BI Reporting with PI Integrator for Business Analytics Version 2020 R2

OSIsoft, LLC
1600 Alvarado Street
San Leandro, CA 94577

Software Versions Used in this Document

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

Software	Version
PI DataLink	2019 SP1
Microsoft Office	2016
PI ODBC Driver	2016 R2
PI SQL Client	2018 R2
PI Integrator for Business Analytics Advanced Edition	2018 R2 SP1
PI OLEDB Enterprise	2019
Microsoft SQL Server	2016
PI Data Archive	2018 SP3
PI Asset Framework	2018 SP3
PI Vision	2019

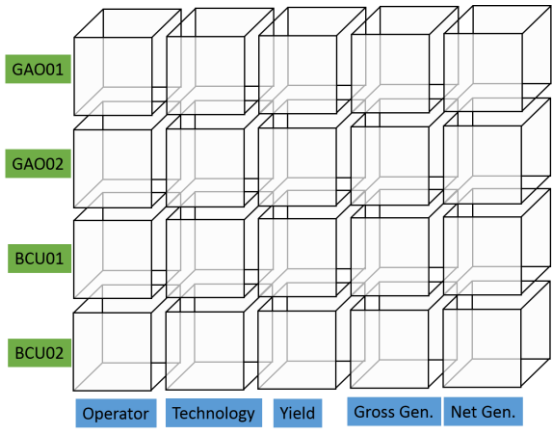
Contents

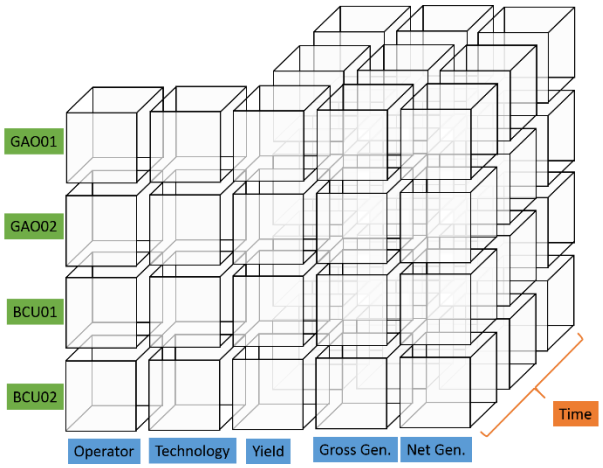
Lesson 1: Business Intelligence and PI Asset Framework

Business intelligence (BI) tools offer solutions to quickly analyze raw, un-normalized, multidimensional data. Values from the PI Data Archive, external metadata, and calculations from Asset Framework can be transformed by business intelligence tools into actionable analysis and interactive reports in order to gain insight into business and operational processes.

One of the key concepts of the course is how PI AF metadata and hierarchy information inherently provides the relationships and dimensions used to filter and slice data in BI tools.

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





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.

Introduction 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 (MS Excel is limited to 1 million rows)
- Cheap – Free download or $9.99 / month per user for Power BI Pro
- Live reporting and centralized web-based dashboards in Office 365 and Power BI Server
- Slick visuals including 3rd Party Visuals in Microsoft AppSource

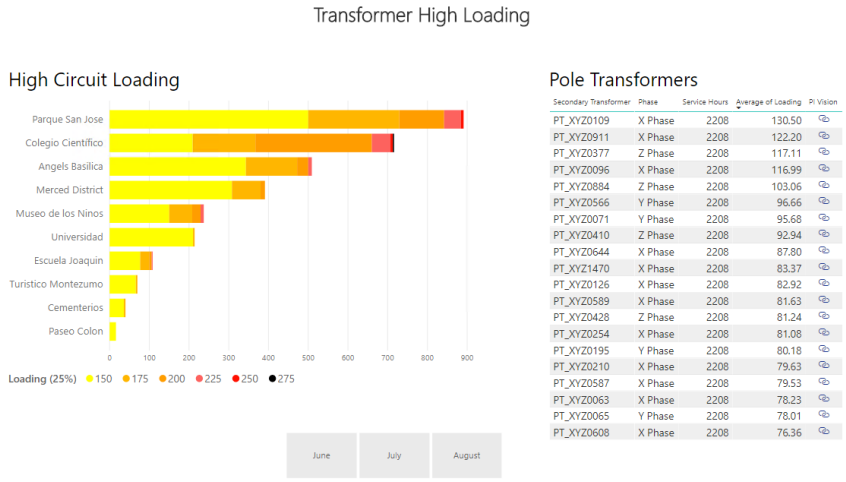
Activity – Inspect a Sample Power BI Report

Objective:

- Explore a sample Power BI Report.

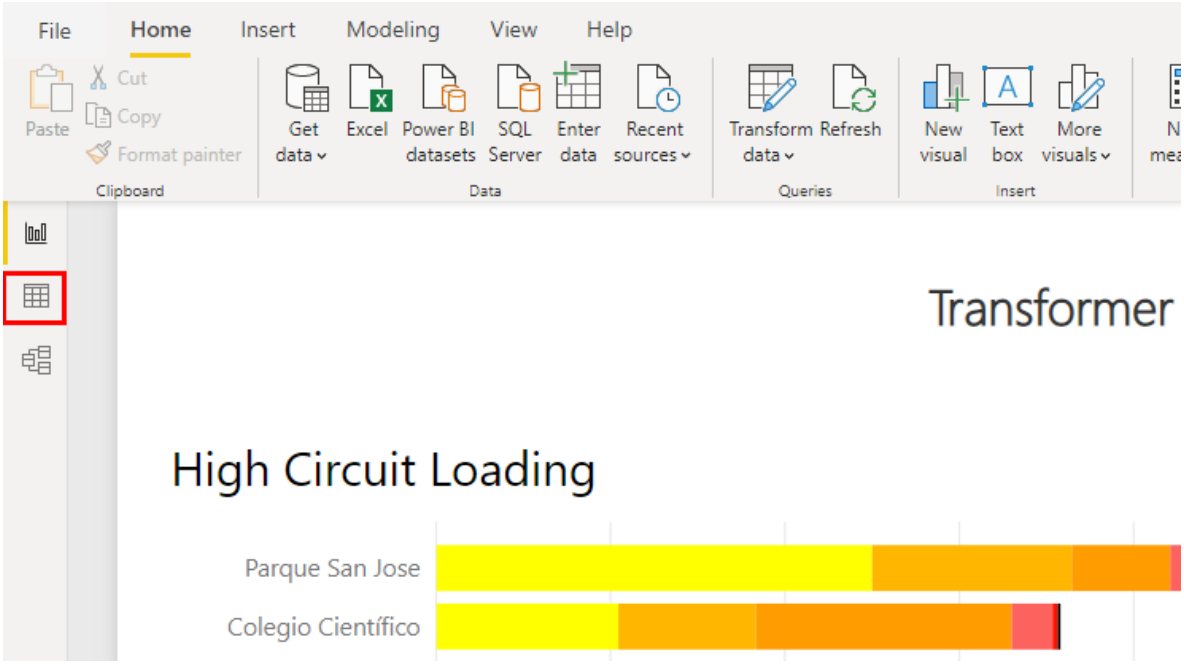
Approach

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

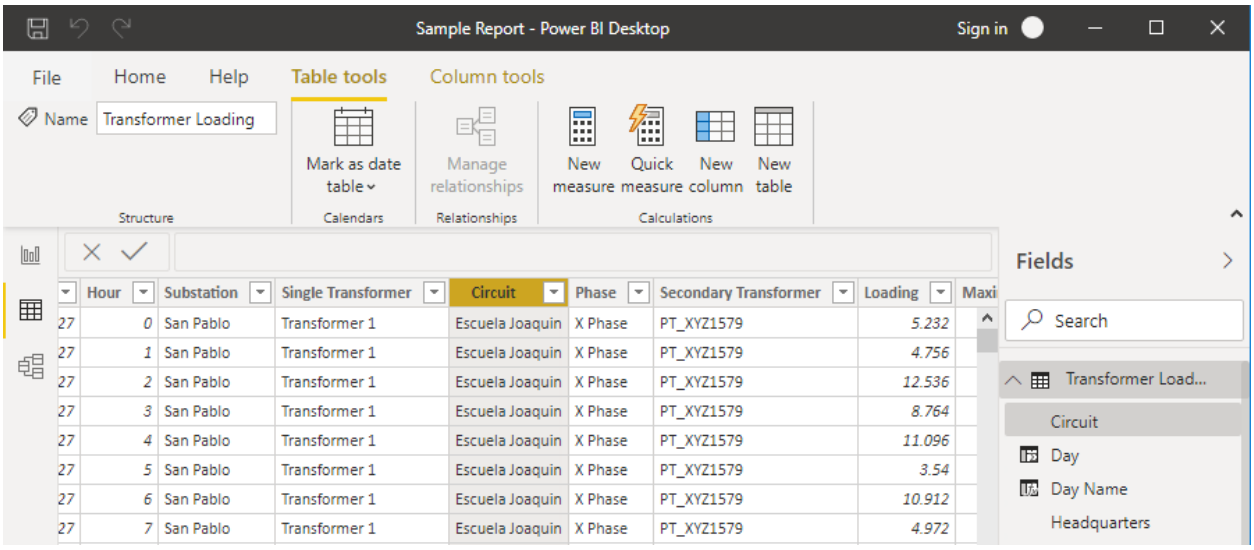


Right now we're looking at the **Report View** where the report and visuals are configured. Start clicking on the visuals and the rest of the report will be filtered to only include the selected items. This is often referred to as **slicing and dicing** the data.

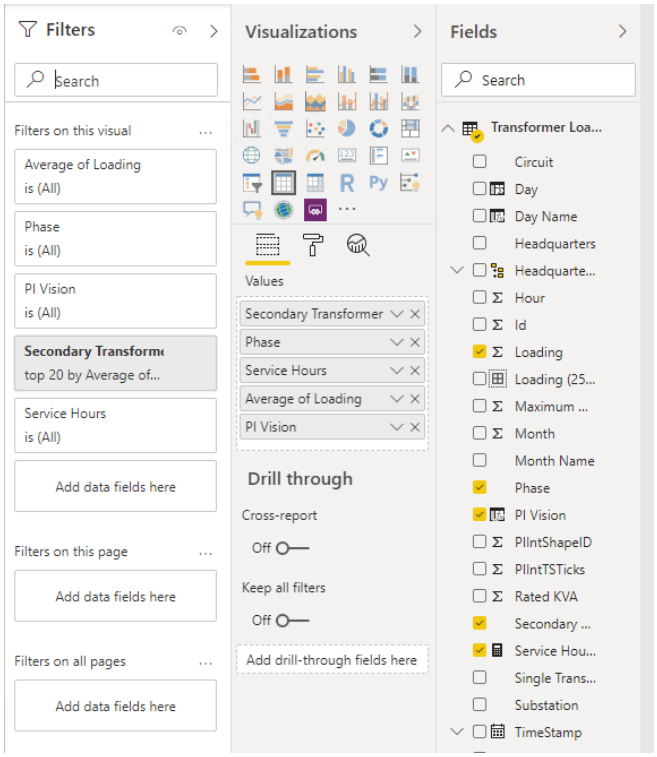
Click on the **Data Tab** to inspect the data set we'll be working with:



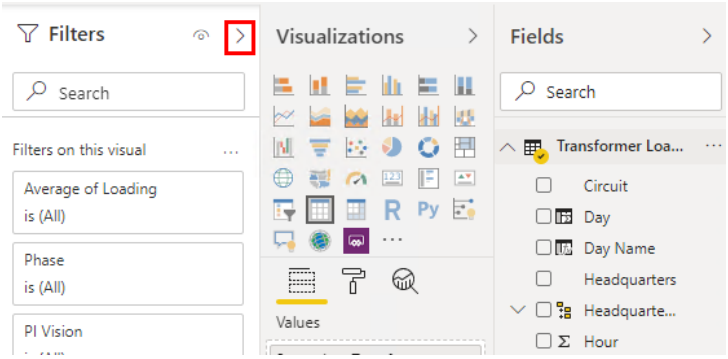
Note that all the columns are available in the **Fields List**:



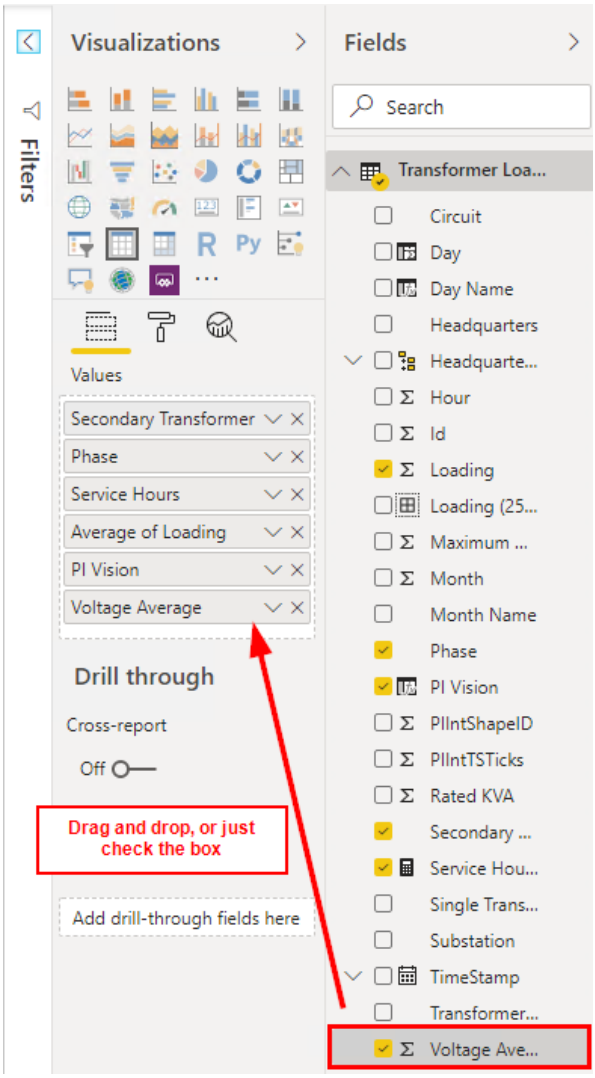
Go back to the Report View, click on the **Pole Transformers Table** visual, and note the **Filters Pane**, **Visualizations Pane,** and **Fields Pane**. These sections are where the bulk of the configuration takes place. Columns from the data set are dragged and dropped from the Fields Pane onto the 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.



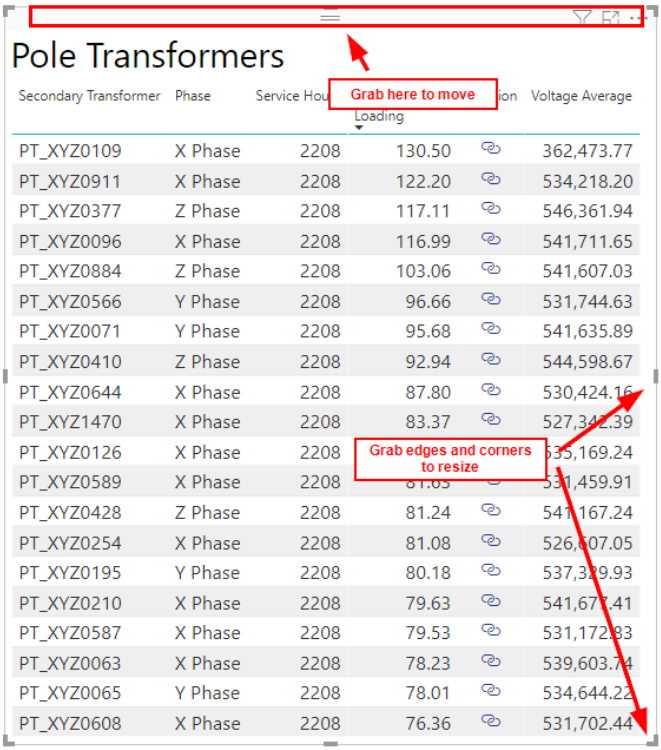
Minimize the Filters Pane for now.



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

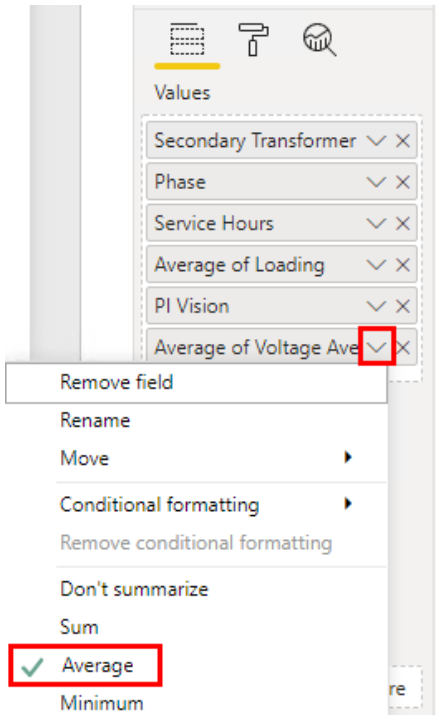


Resize and/or reposition the visual so that the Voltage Average column is visible.

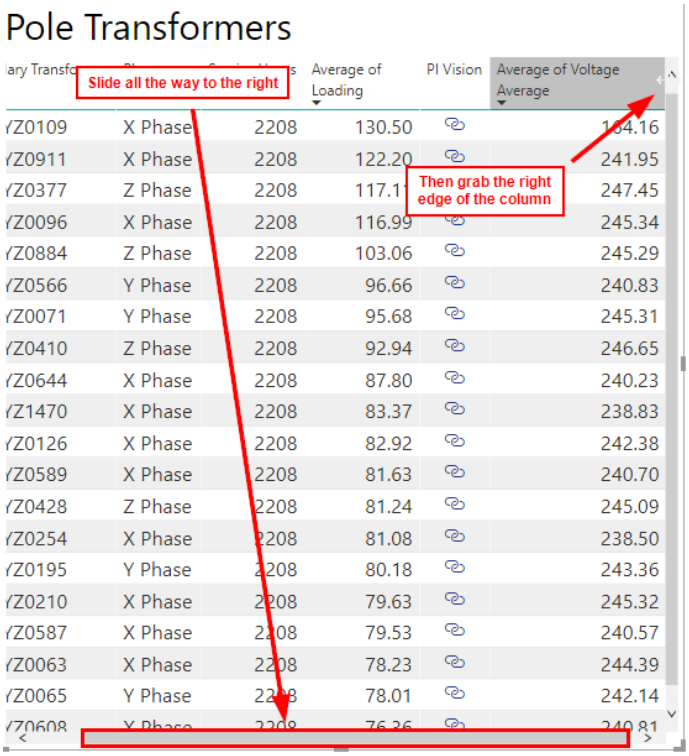


Secondary Transformer	Phase	Service Hou	Loading	ion	Voltage Average
PT_XYZ0109	X Phase	2208	130.50		362,473.77
PT_XYZ0911	X Phase	2208	122.20		534,218.20
PT_XYZ0377	Z Phase	2208	117.11		546,361.94
PT_XYZ0096	X Phase	2208	116.99		541,711.65
PT_XYZ0884	Z Phase	2208	103.06		541,607.03
PT_XYZ0566	Y Phase	2208	96.66		531,744.63
PT_XYZ0071	Y Phase	2208	95.68		541,635.89
PT_XYZ0410	Z Phase	2208	92.94		544,598.67
PT_XYZ0644	X Phase	2208	87.80		530,424.16
PT_XYZ1470	X Phase	2208	83.37		527,342.39
PT_XYZ0126	X Phase	2208			535,169.24
PT_XYZ0589	X Phase	2208			531,459.91
PT_XYZ0428	Z Phase	2208	81.24		541,167.24
PT_XYZ0254	X Phase	2208	81.08		526,507.05
PT_XYZ0195	Y Phase	2208	80.18		537,329.93
PT_XYZ0210	X Phase	2208	79.63		541,677.41
PT_XYZ0587	X Phase	2208	79.53		531,172.83
PT_XYZ0063	X Phase	2208	78.23		539,603.74
PT_XYZ0065	Y Phase	2208	78.01		534,644.22
PT_XYZ0608	X Phase	2208	76.36		531,702.44

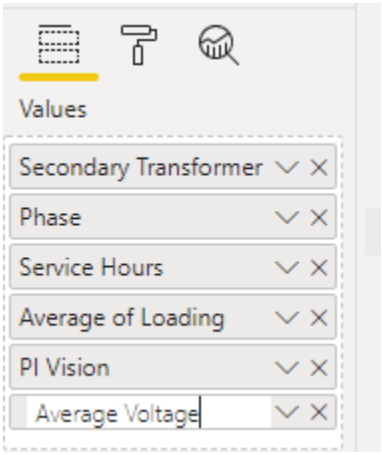
The Voltage Average will be displayed in the table, but by default all the voltages from all the rows will be summed by transformer. It makes more sense to summarize these as an average, so click the drop down and change the summary to **Average**:



