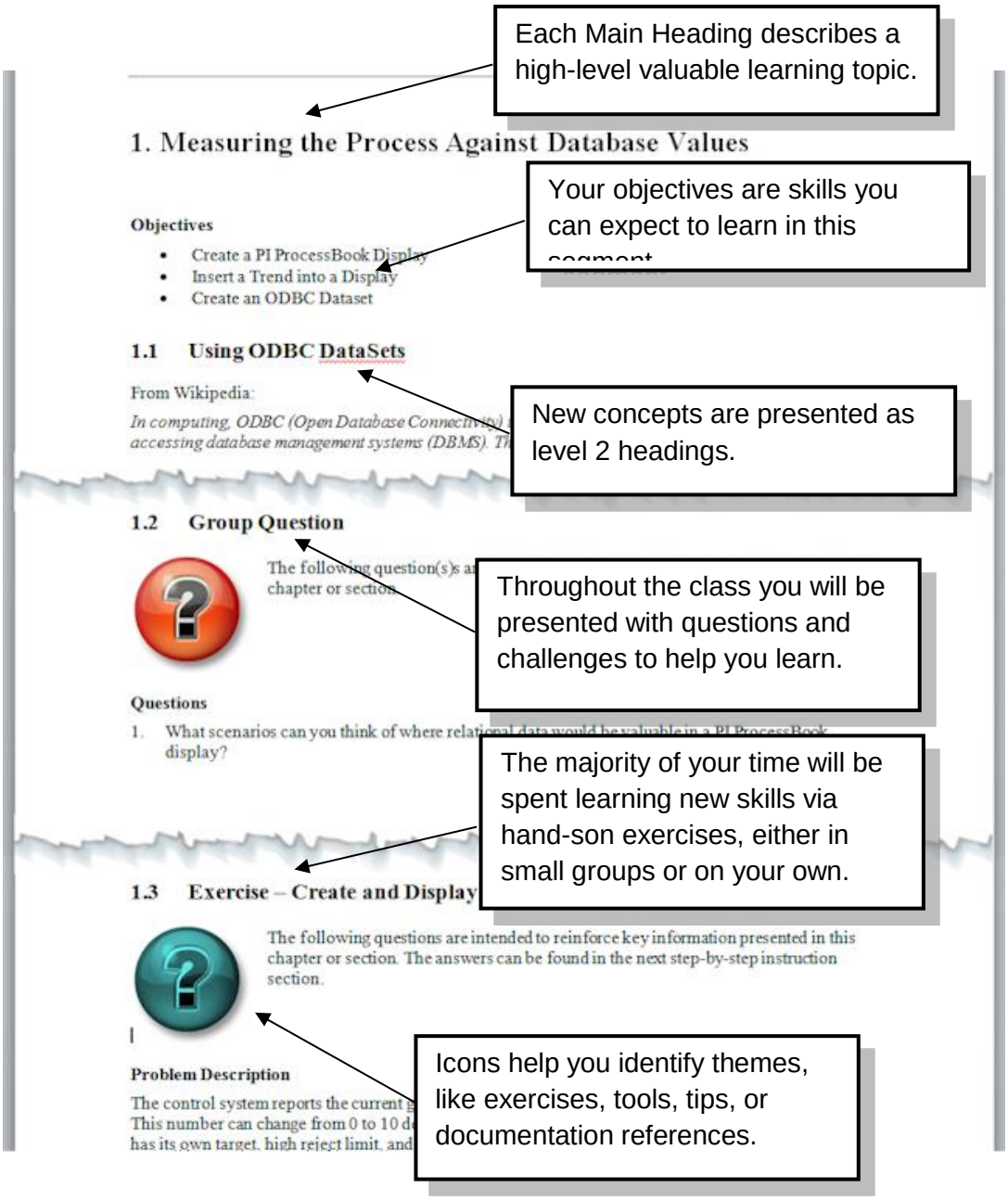
Analyzing PI System Data

Version 2018

How to Use this Workbook



User manuals, Learning workbooks, and other materials used in class can be downloaded from http://techsupport.osisoft.com. Login to an OSIsoft technical support account is required.

Software Versions Used in this Document

The list below describes the software versions used in this version of the course.

Software	Version
PI DataLink	2017 SP2
Microsoft Office	2016
PI ODBC Driver	2016 R2
PI Integrator for Business Analytics Advanced Edition	2018
PI OLEDB Enterprise	2017 R2
Microsoft SQL Server	2014
PI Data Archive	2018
PI Asset Framework	2018
PI Vision	2017 R2 SP1

Contents

1 Welcome

Welcome to the Analyzing PI System Data Course!

Since you are attending this class, you should have some experience with OSIsoft Client Tools (PI ProcessBook, PI DataLink, PI WebParts and PI Vision), either using displays, reports or webpages previously created to analyze your data, or creating these displays, reports and webpages so that others in your organization have access to all the powerful data that resides in the Data Archive and data external to the PI System.

The basic tasks within these tools are presumed to be understood; what you will experience here can be seen as a factory of ideas, a space for OSIsoft customers to realize how powerful existing data can be when analyzed with the advanced options of our tools and additional third party tools, and integrated with non-PI data.

Hope you enjoy!

1.1 Course Environment

The environment for this course is being hosted with Azure. The environment has 3 VM and contain the following:

- PIDC – Domain Controller
- PISRV01 – The server environment
    - Microsoft SQL Server 2014
    - PI Data Archive 2018
    - PI AF Server 2018
    - PI AF Client 2018
    - PI Vision 2017 R2 SP1
    - PI Integrator for Business Analytics Advanced Edition 2018
- PICLIENT01 – This is the primary working environment.
    - PI System Explorer 2018
    - Microsoft Office Professional Plus 2016 (64-bit)
    - Microsoft PowerBI Desktop 2.59.5135.781
    - Google Chrome 68.0.3440.106

The userid for each student is pischool\student01, password will be provided by the instructor.

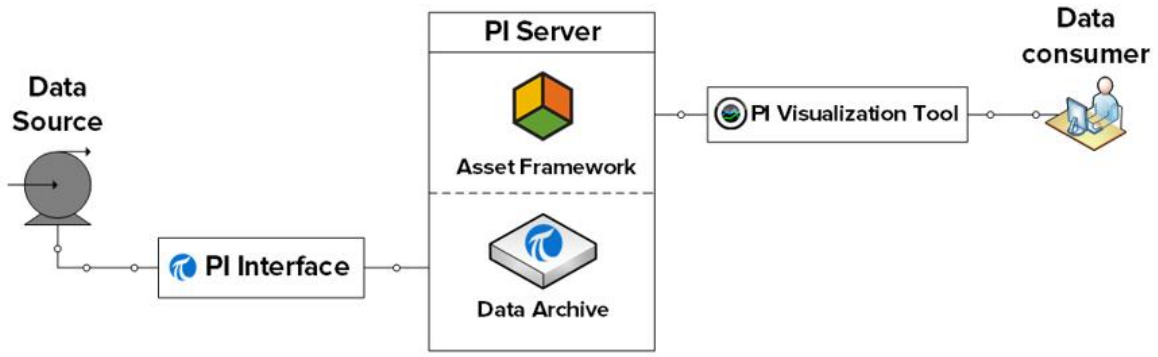
1.2 Review PI System Architecture

Objectives:

- Define the components of a PI System
- Draw a diagram of the architecture of a PI System

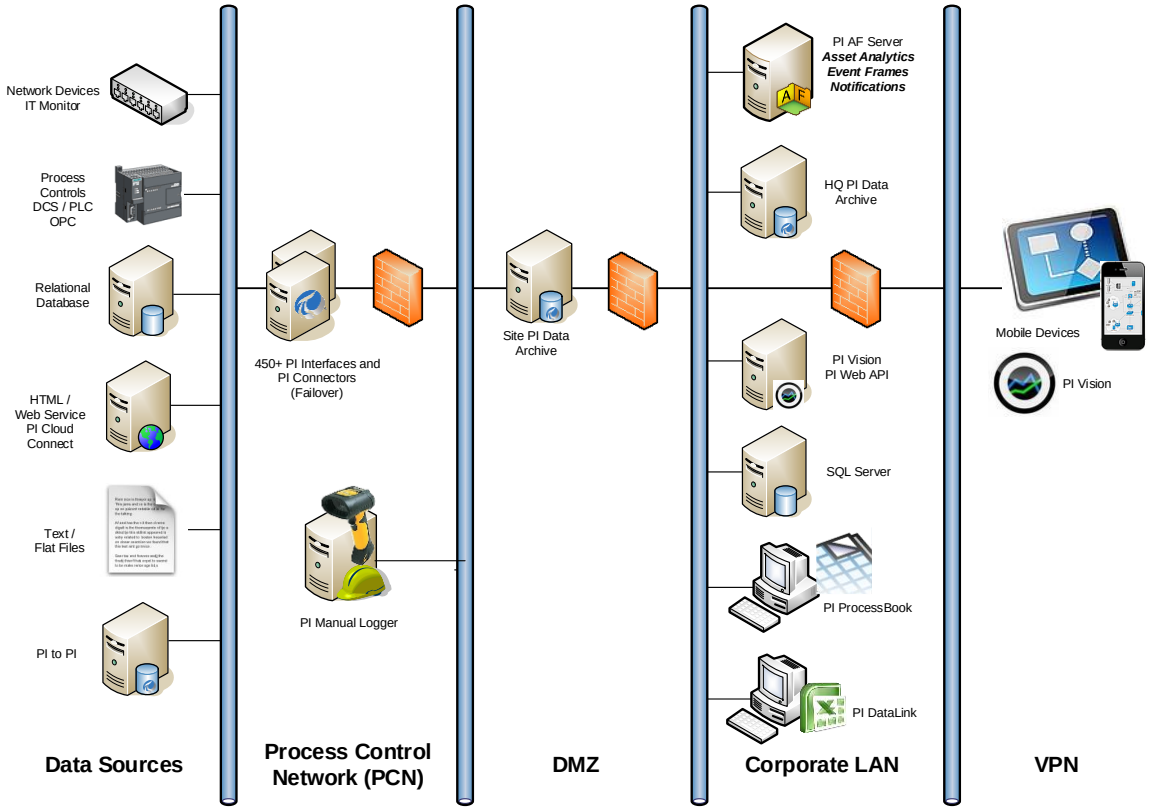
1.2.1 The PI System Described

The PI System collects, stores, and manages data from your plant or process. You connect your data sources to one or more PI Interface nodes. The Interface Nodes get the data from your data sources (control systems, instrumentation, etc) and send it to the Data Archive. The data is organized and given context using PI Asset Framework. Users get data from the Data Archive and Asset Framework and work with it using a variety of client tools, such as PI Vision and PI DataLink.



1.2.2 Architecture of a Typical PI System

Sometimes the architecture can be very simple. Some customers have as few as one or two interfaces feeding data to a single Data Archive. Access to data is through the single Data Archive.



There are often several Data Archives in an organization, aggregating data from lower levels. Some corporations have Data Archives dedicated to servicing their clients with restricted company data.

1.3 Assets and Tags – The Basic Building Blocks in the PI System

Objectives:

- Define an AF Asset with its components element and attributes.
- Define four attribute types: Static (None), PI Point, Formula, and Table Lookup.
- Define a Data Archive Tag with the attributes Tag Name, Descriptor, and Point Source.
- Define the different data types that can be stored in Data Archive Tags.

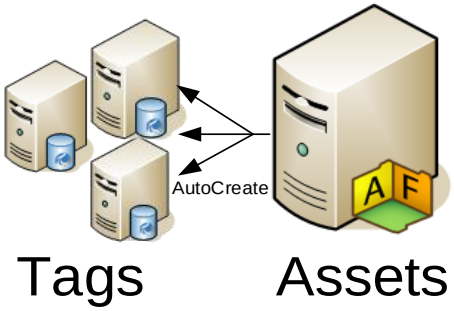


Figure 3: Tag Auto Creation

1.3.1 What is an Asset?

The AF Server is a part of the PI System. It contains asset or "metadata" usually organized according to the assets containing the attributes being monitored. AF can be helpful to users of the Data Archive who know the assets, but are not familiar with attribute nomenclature. With assets, data can be located without understanding the technical details of each piece of equipment. Organized assets help find all of the attributes associated with a specific piece of equipment.

1.3.2 What is an AF Attribute?

Attributes represent a unique property associated with an asset. The attribute maybe a constant, a value from an internal AF table, a value from an external database or a storage point for data in the Data Archive. An AF attribute is simply a single point of measurement. The point has been the traditional storage method of data in the Data Archive. The AF Server can automatically generate points as assets are created.

1.3.3 Some Basic Properties and Why They Are Important to You

AF attributes and Data Archive points have a set of properties that define them. Some common properties used in client tools are for display or informational purposes.

Attribute name

The attribute name is similar in concept to the point description. A detailed name for the attribute may help the user identify the source of the information.

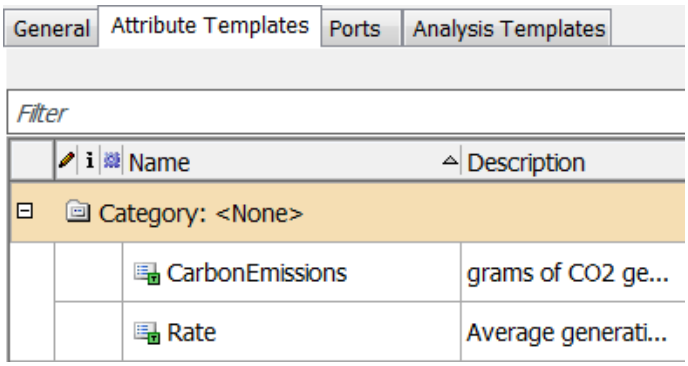


Figure 6: Attribute Name

Tag name

Unique name is used to create points for storage in the Data Archive. Points for data attributes storage can be built through AF templates using substitution parameters for ***local naming convention*** **or can be searched for on the Data Archive.** Creating points through templates, lends consistency in nomenclature making searches easier for PI Administrators. For example, which might be easier to locate in a search?

Point: M03_E1P1_MOTDRV1202_RUNSTAT

Attribute: Machine3 Enclosure 1 Panel 1 Motor Drive 1202 Run Status

Substitution parameters are variables placed in attribute templates for PI point and PI point array data references representing portions of the AF hierarchy.

For example, %Element% is a substitution parameter that represents the element name. After you create an element based on that template, you tell AF to create the data reference. When AF creates the reference, it substitutes the current element name wherever %Element% is present.

Descriptor

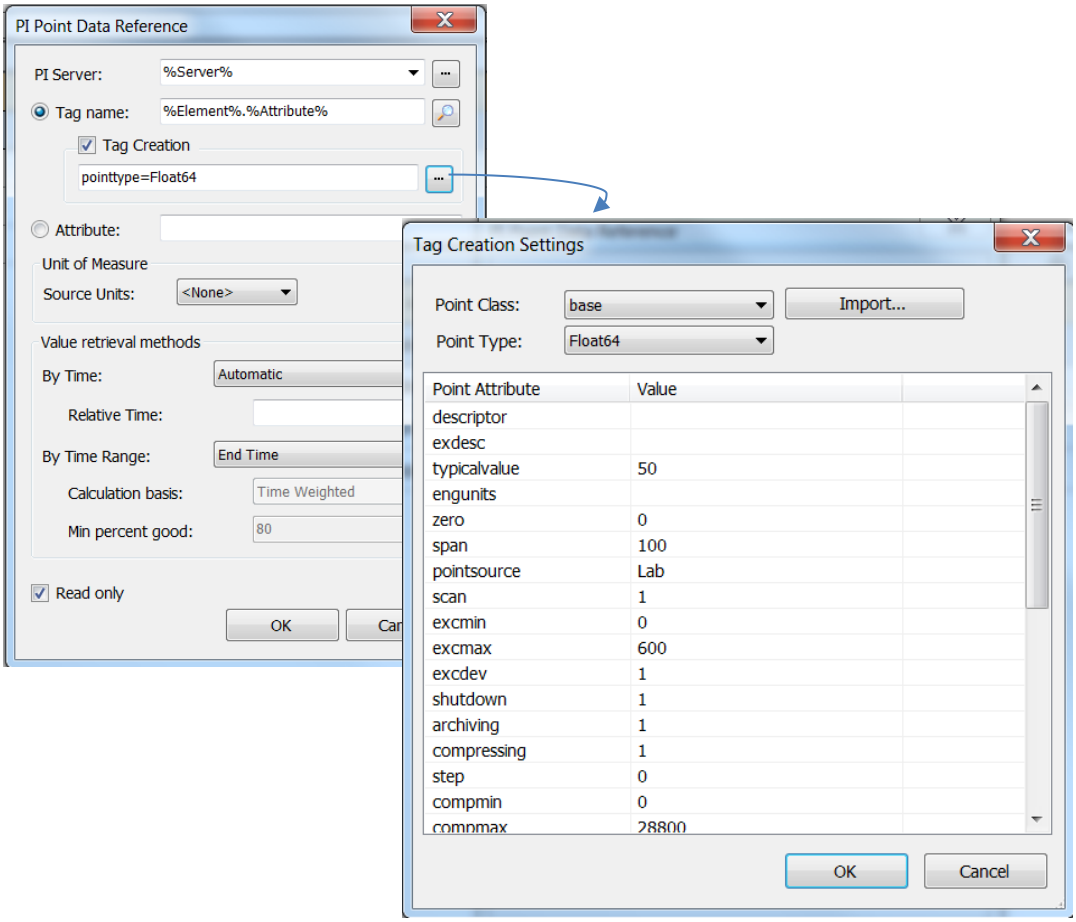
This is the human-friendly description of the Data Archive Point, similar to the attribute. The descriptor is often a **search criterion** since the point name is not always intuitive. Often the point name is some sort of abbreviated convention and the descriptor captures the "full name."

Point source

Points can be related to their interfaces that collect the data by a point attribute called ***pointsource***. Grouping by point source allows all of points associated with a particular device to be identified by searching for all points of a certain **point source**. This assumes that the user knows the point sources in use and that will not be true in most situations.

Point type

The PI point attribute that specifies the data type for the values that a point stores. The possible point types include int16, int32, float16, float32, float64, digital, string, BLOB, and timestamp.



1.4 Discussion

This is a discussion designed to maximize learning in a specific topic area. Your instructor will have questions, and will prompt for communication within the class. This is an open ended section and the result depends on your needs.

Objective: The AF data is to be included within our Business Intelligence tools. What data do we want to see? This can be in the context of your own system, or the example AF databases available in the class.

Approach

- What do we currently use our AF databases for regrading report generation?
- Within the example database, what else do we want to know? What is missing?
- Pros and cons of including external asset related data within the AF structure
- What sort of BI tools would we want to use to view this data?

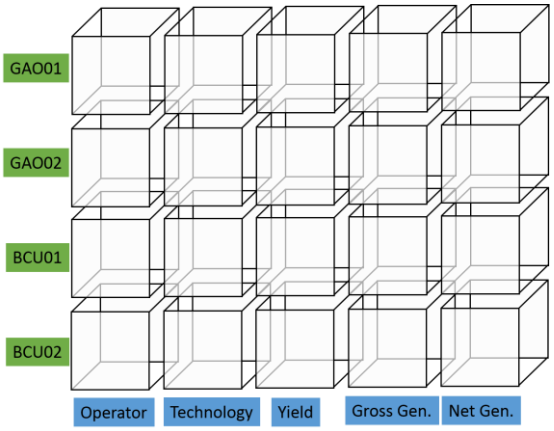
Estimated Completion time 15 minutes.

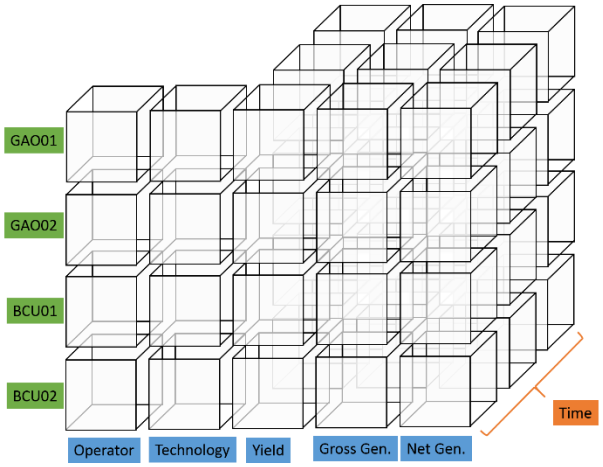
2 Business Intelligence

Business intelligence (BI) tools offer solutions to quickly analyze raw, un-normalized, multidimensional data. In concert with historical values from the Data Archive, metadata and calculations from Asset Framework, and business intelligence tools, users can quickly create interactive reports to gain insight on business and operational processes.

Later on in the course, we will explore the process of preparing the Asset Framework model to add additional dimensions of information to our AF database. The next step is extracting desired information (process data, metadata, and event frame data) from the PI System through PI Data Access tools. This data will be incorporated into a BI cube and used to develop interactive reports that allow us to "slice and dice" our data and bring meaning to our multidimensional data cube.

The Distribution Network and Fleet Generation databases have a comprehensive amount of information including a hierarchy of substations, metadata for each asset. The figure to the right depicts a data cube that captures metadata and real-time data of generating units.





Inclusion of additional attributes through table lookups and analytics on existing attributes allow for the expansion of additional columns (or dimensions) to the data cube above.

Further, historical data, interpolated or compressed, add an additional dimension of information that bring more meaning in Business Intelligence reports.

In the next several chapters in the course, we will be using a pair of AF databases to expose meaningful data that will help management and engineers make better, more informed decisions. Specifically, we will add value through the following:

1. Expose the database in a simpler structure for data processing.
2. Develop analytics within PI AF and PI Integrator for Business Analytics
3. Import the data into Microsoft Power BI
4. Draw actionable conclusions from the resulting data sets in our reports

2.1 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

2.2 Directed Activity – Inspect a Sample Power BI Report

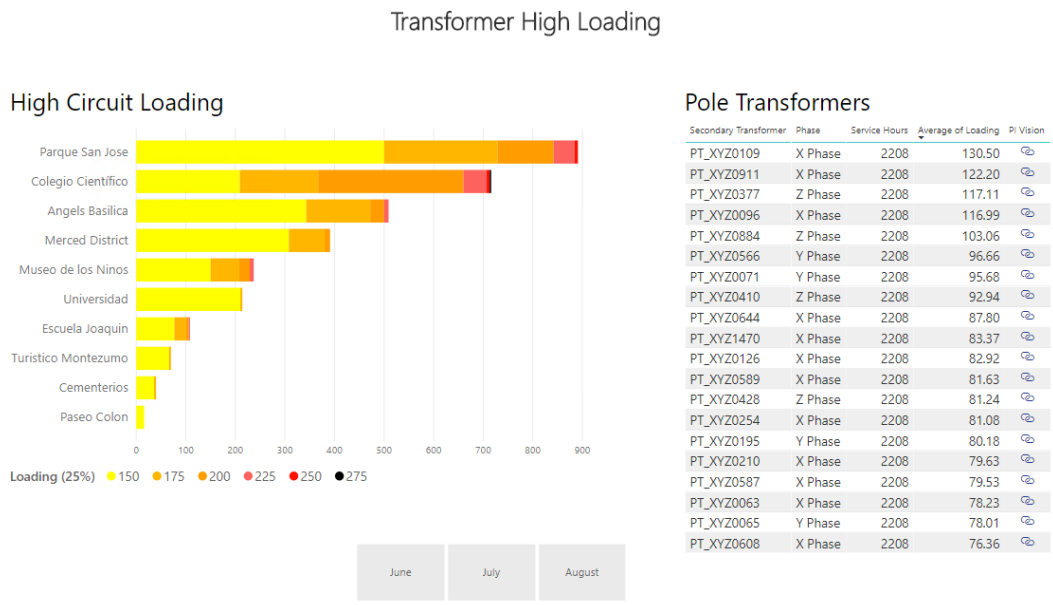
In this part of the class you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

- Explore a sample Power BI Report.

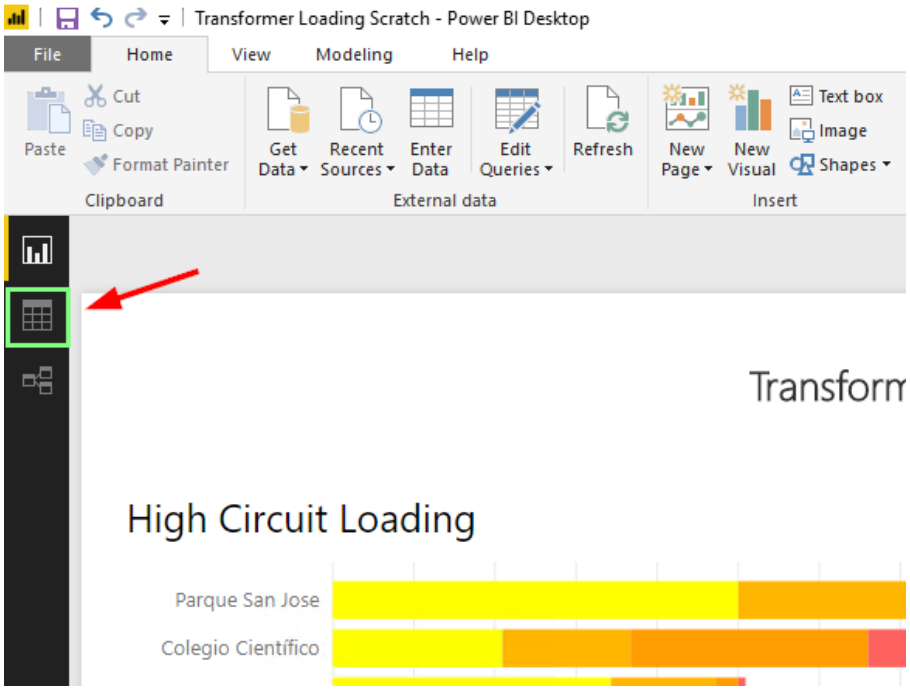
Approach

We'll start by getting a feel for Power BI using a pre-built report. **Open C:\Class\Part 1 - PI Integrator for BA\Solutions\Transformer Loading.pbix**

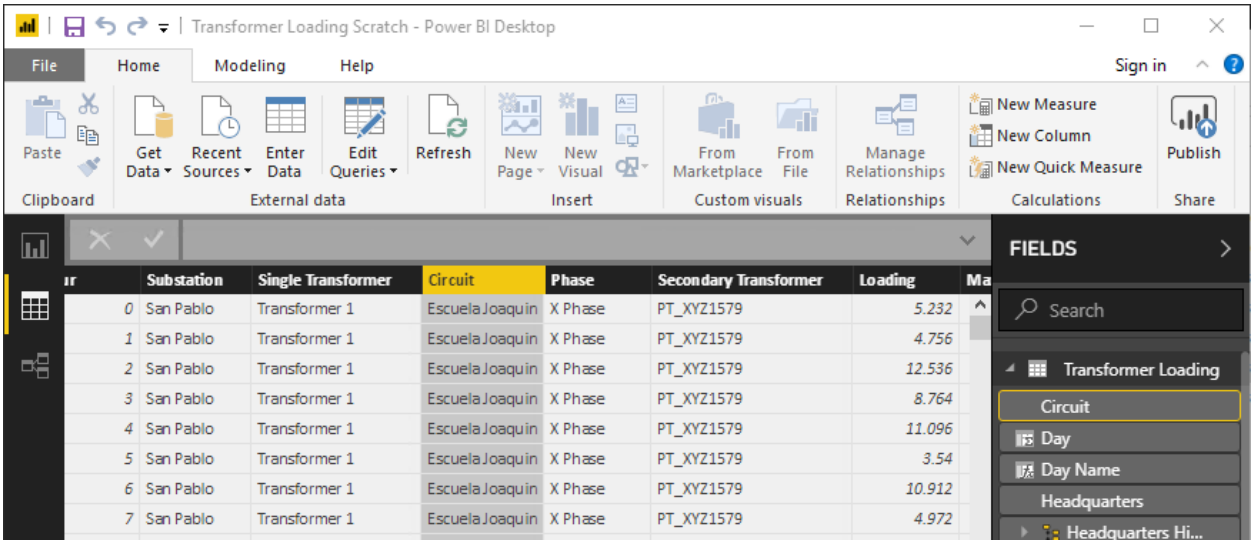


Right now we're looking at the **Report View** where the report and visuals are configured. Start clicking on the visuals and the rest of the report will be filtered to only include the selected items. This is often referred to as **slicing and dicing** the data.

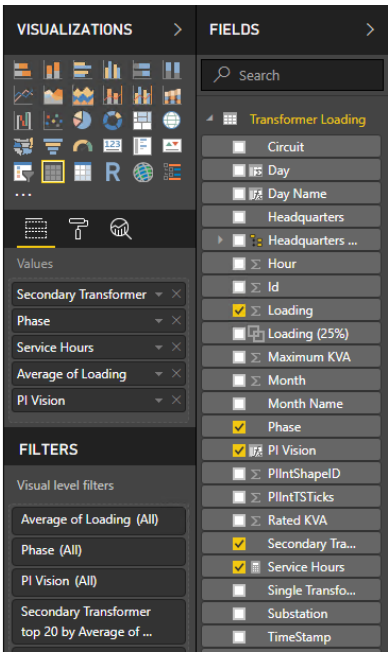
Click on the **Data Tab** to inspect the data set we'll be working with:



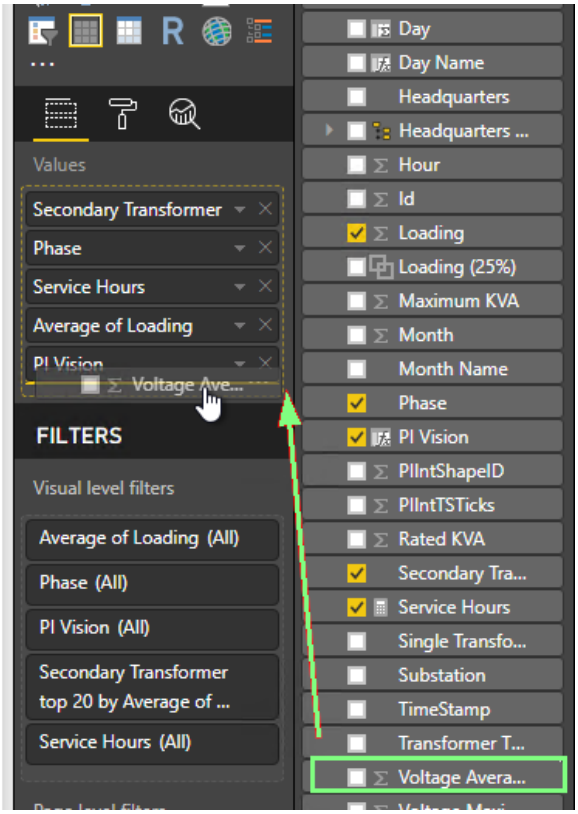
Note that all the columns are available in the **Fields List**:



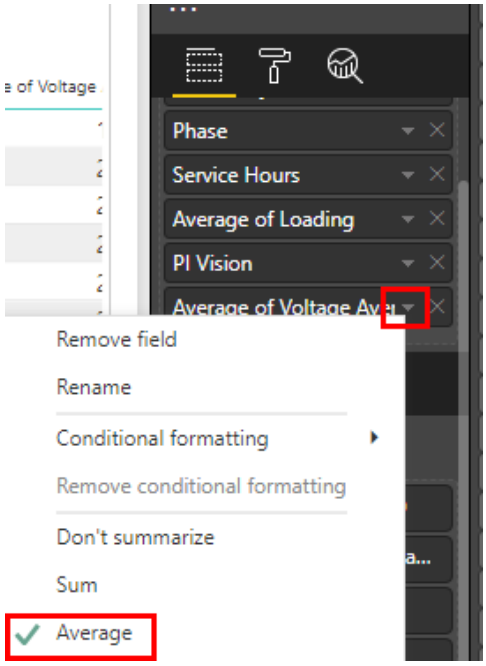
Go back to the Report View, click on the **Pole Transformers Table** visual, and note the **Visualizations Pane** and **Fields Pane**. These sections are where the bulk of the configuration takes place. Columns from the data set are dragged and dropped from the Fields Pane onto the various sections in the Visualizations Pane. We can see that the Secondary Transformer, Phase, Service Hours, Average of Loading, and PI Vision columns are being displayed in the table.



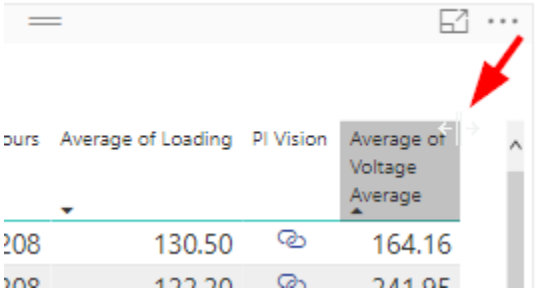
Add the **Voltage Average** column to the table by selecting the table and doing a drag and drop:



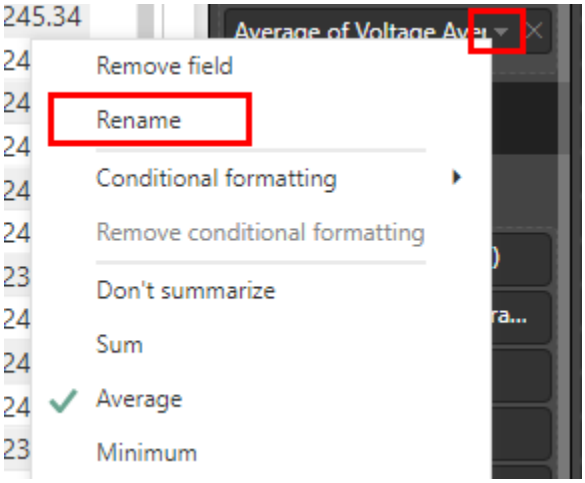
The Voltage Average will be displayed in the table, but by default all the voltages from all the rows will be summed by transformer. It makes more sense to summarize these as an average, so click the drop down and change the summary to **Average**:



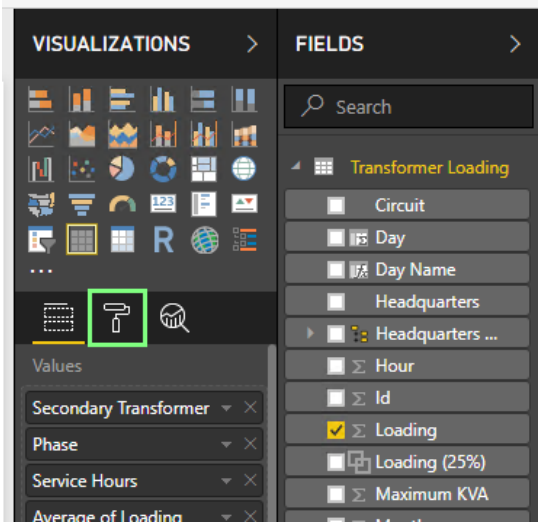
Resize the column directly on the visual similar to Excel:



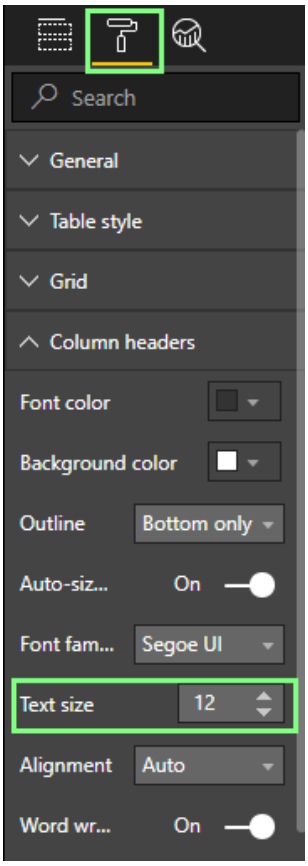
Average of Voltage Average is a pretty weird header name, so rename it to Average Voltage:



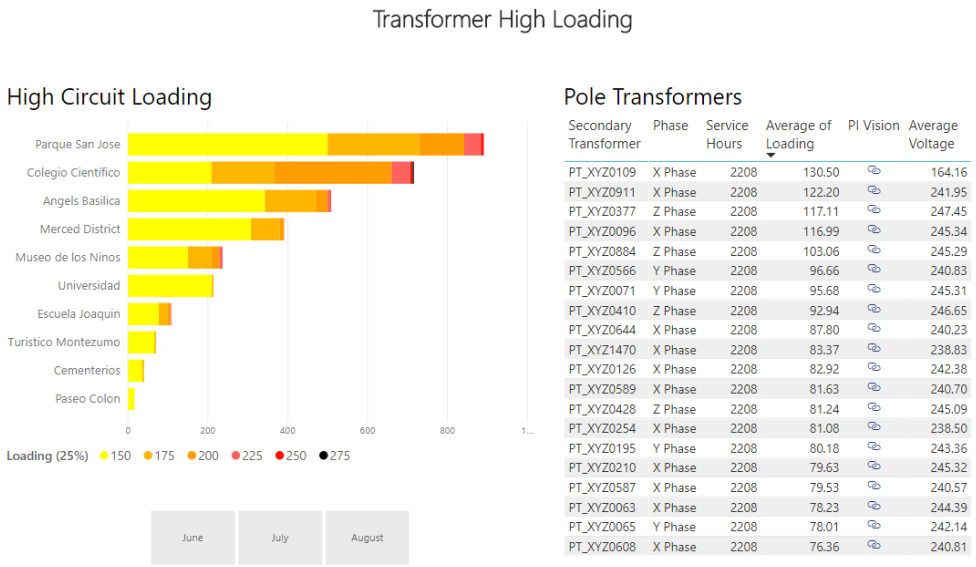
Other **Formatting Options** are available by clicking the paint roller icon:



We'll go over more formatting options in the exercises, so for now just bump up the text size of the column headers and resize columns and visuals so everything fits:



Your report should now look something like this.



We will build this report from scratch in a future exercise.

3 Part 1 – Power BI Reports using PI Integrator for BA

This course will be broken down into two main sets of exercises. In Part 1, we'll use PI Integrator for BA to publish data from PI and spend a lot of time configuring Power BI. In Part 2, we'll make modifications to a PI AF hierarchy and then use PI OLEDB Enterprise to extract the report data.

In Part 1, we will be working with a data set for a power distribution company, which includes electrical characteristics for over 1500 single-phase transformers. The source data comes from a real PI System and will be published in a data-science ready format using PI Integrator for BA. Once this is done, we'll configure an array of Power BI visuals and integrate the results with PI Asset Framework and PI Vision.

The transformers we will be analyzing are secondary transformers that deliver power to homes and businesses, which you may have seen on a pad or pole in your own neighborhood. There are thousands of them to keep track of, making this a difficult problem to solve using Excel.

The transformers themselves are not actually instrumented. The power and voltage characteristics we will analyze have actually been computed by rolling up child Meters in PI AF.

3.1 Directed Activity – PI AF Hierarchy and Data Set

In this part of the class you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

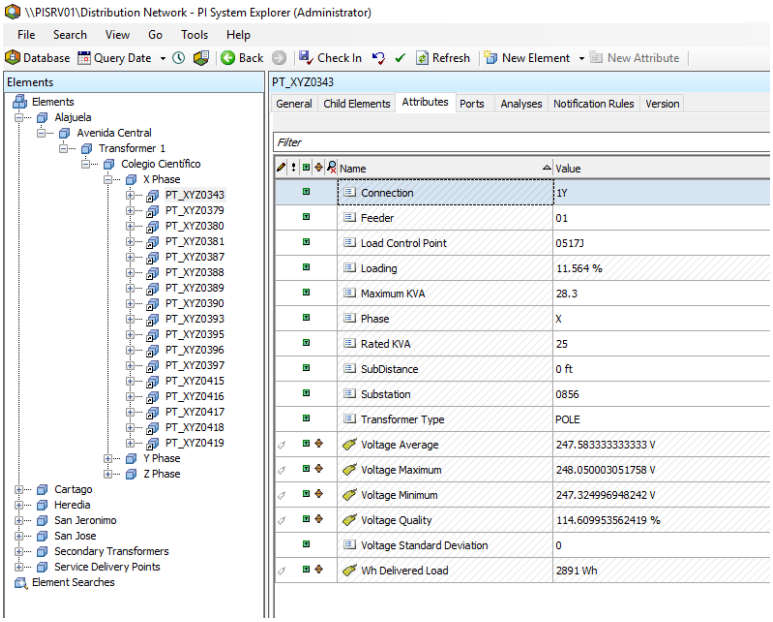
Objective:

- Better understand the data set used in the following chapters

We will take a few minutes to understand where the data set came from and relate the sample Power BI report back to the PI System. We are working with a data set for a fictitious power distribution company. They have built a PI AF Hierarchy for their transformers and meters serving a number of geographical areas. In this course, we will focus on analyzing the transformers.

Open PI System Explorer and head to the **Distribution Network AF database**. Drill down to a level with transformers (names starting with PT_) and inspect the available attributes. We will be using a sub-set of these attributes for all of our analysis, in addition to leveraging the AF hierarchy.

Note that because the Transformers are not instrumented, Voltages, Voltage Quality, and Wh Delivered are rolled up from the child meters using Asset Analytics.



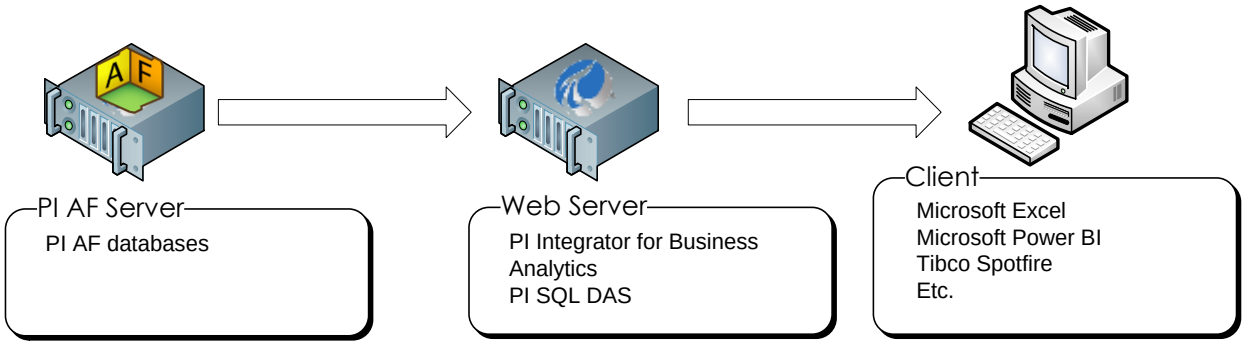
Data from this PI AF hierarchy will be published for use in a Power BI report in a later exercise.

4 PI Integrator for Business Analytics

Getting the data out of the AF structure and into the client tools requires the use of integration software such as the PI Integrator for Business Analytics or PI OLEDB Enterprise. This chapter will discuss the former method of extracting the data.

4.1 Architecture

The PI Integrator for Business Analytics resides on a web server between the client machines and the source AF server. As such, we are not connecting directly to the AF Server but instead to a web server that contains a cache of our desired information. The architecture within our class system however has both the AF Server and Web Server residing on the PISRV01 machine. The PI Integrator for Business Analytics site can be accessed via https://pisrv01.pischool.int:777 or from the desktop. If prompted for credentials, enter your student account, as this has been given access rights.



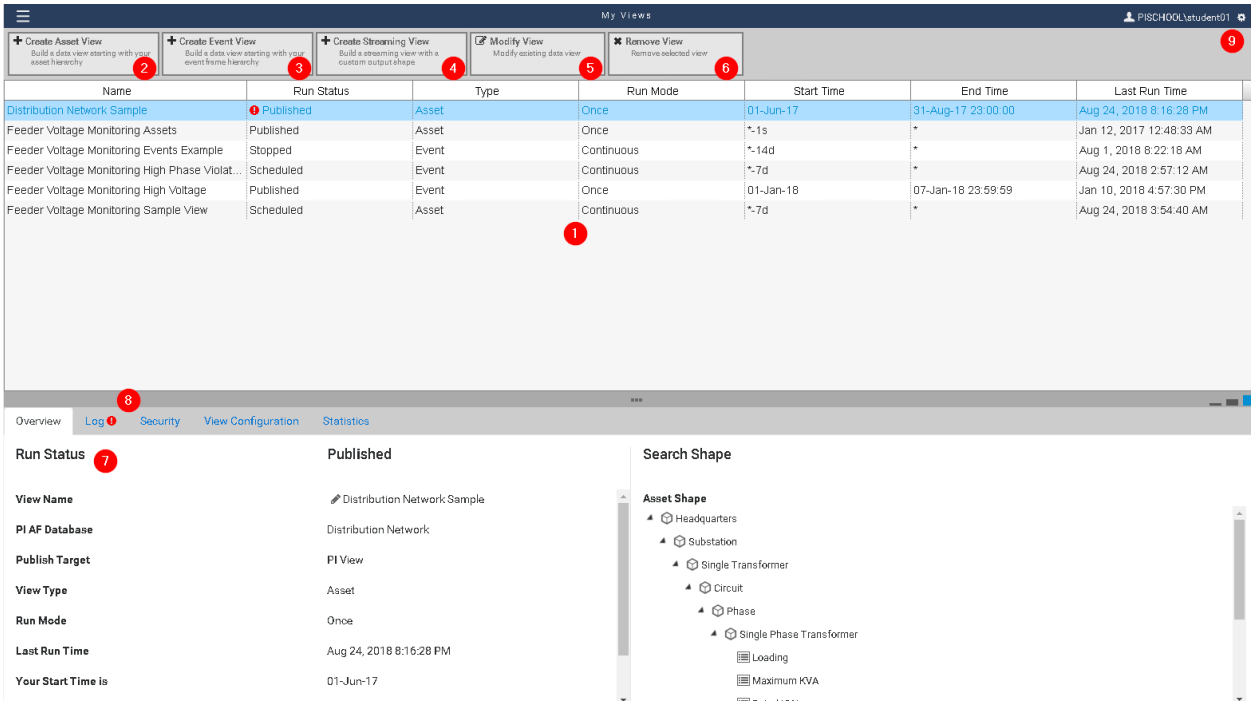
4.2 PI Integrator Web UI

Views can be created within the PI Integrator portal that is hosted on the Web Server machine.

A list of previously generated views is present within the portal on the **My Views** page, allowing for previewing and maintenance. These existing views can also be cloned and modified, allowing different views to be created and utilized within BI client tools.

The following is a breakdown of the **My Views** page layout, and the different operations available.

Note: The information regarding the My Views page layout is available within the PI Integrator for Business Analytics User Guide.



The My Views page shows details about your views.

1. All the views to which you have access are listed in the table
2. Click to create an **Asset View** that is based on Elements and Element Templates
3. Click to create an **Event View** that is based on Event Frames and Event Frame Templates
4. Click to create a **Streaming View** that is based on Event Frames and Event Frame Templates
5. To modify a view, select the view in the table and click **Modify View**.
6. To delete it, click **Remove View**. Deleting a view removes data from the buffer, therefore freeing up space. However, this does not free up the available output streams allowed with your license.
7. For the selected view, the Overview, Log and Security tabs provide the following details about that view:
   - Overview indicates whether the view has been published. This tab also summarizes information about the view, such the PI AF database it uses, when the view was last run, and the shape that it uses. If the view is currently being published, the run status bar indicates progress and you have the option to stop the publishing process.
   - Log displays log information. You can adjust the start and end times, and you can filter the messages to display those of a certain severity, for example, critical errors.
   - Security shows who has access to the view, and if you have sufficient privileges, allows you to change the level of access
8. The red message counter icon at top right show that there are warning and error messages

recorded by PI Integrator for Business Analytics. Click the icon to open the message list.

9. Click the gear icon  at top right to see the version of PI Integrator for Business Analytics and AF you are using.

4.3 Directed Activity – Create the Transformer Loading View

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

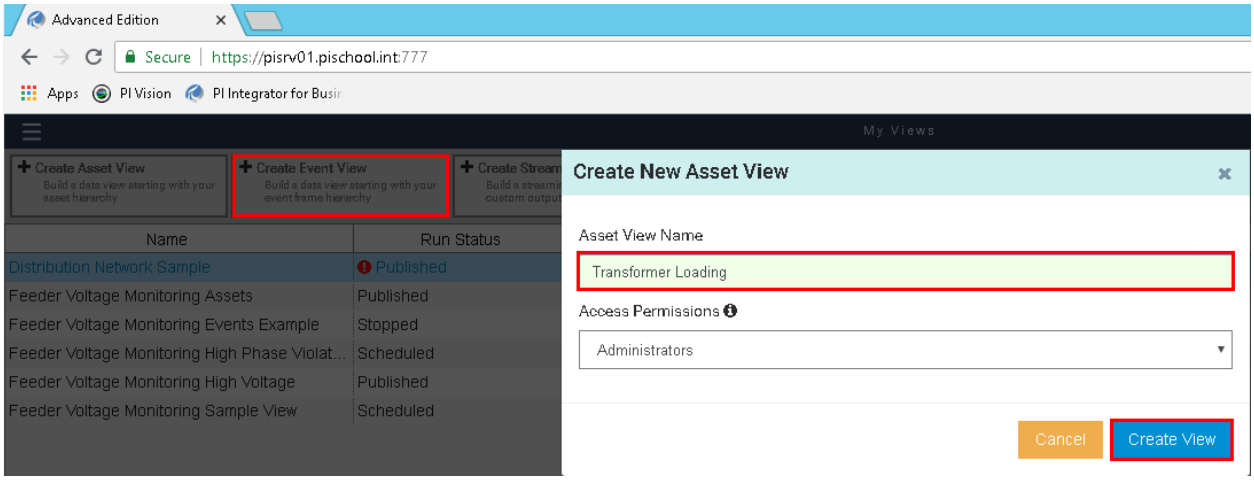
Objective:

- Use the PI Integrator for Business Analytics to create an Asset View, which will be used in later exercises.

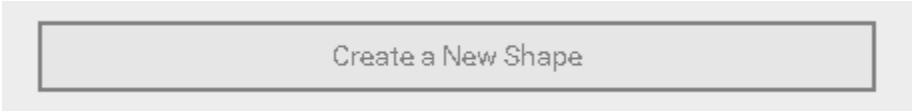
Approach:

Open Google Chrome and Navigate to the PI Integrator for BA Web UI at https://pisrv01.pischool.int:777

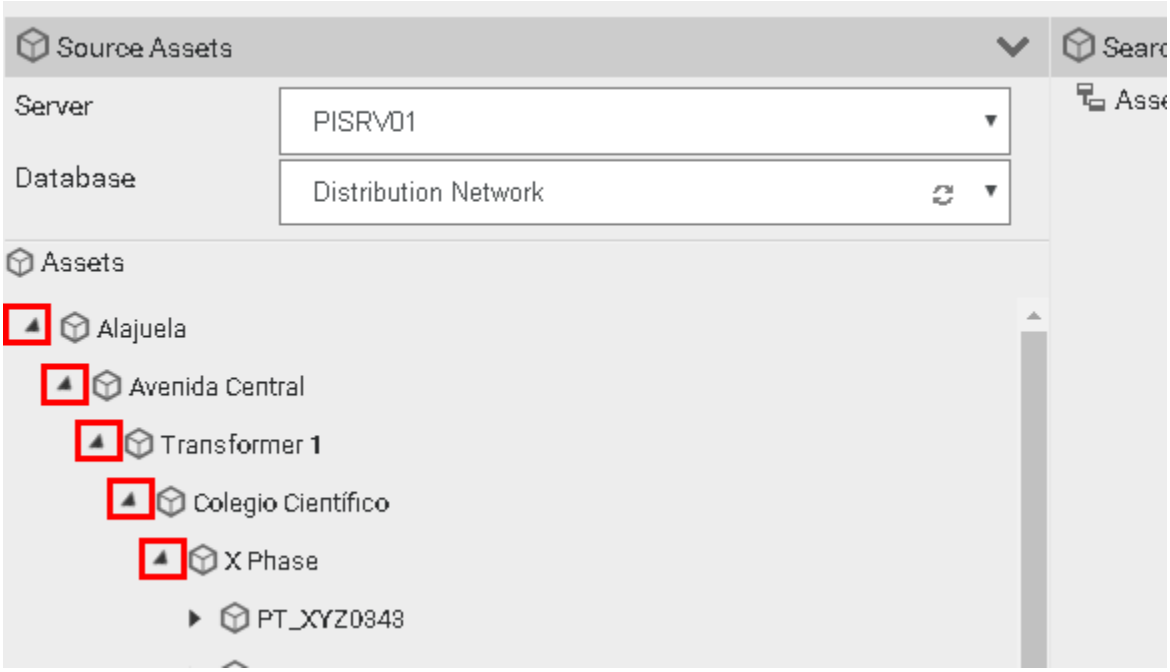
Click **Create Asset View** and name it Transformer Loading, click **Create View**:



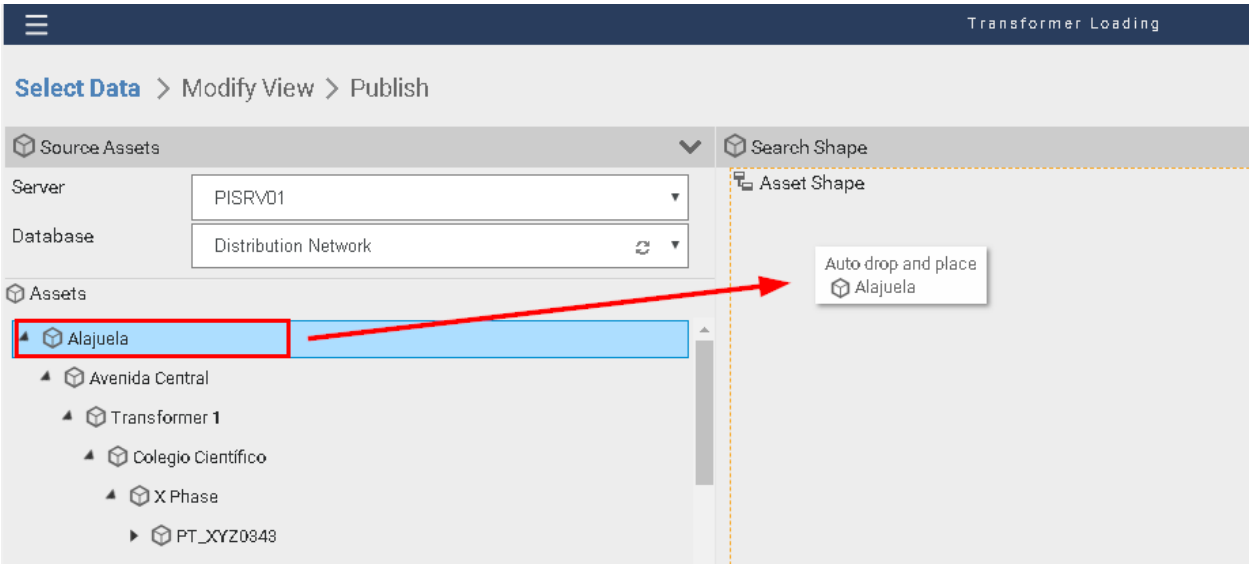
Click **Create a New Shape**



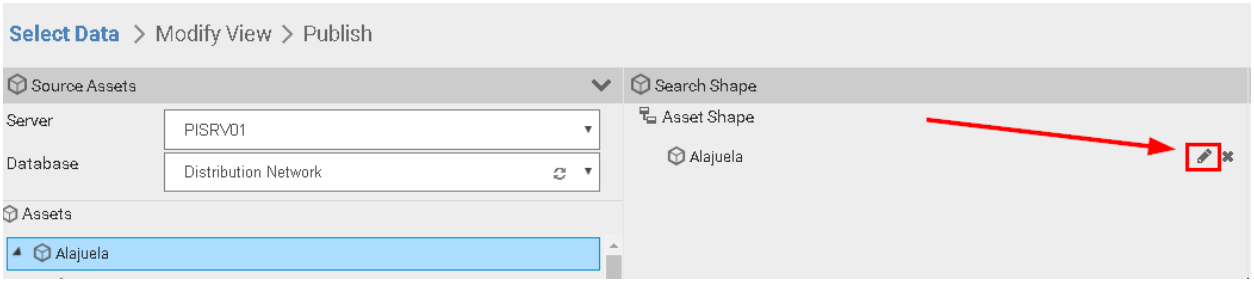
Select Distribution Network as the AF Database, then drill down to PT_XYZ0343.



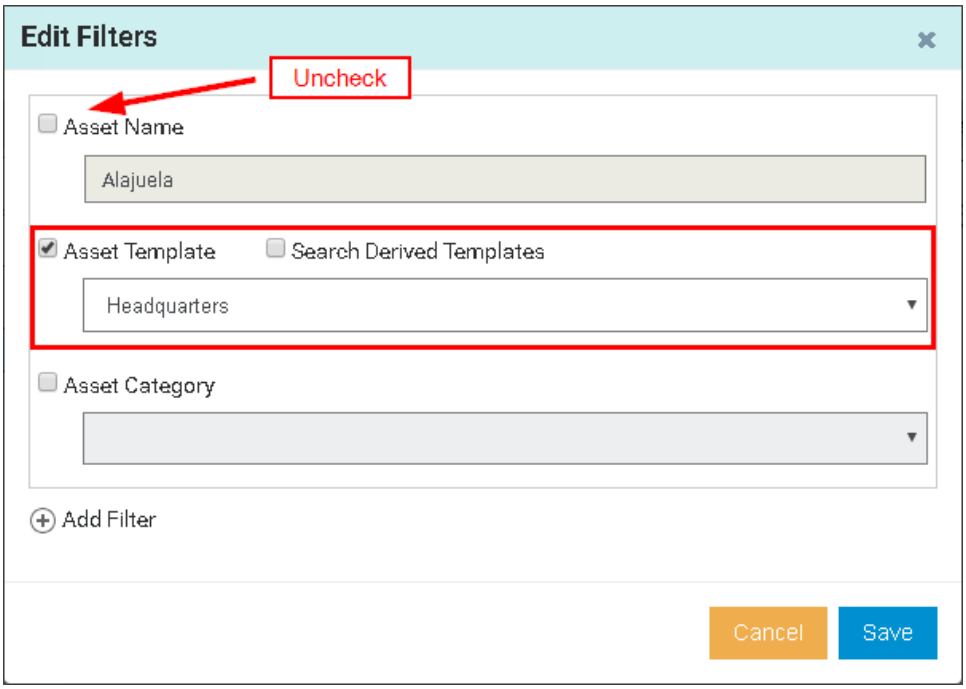
Drag and drop Alajuela to the Shape Builder



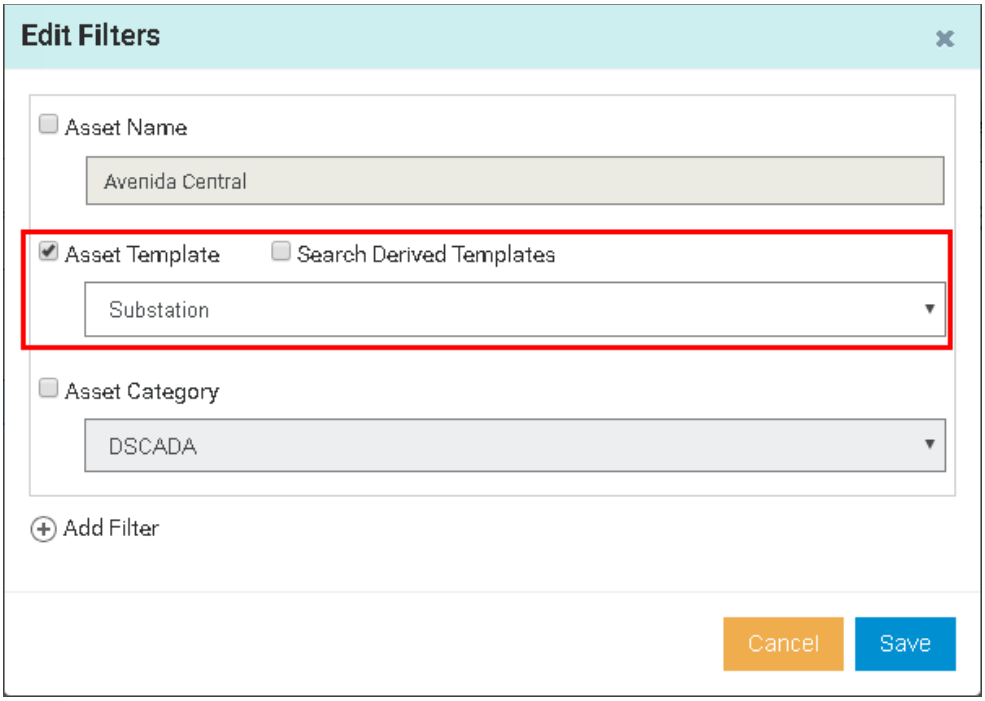
Edit the Filter on Alajuela:



Clear the Asset Name Checkbox, Change it to filter on the Headquarters template, click **Save**:

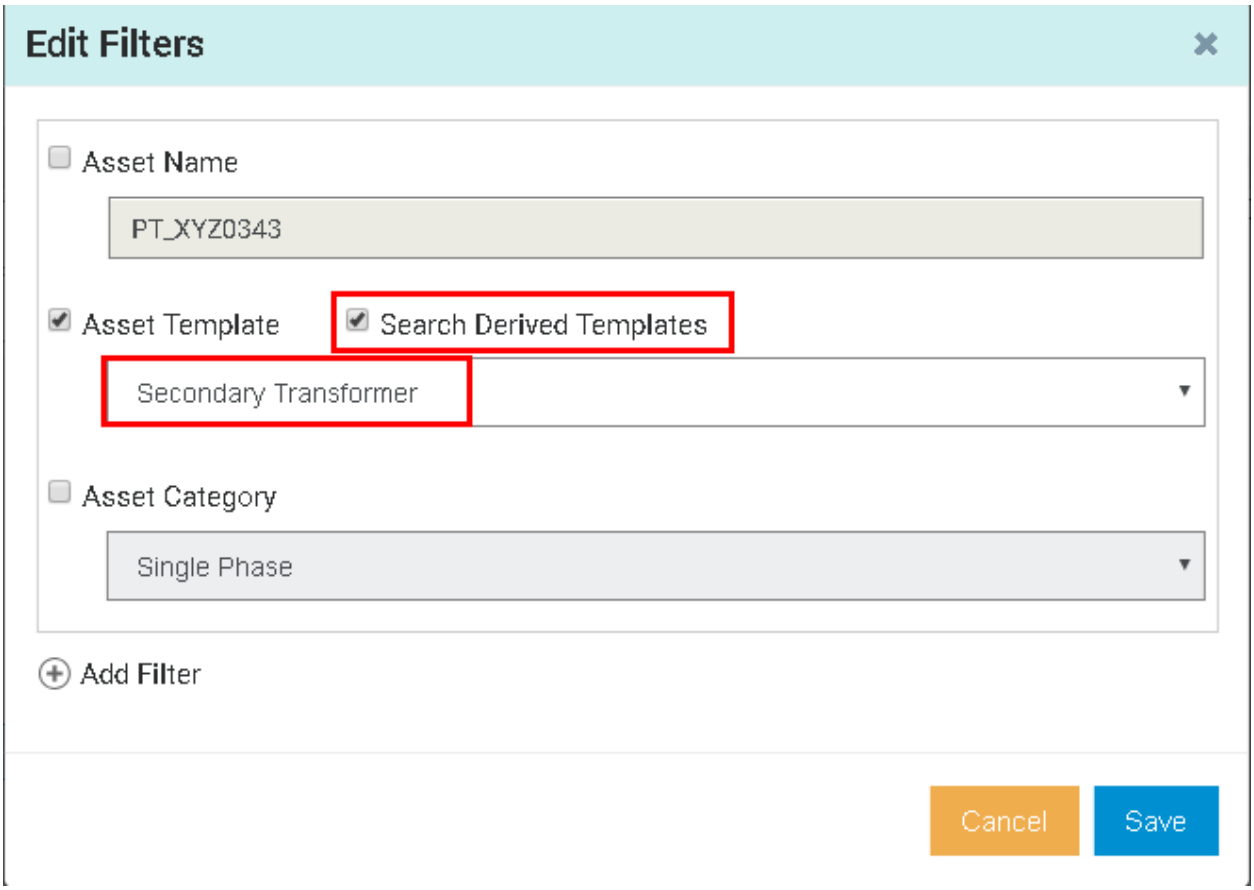


Drag and drop **Avenida Central** to the Shape configuration, and change it to filter on the **Substation** Template:



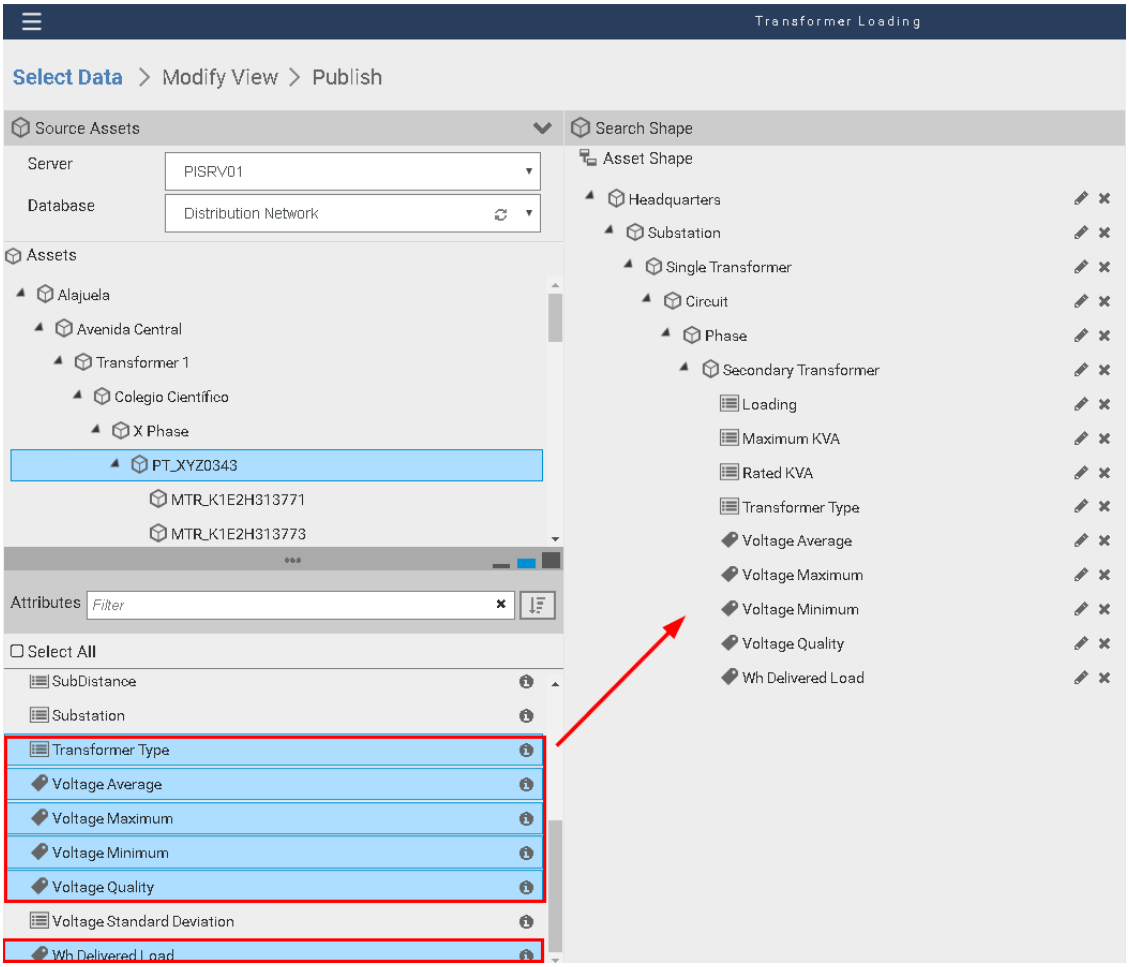
Repeat this pattern for **Transformer 1** (Template = Single Transformer), **Colegio Cientifico** (Template = Circuit), **X Phase** (Template = Phase).

Drag and drop **PT_XYZ0343** and select **Secondary Transformer** as the Template, this time check the box to search derived templates.



