curtailment Start =  
CALCULATE(  
    FIRSTNONBLANK(BlackMesaCurtailmentEvents[Event Frame Start Time], 1 ),  
    FILTER (BlackMesaCurtailmentEvents, BlackMesaCurtailmentEvents[Turbine] =  
'Curtailment Details'[Turbine]),  
    FILTER (BlackMesaCurtailmentEvents,  
BlackMesaCurtailmentEvents[Curtailment Duration] = 'Curtailment  
Details'[Longest Curtailment Duration] )  
)
```

What this is doing is essentially filtering the **BlackMesaCurtailmentEvents** table to only have values for the turbine on each row, and then filtering that table to show only the duration in this table which is equivalent to the longest duration of curtailment. Then we retrieve the first non-blank Event Frame Start Time in this filtered table, which now only contains one row.

- Step 11: We then use a nearly identical definition for our next column, Longest Curtailment End. The only thing that changes is that we look for the first non-blank of the **Event Frame End Time**, rather than the first non-blank of the **Event Frame Start Time**, as before.

```
Longest Curtailment End =
CALCULATE(
    FIRSTNONBLANK(BlackMesaCurtailmentEvents[Event Frame End Time], 1 ),
    FILTER (BlackMesaCurtailmentEvents, BlackMesaCurtailmentEvents[Turbine] =
'Curtailment Details'[Turbine]),
    FILTER (BlackMesaCurtailmentEvents,
BlackMesaCurtailmentEvents[Curtailment Duration] = 'Curtailment
Details'[Longest Curtailment Duration] )
)
```

- Step 12: Now that we have our start and end times, we need to format these for a PI Vision URL. To do this we'll use the **Format()** function. For the **Curtailment Start Vision Formatted** column, the configuration is:

```
Curtailment Start Vision Formatted = FORMAT('Curtailment Details'[Longest
Curtailment Start],"yyyy-mm-ddThh:mm:ssZ")
```

- Step 13: The **Curtailment End Vision Formatted** column is almost the same:

```
Curtailment End Vision Formatted = FORMAT('Curtailment Details'[Longest
Curtailment End],"yyyy-mm-ddThh:mm:ssZ")
```

Finally, we can make a link to our PI Vision display. For this exercise, a PI Vision display has already been created for you, and we'll be linking to the time of our longest curtailment on that display. The URL for the display is <https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay> . Utilizing PI Vision's URL parameters, we can tack on a couple of strings to this URL to make it link to a specific asset and time range. The basic format of a URL with these components is:

```
URL + "?Asset=<PATH TO ASSET> + "&StartTime=<START TIME> + "&EndTime=<END TIME>"
```

- Step 14: In practice, for the column in our table, it should look like this:

```
Vision URL =
"https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay" &
"?Asset=\\PISRv01\Wind Power Generation\Wind Power Generation Fleet\Black
Mesa Wind Farm\" & [Turbine] & "&StartTime=" & [Curtailment Start Vision
Formatted] & "&EndTime=" & [Curtailment End Vision Formatted]
```

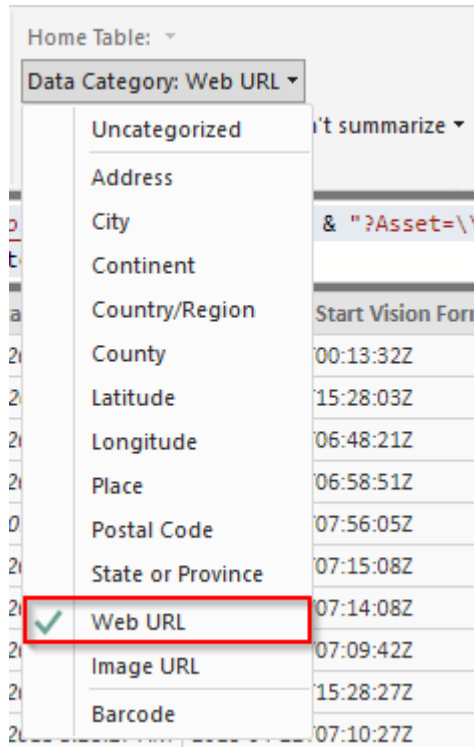


Tip

More information about PI Vision URL parameters can be found in the **Live Library** : <https://livelibrary.osisoft.com/LiveLibrary/content/en/vision-v2/GUID-C643F092-EB07-41EC-8DC8-5981BF2692F4>

Exercise 3 – Curtailment Report

- Step 15: The last thing to do with this table is to make sure that our Vision URL column is recognized by Power BI as URLs. To do this, select the Vision URL column, and then in the Properties Section of the Modeling tab, click the Data Category dropdown and choose **Web URL**:



With this completed, the table you have put together should look like the picture below:

Turbine	Longest Curtailment Duration	Longest Curtailment Start	Longest Curtailment End	Curtailment Start Vision Formatted	Curtailment End Vision Formatted	Vision URL
GE01	215	4/25/2018 12:13:32 AM	4/25/2018 3:48:17 AM	2018-04-25T00:13:32Z	2018-04-25T03:48:17Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE02	101	4/23/2018 3:28:03 PM	4/23/2018 5:09:16 PM	2018-04-23T15:28:03Z	2018-04-23T17:09:17Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE03	97	4/22/2018 6:48:20 AM	4/22/2018 8:25:51 AM	2018-04-22T06:48:21Z	2018-04-22T08:25:51Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE04	91	4/22/2018 6:58:50 AM	4/22/2018 8:29:21 AM	2018-04-22T06:58:51Z	2018-04-22T08:29:21Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE05	1741	4/25/2018 7:56:05 AM	4/26/2018 12:57:36 PM	2018-04-25T07:56:05Z	2018-04-26T12:57:36Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE06	85	4/22/2018 7:15:07 AM	4/22/2018 8:40:38 AM	2018-04-22T07:15:08Z	2018-04-22T08:40:38Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE07	66	4/22/2018 7:14:08 AM	4/22/2018 8:20:37 AM	2018-04-22T07:14:08Z	2018-04-22T08:20:37Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE08	82	4/22/2018 7:09:42 AM	4/22/2018 8:31:27 AM	2018-04-22T07:09:42Z	2018-04-22T08:31:27Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE09	914	4/25/2018 3:28:27 PM	4/26/2018 6:42:42 AM	2018-04-25T15:28:27Z	2018-04-26T06:42:43Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?
GE10	78	4/22/2018 7:10:27 AM	4/22/2018 8:28:27 AM	2018-04-22T07:10:27Z	2018-04-22T08:28:27Z	https://pisrv01.pischool.int/PIVision/#/Displays/3/TurbineDisplay?

Now, we need to associate this link with our larger data table, so we can include the link as a column in our report.

- Step 16: To do this, navigate back to the **BlackMesaCurtailmentEvents** table, and add a column with the following configuration:

Vision URL = LOOKUPVALUE('Curtailment Details'[Vision URL], [Turbine], BlackMesaCurtailmentEvents[Turbine])

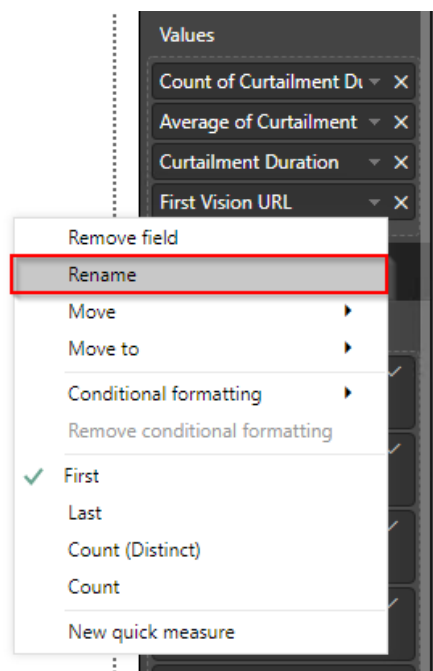
This column uses the **Lookupvalue()** method, which allows us to find a value in another table based on a value in our current table. For us, this means finding the Vision URL from the **Curtailment Details** table, where the **Turbine** value in our row matches the **Turbine** value in the row in the other table.

Once more, we need to make sure that Power BI recognizes these values as Web URLs.


- Step 17: Repeat the steps that we carried out for the **Curtailment Details** table to have this column recognized as Web URLs. The column should look just like the one from the **Curtailment Details** table.

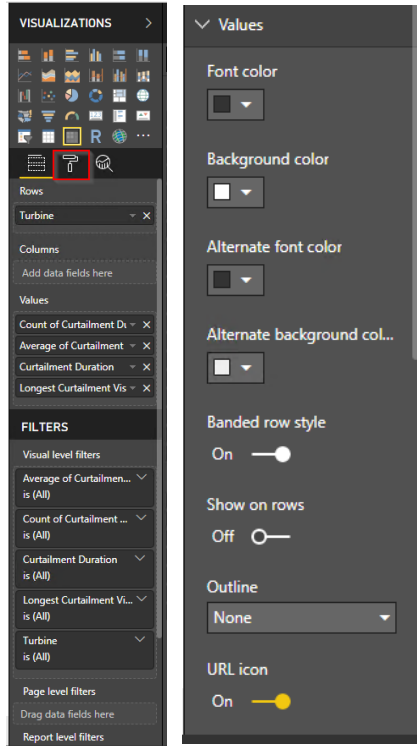
Finally, we can add this URL to our matrix. Return to the Report tab of our Power BI workspace.

- Step 18: Click the matrix we made earlier and drag **Vision URL** from the **BlackMesaCurtailmentEvents** table into the Values section of the visualization. By default, Power BI will want to call this column “First Vision URL”. But we know that this is not accurate. Luckily, Power BI allows us to name our columns manually. Simply click the dropdown on the “First Vision URL” field and select Rename. Rename this field to “Longest Curtailment Vision Display”.














Our matrix is almost done! The last thing we have to do is to change our URLs to symbols, so they don't take up so much space in the matrix.