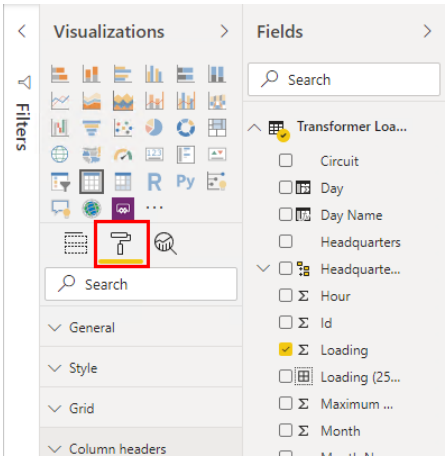
Resize the column directly on the visual similar to Excel:



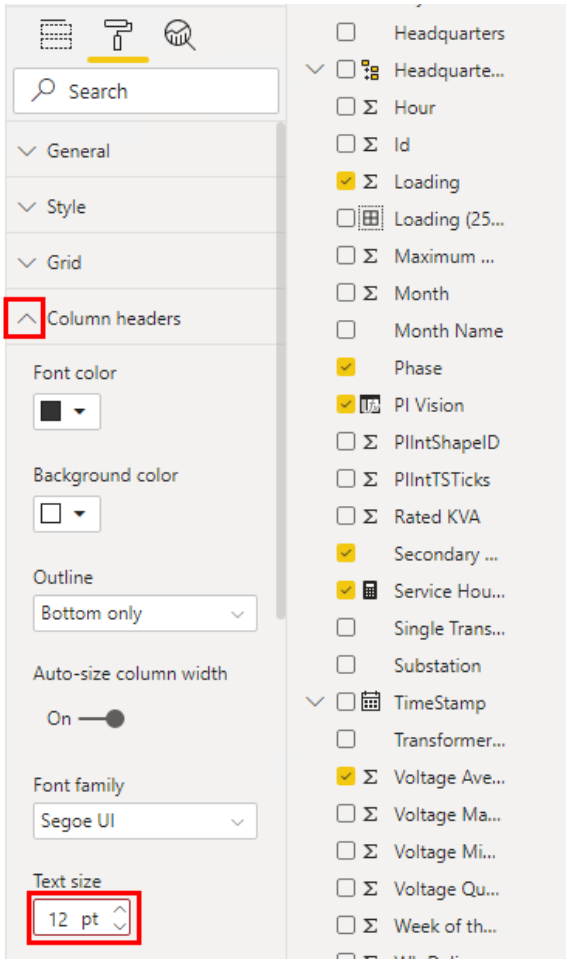
Average of Voltage Average is a pretty weird header name, so double click it and **rename it to Average Voltage**:



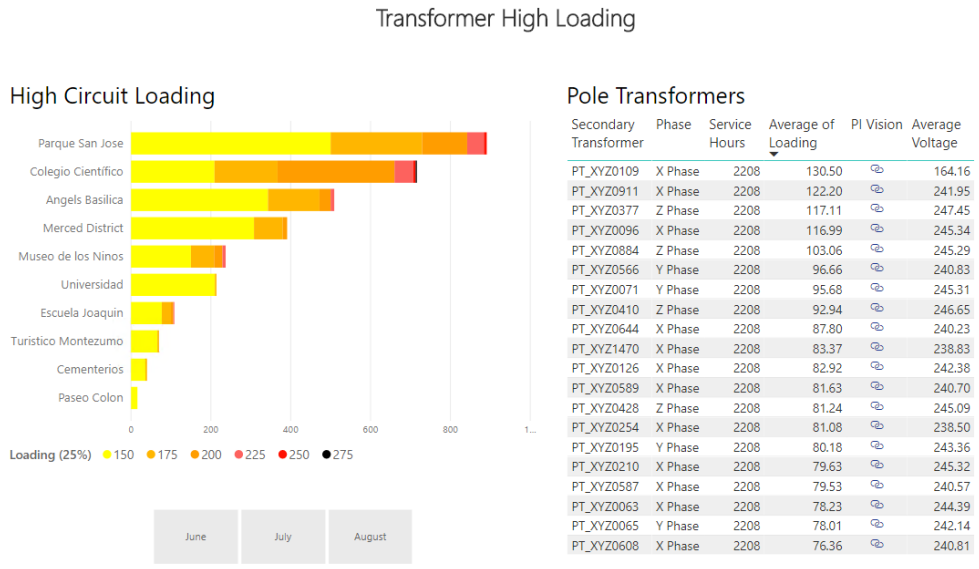
Other **Formatting Options** are available by clicking the paint roller icon:



We'll go over more formatting options in the exercises, so for now just bump up the text size of the column headers to 12:



Adjust the column widths and reposition the visuals to make everything fit. Your report should now look something like this.



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

Lesson 2: Power BI Reports using PI Integrator for BA

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 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.

PI AF Hierarchy and Data Set

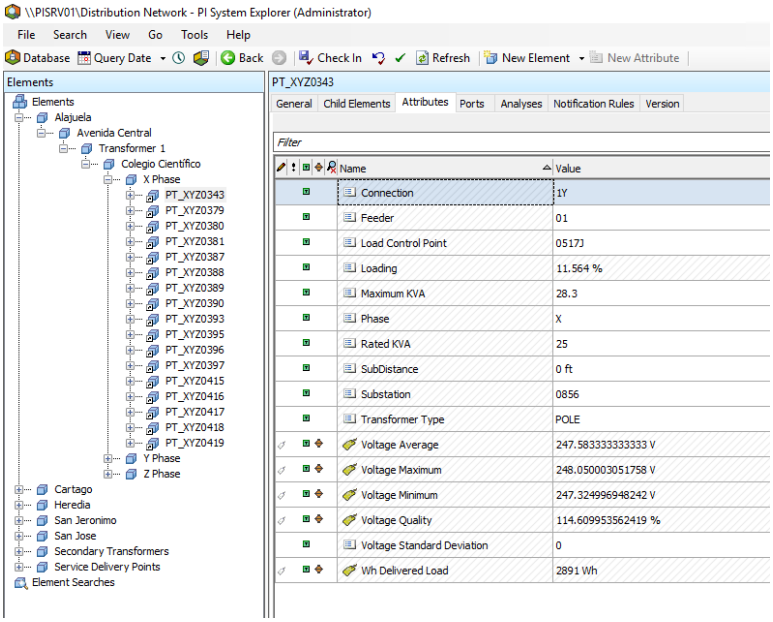
Objective:

- 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.

Lesson 3: PI Integrator for Business Analytics

In simple terms, PI Integrator for Business Analytics reads data from PI Asset Framework and writes the data to a variety of 3rd-party platforms and databases referred to as Targets. The most often used target is a Microsoft SQL Server database.

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 System Access software. This chapter will discuss the former method of extracting the data.

The PI Integrators join your Business Intelligence (BI) infrastructure with OSIsoft's PI System, allowing you to combine high-value Operation Technology (OT) data from the PI System with Information Technology (IT) data for reporting, analytics, and application integrations. The integration of data from OT systems, such as automation and control systems and internet-enabled devices, with data from IT systems, such as transactional and business process systems, increases situational awareness, adds transparency into industrial operations and business processes, and makes it possible to anticipate problems and identify opportunities for process improvements.

Product Information and Features

The section outlines general product, feature, and component information. The latest documentation can be found in the PI Live Library.

Advanced vs Standard Edition

PI Integrator for Business Analytics Advanced Edition serves real-time packets of PI System data to streaming platforms such as Apache Kafka. Streaming platforms assist in operationalizing machine learning models and support Kappa and Lamda architectures for data consumption.

In short, Advanced Edition supports streaming views while Standard Edition does not.

PI Asset Framework Requirement

PI Integrator for Business Analytics requires a PI Asset Framework (PI AF) model to select PI System data to produce decision-ready data. The data can be cleansed using a variety of filters and enhanced with asset, event, and time context from the PI System.

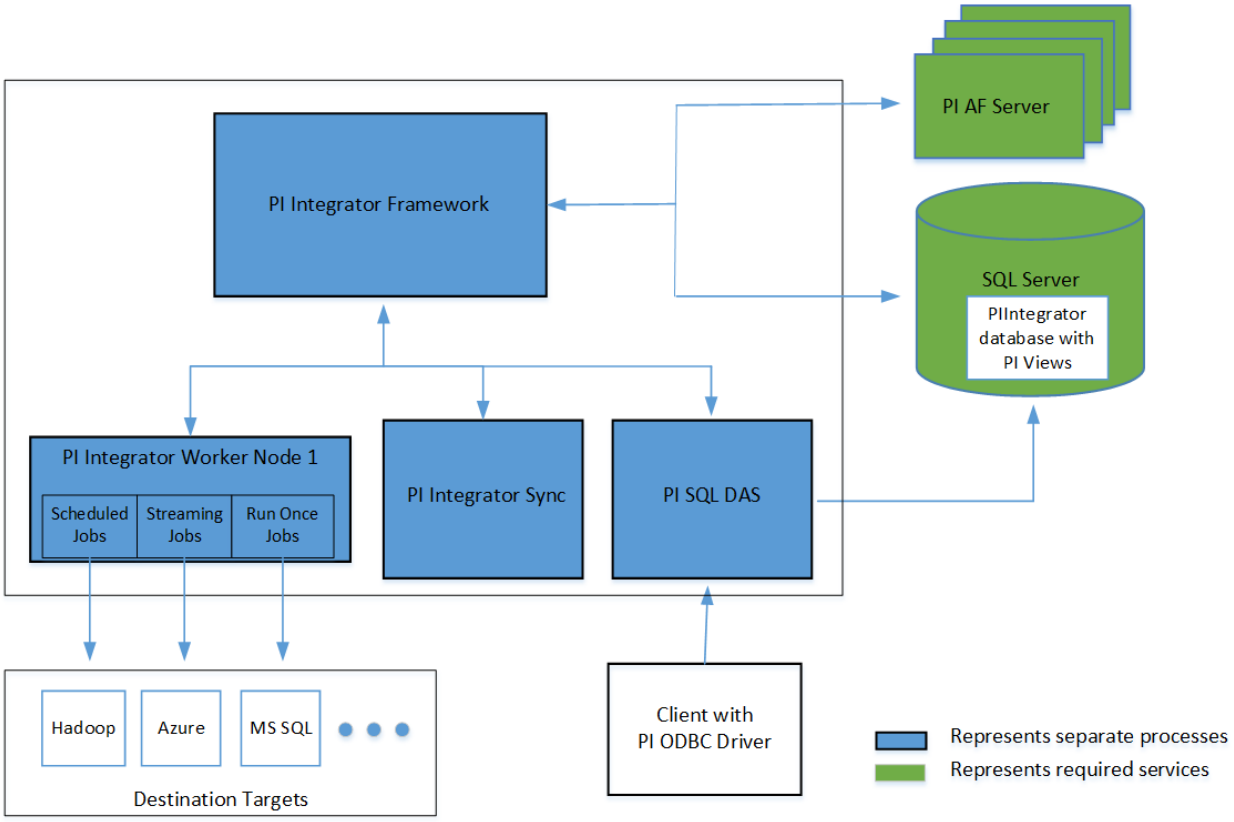
PI Tags cannot be used directly. They must be mapped in PI AF in order to be exposed with PI Integrator for BA.

Licensing by Streams

PI Integrator for Business Analytics is licensed by stream count. A stream is a unique instance of a PI Tag used in one of the PI Integrator views. If the same tag is used in multiple views it only counts as one stream. Publishing static attributes does not consume streams.

General Software Architecture

Consult the PI Live Library for the latest documentation. The system architecture for PI Integrator for BA 2020 from the online documentation is shown below:



The PI Integrator for BA architecture can be confusing because it typically has components on 3 separate servers and that's not even counting Targets:

- All the roles in blue always exist on a single server and represent a set of Windows Services and the Web Page used to configure and administer PI Integrator for BA.
- PI Integrator for BA has backend components in a PI AF Configuration database which are separate from the elements and attributes used as data sources.

- PI Integrator for BA has 3 backend databases on a Microsoft SQL Server: PIIntegratorDB, PIIntegratorLogs, and PIIntegratorStats. These are separate from any configured SQL Server targets.

Targets

As of the PI Integrator for BA 2020 release, the following Targets are supported by the Advanced Edition:

- Amazon Kinesis Data Streams
- Amazon Redshift
- Amazon S3
- Apache Hive
- Apache Kafka
- Azure Data Lake Storage Gen 1
- Azure Event Hubs
- Azure IoT Hub
- Azure SQL Database
- Azure SQL Data Warehouse
- Google BigQuery
- Google Cloud Storage
- Google Cloud Pub/Sub
- Hadoop Distributed File System (HDFS)
- Microsoft SQL Server
- Oracle
- SAP® HANA® ODBC
- Text file
- PI View (configured out of the box)

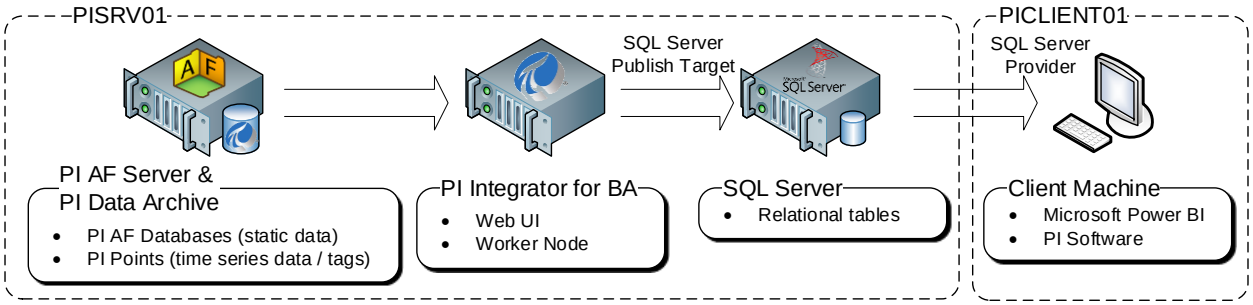
In this course we will only configure PI Integrator for BA to publish data to the Microsoft SQL Server target.

The PI View Target

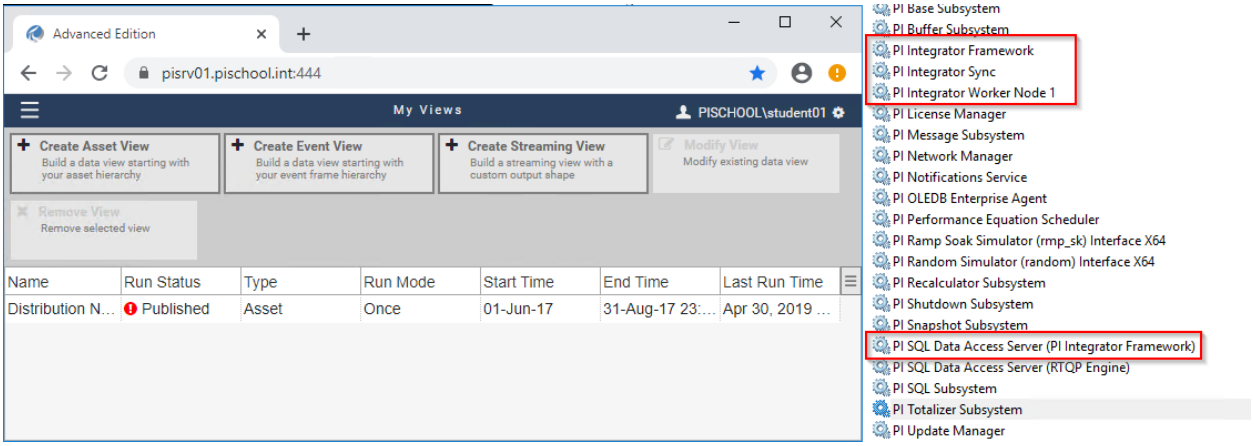
The PI View target is the only target that is configured out of the box. Technically the view data is stored in a SQL Server since it exists in the PIIntegratorDB database, but the supported access method is to use the PI ODBC Driver on the client querying the PI View. PI Views mainly exist for convenience and having a readily available Target when the installation completes. It is intended for testing and development sandbox scenarios only and should not be used if a Microsoft SQL Server target is available.

Architecture Used in Class

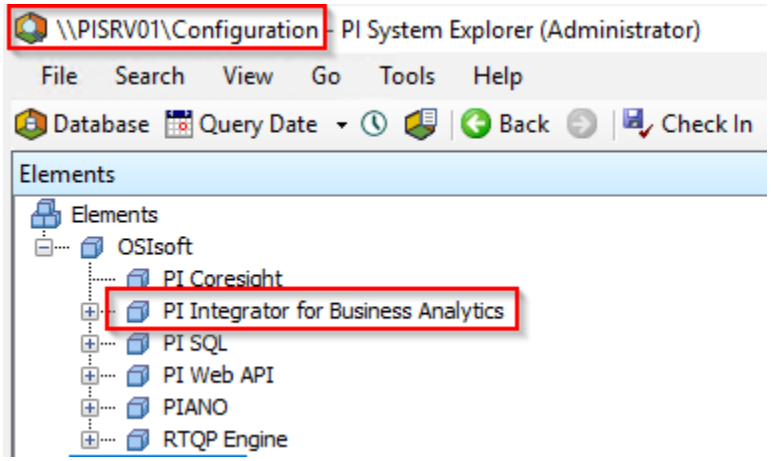
In this course, all server roles including PI Data Archive, PI AF Server, SQL Server, and PI Integrator for BA are all installed on PISRV01. In a production grade architecture, each role would typically have its own dedicated server.



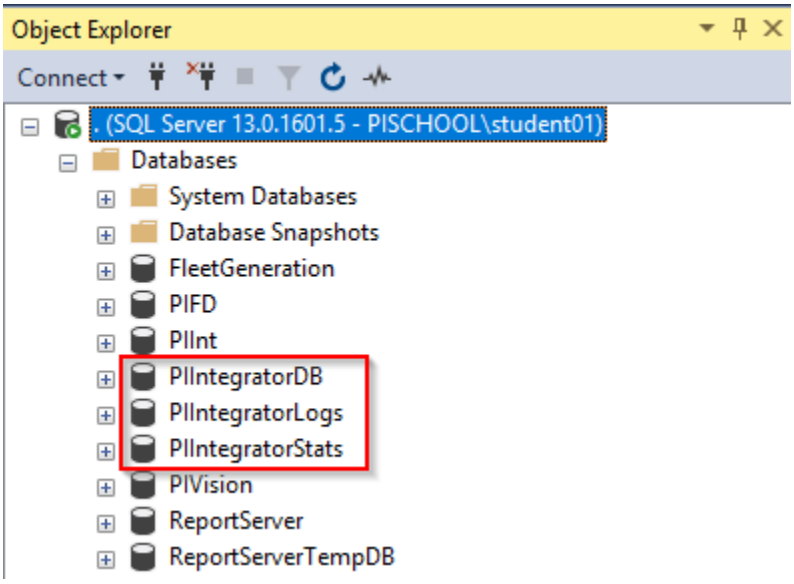
The PI Integrator for BA Web Page and Windows Services on PISRV01 are:



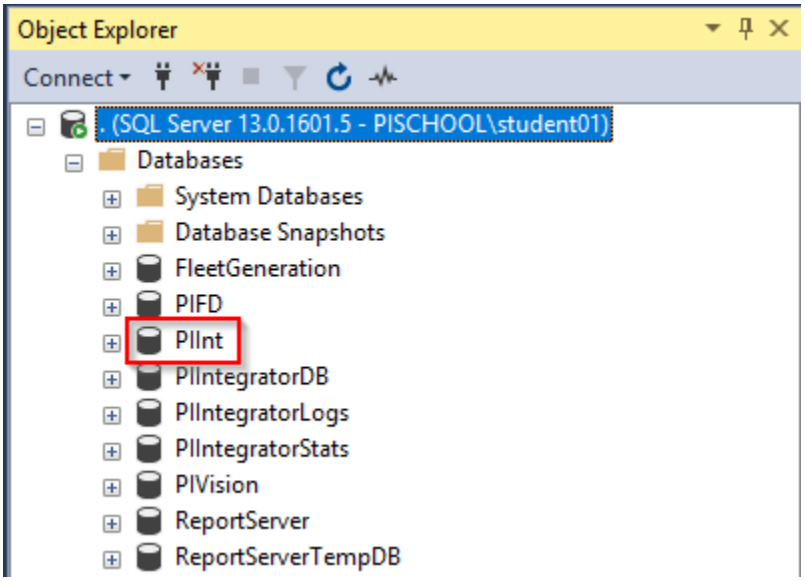
The PI Integrator for BA Configuration database element is:



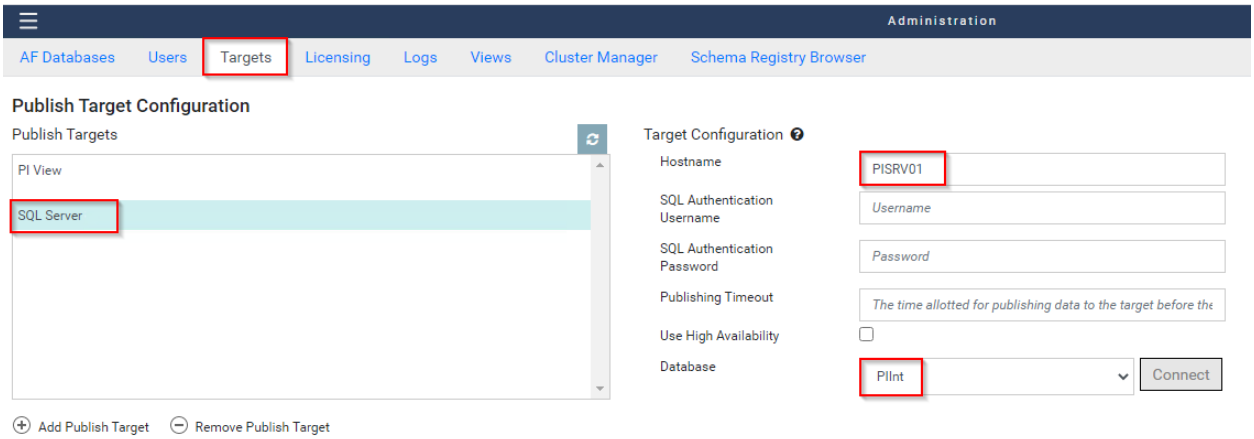
The backend SQL tables are:



The SQL Server Target database is:



The SQL Server target configuration is in the Target section of the Administration page:



In this course we will only configure PI Integrator for BA to publish data to a SQL Server and then use the native SQL Server provider to import the data into Microsoft Power BI.