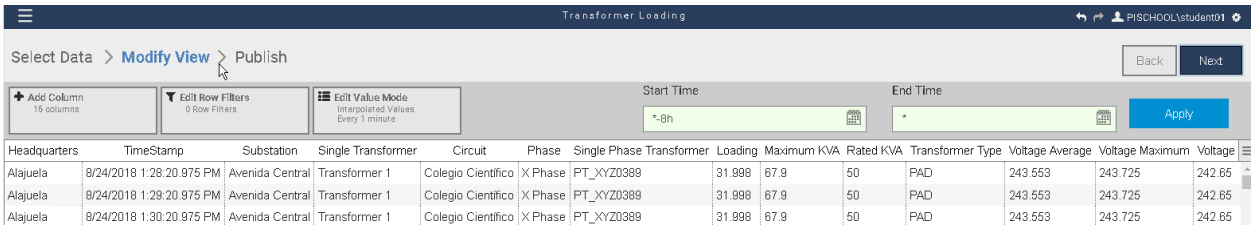
The shape configuration should look like this:

Click **PT_XYZ0343** then hold control and multi-select Loading, Maximum KVA, Rated KVA, Transformer Type, Voltage Average, Voltage Maximum, Voltage Minimum, Voltage Quality, and Wh Delivered Load. Drag and drop these selections to the Shape configuration.

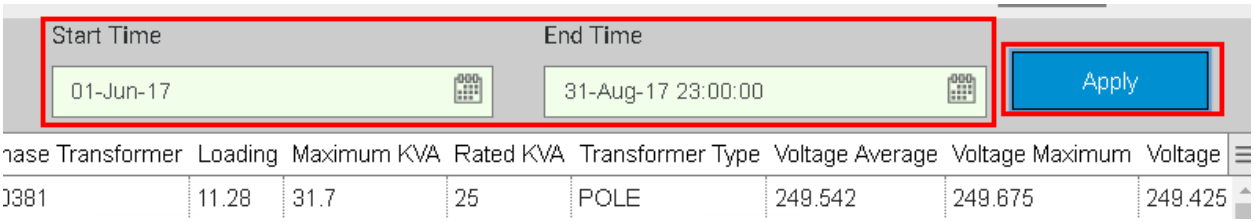


There should be over 100 matches in the preview, click Next in the top right corner.

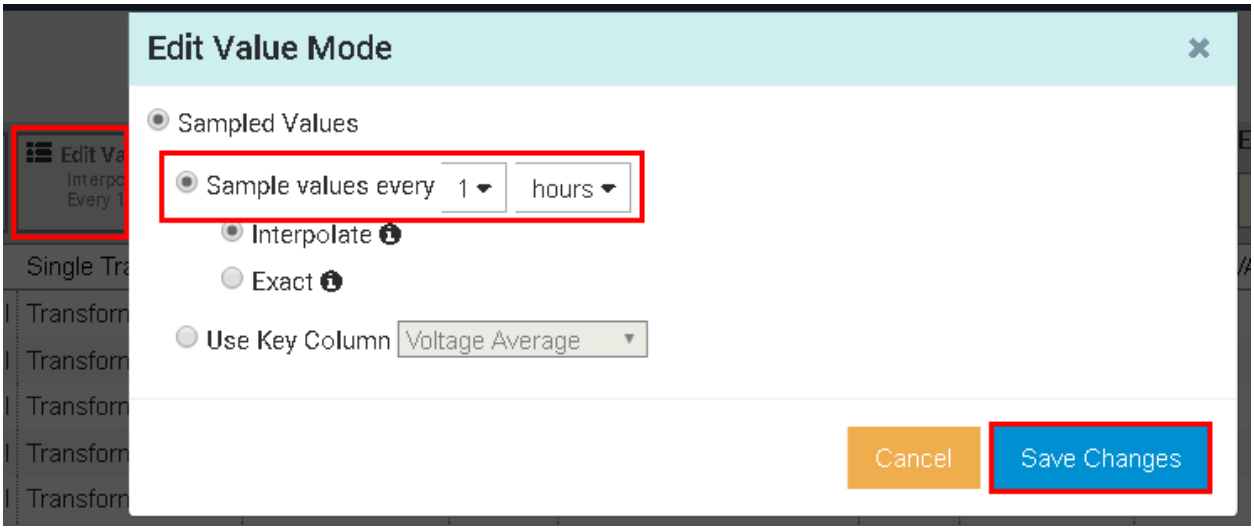
We now see a preview of the data using the default Time Range and interpolation mode.



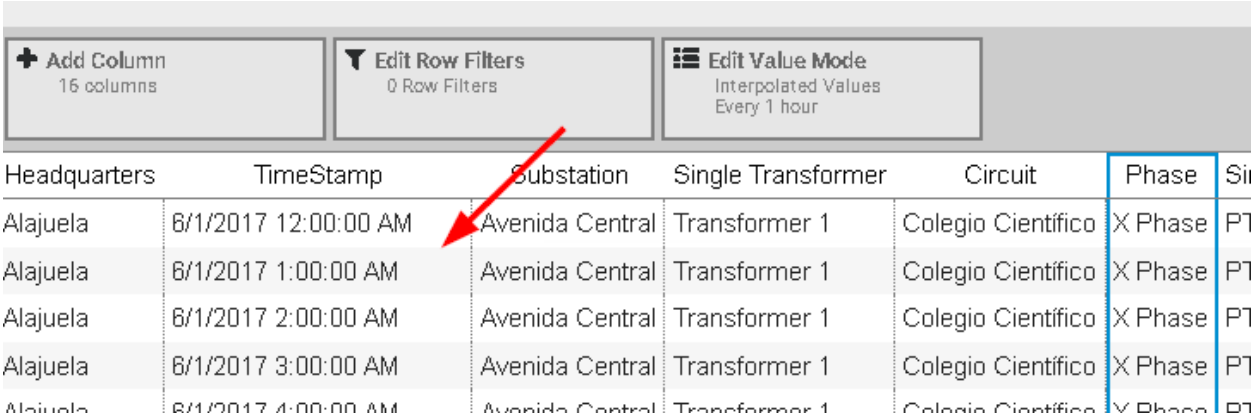
We want to publish Hourly data for the time period 01-Jun-17 00:00:00 to 31-Aug-17 23:00:00. Modify the Start Time and End Time and click Apply:



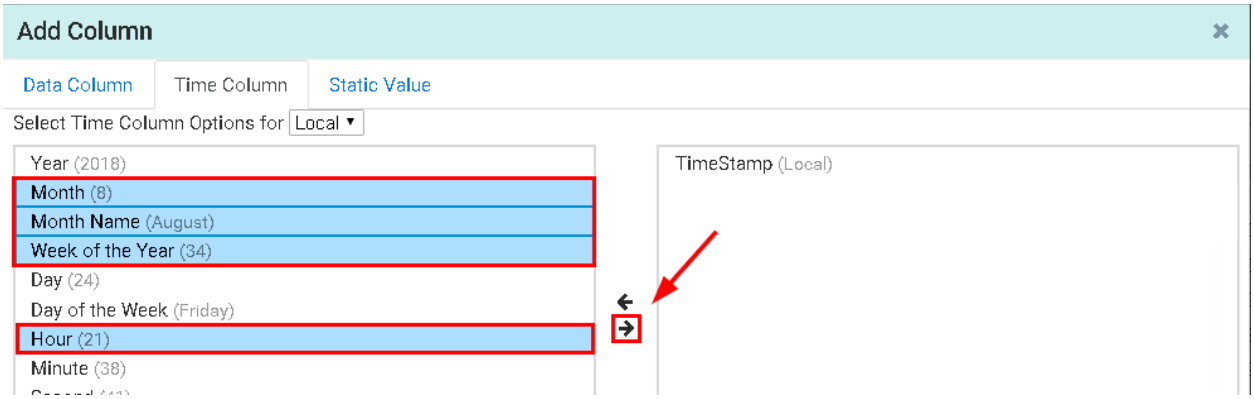
Click **Edit Value Mode** and change the time step to 1 hour, then **Save Changes**:



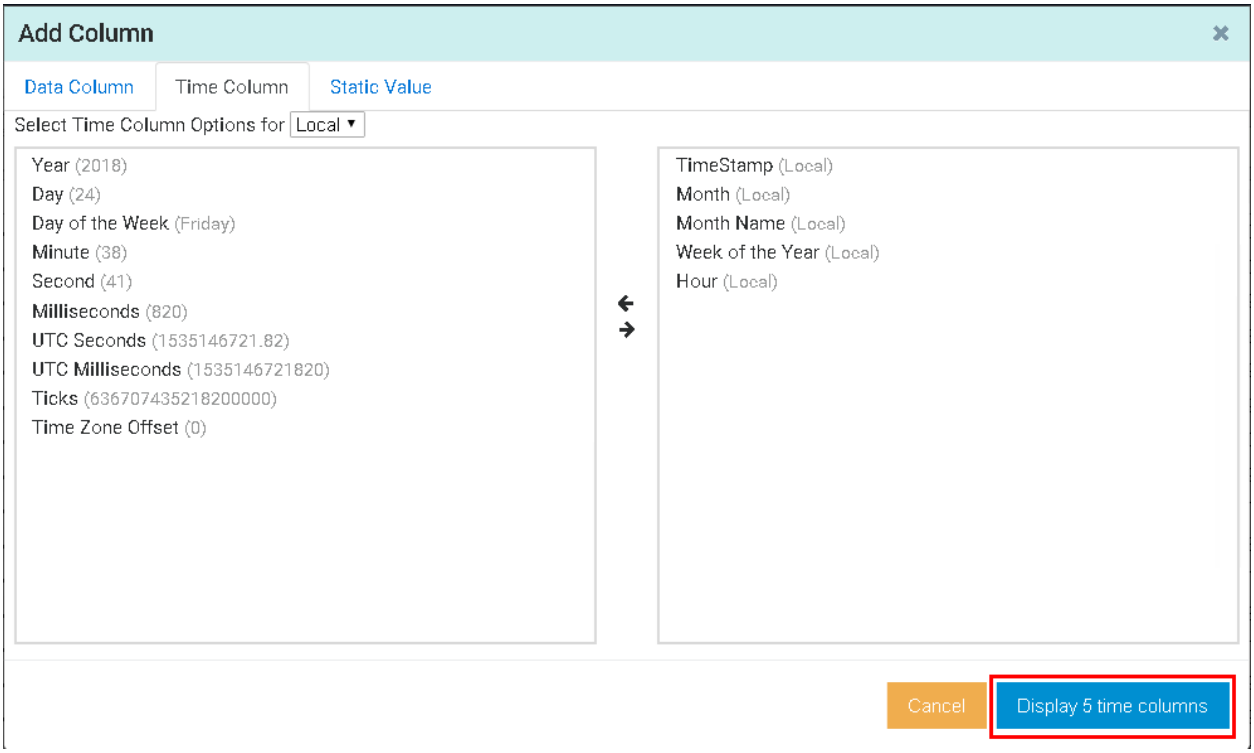
The TimeStamp column should now reflect changes to the Start, End, and Value Mode:



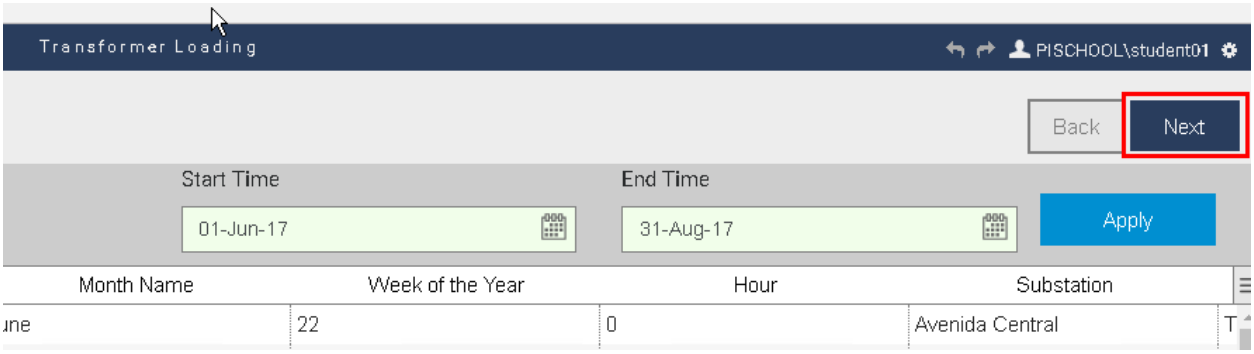
Now we'll add some additional time columns that will come in handy later when building the reports. Click **Add Colum**. Select the **Time Column** tab. Select Month, Month Name, Week of the Year, and Hour, then click the arrow to bump them over to the right:



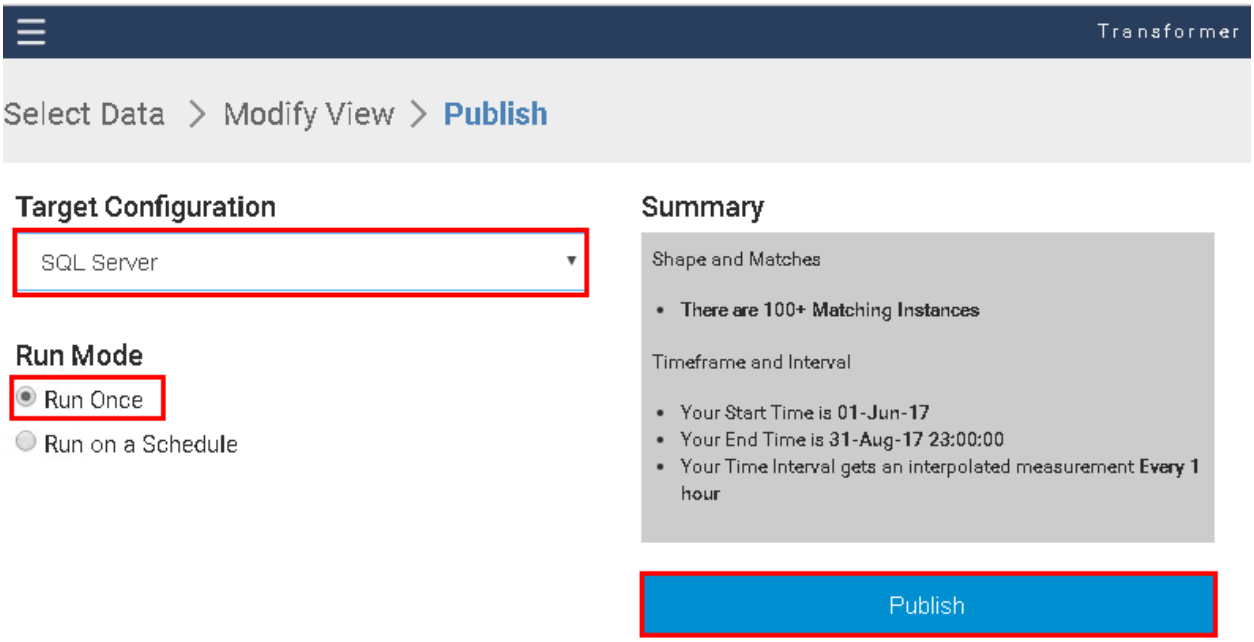
Click Display 5 Time Columns:



Now the time ranges and columns have been specified, click **Next**.



Now we can choose what target to publish to. This depends on the platform used to support front-end application, but for our purposes we'll publish to a SQL Server. Select **SQL Server** for the Target Configuration, Leave Run Once checked, and click **Publish**:



It will take a few minutes to publish the data.

5 Building the Distribution Network Reports

We will now spend a significant amount of time configuring Microsoft Power BI reports. The first step is importing the data.

5.1 Preparing and Importing the Tables

Now that the Transformer Loading table has been published, we will import the SQL table into Power BI.

5.1.1 Directed Activity – Import Data from Microsoft SQL Server.

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

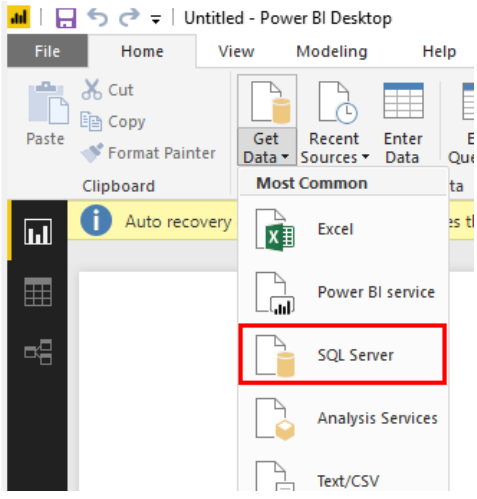
Objective:

Import the Transformer Loading table.

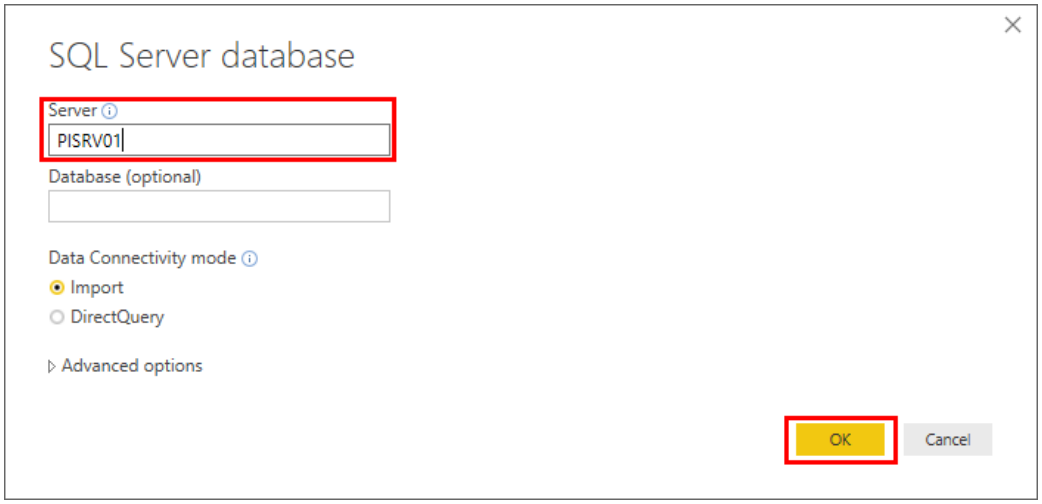
Approach:

Open Microsoft Power BI

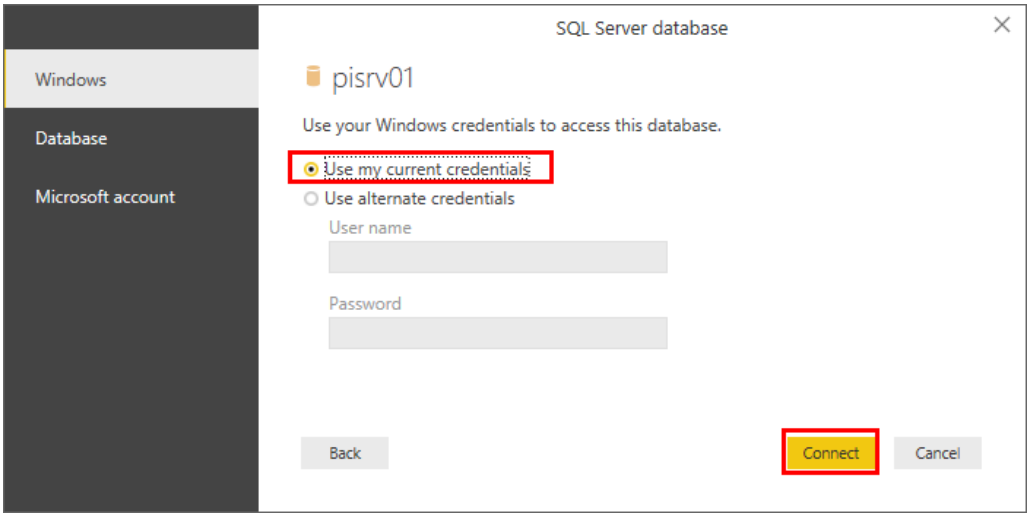
Select **Get Data** -> SQL Server.



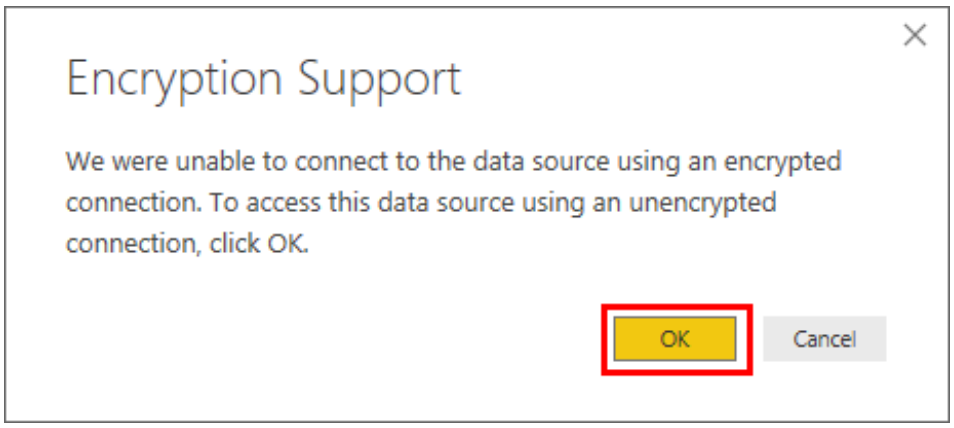
Enter **PISRV01** as the server name and click **OK**.



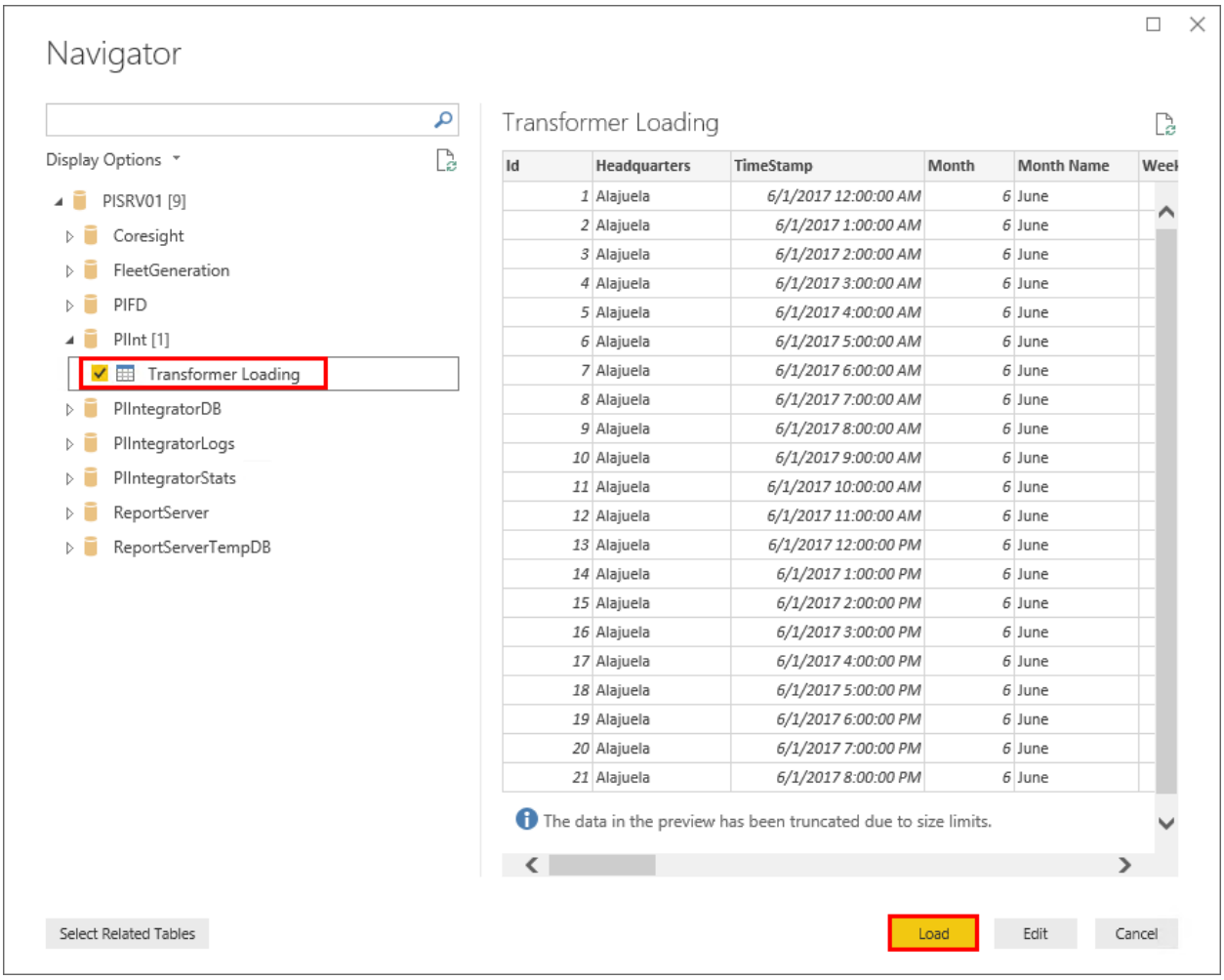
If Prompted, Leave “use my current credentials” selected and click **Connect**:



There may be a warning that the connection is not encrypted, this can be safely ignored, **click OK:**



Expand the PIInt database and Select the **Transformer Loading** table, click **Load**



Note that about 3.8 million rows have been imported.

5.2 Building the Report Visuals

Now that the Transformer Loading table has been imported, the rest of the chapter will be a walkthrough of configuring various report visuals.

In case there were mistakes or problems with the previous steps, a starter .pbix file has been created with the raw data set already imported with columns that will match the exercises exactly.

Open **C:\Class\Part 1 - PI Integrator for BA\Starter File - Part 1 Distribution Network.pbix** and use this as a starting point for the remaining exercises.

5.2.1 Directed Activity – Network Load Profile

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objectives:

- Configure a Hierarchy
- Configure a **Matrix** to show average loading per transformer
- Configure a **Clustered Column Chart** to show the average energy delivered by Headquarters
- Configure a **Line Chart** to show the average energy delivered by hour of the day and by day of the week

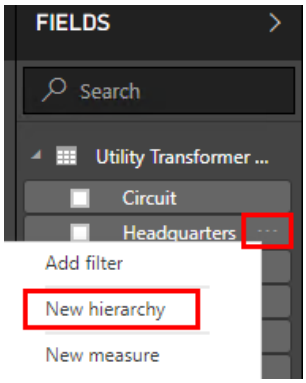
In this exercise, we are going to analyze the loading characteristics of the transformers in our distribution network. We want to look at the demand curve for various levels of the distribution network to help forecast the demand seen by various transformers based on the hour of the day and day of the week.

Loads are expressed as a percentage of the rated value, as well as a raw watt-hours value. The resulting report will help us understand which transformers are working under the highest loads, and see how power consumption changes over time. General Steps:

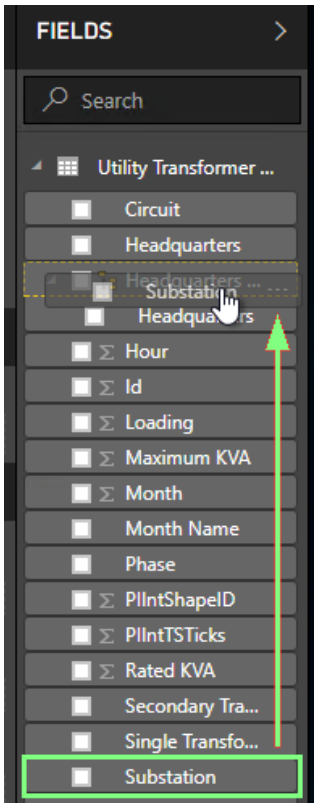
Approach:

Configuring the Hierarchy

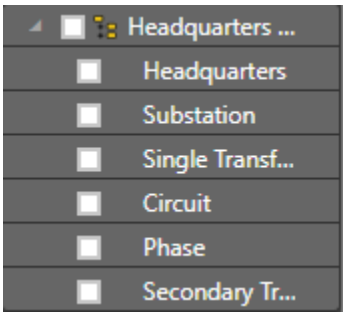
We will now create a hierarchy. In the **Fields List**, click the ellipses next to Headquarters and select **New hierarchy**:



Within the fields list, drag and drop the **Substation** field on top of the new Headquarters hierarchy:



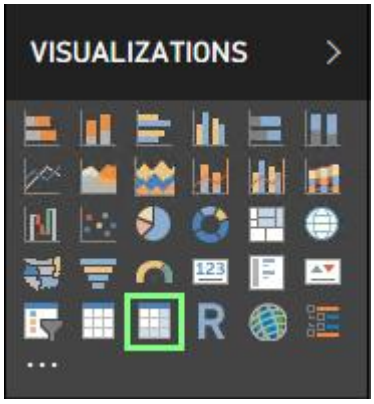
Repeat for **Single Transformer, Circuit, Phase**, and **Secondary Transformer** and reorder so that Substation is below Headquarters in the hierarchy.



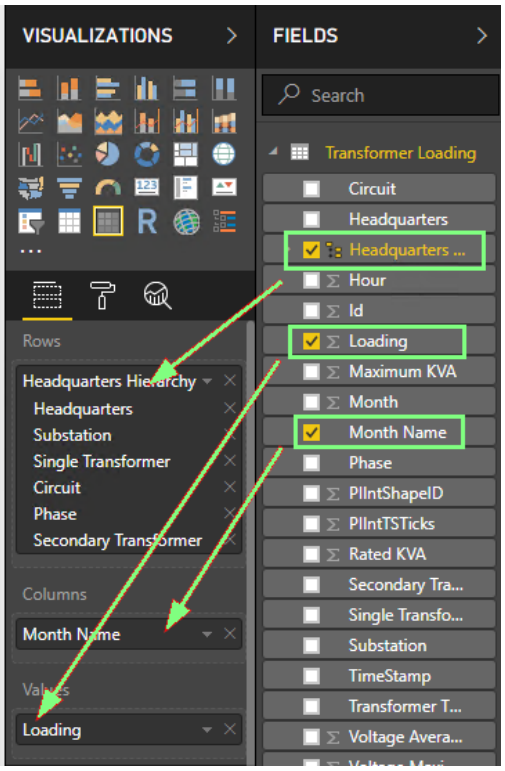
Monthly Average Loading - Matrix

We're going to display the monthly average loading on a per transformer basis and leverage the Hierarchy we just set up.

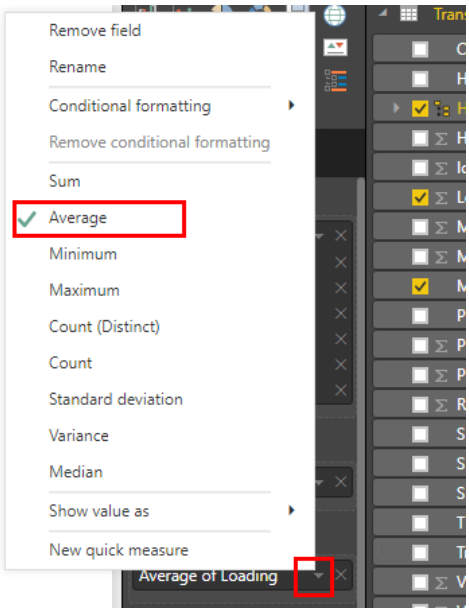
Add a **Matrix** to the canvas by clicking the Matrix icon in the Visualization Pane:



Drag and drop the **Headquarters Hierarchy** for the Rows, **Month Name** for the columns, and **Loading** for the Values:



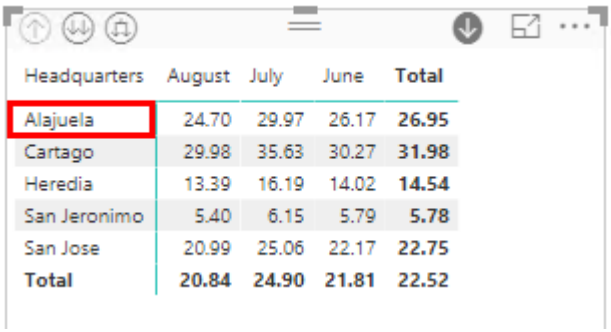
Change the value field to summarize by **Average** Loading:



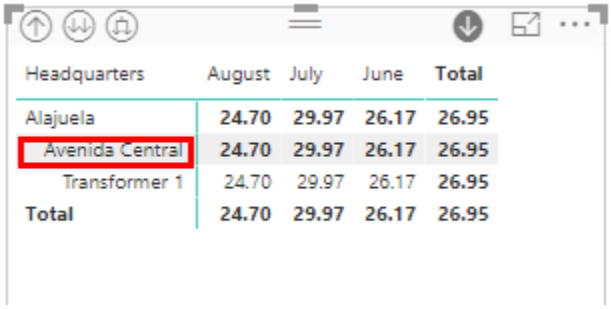
Now we will drill down into the hierarchy. Press the drill down button:

Headquarters	August	July	June	Total
Alajuela	24.70	29.97	26.17	**26.95**
Cartago	29.98	35.63	30.27	**31.98**
Heredia	13.39	16.19	14.02	**14.54**
San Jeronimo	5.40	6.15	5.79	**5.78**
San Jose	20.99	25.06	22.17	**22.75**
Total	**20.84**	**24.90**	**21.81**	**22.52**

Next, navigate through the layers by clicking on the Headquarters or going right click -> drill down:



Headquarters	August	July	June	Total
Alajuela	24.70	29.97	26.17	**26.95**
Cartago	29.98	35.63	30.27	**31.98**
Heredia	13.39	16.19	14.02	**14.54**
San Jeronimo	5.40	6.15	5.79	**5.78**
San Jose	20.99	25.06	22.17	**22.75**
Total	**20.84**	**24.90**	**21.81**	**22.52**

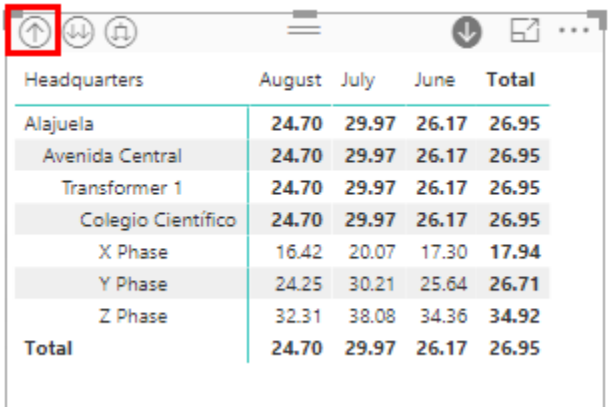


Headquarters	August	July	June	Total
Alajuela	**24.70**	**29.97**	**26.17**	**26.95**
Avenida Central	**24.70**	**29.97**	**26.17**	**26.95**
Transformer 1	24.70	29.97	26.17	**26.95**
Total	**24.70**	**29.97**	**26.17**	**26.95**

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

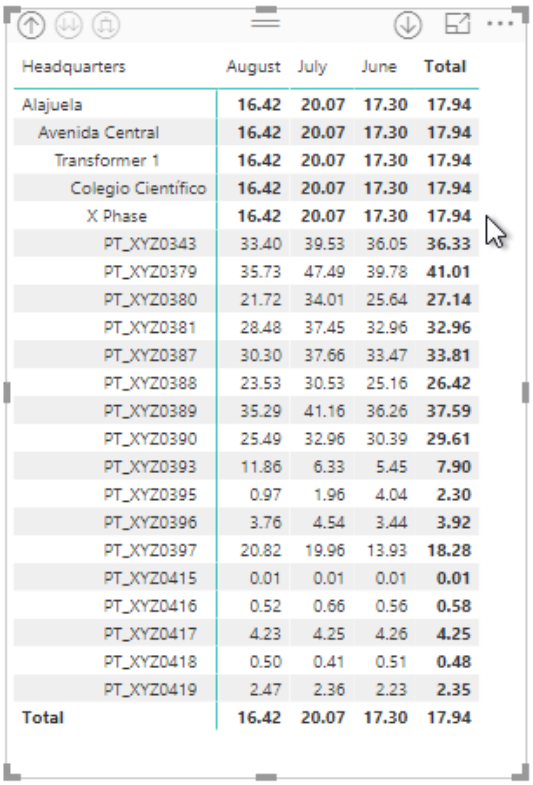
Headquarters	August	July	June	Total
Alajuela	**16.42**	**20.07**	**17.30**	**17.94**
Avenida Central	**16.42**	**20.07**	**17.30**	**17.94**
Transformer 1	**16.42**	**20.07**	**17.30**	**17.94**
Colegio Científico	**16.42**	**20.07**	**17.30**	**17.94**
X Phase	**16.42**	**20.07**	**17.30**	**17.94**
PT_XYZ0343	33.40	39.53	36.05	**36.33**
PT_XYZ0379	35.73	47.49	39.78	**41.01**
PT_XYZ0380	21.72	34.01	25.64	**27.14**
PT_XYZ0381	28.48	37.45	32.96	**32.96**
PT_XYZ0387	30.30	37.66	33.47	**33.81**
PT_XYZ0388	23.53	30.53	25.16	**26.42**
PT_XYZ0389	35.29	41.16	36.26	**37.59**
PT_XYZ0390	25.49	32.96	30.39	**29.61**
PT_XYZ0393	11.86	6.33	5.45	**7.90**
PT_XYZ0395	0.97	1.96	4.04	**2.30**
PT_XYZ0396	3.76	4.54	3.44	**3.92**
PT_XYZ0397	20.82	19.96	13.93	**18.28**
PT_XYZ0415	0.01	0.01	0.01	**0.01**
PT_XYZ0416	0.52	0.66	0.56	**0.58**
PT_XYZ0417	4.23	4.25	4.26	**4.25**
PT_XYZ0418	0.50	0.41	0.51	**0.48**
PT_XYZ0419	2.47	2.36	2.23	**2.35**
Total	**16.42**	**20.07**	**17.30**	**17.94**

Navigate back up the layers by clicking the Up Arrow on the top left of the visual:



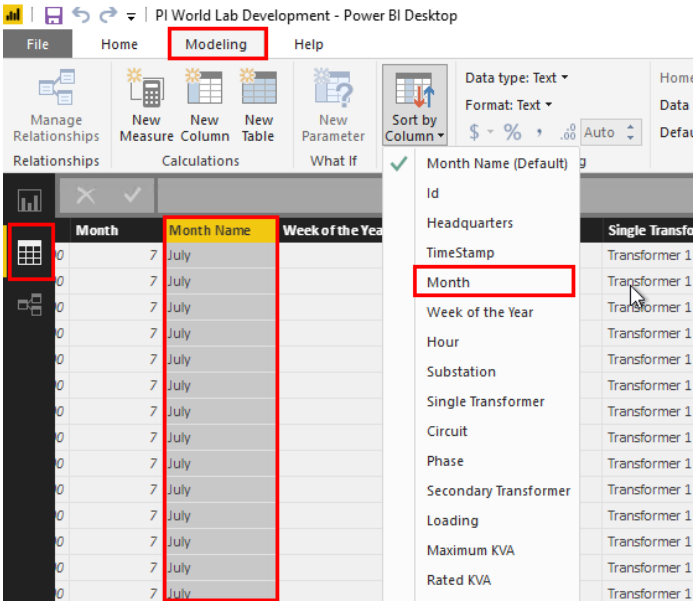
Headquarters	August	July	June	Total
Alajuela	**24.70**	**29.97**	**26.17**	**26.95**
Avenida Central	**24.70**	**29.97**	**26.17**	**26.95**
Transformer 1	**24.70**	**29.97**	**26.17**	**26.95**
Colegio Científico	**24.70**	**29.97**	**26.17**	**26.95**
X Phase	16.42	20.07	17.30	**17.94**
Y Phase	24.25	30.21	25.64	**26.71**
Z Phase	32.31	38.08	34.36	**34.92**
Total	**24.70**	**29.97**	**26.17**	**26.95**

Drill back down to the lowest level, and turn off Drill Down mode. This will allow us to filter the rest of the report by clicking on levels and transformers rather than drilling.



Headquarters	August	July	June	Total
Alajuela	**16.42**	**20.07**	**17.30**	**17.94**
Avenida Central	**16.42**	**20.07**	**17.30**	**17.94**
Transformer 1	**16.42**	**20.07**	**17.30**	**17.94**
Colegio Cientifico	**16.42**	**20.07**	**17.30**	**17.94**
X Phase	**16.42**	**20.07**	**17.30**	**17.94**
PT_XYZ0343	33.40	39.53	36.05	**36.33**
PT_XYZ0379	35.73	47.49	39.78	**41.01**
PT_XYZ0380	21.72	34.01	25.64	**27.14**
PT_XYZ0381	28.48	37.45	32.96	**32.96**
PT_XYZ0387	30.30	37.66	33.47	**33.81**
PT_XYZ0388	23.53	30.53	25.16	**26.42**
PT_XYZ0389	35.29	41.16	36.26	**37.59**
PT_XYZ0390	25.49	32.96	30.39	**29.61**
PT_XYZ0393	11.86	6.33	5.45	**7.90**
PT_XYZ0395	0.97	1.96	4.04	**2.30**
PT_XYZ0396	3.76	4.54	3.44	**3.92**
PT_XYZ0397	20.82	19.96	13.93	**18.28**
PT_XYZ0415	0.01	0.01	0.01	**0.01**
PT_XYZ0416	0.52	0.66	0.56	**0.58**
PT_XYZ0417	4.23	4.25	4.26	**4.25**
PT_XYZ0418	0.50	0.41	0.51	**0.48**
PT_XYZ0419	2.47	2.36	2.23	**2.35**
Total	**16.42**	**20.07**	**17.30**	**17.94**

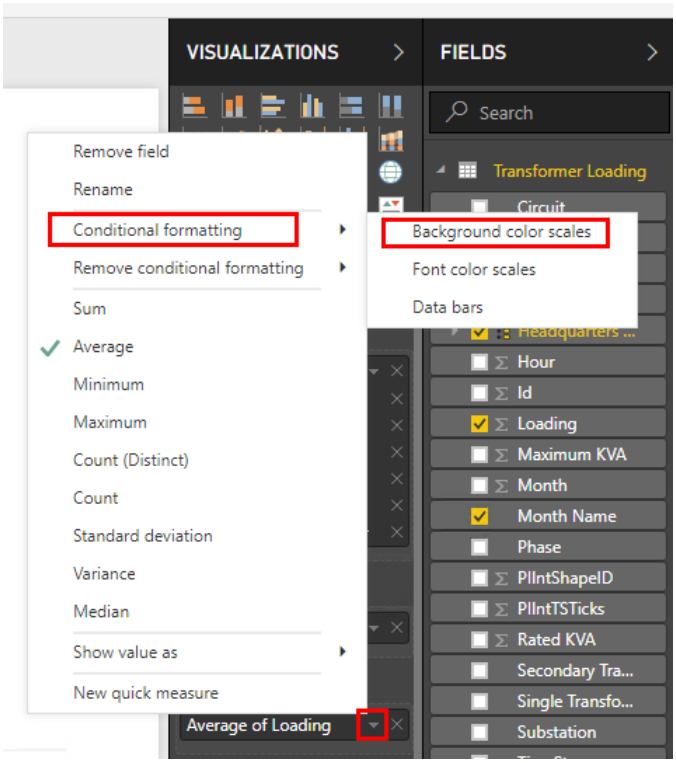
To put the Months in the correct order, we will sort the Month Name column in the data set by the Month column where the months are numbered. Go to the **Data View**, select **Month Name**, open the Modeling Ribbon, and Sort by Column -> Month:



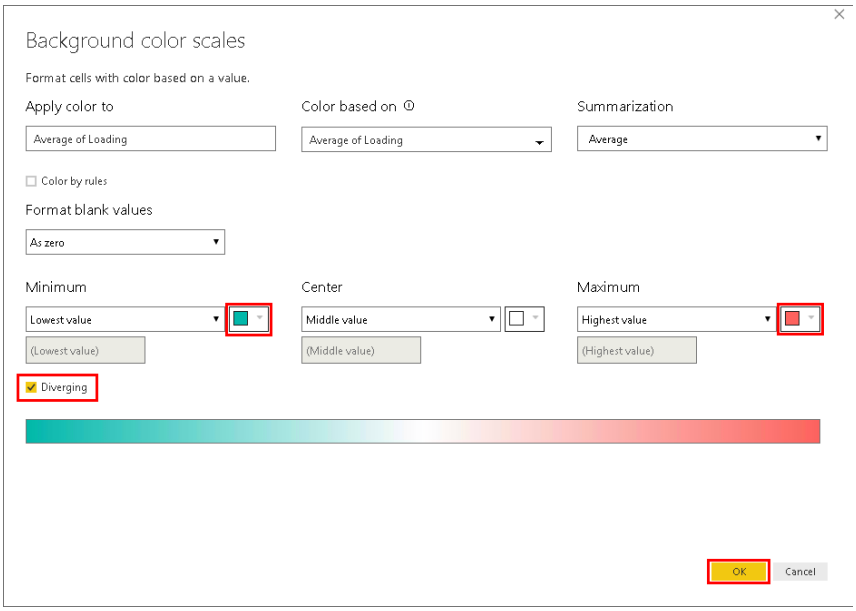
Go back to the **Report View**, the Month headers should now display in chronological order.

Headquarters	June	July	August	Total
Alajuela	26.17	29.97	24.70	26.95
Avenida Central	26.17	29.97	24.70	26.95
Transformer 1	26.17	29.97	24.70	26.95
Colegio Científico	26.17	29.97	24.70	26.95
X Phase	17.30	20.07	16.42	17.94
PT_XYZ0343	36.05	39.53	33.40	36.33
PT_XYZ0379	39.78	47.49	35.73	41.01
PT_XYZ0380	25.64	34.01	21.72	27.14
PT_XYZ0381	32.96	37.45	28.48	32.96
PT_XYZ0387	33.47	37.66	30.30	33.81
PT_XYZ0388	25.16	30.53	23.53	26.42
PT_XYZ0389	26.26	41.16	35.20	37.59
Total	26.17	29.97	24.70	26.95

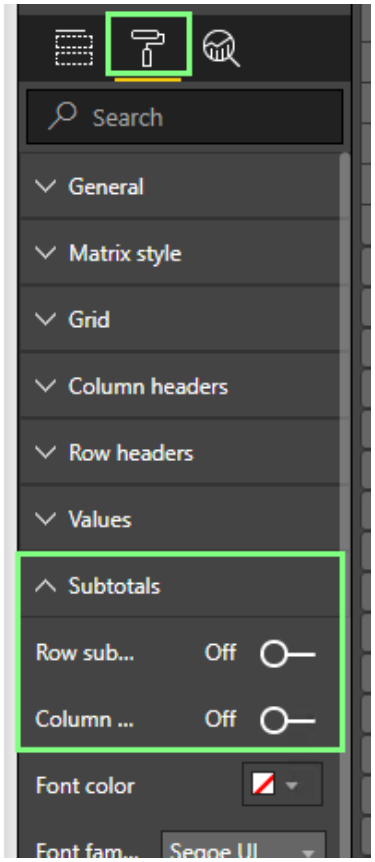
Next, we will add conditional formatting to highlight transformers with high loading. Conditional formatting is set from the Values field in the Visualizations Pane. Select the drop down by Average of Loading and click **Conditional Formatting -> Background Color Scales**:



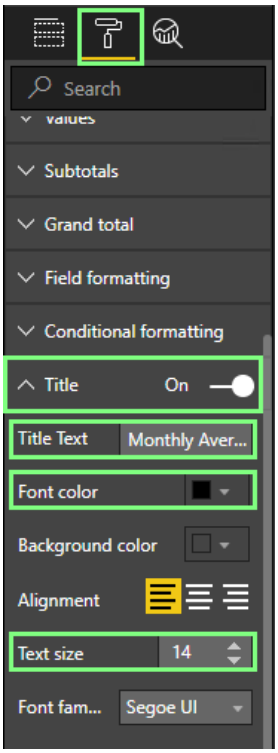
Reverse the minimum and maximum colors so that high numbers are Red. Enable the Diverging option. Click **OK**.



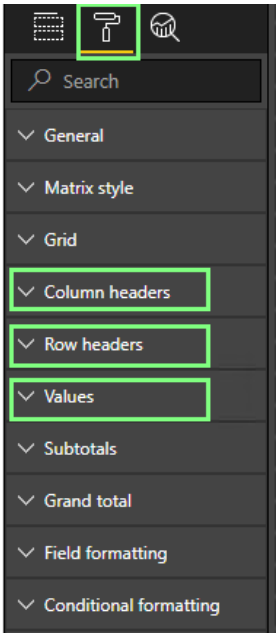
Turn off the **Subtotals** using the formatting options:



Add a **Title** to the Matrix using the Formatting Options. Call it Monthly Average Loading, change the color to black, and bump up the font size:



Bump up the font size of the **Column headers, Row headers, and Values**:



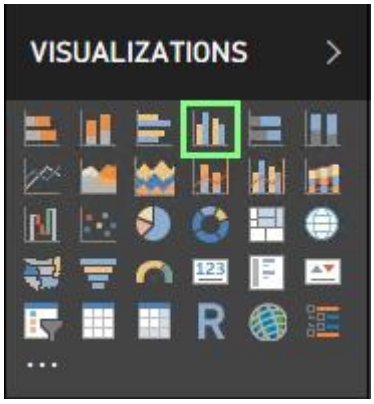
The matrix should look something like this:

Monthly Average Loading

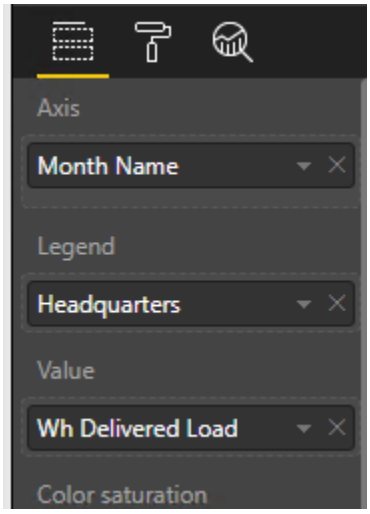
Headquarters	June	July	August
Alajuela			
Avenida Central			
Transformer 1			
Colegio Científico			
X Phase			
PT_XYZ0343	36.05	39.53	33.40
PT_XYZ0379	39.78	47.49	35.73
PT_XYZ0380	25.64	34.01	21.72
PT_XYZ0381	32.96	37.45	28.48
PT_XYZ0387	33.47	37.66	30.30
PT_XYZ0388	25.16	30.53	23.53
PT_XYZ0389	36.26	41.16	35.29
PT_XYZ0390	30.39	32.96	25.49
PT_XYZ0393	5.45	6.33	11.86
PT_XYZ0395	4.04	1.96	0.97
PT_XYZ0396	3.44	4.54	3.76
PT_XYZ0397	13.93	19.96	20.82
PT_XYZ0415	0.01	0.01	0.01
PT_XYZ0416	0.56	0.66	0.52
PT_XYZ0417	4.26	4.25	4.23
PT_XYZ0418	0.51	0.41	0.50
PT_XYZ0419	2.23	2.36	2.47

Watt-hours Delivered by Headquarters – Clustered Column Chart

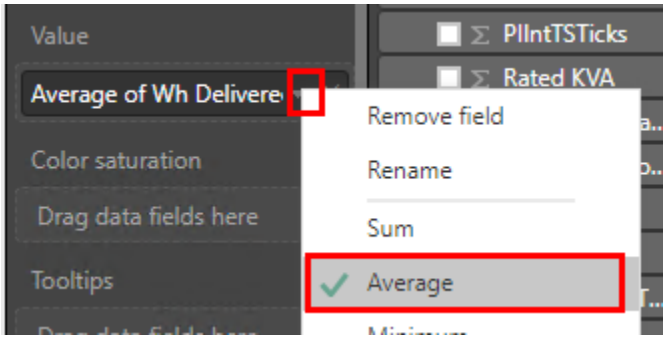
Next we'll configure a new visual to show Average watt-hours delivered on a per month basis. Click some blank space and add a Clustered Column Chart:



Use **Month Name** for the Axis, **Headquarters** for the Legend, and **Wh Delivered** as the Value:

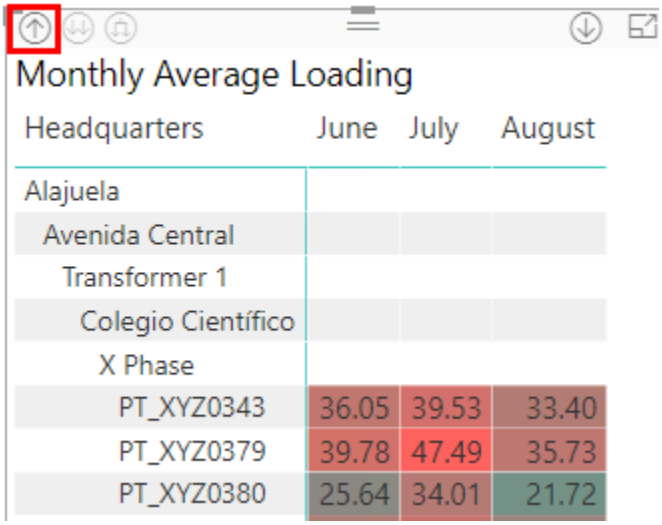


Summarize the Wh Delivered as an **Average**:



Note that only 1 headquarters is shown. **This is because the visual is being filtered by the Matrix configuration.**

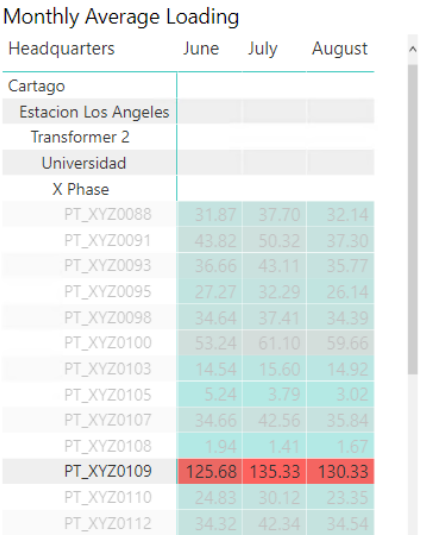
In the **Matrix,** drill up to the top level by repeatedly clicking the Drill Up button:

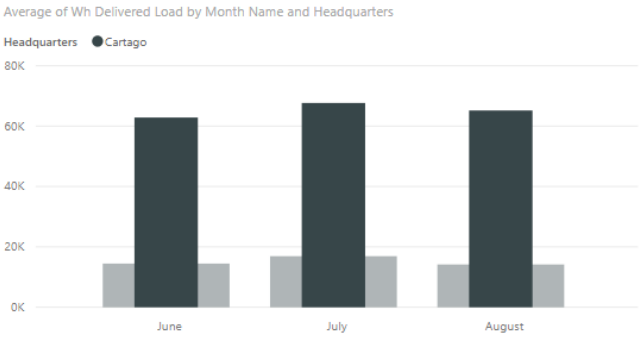


Eventually we'll see the summary broken down by headquarters and all 5 headquarters will be shown in the Clustered Column Chart.

Note that Cartago seems to be the HQ with the highest transformer loads. **Turn on Drill mode** and drill down and follow the highlighting (drill down wherever the load is highest) to discover that transformer PT_XYZ0109 is consistently overloaded. Consider that these are averages, not maximums.

Turn off drill mode and click on PT_XYZ0109 to see the month to month watt-hours delivered for this particular transformer.



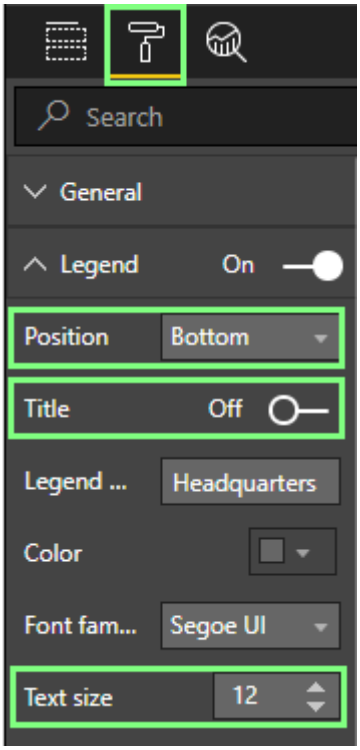


Optionally search for PT_XYZ0109 in PI System Explorer and note how many meters it is responsible for.

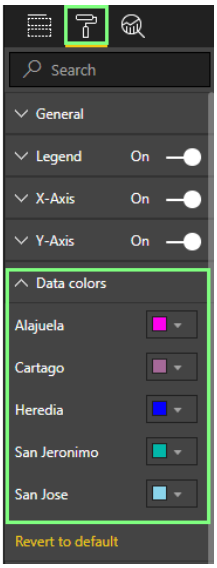
In Power BI, Click PT_XYZ0109 again to deselect it and remove the filtering.

Select the Clustered 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 “Average Wh Delivered by Headquarters” with black text, increase the text size
- Change the colors used for the headquarters. **You will have to Drill Up in the matrix to get all the headquarters to show up first.**

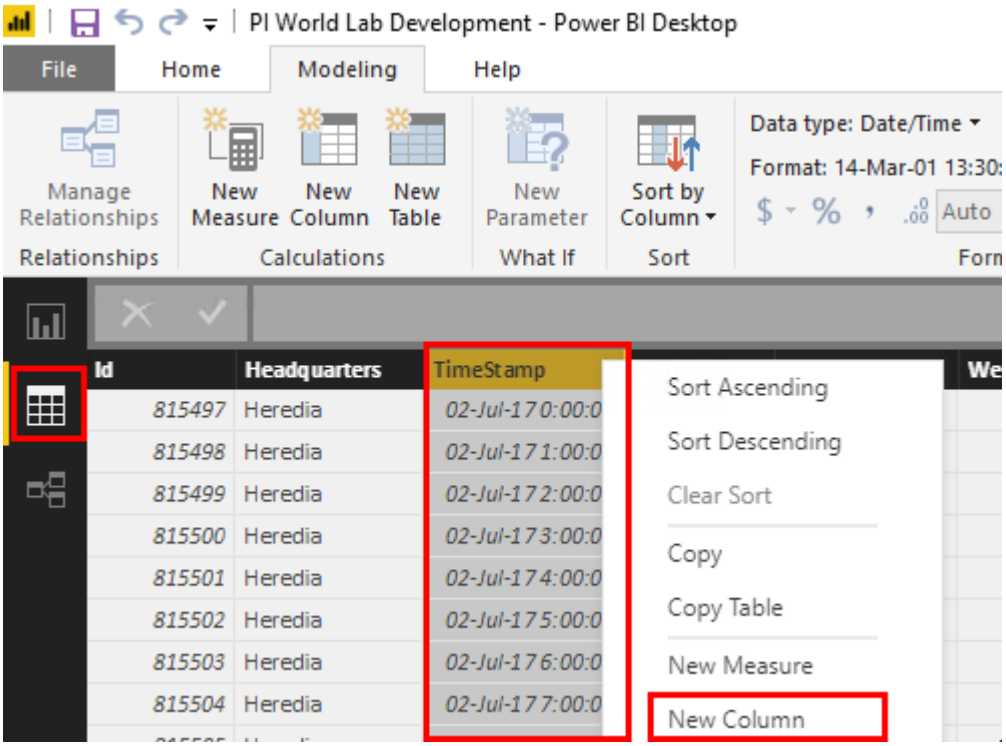


Watt-hours Delivered by Weekday – Line Chart

We also want to see how the delivered watt-hours change throughout the day and on different days of the week. We'll display this information using a Line Chart.

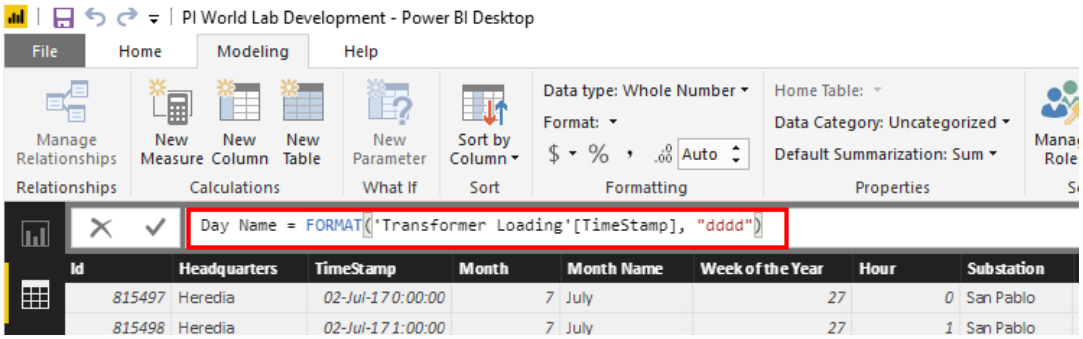
First some setup. We do not have a day of the week column in the original data set like we did for the Month, so we will add one using a **DAX** formula.

Go to the Data View. Select **any** column, then go right click -> New Column:



Add the following DAX formula in the formula bar and click the check box or hit enter. Note that this sets the column name to “Day Name”.

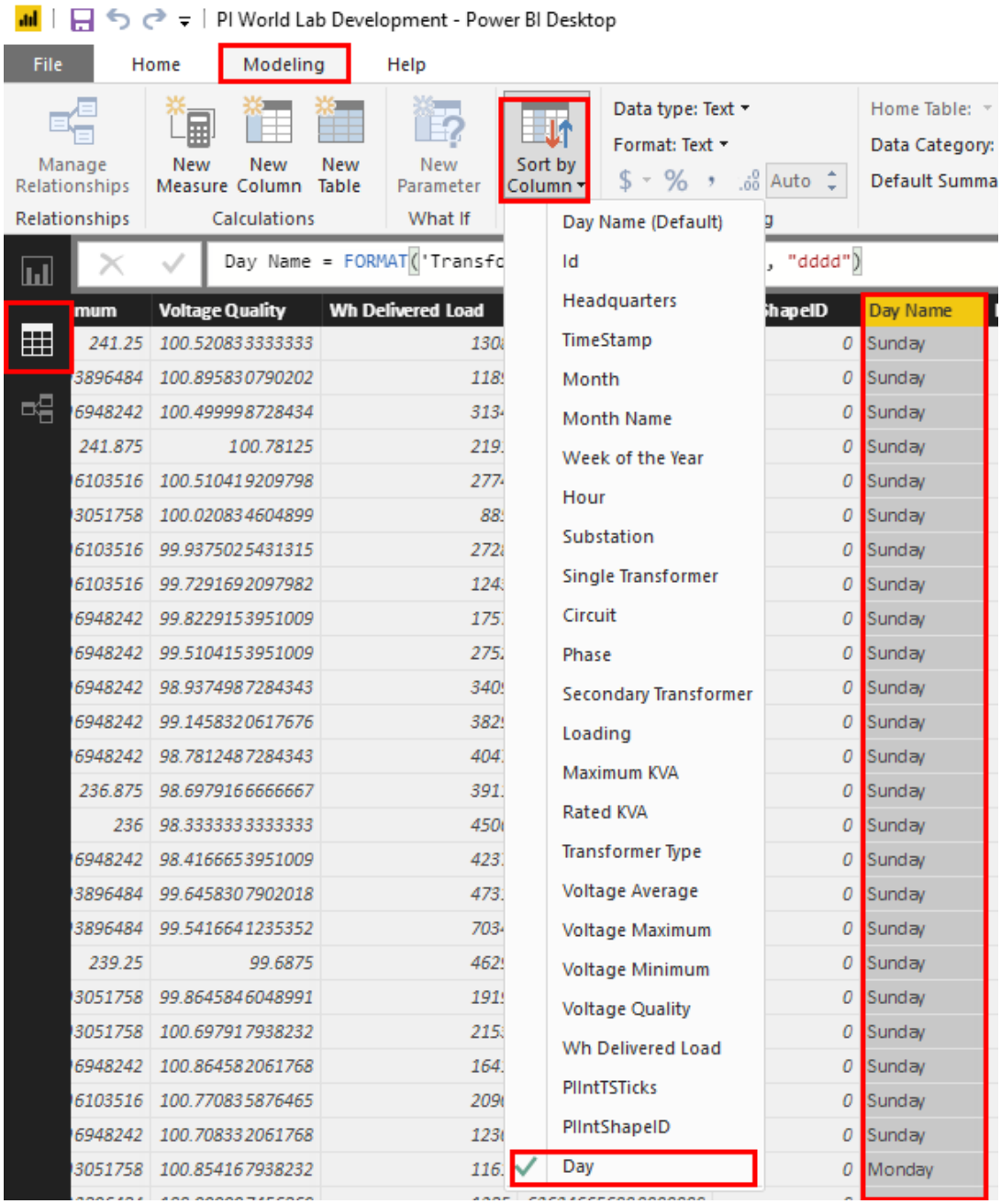
```
Day Name = FORMAT('Transformer Loading'[TimeStamp], "dddd")
```



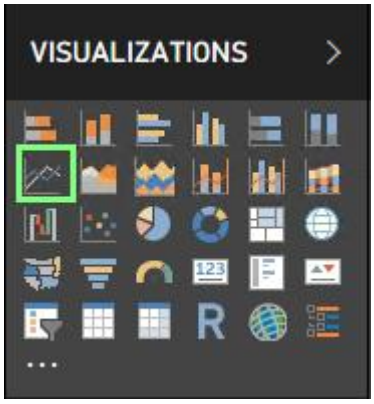
In order for the days of the week to display in the correct order, we will add another column for the numerical day of the week using the WEEKDAY() function:

```
Day = WEEKDAY('Transformer Loading'[TimeStamp])
```

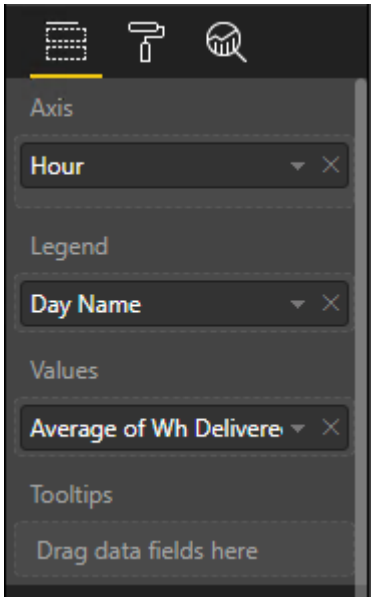
We can then sort the Day Name column by this new Day column:



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



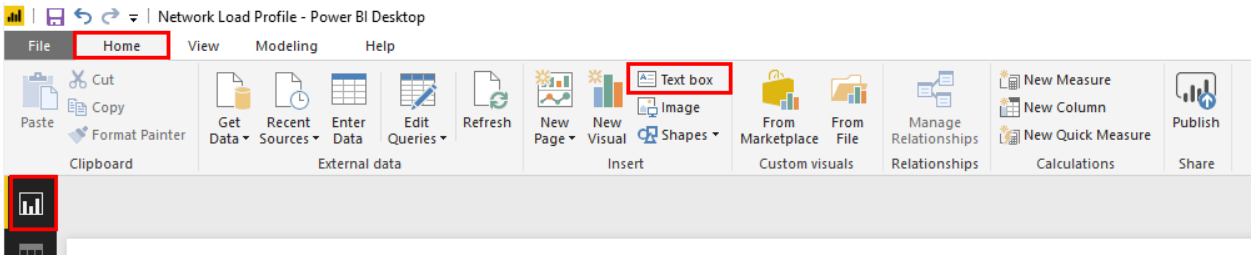
Use **Hour** as the Axis, **Day Name** as the Legend, and **Average of Wh Delivered** as the values (summarize Wh Delivered as an **Average**):



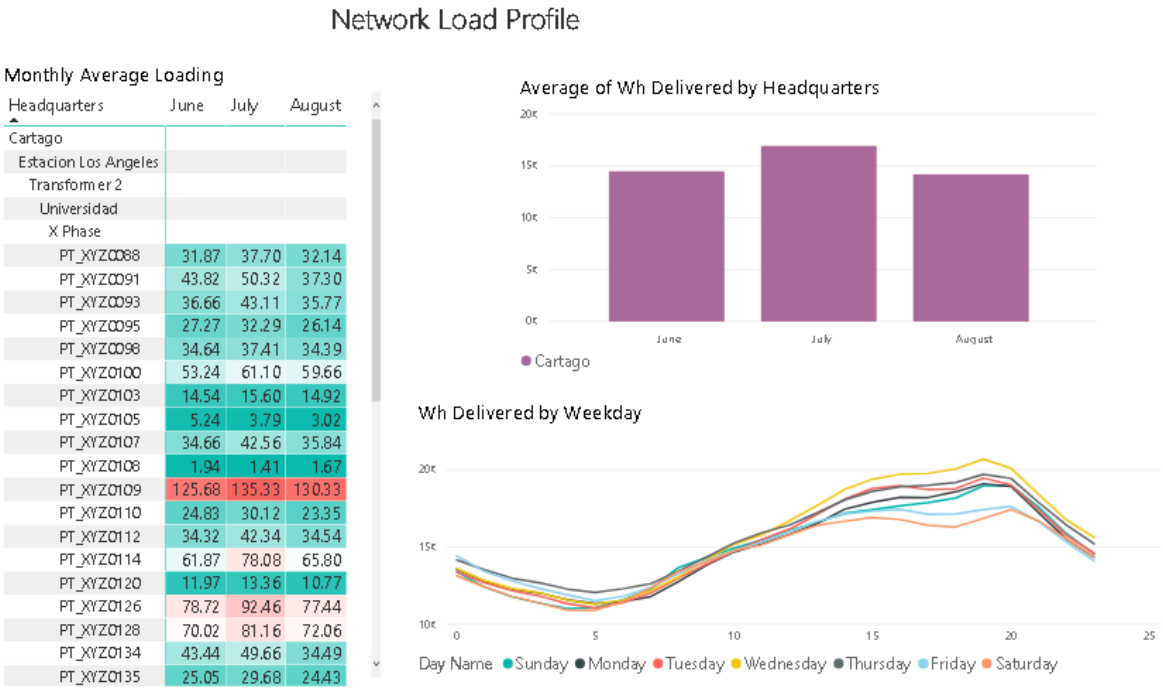
Adjust the formatting of the Line Chart:

- Move the Legend to the bottom
- Change the title to "Average Wh by Weekday", text color black, and increase the text size of the title to match the other visuals
- Optionally change the trend colors

Add a report title by inserting a text box:



The end result should look something like this:



Finally experiment with drilling up and down the hierarchy and filtering the report.