- Step 19: We can do that by first selecting our matrix, and then clicking the **Format** tab () of the Visualizations pane. In the **Values** section, turn the **URL icon option** to **On**:



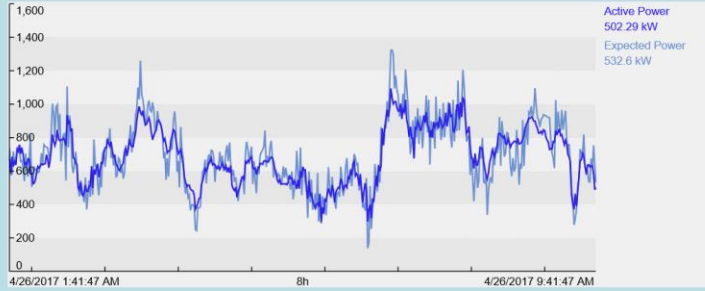
Finally, your matrix should look like the picture below. Click the link icon for each row to be taken to a display for each turbine during their longest curtailment period!

Turbine	Count of Curtailment Duration	Average of Curtailment Duration	Curtailment Duration	Longest Curtailment Vision Displa
GE01	10	70.30	703	
GE02	5	38.40	192	
GE03	4	38.75	155	
GE04	4	36.00	144	
GE05	10	198.90	1989	
GE06	7	32.43	227	
GE07	3	30.33	91	
GE08	9	25.89	233	
GE09	26	102.96	2677	
GE10	4	34.75	139	
Total	82	79.88	6550	

WTG01

Daily Generation: 2121.01 kWh Capacity: 34.334 % State: No Data Operating Efficiency: 101.09 % Overheat Alarm: ●

Turbine Details
Manufacturer
GE
Model
1.5 csCWE
Power Rated
1,500 kW



Wind Curtailment vs Wind Speed and Hour of Day – Line Chart

One common reason that wind turbines are curtailed is due to high wind speeds. If the wind speeds would cause the turbine to generate power beyond its rated capacity, it can cause significant damage to the turbine motor.

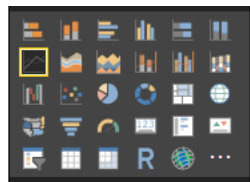
To confirm that's what's happening with our curtailments, we'll make a line plot to correlate between the average wind speed and the number of curtailments. As an added point of analysis, we'll compare both of these parameters against the hour of day, to see if our curtailments are more frequent at a certain time of day.

To begin, we will need to add another column to our **BlackMesaCurtailmentEvents** table to record the hour of the day for the start time of each of our curtailments. As before, go to the data tab of your Power BI workspace, and select New Column. The definition of the column should be the following:

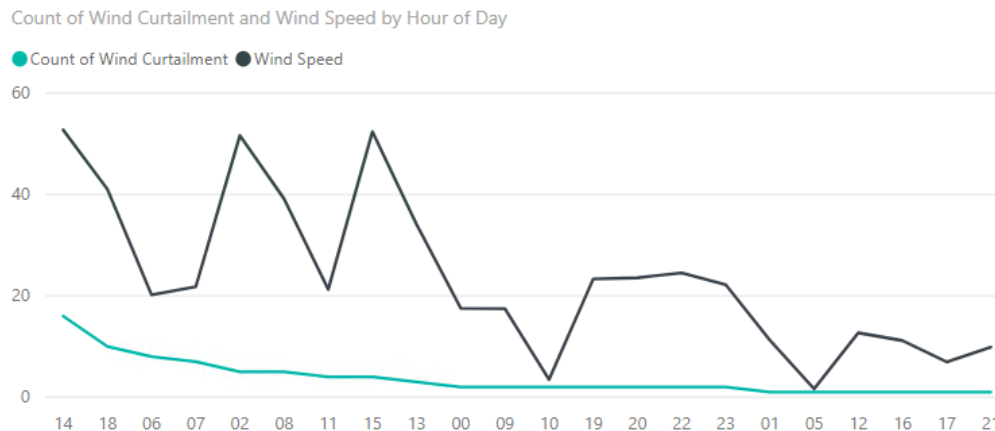
Hour of Day = FORMAT([Event Frame Start Time], "hh")

This function takes the **Event Frame Start Time** column, a timestamp, and formats it as a two digit hour code.

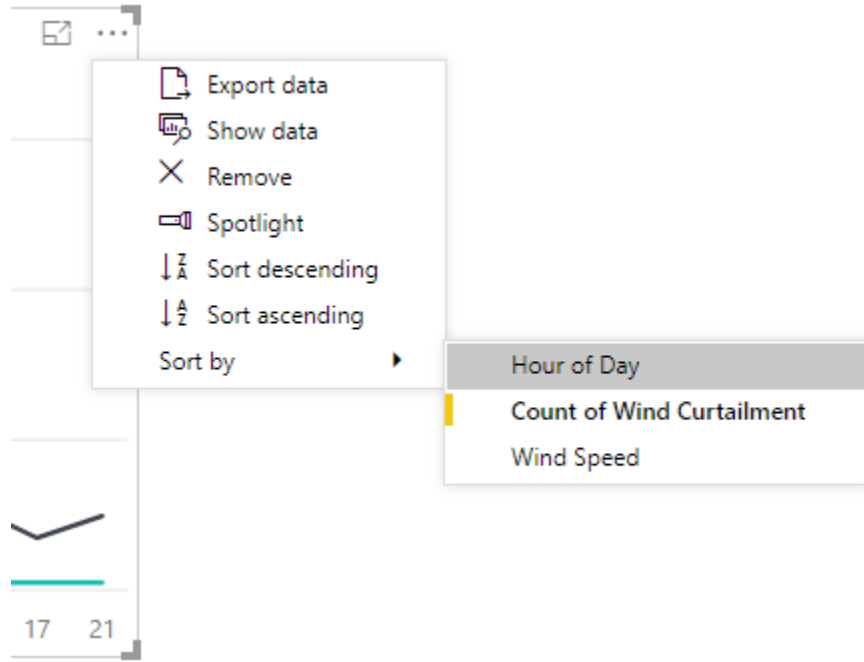
With this column in place, we can now make our line plot. Return to the Report tab of your Power BI workspace, and click a blank area of the report. Then, select the **Line Chart** symbol to make a new line chart:



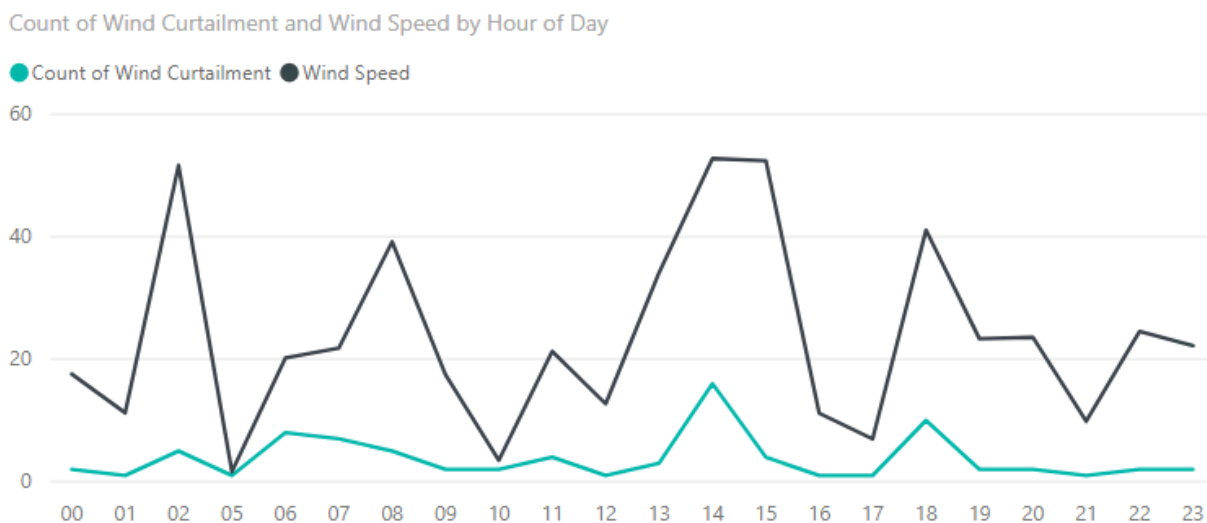
Then, to add the appropriate data to the chart, drag the **Hour of Day** field to the Axis section, **Wind Curtailment** to the **Values** section, and **Wind Speed** to the **Values** section. You'll notice that **Wind Curtailment** automatically becomes **Count of wind curtailment** – which is something that we want. However, you'll also notice that the line chart is sorted by most curtailments, so our Hour of Day axis is out of order:



To fix this, we need to sort the line chart by hour of day, as well as sort ascending. To do this, click the ellipses (...) symbol on the top right of the chart, then select **Sort by** and choose **Hour of Day**. Then, click the ... symbol again and select **Sort ascending**.



Doing these sorts will lead to a line chart that looks like this:

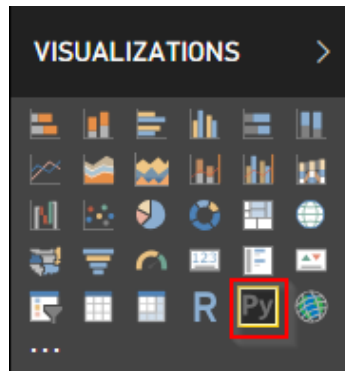


In this plot, we can clearly see that the higher the wind speed, the more curtailments that occur. This is in line with our expectations. As well, we can see that time of day doesn't seem to have a very noticeable correlation with wind speeds or curtailments.

Operational Efficiencies – Violin Plot

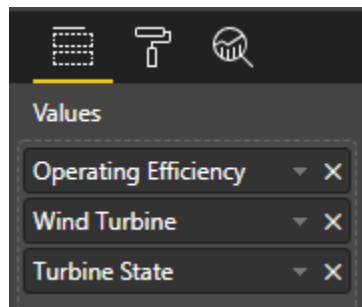
Let's examine the operational efficiencies of each of our wind turbines in the wind farm. We will use a custom symbol to view the efficiencies using python and a library in python called seaborn.

- Step 1: Click on an empty white space on the report and select **python visual** from the **visualizations** pane:



This will open the python script editor and allow us to enter our script and the data we are interested in.

- Step 2: Enter **Operating Efficiency (Don't summarize)**, **Wind Turbine**, and **Turbine state** to the **values** field from the **BlackMesaWindFarm** table:



You will notice that the python script editor has created a dataset (we will talk about this in the appendix), and it gives us a place to write our script in the bottom:

Python script editor

⚠ Duplicate rows will be removed from the data.