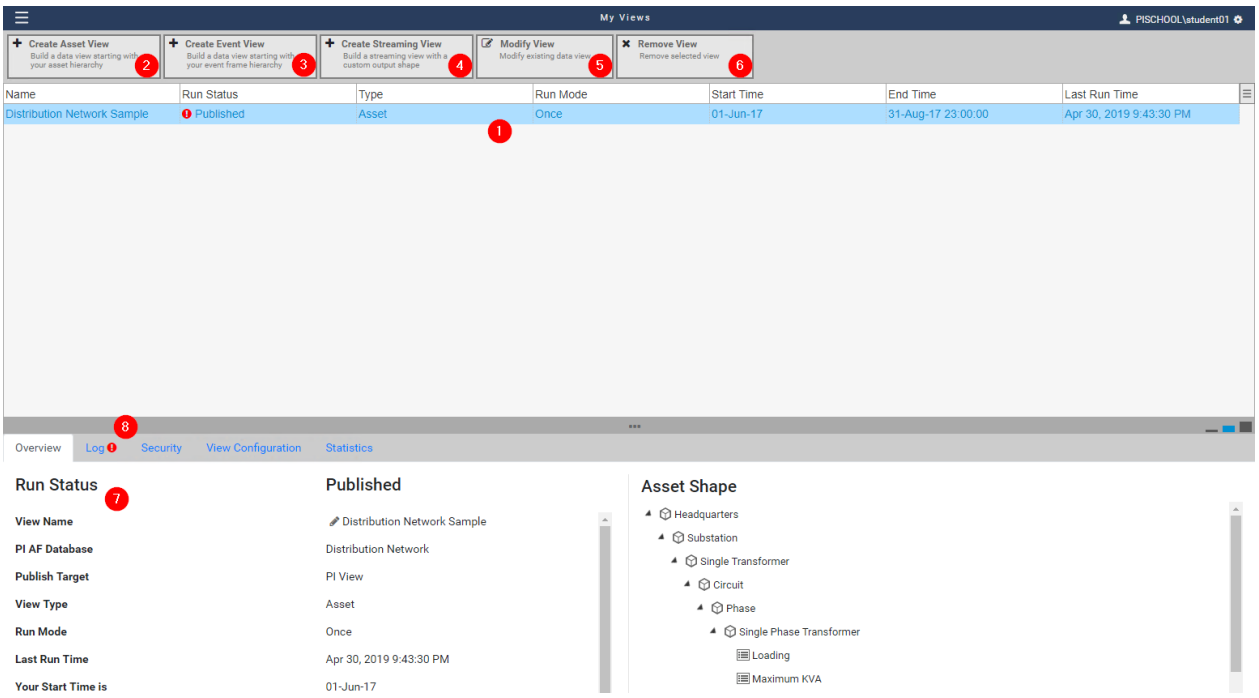
PI Integrator Web UI

The PI Integrator for Business Analytics site can be accessed via https://pisrv01.pischool.int:444 or from the desktop. If prompted for credentials, enter your student account, as this has been given access rights.

Views can be created within the PI Integrator portal that is hosted on PISRV01.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** for publish targets that support streaming such as Apache Kafka, Azure Event Hub, and Azure IoT Hub.
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 additional details about the view.
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.

Activity – Create the Transformer Loading View

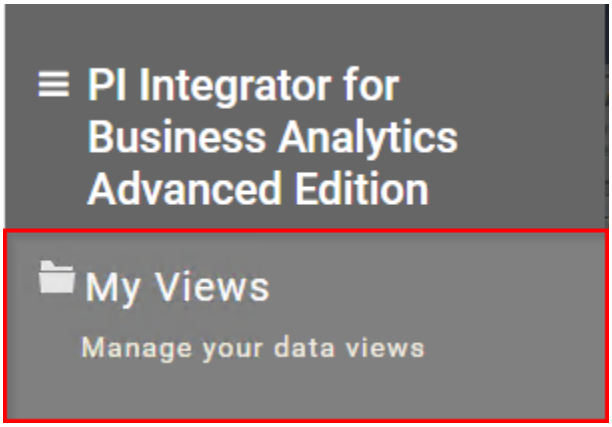
Objective:

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

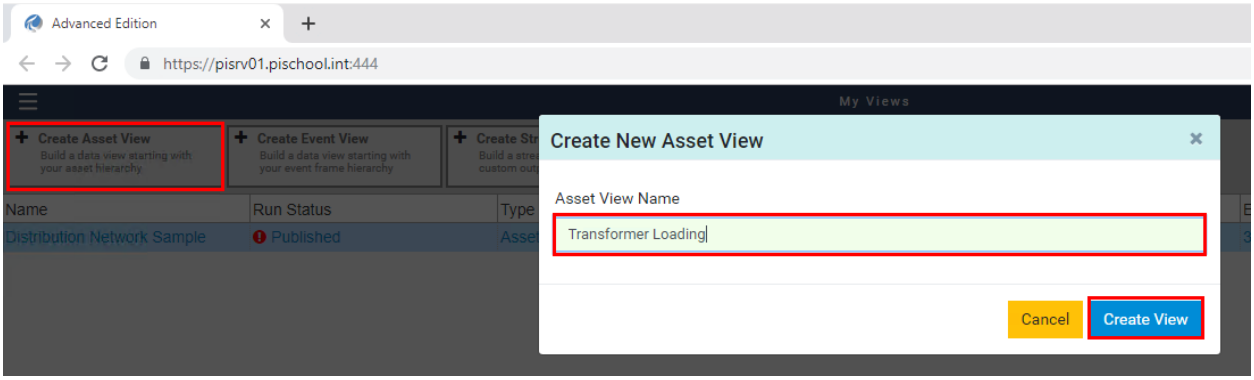
Approach:

Open Google Chrome and Navigate to the PI Integrator for BA Web UI at https://pisrv01.pischool.int:444

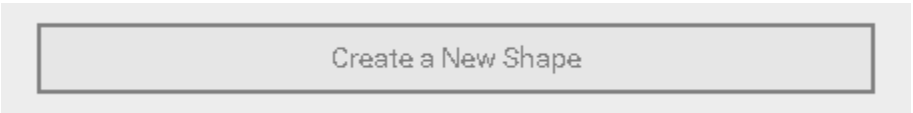
Go back to My Views:



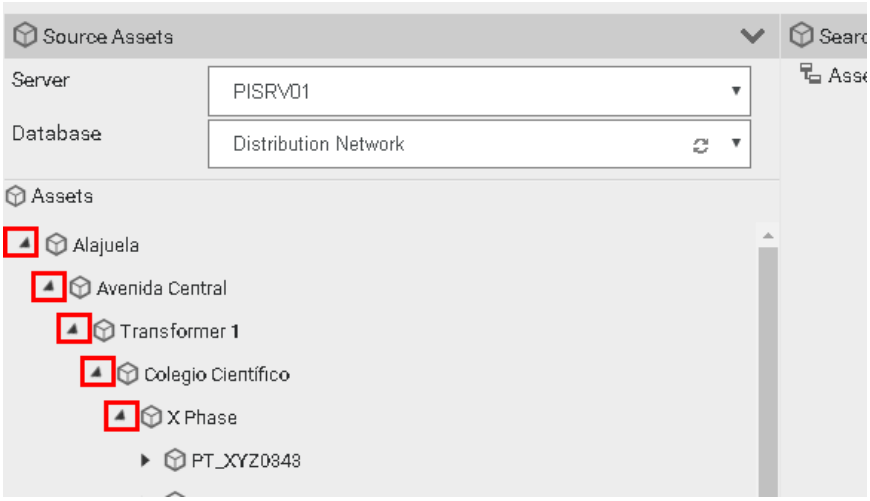
Click **Create Asset View** and name it Transformer Loading, click **Create View**:



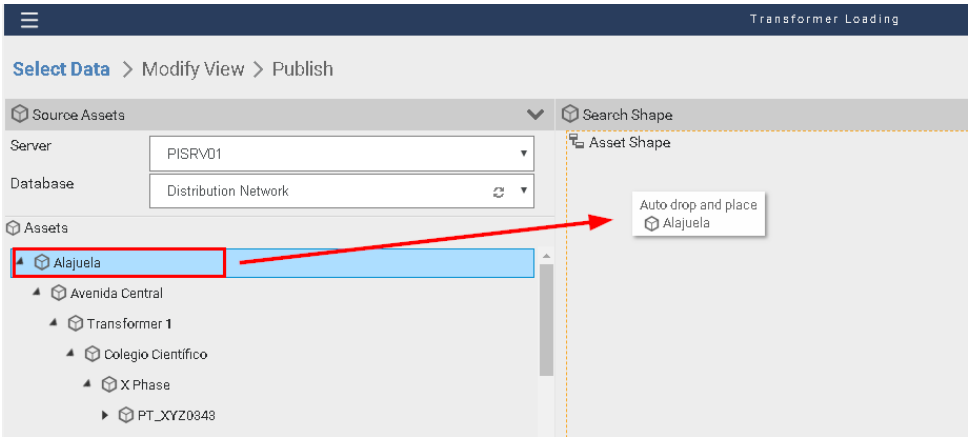
Click **Create a New Shape**



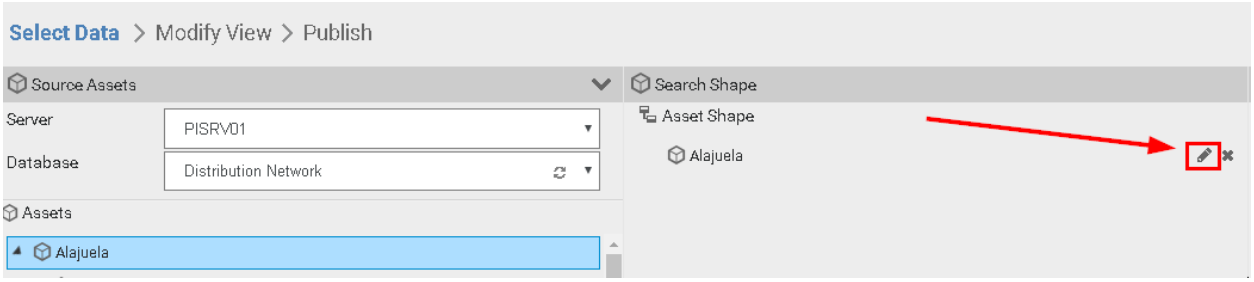
Select Distribution Network as the AF Database, then drill down to PT_XYZ0343.



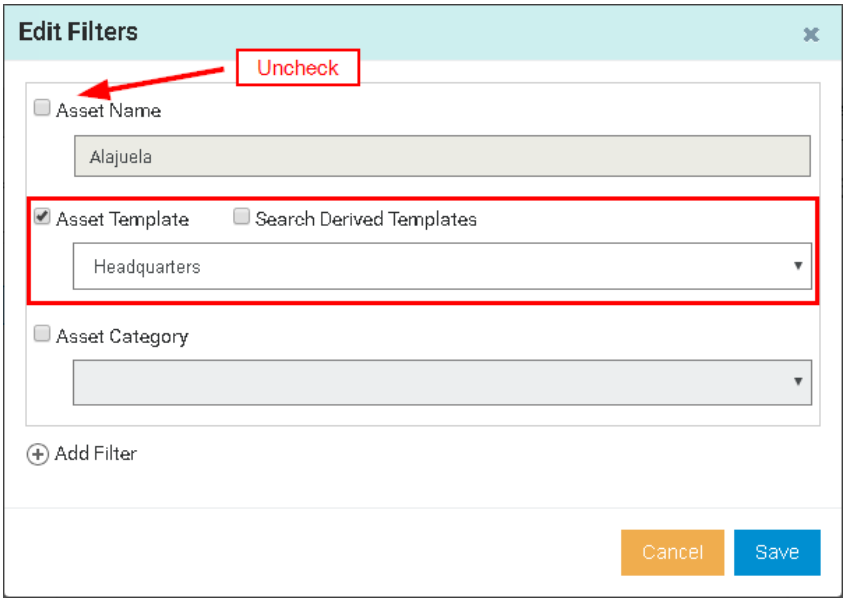
Drag and drop Alajuela to the Shape Builder



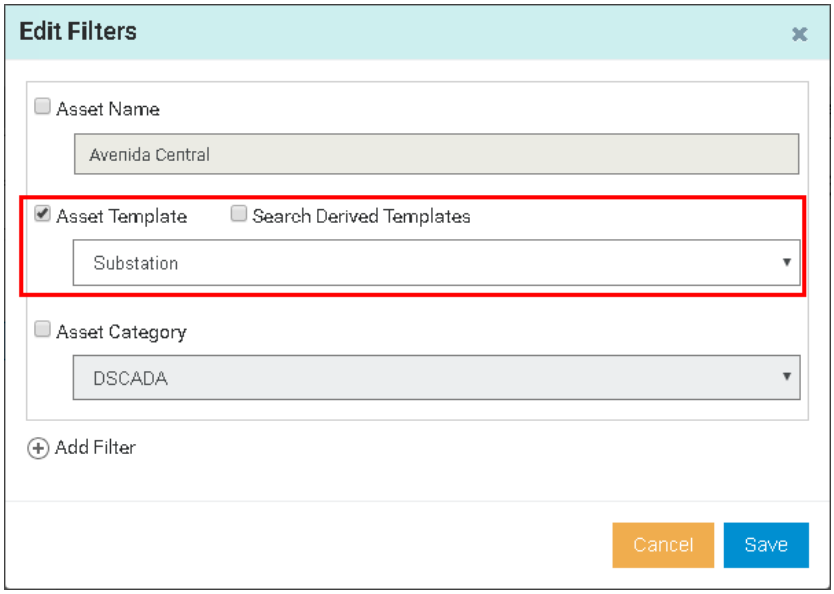
Edit the Filter on Alajuela:



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

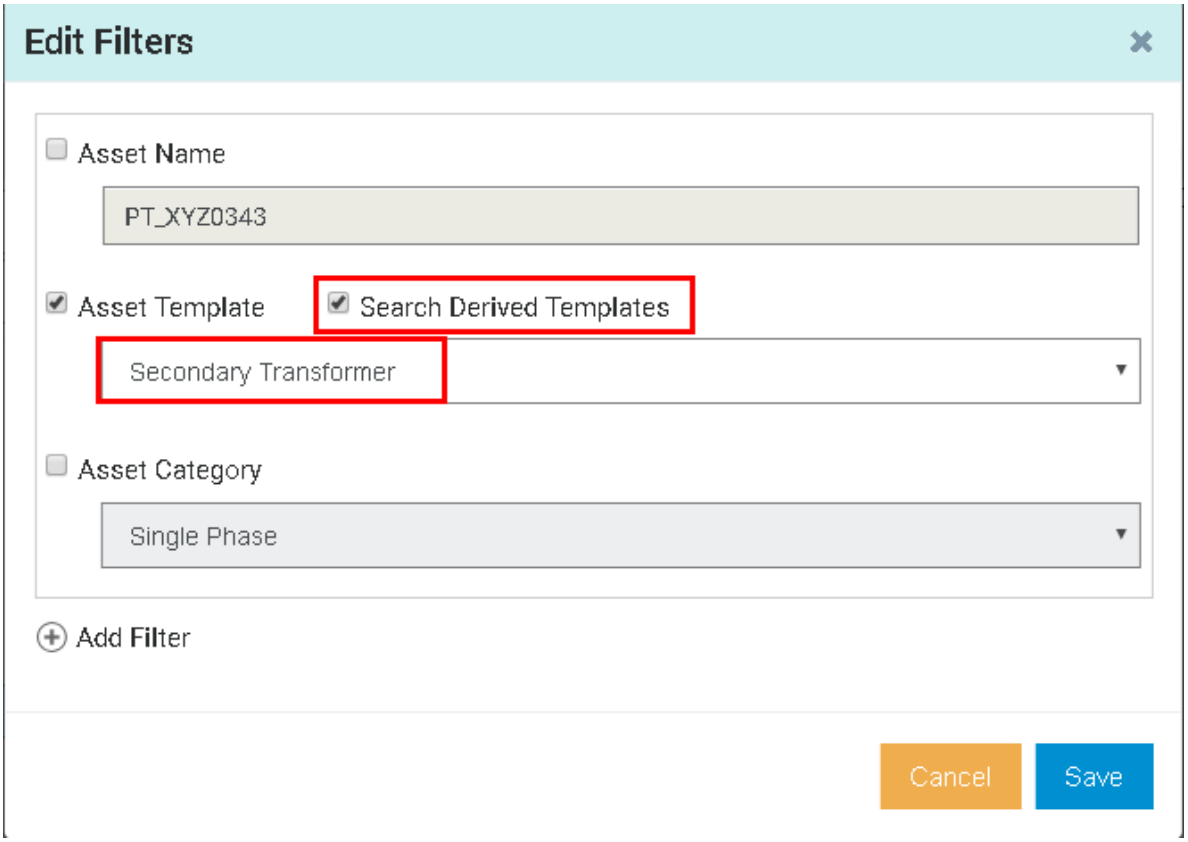


Drag and drop **Avenida Central** to the Shape configuration, and change it to filter on the **Substation** Template:



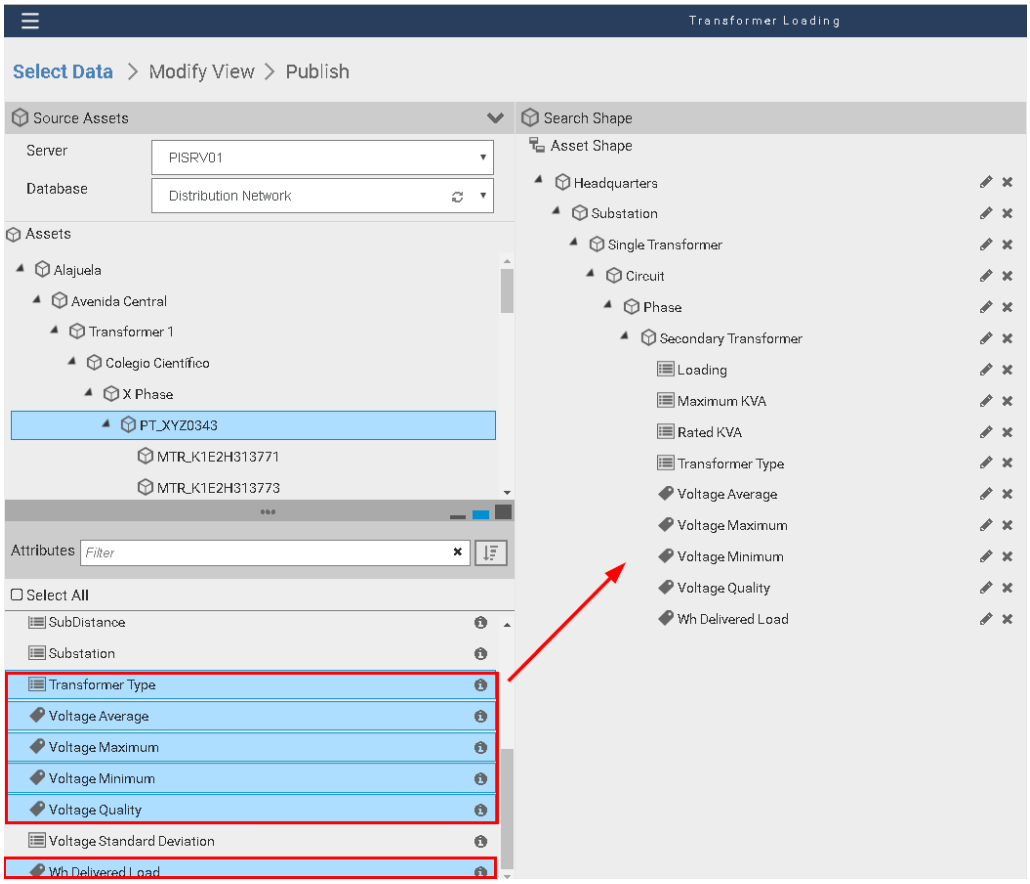
Repeat this pattern for **Transformer 1** (Template = Single Transformer), **Colegio Cientifico** (Template = Circuit), **X Phase** (Template = Phase).