5.2.2 Directed Activity – Transformer Loading Analysis

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

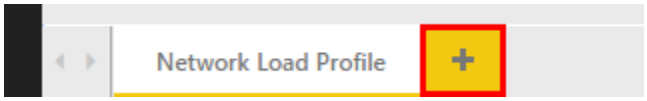
Objectives:

- Configure a **Measure** to calculate service hours
- Configure a **Group** to create bins for different load ranges which can then be used for highlighting and filtering
- Configure a **Stacked Bar Chart** to display the service hours spent in each Load Range by circuit
- Configure a **Table** to show the top 20 transformers by average Loading
- Configure a **Slicer** to filter by Month

In this exercise, we will analyze transformer loading characteristics. The goal is to assess the number of service hours spent in various high load conditions to better understand which transformers are at risk of failing and also assess whether a given transformer should be replaced with one that has a higher capacity.

Approach:

Start a new Page by clicking the **New Page** icon at the bottom of the existing report:

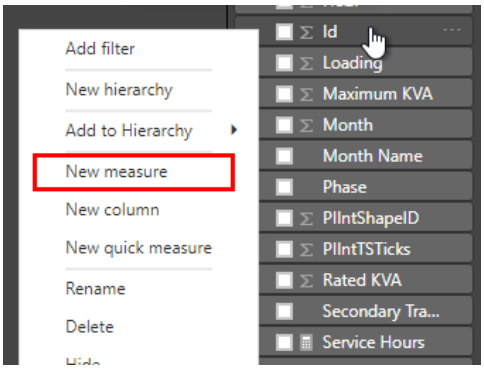


Rename the Page to **Transformer Loading**.

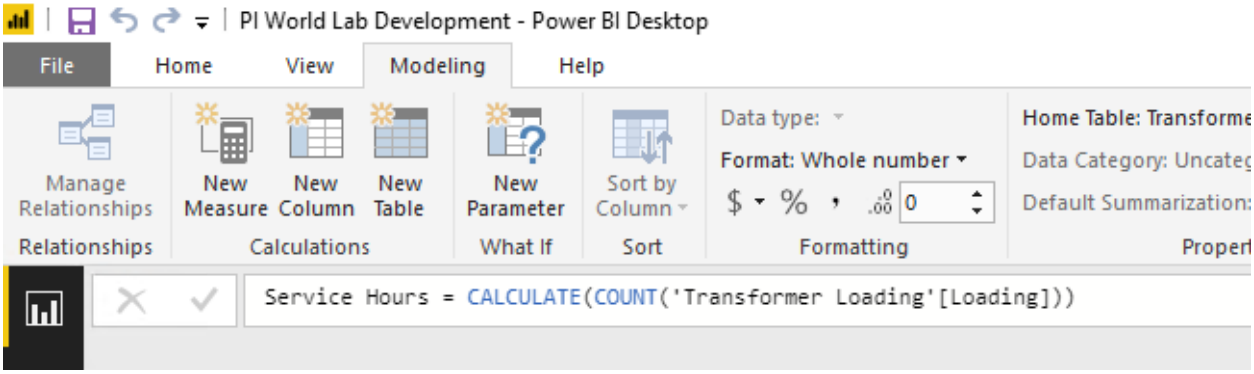
Service Hours

The first thing to do is configure a Measure to calculate service hours. Each row in the data set represents 1 hour, so we can simply count the number of rows that have been filtered through user selection. This should make a bit more sense when it all comes together.

Right click **any** of the fields from the Fields list and select **New measure**:


Add filter
New hierarchy
Add to Hierarchy
New measure
New column
New quick measure
Rename
Delete
Id
Loading
Maximum KVA
Month
Month Name
Phase
PIIntShapeID
PIIntTSTicks
Rated KVA
Secondary Tra...
Service Hours

Enter the below formula into the configuration box:



The raw text is given below for convenience.

```
Service Hours = CALCULATE(COUNT('Transformer Loading'[Loading]))
```

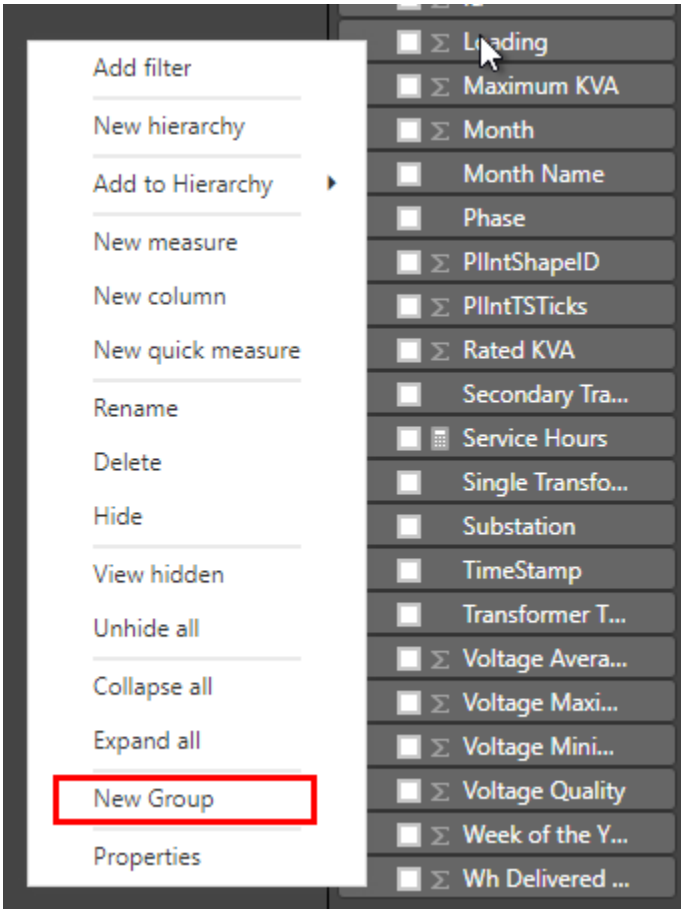
A Note on Measures vs Calculated Columns

From a configuration perspective, Measures and Calculated Columns are configured similarly so the distinction may not be obvious. Measures and calculated columns both use DAX expressions. The difference is the context of evaluation. A measure is evaluated on the fly using a subset of data, whereas a calculated column is pre-calculated at the row level within the table it belongs to. A simple way to put it is that Measures take into account the filtering that has been set by the end user of the report (the stuff they've clicked on), while calculated columns are computed row by row and are not influenced by the report filtering.

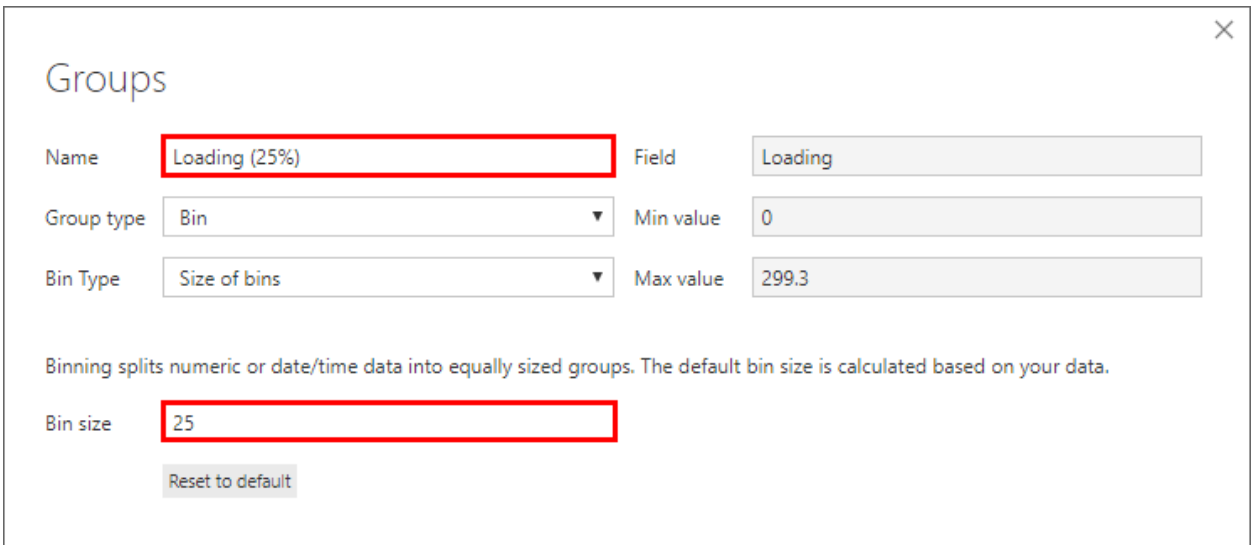
Loading Groups

Different ranges for Loading will be grouped into bins representing different Load Ranges. It is normal for transformers to be operating at Loads higher than 100% of their rating, but loads in the range of 125% and higher are potentially cause for concern. In order to calculate service hours in the different Load Ranges, a group must be configured in the data set for filtering and counting by the Service Hours Measure.

Right click on **Loading** and select **New group**.

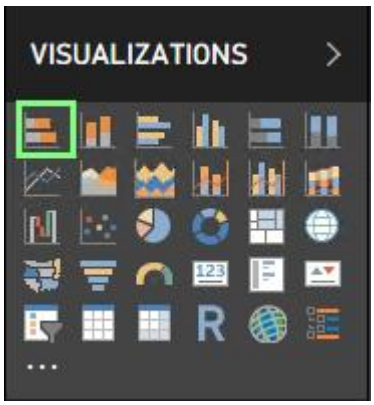


Change the name to **Loading** (**25%**) and set the bin size to 25, then click **OK**.

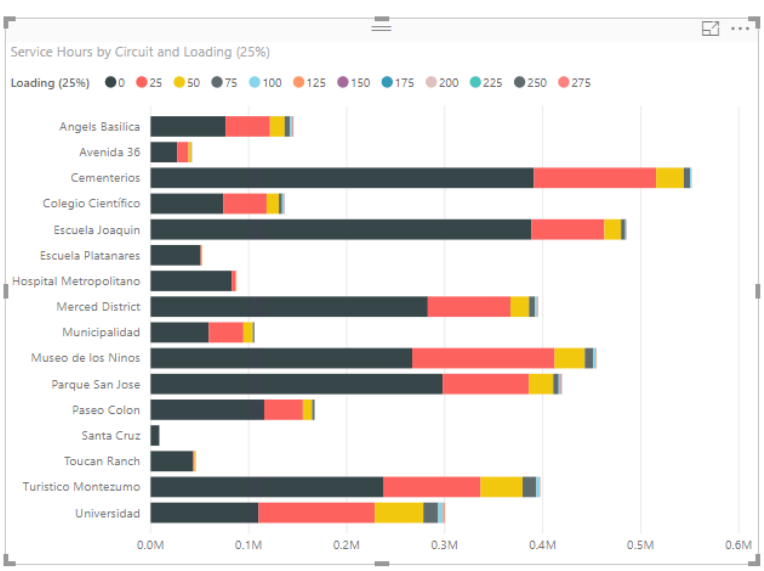


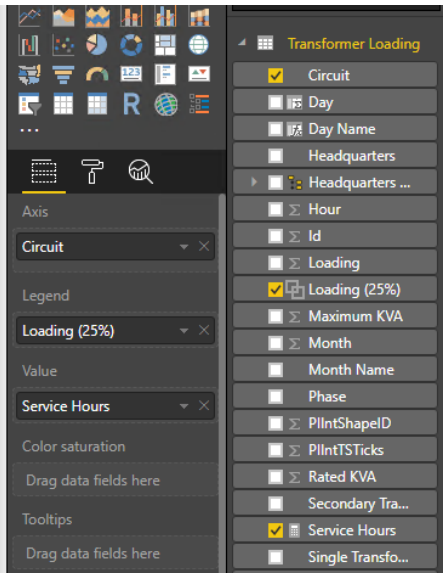
Loading by Circuit – Stacked Bar Chart

Now we can begin to configure the report. Click some empty space and add a Stacked Bar Chart:



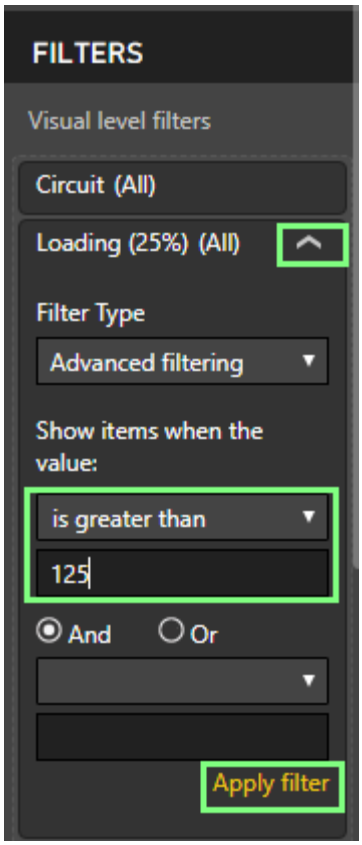
With the Stacked Bar Chart selected, drag and drop Fields from the data set into the field configuration boxes. Use **Circuit** for the Axis, **Loading (25%)** for the Legend, and **Service Hours** for the Value:



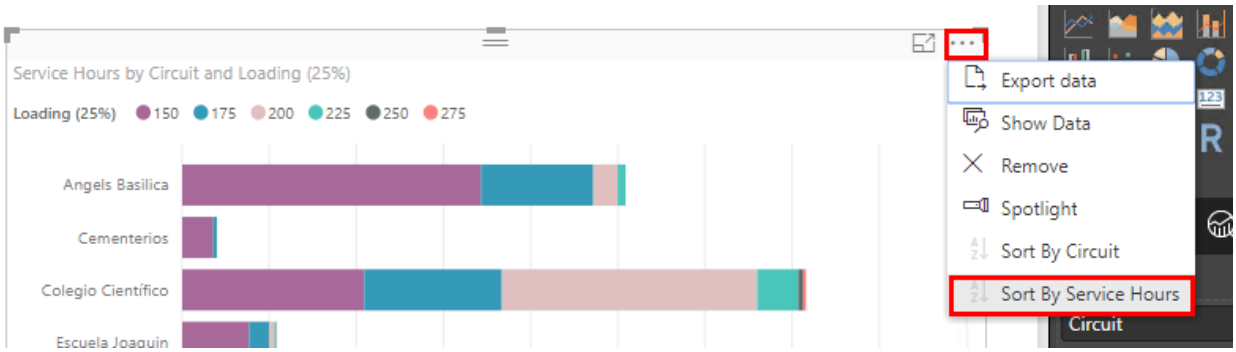


Next we will apply some formatting and filters to make the data set more manageable. We'll change the color scheme and only show Loadings greater than 125%, since loads in the normal range are not of interest to us.

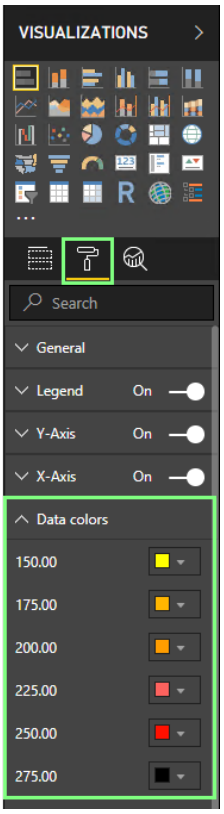
In the Visualizations Pane, with Fields Selected, scroll down to the Visual Level Filters and filter for Loading **greater than** 125%. Be sure to click **Apply Filter**:



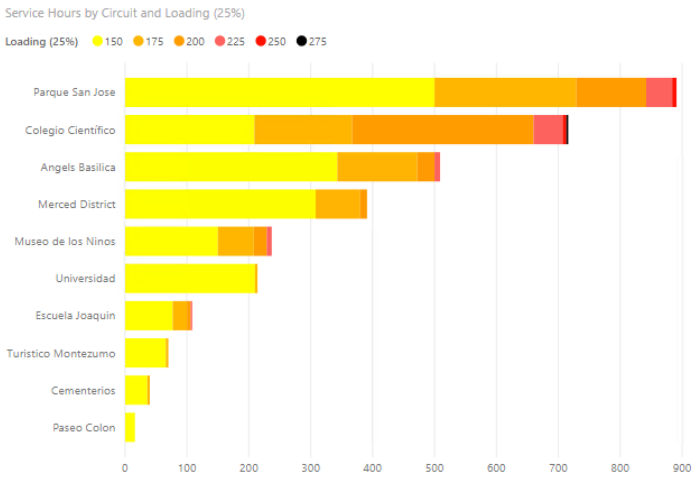
Next go to the Visualization Options and **sort by Service Hours**:



Next change the color scheme. With the Visualization selected, click the Format Icon in the Visualization Pane and adjust the colors to better convey the severity of the loading levels.



The stacked bar chart should now look something like this:

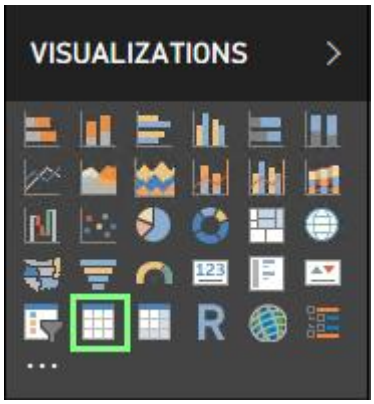


Service Hours and Average Load by Transformer – Table

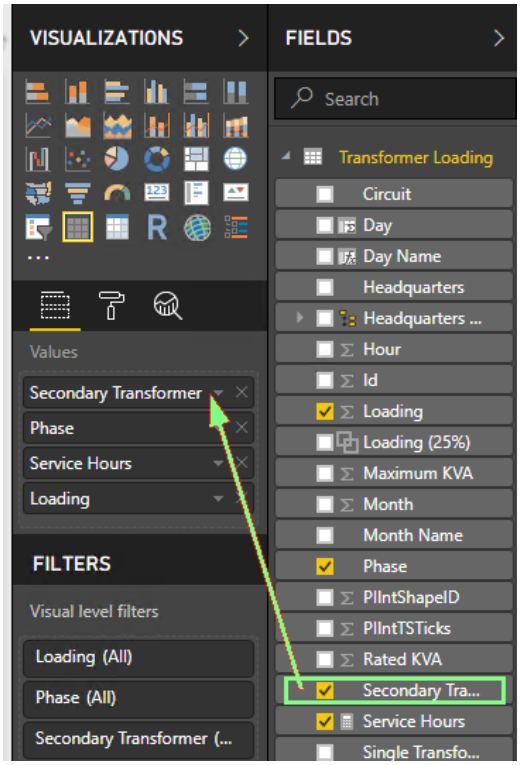
The next visual we will add is a basic table showing the Transformer Name, Phase, Service Hours, and Average Load. We will then filter the table to show only the top 20 transformers by average load. This will give us a quick indicator of which Transformers are consistently overloaded.

Click some blank space on the canvas to deselect any visuals, otherwise you will accidentally convert the Stacked Bar Chart to a Table.

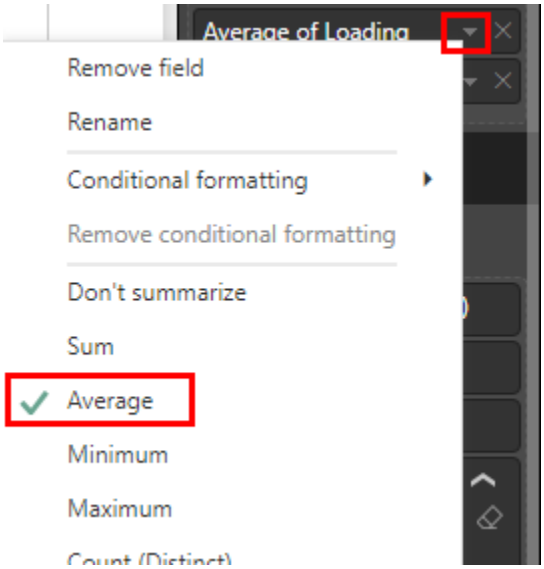
Create a **Table**:



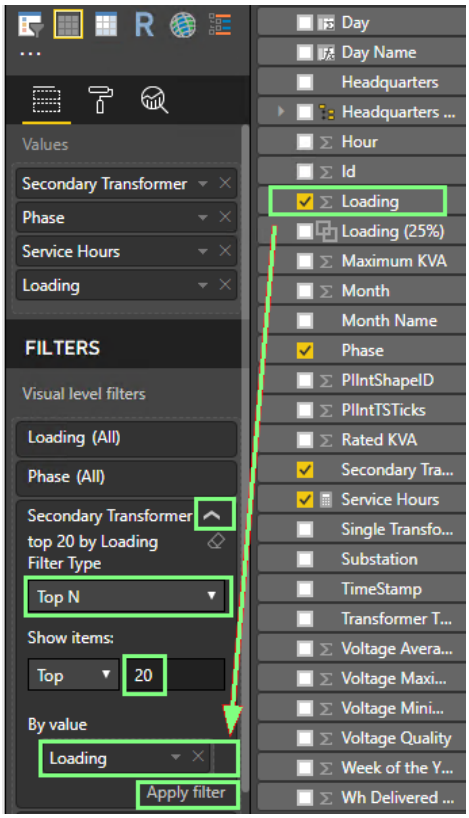
Drag and drop the **Secondary Transformer, Phase, Service Hours**, and **Loading** Fields into the Values section:



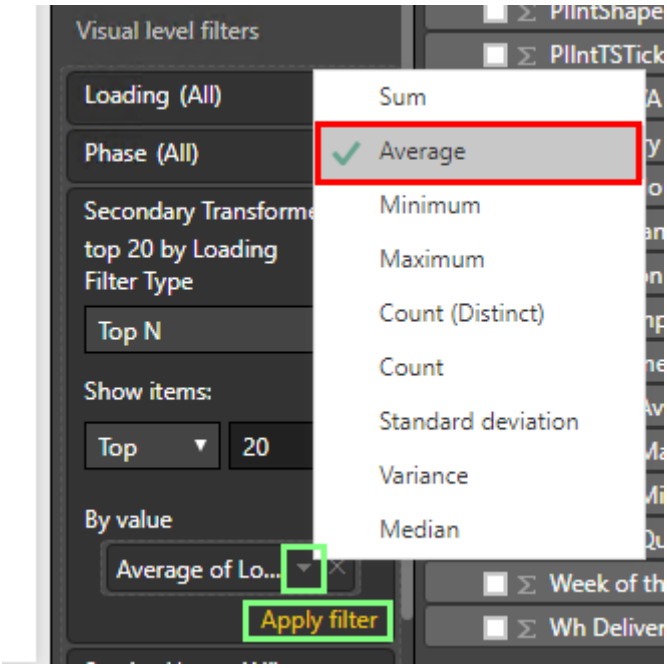
Change the **Loading** Value to summarize by **Average**:



Change the Visual Level Filters to Show the **Top 20** Transformers by **Average Loading**. Be sure to drag and drop **Loading** to the by value field, and be sure to click **Apply filter**.

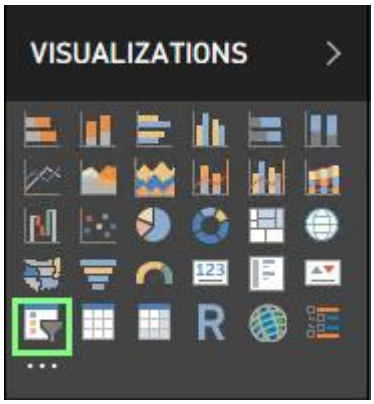


Under By value, change the loading summary to **Average**, and don't forget to **click Apply Filter again**:



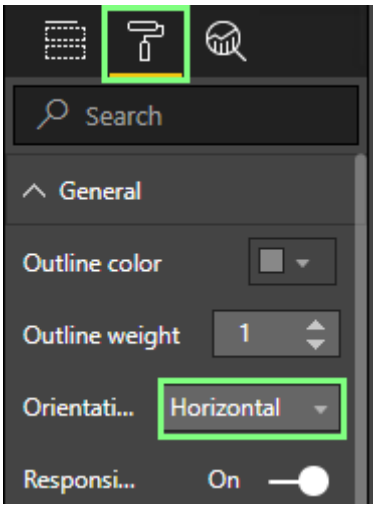
Filtering by Month – Slicer

We'll now add a basic Slicer to filter by Month. Click some blank space and then add a Slicer:

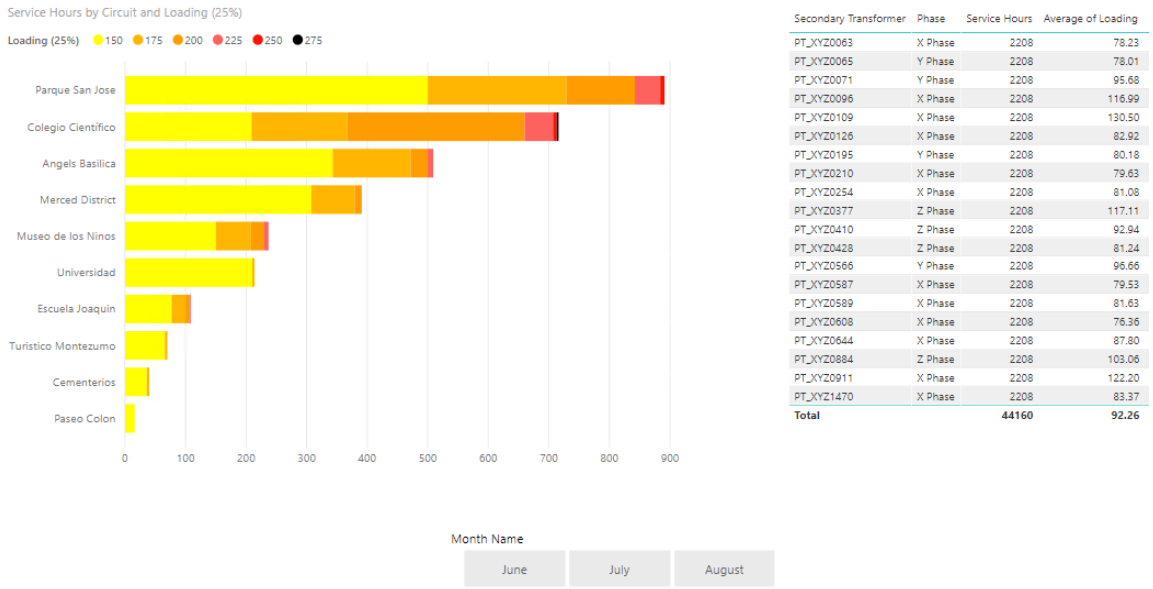


Drag **Month Name** to the field list.

Go into the formatting options and change the orientation to horizontal to change the look of the Slicer:

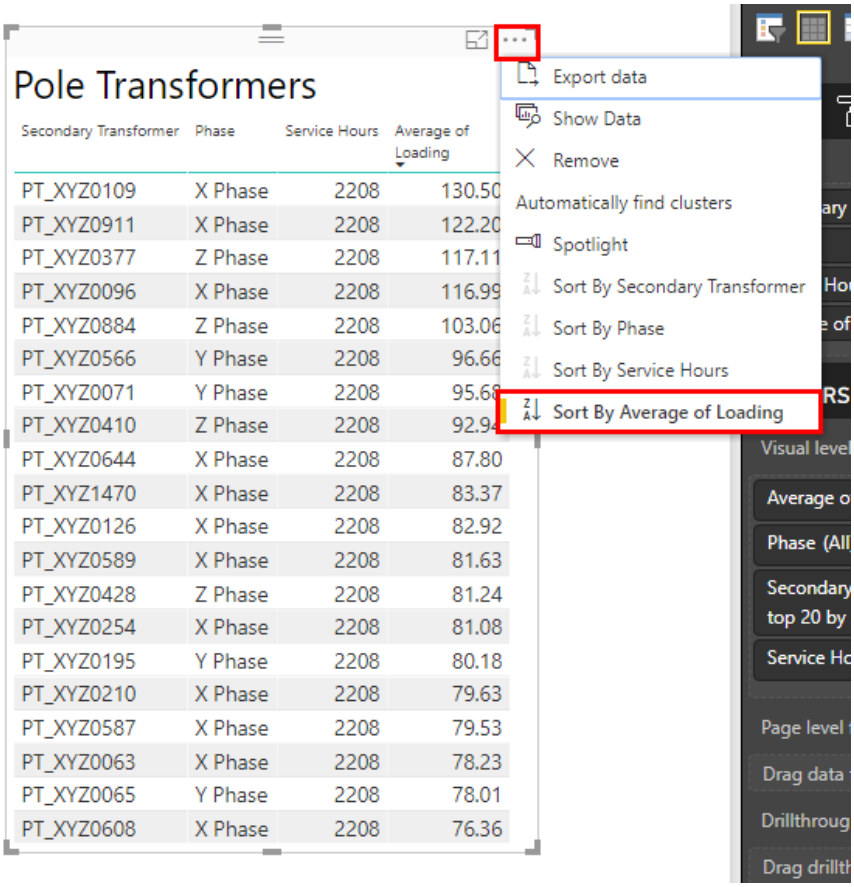


The report should now look something like this:



Secondary Transformer	Phase	Service Hours	Average of Loading
PT_XYZ0063	X Phase	2208	78.23
PT_XYZ0065	Y Phase	2208	78.01
PT_XYZ0071	Y Phase	2208	95.68
PT_XYZ0096	X Phase	2208	116.99
PT_XYZ0109	X Phase	2208	130.50
PT_XYZ0126	X Phase	2208	82.92
PT_XYZ0195	Y Phase	2208	80.18
PT_XYZ0210	X Phase	2208	79.63
PT_XYZ0254	X Phase	2208	81.08
PT_XYZ0377	Z Phase	2208	117.11
PT_XYZ0410	Z Phase	2208	92.94
PT_XYZ0428	Z Phase	2208	81.24
PT_XYZ0566	Y Phase	2208	96.66
PT_XYZ0587	X Phase	2208	79.53
PT_XYZ0589	X Phase	2208	81.63
PT_XYZ0608	X Phase	2208	76.36
PT_XYZ0644	X Phase	2208	87.80
PT_XYZ0884	Z Phase	2208	103.06
PT_XYZ0911	X Phase	2208	122.20
PT_XYZ1470	X Phase	2208	83.37
Total		**44160**	**92.26**

Sort the table by Average Loading:



Click the bars on the Loading by Circuit chart and the Month slicer buttons and note how the service hours and transformers for that load range update on the table.

We will save formatting until the end in case we need to save time, but feel free to adjust the formatting and add a title.

Linking to PI Vision

We have a PI Vision display for Transformers that we can link to from this report. We will utilize PI Vision URL Parameters to set the same Transformer in the PI Vision display that the user clicks on in the Power BI report. The URL parameters reference guide can be found in the PI Live Library.

From within the client virtual machine, Navigate to:
https://pisrv01.pischool.int/PIVision/#/Displays/3/TransformerTrends

Take the above URL and append the following string to it in a text editor, then paste the URL into Chrome:

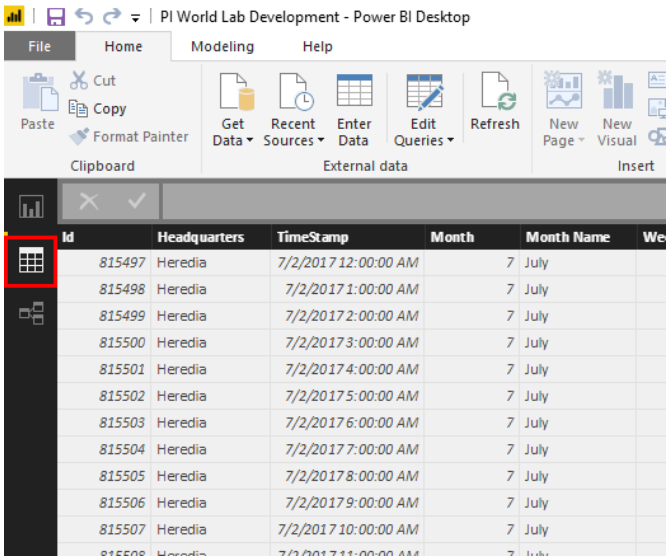
?Asset=\\PISRV01\Distribution Network\Secondary Transformers\PT_XYZ0046

Transformer PT_XYZ0046 should be the selected Asset in the TransformerTrends display.

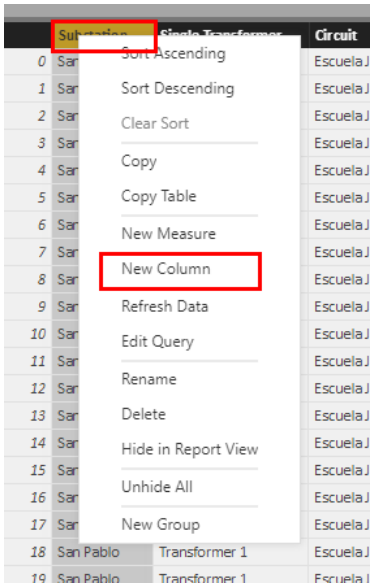
Note that the **?Asset** parameter denotes the path to the Asset in the PI AF hierarchy.

Once that is working, configure a Calculated Column to concatenate the URL with the Transformer asset path.

Go to the **Data** Tab:

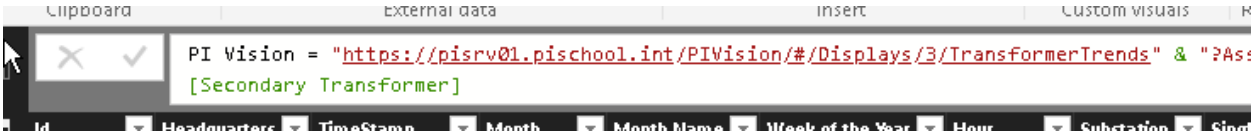


Right click on the header of any column and select **New column**:



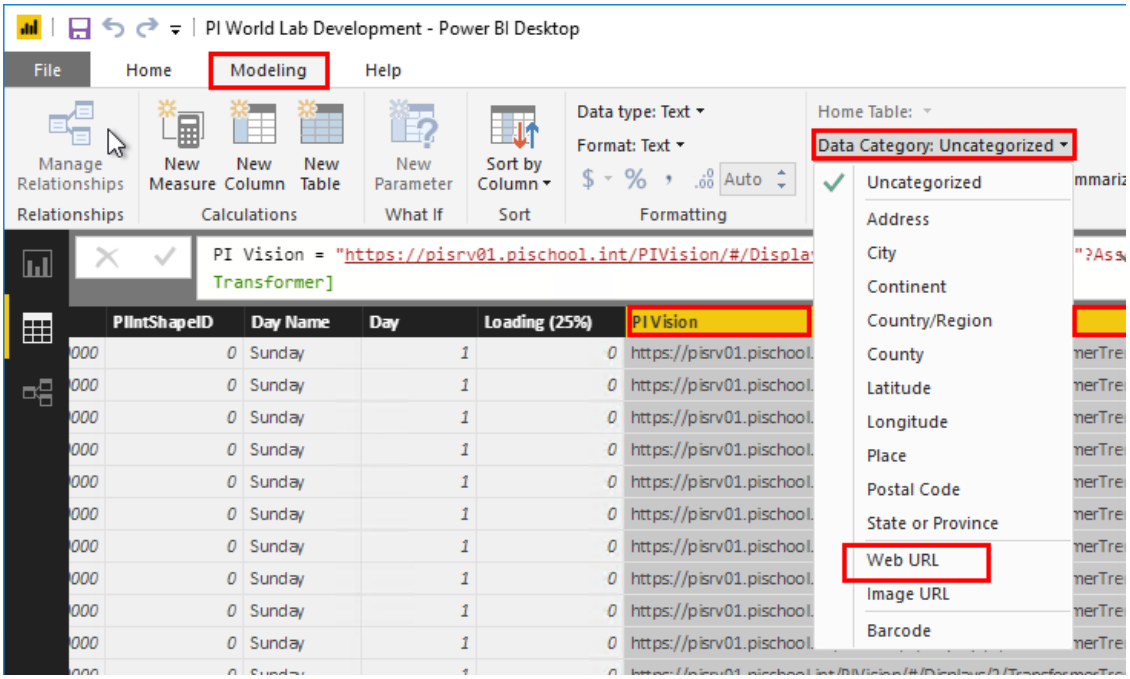
For the DAX formula, enter the following and **hit enter or click the checkmark**:

PI Vision = "https://pisrv01.pischool.int/PIVision/#/Displays/3/TransformerTrends" & "?Asset=\\PISRV01\Distribution Network\Secondary Transformers\" & 'Transformer Loading'[Secondary Transformer]

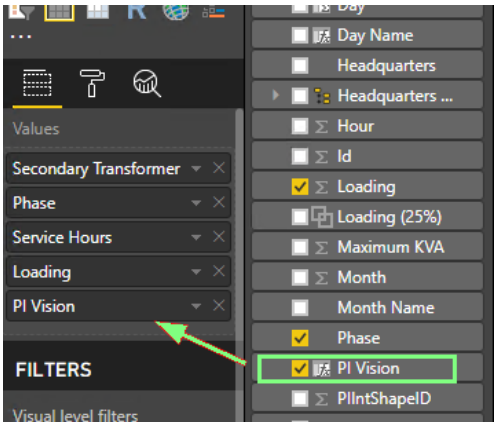


Next scroll all the way to the right and find the **PI Vision column**, then select it.

Go to the **Modeling Ribbon**, and change the **Data Category** to Web URL.

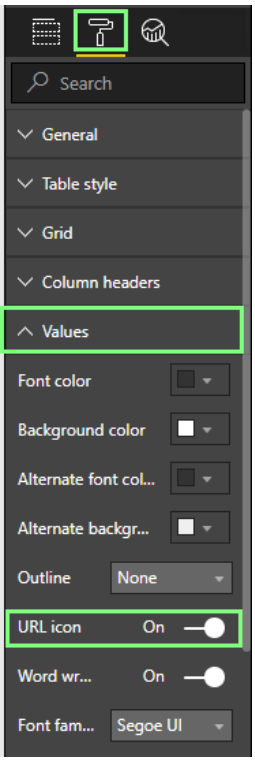


Now go back to the **Report Tab** and select the Table, then drag and drop the **PI Vision** field as one of the table values



The links are now displayed, and they work, but they are not pretty to look at. Luckily Power BI has a feature that addresses this.

Go into the **Formatting Options,** scroll down to the Values section, and turn on the URL icon:



Now the links look much cleaner:

Secondary Transformer	Phase	Service Hours	Average of Loading	PI Vision
PT_XYZ0109	X Phase	2208	130.50	
PT_XYZ0911	X Phase	2208	122.20	
PT_XYZ0377	Z Phase	2208	117.11	
PT_XYZ0096	X Phase	2208	116.99	
PT_XYZ0884	Z Phase	2208	103.06	
PT_XYZ0566	Y Phase	2208	96.66	
PT_XYZ0071	Y Phase	2208	95.68	
PT_XYZ0410	Z Phase	2208	92.94	
PT_XYZ0644	X Phase	2208	87.80	
PT_XYZ1470	X Phase	2208	83.37	
PT_XYZ0126	X Phase	2208	82.92	
PT_XYZ0589	X Phase	2208	81.63	
PT_XYZ0428	Z Phase	2208	81.24	
PT_XYZ0254	X Phase	2208	81.08	
PT_XYZ0195	Y Phase	2208	80.18	
PT_XYZ0210	X Phase	2208	79.63	
PT_XYZ0587	X Phase	2208	79.53	
PT_XYZ0063	X Phase	2208	78.23	
PT_XYZ0065	Y Phase	2208	78.01	
PT_XYZ0608	X Phase	2208	76.36	
Total		**44160**	**92.26**	

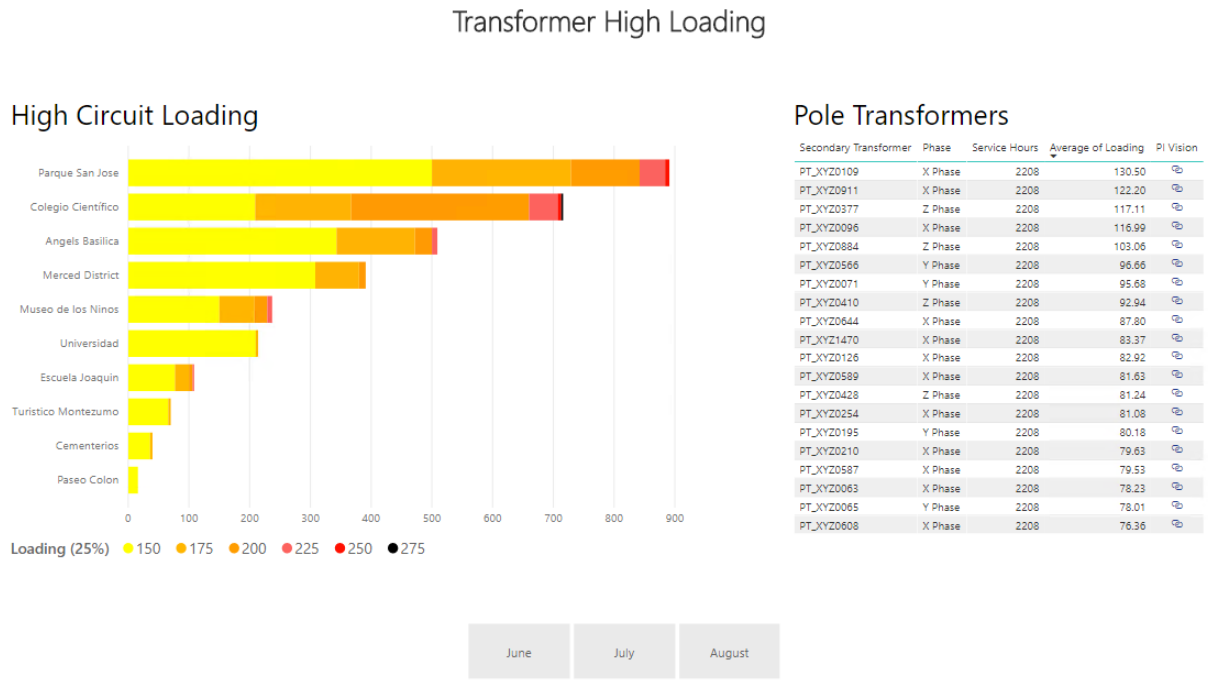
Test the links to confirm that the PI Vision display is launched and the correct transformer is set.

(Optional) Formatting

Take some time to apply formatting to make the report more visually appealing and easier to read.

1. Add a Title text box for the report (Home Ribbon -> Insert Text Box)
2. Add titles for the Stacked Bar Chart and Table, change the font color to black and bump up the font size
3. Adjust the sizes of the header text
4. Resize the columns
5. Move the Legend on the Stacked Bar Chart to the bottom
6. Remove the totals from the Table
7. Remove the header from the Slicer

The end result should look something like this:



Finally test the links and experiment with filtering the report.

5.2.3 Directed Activity – Circuit Voltage Quality

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objectives:

- Configure a **Hierarchy Slicer**
- Configure **Clustered Column Charts** to represent voltage quality distributions
- Configure **Multi-row Cards** to show some basic statistics

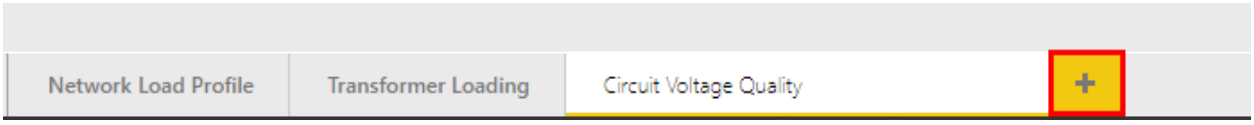
Utilities are required by the Public Utilities Commission to deliver power to customers at a certain voltage quality. We want to check to see if the various circuits in the network are within this range.

Voltage Quality is essentially the ratio of the delivered voltage to the nominal voltage. Our fictitious distribution company aspires to operate at an average quality slightly above 100%. Recall that we are using Asset Analytics to aggregate the average voltage quality from the child meters.

We will configure a report to determine whether this is true across all transformers by plotting the voltage distributions on a per-phase basis. We will use a Hierarchy Slicer for filtering and display some basic statistics using Multi-row Cards.

Approach:

Start a new sheet and name it Circuit Voltage Quality:



Downloading the Hierarchy Slicer

For this part, there is no need to visit the web site, sign up, or download the file. We have downloaded the file for use in class so that students do not need to sign up!

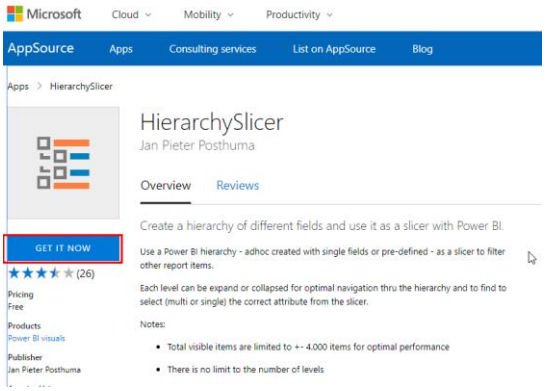
The Hierarchy Slicer is a custom visual that can be used to filter reports and mimic the PI AF hierarchy. This is similar to the PI TreeView from PI WebParts.

Most custom visuals can be found on Microsoft AppSource. We will briefly go through the procedure of how one would normally obtain a custom visual.

Search for a custom visual on Google or within AppSource and you'll arrive at a page like this:

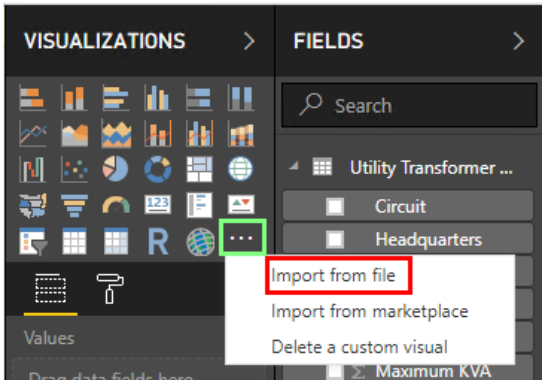
https://appsource.microsoft.com/en-us/product/power-bi-visuals/WA104380820?tab=Overview

At which point you would click Get It Now, sign in using your work or school account, and download the .pbiviz file.

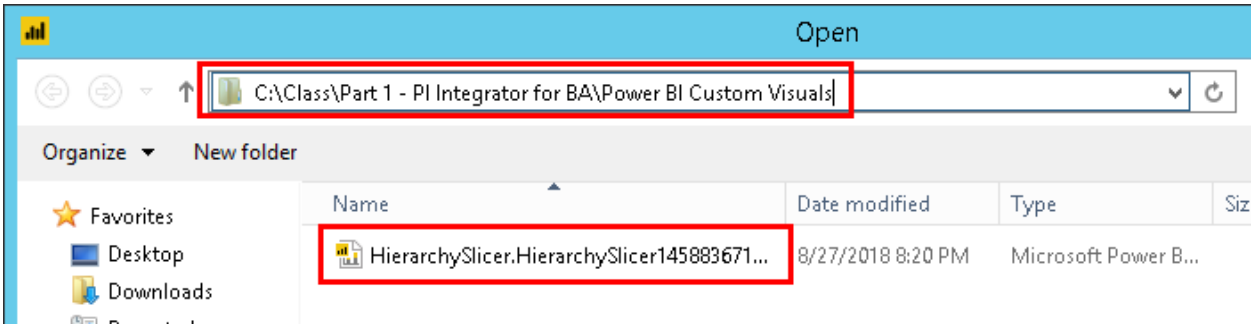


Importing and Configuring the Hierarchy Slicer

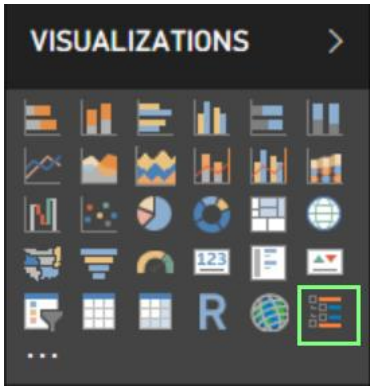
Now it's time to import the custom visual. Open Power BI, click the ellipses within the Visualization Pane, and select Import from file:



Navigate to **C:\Class\Part 1 - PI Integrator for BA\Power BI Custom Visuals** and select the **HierarchySlicer** file.



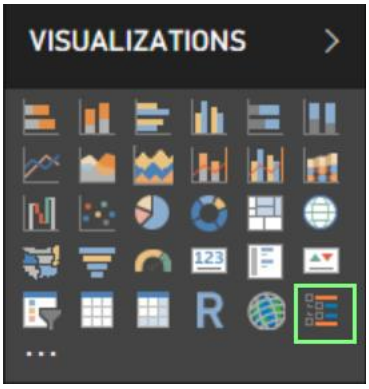
We should now see the Hierarchy Slicer in the list of available visuals:



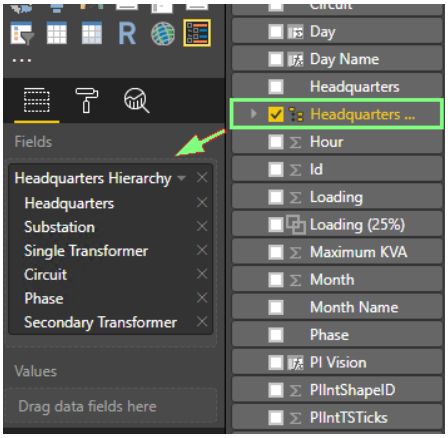
Mimic PI AF Hierarchy – Hierarchy Slicer

This exercise requires the Hierarchy Slicer custom visual be imported and assumes the Hierarchy has been configured.

We will use a Hierarchy Slicer to leverage the existing PI AF hierarchy for filtering. Add a Hierarchy Slicer:



Drag and drop the **Hierarchy** to the visual fields:



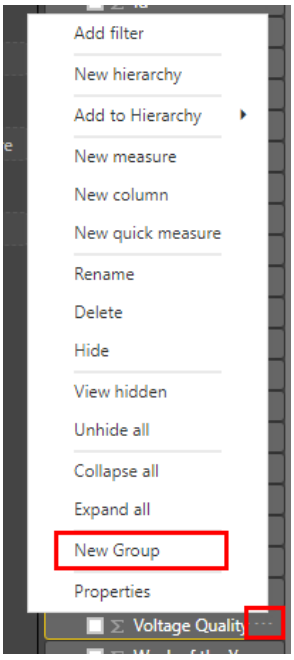
Experiment with the Hierarchy Slicer for a bit by drilling down through the levels. Note that checking a box for a parent will also include the children. This is a great way to visualize how filtering works in Power BI.

Optionally change the Title of the Hierarchy Slicer to Network in the formatting options and increase the text size.

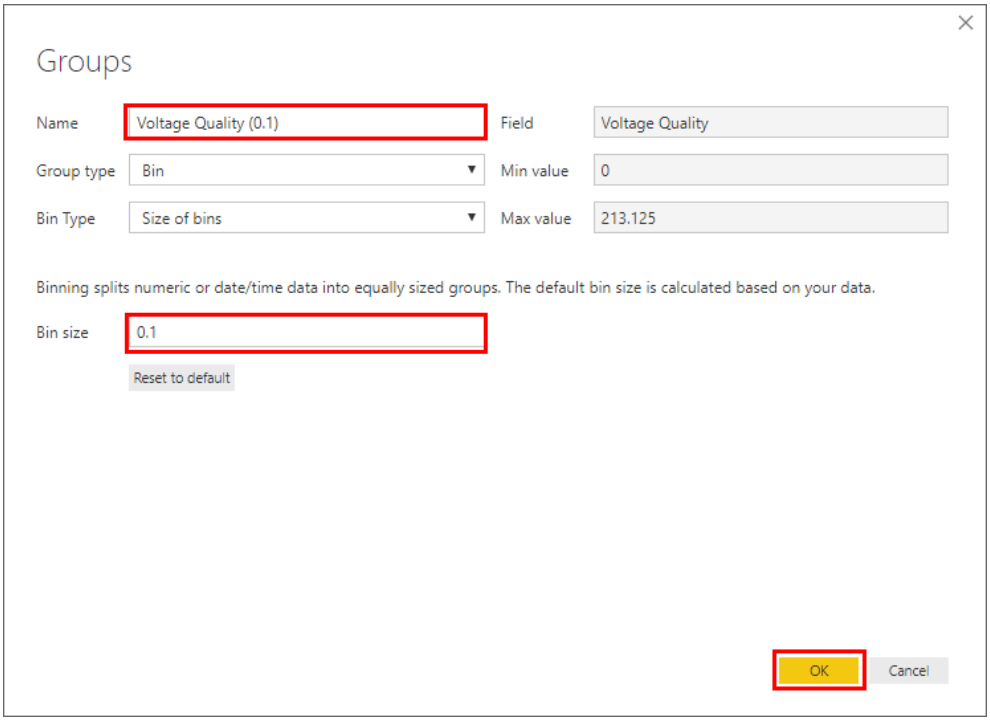
Voltage Quality Profiles – Clustered Column Chart

In this part we'll use a trick to represent a statistical distribution using a combination of a Group and a Clustered Column Chart.

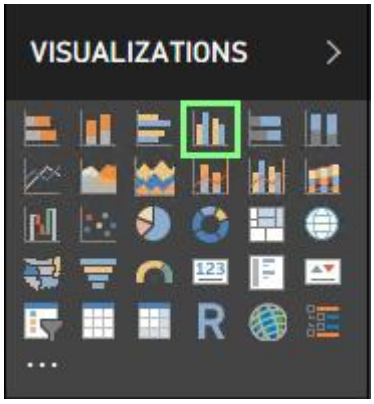
First configure the Group for voltage qualities, using .1 as the bin size. These bins will form the x-axis of the chart. Click the ellipses next to **Voltage Quality** and select **New Group**:



Change the name to **Voltage Quality (0.1)** and the bin size to **0.1** the click **OK**:



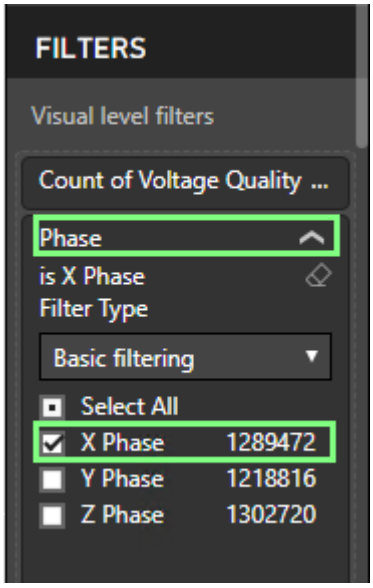
Now add a Clustered Column Chart:



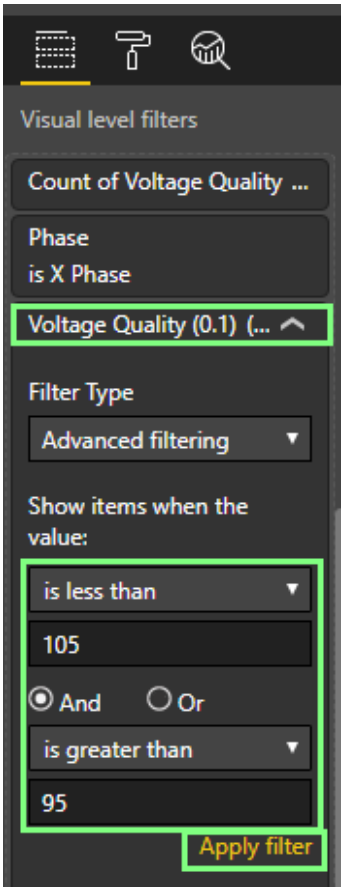
Use the **Voltage Quality (0.1)** Group as the Axis, **Phase** as the Legend, and **Count of Voltage Quality** (summarize Voltage Quality by **Count**) as the Value.

The chart is of course not usable in its current form. We will need to apply filtering to show only the X Phase and remove the outliers from the chart.

Under Filters in the Visualization Pane, filter to only include the **X Phase**:



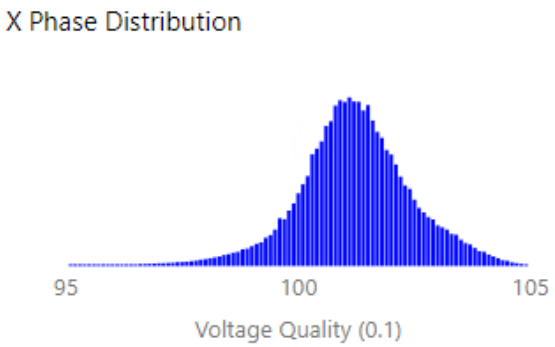
Filter the Voltage Quality (0.1) bins to only include the range of **95 to 105**. Be sure to click **Apply Filter**:



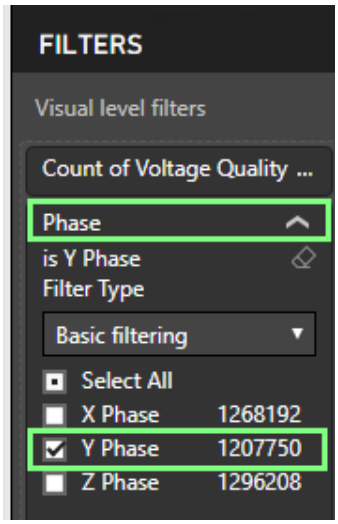
Now apply some **formatting**:

- Change the visual title to “X Phase Distribution”, make the text black, and increase the text size
- Turn off the legend
- Turn off the Y-axis
- Turn on the X-axis title and increase the text size (expand X-Axis in the formatting options)
- Change the Data colors to blue

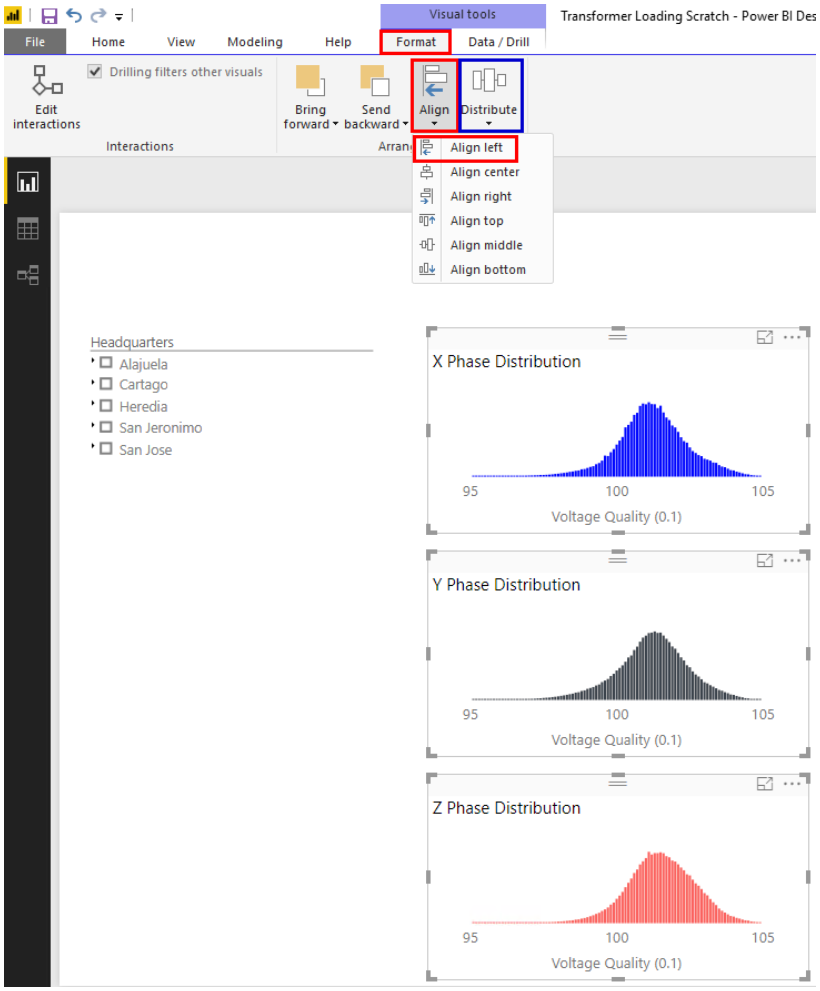
The resulting chart should now look like this:



Now creating the distributions for the Y Phase and Z Phase is easy. Simply **copy and paste** the X Phase Distribution Line Chart and **change the Filters and Titles**:



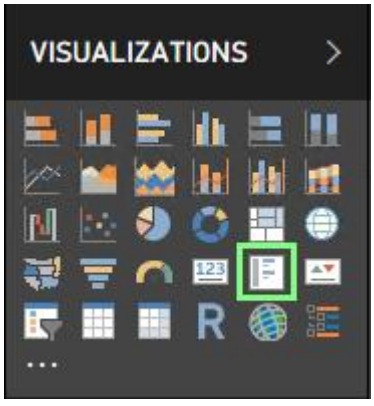
Now select them all and align left then distribute vertically:



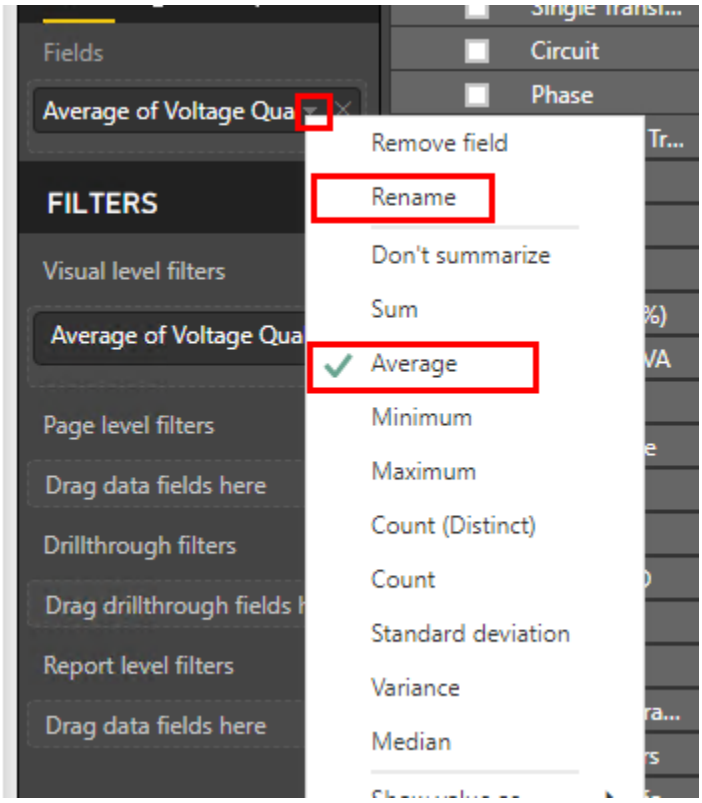
Voltage Quality Statistics – Multi-row Card

We will now configure multi-row cards showing some voltage quality statistics for the different phases.

Click some white space and add a Multi-row Card:

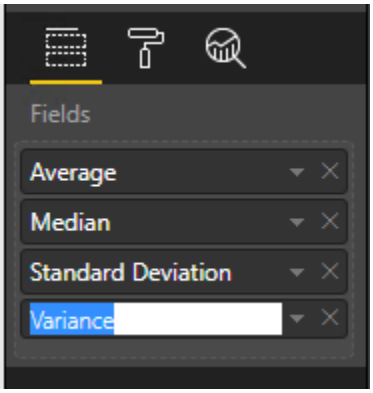


Add **Voltage Quality** as the first field, summarize as **Average** and rename the field to "Average":

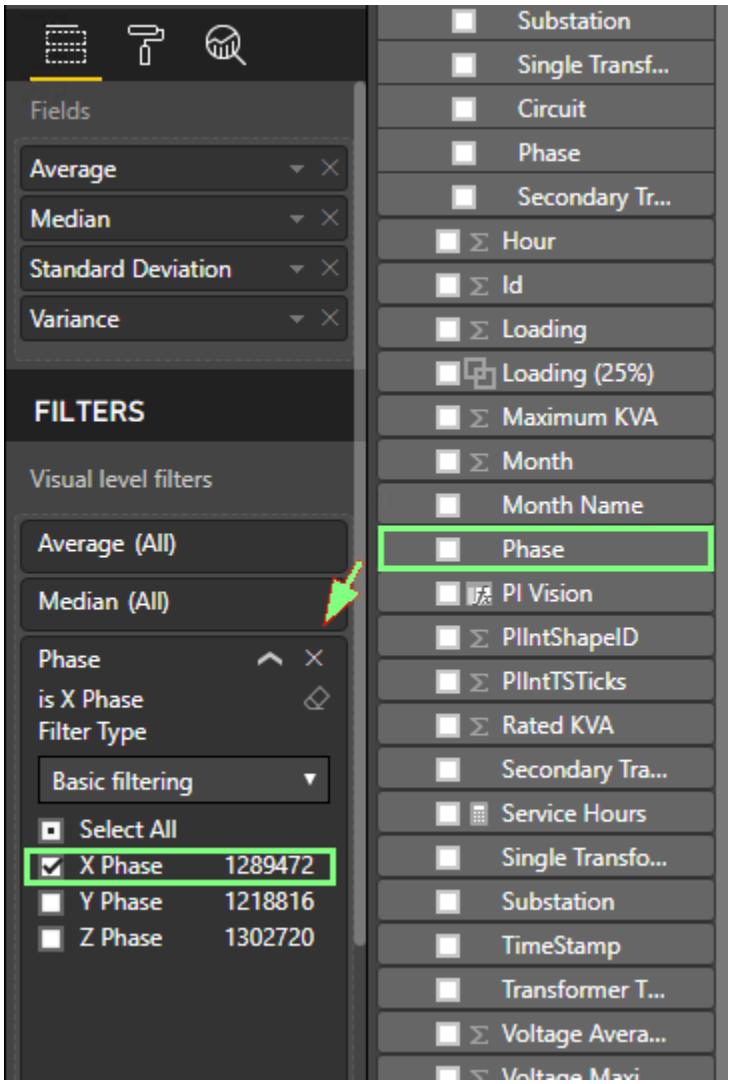


Add Voltage Quality again, this time summarize as **Median**, and rename the field to "Median".