```
# Create dataframe
# dataset = pandas.DataFrame(Operating Efficiency, Wind Turbine, Turbine State)

# Remove duplicated rows
# dataset = dataset.drop_duplicates()
Paste or type your script code here
```

- Step 3: Paste the following script in the python editor:

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

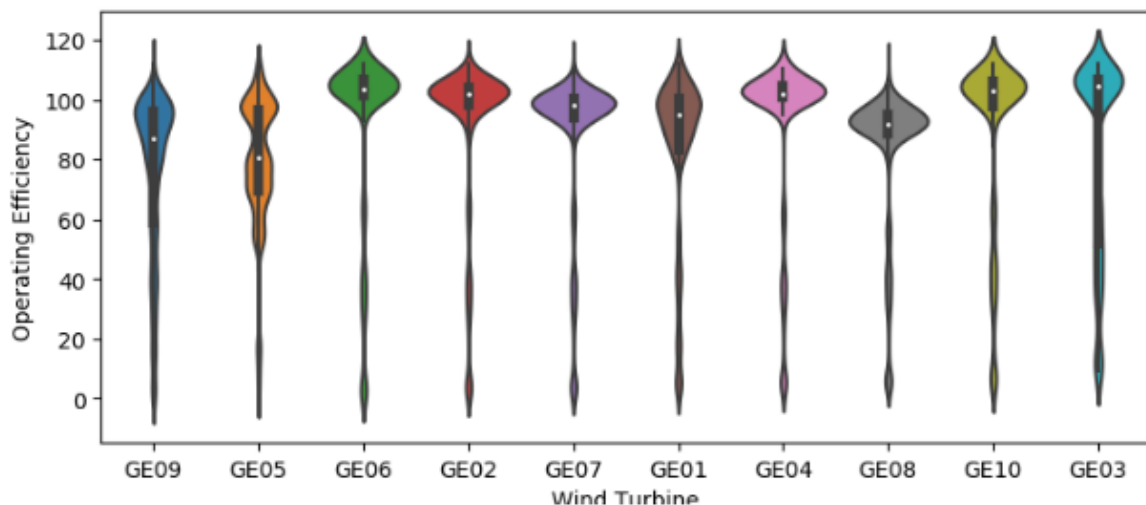
df = dataset
df1 = df.loc[df['Turbine State'] == 'Load Operation']

sns.violinplot(x="Wind Turbine", y="Operating Efficiency", data=df1)
plt.show()
```

You can see we are importing some libraries such as seaborn which has the actual violinplot visual. Also note that we are filtering our dataset to only include rows where the Turbine state is equal to Load Operation. Essentially, we only want to visualize operating efficiencies when the turbines are actually running. Then we simply just call seaborn violin plot and specify our x and y axis.

The violin plot should look something like this:

Operating Efficiency, Wind Turbine and Turbine State



We can see that most turbines have a good spread (more data points) at around 100 while GE09 and GE05 have significantly more at lower efficiencies.

Discussion: Cross reference this visual with our other curtailment visuals, does this make sense?

Takeaways

You should now feel comfortable with the following Power BI features:

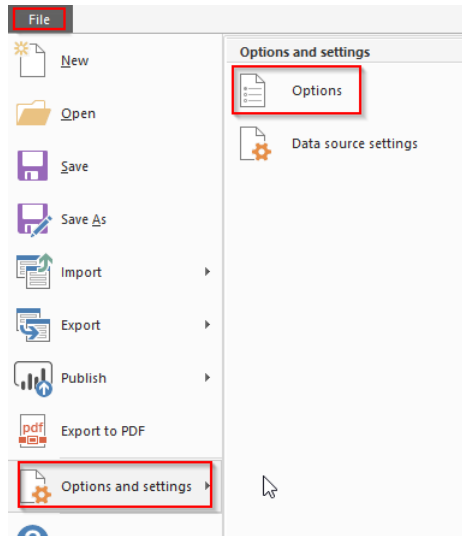
- Using event frames with Power BI
- Leveraging existing PI Vision displays with Power BI reports
- DAX formulas to bring data from one table to another
- Bringing in Python scripts and libraries to create custom symbols

Bringing event frame data into Power BI can reveal a lot of actionable information. With the visualization we put together for this exercise, what might you investigate? Why are certain turbines performing worse than their peers? Is that performance justified?