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:

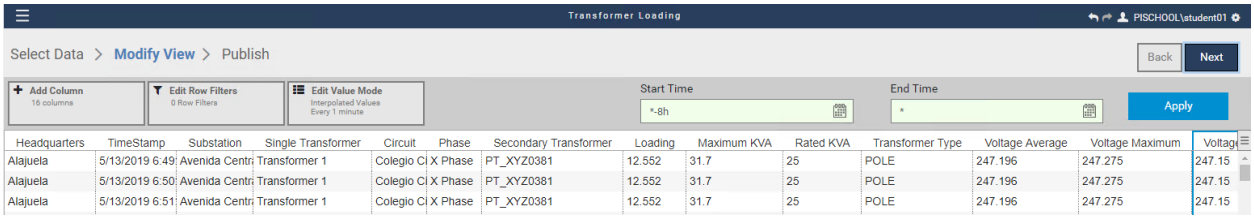
Click **PT_XYZ0343** then **hold control** and multi-select **Loading, Maximum KVA, Rated KVA, Transformer Type, Voltage Average, Voltage Maximum, Voltage Minimum, Voltage Quality,** and **Wh Delivered Load**. Drag and drop these selections to the Shape configuration.



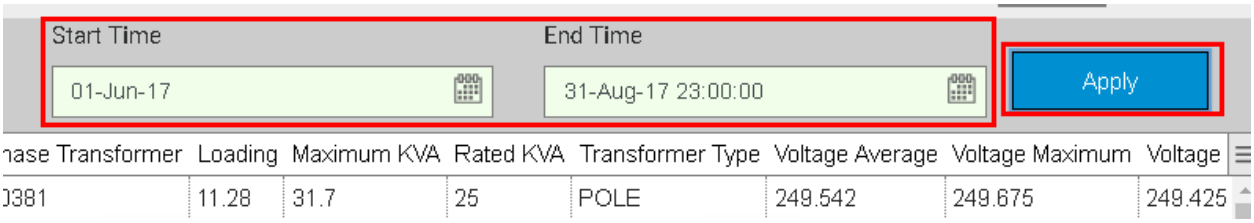
Note that all Transformers share these common attributes because they all use the same template.

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

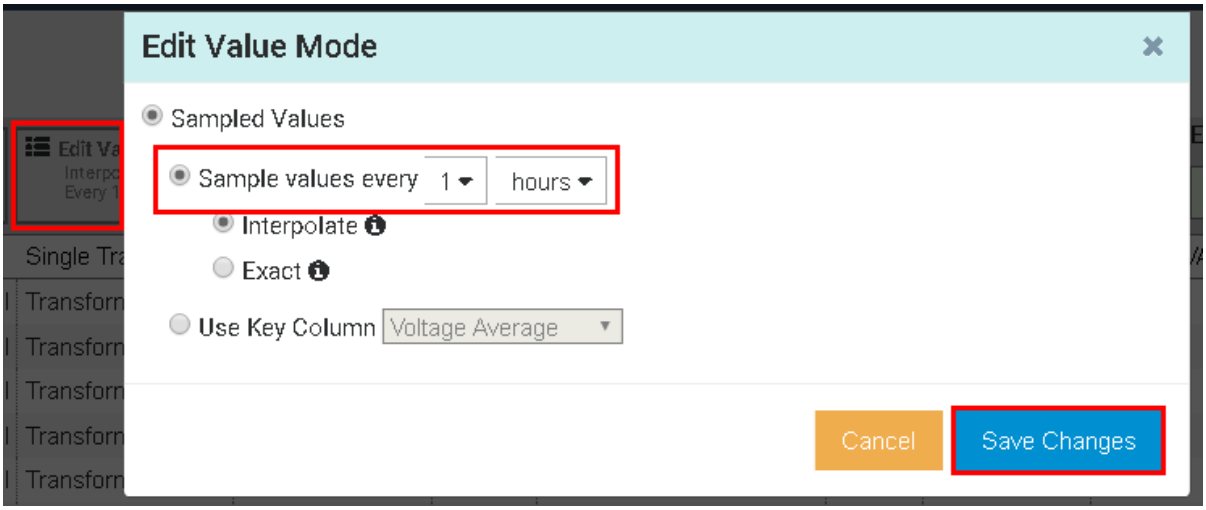
We now see a preview of the data using the default Time Range and interpolation mode. Note that each AF attribute is a column in the preview. PI AF templates inherently provide the table format preferred by BI Tools when analyzing a number of similar assets.



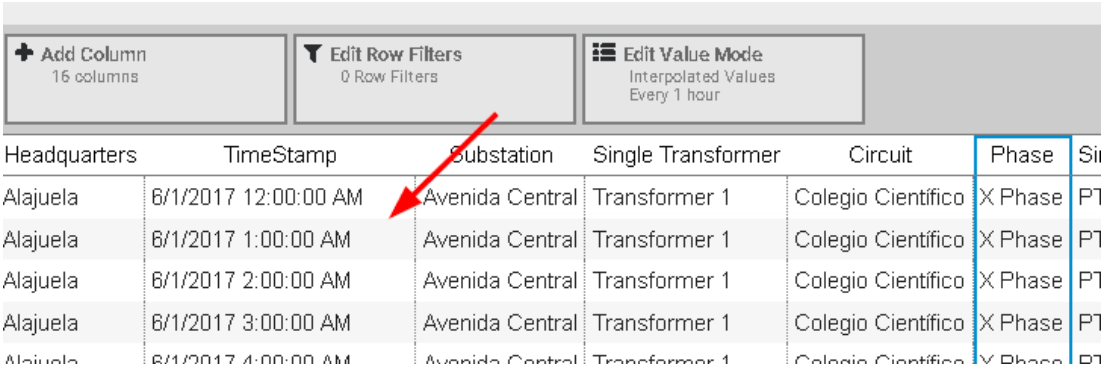
We want to publish Hourly data for the time period 01-Jun-17 00:00:00 to 31-Aug-17 23:00:00. Modify the Start Time and End Time and click Apply:



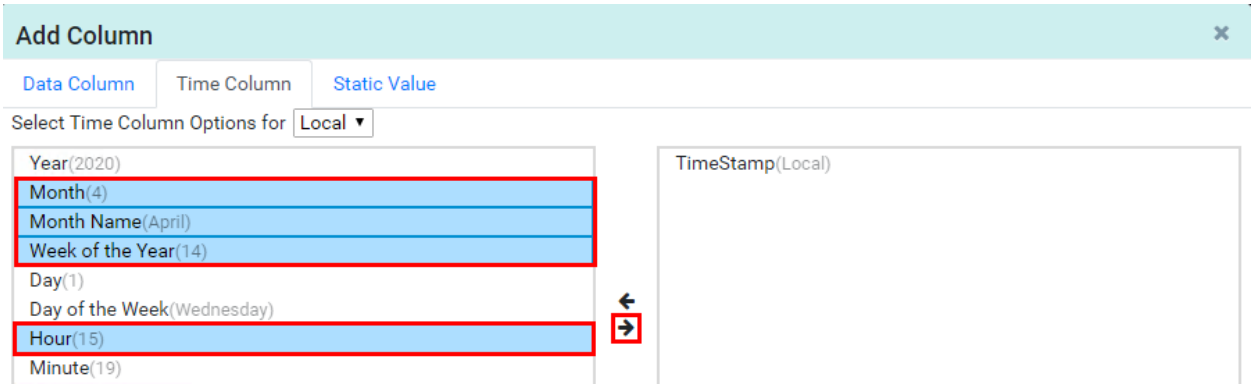
Click **Edit Value Mode** and change the time step to 1 hour, then **Save Changes**:



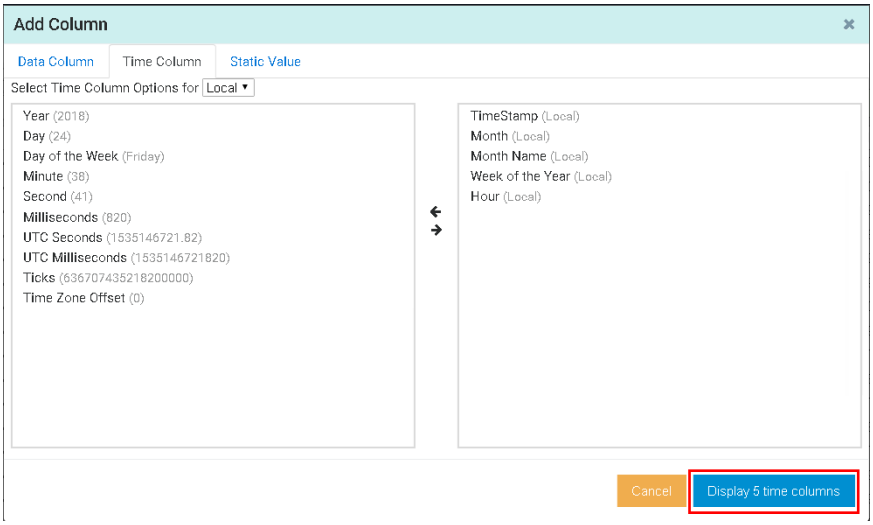
The TimeStamp column should now reflect changes to the Start, End, and Value Mode:



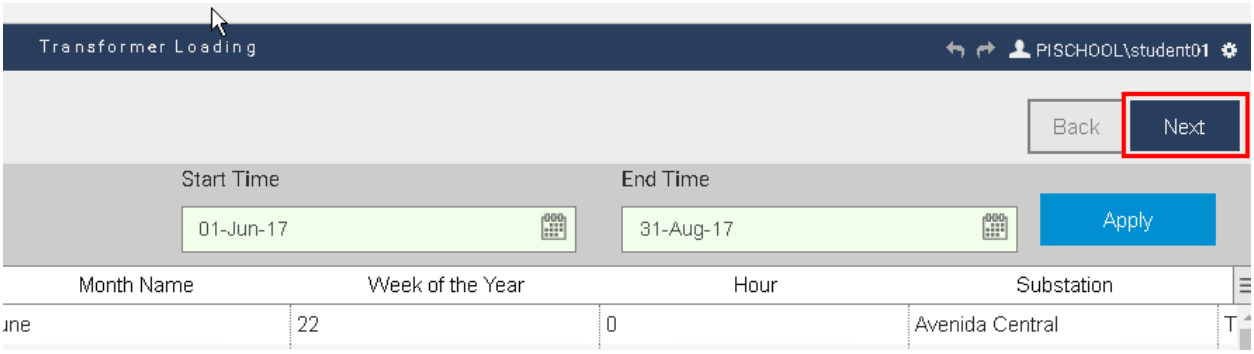
Now we'll add some additional time columns that will come in handy later when building the reports. **Click Add Colum**. Select the **Time Column** tab. Select Month, Month Name, Week of the Year, and Hour, then click the arrow to bump them over to the right:



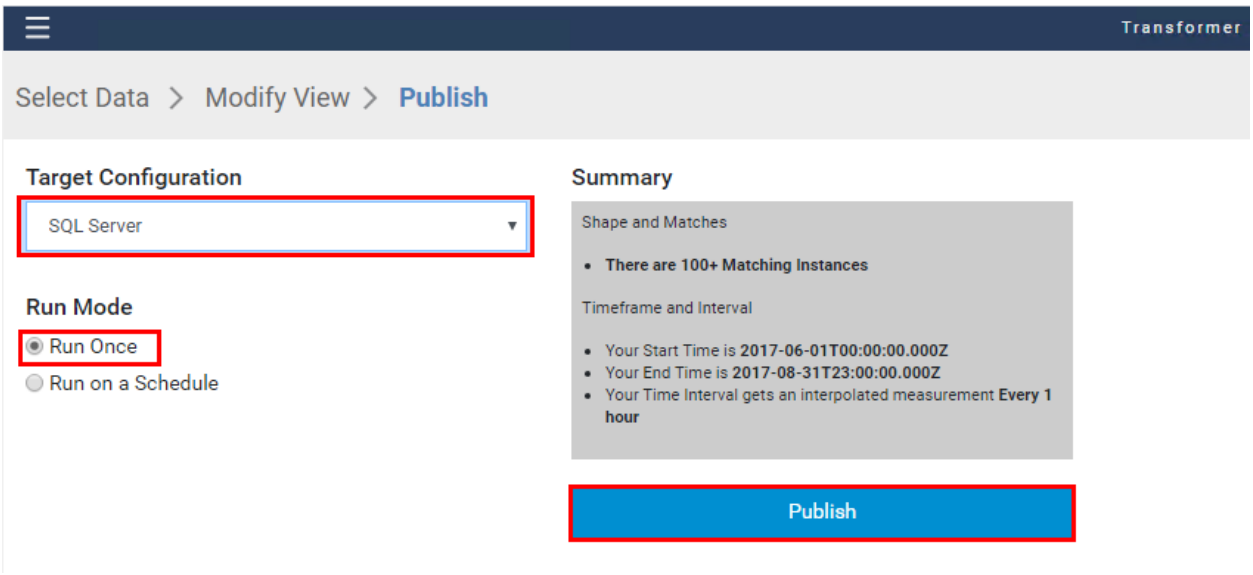
Click Display 5 Time Columns:



Now that 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.

Lesson 4: Building the Distribution Network Reports

We will now spend some time configuring a Microsoft Power BI report. The first step is importing the data.

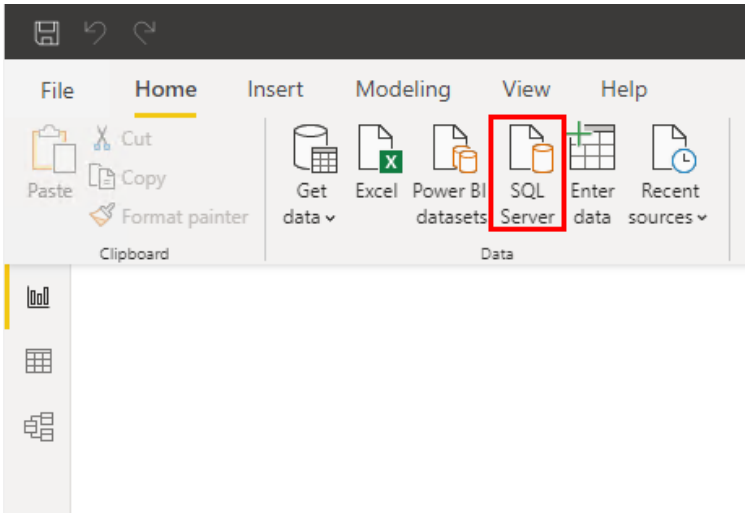
Importing Data into Power BI

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

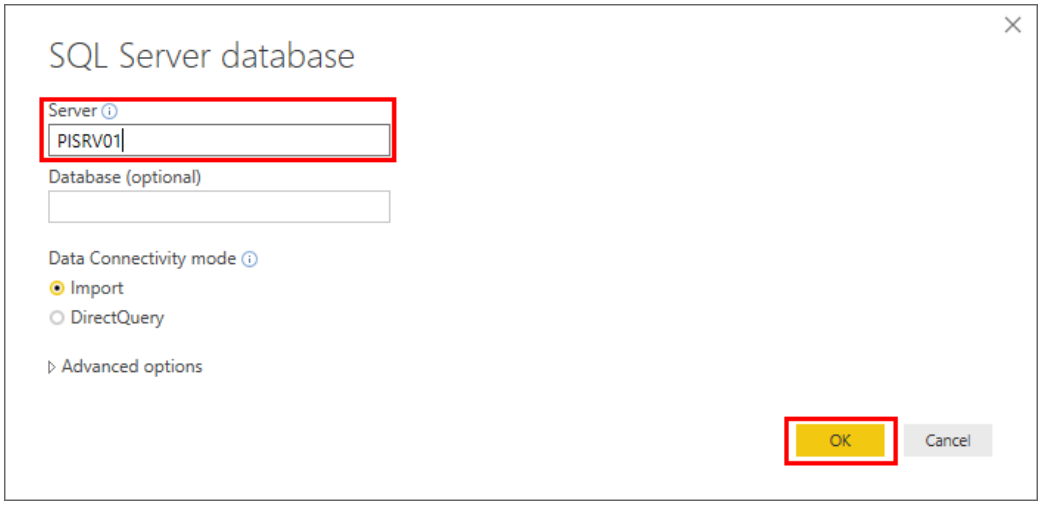
Approach:

Open Microsoft Power BI and start a new report.

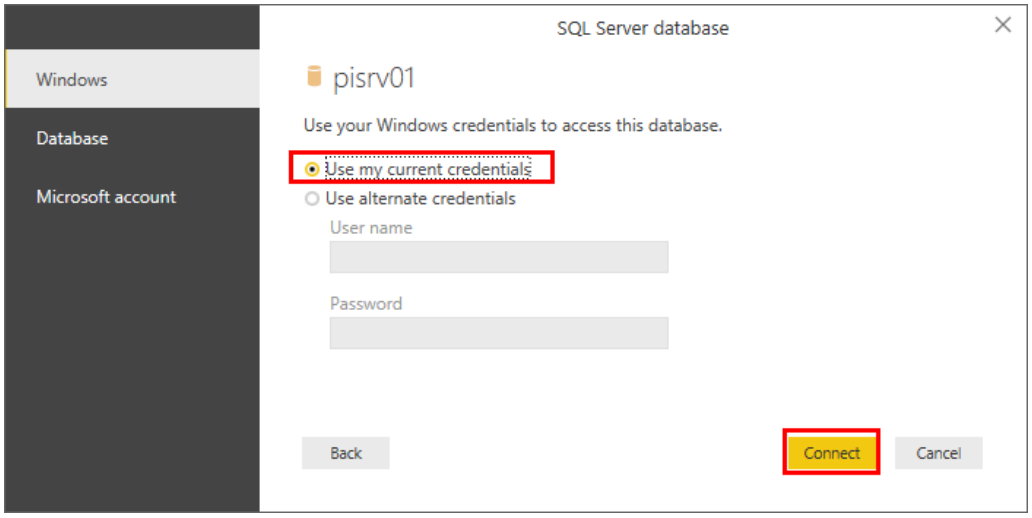
Select **SQL Server** in the Data Group.



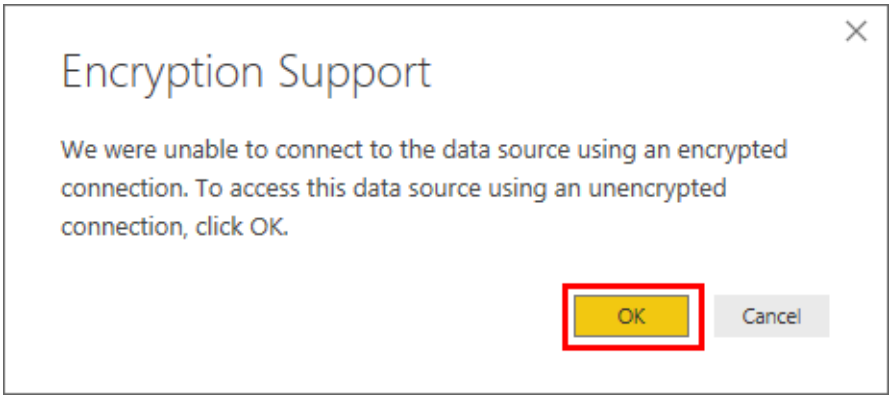
Enter **PISRV01** as the server name and click **OK**.



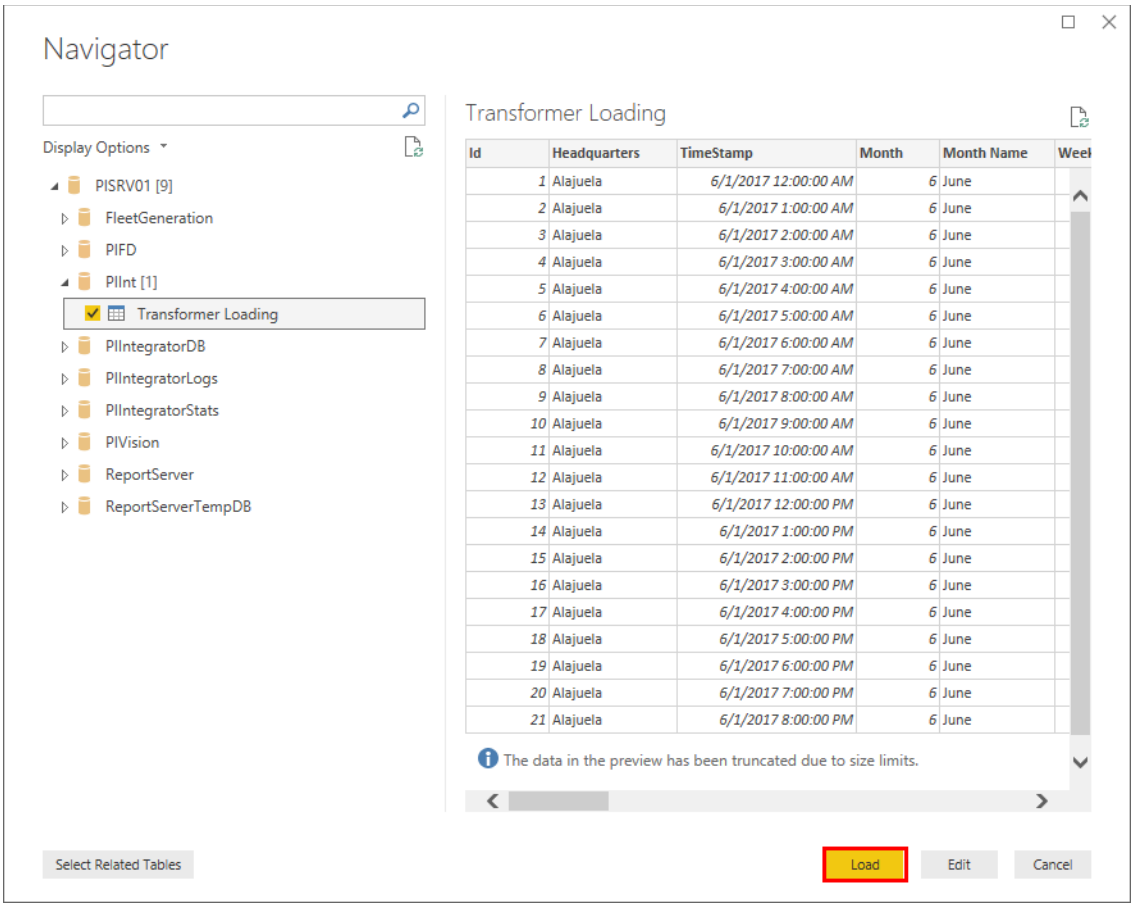
If Prompted, Leave “use my current credentials” selected and click **Connect**:



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. **This exceeds the 1 million row limit in Microsoft Excel.**

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. Work from the previous exercises can be safely discarded.