Repeat for the **Standard Deviation and Variance**:



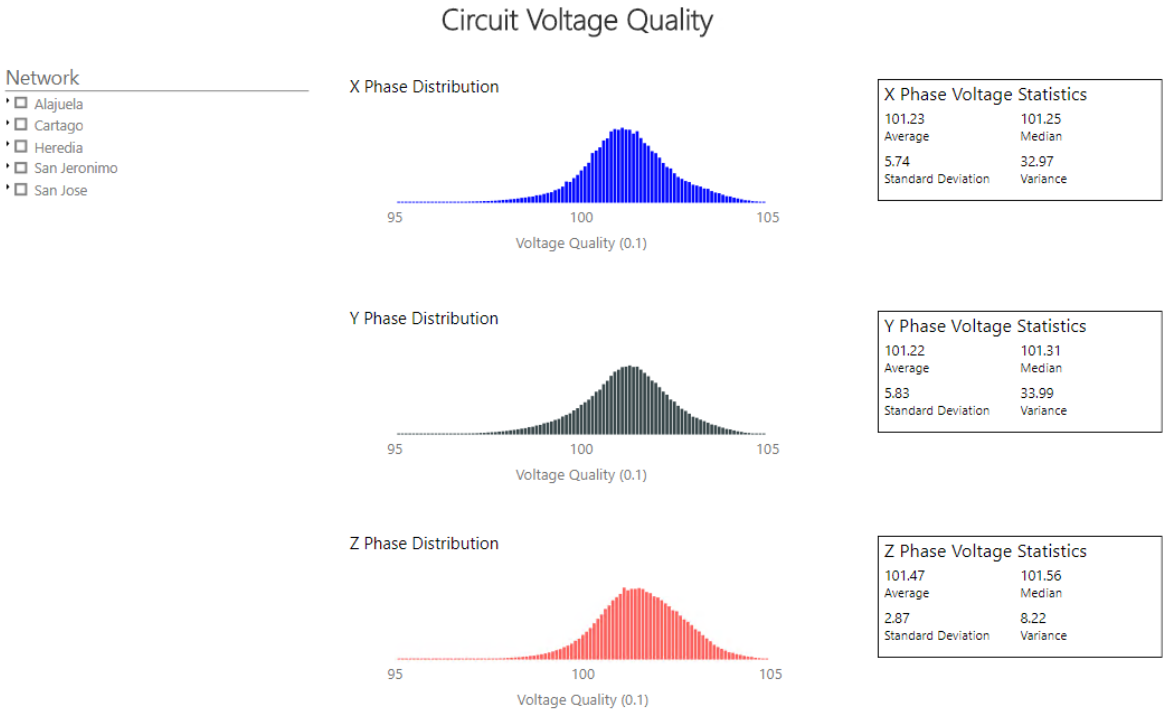
Filter the card to only include X Phase data, similar to the distribution charts. Drag and drop the Phase to the filters list and select X Phase:



Apply some **formatting** changes to the Multi-row Card:

- Add the title “X Phase Voltage Statistics” to the card using the formatting options. Change the text color to black and increase the text size.
- Turn on the border
- Turn off Show bar under Card
- Change the category labels to black text

Once you’re happy with the formatting, copy and paste the X Phase Voltage Statistics card and change the filtering to create cards for the Y Phase and Z Phase, then align them with the distribution charts. The resulting report should look something like this:



Finally experiment with filtering the report using the hierarchy.

5.2.4 Directed Activity – Substations on a Map

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

- Configure a **Map** visual

We want to display the different substations from the Network Load Profile on a map to lend geographical context to the report. This will help assess which substations may be impacted by extreme weather patterns and the relative importance of various substations in terms of delivered power and geographical region.

Approach:

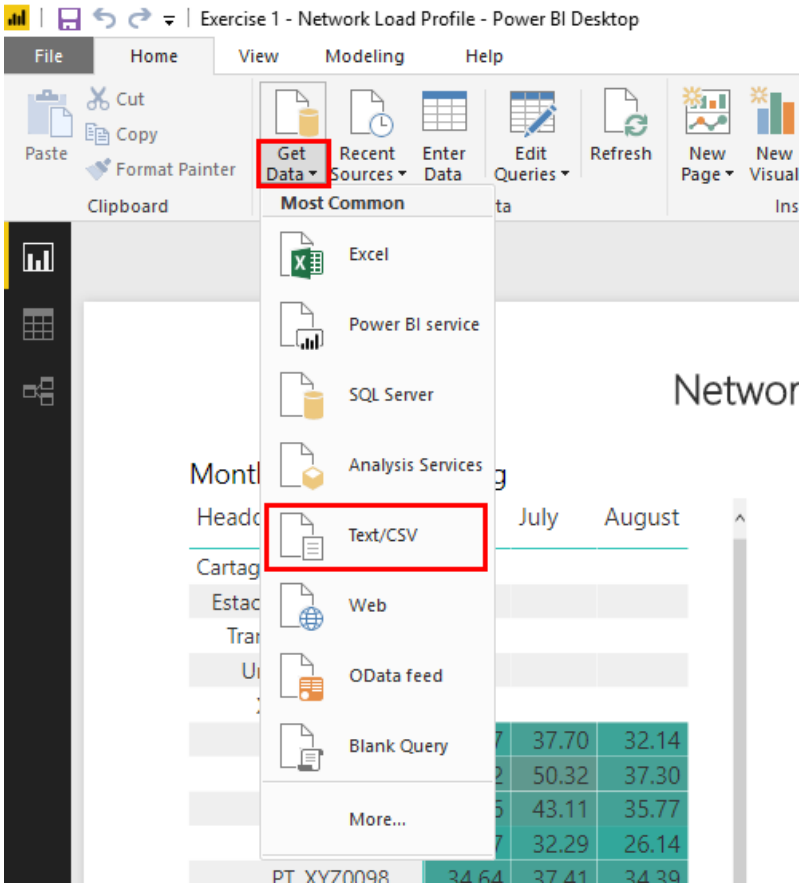
Substations on a Map – Map

We will modify the Report from Exercise 5.2.1 to include a Map of the substations instead of the transformer matrix.

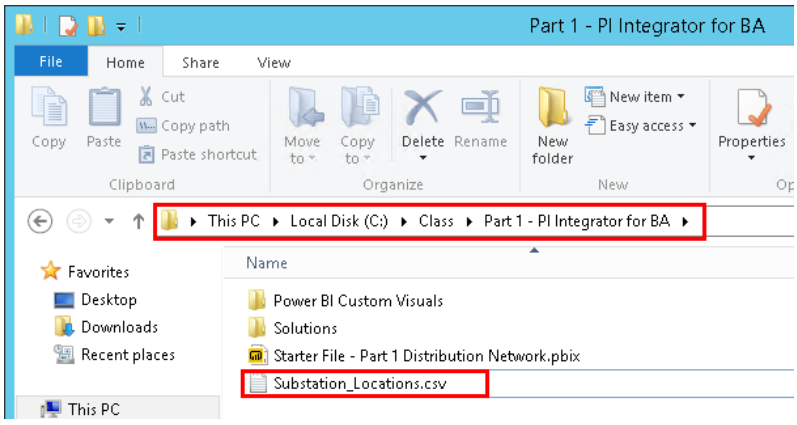
You may want to start with a copy of the solution file **C:\Class\Part 1 - PI Integrator for BA\Solutions\Network Load Profile.pbix** or copy the 5.2.1 report to a new sheet. This is just to back up your work since we will be deleting the Matrix to free up screen real estate.

You may have noticed that the data set we've been working with doesn't really include geocoding information, which is the information required to place each substation on a map. We could potentially use the Substation names and hope for the best, but there will be some ambiguity because several cities in Latin America and even Europe share these names. To keep things explicit we will import the Latitudes and Longitudes of all the substations.

Use **Get Data** and import data from a **Text/CSV** file:

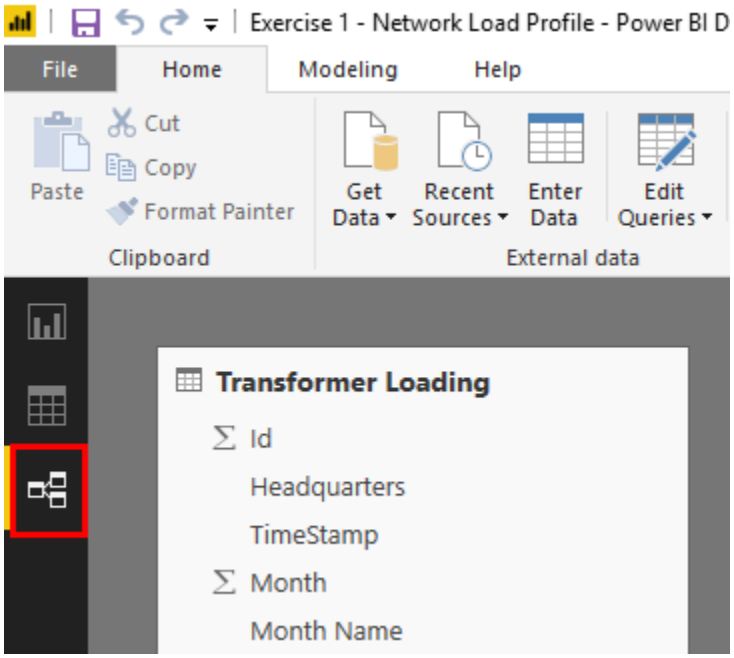


Select the **C:\Class\ Part 1 - PI Integrator for BA\Substation_Locations.csv** file:



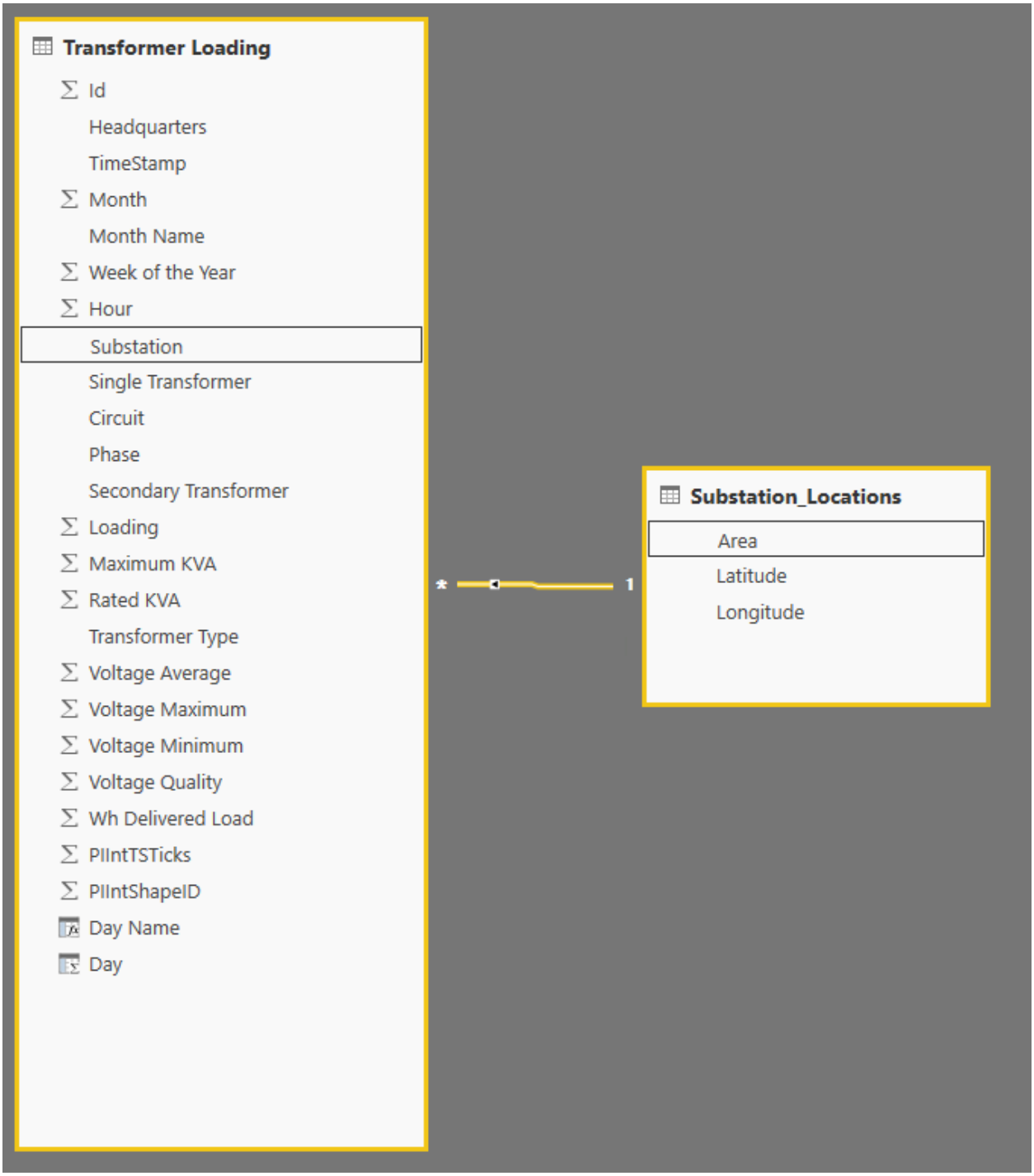
Power BI will automatically detect the headers, leave everything as-is and **click Load**.

Next, we need to define a relationship between the Transformer Loading data set and the new Substation_Locations data set. Head to the **Relationships View**:

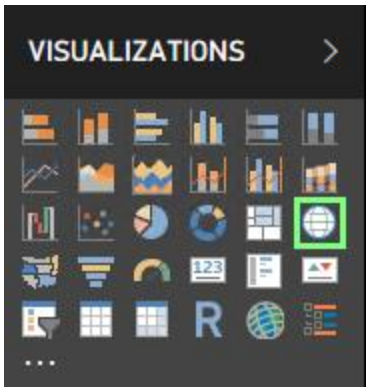


In the relationships view you can configure relationships between different tables. In this case we want to link the two tables using the Substation column from Transformer Loading to lookup the corresponding Latitude and Longitude from Substation_Locations. If you're familiar with SQL queries, configuring relationships between tables in Power BI is similar to choosing an INNER JOIN column. More information about relationships can be found in the Microsoft Power BI documentation.

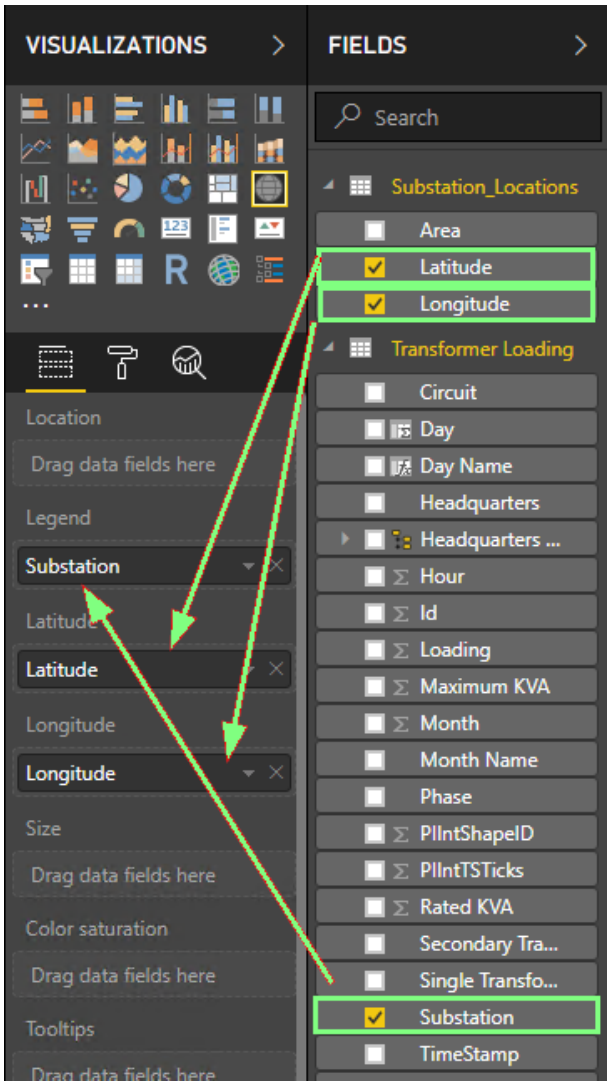
Expand and reposition the tables so that all fields are visible, then **drag and drop** Substation from the Transformer Loading table to the Area field on the Substation_Locations table, then click the relationship line between them to highlight the related columns:



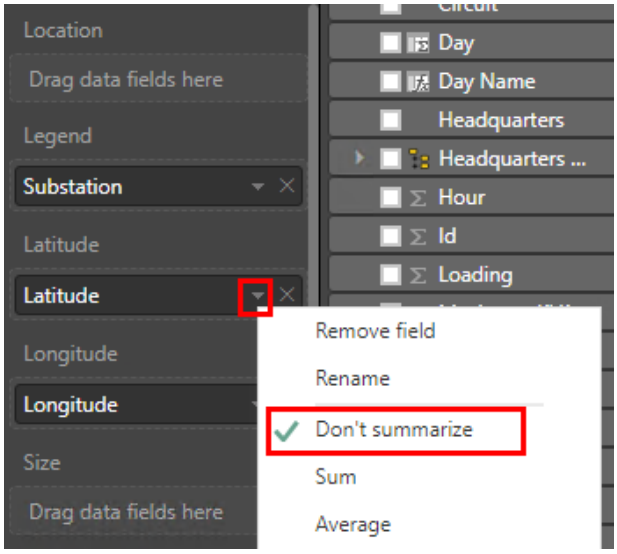
Now that we have the Latitudes and Longitudes, **Delete the Monthly Average Loading Matrix and resize the other visuals to make way for a Map**, then create a Map:



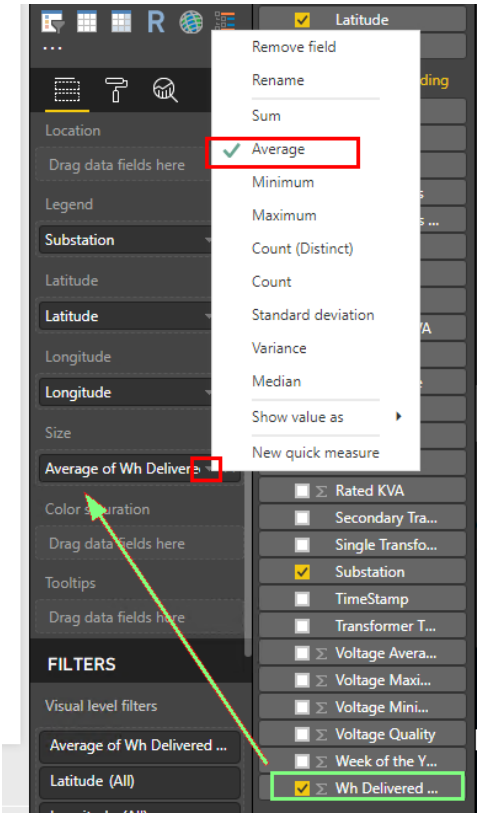
Use **Substation from the Transformer Loading table** as the Legend, and **Latitude and Longitude from the Substation_Locations table**:



Configure Latitude and Longitude as **Don't summarize** as per the warning on the Map visual:



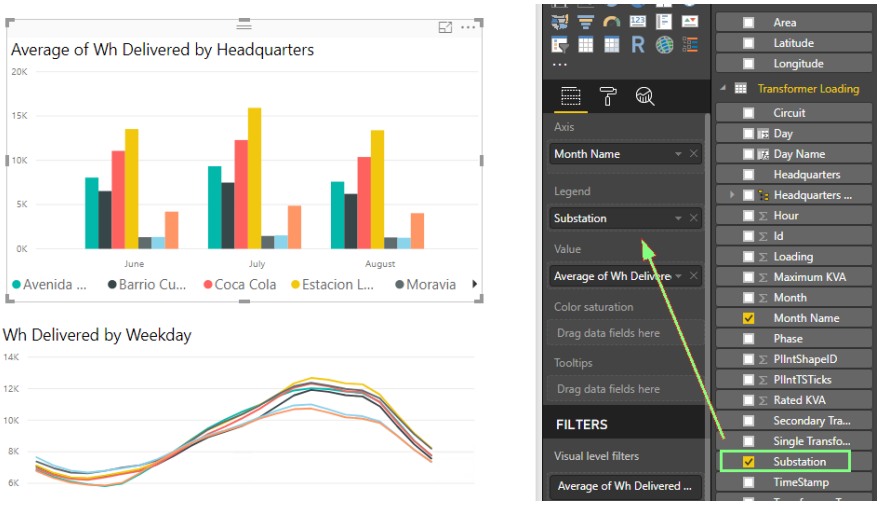
Resize the Map to fill the left side of the report, and add Average of Wh Delivered as the Size field:



Apply some **formatting**:

- Change the Title of the Map to “Average of Wh Delivered by Substation”, change the text to black, and increase the text size
- Move the Legend to the bottom

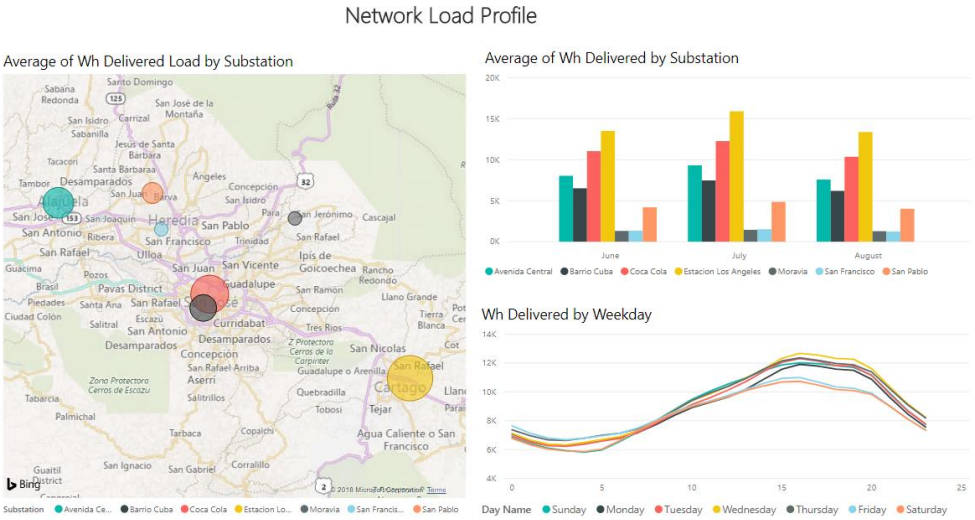
Replace the Legend field in the Clustered Column Chart with Substation:



Apply some more **formatting**:

- Change the Title of “Average of Wh Delivered by Headquarters” to “Average of Wh Delivered by Substation”
- Reposition and resize the visuals so that everything looks clean
- Change the text size of the Legends so that all Substations and Weekdays are visible

The end result should look something like this:



Notice that you can now use the Map to filter for different substations.

6 Part 2 – Power BI Reports using PI OLEDB Enterprise

An alternative to PI Integrator for BA, which is available with the PI System Access license, is PI OLEDB Enterprise. PI OLEDB Enterprise has better future data support and is generally more flexible than PI Integrator for BA. The main drawbacks are the difficulty of writing SQL queries and reduced throughput. For example, PI OLEDB Enterprise will have a hard time reliably importing the 3.8 million rows during a report refresh as was necessary in Part 1.

In Part 2, we will explore the process of preparing the Asset Framework model to add additional dimensions of information to our AF database. The next step is extracting desired information (process data, metadata, and event frame data) from the PI System through PI OLEDB Enterprise. This data will be incorporated into a BI cube and used to develop interactive reports that allow us to “slice and dice” our data and bring meaning to our multidimensional data cube.

6.1 Directed Activity – PI AF Hierarchy and Data Set

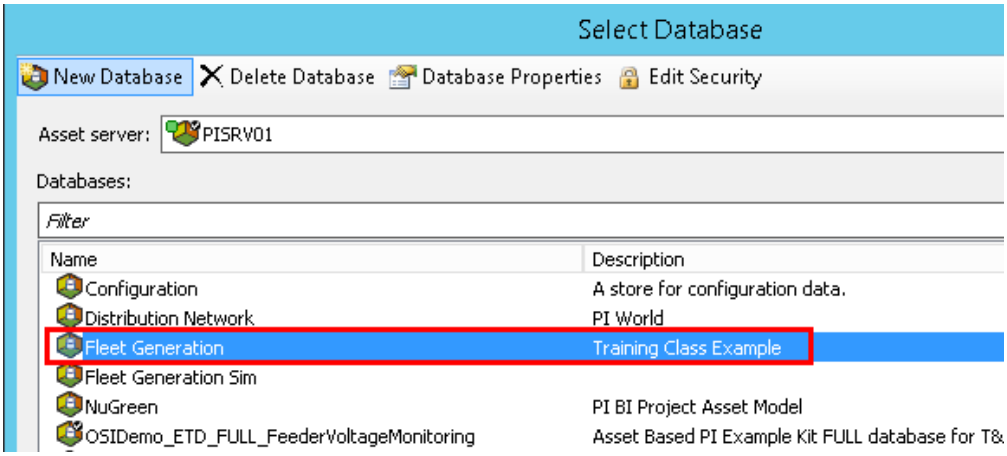
In this part of the class you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

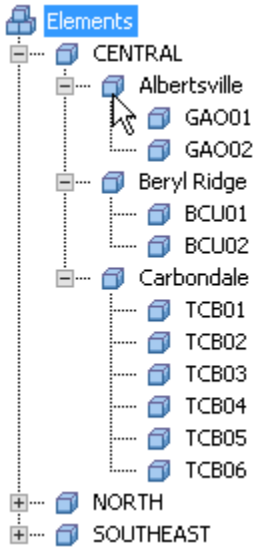
- Better understand the data set used in the following chapters

Approach:

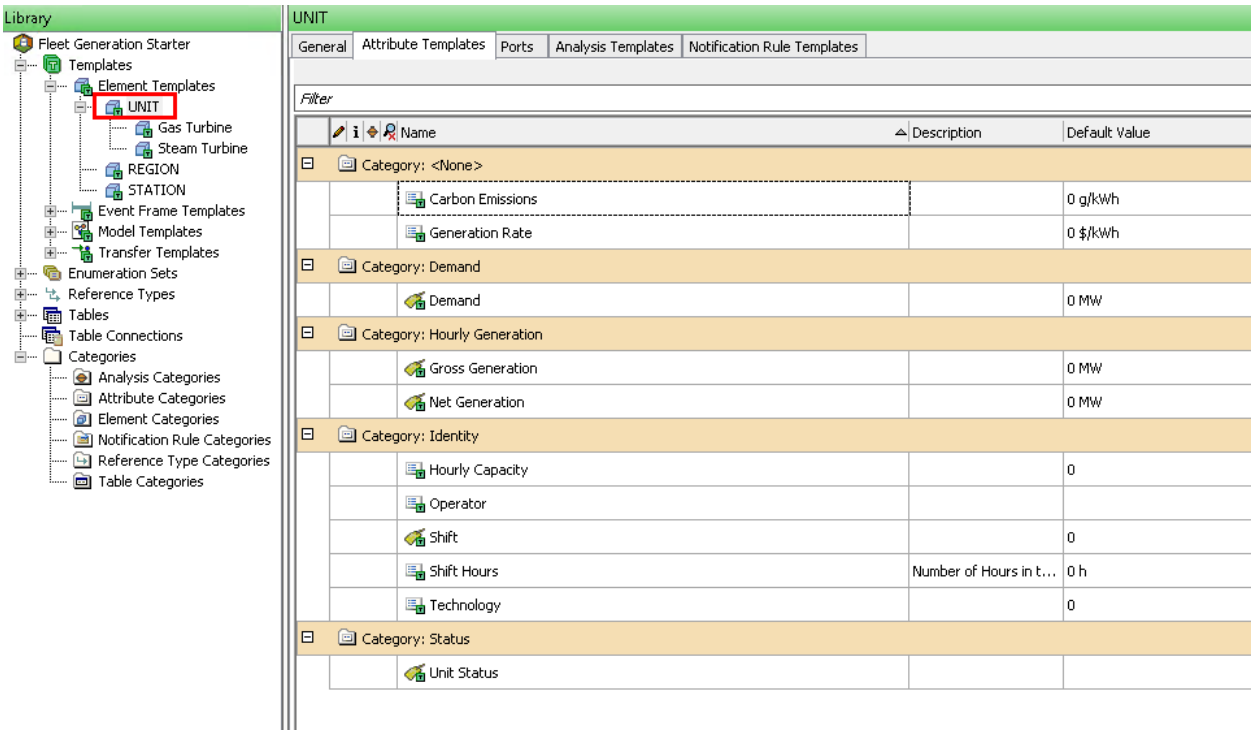
We will take a few minutes to review the Fleet Generation PI AF Database. We wish to analyze a number of KPIs for several generating units. Open **PI System Explorer** and navigate to the **Fleet Generation database**.



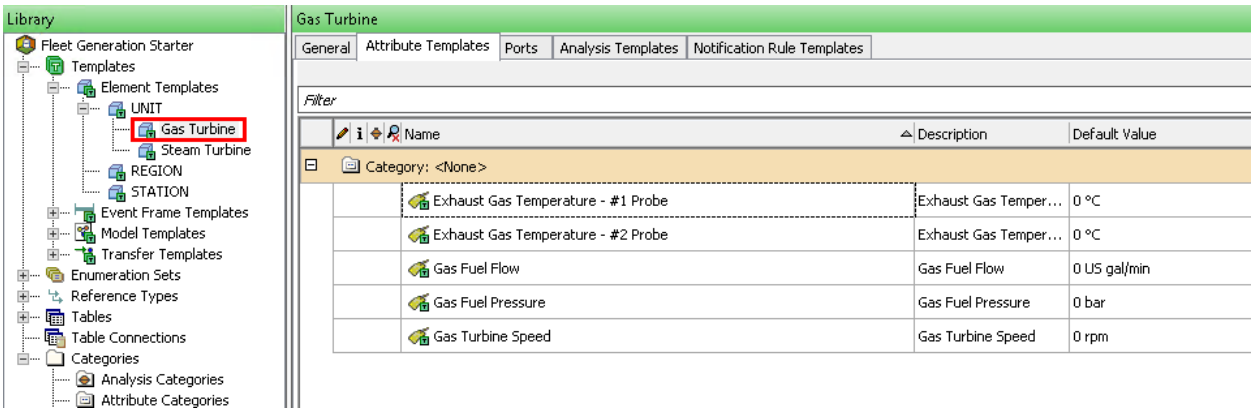
Browse the hierarchy, which is organized into Region, Station, and Unit.



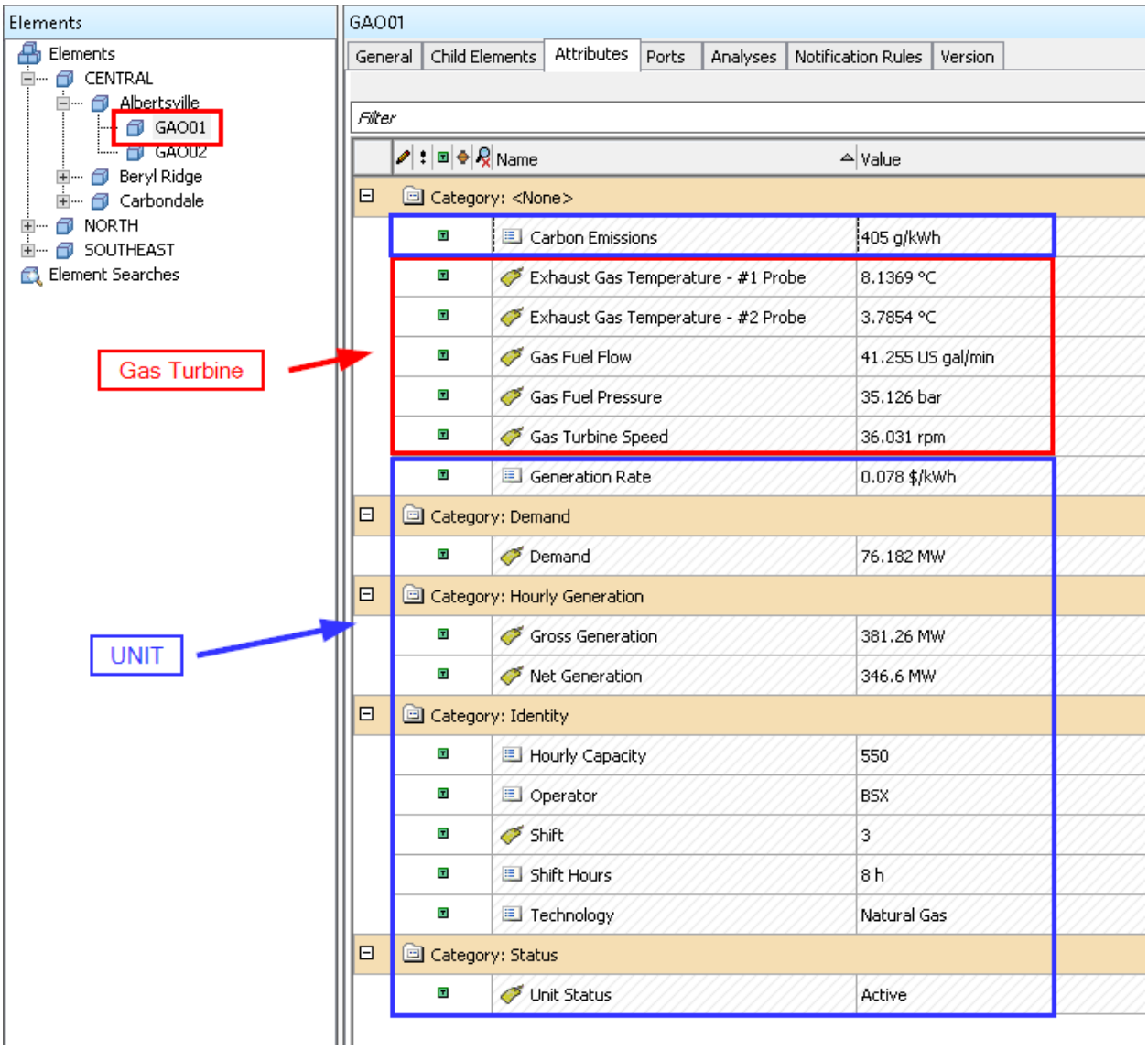
Most of the child elements are based on the generic **Unit template**.



Those in the CENTRAL region are based on the **Gas Turbine template**, which is derived from the UNIT template and has additional attributes.



Gas Turbines have all the attributes from the Gas Turbine template, but also inherit those from the UNIT Template:



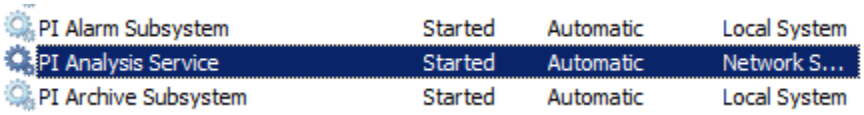
In the following chapters we will augment the AF templates with additional attributes and KPIs.

7 PI Analysis Service

PI Asset Framework is a powerful tool to help model the infrastructure of a company, region, or division. Through PI Asset Framework Formula Data References, you can create simple, on-the-fly calculations. PI Asset Framework also comes packaged with the PI Analysis Service, for more advanced analyses. The analytic capabilities include three analyses types, Expressions, Rollups, and Event Frame Generation, which allow for calculations to be applied at the template level as well as the ability to persist the results back to the PI Data Archive.

7.1 Capabilities of the PI Analysis Service

The PI Analysis Service, runs as a service that monitors all analyses and attributes associated with these analyses.



Expressions:

Expressions allow for multi-lined calculations that utilize mathematical operators and functions, if-conditions, and PI time-based functions to perform advanced analyses. Expressions, created for a given asset type (element template), are automatically applied to all elements of that type.

Rollups:

Rollups allow for the calculation of summary statistics (averages, maximums, minimums) of values from a set of AF attributes. Current statistical values can be written directly to the PI Data Archive.

Event Frame Generation:

PI Analysis Service allows for the automatic detection of events that occur. These events are bookmarked and information for any event type can be retrieved for further analysis.

Scheduling:

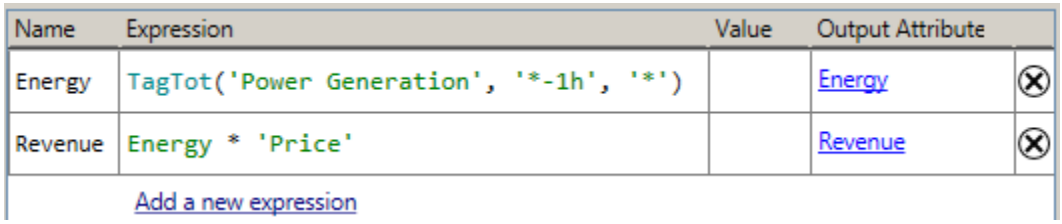
Expressions and Rollups can be scheduled to run whenever a new event arrives into the PI Data Archive or calculated on a periodic basis.

Backfilling:

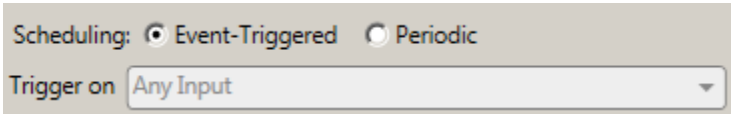
Results from all three types of analyses can be backfilled into the PI System.

7.2 Expressions

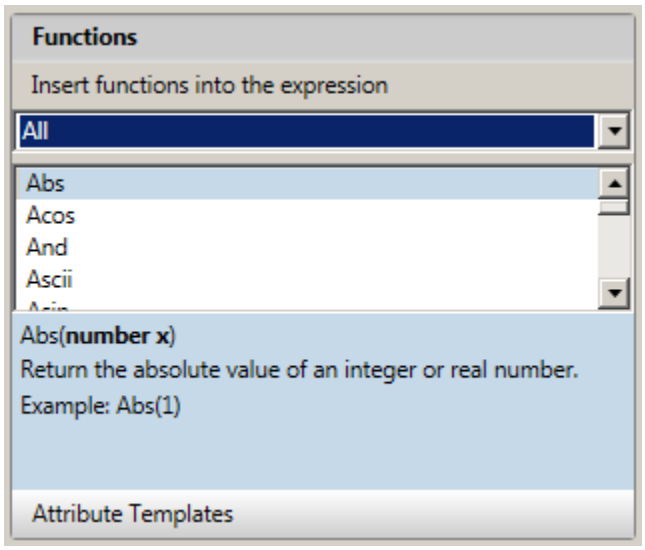
With Expressions, you can implement calculations through a set of built-in functions that take values of attributes in PI Asset Framework as inputs, and outputs results to other PI AF attributes. Expressions can be scheduled to run periodically or scheduled to run whenever the input parameters of the expressions receive a new value.



Multi-line calculation dependency allows for each expression to be written to different output attributes as well as re-using calculated results in subsequent expressions.



Each set of expressions allows for periodic or event-triggered scheduling.



Function Category	Example
Archive Value Statistics	TagAvg, PctGood
Date and Time	Bod, Hour
Logical	And, If
Math	Abs, Sqr
Operators	>, <>, *
PI Data Archive Digital States	DigState, DigText
Point Attributes	TagSpan, TagType
Search and Retrieval	TimeEq, NextEvent
Statistical	Rand, Total
Status	NoOutput, TagBad
String	Len, Text

A set of built-in performance equation-like syntax allows for access to a range of functions. The available options include mathematical and logical operators and functions, date and time functions, PI-specific performance equation functions, and string manipulation functions.

It is recommended to configure analyses at the template level.

The following procedure can be used to configure an Expression analysis using a template:

1) In the AF Database Library, create a new analysis template of type Expression.
2) Define expressions for the calculations in the analysis template.
3) Define the scheduling for the analysis template.
4) Define output attribute templates to store results.
5) Create the PI tags used to store the results.
6) Evaluate and preview the data to validate calculations.
7) Backfill the calculation if required.
8) Confirm the backfilled data
9) Backfill the data for other elements sharing the same template.

7.2.1 Directed Activity – Calculate Utilization for Assets

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

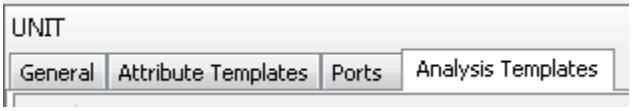
The Utilization is a percentage that represents the amount of electrical power that a unit produced against its theoretical capacity. Configure, test, run, and validate analyses to calculate the percent utilization of all generating units.

Approach:

- In PI System Explorer, navigate to the Library in the Fleet Generation database.
- Under Element Templates, select the UNIT element template.
- Select the Analysis Templates tab to configure the multi-lined expression for Utilization:
  Utilization = Total Hourly Gross Generation / Hourly Capacity
- Specify and configure an attribute template to store the results.
- Schedule the calculation to run periodically every hour.
- Backfill unit GAO01 for the past seven days.

Approach

From the **Unit Template**, found in the Library plug-in of the Fleet Generation database, select the **Analysis Templates tab**.



Configure a **new analysis**. Name the analysis Utilization and set the analysis type to Expression.

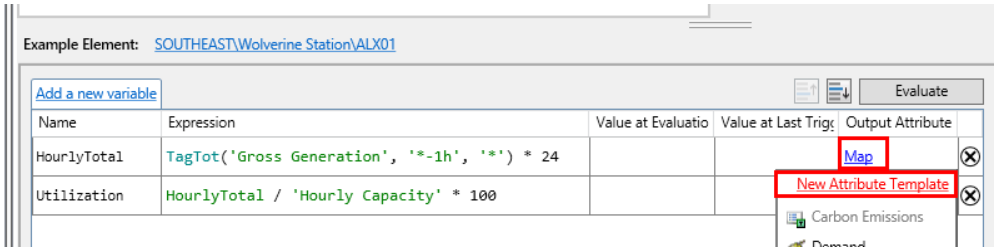
Configure the expressions for the hourly total of Gross Generation and Utilization.

HourlyTotal = TagTot('Gross Generation','*-1h','*') * 24
Utilization = HourlyTotal / 'Hourly Capacity' * 100

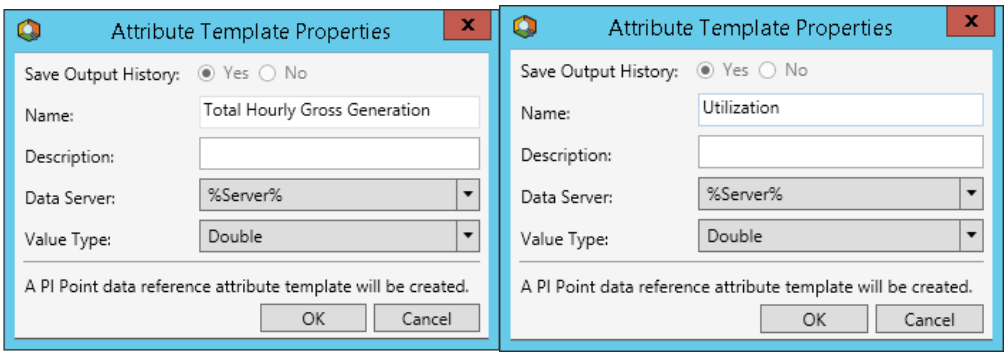
Name	Expression	Value
HourlyTotal	TagTot('Gross Generation','*-1h','*')*24	
Utilization	HourlyTotal / 'Hourly Capacity' * 100	

Note: The HourlyTotal must be multiplied by 24, as the Performance Equation function TagTot assumes the units of the input attributes are per day. Conversion factors should not be used elsewhere with PI Asset Framework, as UOM conversions occur automatically.

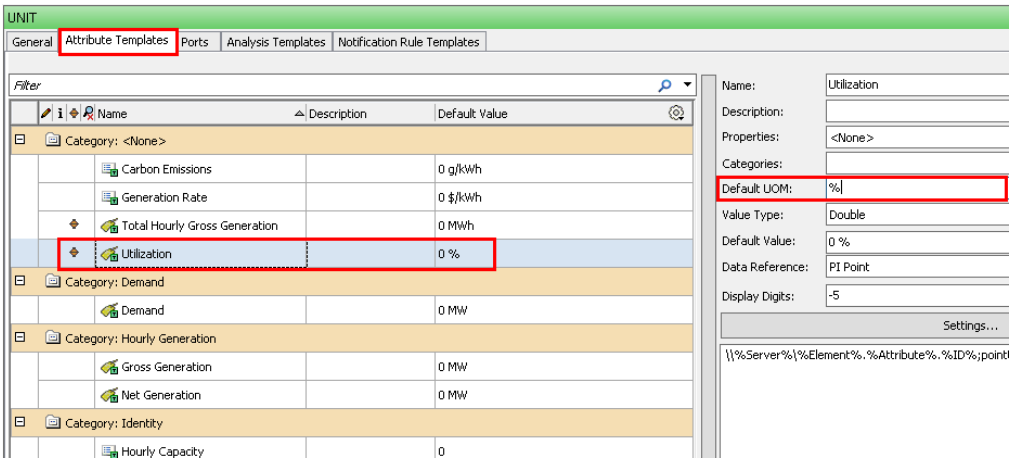
Define two new output attribute templates by clicking **Map** -> New Attribute Template.



Name them **Total Hourly Gross Generation** and **Utilization**, respectively.

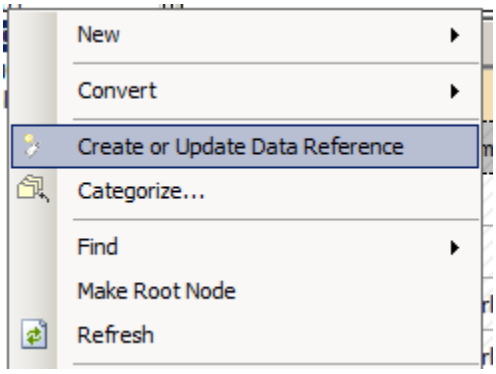


The UOMs can be set to **MWh** and **%** in the Attribute Templates tab:

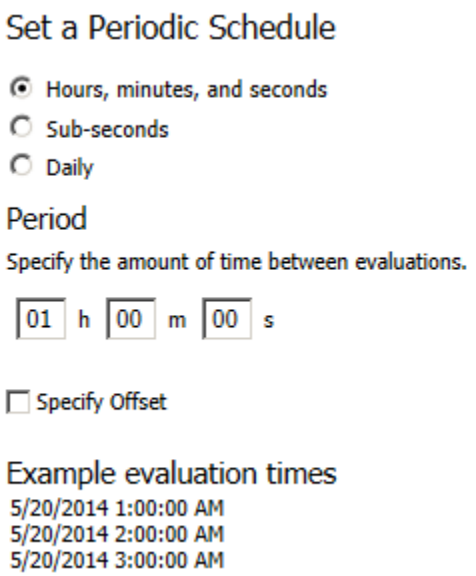


Create the PI Tags

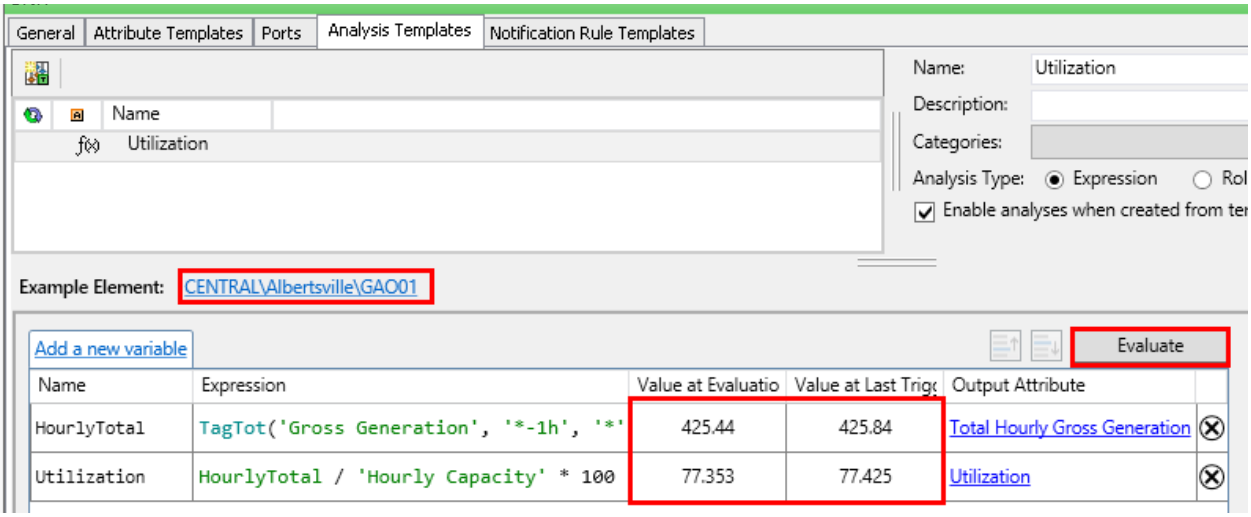
After the new attribute template has been configured, switch over to the Element Hierarchy. The attribute values for the new tags should be “Pt Created.” If not, right-click on the root Elements object. Select Create or Update Data Reference to automatically create the PI tags to store the calculated results.



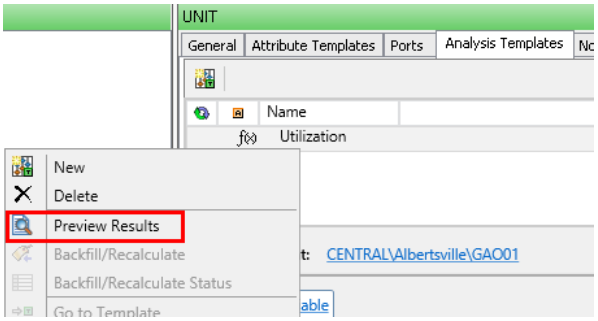
Switch back to the Unit Template Analysis Templates tab to **schedule** the Analysis Template to run periodically at the top of each hour.

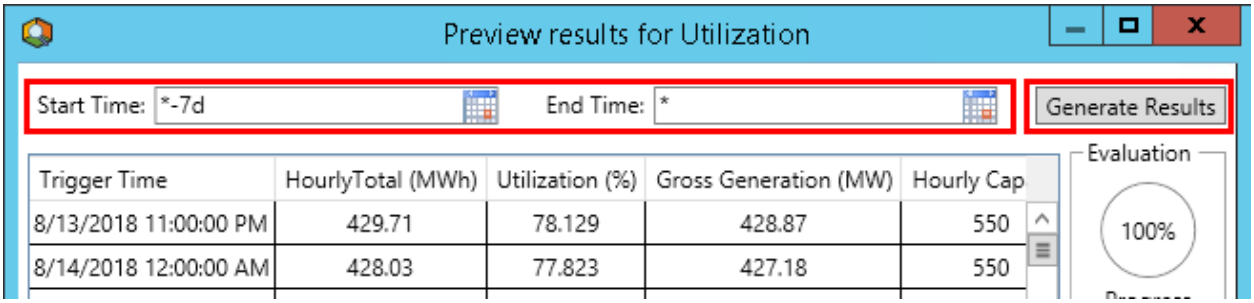


Set GAO01 as the Example Element and click on the Evaluate button to validate the expressions.



Prior to backfilling data into the PI Data Archive, it is usually a good idea to preview the results. Right-click **Utilization** and select **Preview Results**. Look at the results for the past 7 days:





7.2.2 Directed Activity – Backfill Utilization

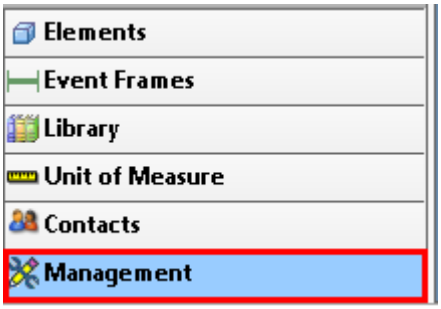
In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

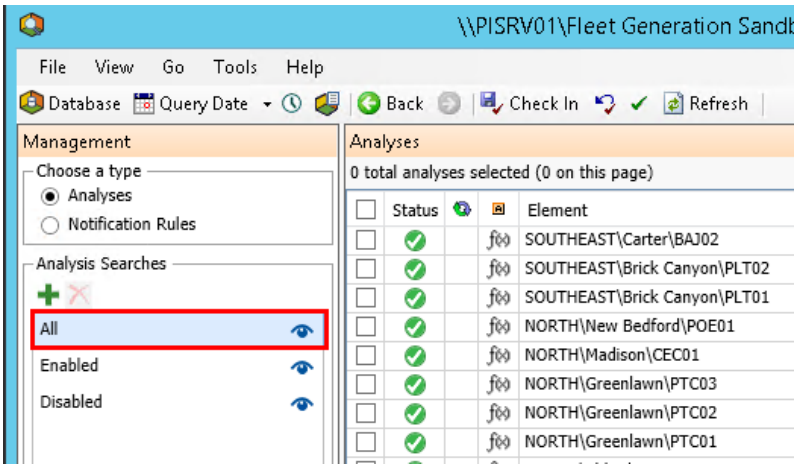
At this point, all the analyses for event frame generation have been set up for all the units of Fleet Generation. In order to calculate past Utilization values and generate history for analysis, the calculations must be backfilled.

Approach:

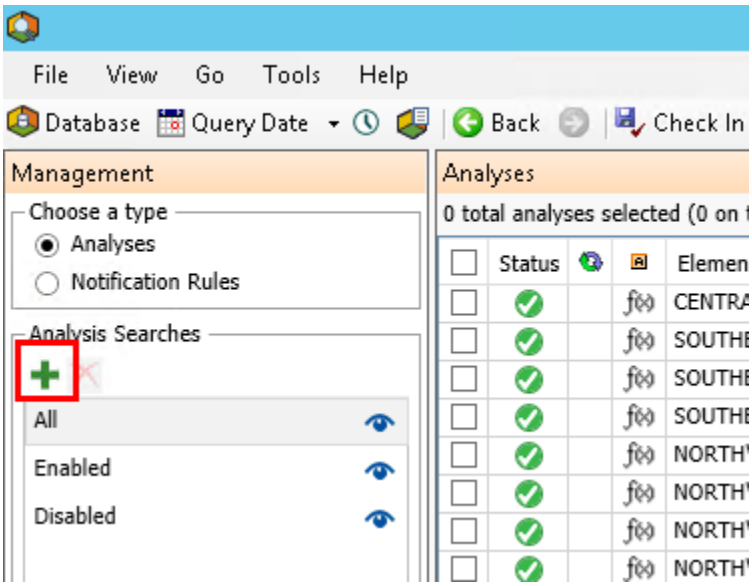
From PI System Explorer, select the Management plugin



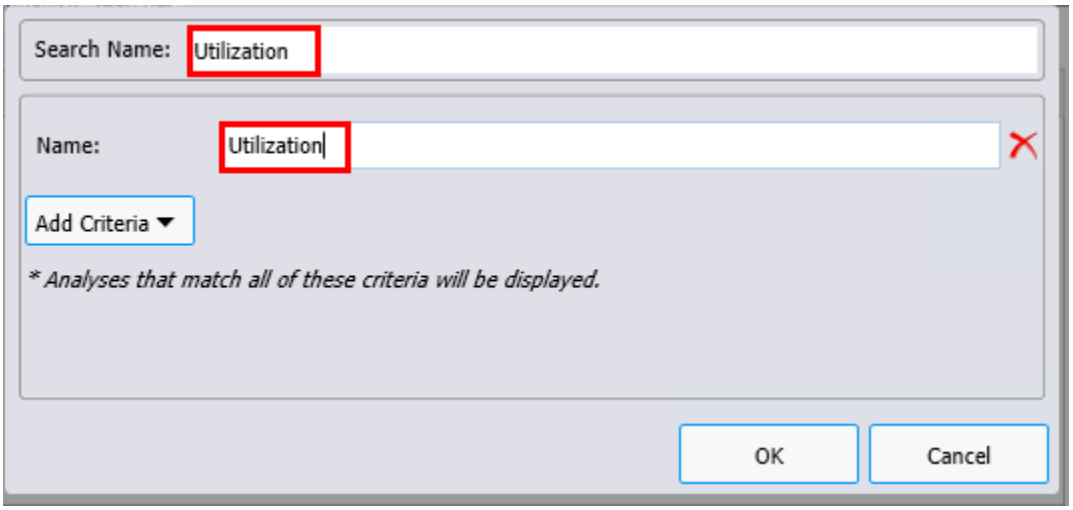
Right now, the only Analyses that exist are those we just created, so one can simply select **All** or **Enabled** to view the Utilization Analyses that we want to backfill.



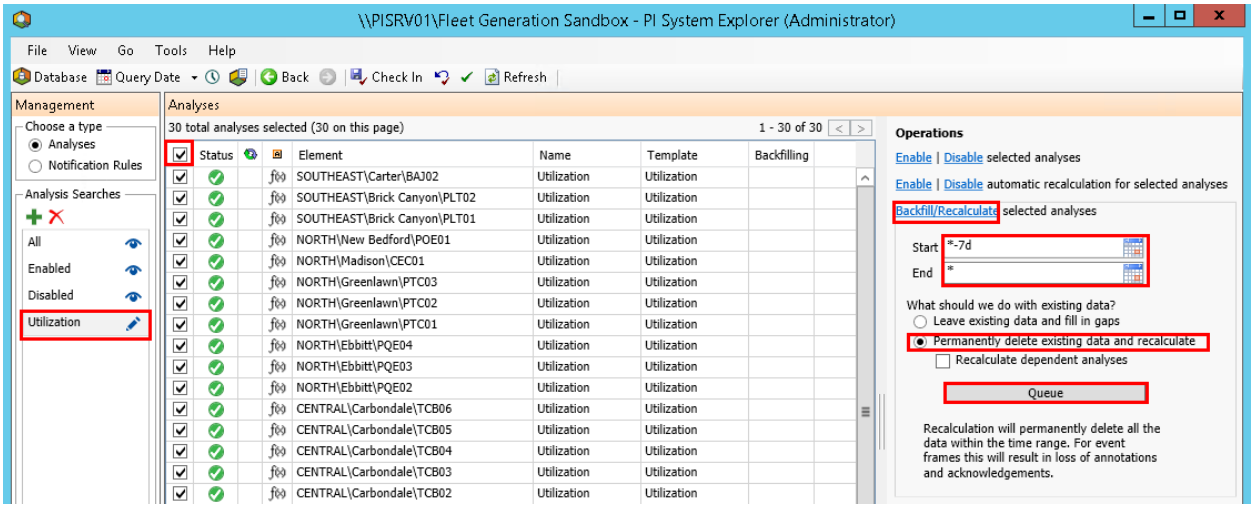
Normally there would be several types of calculations, so we'd want to filter them by setting up a search. Create a new search:



Name the search **Utilization**, then do **Add Criteria -> Name** and enter the name of the Analyses and click **OK**.



Click the checkbox to select **all** Utilization Analyses. Then Select the Backfill/Recalculate operation and set the start time to "***-7d**" and the end time to "*****", select "**Permanently delete existing data and recalculate**", then click **Queue**:



7.2.3 Exercise – Calculate Generating Efficiency

This solo or group activity is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the activity.

Objective:

Not all of the electricity produced by our generators will make it out to the grid. Some will be consumed by the internal circuity in the generator itself. The net generation is defined as the amount of gross generation, or the amount of electricity that a generator produces, less the electricity required to operate the unit. Calculate the generating efficiency, or the *ratio between the net generation to the gross generation*, expressed as a percentage.

Which unit is performing with the greatest efficiency?

Approach:

- In the PI System Explorer, navigate to the Library in the Fleet Generation database.
- Under Element Templates, select the UNIT element template.
- Select the Analysis Templates tab to configure the expression for generating efficiency, named **Generating Efficiency**.
- Specify and configure an attribute named Generating Efficiency to store the results with units of %.
- Schedule the calculation to run periodically every hour.
- Evaluate the calculation using example element GAO01 and preview the results.
- Backfill all Efficiency analyses for the past seven days.

7.3 Rollups

The second analysis capability of the PI Analysis Service Analytics is known as rollups. Rollups allow for the calculation of summary statistics for a set of attribute values.

The types of summary statistics that are allowed are:

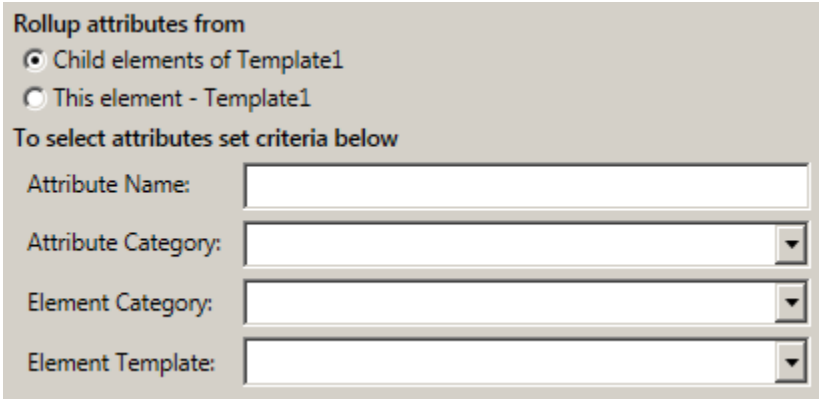
- Sum
- Average
- Minimum
- Maximum
- Count
- Median

Examples of rollup calculations include:

- Total mass of all contents in a tank farm
- Total production from all generating units for a particular site
- Maximum temperature of boilers within a building
- Average engine temperature of mining trucks
- Average temperatures for each asset with varying temperature sensors.

Selecting attributes to rollup

Attributes used in rollup calculations can come from 1) attributes from child elements relative to the element of interest or 2) the element of interest. One can set search criteria to specify the specific attributes to rollup. Depending on the source of the attributes (child elements or current element), the search criteria includes a masking pattern for the 1) Attribute Name, 2) Attribute Category, 3) Element Category, and 4) Element Template.



What is an element Example?

During the configuration of a rollup template analysis, when the source of the attributes to roll up are from the child elements, PI System Explorer is not aware of which parent element to retrieve child elements from. As such, when configuring a roll-up analysis template, you will need to specify an example element. Note that when configuring a roll-up at the element level, one will not need to select an example element as the child elements are from the specific, selected element.

Scheduling and backfilling

Similar to Expressions, the rollup analyses can be scheduled to run as new events come into the PI Data Archive or scheduled to run periodically. The PI Analysis Service also allow the results from Rollup calculations to be written back to the PI Data Archive.

The general process to properly configure and backfill an analysis template is:

1) Create a new analysis of type Rollup.
2) Define the source of the attributes to rollup (child element or current element).
3) Select the type(s) of summary statistics to calculate.
4) Define output attributes to store results.
5) Define the scheduling for the analysis.
6) Create the PI tags used to store the results.
7) Evaluate and preview the data to validate calculations.
8) Backfill the calculation.

7.3.1 Directed Activity – Calculate Average Utilization for Substations

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

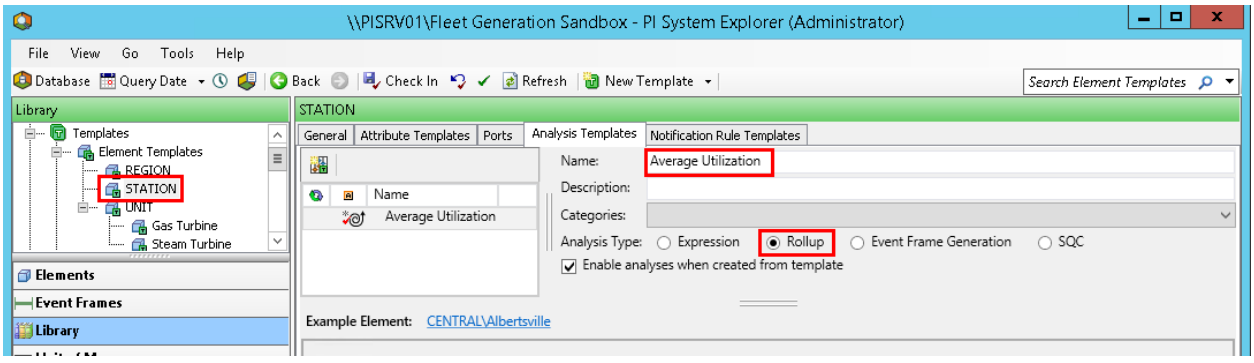
Objective: Management would like to have visibility over the average percent utilization of all generating units for each substation. Roll up the average utilization to the substation level.

Approach:

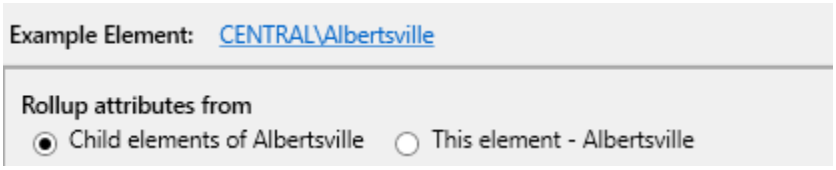
- Open up the Station Element Template from the Fleet Generation Database Library.
- Add a new analysis called Average Utilization with analysis type of Rollup.
- Select Central\Albertsville as the example element.
- Specify the criteria to select the attributes used for the rollup calculation.
- Select the summary statistic function for the average.
- Specify the output attributes (be sure to create the tags).
- Schedule the calculation to be event-triggered.
- Verify data.
- Backfill for the past 7 days.

Step-by-Step Approach

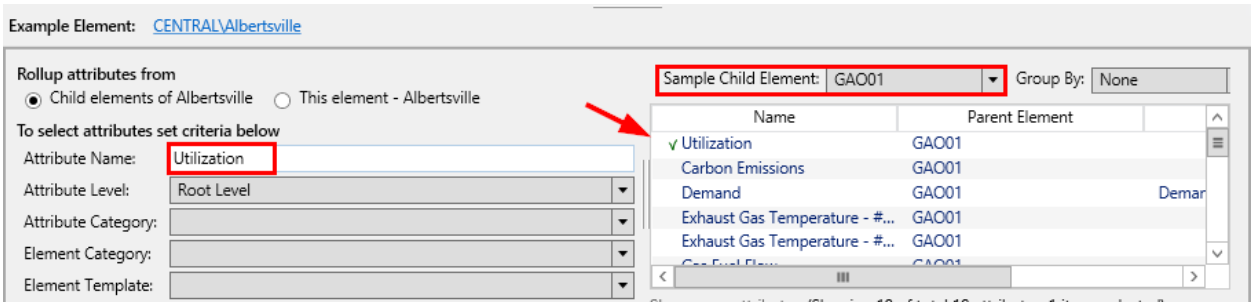
From PI System Explorer, go to the Library. Then select the Station Element Template. From the Analysis Templates tab, create a new Analysis called **Average Utilization** with Analysis Type **Rollup**.



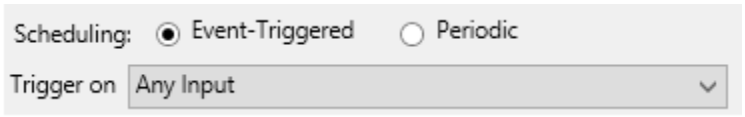
Specify the rollup attributes from child elements and set the example element to be **Central\Albertsville**.