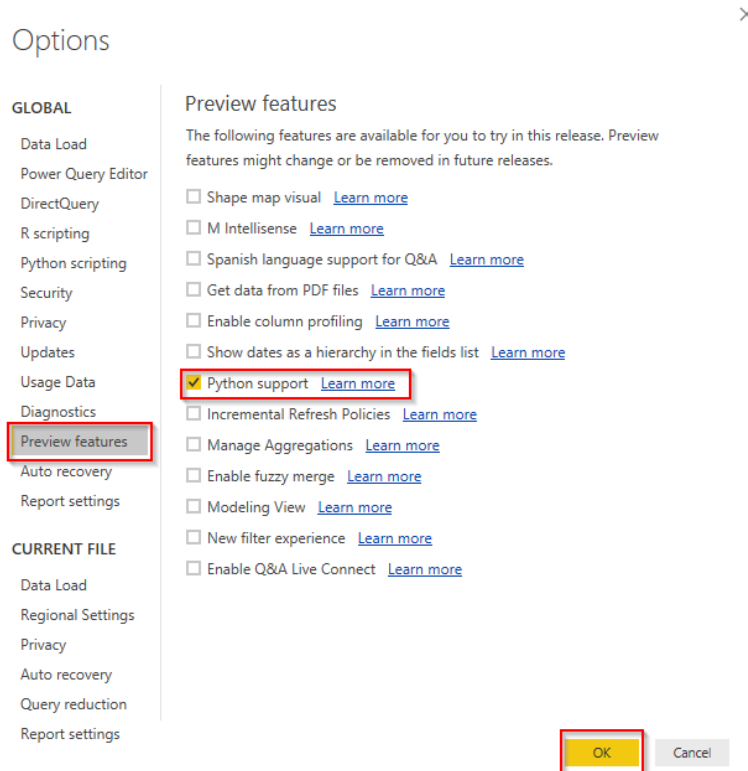
Of course, these exercises only cover a portion of what can be done with event frames in Power BI. The symbols and analyses that you use will be very dependent upon the kind of data your event frames capture. Additionally, the ability to bring in Python scripts and visuals into your report opens up a world of possibilities for further data analysis. What Python libraries have you used to analyze data in the past? Could that be valuable to use with your PI data in a report like this one?

Appendix – Python Scripting in Power BI

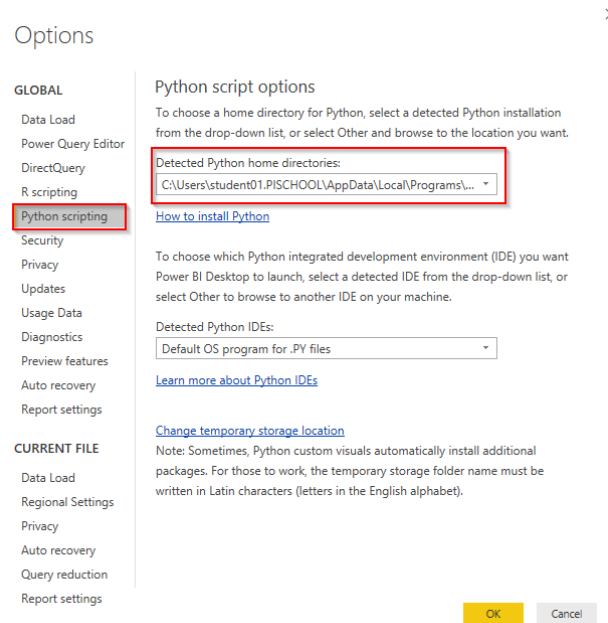
- Step 1: In order to enable python visuals and scripts in Power BI, we need to have the python interpreter installed on machine. This can be installed from the python website: <https://www.python.org/downloads/>
- Step 2: Next, we need to enable python support in Power BI. This can be done from the file menu by selecting options and settings:



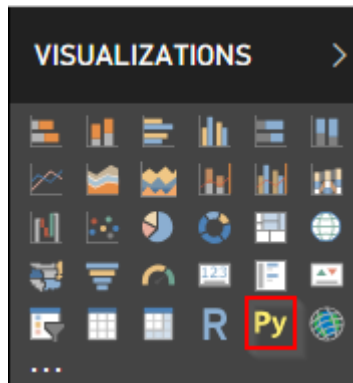
- Step 3: Select python support from the preview features:



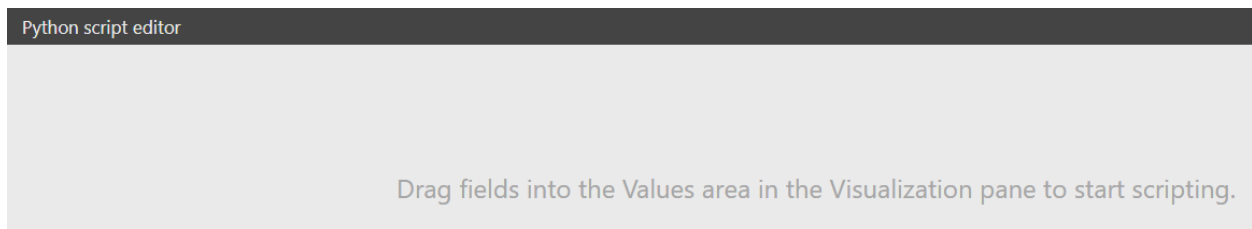
- Step 4: If you have the python interpreter installed then it should auto detect it. If it is unable to, you can select it from the python scripting pane:



- Step 5: Now we should see a python visual in the visualizations pane, go ahead and select it:

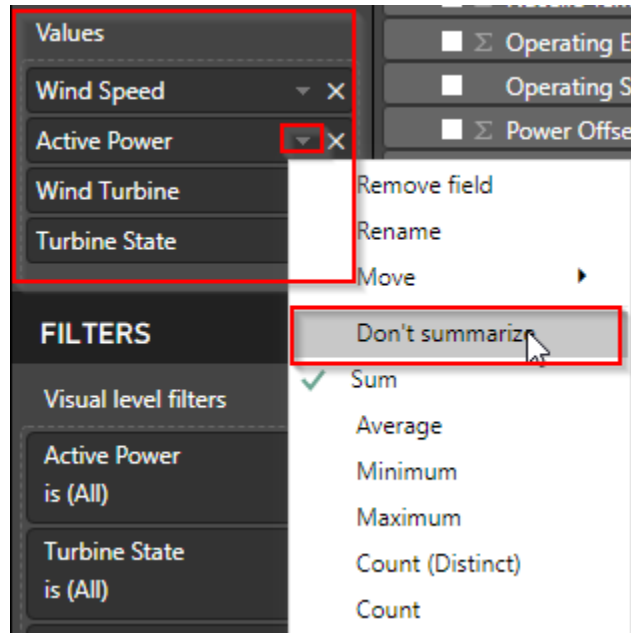


- Step 6: You should get a pop-up which allows you enable script visuals for your report, click enable and you should see the python script editor opens.



Before we can start writing python scripts, we need to create a dataframe, which is essentially the data we will be using in the script.

- Step 7: Drag Wind Speed, Active Power, Wind Turbine, and Turbine State to the values field:



- Step 8: Make sure to select don't summarize for all of the values, we want the entire dataset, i.e. all the rows and not a sum of the rows.

Note the following line: `#dataset = pandas.DataFrame(Wind Speed, Active Power, Wind Turbine, Turbine State)`

This is creating a pandas dataframe, which is a tabular data structure with the specified columns.



Tip

Reference for the pandas dataframe can be found here:

<https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html>

Now that we have our table, in this case called dataset to work with, we can go about creating the power curve we did in exercise 2.

- Step 9: Take a look at the following script:

```
import matplotlib.pyplot as plt
import pandas as pd

df = dataset
turbines = df['Wind Turbine'].unique()

df['clean'] = 1

for i in df.index:
    if (df.at[i, 'Active Power'] < 600) and (df.at[i, 'Wind Speed'] >12):
        df.at[i, 'clean'] = 0
    if df.at[i, 'Turbine State'] != "Load Operation":
        df.at[i, 'clean'] = 0
    if (df.at[i, 'Active Power'] < 300) and (df.at[i, 'Wind Speed'] >9):
        df.at[i, 'clean'] = 0

df1 = df[df['clean'] == 1]

fig, ax = plt.subplots()
for turbine in turbines:
    df2=df1[df1['Wind Turbine'] == turbine]
    ax.scatter(df2['Wind Speed'], df2['Active Power'], label=turbine)
ax.legend()
plt.show()
```

Let's go over this script. You can see that we import two libraries at the top, matplotlib and pandas. We need pandas to create the DataFrame and matplotlib to call our plot function.

```
import matplotlib.pyplot as plt
import pandas as pd
```

Here we set our variable df to our dataset which contains the columns and rows for wind speed etc.

```
df = dataset
```

Now we create a variable called turbines which has all of the unique values under the wind turbine column. So this variable should store 10 value, the name of each wind turbine.

```
turbines = df['Wind Turbine'].unique()
```

Now we can create a new column in our dataset called clean via the tablename['column name'] notation. We will set all rows for this column as 1.

```
df['clean'] = 1
```


We will use the for loop to iterate over the index (all the rows) of our dataframe (table) and if the active power is less than 600 and wind speed is greater than 12 we will set the value of our clean column to 0. This is to eliminate periods where the turbine is performing sub-optimally for any reason. Same with when the turbine state is not load operation (running) and the power is less than 300 at a wind speed of greater than 9.

```
for i in df.index:
    if (df.at[i, 'Active Power'] < 600) and (df.at[i, 'Wind Speed'] >12):
        df.at[i, 'clean'] = 0
    if df.at[i, 'Turbine State'] != "Load Operation":
        df.at[i, 'clean'] = 0
    if (df.at[i, 'Active Power'] < 300) and (df.at[i, 'Wind Speed'] >9):
        df.at[i, 'clean'] = 0
```

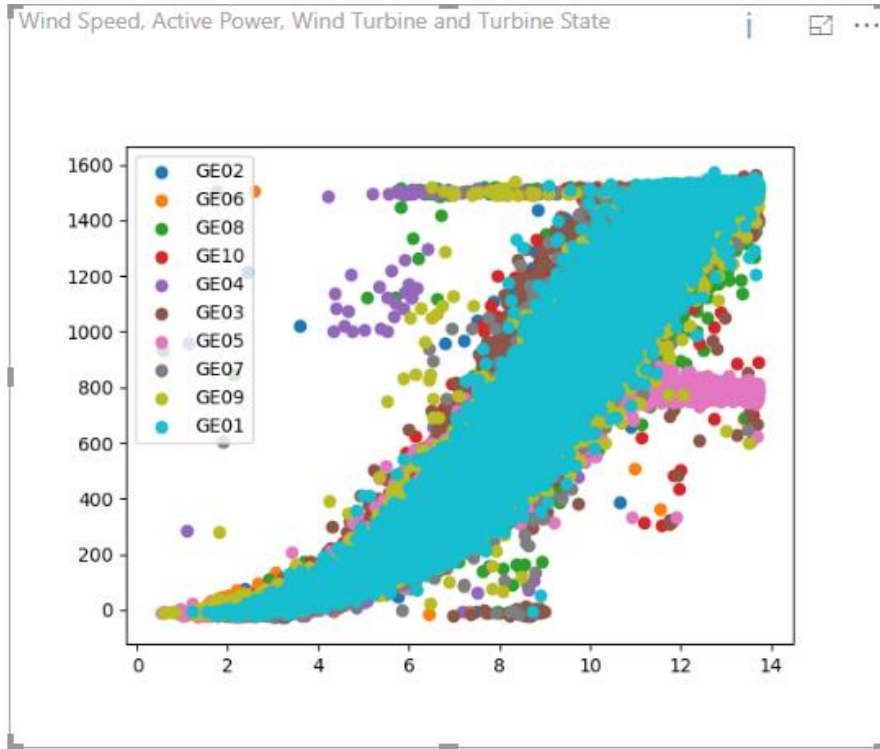
Now we will create a new dataframe (table) called df1 where we only take rows where the clean column has a value of 1, i.e. the filtered data is thrown out

```
df1 = df[df['clean'] == 1]
```