Transformer Loading Analysis

Objectives:

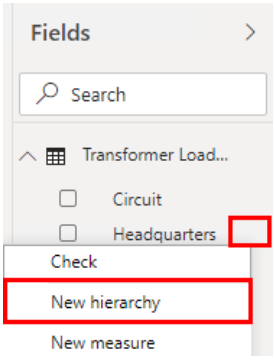
- Configure a **Hierarchy**
- Configure a **Hierarchy Slicer**
- Configure a **Measure** to calculate service hours
- Configure a **Group** to create bins for different load ranges which can then be used for highlighting and filtering
- Configure a **Stacked Bar Chart** to display the service hours spent 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 the following steps, 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.

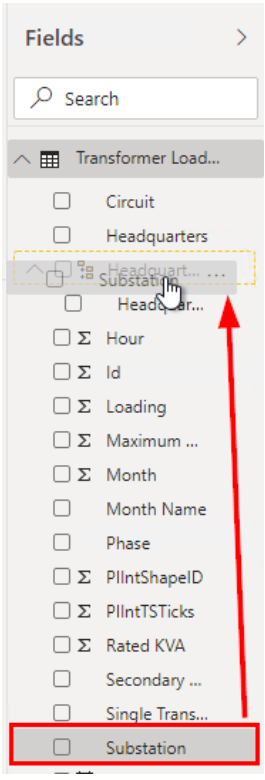
Approach:

Configuring the Hierarchy

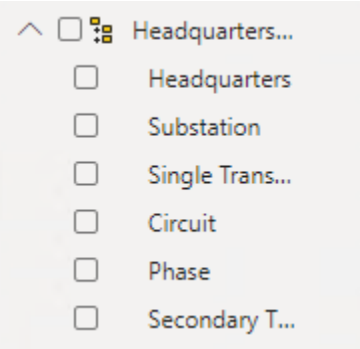
We will now create a hierarchy. In the **Fields List**, click the ellipses next to Headquarters and select **New hierarchy**:



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 to match the below if necessary.



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!

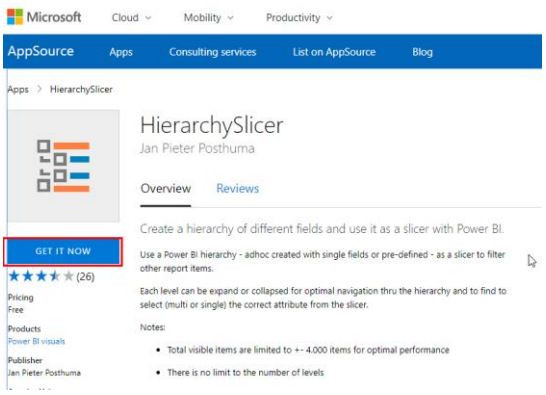
The Hierarchy Slicer is a custom visual that can be used to filter reports and mimic the PI AF hierarchy. This is similar to the PI TreeView from PI WebParts.

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:

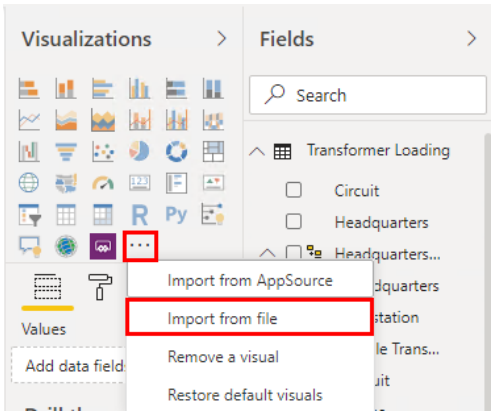
https://appsource.microsoft.com/en-us/product/power-bi-visuals/WA104380820?tab=Overview

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

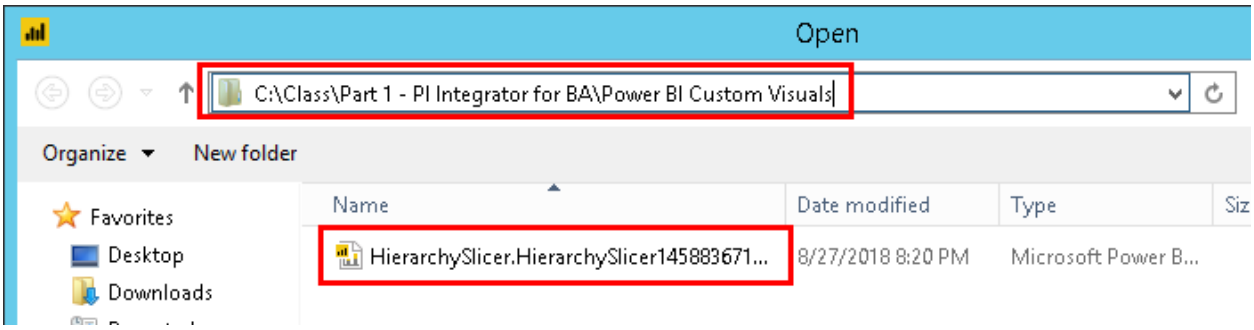


Importing and Configuring the Hierarchy Slicer

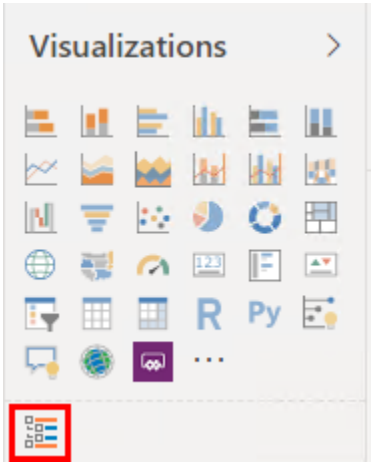
Now it's time to import the custom visual. Open Power BI, click the ellipses within the Visualization Pane, and select Import from file:



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



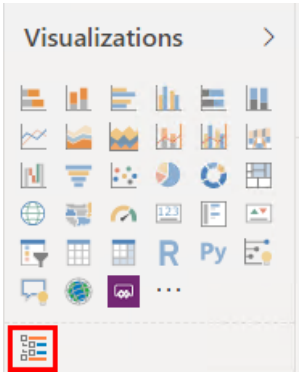
We should now see the Hierarchy Slicer in the list of available visuals:



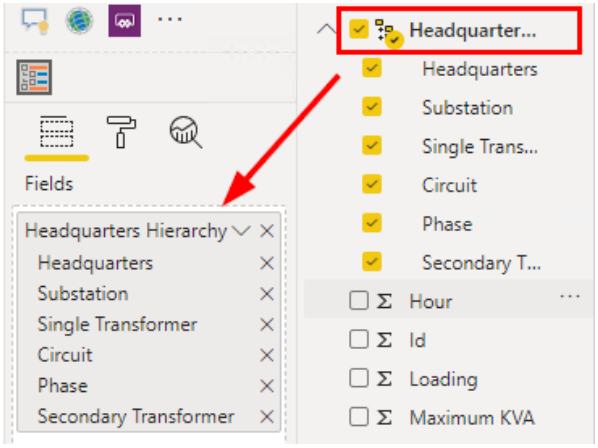
Mimic PI AF Hierarchy – Hierarchy Slicer

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 by clicking the icon:

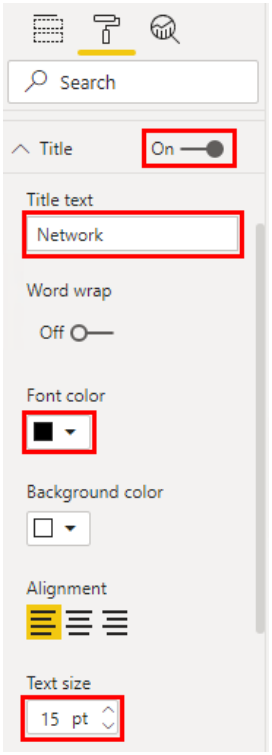


Drag and drop the **Hierarchy** to the visual fields:



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.

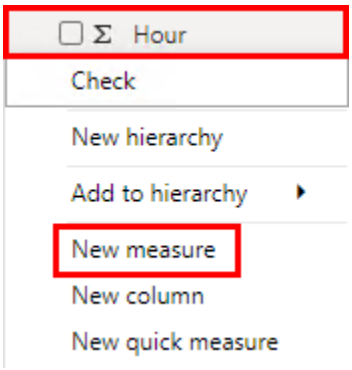
Change the Title of the Hierarchy Slicer to Network in the formatting options. Change the color and increase the text size.



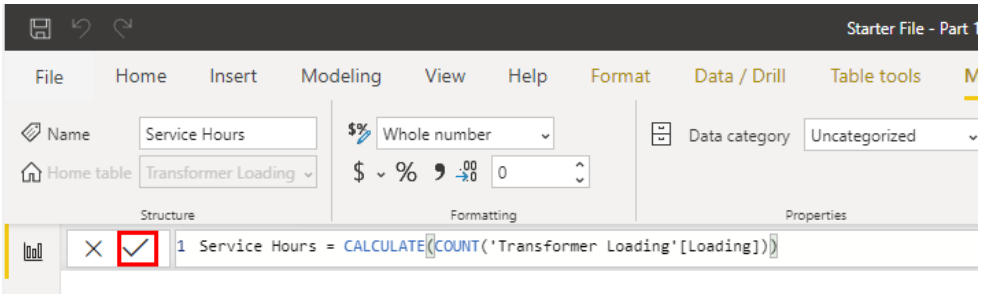
Service Hours

Now we'll 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**:



Enter the below formula into the configuration box and hit **Enter or click the Checkmark**:



The raw text is given below for convenience.

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

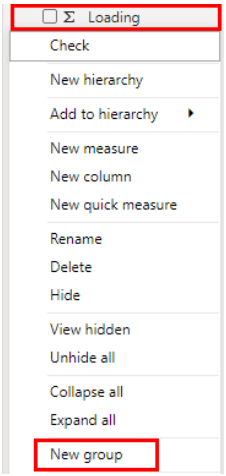
From a configuration perspective, Measures and Calculated Columns are configured similarly so the distinction may not be obvious.

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 to which it belongs. 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.

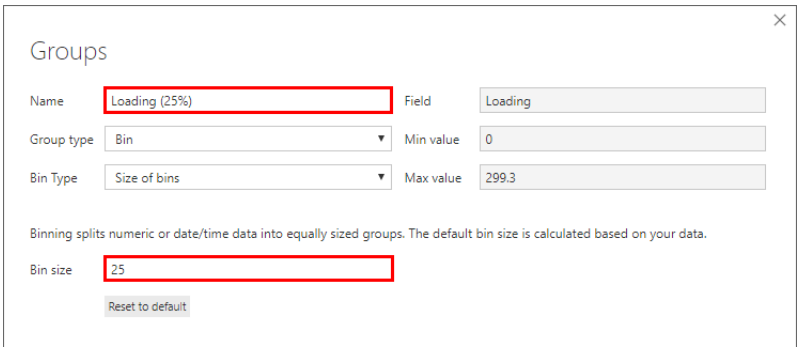
Loading Groups

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**.

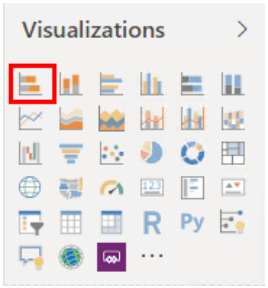


Change the name to **Loading** (**25%**) and set the bin size to 25, then click **OK**.

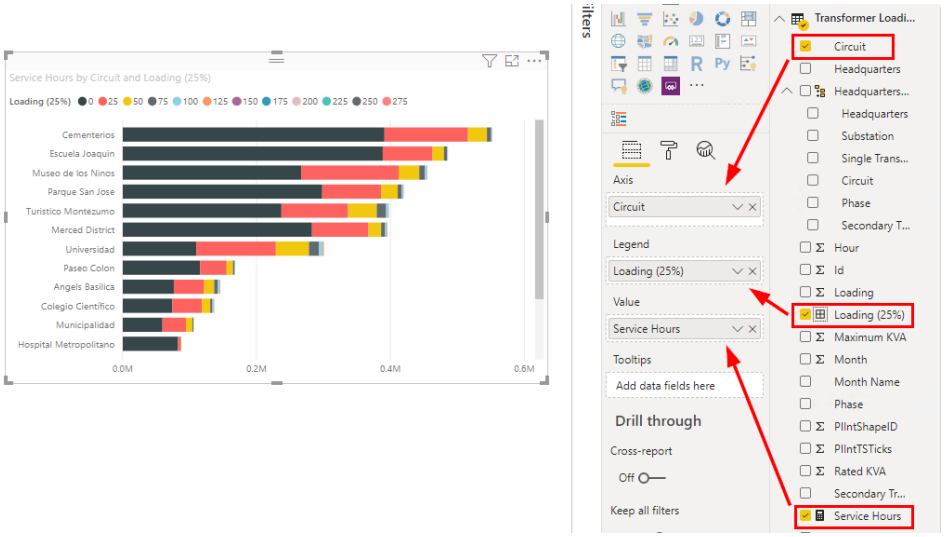


Loading by Circuit – Stacked Bar Chart

Now we can begin to configure the report. **Click some empty space** and then **click the Stacked Bar Chart icon**:

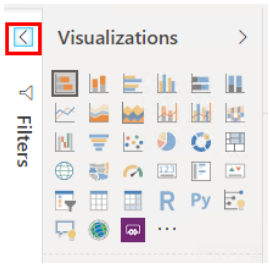


With the Stacked Bar Chart selected, drag and drop Fields from the data set into the field configuration boxes. Use **Circuit** for the Axis, **Loading (25%)** for the Legend, and **Service Hours** for the Value:

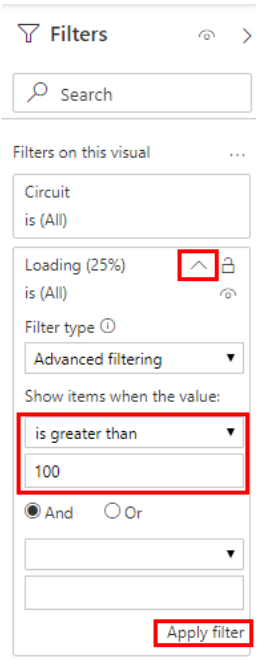


Next we will apply some formatting and filters to make the data set more manageable. We'll change the color scheme and only show Loadings greater than 100%, since loads in the normal range are not of interest to us.

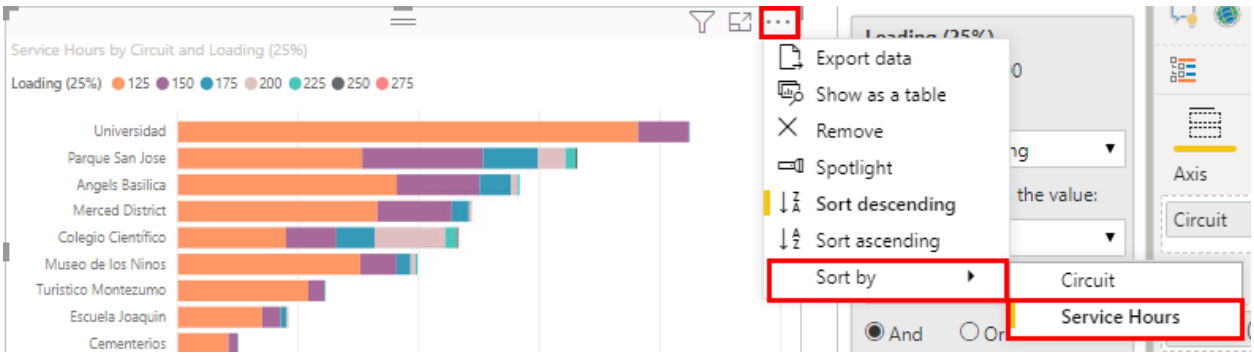
Expand the Filters Pane:



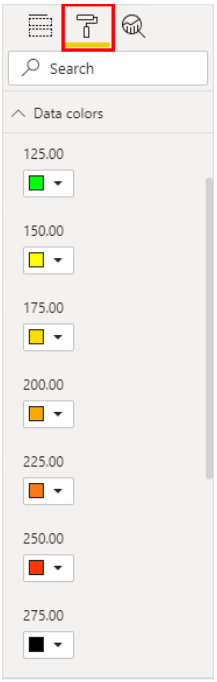
Filter for Loading **greater than** 100%. Be sure to click **Apply Filter**:



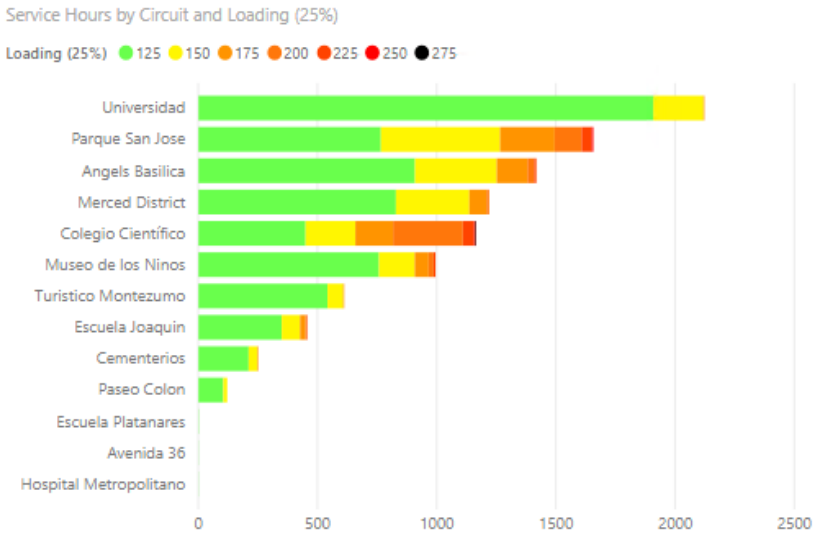
Next go to the Visualization Options and **sort by Service Hours** (done by default in this version of Power BI):



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:

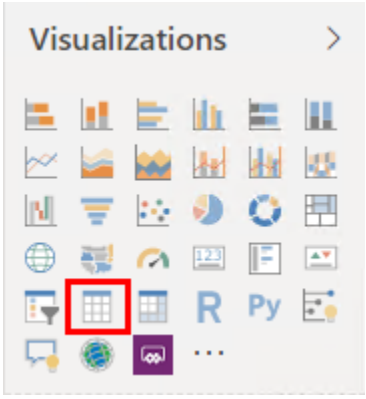


Service Hours and Average Load by Transformer – Table

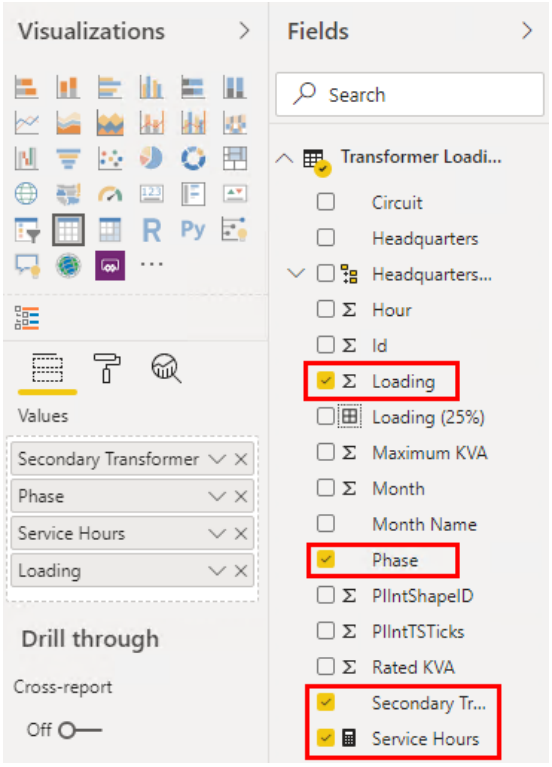
The next visual we will add is a basic table showing the Transformer Name, Phase, Service Hours, and Average Load. We will then filter the table to show only the top 10 transformers by average load. This will give us a quick indicator of which Transformers are consistently overloaded.