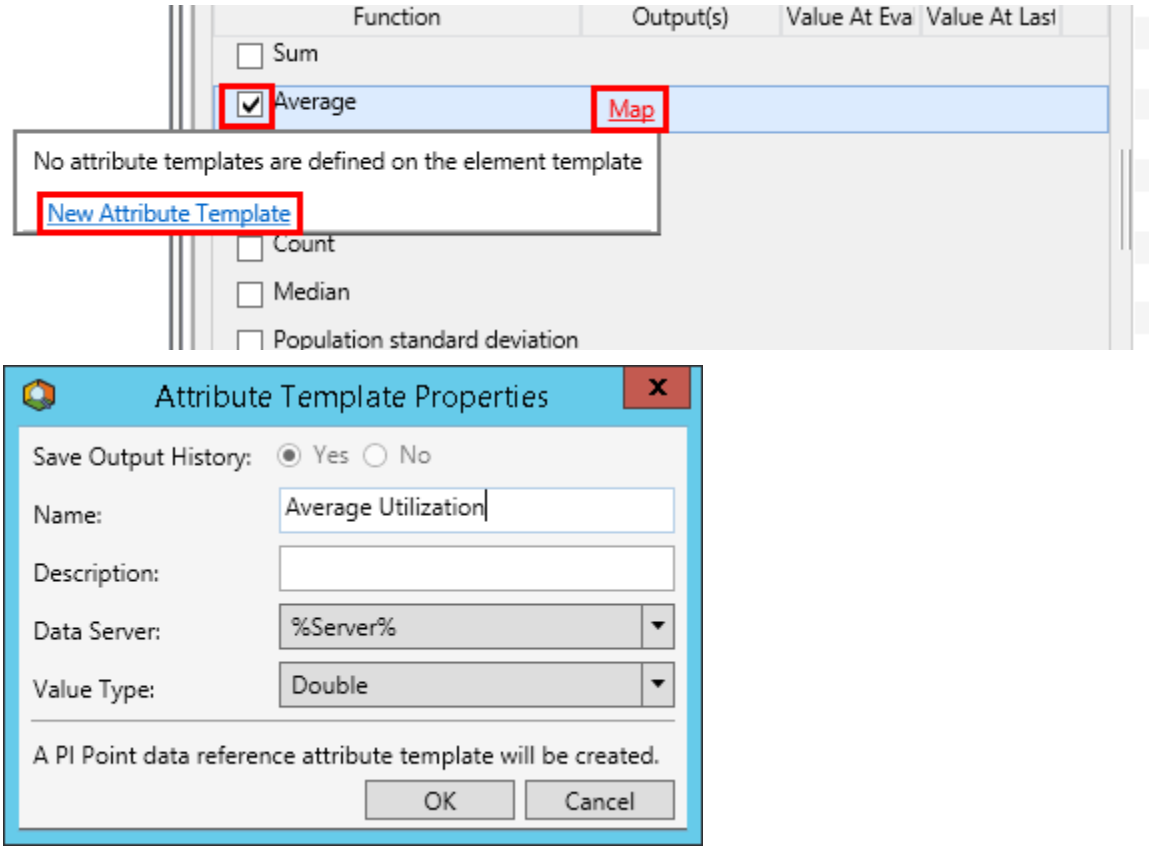
Set the attribute name field to **Utilization**. This mask will automatically select all Utilization attributes from the child elements of the Albertsville station. However in the preview only the Utilization from the Sample Child Element will be shown:



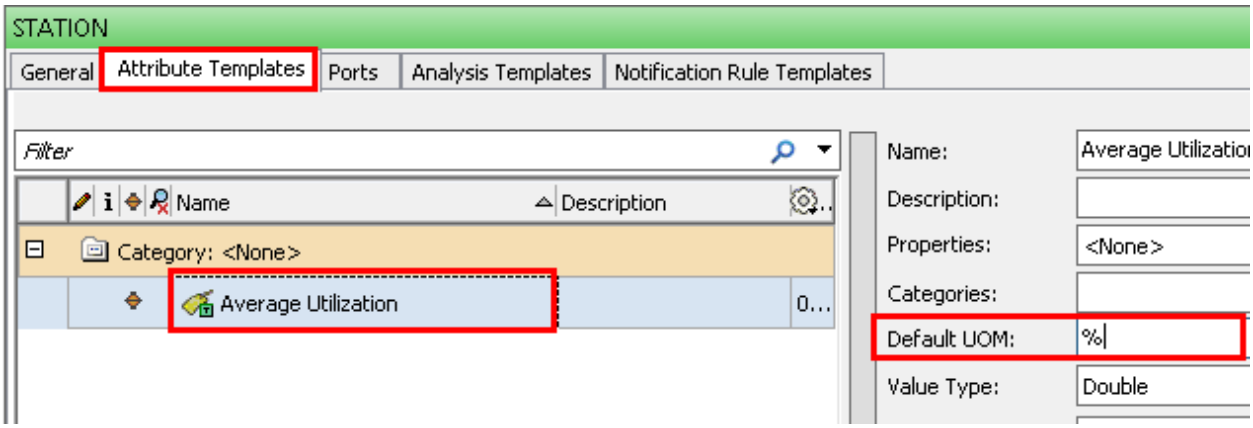
Set the scheduling to be event-triggered. Each time the Utilization analysis finishes calculating each hour, the rollup analysis will run.

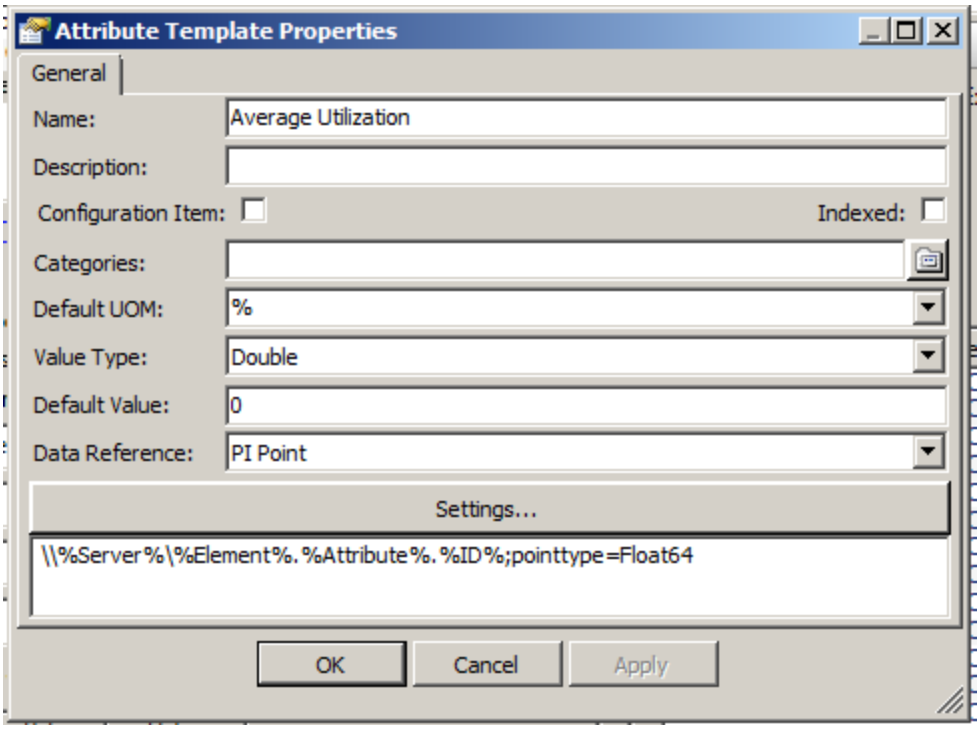


Select **Average** as the rollup function and create a new Output Attribute called **Average Utilization**.

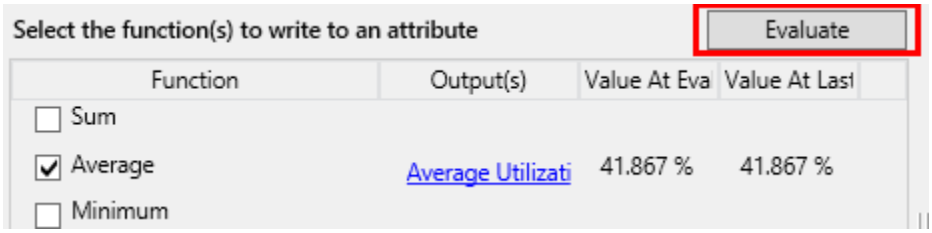


Set the default **UOM** of this new attribute to **%** in the Attribute Templates tab:





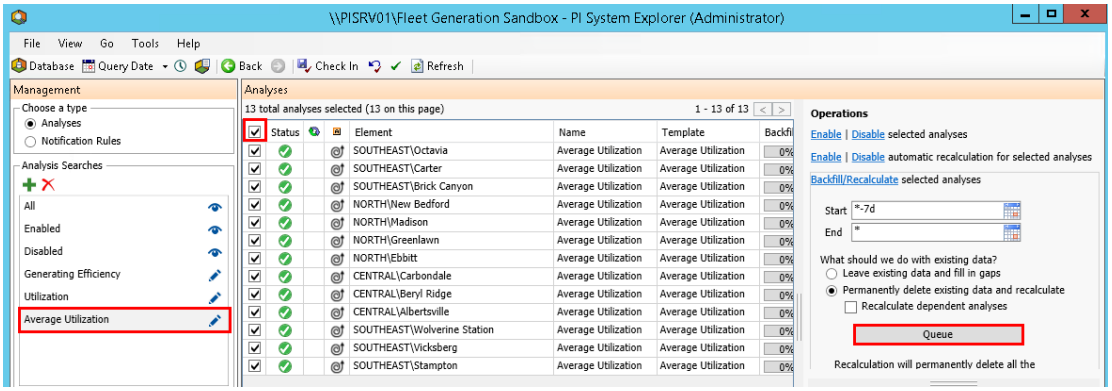
In the **Analysis Templates tab**, Click on the **Evaluate** button to verify the result of the rollup function.



Check-in your changes.

From the element hierarchy, verify that the PI tag exists for the attribute.

From the **Management** pane, backfill your Average Utilization rollup analyses for the past **7 days** and verify the data has been backfilled by trending the Average Utilization attributes.



7.3.2 Exercise – Calculate Total Hourly Gross Generation for Each Station

This solo or group exercise is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the exercise.

Objective:

Management would like to gain more insight into the Total Hourly Gross Generation at each station. Create a **rollup analysis** to totalize the Total Hourly Gross Generation at the station level.

Which station produces the most power?

Approach:

- Open up the Station Element Template from the Fleet Generation Database Library.
- Add a new analysis called Total Hourly Gross Generation with analysis type of Rollup.
- Select Central\Albertsville as the example element.
- Specify the criteria to select the attributes used for the rollup calculation.
- Use the Sum function and output Attribute **Total Hourly Gross Generation**.
- Specify the output attributes (ensure tags are created).
- Set the **UOM** to **MWh**.
- Schedule the calculation to be event-triggered.
- Verify data using Evaluate and Preview Results.
- Backfill for the past 7 days and verify.

8 Event Frame Generation

Events are important process or business time periods that represent something happening that affects your operations. In the PI System, events are known as event frames. Thanks to PI Event Frames, you can analyze your PI data in the context of these events rather than by continuous time periods. Instead of searching by time, PI Event Frames enables users to easily search the PI System for the events they are trying to analyze or report on.

With PI Event Frames, the PI System helps you capture, store, find, compare and analyze the important events and their related data.

PI Event frames represent occurrences in your process that you want to know about, for example:

- Downtime tracking
- Environmental monitoring excursions
- Process excursions
- Product tracking batches
- Equipment startups and shut downs
- Operator shifts

The following table presents some of the features and advantages of PI Event Frames:

Flexibility	✓ Reference multiple elements within the same event. ✓ Support multiple overlapping events on a PI AF element. ✓ Capture any event; a "batch" is just one type of capturable event.
Powerful search	✓ Search by time range, type of event or event frame attribute. ✓ Most common search attributes can be configures as indexed attributes to speed up end-user searches
Scalability	✓ PI Event Frames are extremely scalable.

A PI Event Frame is defined by three characteristics:

1. Name.
2. Start time and end time: defines the event’s time range.
3. Context: event attributes and related assets.

8.1 What are Event Frames?

8.1.1 Creating Event Frames

The Fleet Generation database contains a series of Elements representing the regions and units associated with each generation plant. In order to keep up with the power demands, it is important that the plant is up and running. We need to keep track of the uptime associated with the generation plant.

A 'Unit Status' attribute is associated with each generating plant in our hierarchy. This attribute will be used to monitor the uptime associated with each plant.

8.1.2 Time Range Retrieval Methods

There are three time range retrieval methods, the use of which depends on what data is to be captured, and how it is to be displayed.

Time Range

This method allows a time range to be supplied by the end user. When any single value query is made, this period of time is used for calculations. If, however a period of time is supplied from an application, such as a generated Event Frame or Vision display, then the user specified time range is discarded and the application time period is used.

Time Range Override

The Time Range Override behaves in the same way as the Time Range method during all single value queries, as uses the user specified time period. When a period of time is supplied from an application, the application time range is discarded and the user specified period is used.

Not Supported

Not Supported does not allow for a time range to be supplied by the end user. As such, an error is returned by any request for a single value. If a period of time is supplied however, then this range is adopted by the method for the calculation. The result is then the same from the Time Range method.

There are different use cases for the methods, so care must be taken to ensure the correct method is used.

METHOD	SINGLE VALUE	APPLICATION SUPPLIED
TIME RANGE	User Specified range result	Application Specified range result
TIME RANGE OVERRIDE	User Specified range result	User Specified range result
NOT SUPPORTED	Error: This attribute requires a Time Range to calculate a value in…	Application Specified range result

Single timestamp query results (sample element with 1h specifications)

ⓘ	Not Supported	This attribute requires a Time Range to calculate a value in '…
	Time Range	110.93823012085859
	Time Range Override	110.93823012085859

Application supplied time range query results (sample 3h event frame)

	Not Supported	259.00273501602908
	Time Range	110.93823012085859
	Time Range Override	259.00273501602908

8.1.3 Directed Activity – Create a Temperature Anomaly Event Frame Template

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

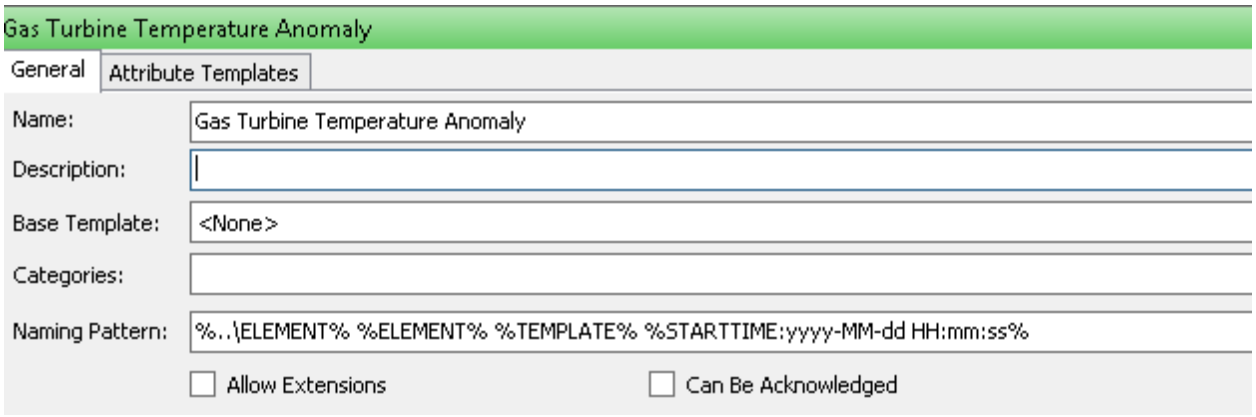
Objective:

The gas turbines in the Fleet Generation database each have two temperature sensors. Create an Event Frame template with appropriate attributes to help monitor and analyze potential issues with gas turbines. The event frame should capture the real-time data specific to gas turbines and the current status and duration of the gas turbine.

Approach:

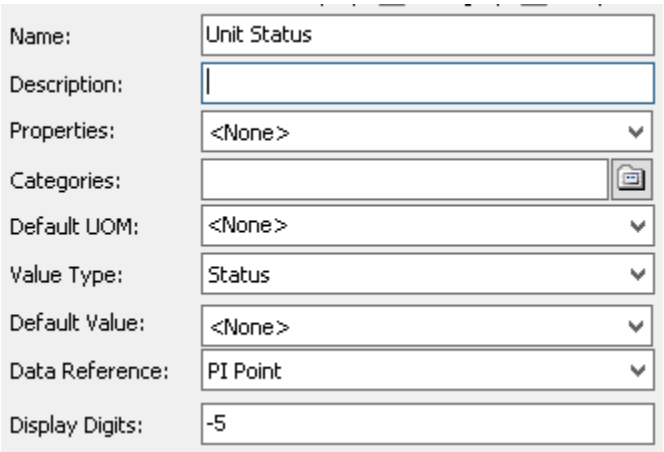
- Create an **Event Frame template**.

Create a template called "**Gas Turbine Temperature Anomaly**". Set the Naming Pattern to %..\ELEMENT% %ELEMENT% %TEMPLATE% %STARTTIME:yyyy-MM-dd HH:mm:ss%



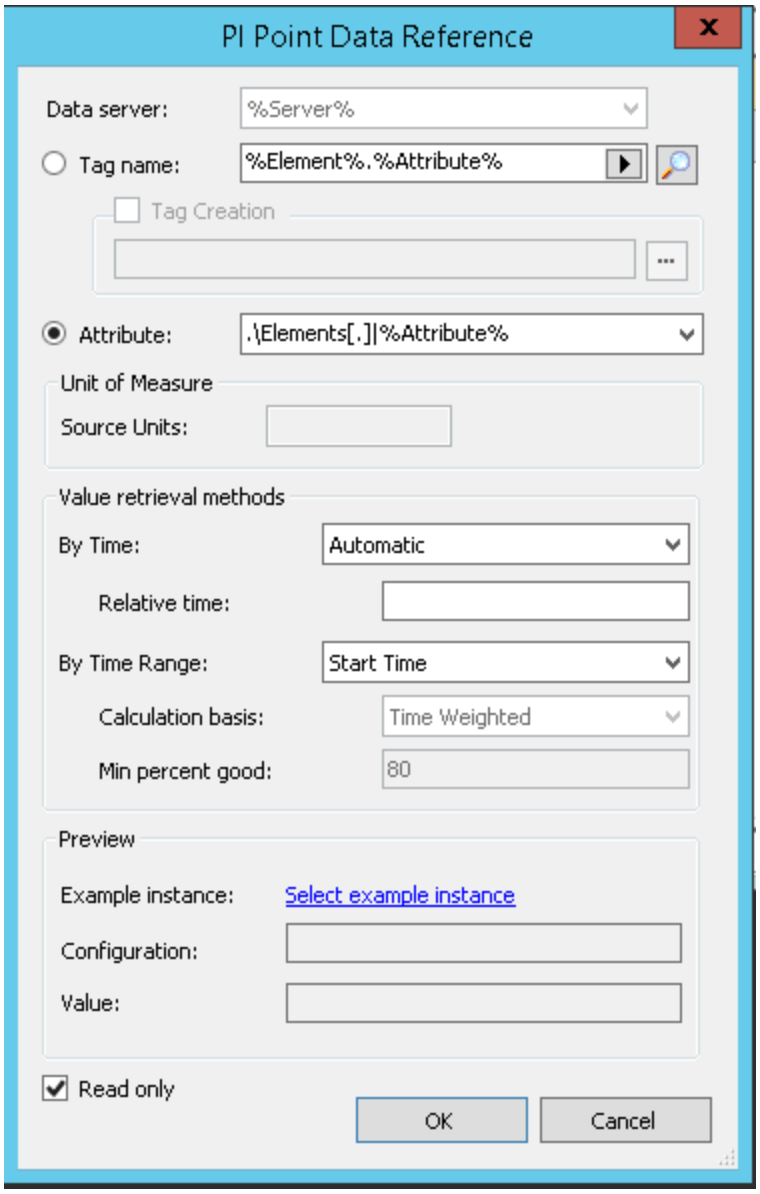
Select the Attribute Templates tab. Right click in the white space to create an attribute.

Name the Attribute **Unit Status**. Select **Enumeration Sets => Status** as the value type.



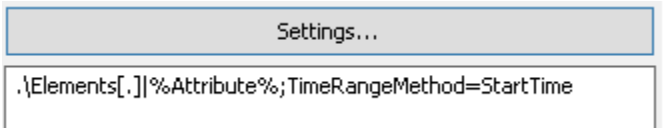
Select the PI Point Data Reference, then select **Settings…**

Click the radio button next to Attribute, and enter **.\Elements[.]|%Attribute%**. The Event Frame references a PI AF Element. The [.] syntax points to this PI EF Template's primary referenced PI AF element within the Elements collection. Set the By Time Range dropdown option to "Start Time."

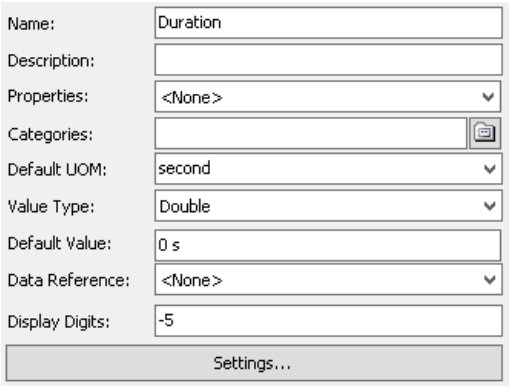


Note: Substitution parameters cannot be used to make a reference to an attribute from the Element Template that is not a PI Tag.

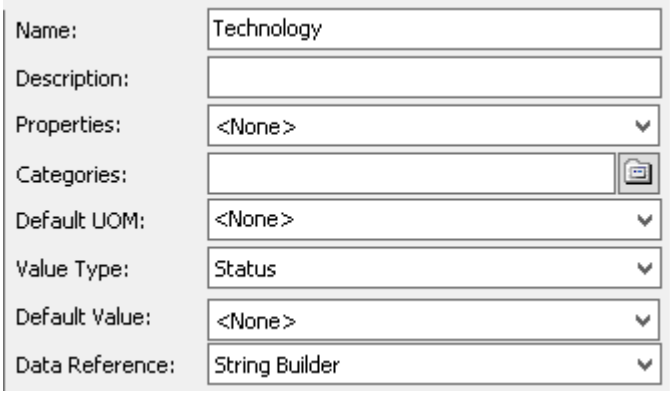
Upon completing the definition, click **OK**. The Settings will be completed as seen below:



Create a second attribute to store the Duration of event frame. **The Duration attribute will be populated by the new EventFrame() function in a later exercise**. It's just a placeholder for now.

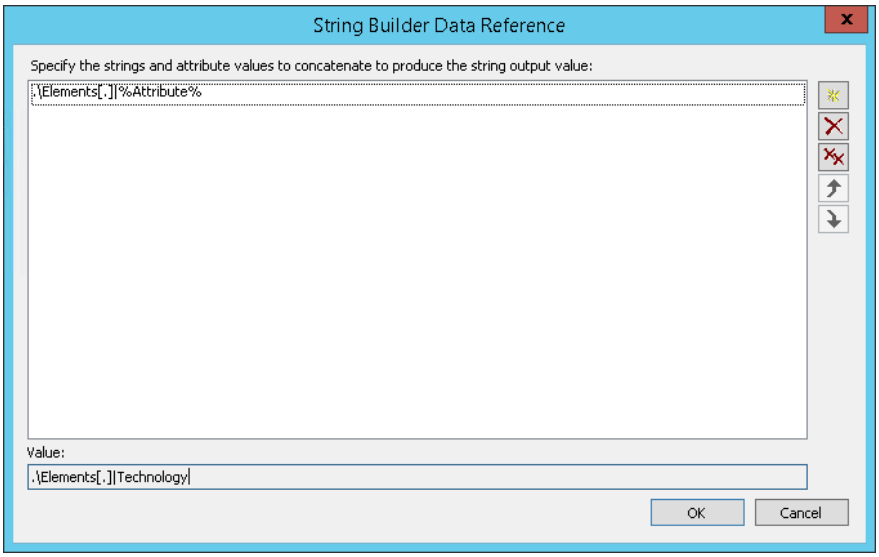


Create a third attribute to store the Technology. For the Value Type, select String and for the Data Reference, select String Builder.



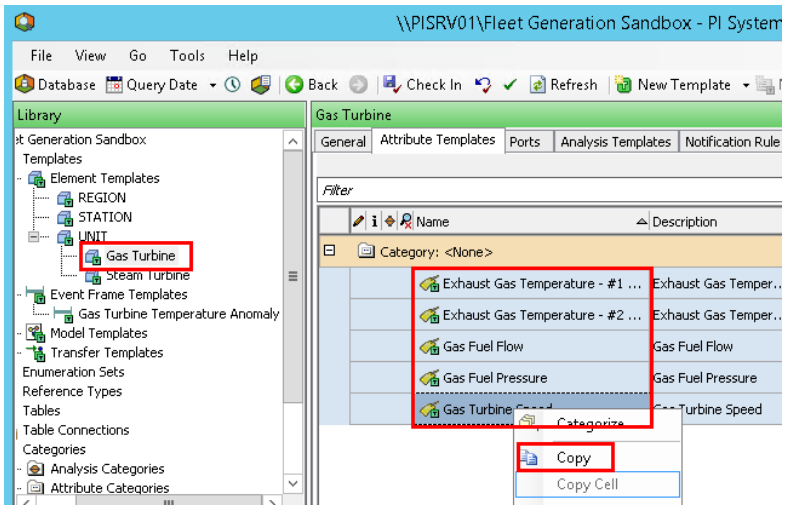
Note: When the event frame attribute's data reference is set to PI Point, the syntax .\Elements[.]|Attribute only allows for the reference to PI Point Data Reference attributes. Element attributes configured as formulas and table lookups cannot be passed to event frames using a PIPoint Data Reference. Instead, for attributes configured as formulas or table lookups, select String Builder as the data reference.

Set the settings for the attribute as .\Elements[.]|%Attribute%:

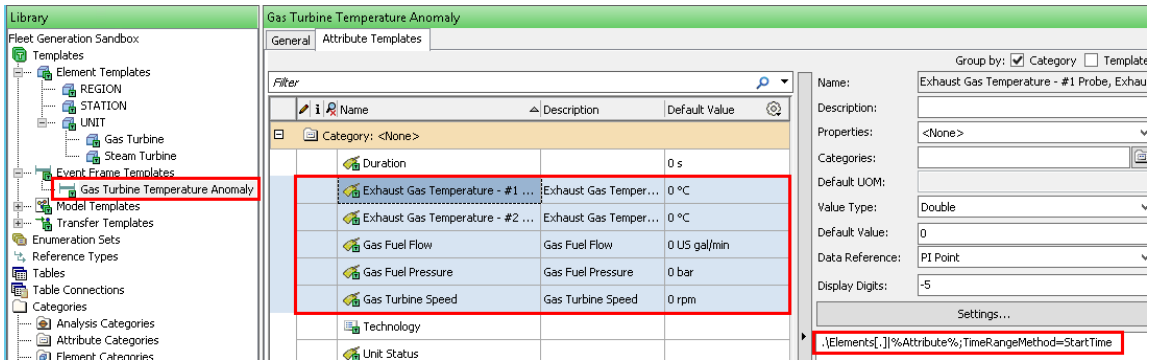


Continue to create the following additional attributes. Make sure units are properly set. **The fastest way to accomplish this is to copy and paste these attributes templates from the Gas Turbine element template**.

Exhaust Gas Temperature - #1 Probe
Exhaust Gas Temperature - #2 Probe
Gas Fuel Flow
Gas Fuel Pressure
Gas Turbine Speed



Once these 5 attribute have been pasted into the Gas Turbine Temperature Anomaly Event Frame Template, select them **all** and enter **.\Elements[.]|%Attribute%;TimeRangeMethod=StartTime** as the configuration string to set the data references and retrieval method in bulk:



Check in your changes

Note: %attribute% will substitute in the name of the event frame attribute template. This will then point to the corresponding attribute in the referenced element. You can also select multiple attributes when making modifications to the attribute configuration.

8.1.4 Exercise - Create Inactivity Event Frame Template

This solo or group activity is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the activity.

Objective:

Generating units sometimes trip or go down. Management would like to understand these downtimes, and determine how much demand was not serviced. Event frames can help capture and bookmark these events for future analysis. Develop an Event Frame template, called **Inactivity** using the same Naming Pattern as the previous exercise, with fields required to track the desired plant information to create reports for management. Specifically, management would like to know the following:

1. Unit Status – Real-time (copy/paste from previous exercise)
2. Duration – in seconds (copy/paste from previous exercise)
3. Technology – Metadata (copy/paste from previous exercise)
4. Hours Down – in hours (simple formula to convert seconds to hours)
5. Demand – Real-time (PI Point data reference)
6. Operator – Metadata (string builder)
7. Carbon Emissions in g/kWh – Metadata (string builder)
8. Total Demand in MWh – Real-time, Aggregation of Demand

Hints:

- For **metadata**, use **String Builder** as the Data Reference.
- For **Total Demand**, configure the attribute's source units as MJ / s By Time as "**Time Range**", Relative time as "**-1s**" and By Time Range as "**Total**"
- Verify correct event frame template configuration through the creation of a test event frame.

8.2 Event Frame Generation

The Event Frames Generation analysis allows for the automated detection and generation of event frames in the PI AF database based on values from trigger attributes. The type of events and the types of data captured inside each event are defined with event frame templates in PI AF.

Some notable features of Event Frame Generation in the PI Analysis Service include the following:

Generate events: Easily configure event generation and automatically generate your events from the trigger tags that are already collecting data in the PI Data Archive.

Handle multiple event types: Generate all your different event types, such as downtime, excursions, batches, and other events, on the same asset with no restrictions on overlapping events.

Standardize using event frame templates and populate event attributes: Different event types have different attributes and information that are important for analysis. Standardize your events using event frame templates, and use the PI Analysis Service to automatically populate event's attributes with data from the PI Data Archive and PI Asset Framework.

Backfill events: PI Analysis Service enables you to define your history backfill time window, then it backfills the events from previous time periods automatically.

Using PI AF element attributes as event triggers or event attribute values: Trigger conditions for event frames can be linked to element attributes.

Configure using PI AF element templates: Apply the configuration of event frame detection and generation to PI AF element templates. The same event detection automatically applies to newly created assets of the same asset type. There is no need to configure the event frame generation again.

Root Cause: Event frames are great for capturing events that have occurred. However, often times, the time period prior to the event provides more information on the cause of the event. PI Analysis Service allows for root cause analysis and will capture a fixed time period (default five minutes) before the event start time for further analysis. This will be recorded as a Child Event Frame.

Time True: The trigger condition for event frames could potentially be noisy. PI Analysis Service allows for the specification of a minimum time true period before an event frame will generate.

8.2.1 Directed Activity – Gas Temperature Anomalies

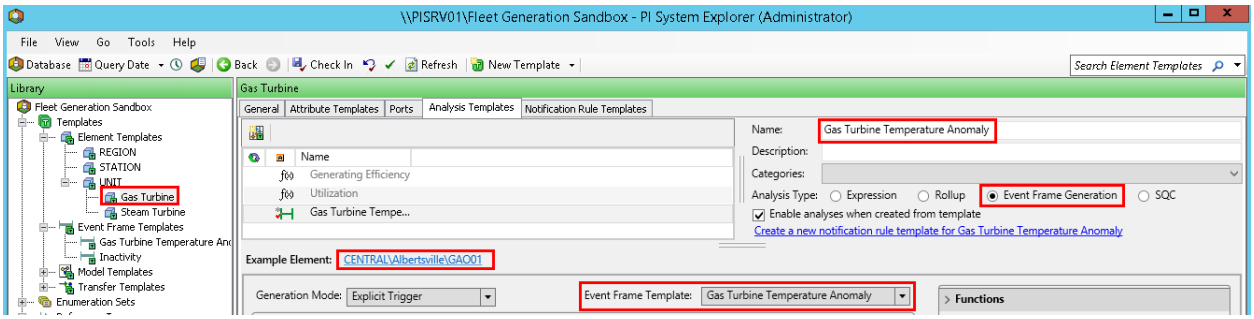
In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

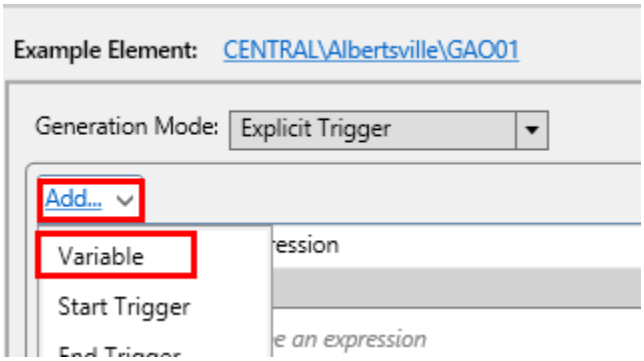
Each gas turbine has multiple temperature sensors. If any temperature reading deviates more than 20% from the average, then servicing is required. Use the Gas Turbine Temperature Anomaly Event Frame Temperature to help define these types of events.

Approach:

From the Fleet Generation Library, select the **Gas Turbine Element Template** and select the **Analysis Templates tab**. Create a new analysis template called **Gas Turbine Temperature Anomaly**, Set the example element to GAO01, and set the Event Frame Template to Gas Turbine Temperature Anomaly.



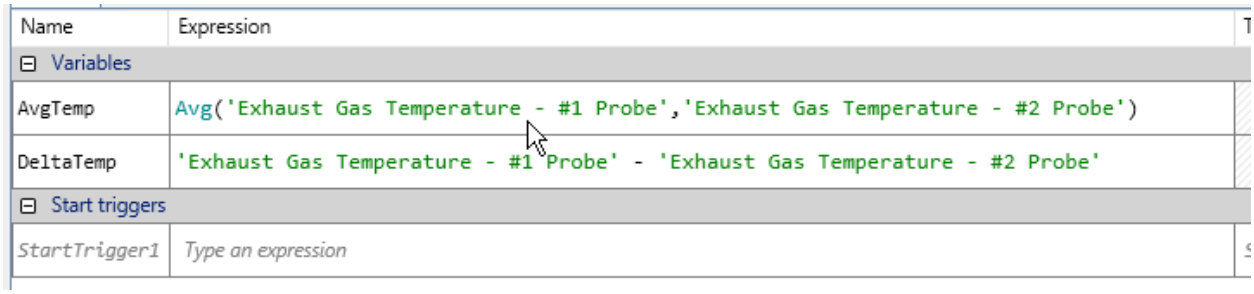
Add two new variables called **AvgTemp** and **DeltaTemp**.



Set the expressions to:

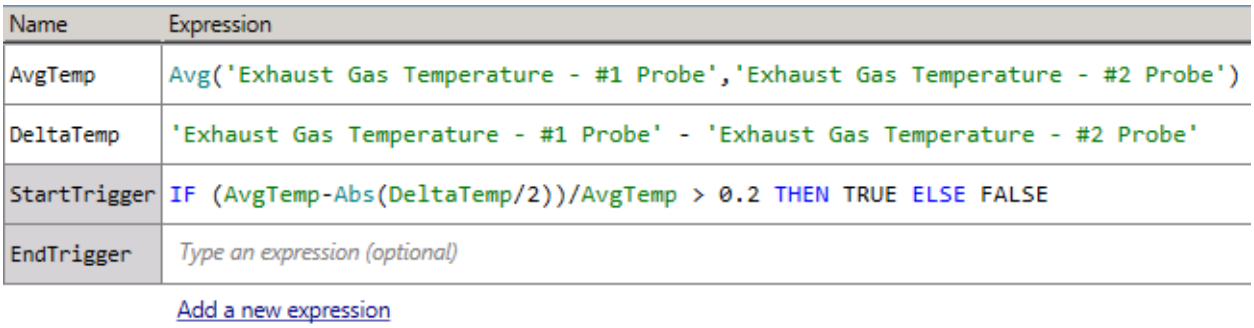
Avg('Exhaust Gas Temperature - #1 Probe','Exhaust Gas Temperature - #2 Probe')

'Exhaust Gas Temperature - #1 Probe' - 'Exhaust Gas Temperature - #2 Probe'

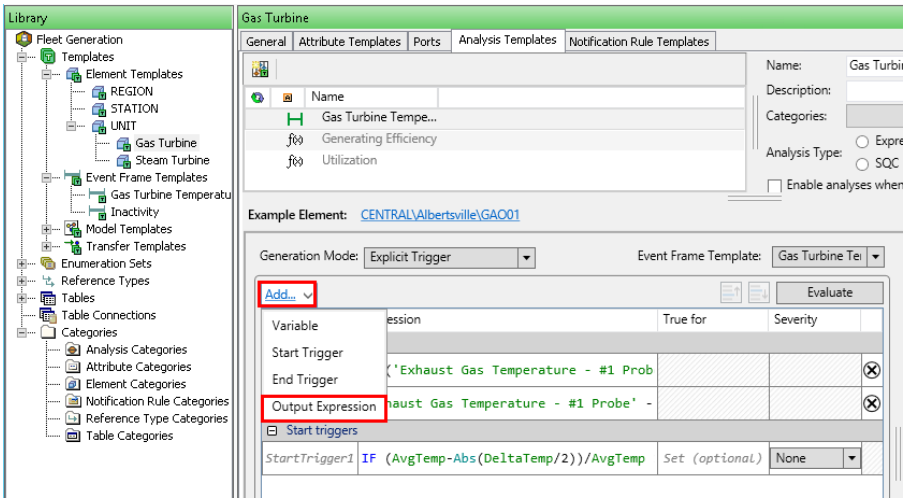


Define the **StartTrigger** as:

IF (AvgTemp-Abs(DeltaTemp/2))/AvgTemp > 0.2 THEN TRUE ELSE FALSE



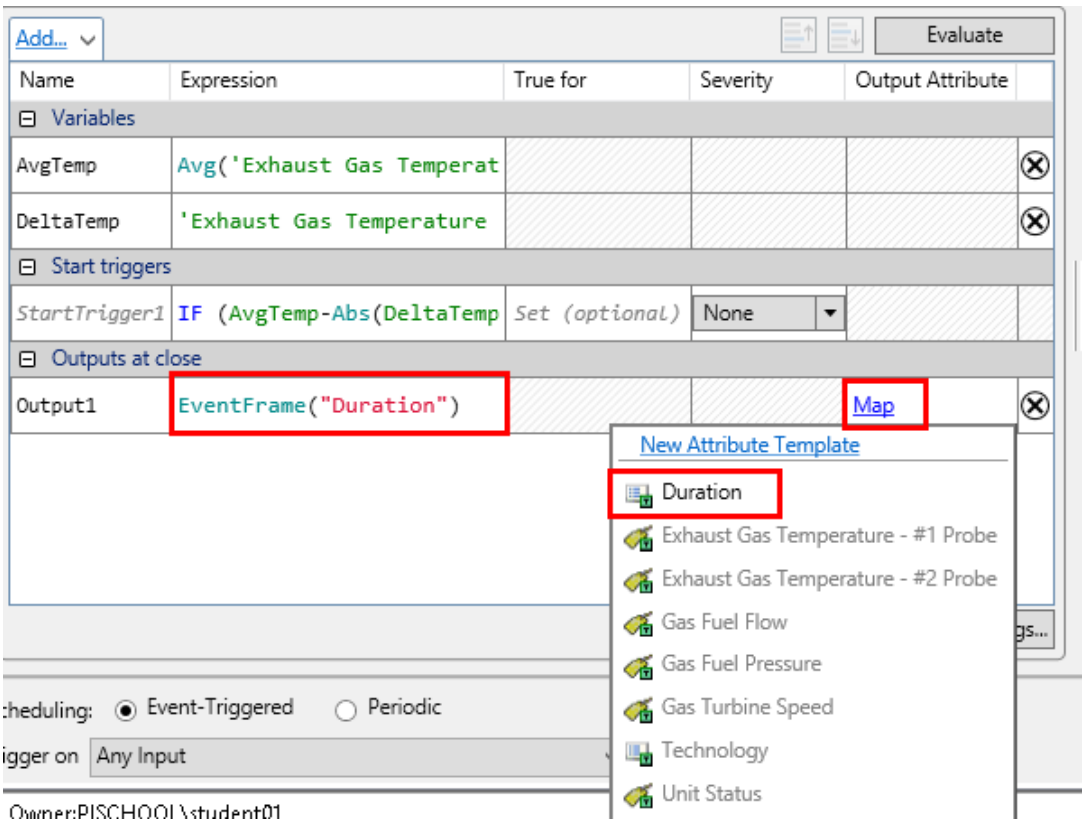
Add an **Output Expression**



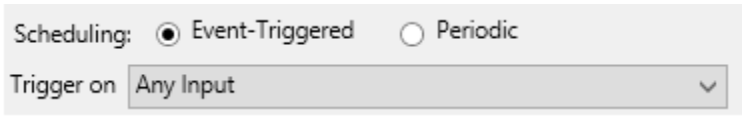
Enter the expression

EventFrame("Duration")

Map the output to the **Duration** attribute

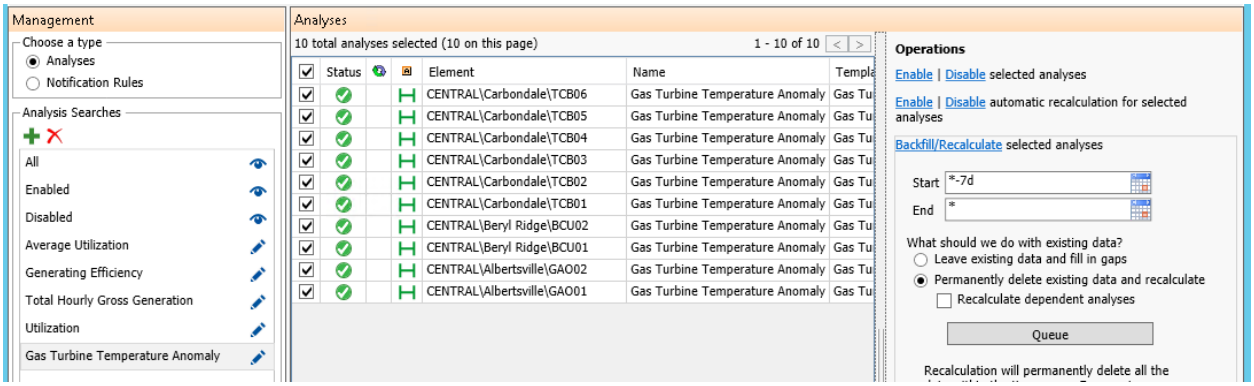


Set the **scheduling** to Event-Triggered and triggering to **Any Input**.



Evaluate and preview the results to confirm there are no syntax errors.

From the Analyses plug-in, backfill event frames for the **past seven days** for **all** Gas Turbine Temperature Anomaly analysis templates.



8.2.2 Exercise - Detect Inactive Units

This solo or group activity is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the activity.

Objective:

Engineering would like to perform a deeper analysis into events over the past week in which the generating units are inactive. Configure the event frame generation to automatically capture new events and detect historical events.

How many inactive events have been occurring?

Approach:

- Open up the **UNIT** Element Template from the Fleet Generation Database Library.
- Add a new analysis called **Inactive Units** with analysis type of Event Frame Generation.
- Specify the event frame template: **Inactivity**.
- Define the trigger condition to automatically detect inactive events.
- Add an Output Expression using the EventFrame(“Duration”) function.
- Verify data.
- Backfill for the past seven days.

8.3 Discussion

This is a discussion designed to maximize learning in a specific topic area. Your instructor will have questions, and will prompt for communication within the class. This is an open-ended section and the result depends on your needs.

Objective: Brainstorm some real world uses for event frames at your own company. Event frames can be used to capture duration and summary information for events such as process excursions or downtime, but how would this be implemented in your workplace?

Approach

- What kinds of events are of interest in your own process?
- Can you think of reliable trigger conditions?
- Do you have all the required data to identify these events?

Estimated Completion time 10 minutes.

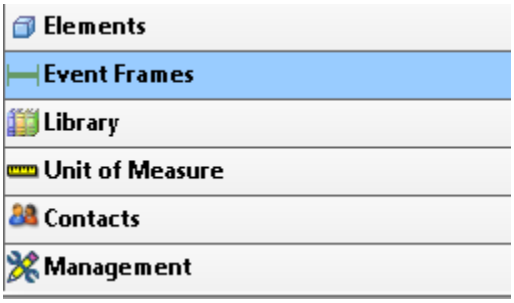
9 Analyzing Events

9.1 Objectives

PI Event Frames are stored in PI AF databases. These event frames can be viewed, filtered, analyzed using PI tools such as PI System Explorer, PI Vision, and PI DataLink.

9.2 PI Event Frames in PI System Explorer

The easiest way to view PI Event Frames is through PI System Explorer. From the Event Frames Pane, you can perform searches against all the event frames within an AF database. You can filter based on specific referenced elements, specific time ranges, and much more.



From the properties of an Event Frame Search, you can specify the following search parameters for the time of the event frame, and the properties of the event frame:

Search type: Specify how to perform an event frame search. Find all event frames that are entirely between a start and end time? Starting or ending between a start and end time?

Search start: Specify the start time for event frame search.

Search end: Specify the end time for event frame search.

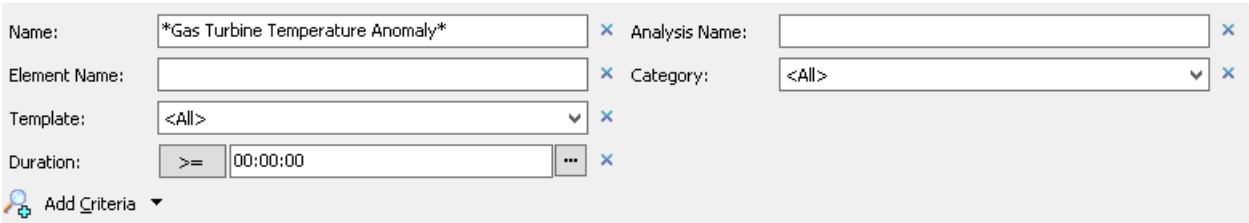
Include descendants: Search for all child event frames in addition to parent event frames.

Event Frame Name: Filter based on the name of an event frame. Can use wildcards.

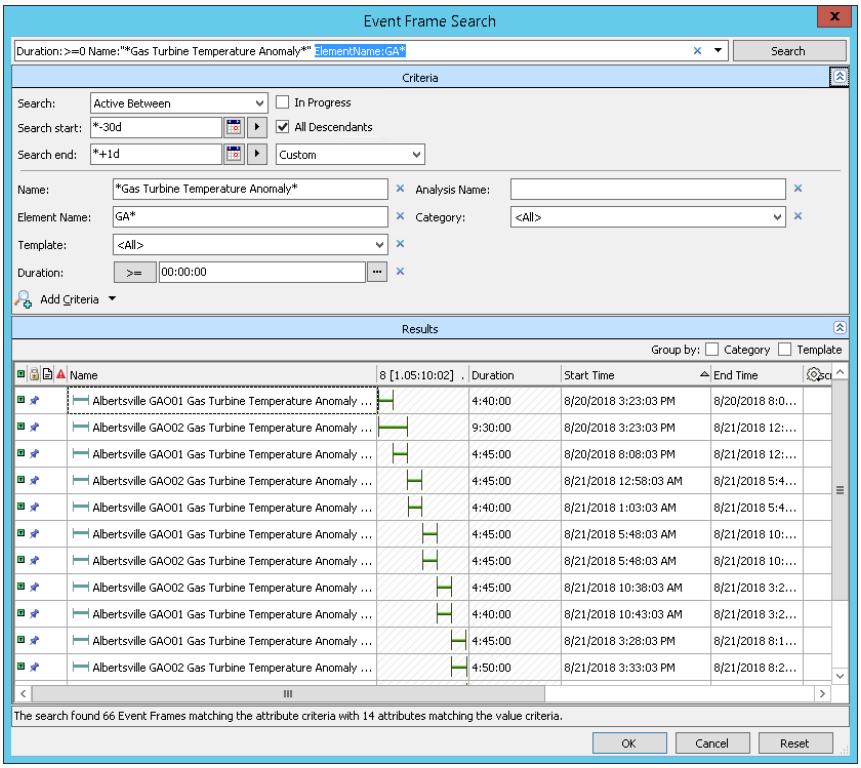
Element Name: Filter based on the name of the referenced element. Can use wildcards.

Template: Filter based on the event frame type.

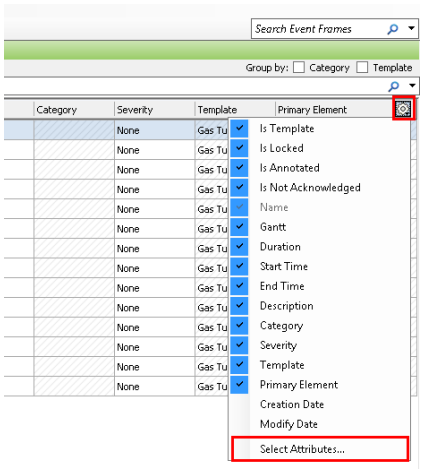
Additional Criteria: Ability to filter based on duration, attribute value, event frame search root, and specify how many results to return.

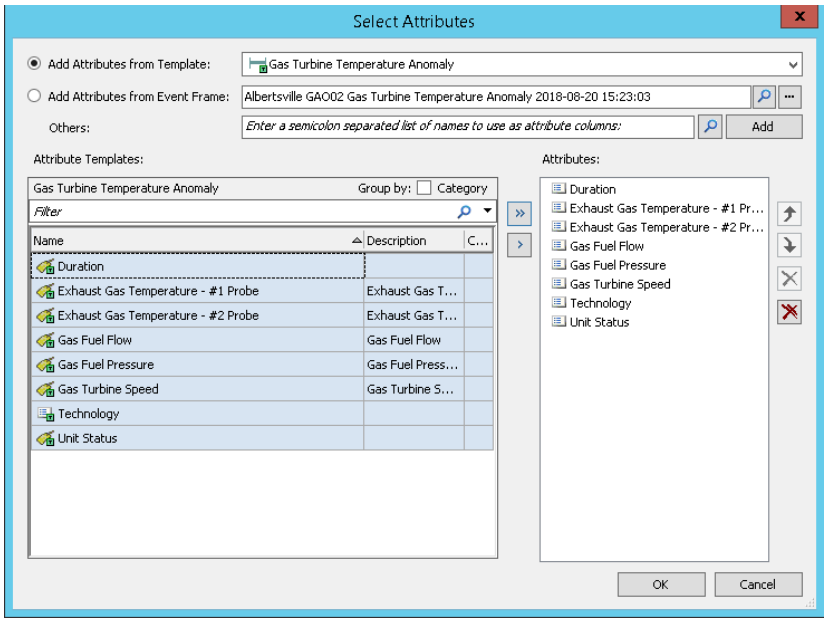


The resulting search query is combined into a string within the search field. This allows for direct manipulation of the data fields without using the menu options.



The default search results bring back fields detailing the duration, start time, end time, description, category, template, and a Gantt chart. Any of these fields can be hidden by using the settings cog on the top right corner of the search results. Additionally, values from the event frame attributes can be pulled back into the search results through this same option list.





9.2.1 Directed Activity – Search for Inactive Events for GAO01

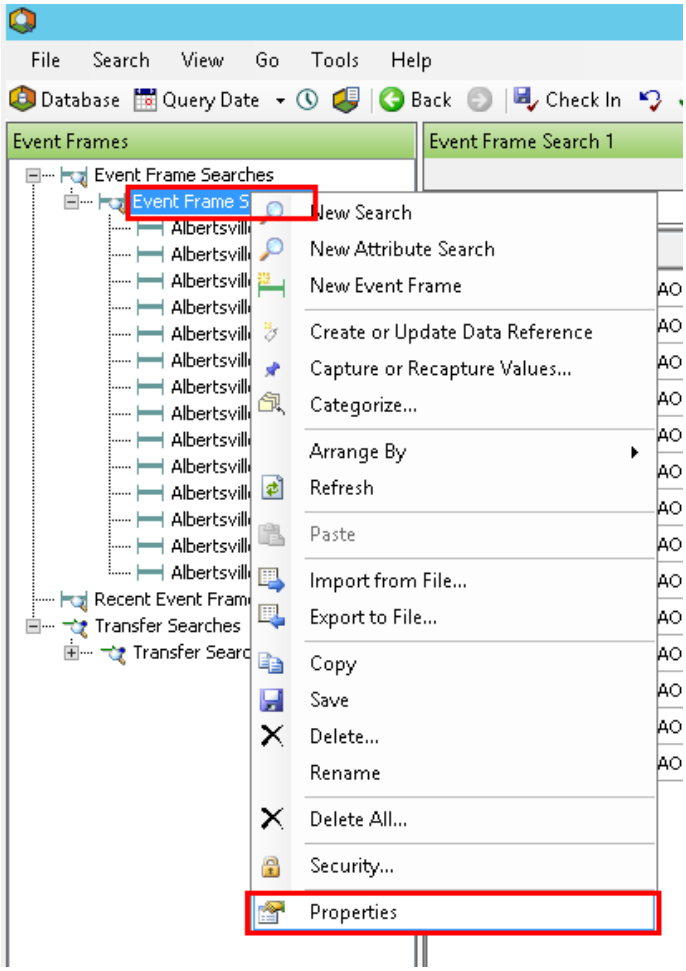
In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

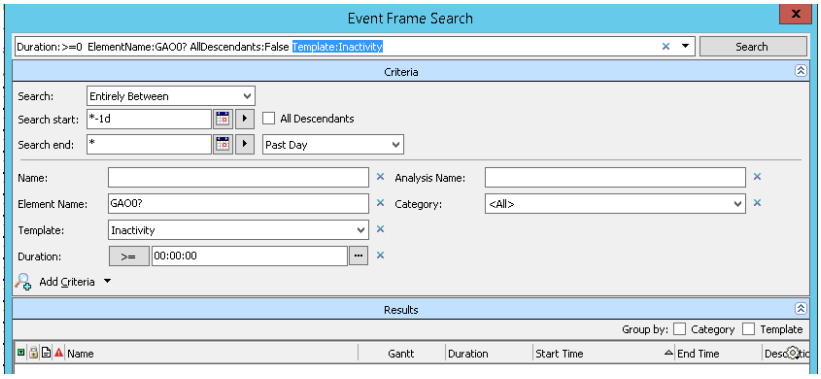
Find all Inactive events for the unit GAO01 and GAO02 over the past 24 hours. Examine the technologies that are involved in these inactive events.

Approach:

Click on the event frame plug-in. Right-click on **Event Frame Search 1** and select **Properties**.



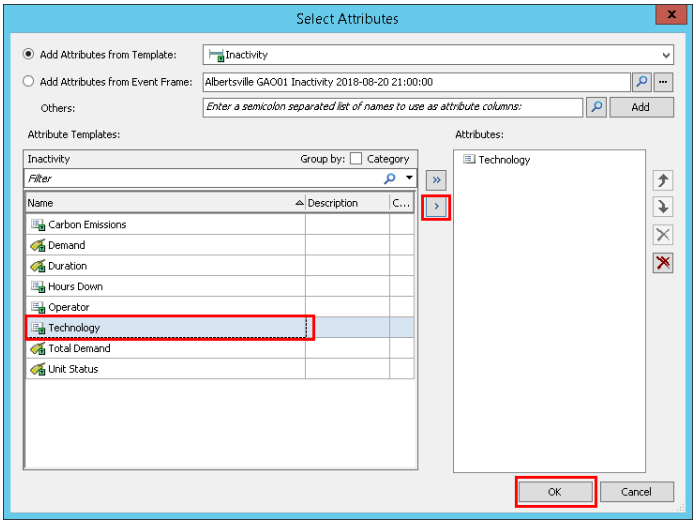
From the Event Frame Search screen, specify the search start to "***-1d**", end to "*****", and uncheck the "All Descendants" checkbox. For the Element Name textbox, specify **GAO0?** and set the Template to **Inactivity**.



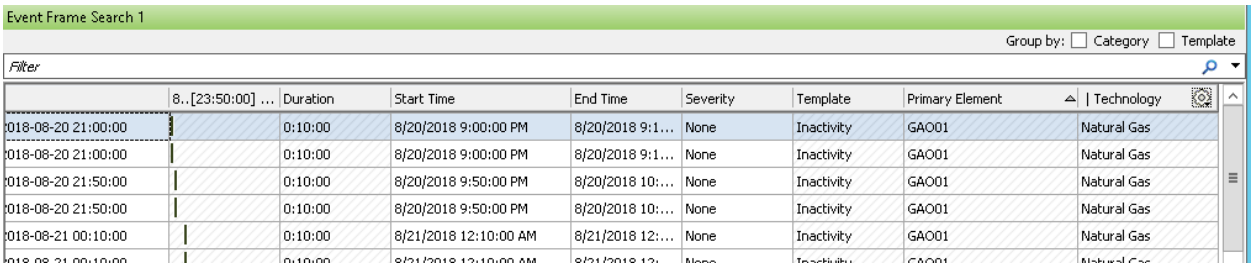
The search will return several inactive event frames. Select **all** of them and click on **OK**.

Click on the gear icon to the right of the fields, and **remove the description and category fields**. Then click on "**Select Attributes."**

Select the **Technology** attribute from the Select Attributes wizard.



Examine the Technology that is leading to the downtime for these Inactive Units.



9.2.2 Exercise – Search for Recent Temperature Anomalies

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

Find all temperature anomaly events for the gas turbines over the past 48 hours that last for more than one hour. Add columns for Fuel Gas Pressure and for each of the two gas temperature sensors.

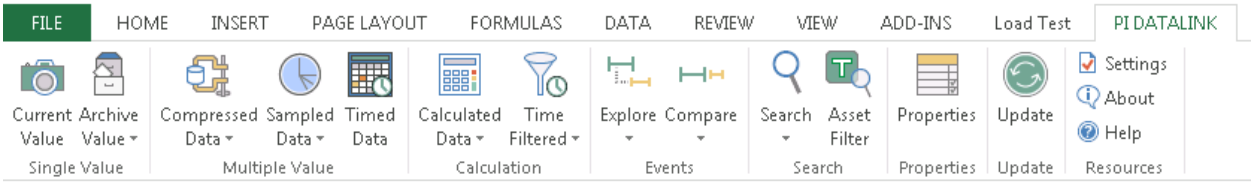
Which unit has the highest starting Gas Fuel Pressure during a temperature anomaly, and when was it?

Approach:

Perform an event frame search and format results for the desired attributes.

9.3 PI Event Frames in PI DataLink

PI DataLink allows you to retrieve current, historical, and calculated data back into Microsoft Excel. In addition to these capabilities, PI DataLink also allows for the retrieval of event frames back into Excel for further analysis.

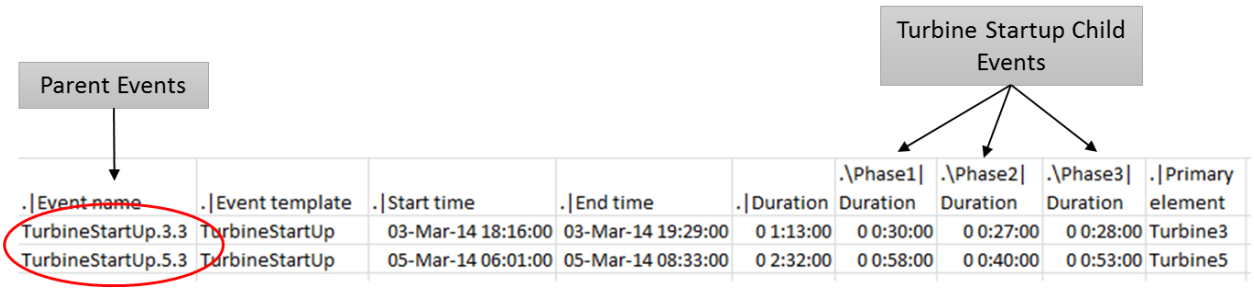


There are two retrieval methods for Event Frames inside of PI DataLink:

Explore: Find Event Frames that meet the specified criteria and display them in a hierarchical format, which is useful to analyze events sharing the same EF template.

Event name	Start time	End time	Primary element	ReasonCode	ShutDownType
BoilerShutDown.5.20130403.1	03-Apr-13 18:00:00	03-Apr-13 19:00:00	Boiler5	P	Planned
BoilerShutDown.5.20130404.1	04-Apr-13 18:00:00	04-Apr-13 19:00:00	Boiler5	P	Planned
BoilerShutDown.5.20130404.2	04-Apr-13 22:04:00	04-Apr-13 23:31:00	Boiler5	E	Emergency
BoilerShutDown.5.20130405.1	05-Apr-13 18:00:00	05-Apr-13 19:00:00	Boiler5	P	Planned

Compare: Find Event Frames that meet the specified criteria and compare their attributes in a flat format. This allows a flat list of events with attributes relating to child events all within a single row.



For either the Compare or Explore Events, you can specify parameters to search for specific event frames. You can specify the following:

Database: AF Database to search against.

Event Name: Search pattern to search for specifically named event frames.

Search Start: Search for all event frames that occurred after this time.

Search End: Search for all event frames that occurred before this time.

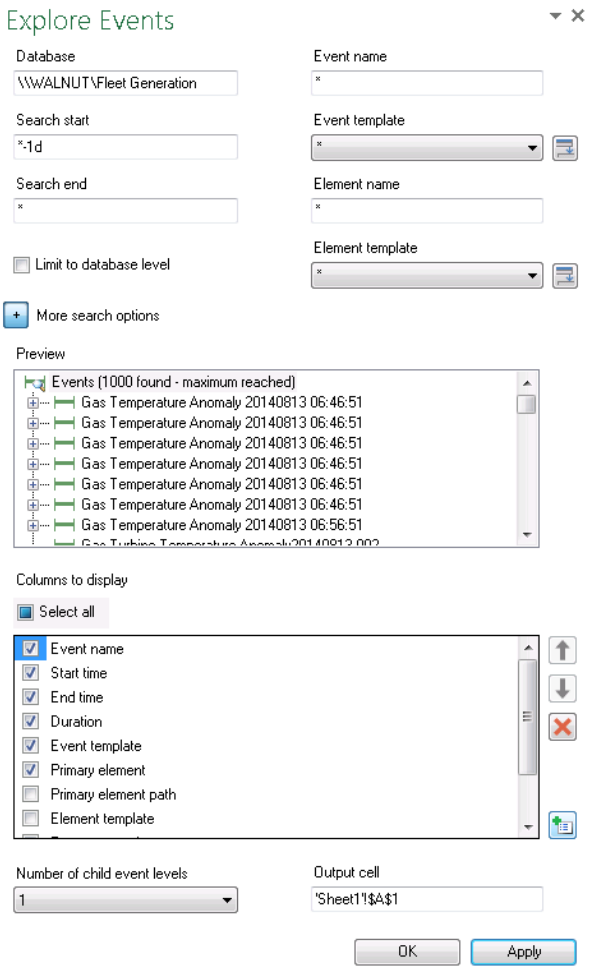
Event Template: Search for specific types of events.

Element Template: Search based off of the type of referenced element.

Element Name: Search pattern for the name of the event frame.

More search options: Search based on attribute values, duration, and category.

Number of child event levels: Only for “Explore Events” and allows for the hierarchical display of events.



Searching for event frames can be based off multiple attributes.

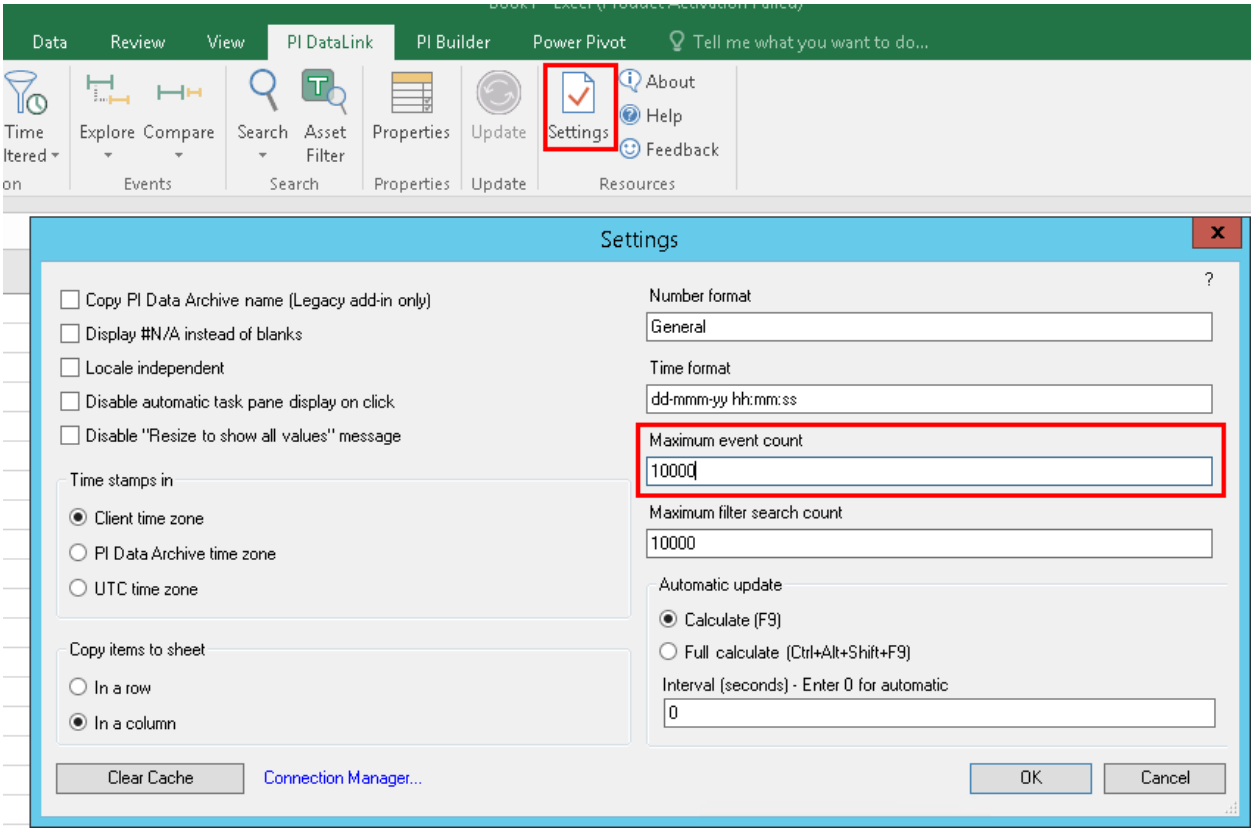
Attribute value filters

Attribute	Operator	Value
Technology	=	Natural Gas
Gas Fuel Presssure	>=	50

When searching with Explore Events, the results can be displayed hierarchically based on the relationships between child and parent event frames.

Event name	Child 1	Start time	End time	Duration
Gas Temperature Anomaly 20140813 11:16:51		8/13/14 11:16 AM	8/13/14 11:51 AM	0.024306
Gas Temperature Anomaly 20140813 11:16:51	Root Cause	8/13/14 10:46 AM	8/13/14 11:16 AM	0.020833

To return more than 1000 event frames in the search preview, go to **Settings** in the ribbon. **Change the setting to 10,000 Event Frames.**



9.3.1 Directed Activity – How many temperature deviations occurred?

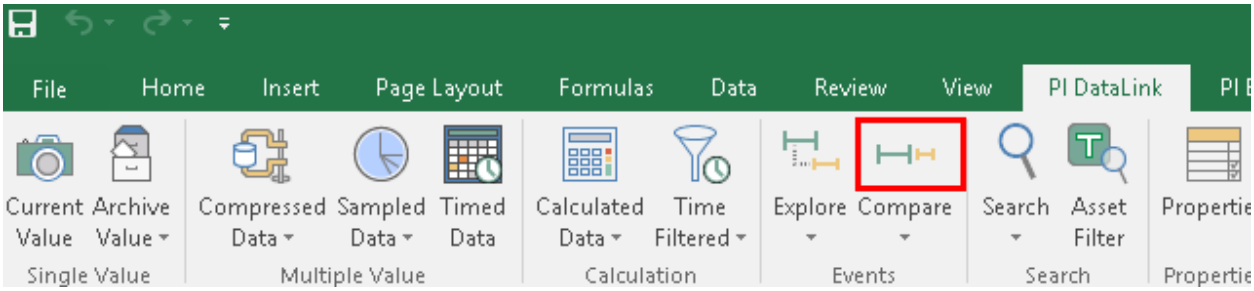
In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

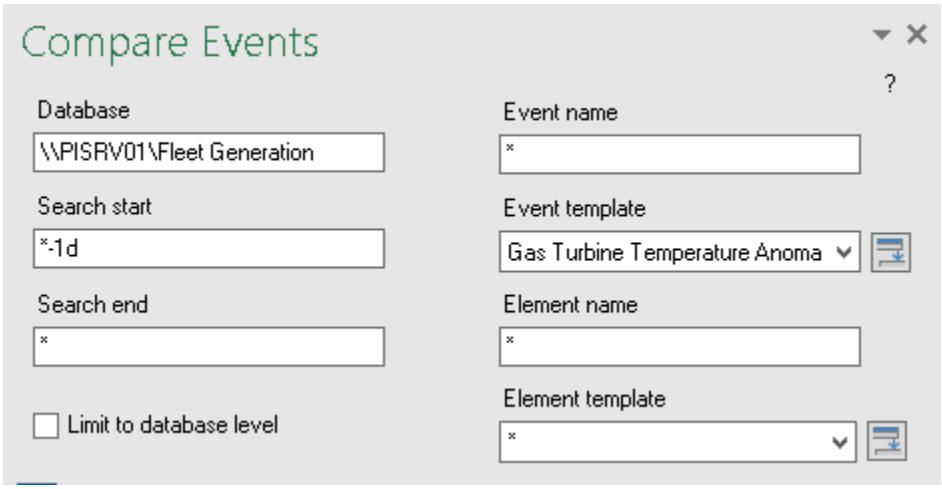
Temperature deviations could potentially mean damaged machinery. Engineering is interested in analyzing the Natural Gas units. Find out how many instances of temperature deviations occurred for gas turbines that lasted for more than 30 minutes.

Approach:

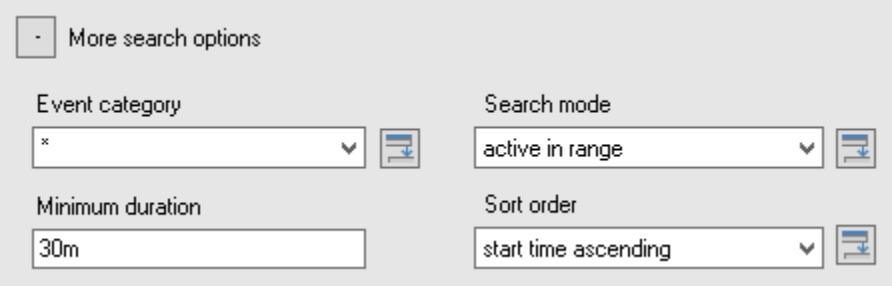
From the PI DataLink tab in of Excel, select cell **A1** and click **Compare** in the ribbon.