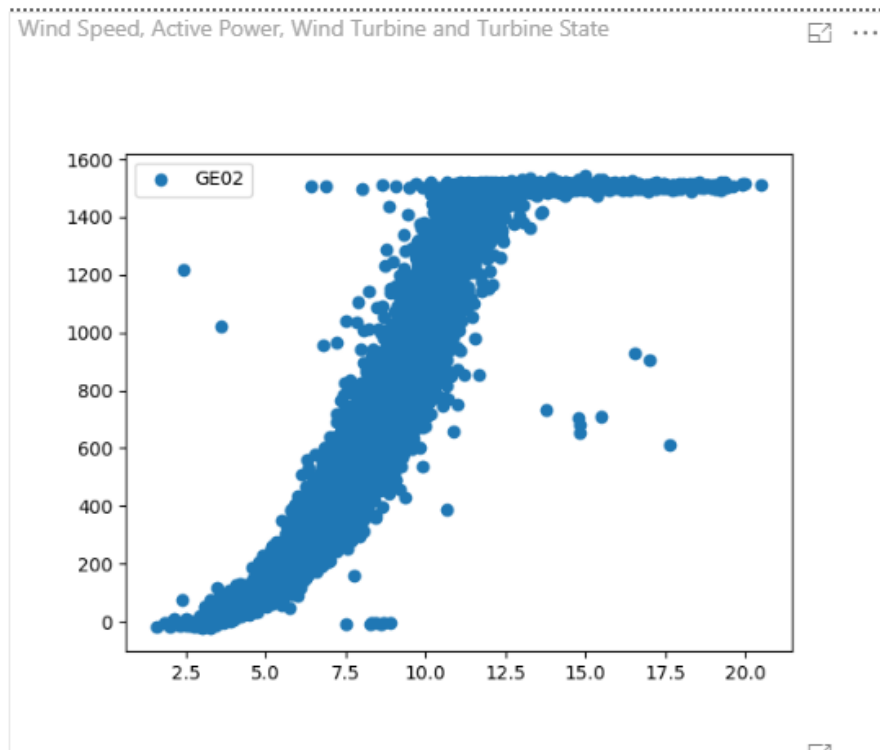
Next we call plt.subplots to create a plot for each wind turbine (our turbines variable), and we only want to plot the data for that wind turbine so we create a new dataframe (df2) which only contains rows for the wind turbine we are currently iterating through, and the plot we call is a scatter plot (ax.scatter) and give our x and y axis.

```
fig, ax = plt.subplots()
for turbine in turbines:
    df2=df1[df1['Wind Turbine'] == turbine]
    ax.scatter(df2['Wind Speed'], df2['Active Power'], label=turbine)
ax.legend()
plt.show()
```

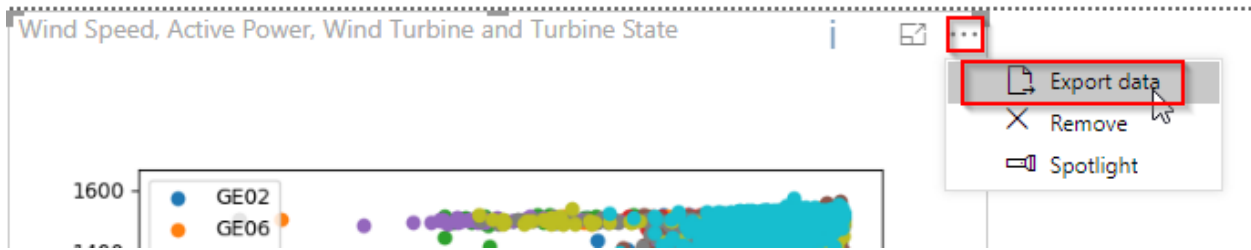
Now you should have a plot which looks like a more traditional power curve after the data has been filtered.



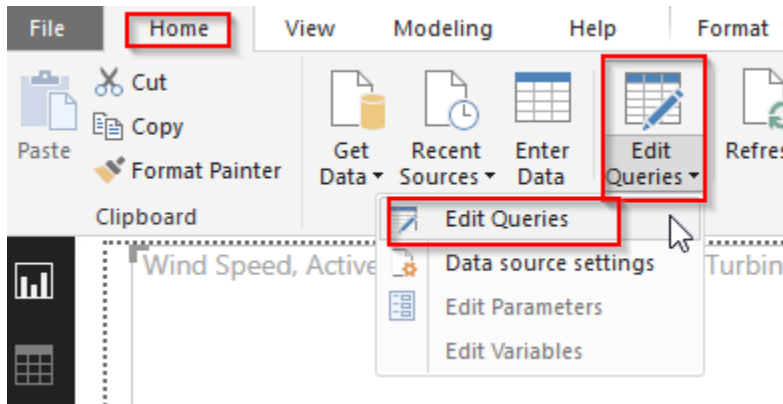
- Step 10: We can splice this to view data for just 1 wind turbine (GE02) and you can see the power curve is much clearer:



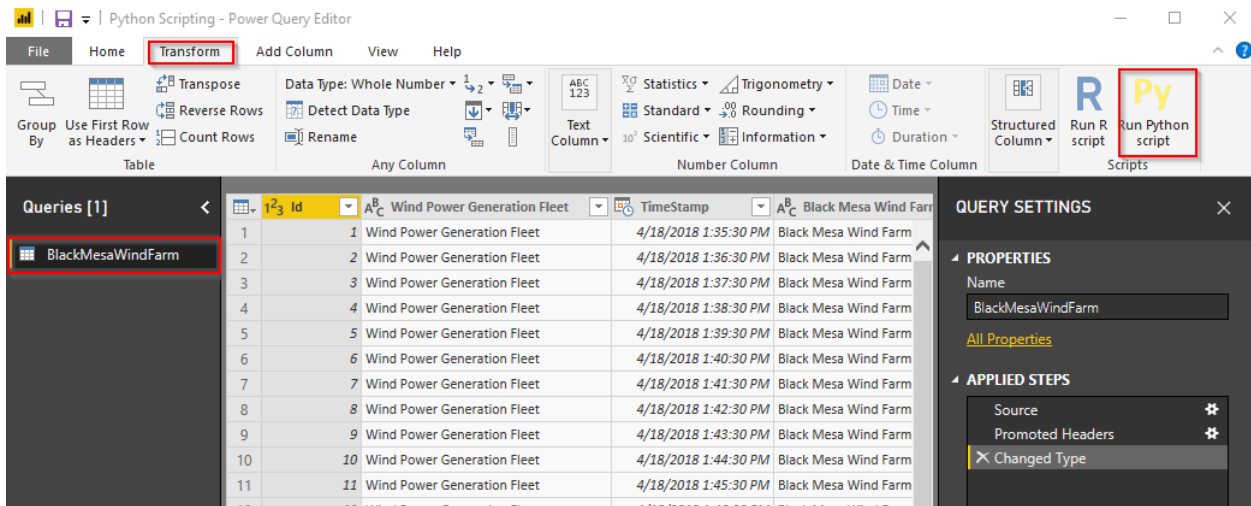
- Step 11: Now we can export the dataframe we just created:



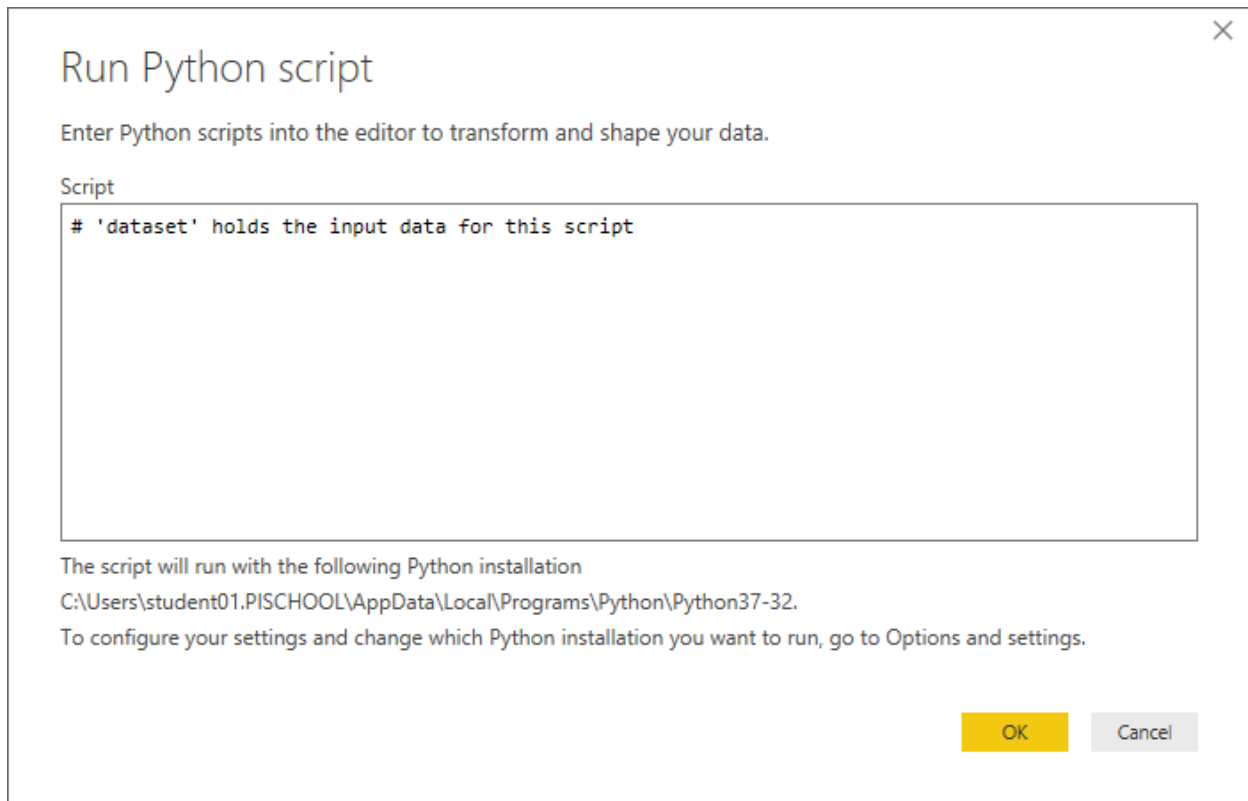
- Step 12: Save it to your desired location and open up the CSV, you should see your new table. This allows use to create and modify large datasets directly in Power BI and create new datasets to our liking.
- Step 13: Alternatively, we can modify dataset before we enter them into Power BI. From the home menu we can select edit queries:



- Step 14: This will open the Power Query Editor where we can select transform and run python script:



Here you can type the python script just like we did in the python visual editor:



Note that it creates a dataframe initially for us called **dataset** which has all of the columns from our **BlackMesaWindFarm** and now we can reference that dataset and filter it how we like to create and modify the **BlackMesaWindFarm** table.



Have an idea how to
improve our products?

**OSIsoft wants to hear
from you!**

<https://feedback.osisoft.com/>





Save the Date!

OSIsoft PI World Users Conference in Gothenburg, Sweden. September 16-19, 2019.

Register your interest now to receive updates and notification early bird registration opening.

https://pages.osisoft.com/UC-EMEA-Q3-19-PIWorldGBG-RegisterYourInterest_RegisterYourInterest-LP.html?_ga=2.20661553.86037572.1539782043-591736536.1533567354