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

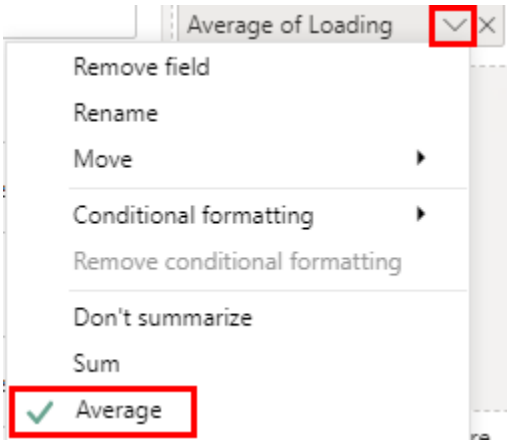
Create a **Table**:



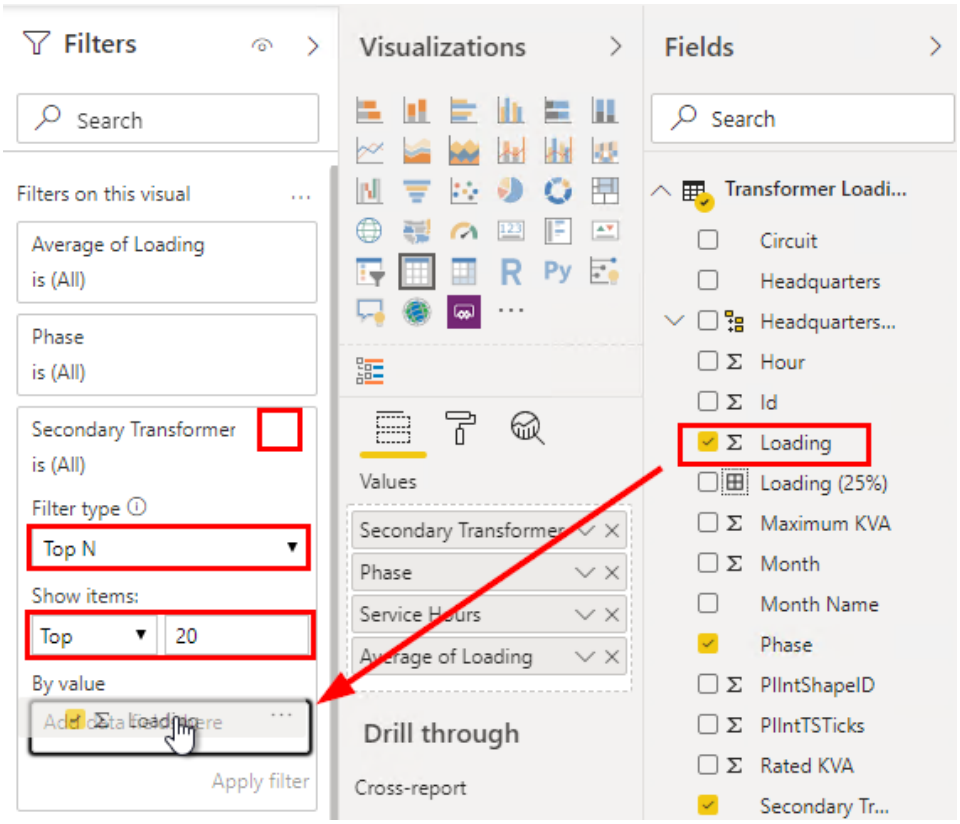
Drag and drop the **Secondary Transformer, Phase, Service Hours**, and **Loading** Fields into the Values section:



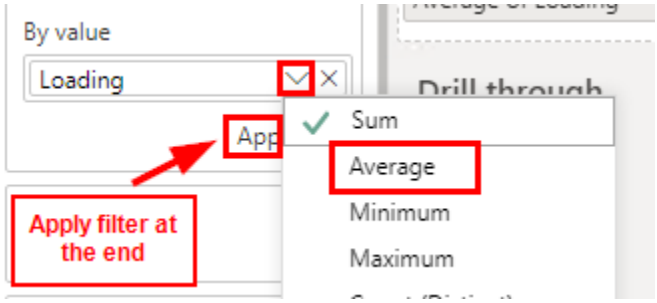
Change the **Loading** Value to summarize by **Average**:



Change the Visual Level Filters to Show the **Top 20** Transformers by **Loading**.

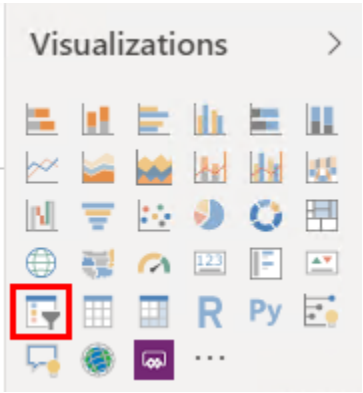


Change to summarize **Loading as Average**, then be sure to click **Apply filter**.



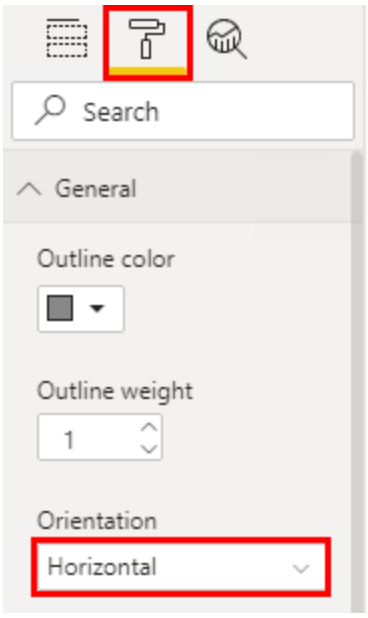
Filtering by Month – Slicer

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

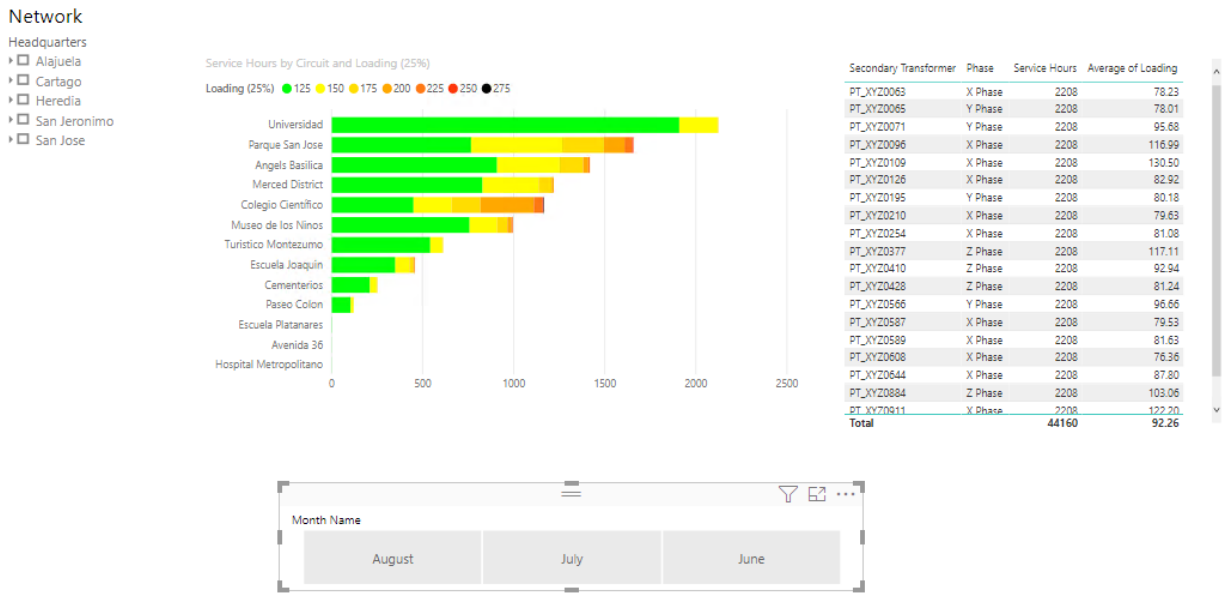


Drag **Month Name** to the field list.

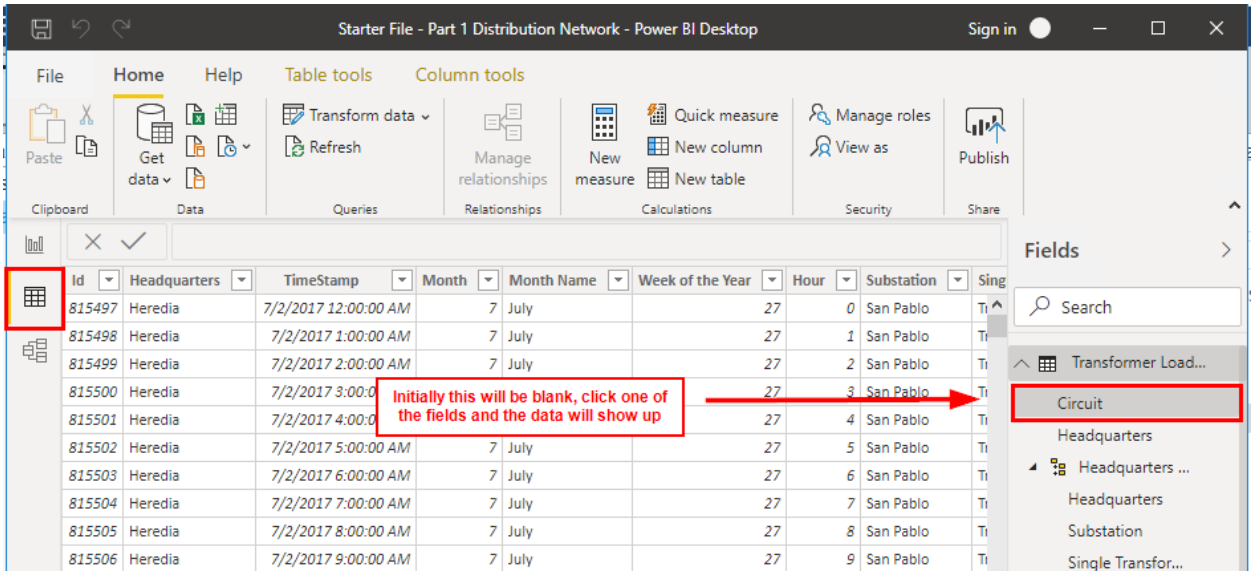
Go into the formatting options and change the orientation to **horizontal** to change the look of the Slicer.



Reposition & Resize the slicer so all months are in a single row. **Reposition & Resize the table** and stacked bar chart:

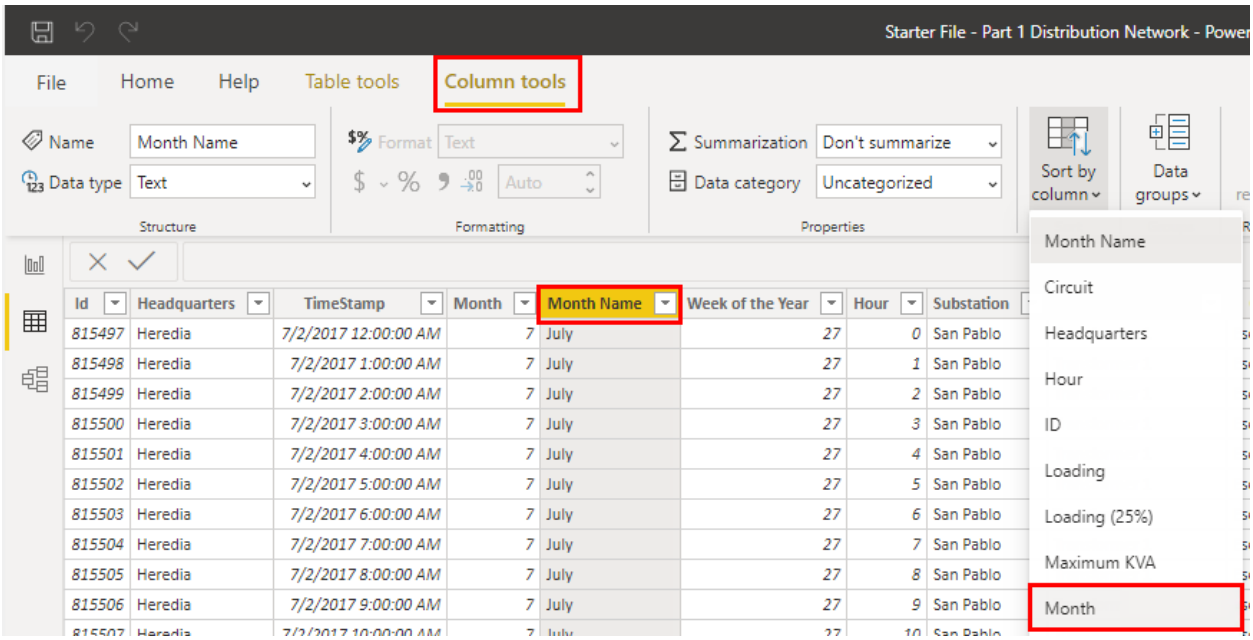


To put the Months in chronological order, we will sort the Month Name column in the data set by the Month column where the months are numbered. Go to the **Data View** and click one of the fields to make the data show up:

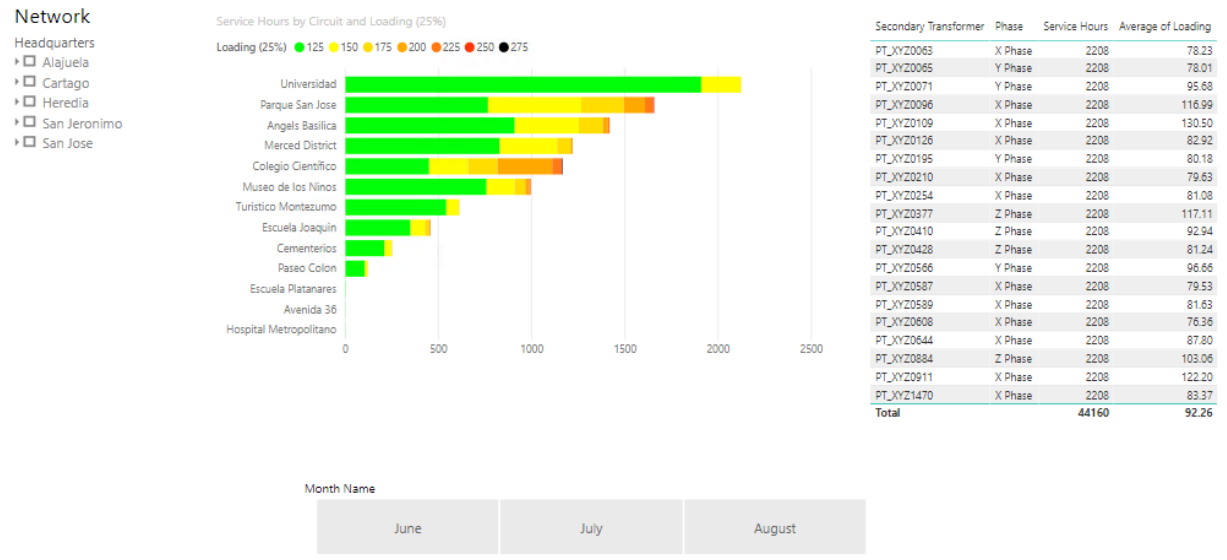


Id	Headquarters	TimeStamp	Month	Month Name	Week of the Year	Hour	Substation
815497	Heredia	7/2/2017 12:00:00 AM	7	July	27	0	San Pablo
815498	Heredia	7/2/2017 1:00:00 AM	7	July	27	1	San Pablo
815499	Heredia	7/2/2017 2:00:00 AM	7	July	27	2	San Pablo
815500	Heredia	7/2/2017 3:00:0			27	3	San Pablo
815501	Heredia	7/2/2017 4:00:0			27	4	San Pablo
815502	Heredia	7/2/2017 5:00:00 AM	7	July	27	5	San Pablo
815503	Heredia	7/2/2017 6:00:00 AM	7	July	27	6	San Pablo
815504	Heredia	7/2/2017 7:00:00 AM	7	July	27	7	San Pablo
815505	Heredia	7/2/2017 8:00:00 AM	7	July	27	8	San Pablo
815506	Heredia	7/2/2017 9:00:00 AM	7	July	27	9	San Pablo

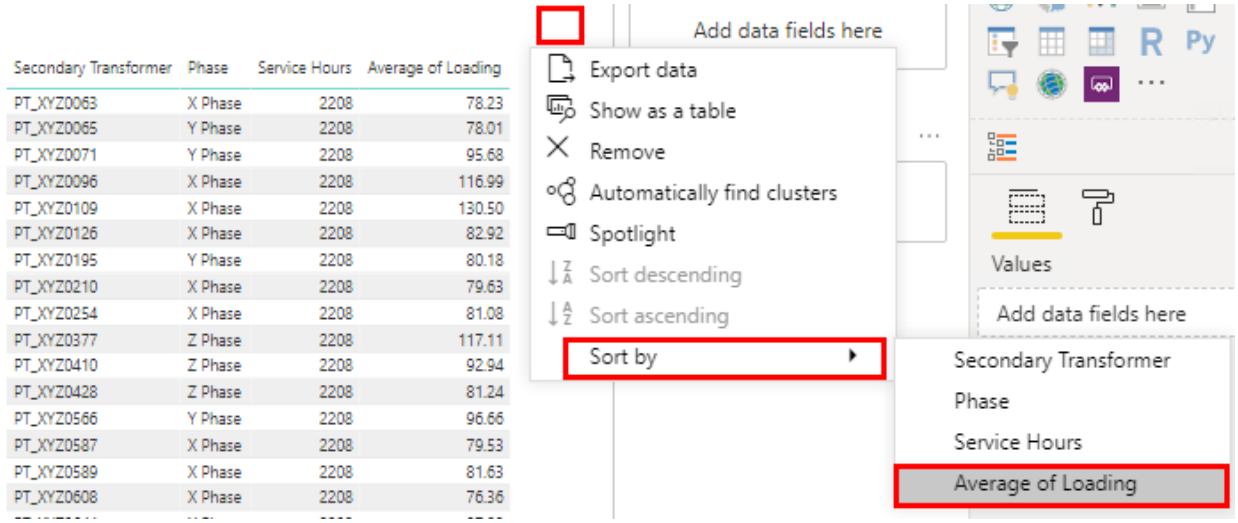
Select the **Month Name** column, open the **Column Tools** Ribbon, and **Sort by Column -> Month**:



The report should now look something like this:



Sort the table by Average of 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/2/TransformerTrends

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

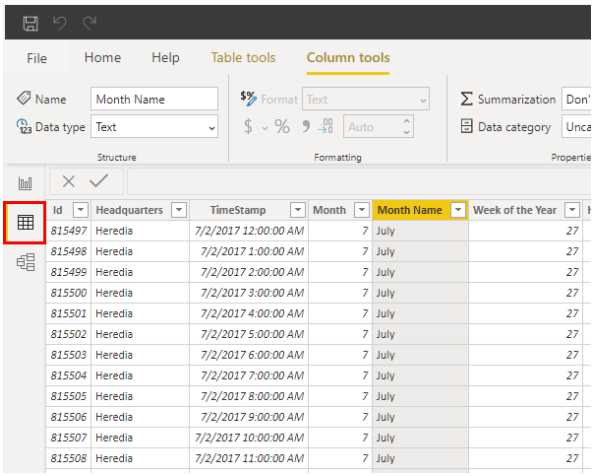
?Asset=\\PISRV01\Distribution Network\Secondary Transformers\PT_XYZ0046

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.

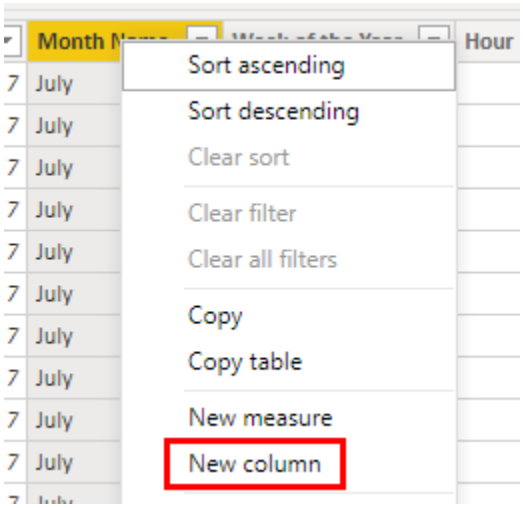
Once that is working, configure a Calculated Column to concatenate the URL with the Transformer asset path.