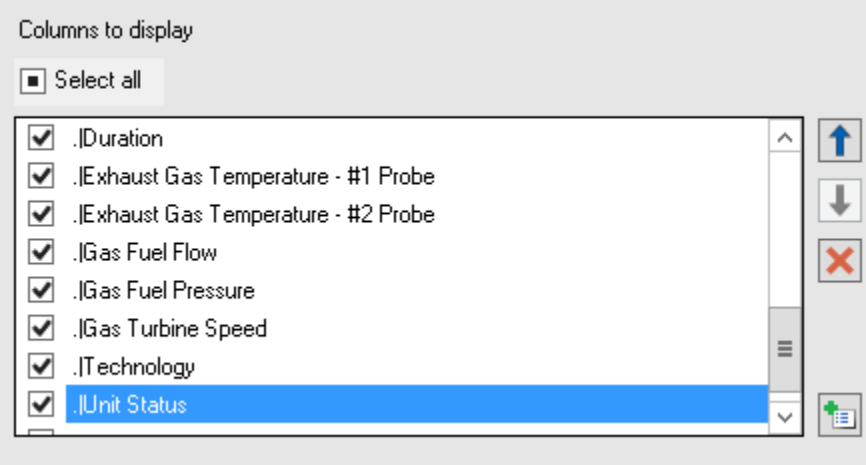
Specify the Database as **\\PISRV1\Fleet Generation**, Event name as "*****", Search start as "***-1d**", and Event template as "**Gas Turbine Temperature Anomaly**."



From **More Search Options**, set the minimum duration to **30 minutes**.



Select the columns that you would like to display:



9.3.2 Exercise – Analyzing Inactivity

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

Inactivity events can be costly as the generating units are not generating any power. Analyze with PI DataLink the total number of Inactivity events as well as the total amount of time the units were in an Inactive state for the 24 hours.

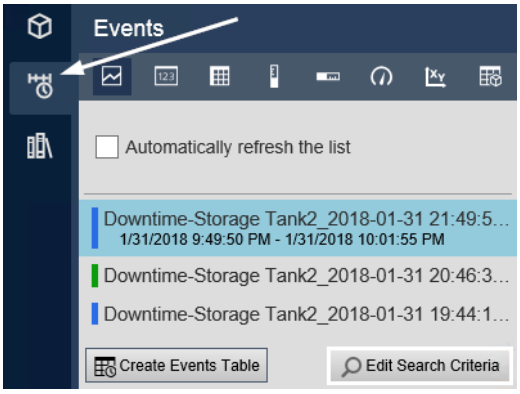
Which generating unit had the most downtime events? Which generating unit had the largest total downtime?

Approach:

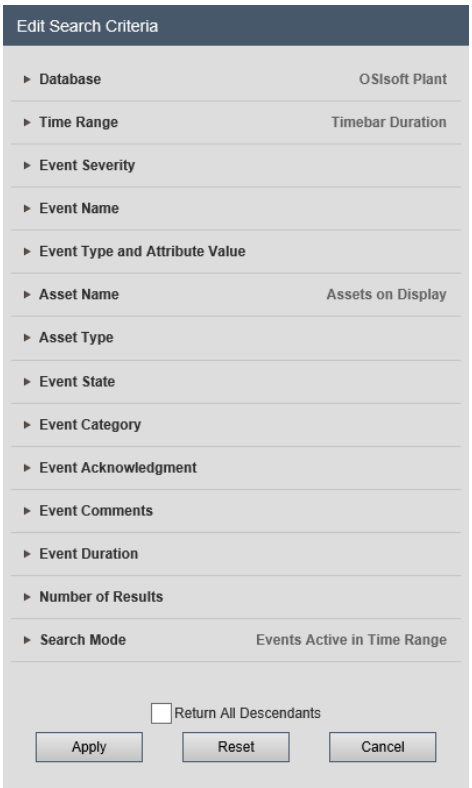
Use PI DataLink to search for PI Event Frames and specify which attributes to return. Use Excel to aggregate the events.

9.4 PI Event Frames in PI Vision

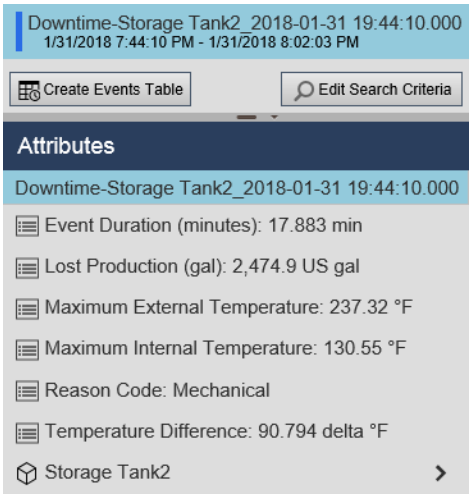
PI Vision enables you to view and analyze your PI data during the time range of a particular event. For example, you may want to examine the performance of an asset during an operator shift or compare the data for several assets during a downtime period.



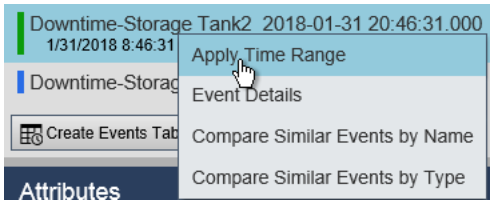
To view events, open the Events tab on the left side. Here you will find events related to your process, the color to the left of each event indicates its severity. By default, the time range of the display and the context of the symbols in the display determine what events are shown in the Events list in PI Vision. To discover additional events, modify the time range or choose *Edit Search Criteria.* When you edit the search criteria, there are a number of filtering options to find the Event Frames you are looking for.



You can select an event to find its Data Items (event attributes) and its start and end time.



By right clicking on an event, you can choose *Apply Time Range* apply the event's time range to the display.



9.4.1 Directed Activity – Inactivity Events in PI Vision

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

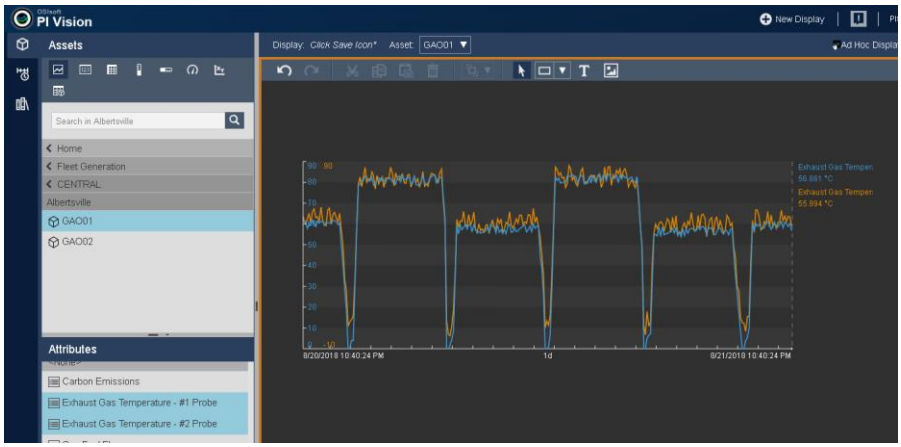
Objective:

Visualize Inactivity Events using PI Vision.

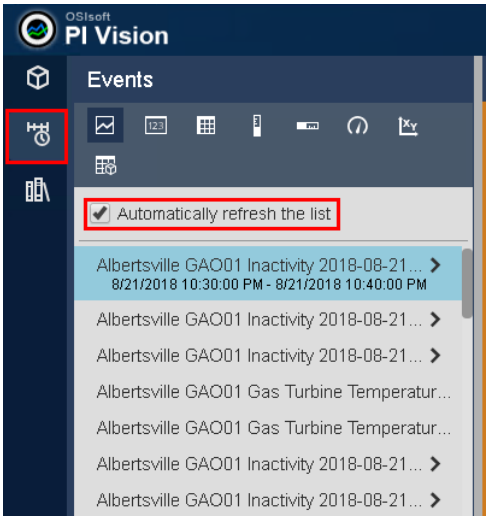
Approach:

Create a new PI Vision display. Drill down to asset GAO01 in the Fleet Generation database

Trend the **Exhaust Gas Temperature Probes** for the past **24 hours**.

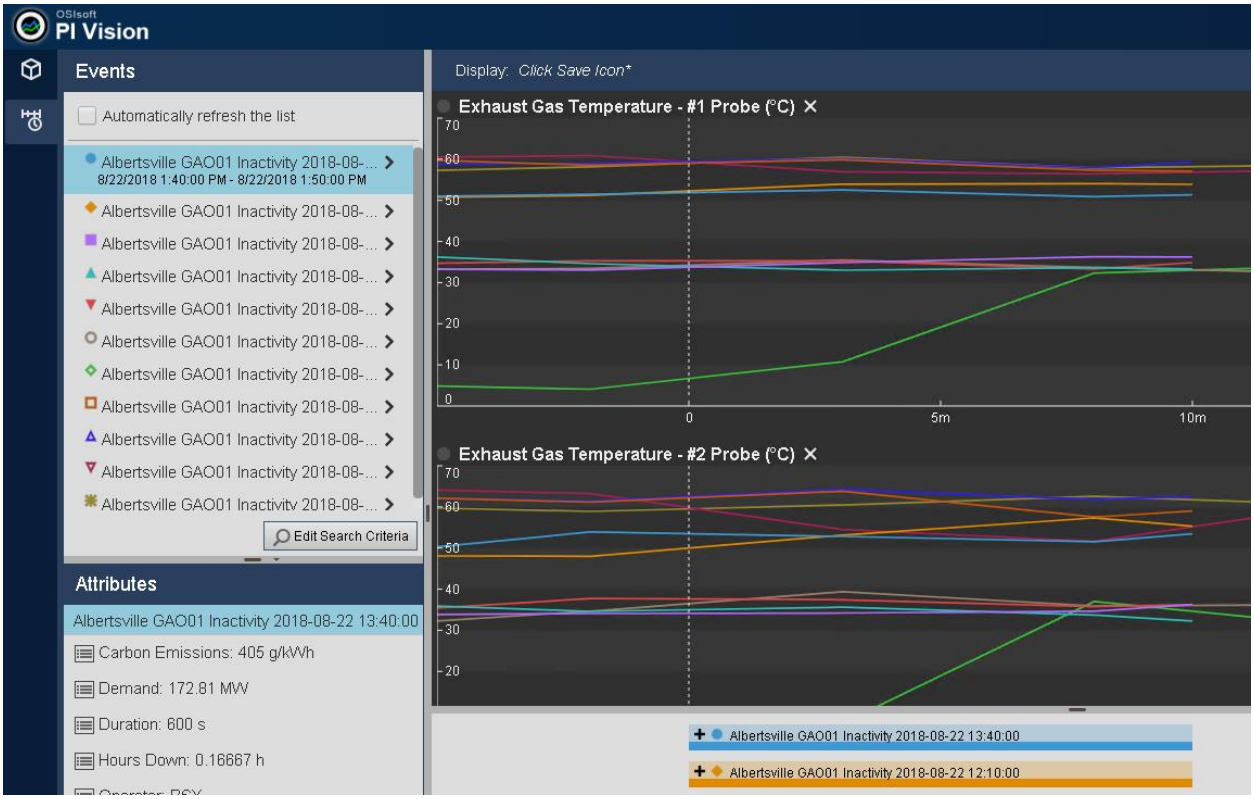


Click on Events in the top left and check “Automatically refresh the list”. By default, this will load Event Frames for Assets on the display (in this case Turbine GAO01).

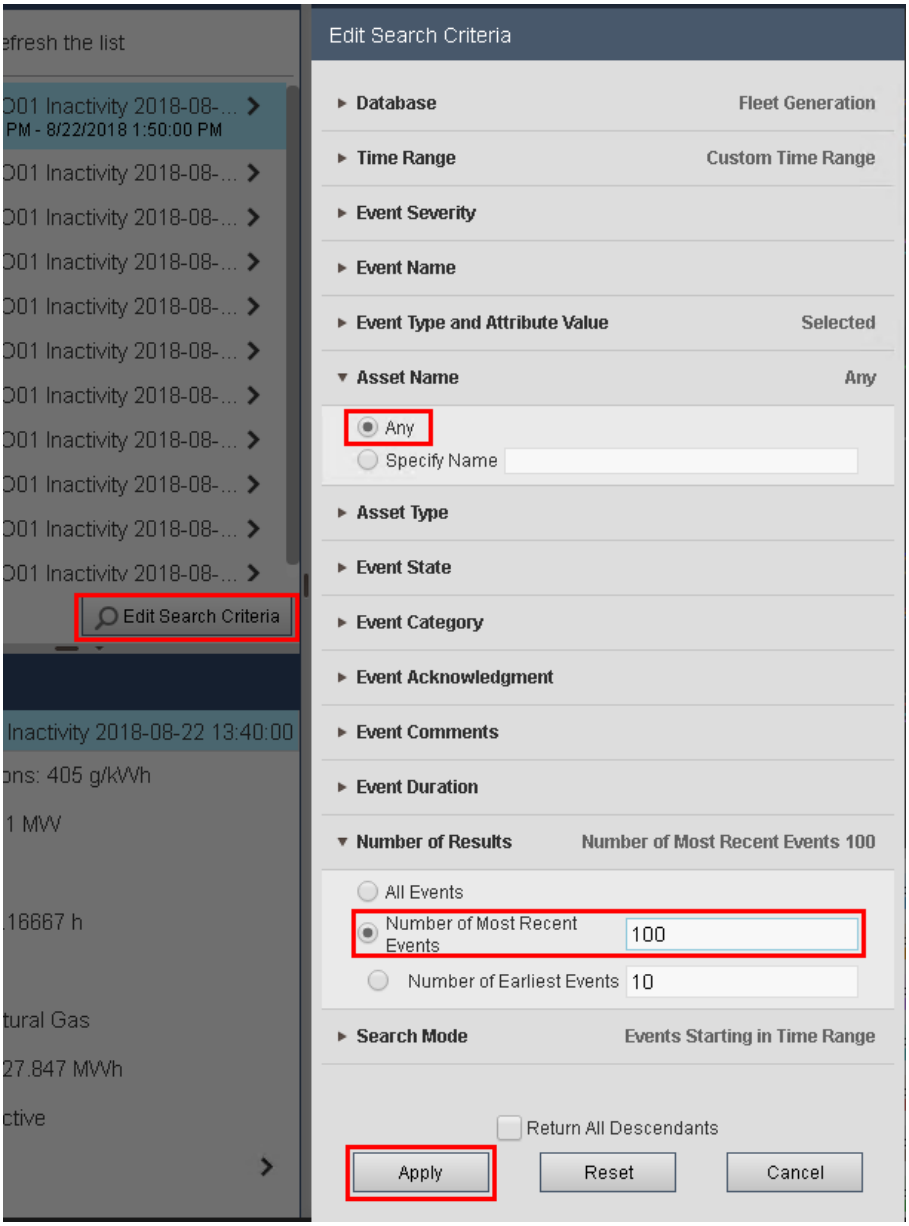


Right-click on one of the **Inactivity** Events and select **Apply Time Range.** The time range will be applied to the temperature trends.

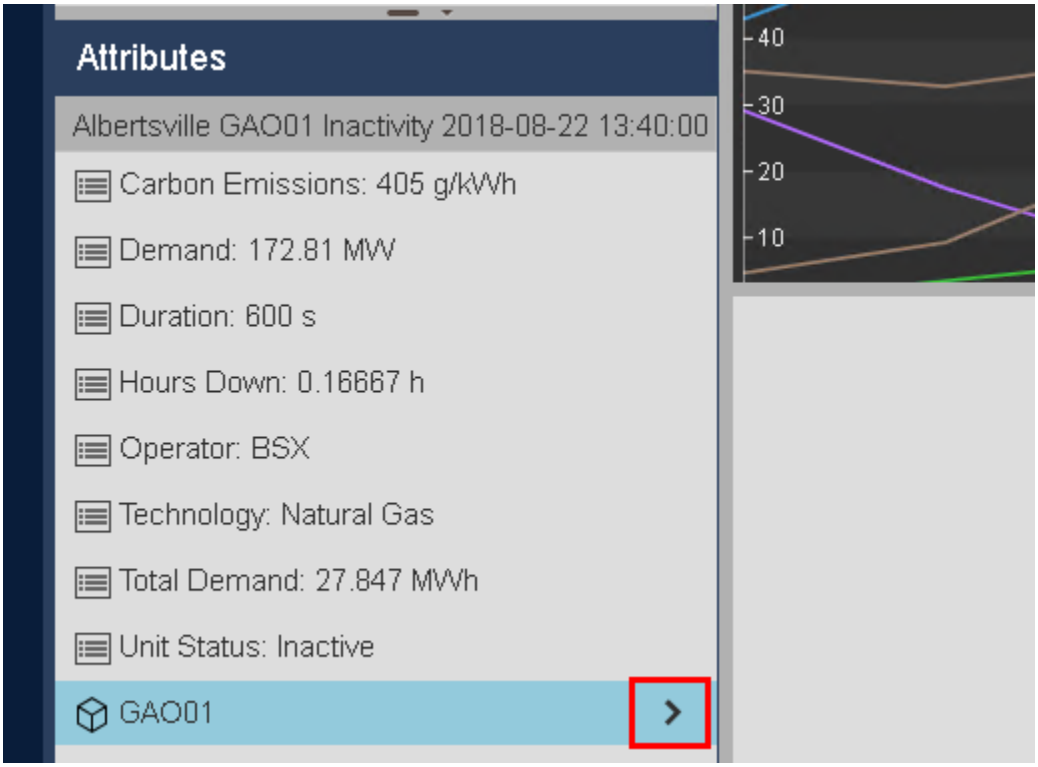
Right-click on one of the Events and select **Compare Similar Events by Type.** Trends of the Event Frame trigger attributes for the selected Event Frame and 10 recent event frames will be shown.



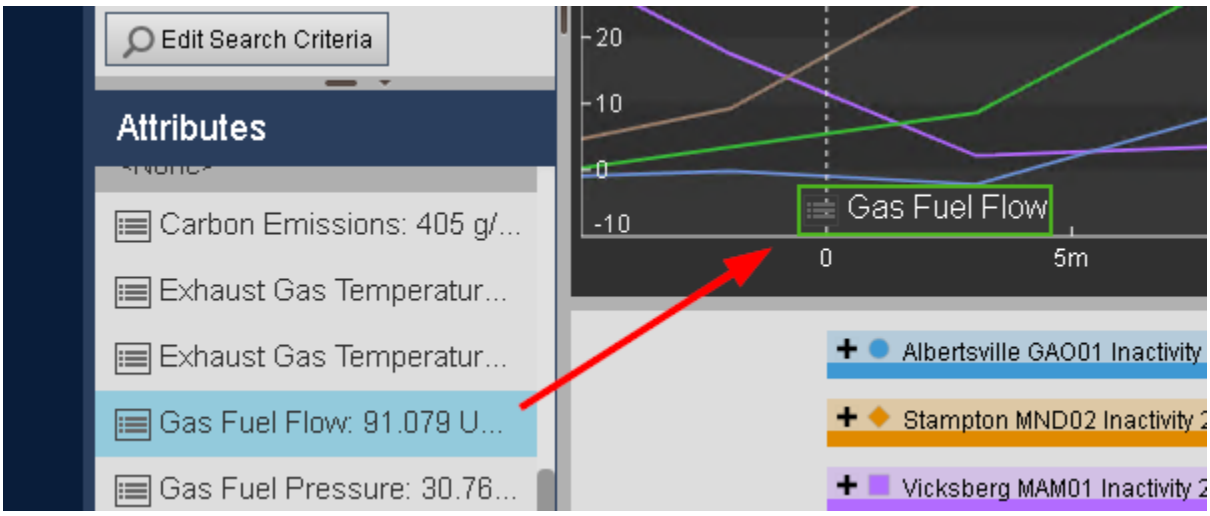
Edit Search Criteria to compare 100 Inactivity Events for All Turbines:



Other attributes from the Event Frames can be trended, but instead we will trend attributes that are not included in the Event Frame but are included in the Asset. In the Attributes Pane, drill into the turbine:



Then drag/drop the Fuel Gas Flow onto the trend area to add new trends



Use the scroll wheel on the right to scroll down and see the new trends

9.5 Discussion

This is a discussion designed to maximize learning in a specific topic area. Your instructor will have questions, and will prompt for communication within the class. This is an open ended section and the result depends on your needs.

Objective: Event frames can be difficult to grasp at first. Let's repeat the discussion from the previous chapter now that you've seen some examples. Brainstorm some real world uses for event frames at your own company. Event frames can be used to capture duration and summary information for events such as process excursions or downtime, but how would this be implemented in your workplace?

Approach

- What kinds of events are of interest in your own process?
- Can you think of reliable trigger conditions?
- Do you have all the required data to identify these events?

Estimated Completion time 10 minutes.

10 PI OLEDB Enterprise SQL Queries

SQL stands for Structured Query Language. SQL is an American National Standards Institute (ANSI) definition for the language used to communicate with relational database systems. It is used by virtually all relational databases in the world today. (Even the PI Data Archive has a SQL Subsystem that can act as a translator to make it “look” like a relational database). SQL Commands are often called “**SQL Statements**.” They can be executed interactively or as stored procedures.

The good part is that it is a standard and that every relational database you encounter will understand it. There is no need to learn many languages. However, there is a down side. Most databases have unique extensions and/or syntaxes that are unique to those systems.

To give a simple example, when passing dates into Access you use pound signs (#) for surrounding dates. On the other hand, in SQL Server you need to use apostrophes (').

Access: [...] WHERE dtColumn >= #2001-11-05#

SQL Server: [...] WHERE dtColumn >= '20011105'

A SQL result set is a set of rows from a database, as well as meta-information about the query such as the column names, the data types and sizes of each column. Depending on the database system, the number of rows in the result set may or may not be known. Usually, this number is not known up front because the result set is built on-the-fly.

This flexibility allows for complex queries to be constructed and saved to return a very specific subset of information from the AF Database that would be either too cumbersome or impossible through the likes of PI System Explorer or PI Datalink.

Trivia: The result is stored in a result table, called the result-set. This table is held in memory.

This is often referred to in code as rs.

10.1 Dissecting the Syntax

A common SQL syntax starting command is **SELECT** which is used to query the database. The data retrieved from the statement is based on the criteria specified in the SELECT statement.

Following the **SELECT** command identifies the columns to be selected from the tables(s).

SELECT * - retrieves all the columns from the table being referenced.

SELECT column1, column2, column3 – retrieves 3 columns of the table being referenced.

The **FROM** command identifies the first (or perhaps only) table being queried.

SELECT * FROM tablename – retrieves all the columns from tablename.

SELECT column1, column2, column3 FROM tablename – retrieves all data for the 3 columns of tablename.

The **WHERE** command contains criteria to filter the data being retrieved.

The conditional operators include:

equal (=)
greater than (>)
less than (<)
greater than or equal (>=)
less than or equal (<=)
not equal to (<>)
LIKE (which is a pattern matching operator)

Note: If the conditional clause is set to compare to text, the text value is encased in single quotes ('text').

SELECT * from tablename WHERE column1 = 5

Retrieves only rows where column1 has a value equal to the number 5.

AND and OR statements

- **AND** indicates both statements must be TRUE for the row to be returned when the query is executed.
    - SELECT column1, column2, column3 from tablename WHERE column1 = 5 **and** column2 = 'junk'
    - Retrieve only rows where column1 has a value equal to the number 5 **and** column2 value equals junk.
- **OR** returns data rows if either condition is met
    - SELECT column1, column2, column3 from tablename WHERE column1 = 5 **or** column2 = 'junk'
    - Retrieve only rows where column1 has a value equal to the number 5 **or** column2 value equals junk.

The **LIKE** operator is used to search for a specific pattern in a column. In conjunction with the LIKE operator a **wildcard of %** is used for comparison. The % can represent a single character or multiple characters. Another wildcard is the underscore (_) which can be used to represent a single character.

SELECT * from tablename WHERE column2 LIKE '%unk'

Retrieves rows from tablename where column2 values end with the letters 'unk'

SELECT * from tablename WHERE column2 LIKE '%un%'

Retrieves rows from tablename where column2 values contain the letters 'un'

SELECT * from tablename where column2 like '_un_'

Retrieves rows from the tablename where column 2 values only contains 4 characters and the middle two characters are un.

SELECT * from tablename WHERE column2 LIKE 'j%'

Retrieves rows from tablename where column2 values start with the letter 'j'

To work with column/table names which have special characters, such as a space, use square brackets:

If you wish to SELECT a column called *Product Orders*, enclose it in square brackets: [Product Orders]

If you're referring to a table whose full path is *Fleet Generation, Region, Station, Unit*, that must be written as [Fleet Generation].[SouthEast].[Brick Canyon].[PLT02]

Any name may be wrapped in square brackets, so when in doubt as to what constitutes a special character, wrap the name in square brackets.

10.2 PI OLEDB Provider or Enterprise? What's the difference?

PI OLEDB Provider is an OLEDB data provider that provides access to the PI System. Given the correct security, the PI OLEDB Provider allows read/write access to the PI System Archive.

PI OLEDB Enterprise is an OLEDB data provider which provides access to the PI System in a relational view, accessible through SQL queries. The PI OLEDB Enterprise provider supports read-only access to asset and event data stored in the PI Asset Framework (AF), such as AF Elements, AF Attributes and PI Event Frames.

Both data providers can be called by using the PI ODBC via a SQL DAS server to allow remote query execution, minimizing the required software to be installed and configured on endpoint machines.

10.2.1 PI SQL Commander

The PI OLEDB Enterprise installation includes a test environment which handles the OLE connection process and allows the user to execute queries and perform other tasks. This test environment is *PI SQL Commander.*

PI SQL Commander is the user interface to assist with creating queries, transpose functions, and views against PI AF using PI OLEDB Enterprise. This user interface also provides access to the classic PI OLEDB Provider which builds queries against the PI Data Archive components without knowledge of PI AF.

PI SQL Commander Lite is an application to navigate a relational view of the PI System using SQL Queries that are exposed by PI ODBC. This can be used to create, edit, test and save SQL queries of PI System data. It does not support the more advanced functions found within PI SQL Commander however, like transpose value function creation, or view creation. Previously created transpose value functions or views may be used however.

10.2.2 Directed Activity – Review Predefined Queries

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

Review predefined queries associated with the tables defined in PI SQL Commander.

Approach:

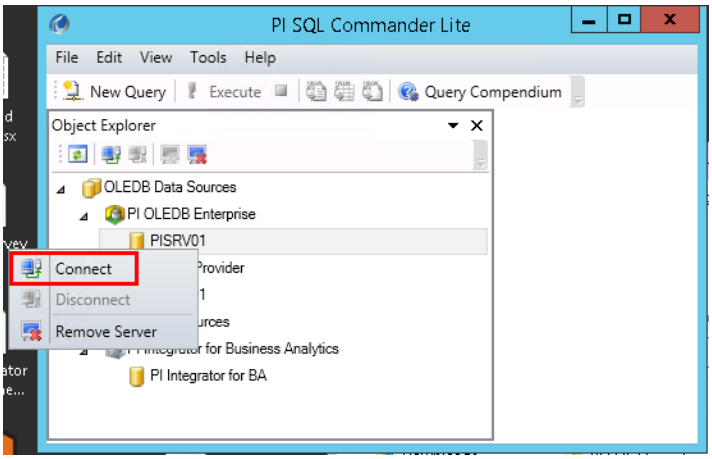
- Open PI SQL Commander
- Navigate to the Fleet Generation Database/Catalog
- Execute a Predefined Query associated with the Element Hierarchy table.

Launch PI SQL Commander -
Click Start > All Programs > PI System > PI SQL Commander.

In **PI SQL Commander**, verify that your **PI SQL Object Explorer** is visible. If it is not, click **View > PI SQL Object Explorer**.

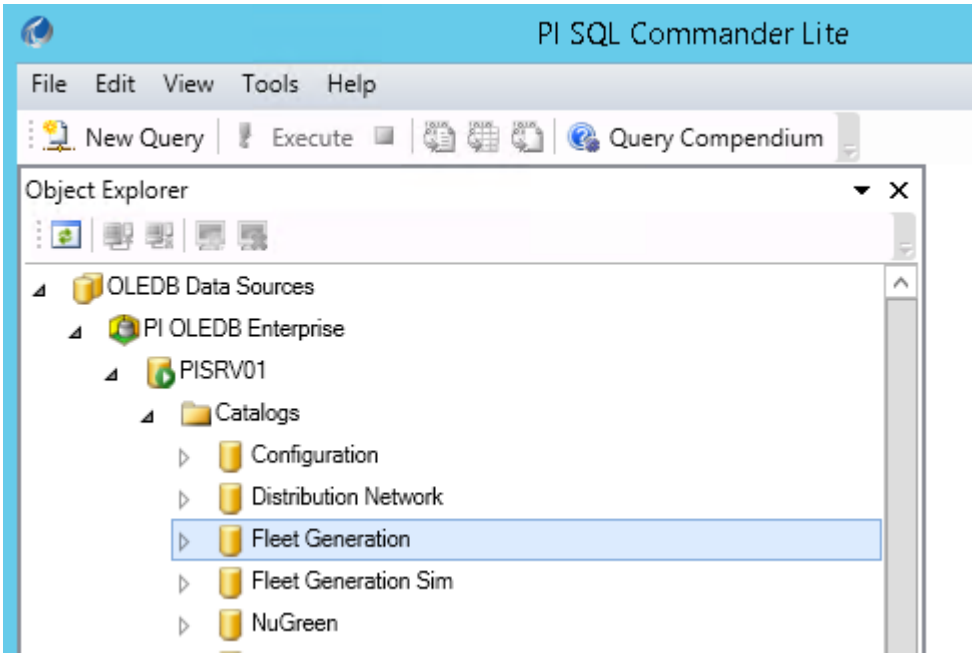
Through PI SQL Commander, either a PI AF Server or a PI Data Archive can be accessed through SQL statements based on the item selected for connection.

From within the PI SQL Commander Object Explorer, connect to your AF Server, in this example, **PISRV1**, by right-clicking Select **Connect** then select **Windows Integrated Security**.



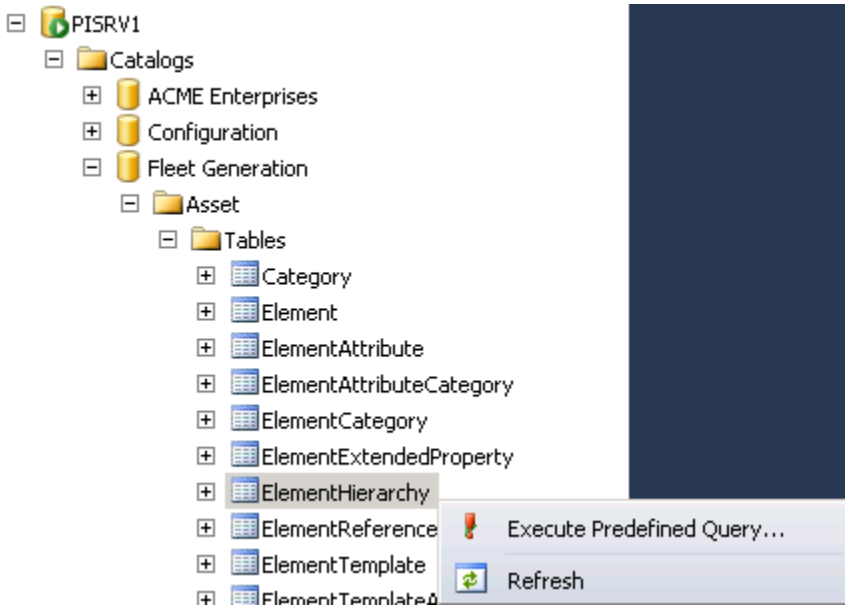
An arrow next to the server icon indicates that the connection is successful:

After connecting to the PI AF server, you will see a catalog list in the PI SQL Object Explorer. The catalogs listed correspond to each of the PI AF databases you have configured for this PI AF server. We will be using the Fleet Generation database throughout this course.



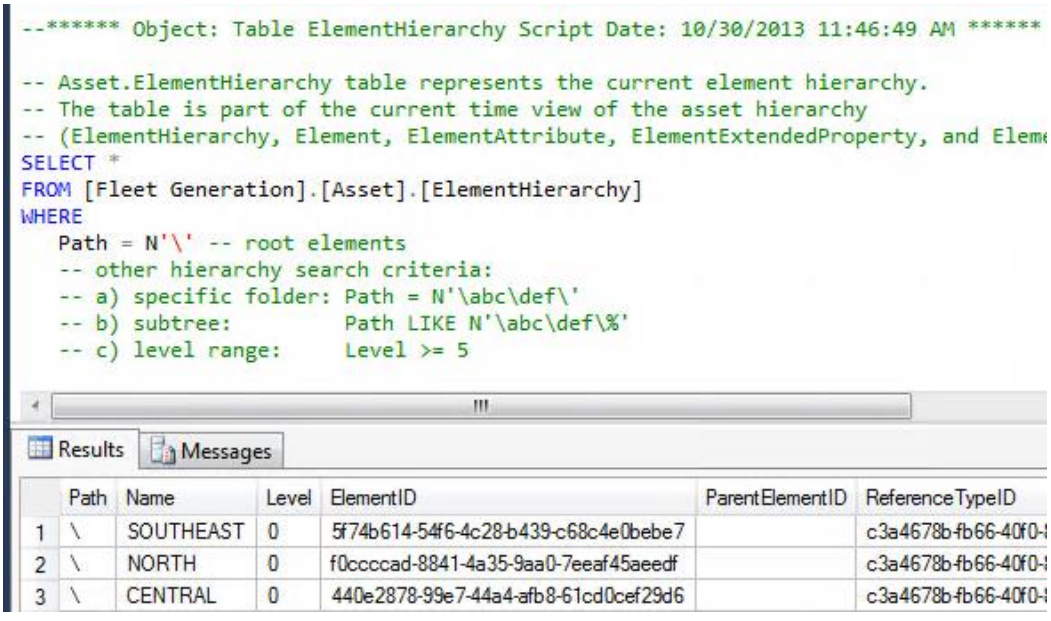
Right-click an object in the catalog that represents a table, view or function, and then select **Execute Predefined Query**.

PI OLEDB Enterprise includes one sample SQL query for each table in the catalogs for PI AF server.



This is the environment for building and testing PI OLEDB Enterprise queries. Queries, written in the editor, can be executed with their results shown in the grid.

Upon selecting Execute Predefined Query, a query window will appear with a Select statement for the ElementHierarchy will be developed.




```
SELECT *
FROM [Fleet Generation].[Asset].[ElementHierarchy]
WHERE
  Path = N'\' -- root elements
```

The above query does not yield all elements, just the elements at the Region level.

(Note: the “N” declares the path string to be Unicode, which permits lots of different characters. It will be omitted throughout this document since normally we are only dealing with standard ASCII characters.)

Modify the query to retrieve all the elements and hit **Execute**.

```
SELECT *
FROM [Fleet Generation].[Asset].[ElementHierarchy]
WHERE
  Path like N'\%'
```

OR

```
SELECT *
FROM [Fleet Generation].[Asset].[ElementHierarchy]
```

Returns the same results.

Results | Messages

	Path	Name	Level	ElementID	ParentElementID
1	\	CENTRAL	0	7805ed21-263d-45fb-a0bb-d55fea4c2970	
2	\	NORTH	0	a43abd6d-df40-4005-bc7f-2a958e73034d	
3	\	SOUTHEAST	0	e4ba7143-8d32-454b-8b02-e524d1498951	
4	\CENTRAL\	Carbondale	1	8a192c08-c440-4e55-9c8e-ba9afd445ab6	7805ed21-263d-45fb-a0bb-d55fea4c2970
5	\CENTRAL\	Beryl Ridge	1	c8572d05-16f0-49d0-a4aa-6235fe70c4d1	7805ed21-263d-45fb-a0bb-d55fea4c2970
6	\CENTRAL\	Albertsville	1	77d9cf83-3dbd-4b39-99ef-55cbfbcb010c	7805ed21-263d-45fb-a0bb-d55fea4c2970
7	\NORTH\	New Bedford	1	549ddf1f-c857-4a98-89b0-254c1b8016e7	a43abd6d-df40-4005-bc7f-2a958e73034d
8	\NORTH\	Madison	1	067c7347-9301-46e9-8dec-110e534326b3	a43abd6d-df40-4005-bc7f-2a958e73034d
9	\NORTH\	Greenlawn	1	c8f6703c-6cdf-4f94-b39f-b58298a4961d	a43abd6d-df40-4005-bc7f-2a958e73034d
10	\NORTH\	Ebbitt	1	35d3ca11-9a51-4c7a-bed5-127d6c6dae3c	a43abd6d-df40-4005-bc7f-2a958e73034d
11	\SOUTHEAST\	Wolverine Station	1	114733ee-8872-45c0-84ba-3d94376426ea	e4ba7143-8d32-454b-8b02-e524d1498951

10.3 Table Aliases

Sometime table name or columns are lengthy or lack clarity. Using an **ALIAS** can simplify typing and clarify table field names that are otherwise unclear. The "**AS**" command defines an **ALIAS** for the item prior to the **AS** with the abbreviation following the command.

```
SELECT eh.* FROM [Fleet Generation].[Asset].[ElementHierarchy] as eh
```

In the above statement, **eh** can be used to identify the table instead of the full [Fleet Generation].[Asset].[ElementHierarchy] table name. Aliases become more significant when creating joins.

10.4 JOIN Statements

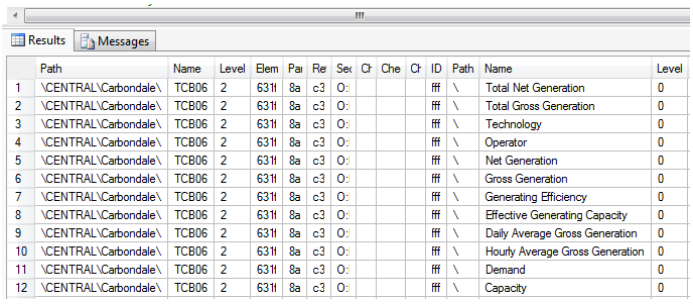
Rarely does data exist in one place or in one table. Sometimes the results of a query have to come from a correlation of two or more distinct tables. To JOIN tables, a relationship is required between the tables and must be identified in the SQL statement.

Within the joining operations, we want a result set than contains assets with useful information from both tables, like performing a logical AND operation. There should be no gaps where a match could not be found. This is called an INNER JOIN, and is the default joining operation used by PI SQL Commander. Therefore, INNER JOIN and JOIN may be used interchangeably. If we want to include these empty values, then an OUTER JOIN can be used, the equivalent to the logical OR.

Two key words are used when creating joins between tables. **The words JOIN and ON can be used in the statement to identify the relationship between the tables being used.** The key word ON sets up the relationship of columns in the selected tables so the desired rows are returned.

```
SELECT *
FROM [Fleet Generation].[Asset].[ElementHierarchy] as EH JOIN [Fleet
Generation].[Asset].[ElementAttribute] as EA
ON eh.name = ea.name
```

Returns no records. Below is a result of the next query, but shows the names in the tables are not the same.



Results | Messages

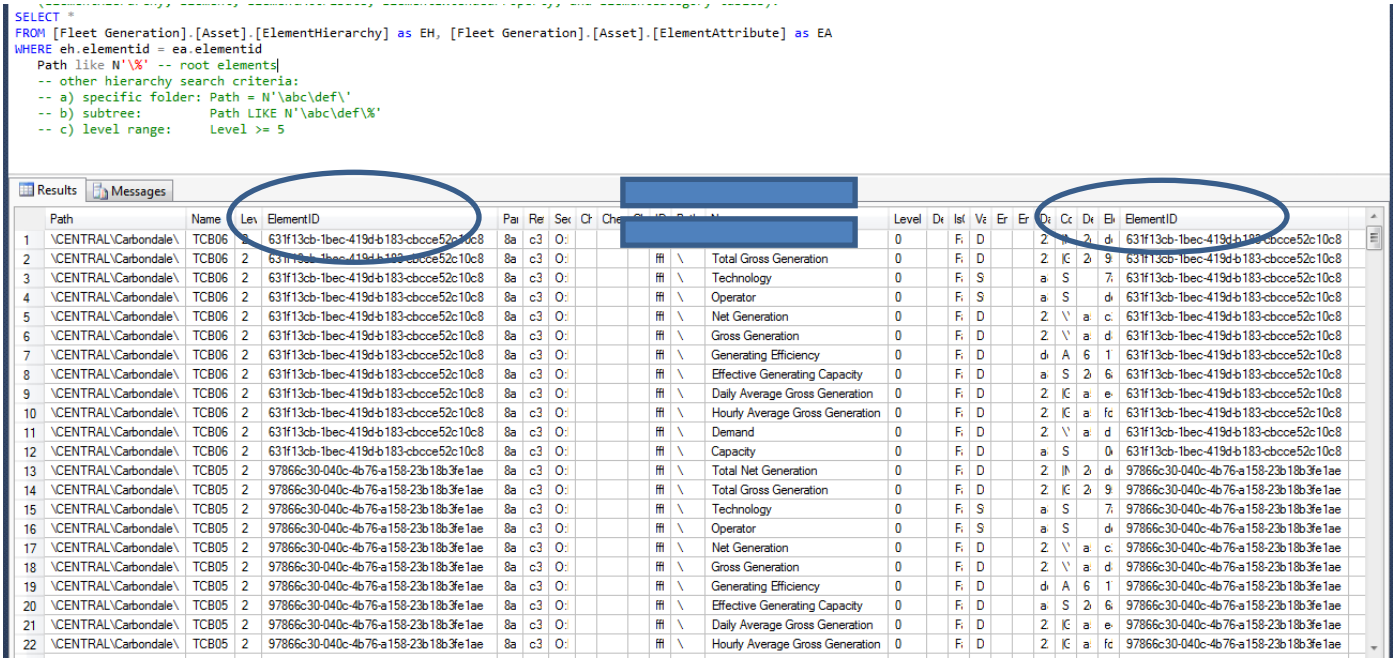
	Path	Name	Level	Elem	Pai	Re	Sec	Ch	Che	Ch	ID	Path	Name	Level
1	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Total Net Generation	0
2	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Total Gross Generation	0
3	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Technology	0
4	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Operator	0
5	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Net Generation	0
6	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Gross Generation	0
7	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Generating Efficiency	0
8	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Effective Generating Capacity	0
9	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Daily Average Gross Generation	0
10	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Hourly Average Gross Generation	0
11	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Demand	0
12	\CENTRAL\Carbondale\	TCB06	2	631f	8a	c3	O:				fff	\	Capacity	0

Even though both tables have columns called names, they do not identify identical fields.

Note: Columns named the same are not necessarily referring to the same item. For example, ‘id’ is a column that is frequently found in tables representing a unique identifier for the row, but rarely do they refer to the same item from table to table.

However, the columns named ElementID in both tables are actually the same and return a listing of all attributes for all elements defined.

```
SELECT *
FROM [Fleet Generation].[Asset].[ElementHierarchy] AS EH join [Fleet
Generation].[Asset].[ElementAttribute] AS EA
on eh.elementid = ea.elementid
```



```
SELECT *
FROM [Fleet Generation].[Asset].[ElementHierarchy] as EH, [Fleet Generation].[Asset].[ElementAttribute] as EA
WHERE eh.elementid = ea.elementid
   Path like N'\%' -- root elements
   -- other hierarchy search criteria:
   -- a) specific folder: Path = N'\abc\def\'
   -- b) subtree:         Path LIKE N'\abc\def\%'
   -- c) level range:     Level >= 5
```

Results | Messages

	Path	Name	Lev	ElementID	Level	ElementID
1	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
2	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
3	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
4	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
5	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
6	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
7	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
8	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
9	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
10	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
11	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
12	\CENTRAL\Carbondale\	TCB06	2	631f13cb-1bec-419d-b183-cbcce52c10c8	0	631f13cb-1bec-419d-b183-cbcce52c10c8
13	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae
14	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae
15	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae
16	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae
17	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae
18	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae
19	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae
20	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae
21	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae
22	\CENTRAL\Carbondale\	TCB05	2	97866c30-040c-4b76-a158-23b18b3fe1ae	0	97866c30-040c-4b76-a158-23b18b3fe1ae

10.4.1 Field Aliases:

There's an unsightly problem with the query: multiple columns are named the same (*Name*) but are not the same. For anyone reading these query results, this is not helpful.

The solution is to rename the columns. Just as a table can be aliased, so can a column be aliased. The keyword **AS** is used anytime an **ALIAS** is defined, whether the field is a table or column name.

SELECT eh.Name AS [Element Name], ea.Name [Attribute Name]

The above statement gives meaningful names to the columns in the respective tables.

10.4.2 Directed Activity – Element descriptions

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

To extract the elements in the Fleet Generation database along with their descriptions.

Approach:

- Execute the **Predefined Query** associated with the **Element Hierarchy** – modify to obtain all elements.
- Review the fields in the Element table.
- Determine potential relationships between the tables to create joins.
- Modify the Element Hierarchy query to add the appropriate join information to extract the description of the elements.

Locate the Element Hierarchy within SQL.

- Execute the Predefined Query.

```
SELECT *
FROM [Fleet Generation].[Asset].[ElementHierarchy]
WHERE
  Path = N'\'
```

- Modify the query to obtain all elements

```
SELECT *
FROM [Fleet Generation].[Asset].[ElementHierarchy]
```

Review the ElementHierarchy table, which gives us the Path and Name (among other things) of the elements in the hierarchy, but no description:

<AF Database>.Asset.ElementHierarchy

Name	Data type	Is nullable?
Path	String	No
Name	String	No
Level	Int16	No
ElementID	Guid	No
ParentElementID	Guid	Yes

Review the Element table. Is there a link between the Element and Element Hierarchy table?

<AF Database>.Asset.Element

Name	Data type	Is nullable?
ID	Guid	No
Name	String	No
Description	String	Yes
Comment	String	Yes
Revision	Int32	No

Each element pointer within the hierarchy (i.e. each row in the ElementHierarchy table) corresponds to an element object from the overall set of Elements (i.e. a row in the Element table). Behind the scenes, these objects are linked by GUIDs (Globally Unique Identifiers). The purpose of a GUID is to give individual objects an identifier guaranteed to be unique. Meaningless to the human eye, they look like:

9abd6084-6c74-4645-a7a0-833f6c25de3d

GUIDs (often contained in table columns ending in *ID*) are how each table relates each row (element pointer) in *ElementHierarchy* to each row (element) in *Element*:

<AF Database>.Asset.ElementHierarchy

Name	Data type	Is nullable?
Path	String	No
Name	String	No
Level	Int16	No
ElementID	Guid	No
ParentElementID	Guid	Yes
ReferenceTypeID	Guid	No
SecurityDescriptor	String	No
CheckOutTime	DateTime	Yes
CheckOutUserName	String	Yes
CheckOutMachineName	String	Yes

<AF Database>.Asset.Element

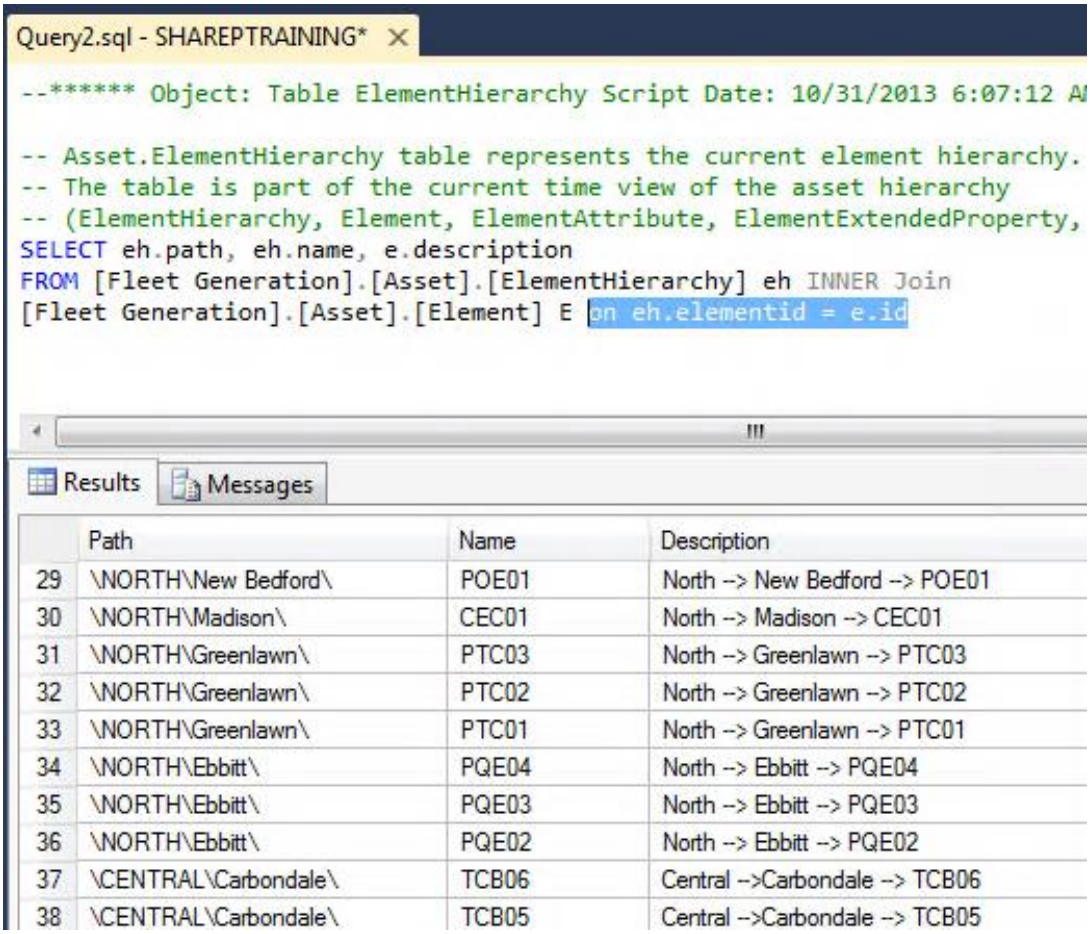
Name	Data type	Is nullable?
ID	Guid	No
Name	String	No
Description	String	Yes
Comment	String	Yes
Revision	Int32	No
HasChildren	Boolean	No
HasMultipleVersions	Boolean	No
ElementTemplateID	Guid	Yes
DBReferenceTypeID	Guid	Yes

- Modify the Element Hierarchy table to include the description from the Element table.

```
SELECT eh.path, eh.name, e.description
FROM [Fleet Generation].[Asset].[ElementHierarchy] eh INNER Join [Fleet
Generation].[Asset].[Element] E on eh.elementid = e.id
```

Note: In the above statement, the tables have ALIASes, but the word “AS” is not in the statement as it is understood.

The result of the above query yields the name of the element and the description associated with the element.



Query2.sql - SHAREPTRAINING*

```
--****** Object: Table ElementHierarchy Script Date: 10/31/2013 6:07:12 A
-- Asset.ElementHierarchy table represents the current element hierarchy.
-- The table is part of the current time view of the asset hierarchy
-- (ElementHierarchy, Element, ElementAttribute, ElementExtendedProperty,
SELECT eh.path, eh.name, e.description
FROM [Fleet Generation].[Asset].[ElementHierarchy] eh INNER Join
[Fleet Generation].[Asset].[Element] E on eh.elementid = e.id
```

Results | Messages

	Path	Name	Description
29	\NORTH\New Bedford\	POE01	North --> New Bedford --> POE01
30	\NORTH\Madison\	CEC01	North --> Madison --> CEC01
31	\NORTH\Greenlawn\	PTC03	North --> Greenlawn --> PTC03
32	\NORTH\Greenlawn\	PTC02	North --> Greenlawn --> PTC02
33	\NORTH\Greenlawn\	PTC01	North --> Greenlawn --> PTC01
34	\NORTH\Ebbitt\	PQE04	North --> Ebbitt --> PQE04
35	\NORTH\Ebbitt\	PQE03	North --> Ebbitt --> PQE03
36	\NORTH\Ebbitt\	PQE02	North --> Ebbitt --> PQE02
37	\CENTRAL\Carbondale\	TCB06	Central -->Carbondale --> TCB06
38	\CENTRAL\Carbondale\	TCB05	Central -->Carbondale --> TCB05

10.4.3 Exercise - Query for Specific Elements

This solo or group activity is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the activity.

Objective:

To extract the elements in the Fleet Generation database that are “Units” (Element Template) and are located in the North Region. The fields that we want in our result set are the Unit Name, Path, and Description.

Approach:

- Start with the query from the previous Directed Activity

```
SELECT eh.path, eh.name, e.description
FROM [Fleet Generation].[Asset].[ElementHierarchy] eh INNER Join [Fleet
Generation].[Asset].[Element] E on eh.elementid = e.id
```

- Review the fields in the Element Templates table.
- Determine potential relationships between the tables to create joins.
    - Hint: The Element table has a field called ElementTemplateID
- Append a **WHERE** clause to filter based on the Path and Element Template.
- Determine the fields to return and the tables associated to each field.

10.5 Built-in Functions

PI SQL Commander has some built-in functions specific to the PI System. If you are familiar with SQL, you may already be familiar with functions. For example, aggregation functions such as Max() or Avg() return the maximum or average of a group of rows.

An entire list of built-in functions is available in the user guide for PI OLEDB Enterprise.

One of the PI functions that will be used in subsequent exercises is ParentName(). Instead of returning the complete PATH. The ***ParentName*** function of PI OLEDB Enterprise is used to break up the AF element path name into separate columns of the table. The strings in double quotes are used to rename the column name in the table to something perhaps better suited for reporting. Again, the "eh" **ALIAS** prefix is required to identify the source of the field.

```
SELECT
eh.Name [Unit]
, ParentName(eh.Path,0) [Station]
, ParentName(eh.Path,1) [Region]
FROM [Fleet Generation].[Asset].[ElementHierarchy] eh
Where eh.Level=2
```

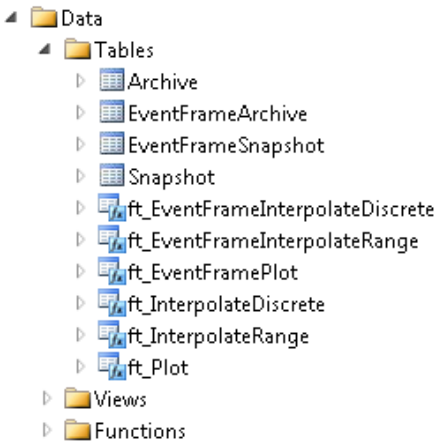
Results | Messages

	Unit	Station	Region
1	GA002	Albertsville	CENTRAL
2	GA001	Albertsville	CENTRAL
3	BCU02	Beryl Ridge	CENTRAL
4	BCU01	Beryl Ridge	CENTRAL
5	TCB06	Carbondale	CENTRAL
6	TCB05	Carbondale	CENTRAL
7	TCB04	Carbondale	CENTRAL

10.6 Data Tables

In the previous sections, we saw the process to query for elements from the Fleet Generation database through a series of table joins between Asset Framework object tables within PI SQL Commander. The tables within PI SQL Commander are not limited solely to Elements, Element Hierarchy, and Element Templates.

Within PI SQL Commander, there are several tables under the [AF Database].[Data] path that will allow the user to extract real-time and archive values from the PI Data Archive. A query against these tables will return either Element Attribute data or Event Frame data. In order to utilize these tables, a query needs to have an INNER JOIN to the ElementAttribute table and a specific Data table. The ElementAttribute table allows for the mapping between the data and specific attributes associated with a set of elements.



The tables corresponding to Element Attribute data are listed below:

Table	Description
Archive	Returns archive / compressed data
Snapshot	Returns values in the snapshot (current values)
ft_InterpolateDiscrete	Returns interpolated value given timestamp
ft_InterpolateRange	Returns interpolated values at evenly distributed timestamps
ft_Plot	Returns minimum data required for trending

Note: Similar tables exist for data from Event Frame attributes. Typically, only the Event Frame Snapshot table is queried against as each Event Frames contain individual start and end times.

10.6.1 Directed Activity – Snapshot Values

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective: Create a query to extract real-time values for the Gross Generation and Net Generation attributes of all units. The fields in the result set should contain Element Name, Station Name, Region Name, Attribute Name, Timestamp, and Value.

Approach:

The creation of this query requires several steps.

1) Run the **predefined query** of the Element table to obtain **all** elements.

2) Remove the **WHERE** clause.

```
SELECT *
FROM [Fleet Generation].[Asset].[Element] e
```

3) Apply an **INNER JOIN** to the **Element** table.

```
SELECT *
FROM [Fleet Generation].[Asset].[Element] e
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
ON e.ID = eh.ElementID
```

4) Apply an **INNER JOIN** to the **ElementTemplate** table.

```
SELECT *
FROM [Fleet Generation].[Asset].[Element] e
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
ON e.ID = eh.ElementID
INNER JOIN [Fleet Generation].[Asset].[ElementTemplate] et
ON e.ElementTemplateID = et.ID
```

5) Add a **WHERE** clause to return only elements that are **Units**.

```
SELECT *
FROM [Fleet Generation].[Asset].[Element] e
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
ON e.ID = eh.ElementID
```

```
INNER JOIN [Fleet Generation].[Asset].[ElementTemplate] et
ON e.ElementTemplateID = et.ID
WHERE et.Name = 'Unit'
```

6) Apply an **INNER JOIN** to the **ElementAttribute** table and modify the **WHERE** clause to return only attributes that are either "**Gross Generation**" or "Net **Generation**."

```
SELECT *
FROM [Fleet Generation].[Asset].[Element] e
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
ON e.ID = eh.ElementID
INNER JOIN [Fleet Generation].[Asset].[ElementTemplate] et
ON e.ElementTemplateID = et.ID
INNER JOIN [Fleet Generation].[Asset].[ElementAttribute] ea
ON ea.ElementID = e.ID
WHERE et.Name = 'Unit' and (ea.Name = 'Gross Generation'
OR ea.Name = 'Net Generation')
```

7) Apply an **INNER JOIN** to the **Data Snapshot** table and specify the fields for the result set.

```
SELECT
eh.Name [Unit]
, ParentName(eh.Path,0) [Station]
, ParentName(eh.Path,1) [Region]
, ea.Name [Attribute]
, s.Time
, s.Value
FROM [Fleet Generation].[Asset].[Element] e
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
ON e.ID = eh.ElementID
INNER JOIN [Fleet Generation].[Asset].[ElementTemplate] et
ON e.ElementTemplateID = et.ID
INNER JOIN [Fleet Generation].[Asset].[ElementAttribute] ea
ON ea.ElementID = e.ID
INNER JOIN [Fleet Generation].[Data].[Snapshot] s
ON s.ElementAttributeID = ea.ID
WHERE et.Name = 'Unit' and (ea.Name = 'Gross Generation'
OR ea.Name = 'Net Generation')
```

10.6.2 Exercise – Interpolated data

This solo or group exercise is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the exercise.

Objective:

Create a query to extract hourly interpolated data for the Demand attribute of all UNITs over the past four hours. The fields for the result set should include Element Name, Attribute Name, Timestamp, and Value.

At what time does the demand tend to be high over all units?

Approach:

The creation of this query requires several steps.

1) Run the predefined query of the **ft_InterpolateRange** table.

2) Remove the portion of the WHERE clause that filters the results to all root elements.

3) Modify the **Start Time, End Time** and **TimeStep**.

4) Apply an **INNER JOIN** to the **Element** table, which is required to join to the **ElementTemplate** table

5) Apply an **INNER JOIN** to the **ElementTemplate** table, which is required to filter for the **UNIT** template (et.Name)

6) Add a WHERE clause to return only elements that are Units.

7) Append to WHERE clause to return only the "**Demand**" attribute.

8) Restrict the SELECT to return the desired fields.

10.7 Data Transpose Functions & Function Tables

As seen above, the data comes back in tabular form, but does not lend itself to easy interpretation.

Below is the query from the previous section that illustrates the difficulty in reviewing the snapshot data for the attributes for the Elements. Notice that the attributes as returned in rows.

```
SELECT eh.name, ea.name, s.time, s.value
FROM [Fleet Generation].[Asset].[ElementHierarchy] eh
INNER JOIN [Fleet Generation].[Asset].[ElementAttribute] ea
ON ea.ElementID = eh.ElementID
INNER JOIN [Fleet Generation].[Data].[Snapshot] s
ON s.ElementAttributeID = ea.ID
OPTION (FORCE ORDER, EMBED ERRORS)
```

A portion of the results from the above query is displayed below.

Results | Messages

	Name	Name	Time	Value
1	ALX01	Yield	2013-10-31 06:34:42.000	72.82847
2	ALX01	Unit Status	2013-10-31 06:09:42.000	Active
3	ALX01	Technology	1970-01-01 00:00:00.000	Cogeneration
4	ALX01	Shift Hours	1970-01-01 00:00:00.000	12
5	ALX01	Shift	2013-10-31 06:00:11.000	1
6	ALX01	Operator	1970-01-01 00:00:00.000	COG
7	ALX01	Net Generation	2013-10-31 06:34:42.000	140386.2
8	ALX01	Hourly Capacity	1970-01-01 00:00:00.000	8500
9	ALX01	Gross Generation	2013-10-31 06:34:42.000	148570.1
10	ALX01	Generating Efficiency	2013-10-31 06:34:42.000	94.49159

Data presented in this format is typically difficult to handle for reports and BI Tools. Instead, we prefer to ***rotate or transpose*** the values as:

Element	El	Tc	Tc	Technology	Operator	Net Generation	Gross Generation	Generating Effici
\NORTH\Ebbitt\PQE03	a	9	7	Geothermal	BSX	101840.5	80625.76	80.17107
\NORTH\Ebbitt\PQE02	8	5	4	Geothermal	BSX	670097.6	672455.1	98.72289
\SOUTHEAST\Wolverine Station\ALX01	7	1	1	Cogeneration	COG	135907.7	129976.7	99.78031
\SOUTHEAST\Vicksberg\MAM04	8	1	1	Coal	BSX	17412.14	18153.54	92.0494
\SOUTHEAST\Vicksberg\MAM03	al	3	3	Coal	BSX	362175.5	374015.1	99.71217
\SOUTHEAST\Vicksberg\MAM02	ff	1	1	Coal	BSX	143047.1	145616.9	98.60997
\SOUTHEAST\Vicksberg\MAM01	e	1	1	Coal	BSX	138275	138294.6	99.29276

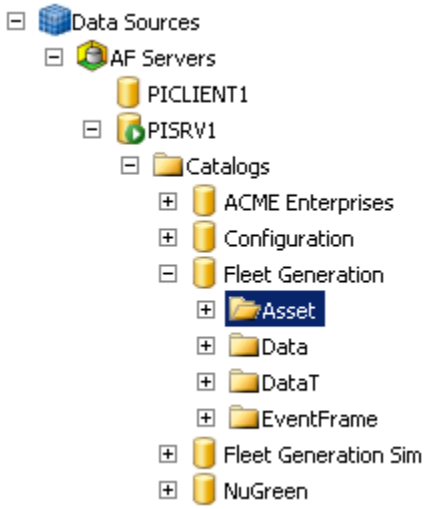
As the same column headers are used for every row in the table, every element needs to have the same set of attributes in order to populate these cells. This brings us back to using **AF Templates** for all our elements, so we can get these common properties. Transpose functions are only available for templatized elements!

This is similar to that of the final view created by the PI Integrator for Business Analytics. PI OLEDB Enterprise can also generate transpositions similar to what is above. A wizard walks you through the process of creating a transpose function for any Element Template of your choosing.

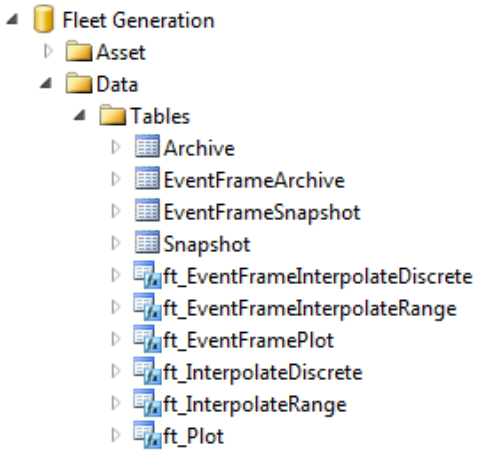
10.7.1 Transpose Function Wizard

For many use cases, attribute values need to be returned in a way so that each column represents an attribute. This is contrary to a typical relational representation, where each value of each attribute is normally represented in consecutive rows. To represent multiple attributes in this "one column per attribute" format, one could join data tables with itself multiple times, but the resulting query string would be rather large and complex. To help with this, we provide a way to create custom Table-Valued Functions (TVFs) and derived function tables, to get "transposed" result sets of the related data tables.

Under each PI AF database branch, there are four folders, “Assets”, “Data”, “DataT” and “EventFrame”.



The “Tables” folder under “Data” shows the tables and columns that provide access to snapshot and historical data from the PI System.

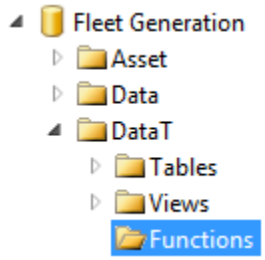


Under both the “Assets” and “Data” folders there are two additional folders called “Views” and “Functions”. These folders are initially empty and provide places for you to organize the views and functions you create.

In general, creating and editing queries and views is a restricted activity. The changes are contained in the PI AF SQL Database and access will be controlled by the database administrator.

The “DataT” branch of the hierarchy is for working with transpose functions.

- Transpose functions allow you to obtain tables of PI AF information based on AF element templates.
- This folder comes with the same subfolder structure as “Assets” and “Data”, but they are initially empty until they are **manually created**.
- Transpose functions can be create using the wizard discussed in the next section.



10.7.2 Directed Activity – Create Dynamic Asset Transpose Functions

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

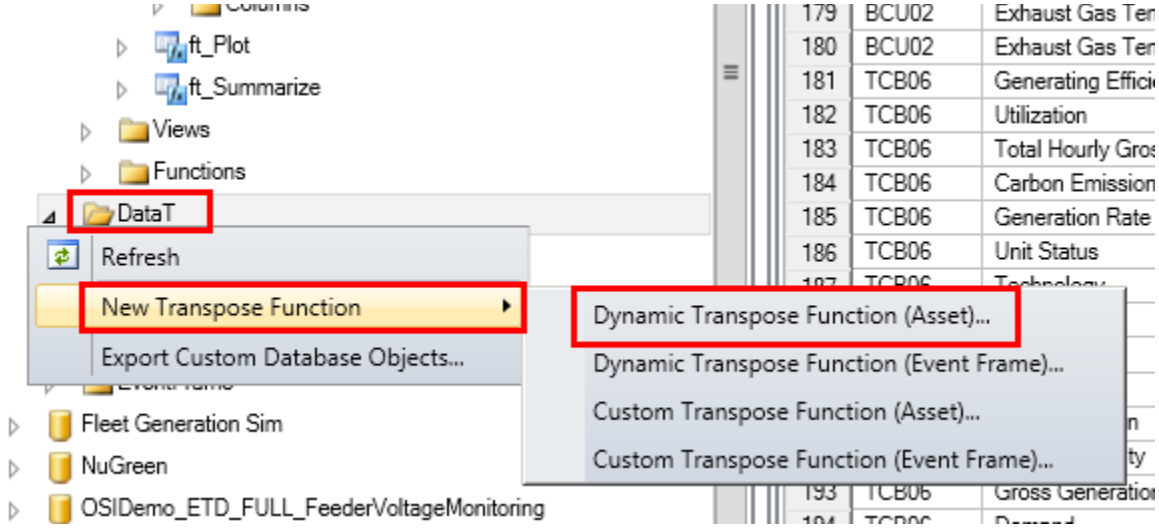
Objective: Create all Dynamic Asset Transpose Functions for the Fleet Generation database to be used in analyzing plant generation data.

Approach:

There are four transpose functions available. Each transpose function returns a dataset made up of columns for every attribute of an element template, where each row returns values based on a different time basis.