Go to the **Data** Tab:



Id	Headquarters	TimeStamp	Month	Month Name	Week of the Year
815497	Heredia	7/2/2017 12:00:00 AM	7	July	27
815498	Heredia	7/2/2017 1:00:00 AM	7	July	27
815499	Heredia	7/2/2017 2:00:00 AM	7	July	27
815500	Heredia	7/2/2017 3:00:00 AM	7	July	27
815501	Heredia	7/2/2017 4:00:00 AM	7	July	27
815502	Heredia	7/2/2017 5:00:00 AM	7	July	27
815503	Heredia	7/2/2017 6:00:00 AM	7	July	27
815504	Heredia	7/2/2017 7:00:00 AM	7	July	27
815505	Heredia	7/2/2017 8:00:00 AM	7	July	27
815506	Heredia	7/2/2017 9:00:00 AM	7	July	27
815507	Heredia	7/2/2017 10:00:00 AM	7	July	27
815508	Heredia	7/2/2017 11:00:00 AM	7	July	27

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

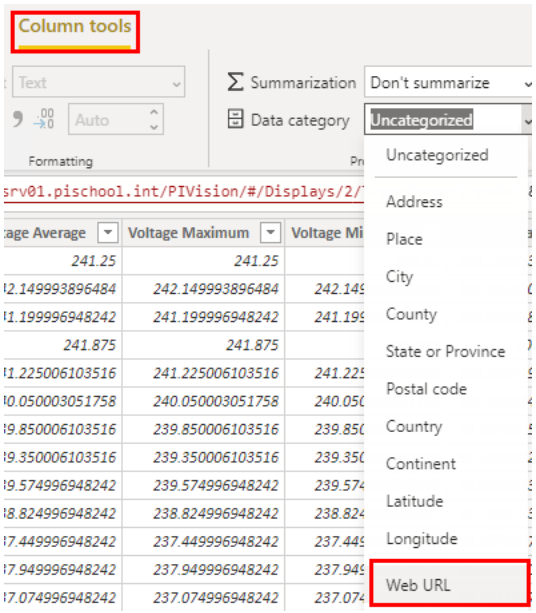


For the DAX formula, enter the following and **hit enter or click the checkmark**:

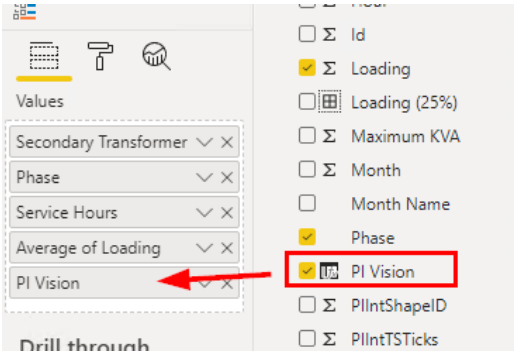
```
PI Vision = "https://pisrv01.pischool.int/PIVision/#/Displays/2/TransformerTrends" &
"?Asset=\\PISRV01\Distribution Network\Secondary Transformers\" & 'Transformer
Loading'[Secondary Transformer]
```

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

Go to the **Column Tools** ribbon, and change the **Data Category** to Web URL.

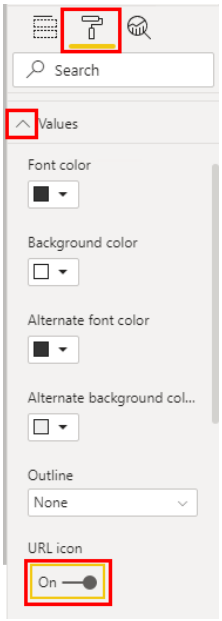


Now go back to the **Report Tab** and select the Table, then drag and drop the **PI Vision** field as one of the table values



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.

Lesson 5: Building the Fleet Generation Report

Now rather than a one-time export let's build a report that pulls current data upon refresh. To do so we'll configure a continuous Asset View that updates on a schedule. We'll be using a different AF Database this time: Online Fleet Generation.

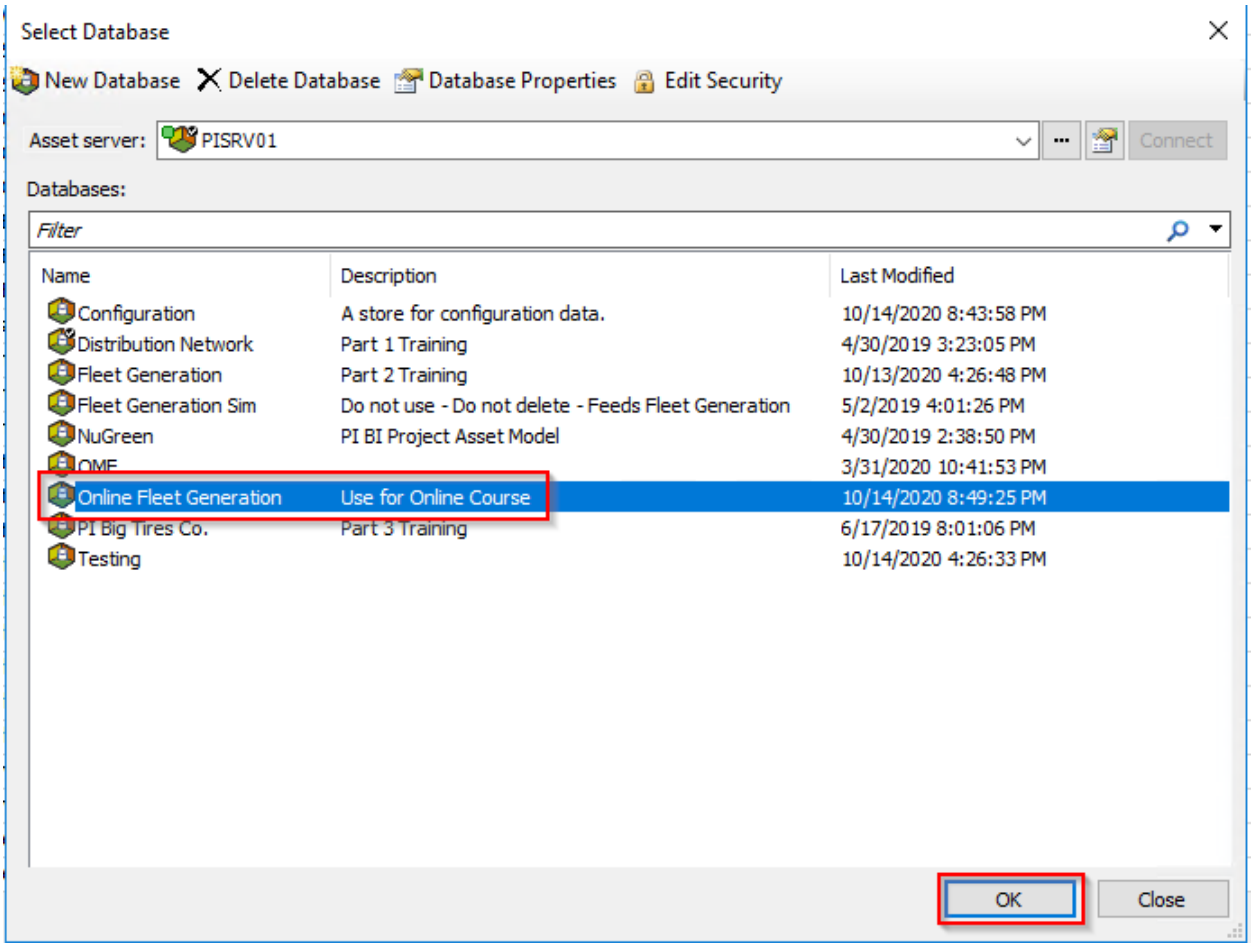
The Online Fleet Generation Database

We wish to analyze a number of KPIs for several generating units in a fictitious power generation company. All data is simulated / random.

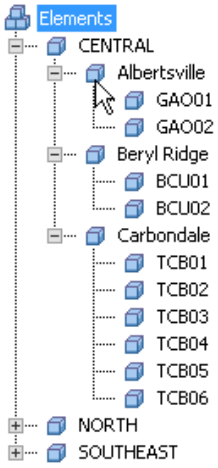
Open **PI System Explorer** and navigate to the **Online Fleet Generation database**.

The regular Fleet Generation database is used for the classroom version of this course where PI SQL Client is part of the core material.

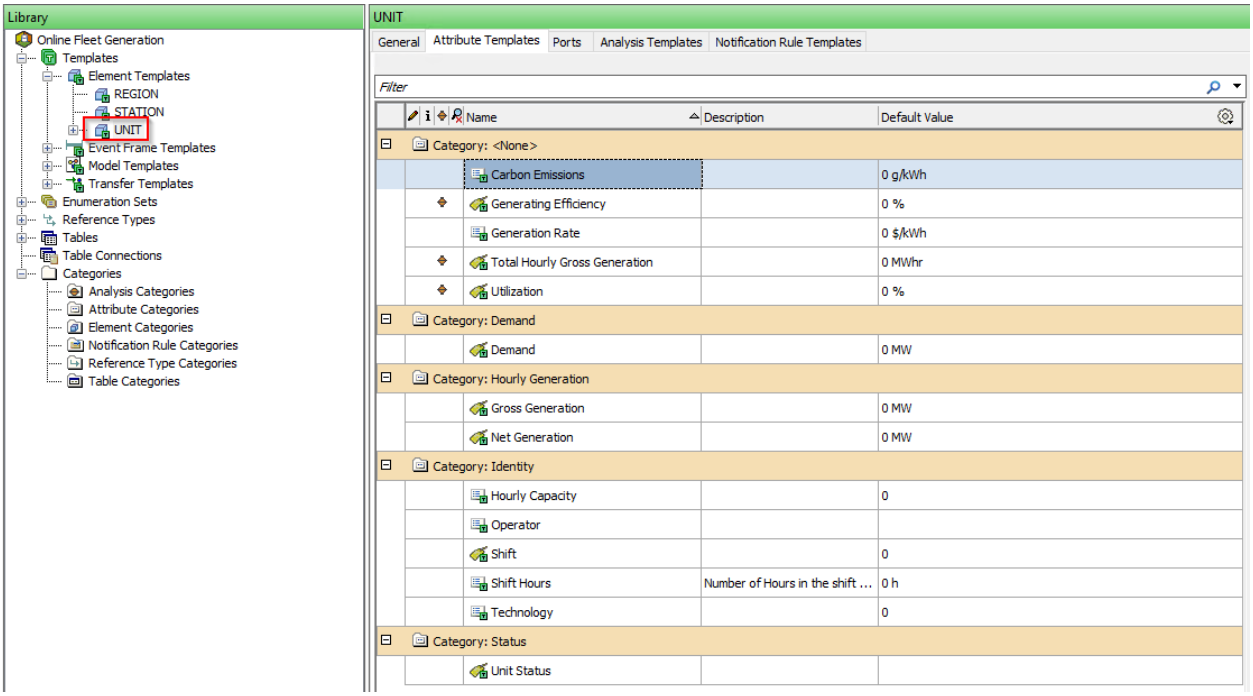
The Fleet Generation Sim database generates data for both the Fleet Generation and Online Fleet Generation databases.



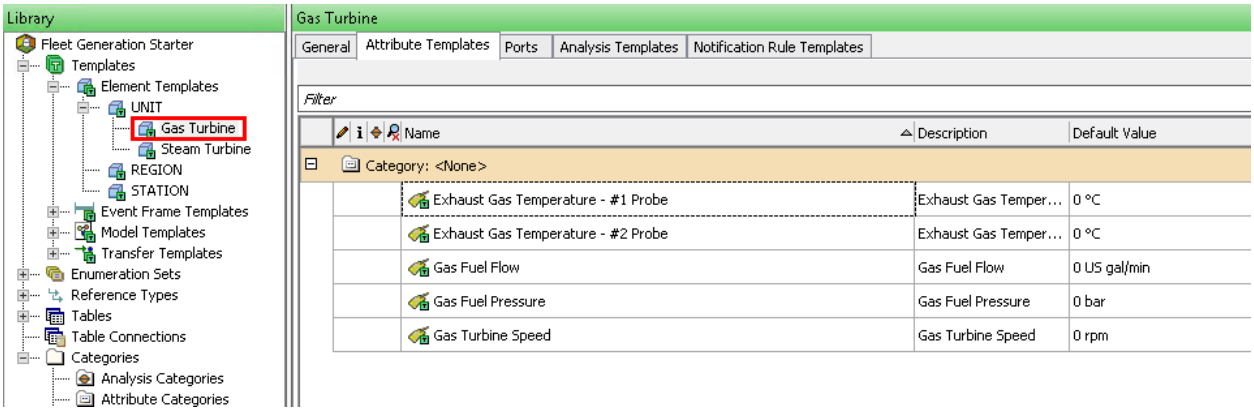
Browse the hierarchy, which is organized into Region, Station, and Unit.



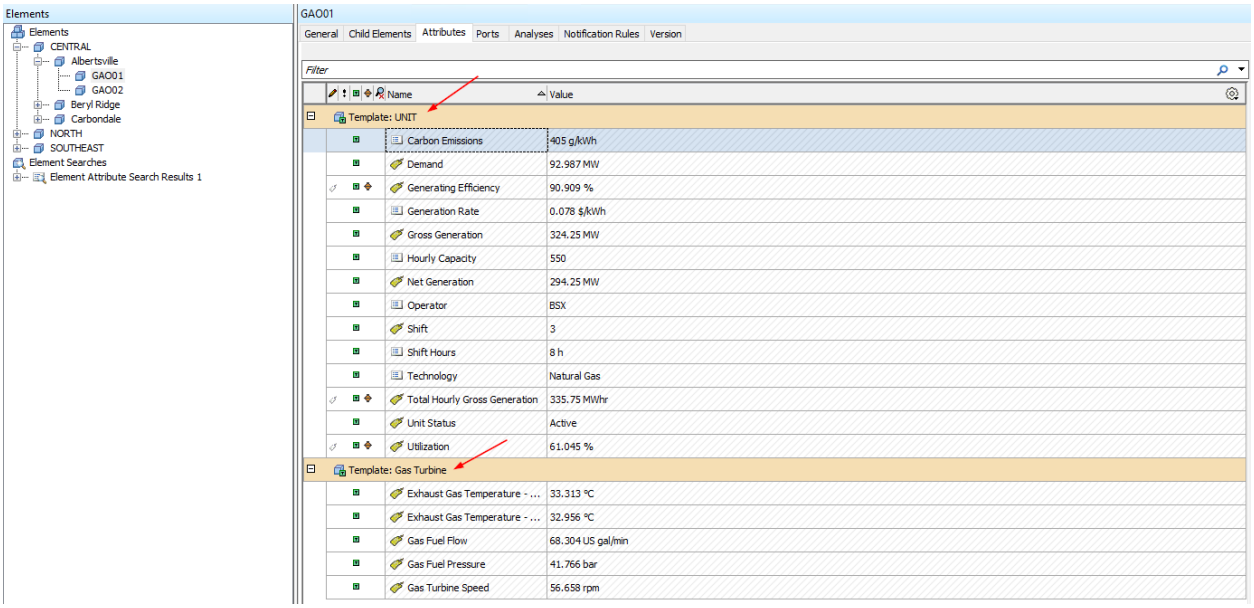
Most of the child elements are based on the generic **Unit template**.



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:



Preparing and Importing the Tables

For the report, we are going to separate the time-series data from the static data and configure table relationships to join the data sets together. Technically, we could design the Asset View 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. A table with one row per unit will also be required for the Final Challenge in order to join Event Frames and geospatial information.

Activity – Publish Unit Specifications (static data) Table

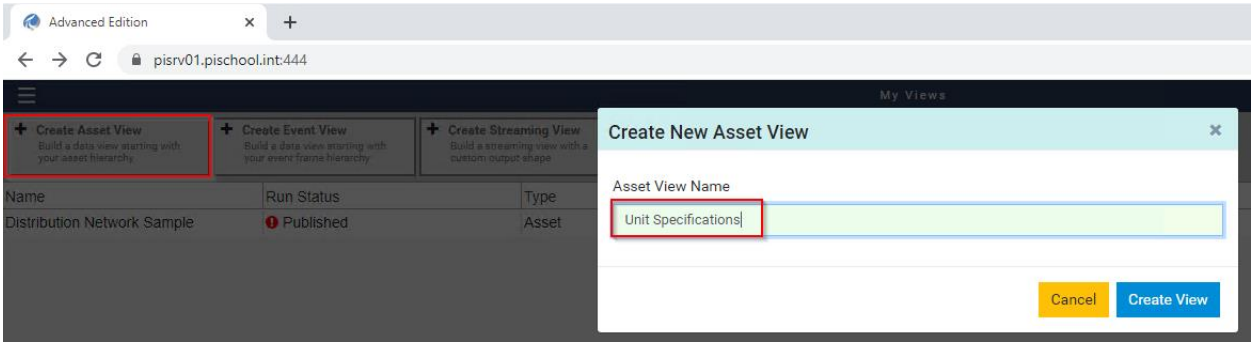
Objective:

- Publish an Asset View containing static attribute data

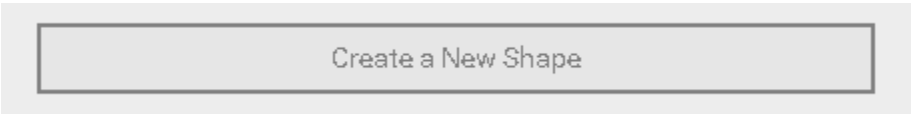
Approach:

Open Google Chrome and Navigate to the PI Integrator for BA Web UI at https://pisrv01.pischool.int:444

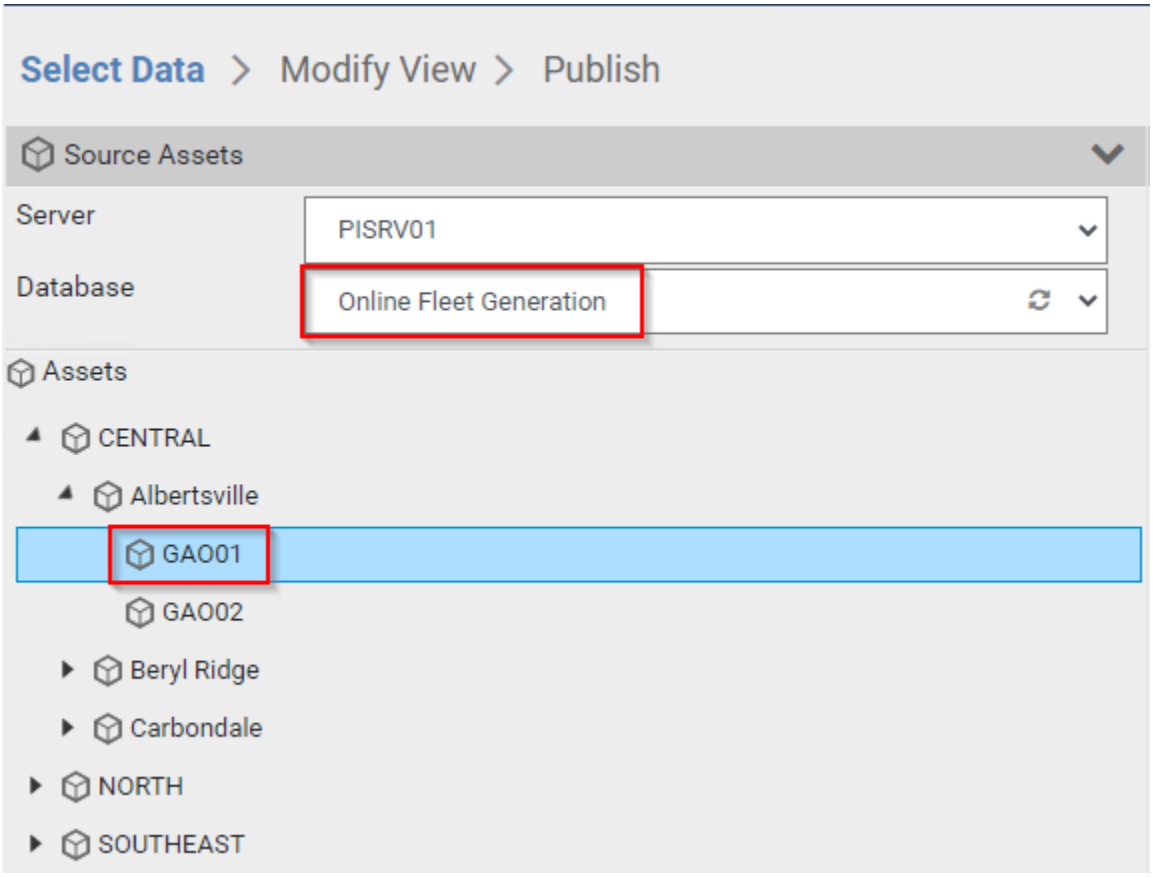
Click **Create Asset View** and name it **Unit Specifications**, click **Create View**:



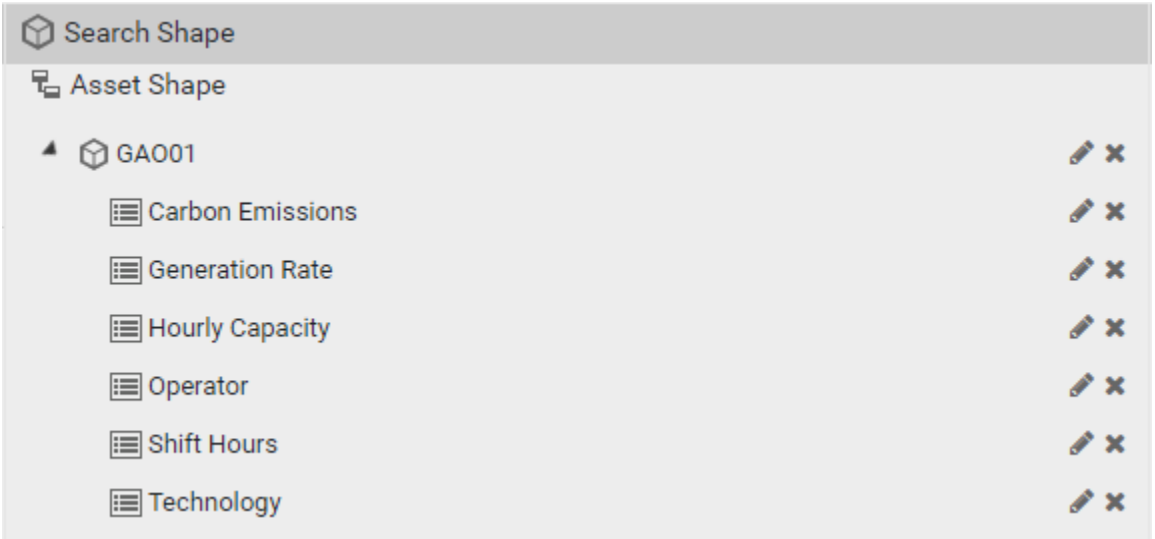
Click **Create a New Shape**



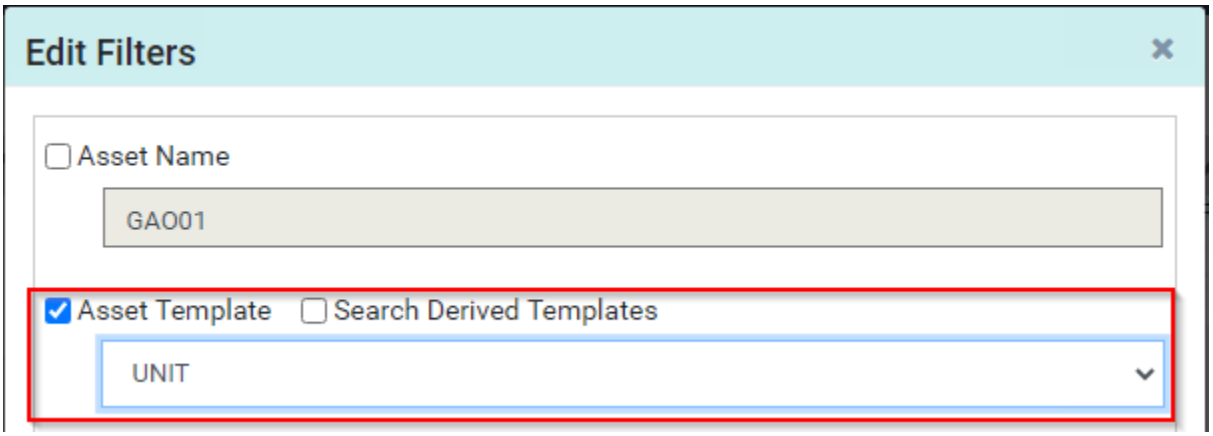
Select Online Fleet Generation as the AF Database, then drill down to GAO01.