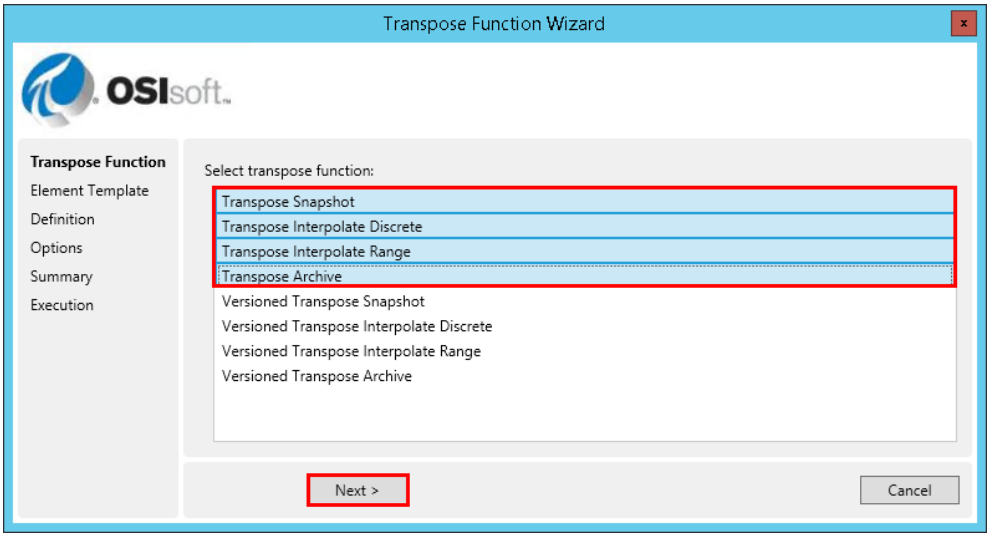
Function	Snapshot or Archive	Rows/element returned
Transpose Snapshot	Snapshot	1 row per element attribute for element(s) selected
Transpose Archive	Archive	1 row per element archived attribute value for element(s) selected over a specific time range
Transpose Interpolate Discrete	Interpolated	1 row per element returns interpolated value based on timestamp
Transpose Interpolate Range	Interpolated	1 row per element returned for each interval for each element attribute based on time range and interval

Access the Transpose Function Wizard by right-clicking on the “DataT” folder under the AF Database catalog you wish to build a transpose function for, and select “New Transpose Function -> Dynamic Transpose Function (Asset)…”



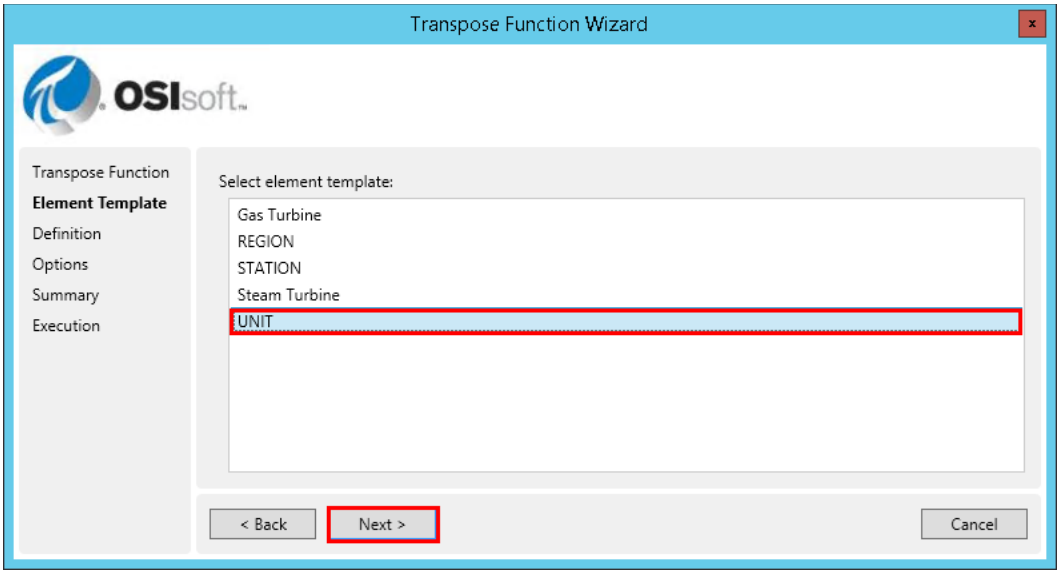
Note: Dynamic Transpose Functions will automatically reflect additions and deletions of Attributes from the template, whereas Custom Transpose Functions allow you to explicitly select a set of Attributes which will remain static.

We will create all of them, hold control and multi-select the top four then click Next:

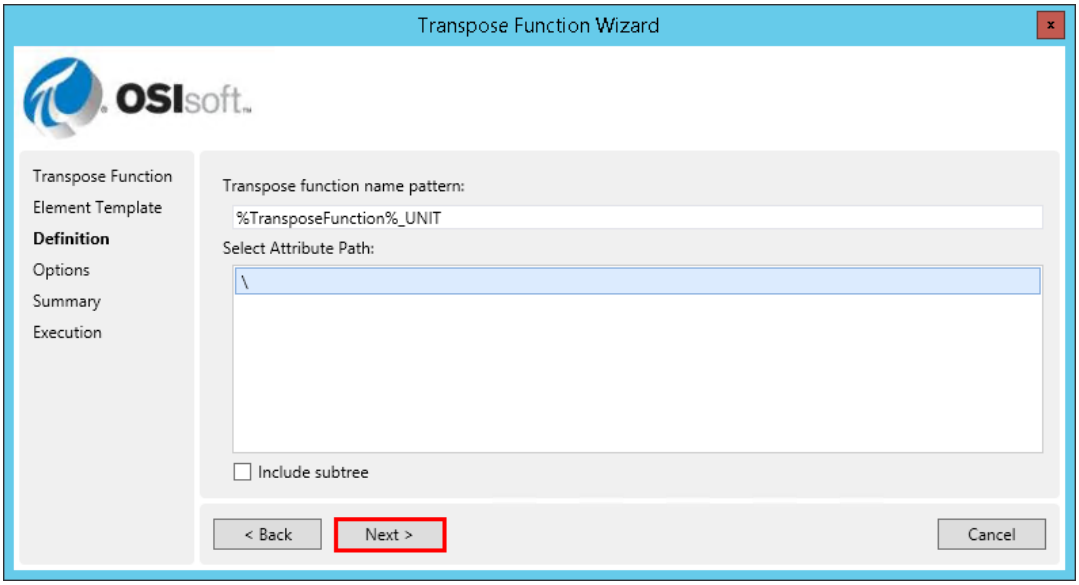


Note: Versioned Transpose functions consider PI AF versioning. We will not be using these.

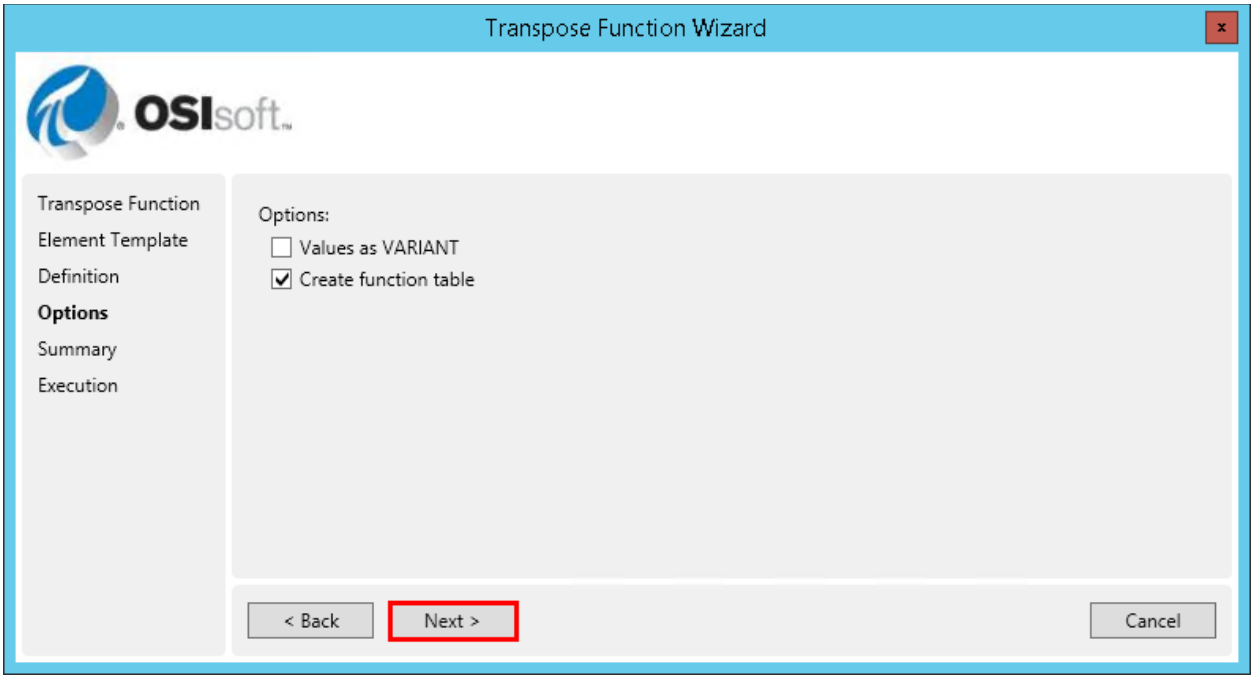
Select the PI AF template(s) you want to create transpose function for. You can select as many as you want, the wizard can build multiple transpose functions per pass. Here, we will select "UNIT" and click **Next**.



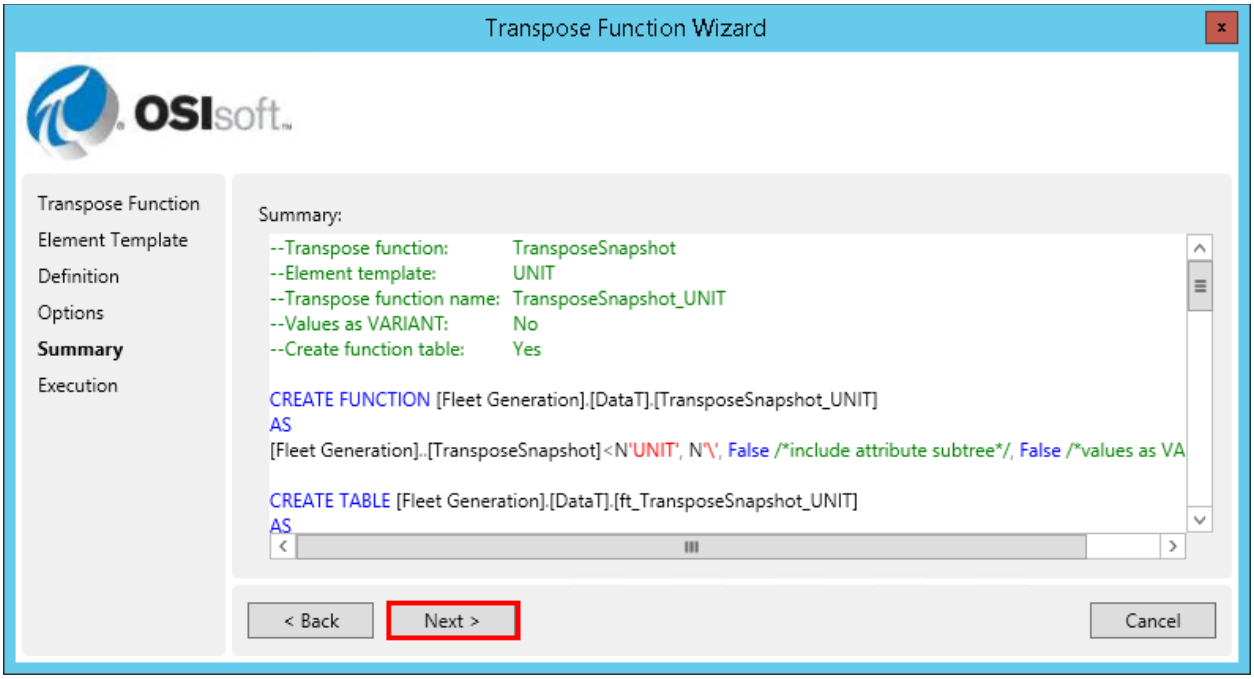
If you want to use non-default Transpose Function names or the element template has sub-attributes, modifications can be made here. In our case just leave the defaults and click **Next**.



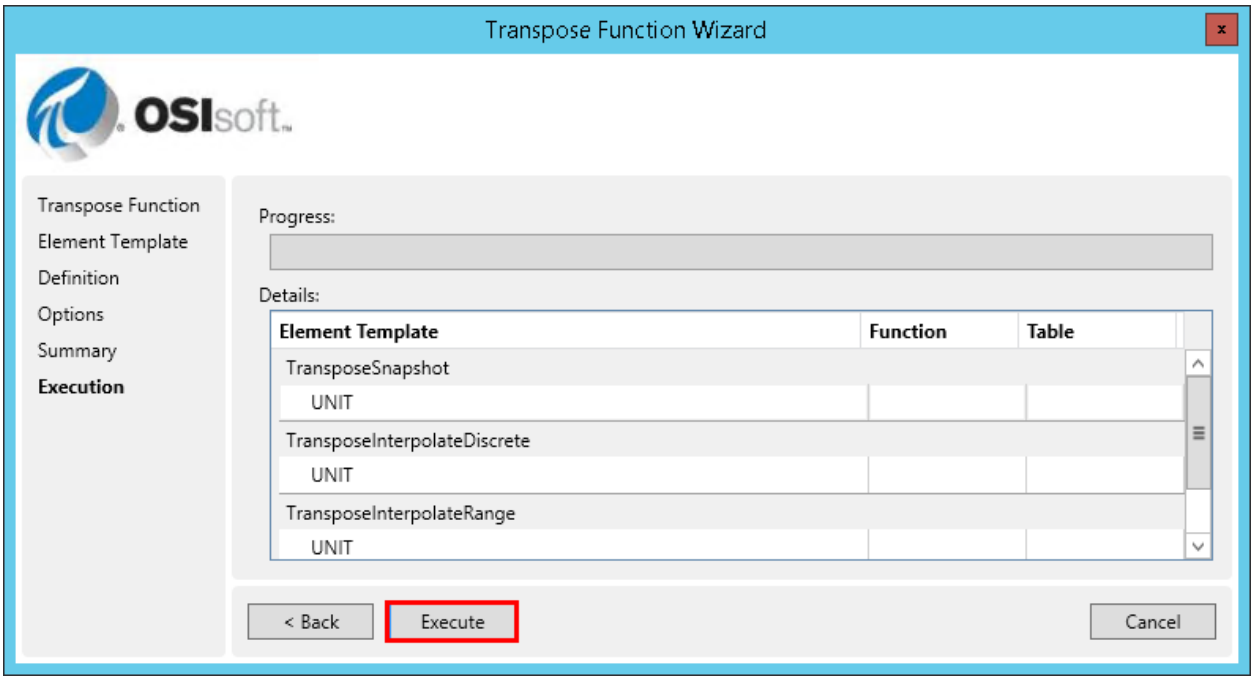
Leave the defaults for the next window. You have the option to return values using the VARIANT data type, which may be useful with some reporting tools. We want to create the function table.



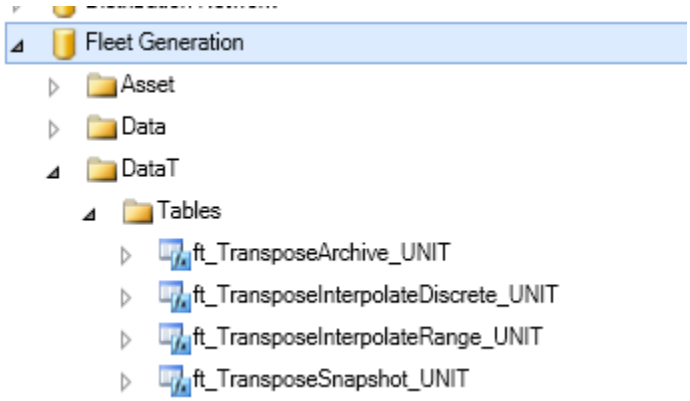
Review the summary and click **Next**.



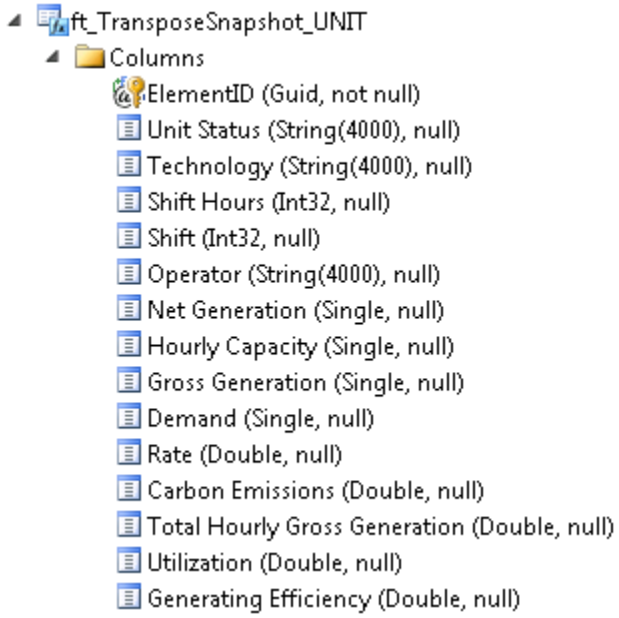
Click **Execute.** Transpose function creation should be successful.



From the PI SQL Commander hierarchy, the transpose tables and functions created by the wizard should appear under the “DataT” folder of the “Fleet Generation” PI AF database.

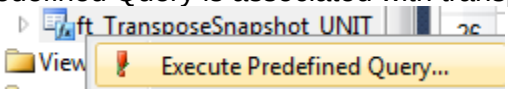


As for how to use your newly-created Function, examine the snapshot function it in *Object Explorer*:



Reality check: If we call the TransposeSnapshot_UNIT function, the same columns exist in the function as in the original template, such as Effective Generating Capacity, Generating Efficiency, etc.

A Predefined Query is associated with transpose functions. Execute the query.



Results | Messages

	Element	ElementID	Unit Status	Technology	Shift I	Shift	Operator	Net Generation	Hourly Ca	Gross Gener
1	\CENTRAL\Albertsville\GA002	e7bfa520-e806-1	Active	Natural Gas	8	2	BSX	0	650	30
2	\CENTRAL\Albertsville\GA001	e7bfa51d-e806-1	Active	Natural Gas	8	2	BSX	307.2737	550	338.0011
3	\CENTRAL\Beryl Ridge\BCU02	e7bfa529-e806-1	Active	Natural Gas	8	2	BSX	493.1268	550	542.4395
4	\CENTRAL\Beryl Ridge\BCU01	e7bfa526-e806-1	Inactive	Natural Gas	8	2	BSX	225.9575	600	255.9575
5	\CENTRAL\Carbondale\TCB06	e7bfa53e-e806-1	Inactive	Natural Gas	8	2	BSX	83.21286	600	113.2129
6	\CENTRAL\Carbondale\TCB05	e7bfa53b-e806-1	Inactive	Natural Gas	8	2	BSX	600	650	660

10.7.3 Exercise – Create Dynamic Event Frame Transpose Functions

This solo or group activity is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the activity.

Objective:

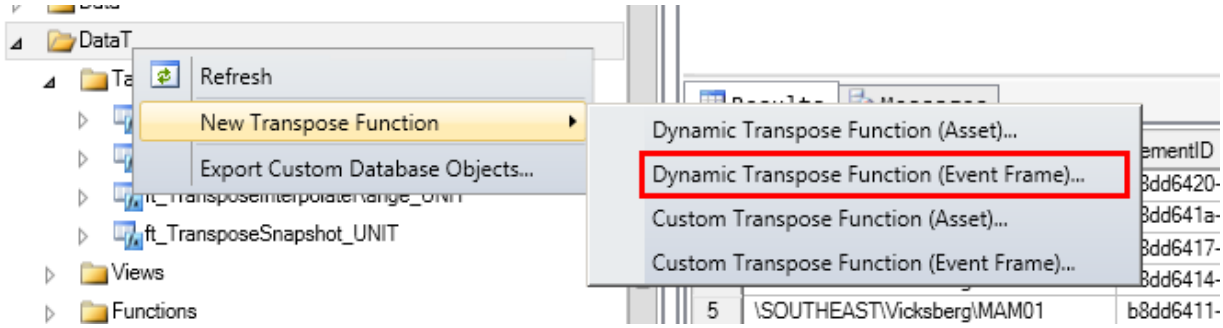
Attributes from the generation units and the event frames will be used to analyze production data from the plants.

Approach:

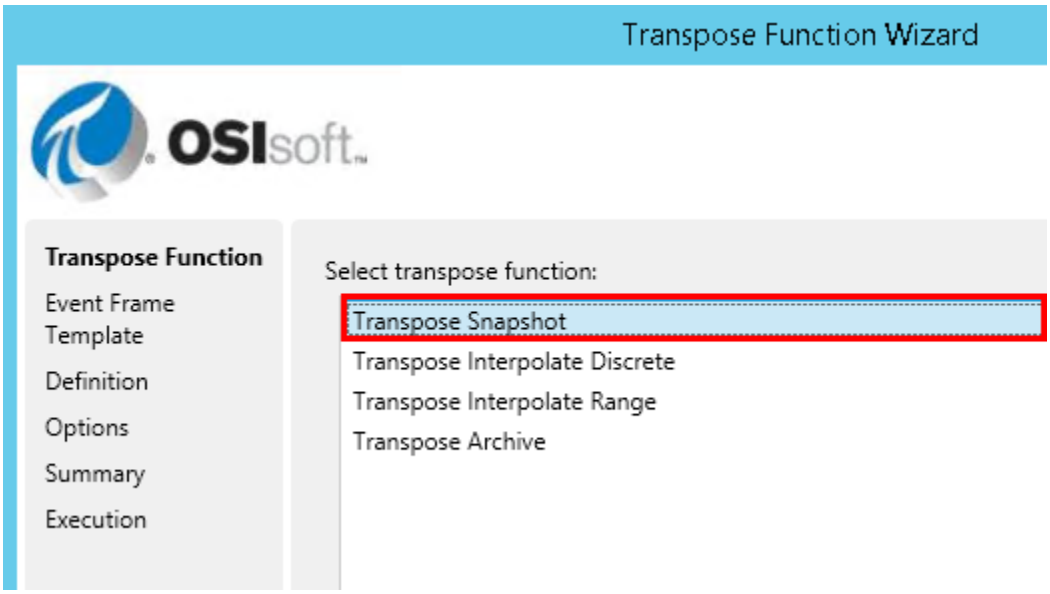
Use the **transpose function** wizard to create a **Snapshot event frame function** using the Inactivity and Gas Turbine Temperature Anomaly template.

Verify the results of the transpose function through the execution of the pre-defined query.

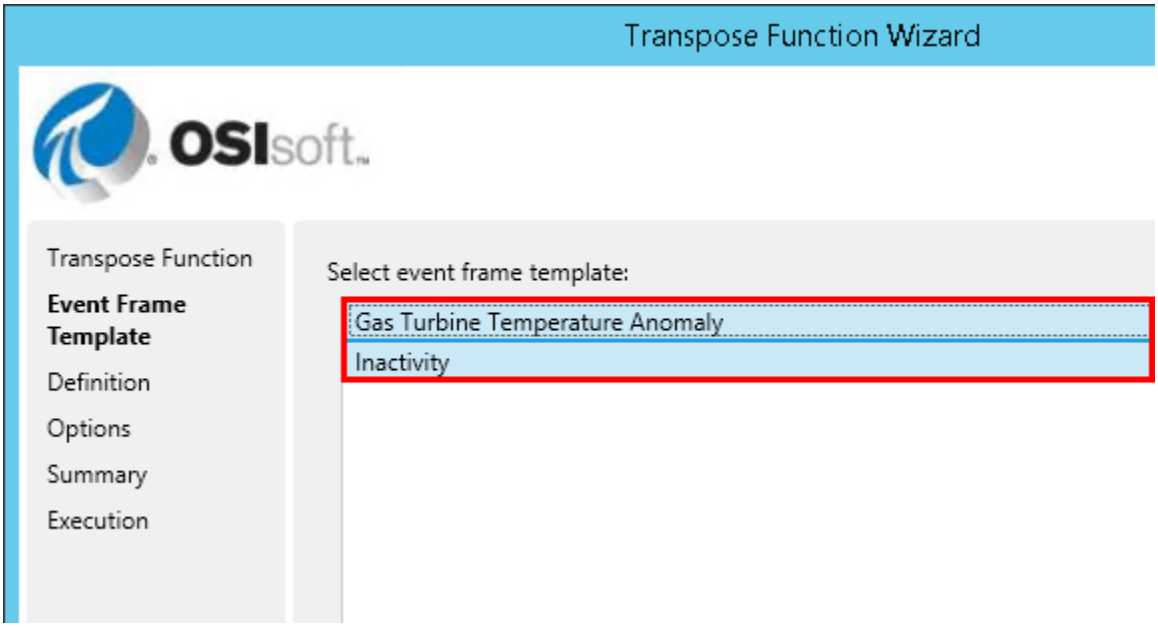
Hint: The steps are almost identical to the ones used when creating an Asset transpose function.



Create **Transpose Snapshot** only:



Create for both **Gas Turbine Temperature Anomaly** and **Inactivity**.



Leave **all defaults** from this point on. Next, Next, Next, Next, Execute.

10.8 UNION Statements

You may have noticed that the Asset Transpose functions we created only return Assets that use the UNIT template and not those that use the Gas Turbine template, despite the Gas Turbine template being derived from the UNIT template via template inheritance. Unfortunately, this is a limitation of PI OLEDB Enterprise. One way to address this is with UNIONs.

In simple terms, UNIONs take the results of two queries and stack the result sets on top of each other to form a single result set. One limitation of UNIONs is that the input result sets must have identical columns, which may require removing and aliasing columns to match the data sets. This will be demonstrated in the following exercise.

The syntax is quite simple, place the keyword UNION in between two queries to union them together. The OPTION statement must be at the very end:

```
SELECT * FROM Table1 WHERE Condition='TRUE'
UNION
SELECT * FROM Table2 WHERE Condition='TRUE'
OPTION (FORCE ORDER, IGNORE ERRORS, EMBED ERRORS)
```

10.8.1 Directed Activity – UNITs and Gas Turbines in a single result set

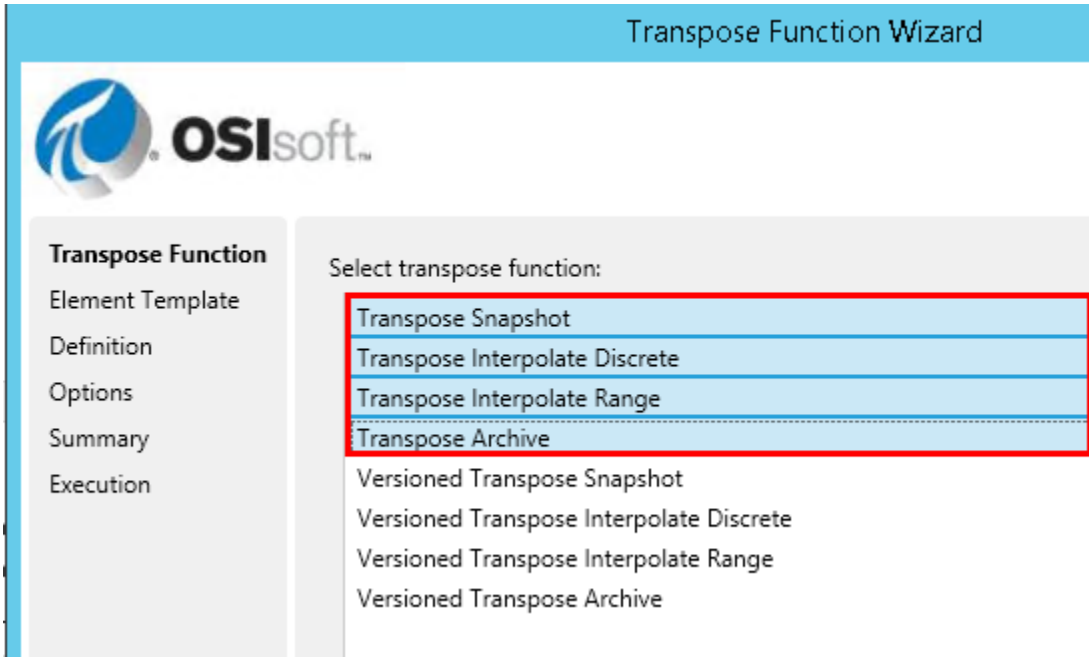
In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective: Create a query to display Snapshots for all UNITs, including Gas Turbines.

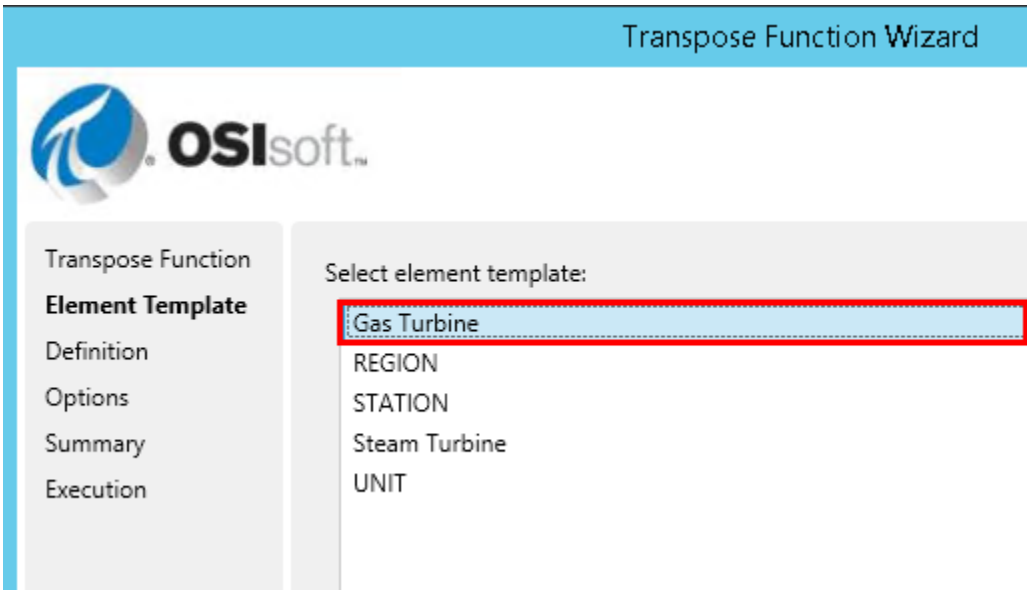
Approach:

The creation of this query requires several steps.

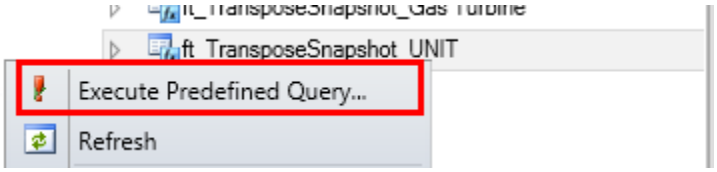
Create **Dynamic** Asset Transpose Functions for **Gas Turbines**. This is identical to a previous exercise except we will select the Gas Turbine Template. Create **all** non-versioned Transpose Functions



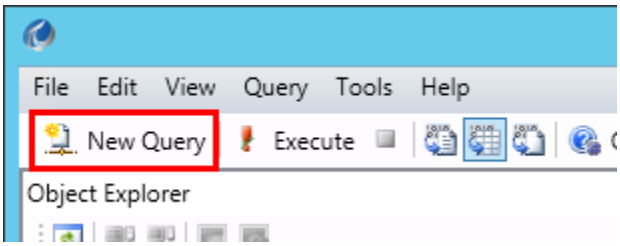
Select Gas Turbine, then **Next, Next, Next, Next, Execute** (leave all defaults after this point)



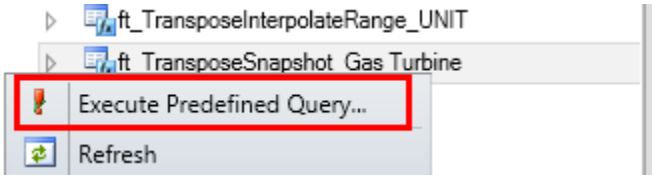
Execute the predefined query for **ft_TransposeSnapshot_UNIT**, make note of the resulting column set.



Create a new query. **The Goal here is to preserve the text of the UNIT query.**



Execute the predefined query for **ft_TransposeSnapshot_Gas Turbine**, note that there are five additional columns for those attributes unique to Gas Turbines.



Change **ts.*** to explicitly select only those columns that are common to the UNIT template for both queries, then UNION them together and remove one of the OPTION lines (which has to be the last line). Since this is tedious, just copy/paste the below query.

```
SELECT eh.Path + eh.Name Element, ts.[Generating Efficiency],
ts.[Utilization],
ts.[Total Hourly Gross Generation], ts.[Carbon Emissions], ts.[Generation
Rate],
ts.[Unit Status], ts.[Technology], ts.[Shift Hours], ts.[Shift], ts.[Operator],
ts.[Net Generation], ts.[Hourly Capacity], ts.[Gross Generation], ts.[Demand]
FROM [Fleet Generation].[Asset].[ElementTemplate] et
INNER JOIN [Fleet Generation].[Asset].[Element] e
        ON et.ID = e.ElementTemplateID
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
        ON e.ID = eh.ElementID
INNER JOIN [Fleet Generation].[DataT].[ft_TransposeSnapshot_Gas
Turbine] ts
        ON eh.ElementID = ts.ElementID
WHERE et.Name = N'Gas Turbine'
UNION
```

```
SELECT eh.Path + eh.Name Element, ts.[Generating Efficiency],
ts.[Utilization],
ts.[Total Hourly Gross Generation], ts.[Carbon Emissions], ts.[Generation
Rate],
ts.[Unit Status], ts.[Technology], ts.[Shift Hours], ts.[Shift], ts.[Operator],
ts.[Net Generation], ts.[Hourly Capacity], ts.[Gross Generation], ts.[Demand]
FROM [Fleet Generation].[Asset].[ElementTemplate] et
INNER JOIN [Fleet Generation].[Asset].[Element] e
        ON et.ID = e.ElementTemplateID
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
        ON e.ID = eh.ElementID
INNER JOIN [Fleet Generation].[DataT].[ft_TransposeSnapshot_UNIT] ts
        ON eh.ElementID = ts.ElementID
WHERE et.Name = N'UNIT'
OPTION (FORCE ORDER, IGNORE ERRORS, EMBED ERRORS)
```

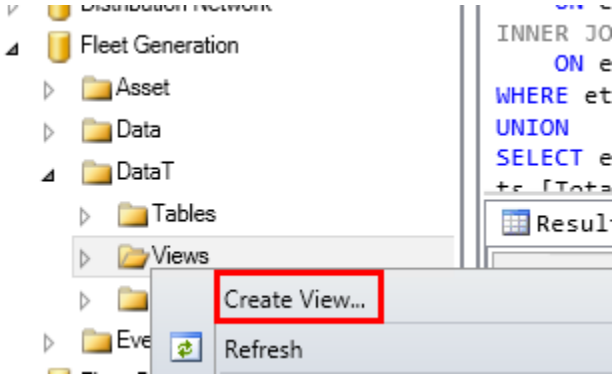
10.9 Saved Views

Often Administrators would prefer to create Views for end-users who are not familiar with SQL queries. Often Views are queried using a basic SELECT * query to return all data without any WHERE clause and without selecting individual columns. This masks the complexity and size of the query (eg. table JOINS and UNIONs of several tables) but places the burden of maintaining the query on the administrator. In future exercises we will be using the queries directly, but it is still useful to know how to create and query views in PI SQL Commander.

10.9.1 Creating dataset views

PI SQL Commander supports the creation of views. Views allow you to name a stored query and it is this name that appears in the table list when importing data into BI clients. Views are the easiest way to allow users to select which datasets they want from PI AF when creating a report, as they do not need to understand the complexity of the underlying SQL query.

Views are created using SQL syntax, but OLEDB Enterprise can give you a template to start with. If you're trying to create a saved query showing information about assets, consider creating it in the Asset schema (folder). If you have a saved query showing data values, for organizational purposes, place it in the Data schema. The image below shows a right-click menu giving the *Create View* option:



Selecting Create View produces the beginning of a query:

```
CREATE VIEW [Fleet Generation].[DataT].[ <view name> ]
AS
<query>
```

At this point, it is a matter of naming the View by replacing <view name> and copy pasting the query by into <query> placeholder.

10.9.2 Directed Activity – View Creation for Unit Performance

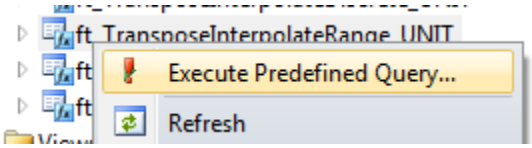
In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

Create a view, Unit Performance, using previously created asset interpolated range transpose functions for frequently changing process data (Transpose Interpolate Range).

Approach:

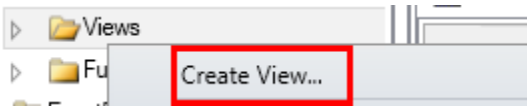
Run the **Unit Interpolated Range transpose function** using the Execute Predefined Query. Note that no INNER JOIN to the ElementTemplate table is necessary as the transpose function was created against the UNIT template.



Modify the **date range** to include data from the past week and a one hour time slice.

```
AND tir.StartTime = DATE('t-7d')
AND tir.EndTime = DATE('t')
AND tir.TimeStep = '1h'
```

Create a view from the transpose function.



Fill in the required information to create the view.

```
CREATE VIEW [Fleet Generation].[DataT].[ <view name> ]
AS
<query>
```

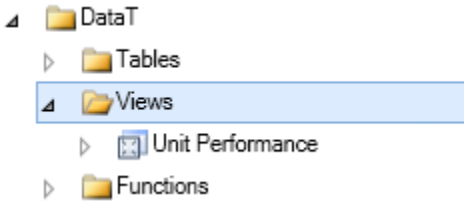
Name the View: **Unit Performance**
Use modified transpose function for the query

```
CREATE VIEW [Fleet Generation].[DataT].[Unit Performance]
AS
SELECT eh.Path + eh.Name Element, tir.*
FROM [Fleet Generation].[Asset].[ElementTemplate] et
INNER JOIN [Fleet Generation].[Asset].[Element] e
        ON et.ID = e.ElementTemplateID
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
        ON e.ID = eh.ElementID
INNER JOIN [Fleet Generation].[DataT].[ft_TransposeInterpolateRange_UNIT] tir
        ON eh.ElementID = tir.ElementID
WHERE et.Name = N'UNIT'
        AND tir.StartTime = DATE(N't-7d')
        AND tir.EndTime = DATE(N't')
        AND tir.TimeStep = N'1h'
OPTION (FORCE ORDER, IGNORE ERRORS, EMBED ERRORS)
```

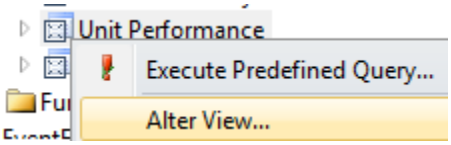
Execute the function.

If successful, a successful message will display, otherwise, an error will be displayed in the lower region of the query section.

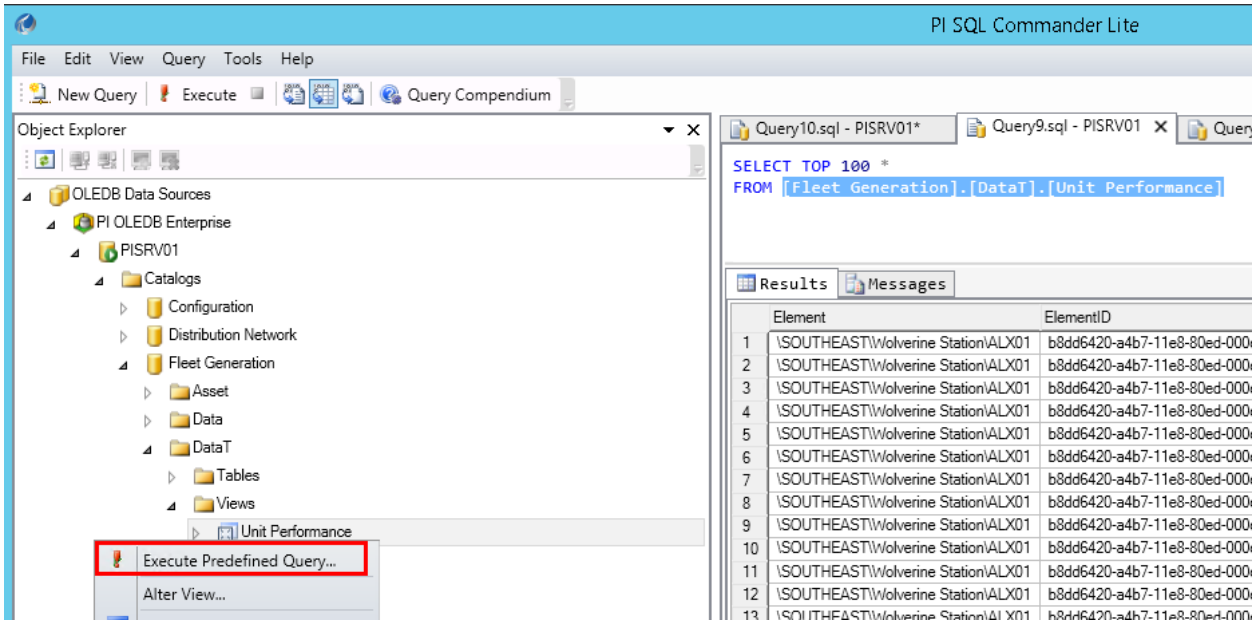
Refresh the View section and verify the Unit Performance View is present.



The definition for the view can be seen by selecting the Alter option in the View folder.



Execute the predefined query to see how the view is referenced ([Fleet Generation].[DataT].[Unit Performance]) and confirm the expected data is returned:



10.10 Importing PI OLEDB Enterprise data to Power BI

The first thing to do when using a client is to import the data you want to analyze. Importing data requires connecting to the data source holding the data, specifying the data you need from the data source (by selecting a database table, view, or writing a query), and then importing the data into the client tool. The following steps will describe how to import the complete datasets from the PI OLEDB Enterprise views defined in the previous sections.

10.10.1 Directed Activity – Importing Data Using PI OLEDB Enterprise

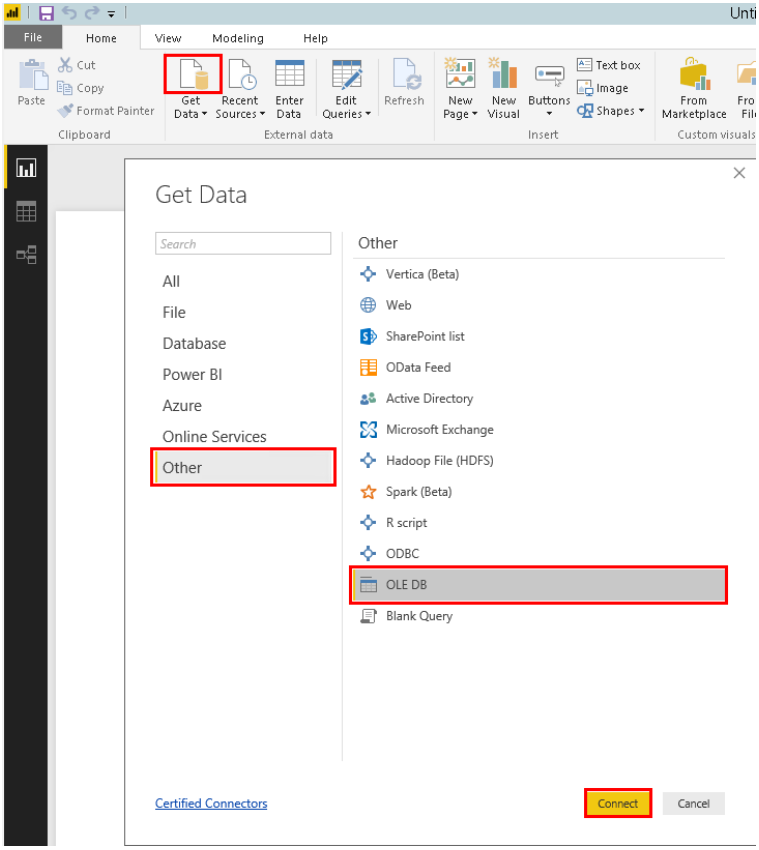
In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

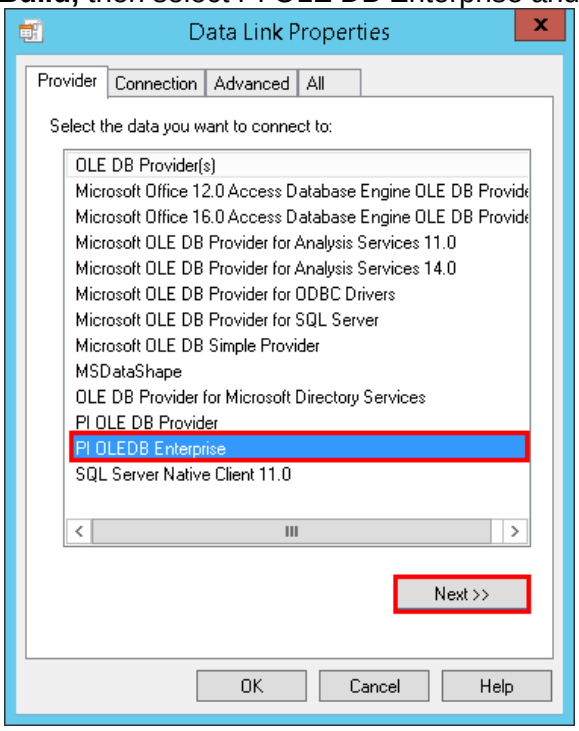
Import Transpose function query results into Power BI.

Approach:

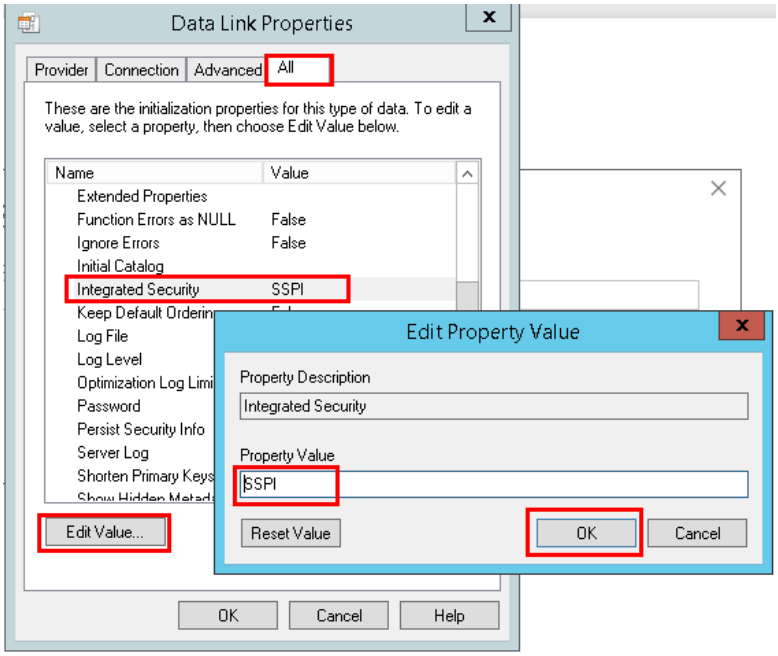
Open MS Power BI Desktop
Select "**Get Data**" and browse to "**other**." Select **OLE DB** and click "**Connect**".



Select **Build,** then select PI OLE DB Enterprise and click **Next**

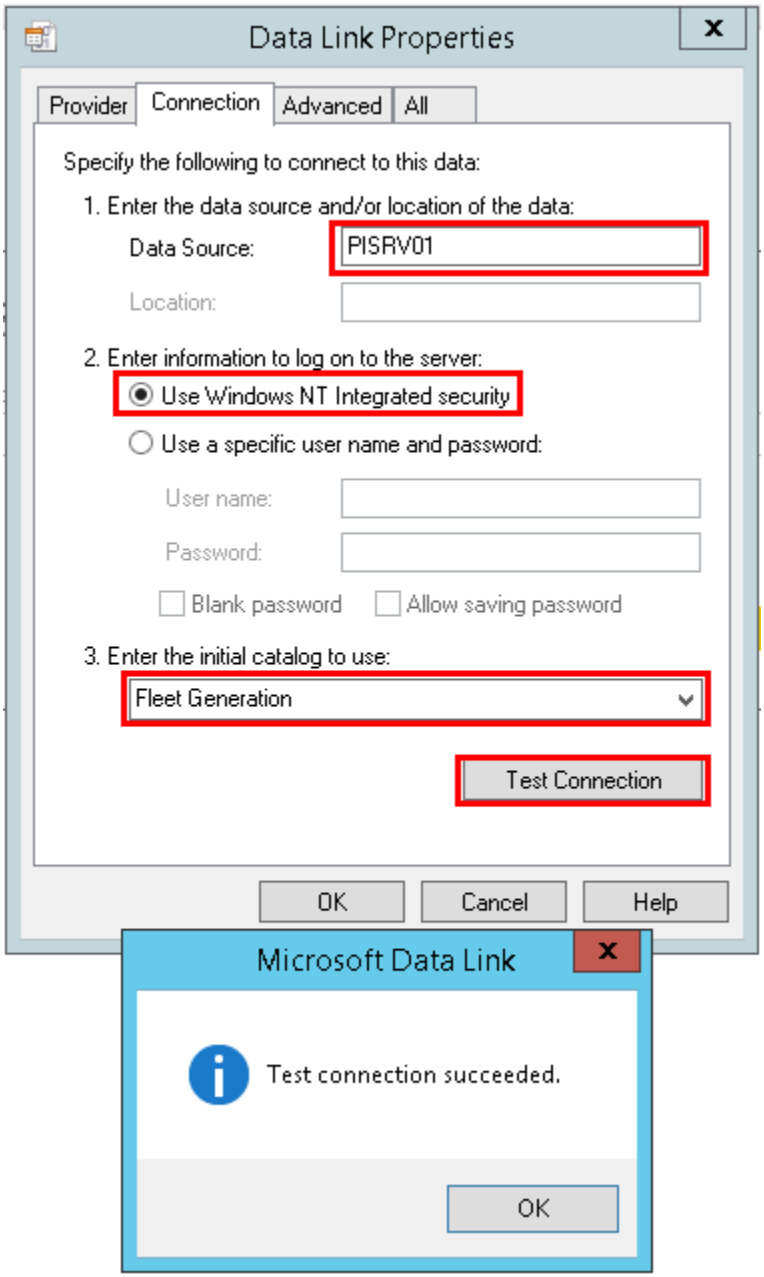


Skip to the **All** tab, and edit the Integrated Security property to **SSPI. This is required to authenticate the connection using the running user's credentials.**



Go back to the Connection tab, enter **PISRV01** as the Data Source, check '**Use Windows NT Integrated Security**', select **Fleet Generation** as the Initial Catalog, and then test the connection.

Use Windows NT Integrated Security basically means connect using the user's login (pischool\student01) and password, and hence the user's level of access. SSPI must have been added in the previous step to use this option. The alternative is to hard-code a user and password, which may give more access than a user is entitled to.

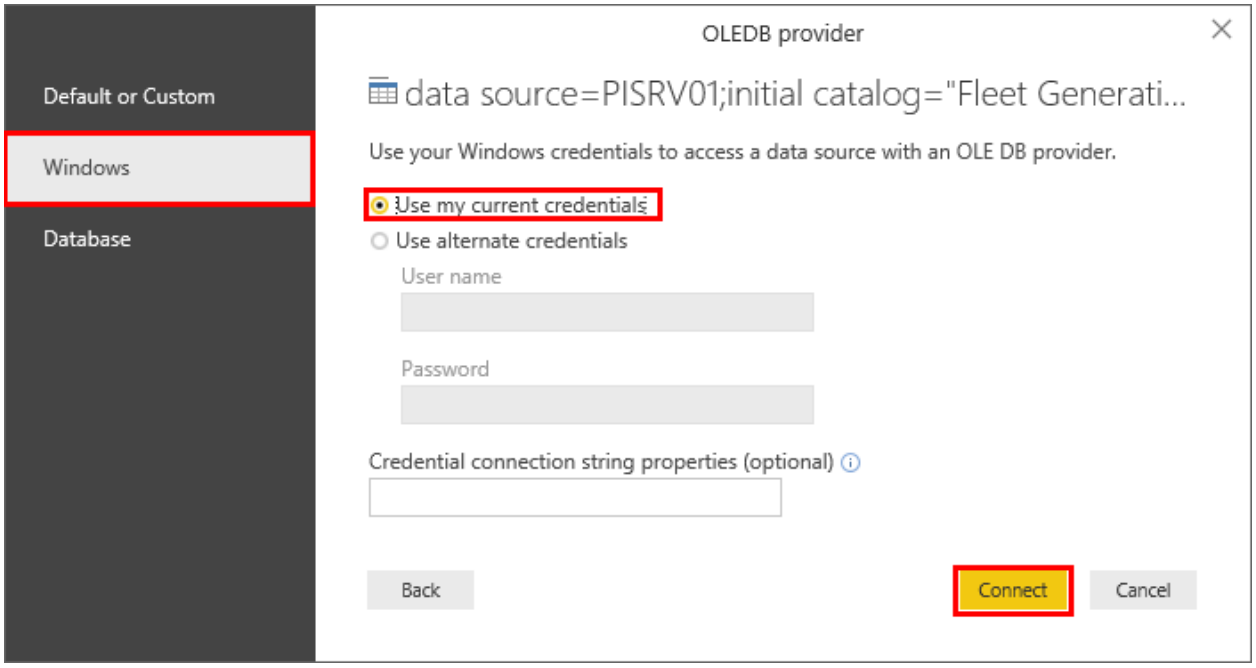


Skip the **Advanced** tab and click **OK** to complete the connection string.

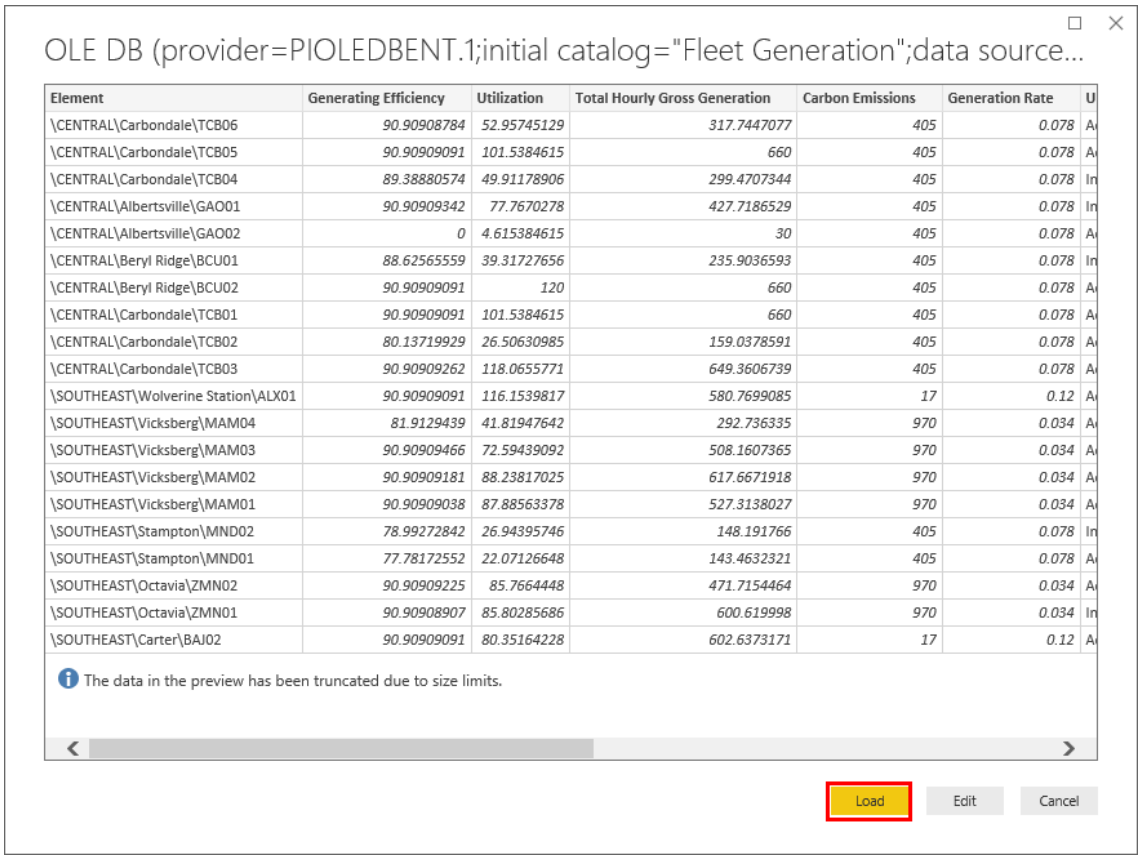
Expand **Advanced options**, here you can enter any SQL statement that has been tested in PI SQL Commander. In this case enter the Interpolate Range UNION query and click OK:

```
SELECT eh.Path + eh.Name Element, ts.[Generating Efficiency],
ts.[Utilization],
ts.[Total Hourly Gross Generation], ts.[Carbon Emissions], ts.[Generation
Rate],
ts.[Unit Status], ts.[Technology], ts.[Shift Hours], ts.[Shift], ts.[Operator],
ts.[Net Generation], ts.[Hourly Capacity], ts.[Gross Generation], ts.[Demand]
FROM [Fleet Generation].[Asset].[ElementTemplate] et
INNER JOIN [Fleet Generation].[Asset].[Element] e
        ON et.ID = e.ElementTemplateID
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
        ON e.ID = eh.ElementID
INNER JOIN [Fleet Generation].[DataT].[ft_TransposeSnapshot_Gas
Turbine] ts
        ON eh.ElementID = ts.ElementID
WHERE et.Name = N'Gas Turbine'
UNION
SELECT eh.Path + eh.Name Element, ts.[Generating Efficiency],
ts.[Utilization],
ts.[Total Hourly Gross Generation], ts.[Carbon Emissions], ts.[Generation
Rate],
ts.[Unit Status], ts.[Technology], ts.[Shift Hours], ts.[Shift], ts.[Operator],
ts.[Net Generation], ts.[Hourly Capacity], ts.[Gross Generation], ts.[Demand]
FROM [Fleet Generation].[Asset].[ElementTemplate] et
INNER JOIN [Fleet Generation].[Asset].[Element] e
        ON et.ID = e.ElementTemplateID
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
        ON e.ID = eh.ElementID
INNER JOIN [Fleet Generation].[DataT].[ft_TransposeSnapshot_UNIT] ts
        ON eh.ElementID = ts.ElementID
WHERE et.Name = N'UNIT'
OPTION (FORCE ORDER, IGNORE ERRORS, EMBED ERRORS)
```

Click the **Windows** tab and select '**Use my current credential'**, then click **Connect**. This should import the query results.

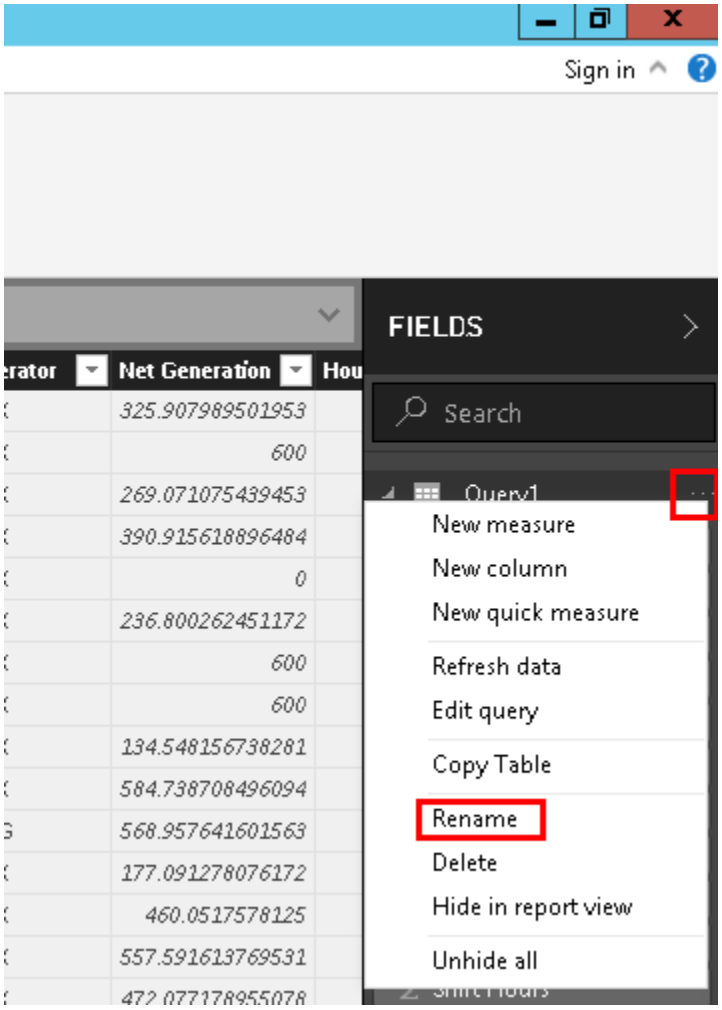


Inspect the preview, then click **Load**.



Element	Generating Efficiency	Utilization	Total Hourly Gross Generation	Carbon Emissions	Generation Rate	U
\CENTRAL\Carbondale\TCB06	90.90908784	52.95745129	317.7447077	405	0.078	A
\CENTRAL\Carbondale\TCB05	90.90909091	101.5384615	660	405	0.078	A
\CENTRAL\Carbondale\TCB04	89.38880574	49.91178906	299.4707344	405	0.078	In
\CENTRAL\Albertsville\GAO01	90.90909342	77.7670278	427.7186529	405	0.078	In
\CENTRAL\Albertsville\GAO02	0	4.615384615	30	405	0.078	A
\CENTRAL\Beryl Ridge\BCU01	88.62565559	39.31727656	235.9036593	405	0.078	In
\CENTRAL\Beryl Ridge\BCU02	90.90909091	120	660	405	0.078	A
\CENTRAL\Carbondale\TCB01	90.90909091	101.5384615	660	405	0.078	A
\CENTRAL\Carbondale\TCB02	80.13719929	26.50630985	159.0378591	405	0.078	A
\CENTRAL\Carbondale\TCB03	90.90909262	118.0655771	649.3606739	405	0.078	A
\SOUTHEAST\Wolverine Station\ALX01	90.90909091	116.1539817	580.7699085	17	0.12	A
\SOUTHEAST\Vicksberg\MAM04	81.9129439	41.81947642	292.736335	970	0.034	A
\SOUTHEAST\Vicksberg\MAM03	90.90909466	72.59439092	508.1607365	970	0.034	A
\SOUTHEAST\Vicksberg\MAM02	90.90909181	88.23817025	617.6671918	970	0.034	A
\SOUTHEAST\Vicksberg\MAM01	90.90909038	87.88563378	527.3138027	970	0.034	A
\SOUTHEAST\Stampton\MND02	78.99272842	26.94395746	148.191766	405	0.078	In
\SOUTHEAST\Stampton\MND01	77.78172552	22.07126648	143.4632321	405	0.078	A
\SOUTHEAST\Octavia\ZMN02	90.90909225	85.7664448	471.7154464	970	0.034	A
\SOUTHEAST\Octavia\ZMN01	90.90908907	85.80285686	600.619998	970	0.034	In
\SOUTHEAST\Carter\BAJ02	90.90909091	80.35164228	602.6373171	17	0.12	A

Change the name of the table from **Query1** to **Unit Specifications**



10.11 Discussion

This is a discussion designed to maximize learning in a specific topic area. Your instructor will have questions, and will prompt for communication within the class. This is an open ended section and the result depends on your needs.

Objective:

Discuss differences between PI Integrator for BA and PI OLEDB Enterprise

Approach

- Which method do you prefer to create views? PI Integrator for Business Analytics or for PI OLEDB Enterprise?
- Pros and Cons of both systems?
- What format would we like the data to be in for processing by BI clients?
- What should be added to the SQL queries to improve the format?
- Do these queries match what we want in our reports?
- If not, what is lacking?

Estimated Completion time 10 minutes.

11 Building the “Fleet Generation” Report

We have now done enough preparation work that we can finally start building the report.

11.1 Preparing and Importing the Tables

In our case, we are going to separate the time-series data from the static data and configure table relationships to join the data sets together. Ideally, PI AF can be the glue that ties multiple data sources together and lends context to the data. Technically, we could put all of our static data into PI AF using table lookups and by assigning attribute values, then designing the queries such that the result set is a single table. However, in real life not all of the data is always in PI and several data sources must be joined together. This can of course be done at the query level, but also in Power BI.

There are ways to avoid doing many of the following steps, but this will help prepare you for real world reports where constant modifications and fine-tuning must be performed.

11.1.1 Directed Activity – Select Static Data Only from Snapshot Query

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

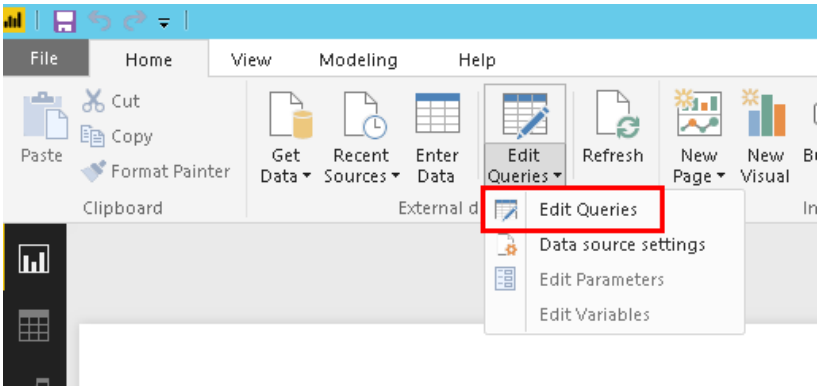
Objective:

Modify the existing query to only include static data.

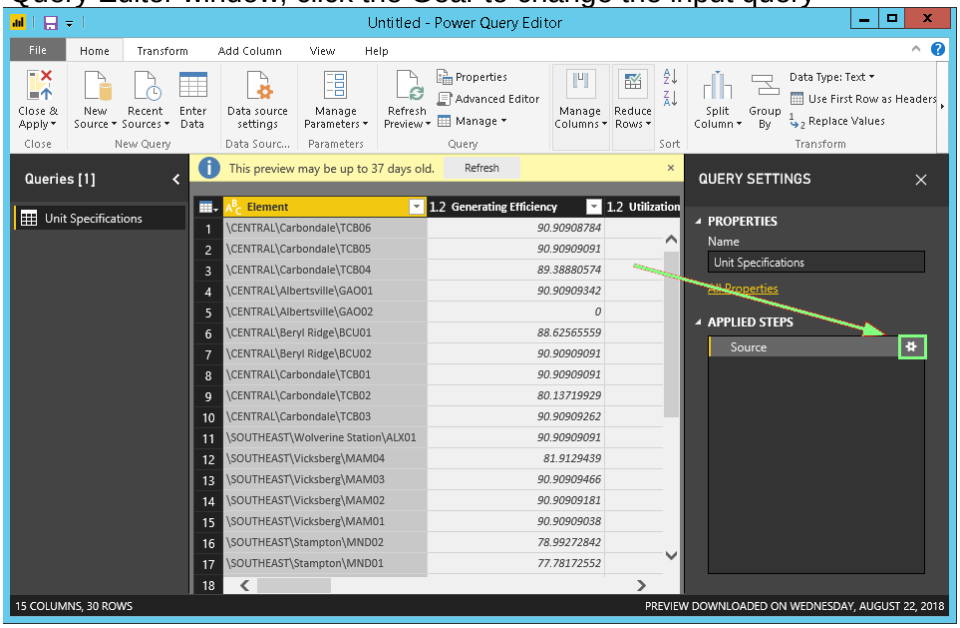
Approach:

- Open the Power BI query editor to inspect a pre-existing query
- Modify the SELECT statement to only include static data columns
- Replace the query in the query editor with a new query

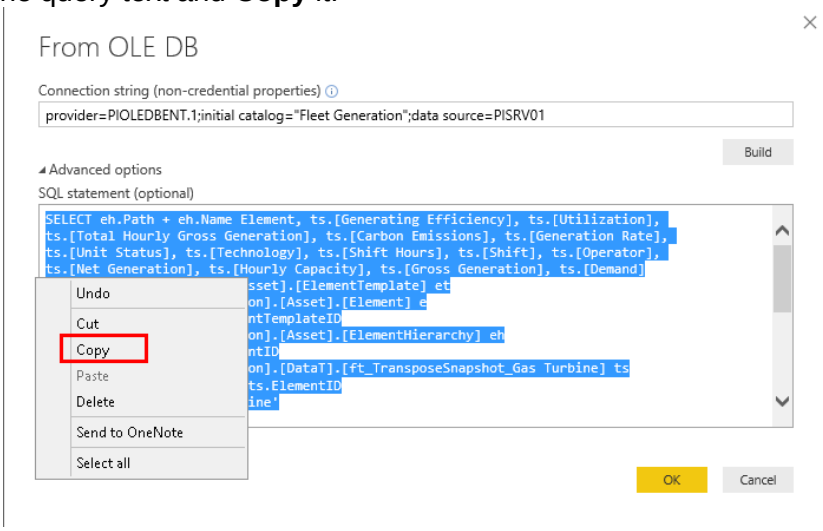
On the Home tab, **Edit Queries**



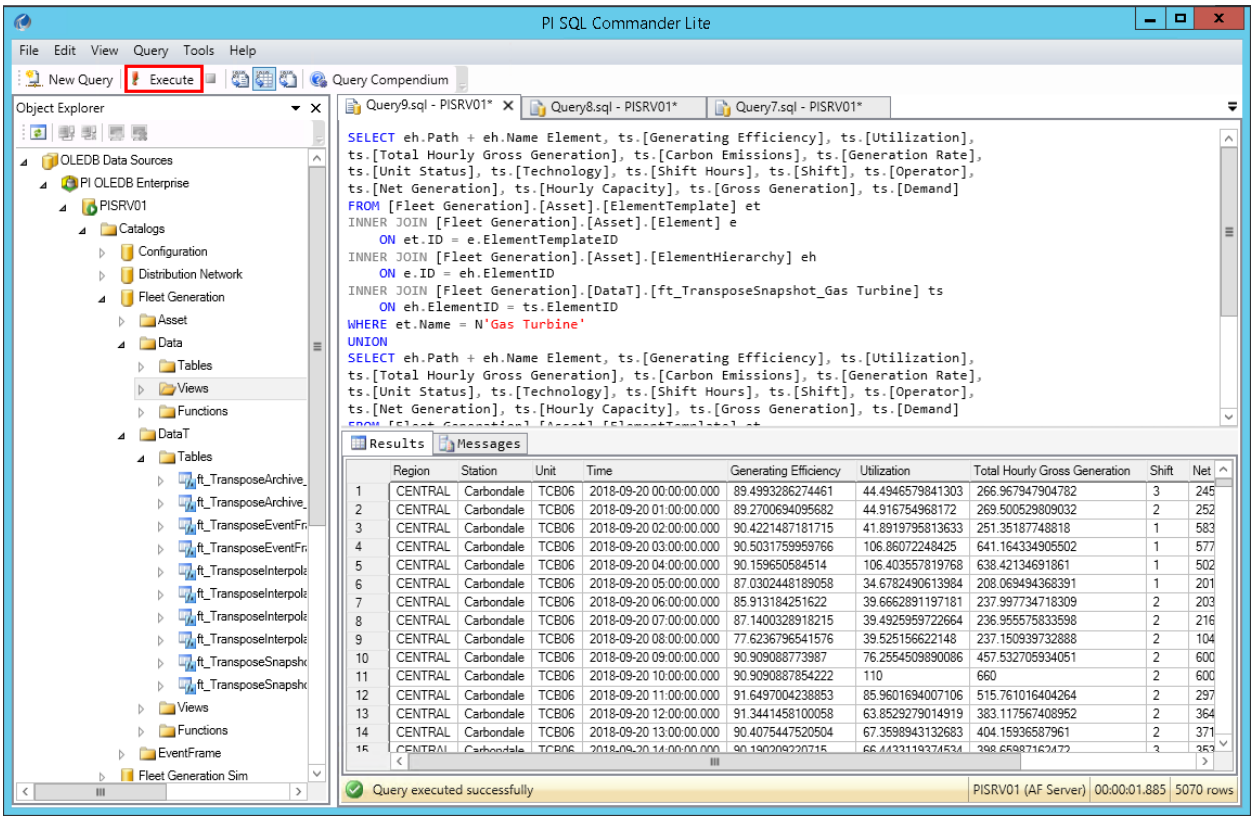
In the Query Editor window, click the Gear to change the input query



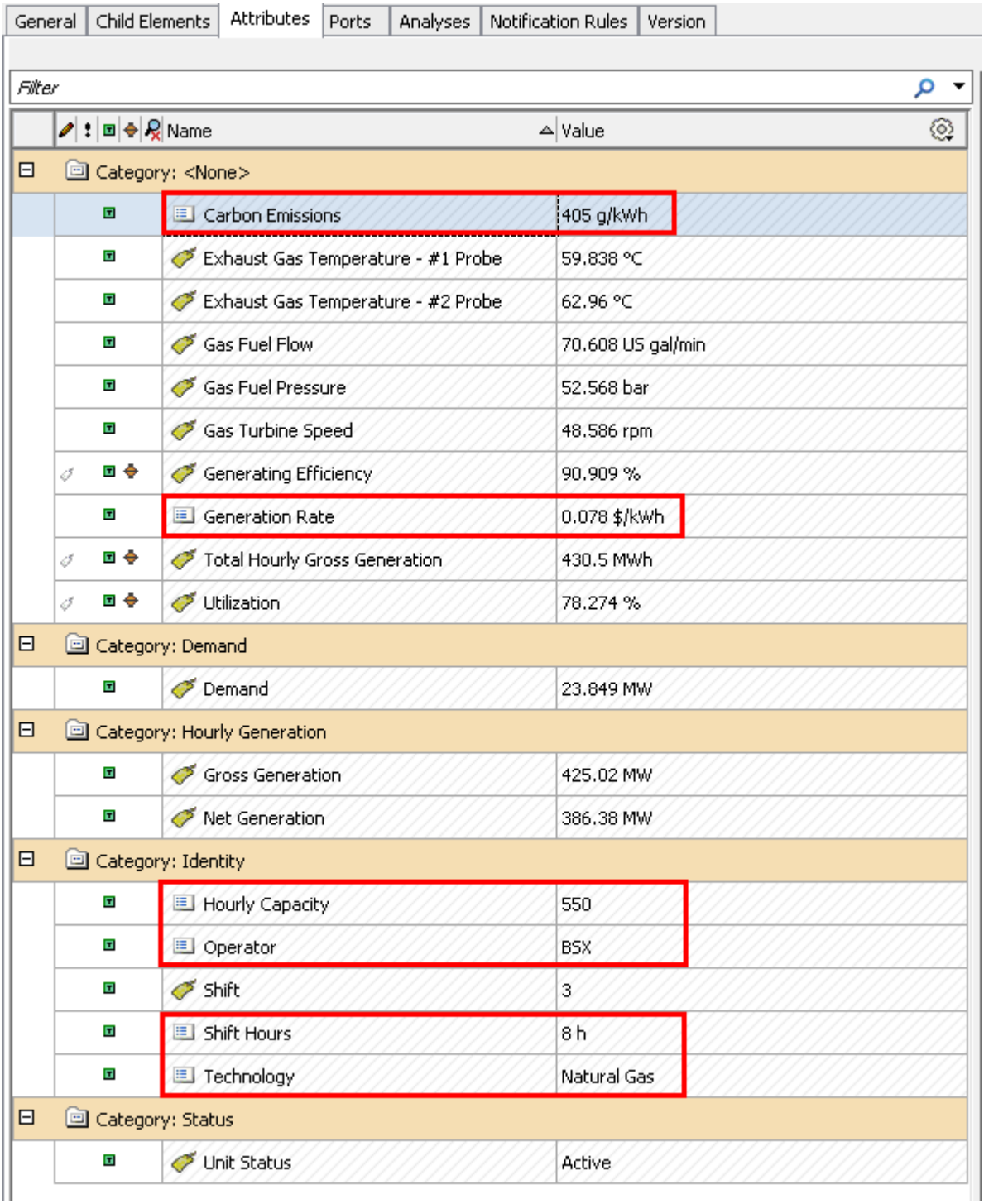
Select all the query text and **Copy** it:



Paste the query into PI SQL Commander. **Execute it as a sanity check.**



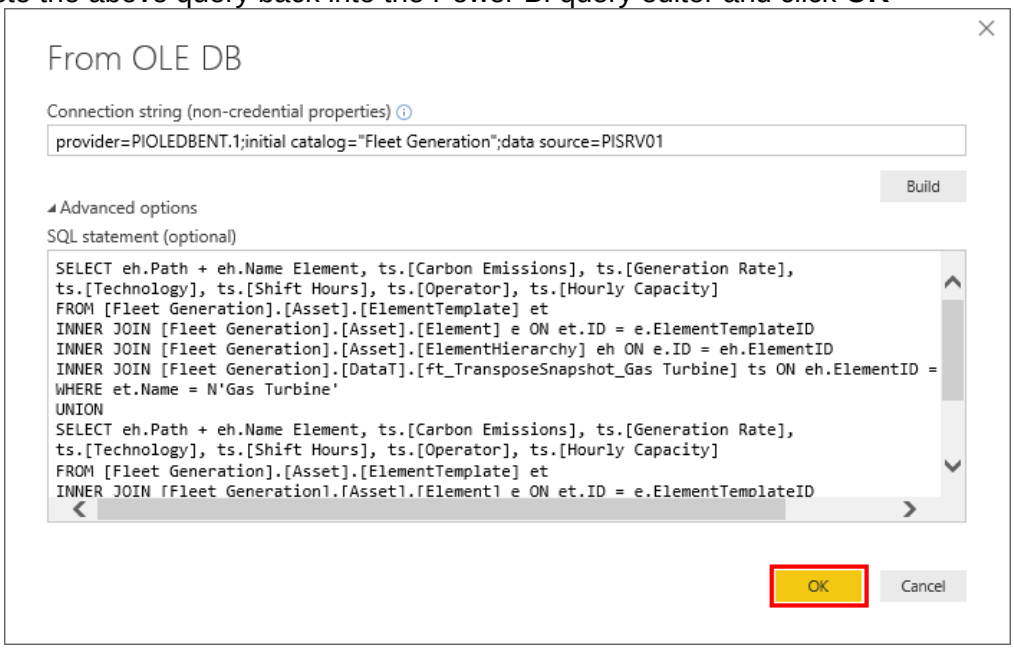
Now the actual modifications. Head over to PI SQL Commander and edit the select statement to only include static attributes: Carbon Emissions, Generation Rate, Hourly Capacity, Operator, Shift Hours, and Technology.



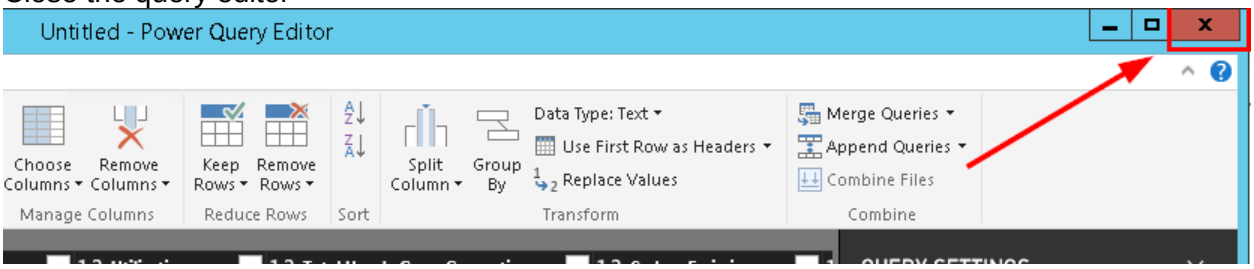
The resulting query is then

```
SELECT eh.Path + eh.Name Element, ts.[Carbon Emissions], ts.[Generation
Rate],
ts.[Technology], ts.[Shift Hours], ts.[Operator], ts.[Hourly Capacity]
FROM [Fleet Generation].[Asset].[ElementTemplate] et
INNER JOIN [Fleet Generation].[Asset].[Element] e ON et.ID =
e.ElementTemplateID
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh ON e.ID =
eh.ElementID
INNER JOIN [Fleet Generation].[DataT].[ft_TransposeSnapshot_Gas Turbine] ts
ON eh.ElementID = ts.ElementID
WHERE et.Name = N'Gas Turbine'
UNION
SELECT eh.Path + eh.Name Element, ts.[Carbon Emissions], ts.[Generation
Rate],
ts.[Technology], ts.[Shift Hours], ts.[Operator], ts.[Hourly Capacity]
FROM [Fleet Generation].[Asset].[ElementTemplate] et
INNER JOIN [Fleet Generation].[Asset].[Element] e ON et.ID =
e.ElementTemplateID
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh ON e.ID =
eh.ElementID
INNER JOIN [Fleet Generation].[DataT].[ft_TransposeSnapshot_UNIT] ts ON
eh.ElementID = ts.ElementID
WHERE et.Name = N'UNIT'
OPTION (FORCE ORDER, IGNORE ERRORS, EMBED ERRORS)
```

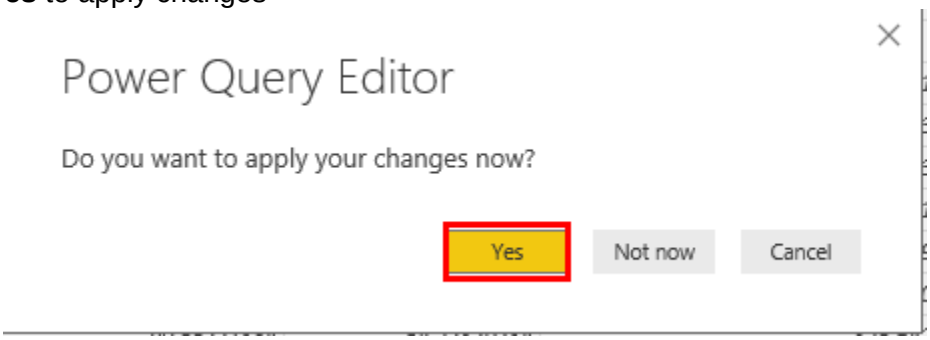
Paste the above query back into the Power BI query editor and click **OK**



Close the query editor



Click **Yes** to apply changes



It should reload the data (30 rows) successfully.

Carbon Emissions	Generation Rate	Technology	Shift Hours	Operator	Hourly Capacity	Element
405	0.078	Natural Gas	8	BSX	600	\CENTRAL\Carbondale\TCB06
405	0.078	Natural Gas	8	BSX	650	\CENTRAL\Carbondale\TCB05
405	0.078	Natural Gas	8	BSX	600	\CENTRAL\Carbondale\TCB04
405	0.078	Natural Gas	8	BSX	550	\CENTRAL\Albertsville\GA001
405	0.078	Natural Gas	8	BSX	650	\CENTRAL\Albertsville\GA002
405	0.078	Natural Gas	8	BSX	600	\CENTRAL\Beryl Ridge\BCU01
405	0.078	Natural Gas	8	BSX	550	\CENTRAL\Beryl Ridge\BCU02
405	0.078	Natural Gas	8	BSX	650	\CENTRAL\Carbondale\TCB01
405	0.078	Natural Gas	8	BSX	600	\CENTRAL\Carbondale\TCB02
405	0.078	Natural Gas	8	BSX	550	\CENTRAL\Carbondale\TCB03
17	0.12	Wind	12	COG	500	\SOUTHEAST\Wolverine Station\ALX01
970	0.034	Coal	12	BSX	700	\SOUTHEAST\Vicksberg\MAM04
970	0.034	Coal	12	BSX	700	\SOUTHEAST\Vicksberg\MAM03
970	0.034	Coal	12	BSX	700	\SOUTHEAST\Vicksberg\MAM02
970	0.034	Coal	12	BSX	600	\SOUTHEAST\Vicksberg\MAM01
405	0.078	Natural Gas	12	BSX	550	\SOUTHEAST\Stampton\MND02
405	0.078	Natural Gas	12	BSX	650	\SOUTHEAST\Stampton\MND01
970	0.034	Coal	12	BSX	550	\SOUTHEAST\Octavia\ZMN02
970	0.034	Coal	12	BSX	700	\SOUTHEAST\Octavia\ZMN01
17	0.12	Wind	12	PRT	750	\SOUTHEAST\Carter\BAJ02
17	0.12	Wind	12	BSX	550	\SOUTHEAST\Brick Canyon\PLT02
17	0.12	Wind	12	BSX	500	\SOUTHEAST\Brick Canyon\PLT01
17	0.12	Wind	8	COG	600	\NORTH\New Bedford\POE01
17	0.12	Wind	8	COG	600	\NORTH\Madison\CEC01
970	0.034	Coal	8	PEE	750	\NORTH\Greenlawn\PTC03
970	0.034	Coal	8	NOP	500	\NORTH\Greenlawn\PTC02
970	0.034	Coal	8	NOP	600	\NORTH\Greenlawn\PTC01
405	0.078	Natural Gas	8	BSX	550	\NORTH\Ebbitt\PQE04
405	0.078	Natural Gas	8	BSX	500	\NORTH\Ebbitt\PQE03
405	0.078	Natural Gas	8	BSX	650	\NORTH\Ebbitt\PQE02

11.1.2 Directed Activity – Split the Element Column

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

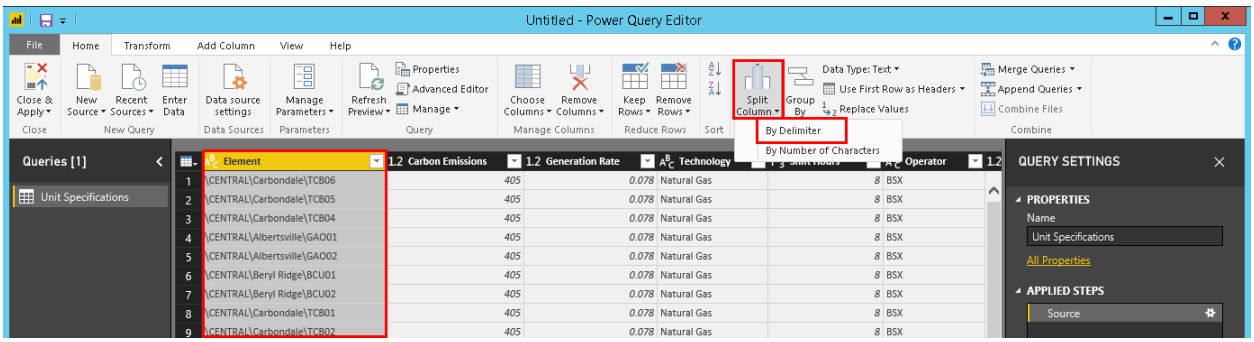
There is still a problem with the data set. We will need to split the Element column into the Region, Station, and Unit name. This could of course be done in the query, but instead we'll use the Power BI split columns feature.

Approach:

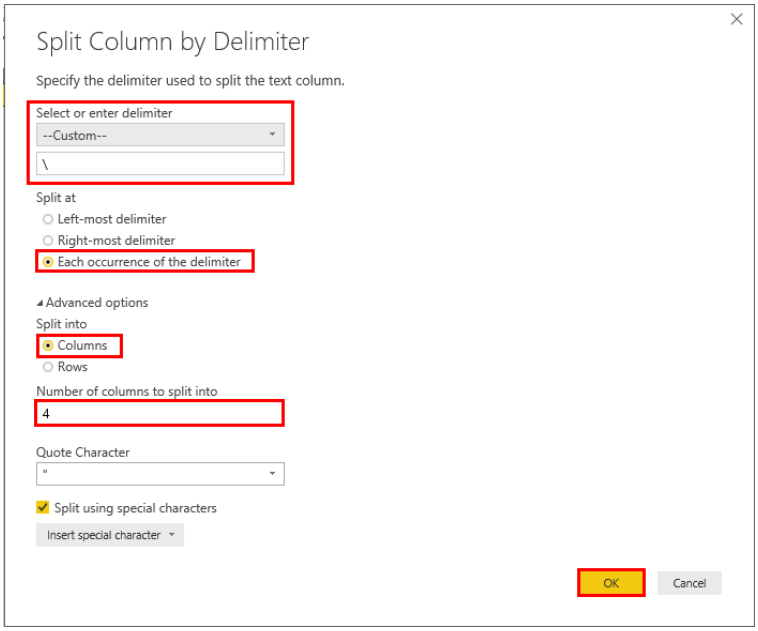
- Open the Power BI query editor and use Split Columns

On the Home tab, **Edit Queries**

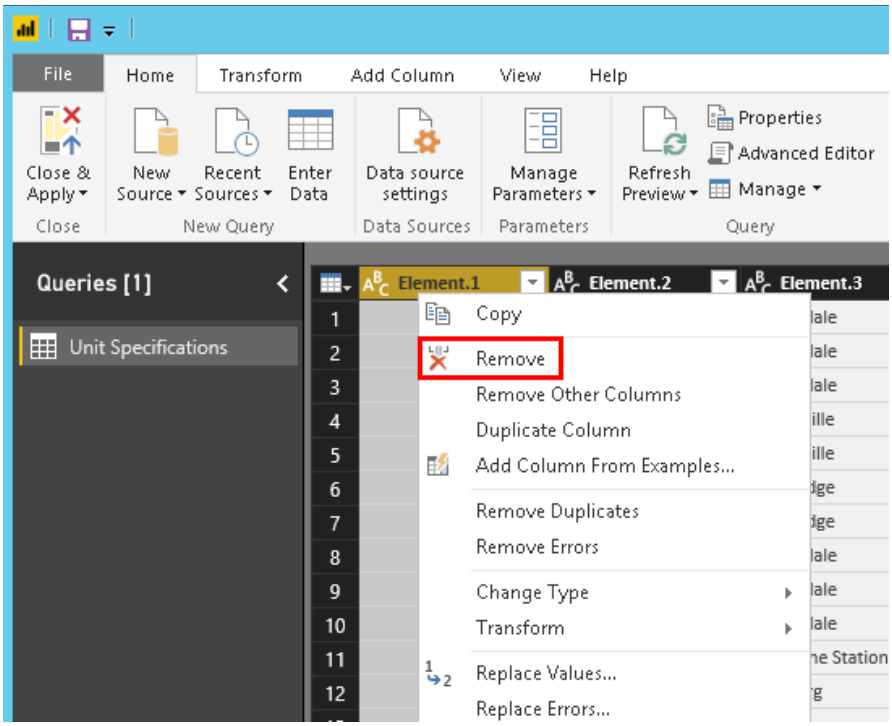
With the **Element** column selected, select **Split Column -> By Delimiter**



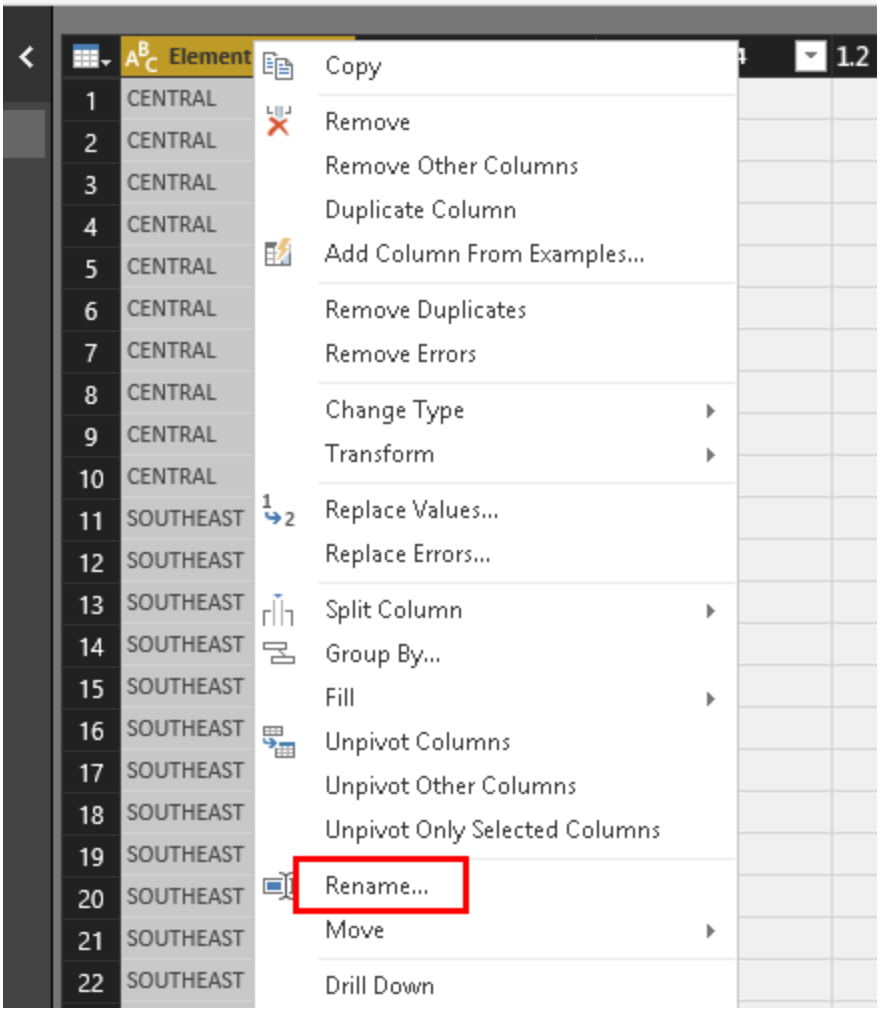
Power BI automatically detected that \ is probably the desired delimiter. The defaults should do exactly what we want here. Just in case though, ensure the following settings are selected and click **OK**:



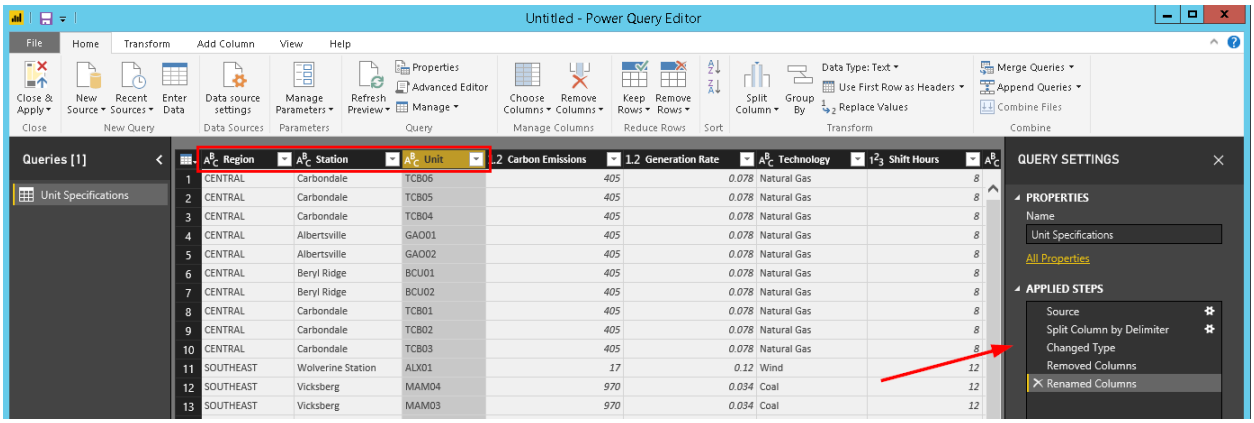
Power BI creates a **blank column** for the first split, **Right-click -> remove** it:



Rename **Element.2** to **Region**, **Element.3** to **Station**, and **Element.4** to **Unit**:



It should now look like the following. Also, note that a number of transformation steps have been applied. Whenever the report is refreshed, these transformations will be performed on the raw input:



Close the query editor, Click **Yes** to apply changes. Now the data is in a suitable format.

Carbon Emissions	Generation Rate	Technology	Shift Hours	Operator	Hourly Capacity	Region	Station	Unit
405	0.078	Natural Gas	8	BSX	600	CENTRAL	Carbondale	TCB06
405	0.078	Natural Gas	8	BSX	650	CENTRAL	Carbondale	TCB05
405	0.078	Natural Gas	8	BSX	600	CENTRAL	Carbondale	TCB04
405	0.078	Natural Gas	8	BSX	550	CENTRAL	Albertsville	GAO01
405	0.078	Natural Gas	8	BSX	650	CENTRAL	Albertsville	GAO02
405	0.078	Natural Gas	8	BSX	600	CENTRAL	Beryl Ridge	BCU01

11.1.3 Directed Activity – Import the Interpolate Range data

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

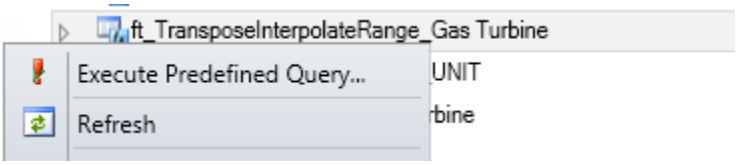
Objective:

Start with predefined queries for Units and Gas Turbines, restrict the result set to exclude static data, and union the results to form a single table.

Approach:

- Execute predefined query for TransposeInterpolateRange
- Modify the SELECT statement to only include static data columns
- Replace the query in the query editor with a new query

Go to PI SQL Commander and execute the predefined query for **ft_TransposeInterpolateRange_Gas Turbine**.



Modify the select statement to split the Element column into Region, Station, and Unit using the **ParentPath()** function. Recall that we did this in Power BI in a previous exercise. Execute the Query to ensure there are no syntax errors.

```
SELECT ParentName(eh.Path,1) as [Region], ParentName(eh.Path,0) as
[Station], eh.Name as [Unit], tir.*
```

Also replace the **tir.*** part to exclude ElementID, StartTime, EndTime, and Timestep columns so that only the Attributes that change over time are included. We will also exclude those attributes unique to Gas Turbines. Execute the Query to ensure there are no syntax errors.

```
SELECT ParentName(eh.Path,1) as [Region], ParentName(eh.Path,0) as
[Station], eh.Name as [Unit],
tir.Time, tir.[Generating Efficiency], tir.[Utilization],
tir.[Total Hourly Gross Generation], tir.[Shift],
tir.[Net Generation], tir.[Gross Generation], tir.[Demand]
```

Change the StartTime to T, EndTime to T-7d, and Timestep to 1h

```
        AND tir.StartTime = DATE(N'T-7d')
        AND tir.EndTime = DATE(N'T')
        AND tir.TimeStep = N'1h'
```

Copy and paste the query before the OPTION statement, then replace Gas Turbine with UNIT in the reference to ft_TransposeInterpolateRange_Gas Turbine and the WHERE clause, and insert a UNION:

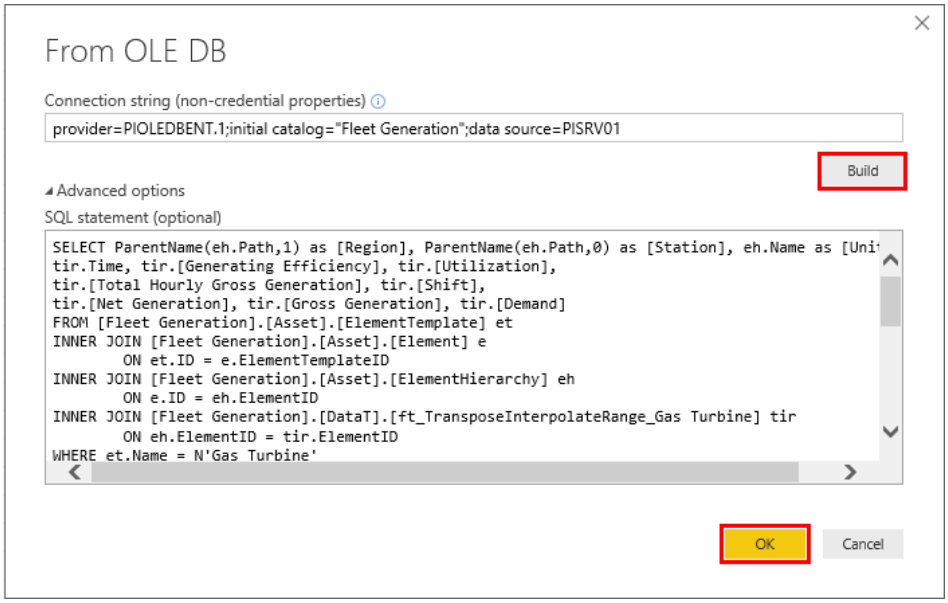
```
SELECT ParentName(eh.Path,1) as [Region], ParentName(eh.Path,0) as
[Station], eh.Name as [Unit],
tir.Time, tir.[Generating Efficiency], tir.[Utilization],
tir.[Total Hourly Gross Generation], tir.[Shift],
tir.[Net Generation], tir.[Gross Generation], tir.[Demand]
FROM [Fleet Generation].[Asset].[ElementTemplate] et
INNER JOIN [Fleet Generation].[Asset].[Element] e
        ON et.ID = e.ElementTemplateID
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
        ON e.ID = eh.ElementID
INNER JOIN [Fleet Generation].[DataT].[ft_TransposeInterpolateRange_Gas
Turbine] tir
        ON eh.ElementID = tir.ElementID
WHERE et.Name = N'Gas Turbine'
        AND tir.StartTime = DATE(N'T-7d')
        AND tir.EndTime = DATE(N'T')
        AND tir.TimeStep = N'1h'
UNION
SELECT ParentName(eh.Path,1) as [Region], ParentName(eh.Path,0) as
[Station], eh.Name as [Unit],
tir.Time, tir.[Generating Efficiency], tir.[Utilization],
tir.[Total Hourly Gross Generation], tir.[Shift],
tir.[Net Generation], tir.[Gross Generation], tir.[Demand]
FROM [Fleet Generation].[Asset].[ElementTemplate] et
INNER JOIN [Fleet Generation].[Asset].[Element] e
        ON et.ID = e.ElementTemplateID
INNER JOIN [Fleet Generation].[Asset].[ElementHierarchy] eh
        ON e.ID = eh.ElementID
INNER JOIN [Fleet
Generation].[DataT].[ft_TransposeInterpolateRange_UNIT] tir
        ON eh.ElementID = tir.ElementID
WHERE et.Name = N'Unit'
```

```
        AND tir.StartTime = DATE(N'T-7d')
        AND tir.EndTime = DATE(N'T')
        AND tir.TimeStep = N'1h'
OPTION (FORCE ORDER, IGNORE ERRORS, EMBED ERRORS)
```

Execute the query to make sure it still works. Then head back to Power BI.

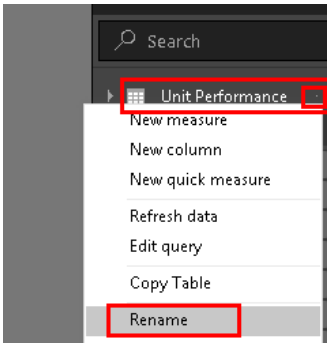
In Power BI, do **Get Data -> OLE DB**. Build the connection string, enter the query where it says **Advanced options**, and click **OK**. Inspect the preview and Load the data.

We've done this before so there isn't a screenshot for every click this time.



It should import 5070 rows, 30 units x 24 hours x 7 days = 5040, plus 30 rows (1 per unit) for the start time.

Rename **Query1** to **Unit Performance**



11.1.4 Directed Activity – Inspect Table Relationship

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

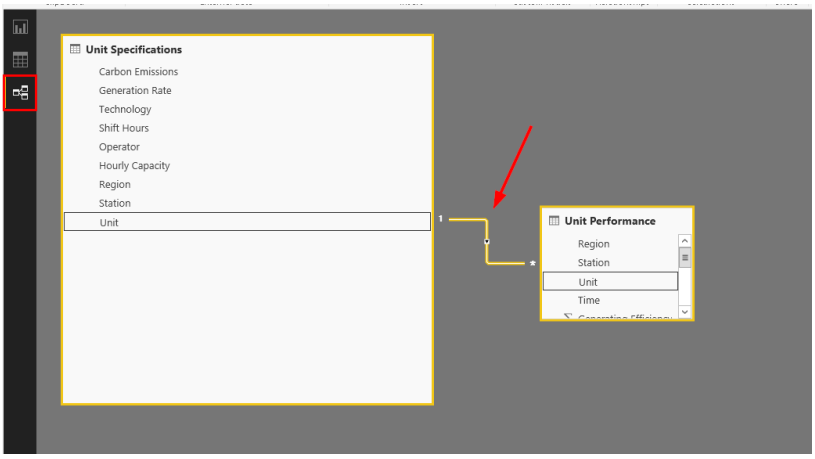
Objective:

Inspect the automatically created table relationship. Power BI should have detected two identically named columns exhibiting a one-to-many relationship.

Approach:

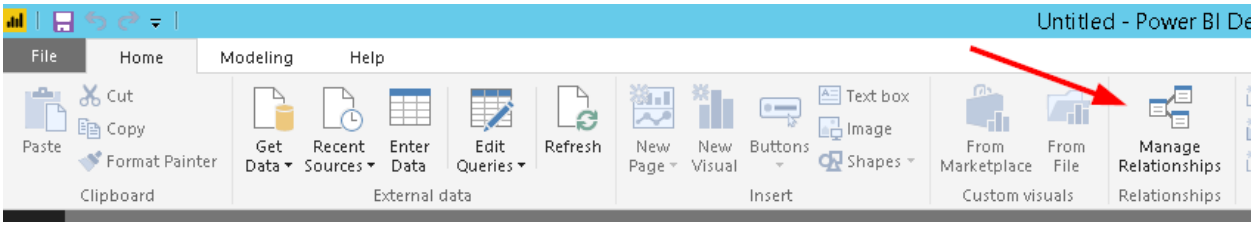
- Open the Power BI relationships tab and inspect the existing relationship

In Power BI, Go to the **Relationships** tab, then move the Unit Performance table to the right so that the relationship line is clearly visible and click on the line:



We can see that Power BI has already detected the relationship between the two tables. This can be thought of as a graphical representation of an INNER JOIN statement. These tables are now joined on the Unit column. For this to work, one of the tables must only contain unique values in the Unit column (ie. the column can serve as a key), as is the case here. This is referred to as a one-to-many relationship in some documentation. Each Unit only appears **once** in the Unit Specifications table, whereas each Unit appears **many** times in the Unit Performance table.

Relationships can be manually defined using a drag and drop interface, or through Manage Relationships.



However at this point there is no need.

11.2 Augmenting the Data using DAX

Next we will add a few calculations to the Unit Performance table that will help assess the total Emissions produced and the total cost of generation. We will also add columns for the day of the week and sort the Weekday in Sunday -> Saturday order.

11.2.1 Directed Activity – Calculate the amount of CO2 produced every hour

In this part of the class, you will perform a learning activity to explore the different concepts presented in this chapter or section. You may be invited to watch what the instructor is doing or perform the same steps at the same time. You may play a game or hold a quiz. Your instructor will have directions.

Objective:

Add a DAX formula Calculate the amount of CO2 produced every hour

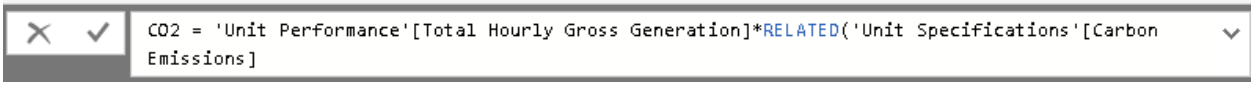
Approach:

- Add and additional column to the Unit Performance table with the amount of carbon emissions produced.

In Power BI, navigate to the **Data Tab** and select the **Unit Performance** table.

Right-click any column and add a **new column**. Enter the following formula:

CO2 = 'Unit Performance'[Total Hourly Gross Generation]*RELATED('Unit Specifications'[Carbon Emissions]



Note that Total Hourly Gross Generation has units of MWh, and Carbon Emissions has units of g/kWh. Grams/kWh is the same as Kilograms/MWh, and therefore the result will be in KG.

11.2.2 Exercise – Calculate the Generation Cost

This solo or group exercise is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the exercise.

Objective: Add the cost calculation column to your Unit Performance table

Do you prefer having the calculations within AF as a formula data reference, or within the BI client tools? What are some advantages and disadvantages of each?

Approach:

- Add and additional column named **Cost** to the Unit Performance table with the dollar cost per hour.
- Take note of the input units. Cost should be in dollars.

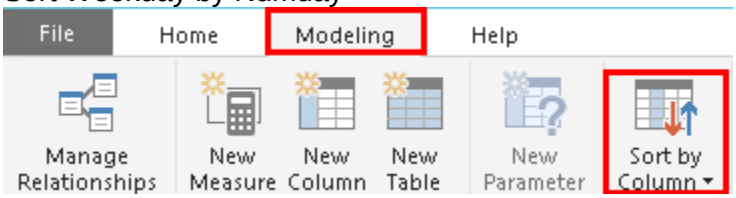
11.2.3 Exercise – Add Column for Day of the Week and sort

This solo or group exercise is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the exercise.

Objective: Add the day of the week to your Unit Performance table, also add a column with the numerical day of the week and sort by this value

Approach:

- Add and additional column named **Weekday** which shows the day of the week as a string using the FORMAT() function
- Add another column named Numday which gives the numerical day of the week using the WEEKDAY() function
- Sort Weekday by Numday



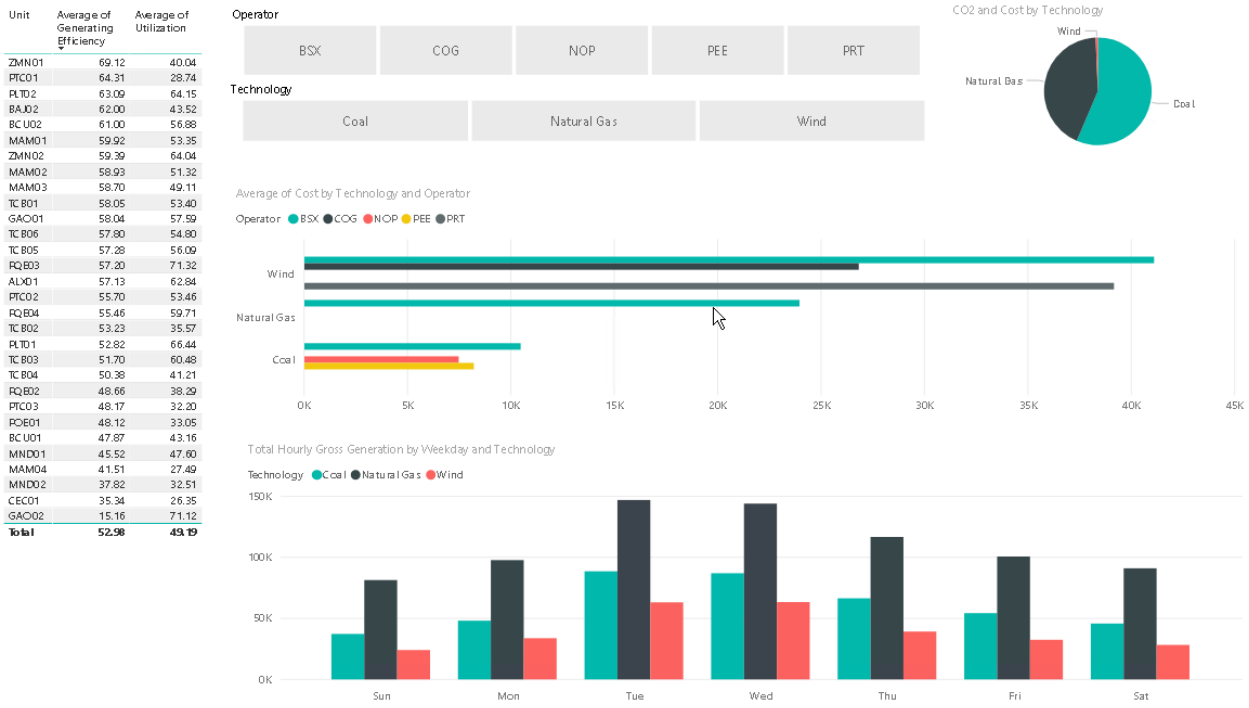
11.3 Configuring the Visualizations

Now we will add visuals to the report to convey useful information about the generating units.

11.3.1 Exercise – Build the Report

This solo or group exercise is designed to maximize learning in a specific topic area. Your instructor will have instructions, and will coach you if you need assistance during the exercise.

Objective: Build an interactive Report comparing KPIs for different generation technologies and operators.



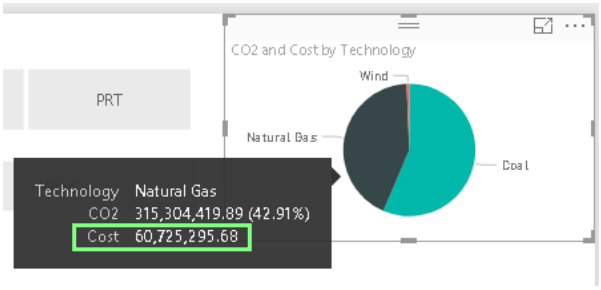
Unit	Average of Generating Efficiency	Average of Utilization
ZMN01	69.12	40.04
PTC01	64.31	28.74
PLT02	63.09	64.15
BAJ02	62.00	43.52
BCU02	61.00	56.88
MAM01	59.92	53.35
ZMN02	59.39	64.04
MAM02	58.93	51.32
MAM03	58.70	49.11
TCB01	58.05	53.40
GAO01	58.04	57.59
TCB06	57.80	54.80
TCB05	57.28	56.09
FQB03	57.20	71.32
ALX01	57.13	62.84
PTC02	55.70	53.46
FQB04	55.46	59.71
TCB02	53.23	35.57
PLT01	52.82	66.44
TCB03	51.70	60.48
TCB04	50.38	41.21
FQB02	48.66	38.29
PTC03	48.17	32.20
POE01	48.12	33.05
BCU01	47.87	43.16
MND01	45.52	47.60
MAM04	41.51	27.49
MND02	37.82	32.51
CEC01	35.34	26.35
GAO02	15.16	71.12
Total	**52.98**	**49.19**

Approach:

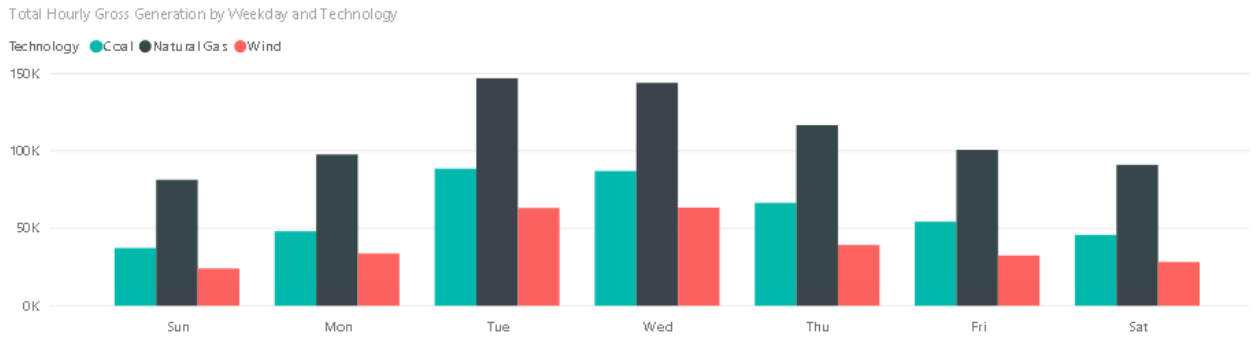
- Add a **Table** showing Average Generating Efficiency and Average Utilization by Unit

Unit	Average of Generating Efficiency	Average of Utilization
ZMN01	69.12	40.04
PTC01	64.31	28.74
PLT02	63.09	64.15
BAJ02	62.00	43.52
BC U02	61.00	56.88
MAM01	59.92	53.35
ZMN02	59.39	64.04
MAM02	58.93	51.32
MAM03	58.70	49.11
TC B01	58.05	53.40
GAO01	58.04	57.59
TC B06	57.80	54.80
TC B05	57.28	56.09
FQB03	57.20	71.32
ALX01	57.13	62.84
PTC02	55.70	53.46
FQB04	55.46	59.71
TC B02	53.23	35.57
PLT01	52.82	66.44
TC B03	51.70	60.48
TC B04	50.38	41.21
FQB02	48.66	38.29
PTC03	48.17	32.20
POE01	48.12	33.05
BC U01	47.87	43.16
MND01	45.52	47.60
MAM04	41.51	27.49
MND02	37.82	32.51
CEC01	35.34	26.35
GAO02	15.16	71.12
Total	**52.98**	**49.19**

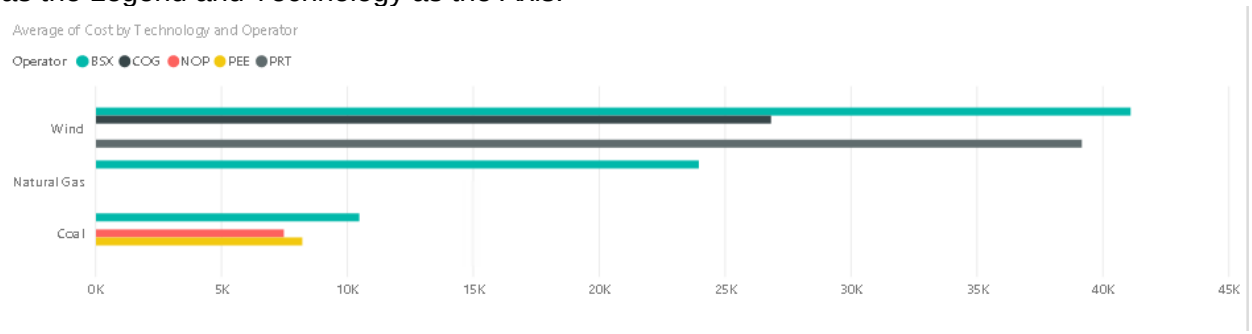
- Add a **Pie Chart** showing how the **C02 emissions** from each generation technology contribute to the whole. Add a **Tooltip** that shows the **Cost** when the user hovers over the Pie Chart



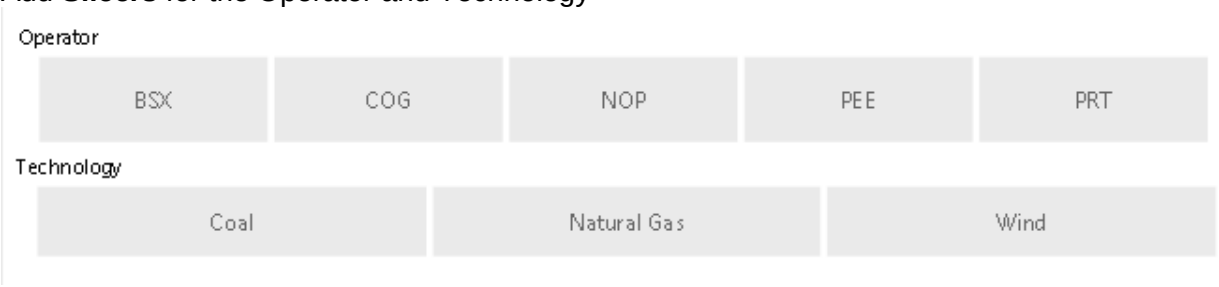
- Add a **Clustered Column Chart** showing the Sum of Total Hourly Gross Generation with Technology as the Legend and Weekday as the Axis



- Add a **Clustered Bar Chart** showing the Average Hourly Cost with Operator as the Legend and Technology as the Axis.



- Add **Slicers** for the Operator and Technology



- Optionally improve the look and feel of the report through the use of formatting. Bump up the font sizes, adjust column names and titles, etc.

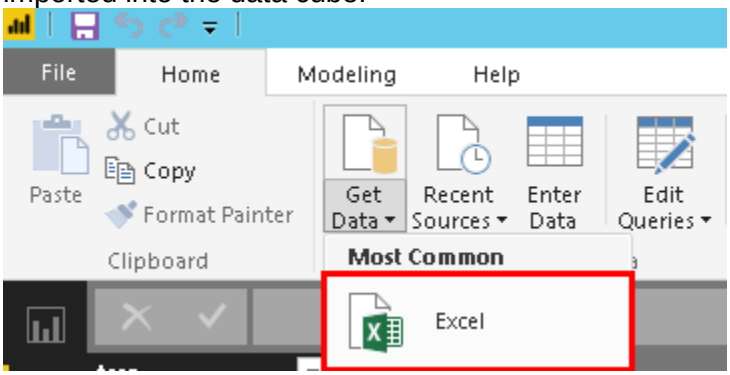
12 Final Exercise: Create a Report

Objective:

Determine the carbon footprint of each unit and display on a US map. Also create a report to analyze downtime (Inactivity) events.

Approach:

- Create a new Sheet in the Fleet Generation Report (the imported tables will be re-used)
- Geospatial information for all units in Fleet Generation is located in **C:\Class\Final Exercise\Unit Coordinates.xlsx**. This data will need to be imported into the data cube.



.

- To get the Inactivity Events, you can either use PI OLEDB Enterprise or PI Integrator for BA.
    - You need a column to form the relationship between the Unit Specifications table and the Inactivity Event Frames, it's probably easiest to join on Unit Name (GAO01, etc).
    - Extract Event frames for the last 7 days
    - **If using PI Integrator for BA** to publish the event frames, it's probably easiest to add the Unit Name to the Event Frame template.
    - **If using PI OLEDB Enterprise**, start with the ft_TransposeEventFrameSnapshot_Inactivity predefined query and modify it as necessary.

- Import the Inactivity events for the last 7 days using whichever method you prefer.

- Create the table relationships (should happen automatically if all columns are named Units).
    - Between the Unit Specifications table and the longitude/latitude table
    - Between the Unit Specifications table and the Inactivity query results

- Insert a map within the client to display the location of each of the units and the associated total hourly carbon emissions.

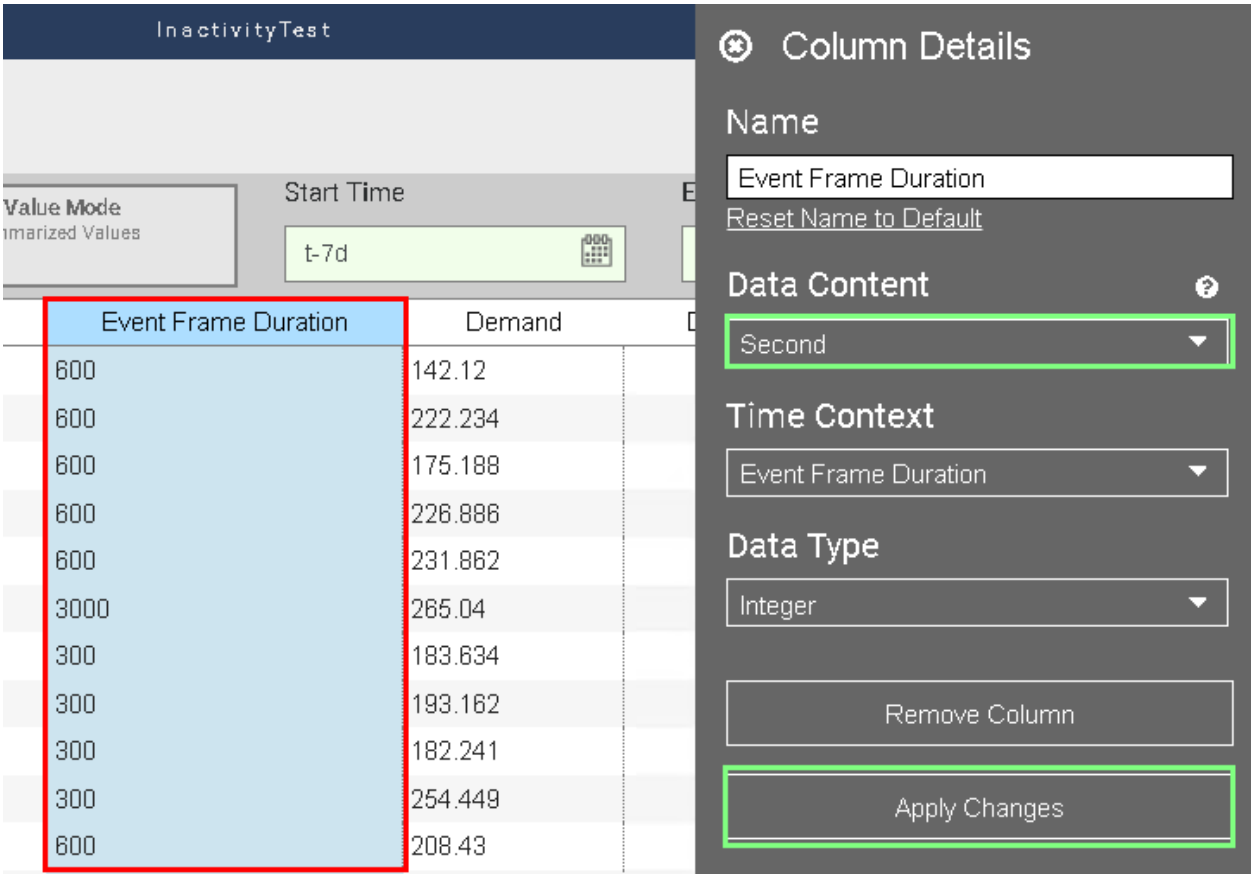
- Insert a table showing the number of downtime events (Inactivity Event Frames) and average duration of event frames for each unit. Add the Average Utilization to the same table.

- Configure the report in such a way that the Table relationships are tested. Use data from multiple tables in the same Visual.

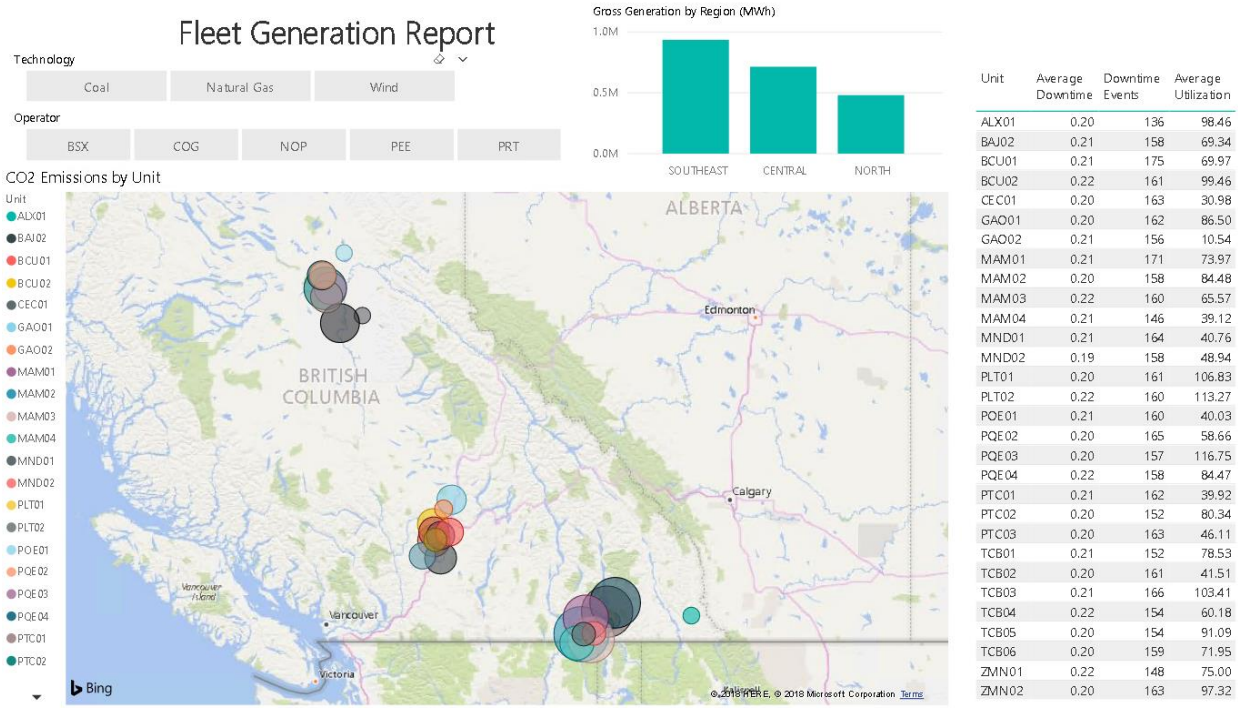
- Customize the display to make it more user friendly for later use and report generation. Improve the formatting and add slicers.

Hints:

- If using PI Integrator for BA to publish the Inactivity Event frames, the Data Context must be set to Second or else it will round to the nearest whole hour (which will always be zero).



A sample of what the report could look like:



Unit	Average Downtime	Downtime Events	Average Utilization
ALX01	0.20	136	98.46
BAJ02	0.21	158	69.34
BCU01	0.21	175	69.97
BCU02	0.22	161	99.46
CEC01	0.20	163	30.98
GAO01	0.20	162	86.50
GAO02	0.21	156	10.54
MAM01	0.21	171	73.97
MAM02	0.20	158	84.48
MAM03	0.22	160	65.57
MAM04	0.21	146	39.12
MND01	0.21	164	40.76
MND02	0.19	158	48.94
PLT01	0.20	161	106.83
PLT02	0.22	160	113.27
POE01	0.21	160	40.03
PQE02	0.20	165	58.66
PQE03	0.20	157	116.75
PQE04	0.22	158	84.47
PTC01	0.21	162	39.92
PTC02	0.20	152	80.34
PTC03	0.20	163	46.11
TCB01	0.21	152	78.53
TCB02	0.20	161	41.51
TCB03	0.21	166	103.41
TCB04	0.22	154	60.18
TCB05	0.20	154	91.09
TCB06	0.20	159	71.95
ZMN01	0.22	148	75.00
ZMN02	0.20	163	97.32

The above report can be found in **C:\Class\Final Exercise\Solution**

13 Appendix A Substitution Parameters

Defining the Substitution Parameters
The substitution parameters are listed in the following table. The ones in bold are the commonly used “Name” substitution parameters for Elements, Attributes, or Event Frames.

Parameter	Will be replaced by this object’s name:
%..\Element%	The name of the owning element of the element in which the attribute resides. To retrieve further ancestors, use the '..\' notations, such as %..\..\Element%.
%..|Attribute%	The name of the owning attribute in which the attribute resides. To retrieve further ancestors, use the '..|' notations, such as %..|..|Attribute%.
%@Attribute%	The value of the attribute referenced. To retrieve further ancestors, use the '..|' notations, such as %@..|..|Attribute%.
%\Element%	The name of the root AF Element in which the attribute resides.
%<Environment Variable>%	The matching System Environment Variable’s value. For example %COMPUTERNAME% is replaced with the name of the computer on which the Data Reference is executing.
%Analysis%	The name of the analysis if it can be obtained from the context.
%Attribute%	The name of the attribute that holds this data reference.
%AttributeId%	The attribute ID that holds this data reference.
%Database%	The name of the AF Database in which the attribute resides.
%Description%	The description of the attribute that holds this data reference.
%Element%	The name of the AF Element in which the attribute resides.
%ElementDescription%	The description of the element in which the attribute resides.
%ElementId%	The element ID that holds this data reference.
%EndTime%	The local end time if it can be obtained from the time context.
%Model%	The name of the model if it can be obtained from the context.
%Server%	The name of the default PI Data Archive of the AF Database in which the attribute resides.
%StartTime%	The local start time if it can be obtained from the time context.
%System%	The name of the PI System in which the attribute resides.

%Time%	The local time if it can be obtained from the time context.
%UtcEndTime%	The coordinated universal (UTC) end time if it can be obtained from the time context.
%UtcStartTime%	The coordinated universal (UTC) start time if it can be obtained from the time context.
%UtcTime%	The coordinated universal (UTC) time if it can be obtained from the time context.
.\	The current reference
[.]	The default object of the parent collection. For example .\Elements[.] |Temperature returns the temperature attribute from the primary element of the current reference's Elements collection.
[@filter=text]	The search string in text (e.g. Tank*) matches the given filter. Supported filters are: @Name, @Index, @Template, @Category, @ReferenceType, @Description, @Type, @UOM.
[@Index=#]	Returns the result at location # from the collection result.

14 Appendix B Performance Equation Operands and Functions

Taken from the PI Data Archive Application User Guide

Operands in Performance Equations

Operand Type	Syntax Requirements	Examples
Numbers	(none)	1342 98.6 .0015 1.2e2
Tagnames	In single quotes	'sinusoid' 'ba:level.1' 'ba.phase.1'
PI Time Expressions	In single quotes	'01-dec-03' '16-jul-94' '*'
Strings	In double quotes	"string string string" "sinusoid"
Functions	Must be a Performance Equation function	TagVal('sinusoid') TagAvg('sinusoid') Cos('sinusoid')

Functions Listed By Type

The following tables list all functions by type. This list can also be found in the PIPC\HELP\PEReference.chm help file.

Math Functions

Name	Description
Abs	Absolute value
Asin	Arc sine
Acos	Arc cosine
Atn	Arc tangent
Atn2	Arc tangent (two arguments)
Cos	Cosine
Cosh	Hyperbolic cosine
Exp	Exponential
Float	Conversion of string to number
Frac	Fractional part of number
Int	Integer part of number
Log	Natural logarithm
Log10	Common logarithm
Poly	Evaluate polynomial
Round	Round to nearest unit
Sgn	Numerical sign
Sin	Sine
Sinh	Hyperbolic sine
Sqr	Square root
Tanh	Hyperbolic tangent

Tan	Tangent
Trunc	Truncate to next smaller unit

Aggregate Functions

Name	Description
Avg	Average
Max	Maximum
Median	Median selector
Min	Minimum
PStDev	Population standard deviation
SStDev	Sample standard deviation
Total	Sum

Miscellaneous Functions

Name	Description
BadVal	See if a value is bad (not a number or time)
Curve	Get value of a curve
DigState	Get digital state from a string
IsDST	Test whether a time is in local daylight savings time period
IsSet	Test if a PI value is annotated, substituted, or questionable
StateNo	The code number of a digital state
TagBad	See if a point has an abnormal state

PI Archive Retrieval

Name	Description
NextEvent	Time of a point's next Archive event
NextVal	Point's next value after a time
PrevEvent	Time of a point's previous Archive event
PrevVal	Point's previous value before a time
TagVal	Point's value at a time

PI Archive Search

Name	Description
FindEq	Timestamp when point = value
FindGE	Timestamp when point >= value

FindGT	Timestamp when point > value
FindLE	Timestamp when point <= value
FindLT	Timestamp when point < value
FindNE	Timestamp when point ~= value
TimeEq	Total period when point = value
TimeGE	Total period when point >= value
TimeGT	Total period when point > value
TimeLE	Total period when point <= value
TimeLT	Total period when point < value
TimeNE	Total period when point ~= value

PI Archive Statistics

Name	Description
EventCount	Number of Archive events
PctGood	Percent of good time in a period
Range	Range of minimum to maximum value
StDev	Time-weighted standard deviation
TagAvg	Time-weighted average
TagMean	Event-weighted average
TagMax	Maximum value in a period
TagMin	Minimum value in a period
TagTot	Time integral over a period

Point Attributes

Name	Description
TagDesc	Get a point's descriptor
TagEU	Get a point's engineering unit string
TagExDesc	Get a point's extended descriptor
TagName	Get a point's name
TagNum	Get a point's ID
TagSource	Get a point's point source string
TagSpan	Get a point's span
TagType	Get a point's type character
TagTypVal	Get a point's typical value
TagZero	Get a point’s zero value

Time Functions

Name	Description
Bod	Timestamp for beginning of the day for given time
Bom	Timestamp for beginning of the month for given time

Bonm	Timestamp for first of the next month for given time
Day	Day of the month from a time
DaySec	Seconds since midnight from time
Hour	Hour from a time
Minute	Minute from a times
Month	Month from a time
Noon	Timestamp for local noon of day of a times
ParseTime	Convert character string to time
Second	Second from a times
Weekday	Day of the week from a times
Year	Year from a time
Yearday	Day of the year from a time

String Functions

Name	**Description**
Ascii	ASCII character code for a character
Char	String for ASCII character code(s)
Compare	Wild comparison of two strings

DigText	Text for a digital state
Format	Formatting of a numerical number
InStr	Instance of a sub-string
LCase	Conversion of all characters to lower case
Len	Length of a string
Left	First characters in a string
LTrim	Removal of blanks on the left side of a string
Mid	Extraction of a sub-string from a string
Right	Last characters in a string
RTrim	Removal of blanks on the right side of a string
Trim	Removal of blanks on both sides of a string
UCase	Conversion of all characters to upper case

String Conversion

Name	**Description**
Concat	Concatenate two or more strings
String	String representing any PI value

Text	Concatenation of strings for a series of PI value arguments

15 Appendix C PI SQL Commander Table Relationships