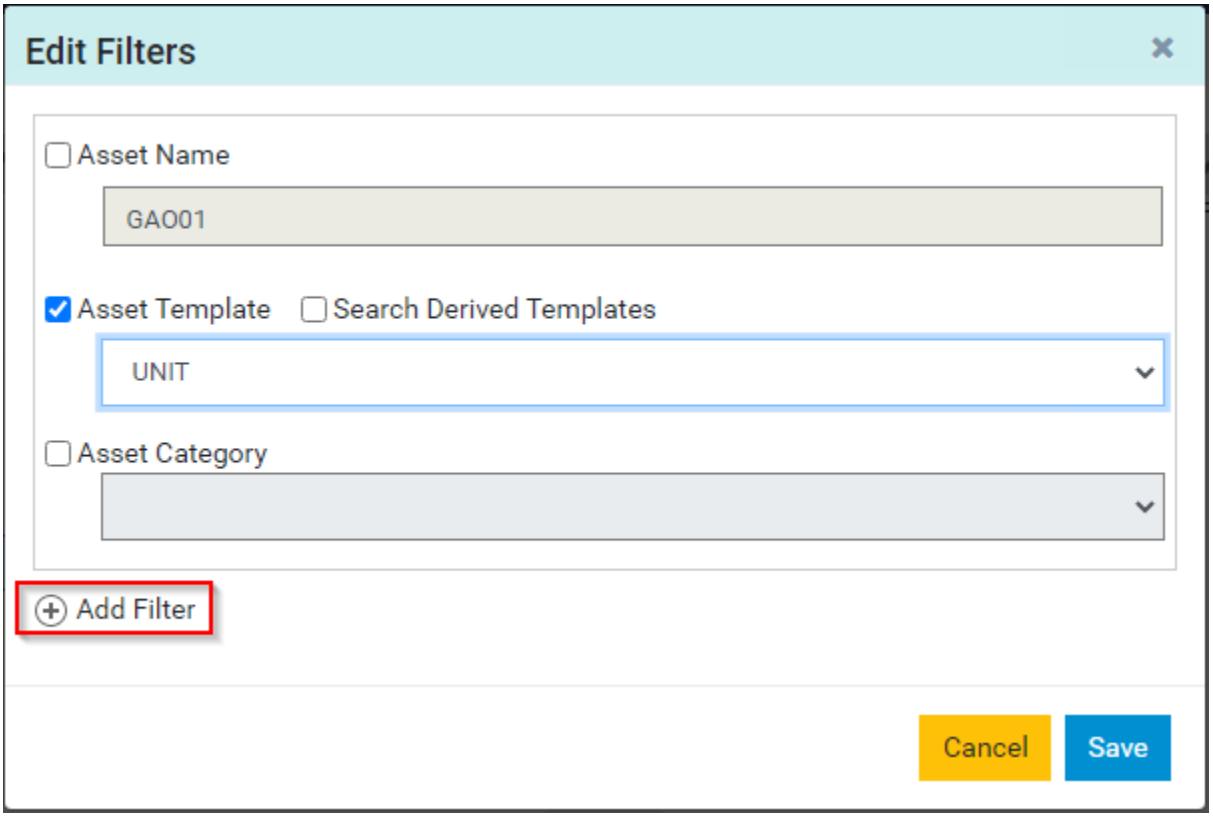
The static attributes are: **Carbon Emissions, Generation Rate, Hourly Capacity, Operator, Shift Hours,** and **Technology**. Drag them to the shape configuration:



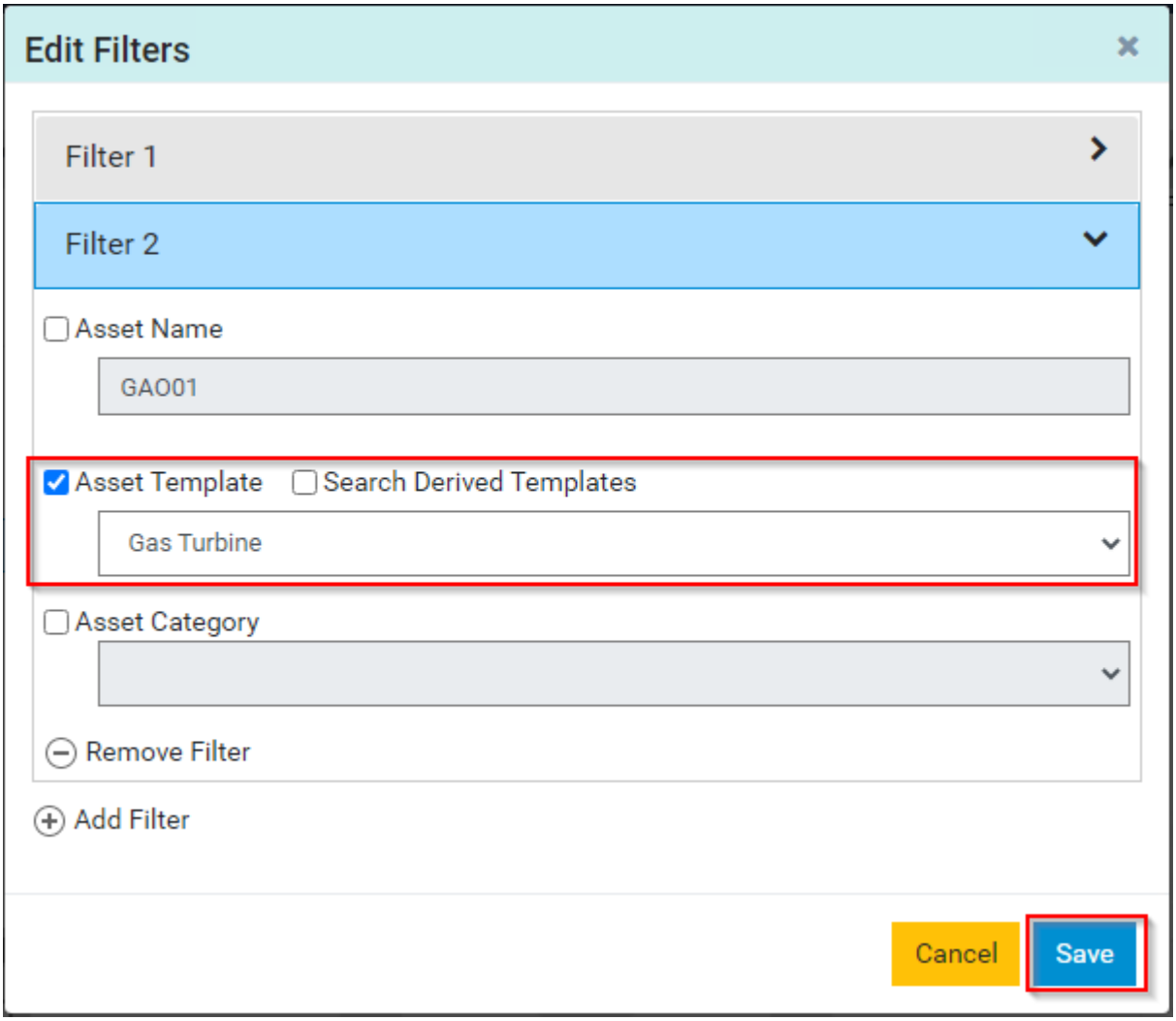
Edit the shape. Uncheck the box next to Asset Name and match on the UNIT template:



Add a filter to also include Gas Turbines:

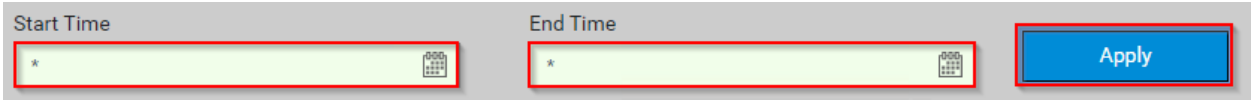


Uncheck the box next to Asset Name and match on the Gas Turbine template then Save:



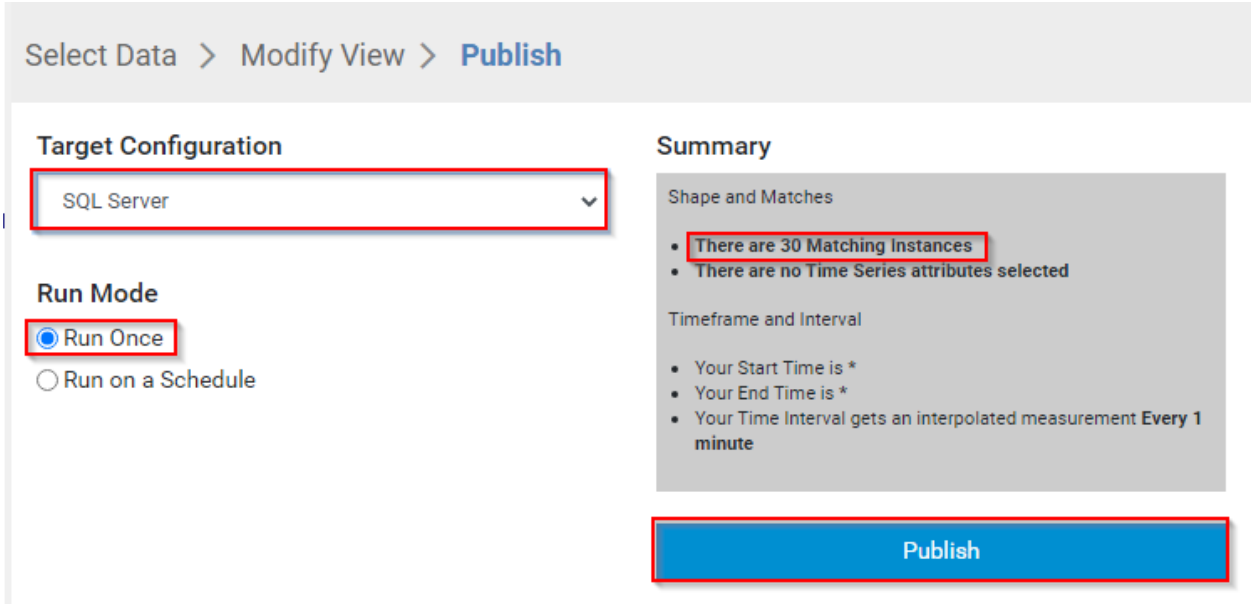
Click **Next.**

Change the Start Time and End Time to * since we only want one row per unit and click Apply. **The preview may not be accurate (missing some units)**.

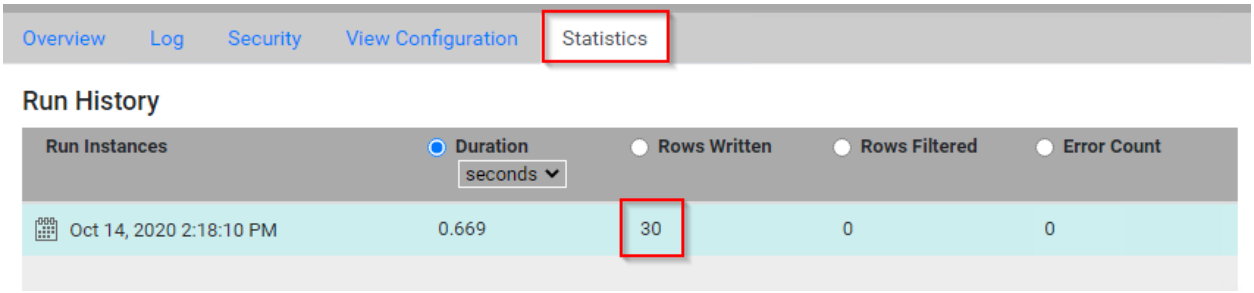


Click **Next.**

Publish to the SQL Server Target (run once):



When publishing is finished, check the statistics to confirm that 30 rows were published:



Activity – Publish Unit Performance (dynamic data) Table

Objective:

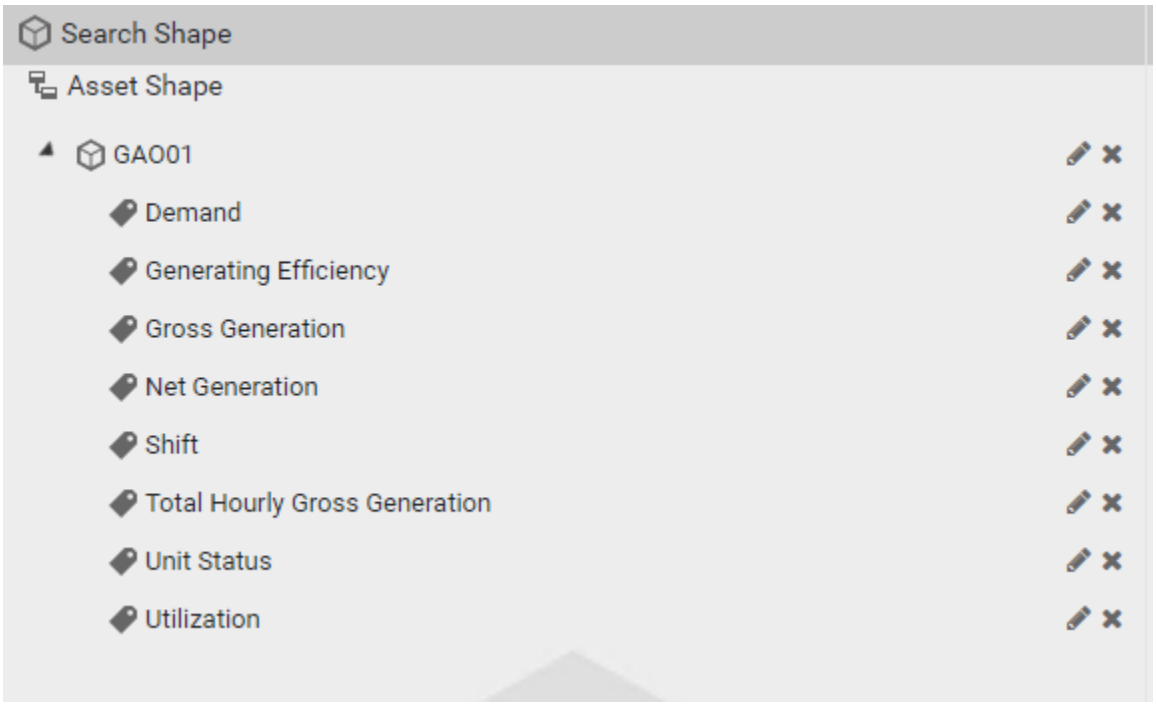
- Publish an Asset View containing dynamic attribute data

Approach:

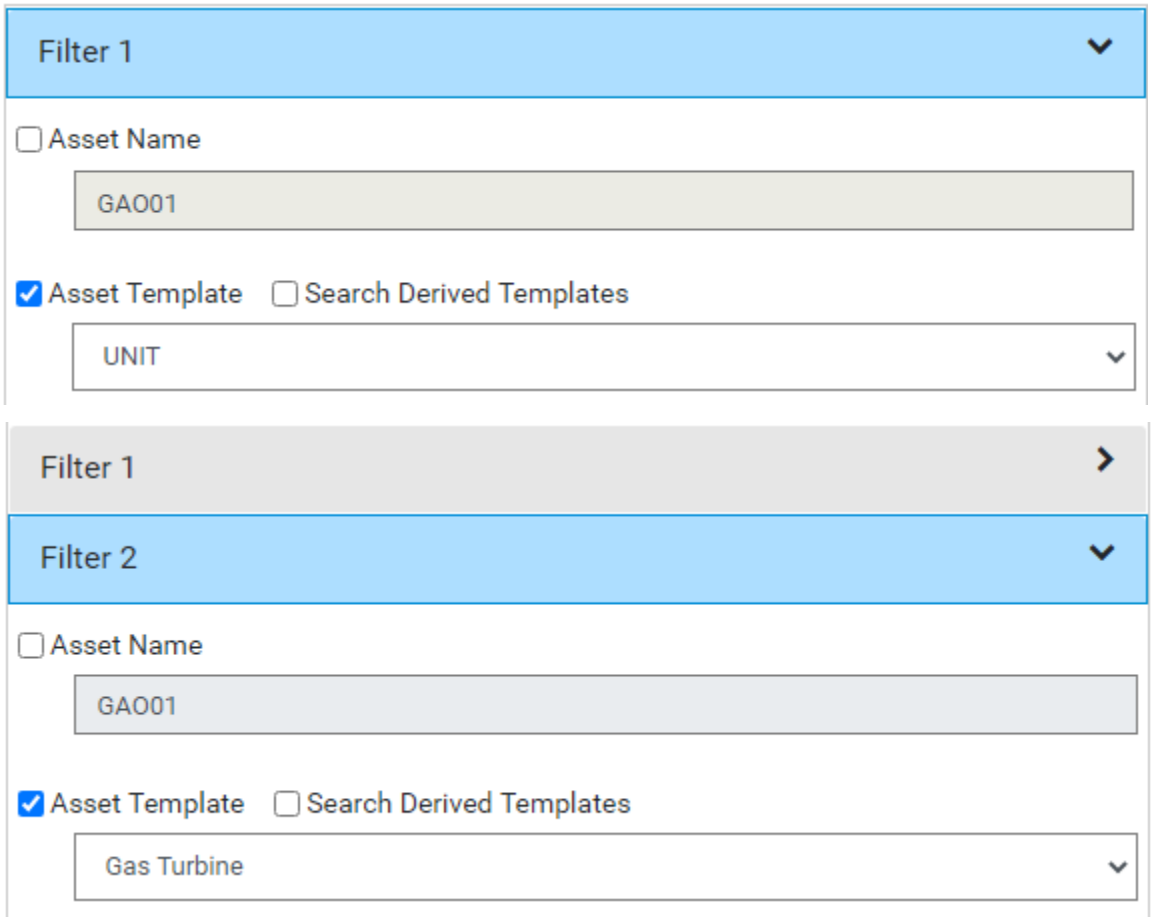
Create a new **Asset View** name **Unit Performance**.

Create a new shape. Navigate to GAO01 in the Online Fleet Generation database.

Drag **Demand, Generating Efficiency, Gross Generation, Net Generation, Shift, Total Hourly Gross Generation, Unit Status,** and **Utilization** to the shape configuration (exclude those attributes specific to Gas Turbines):

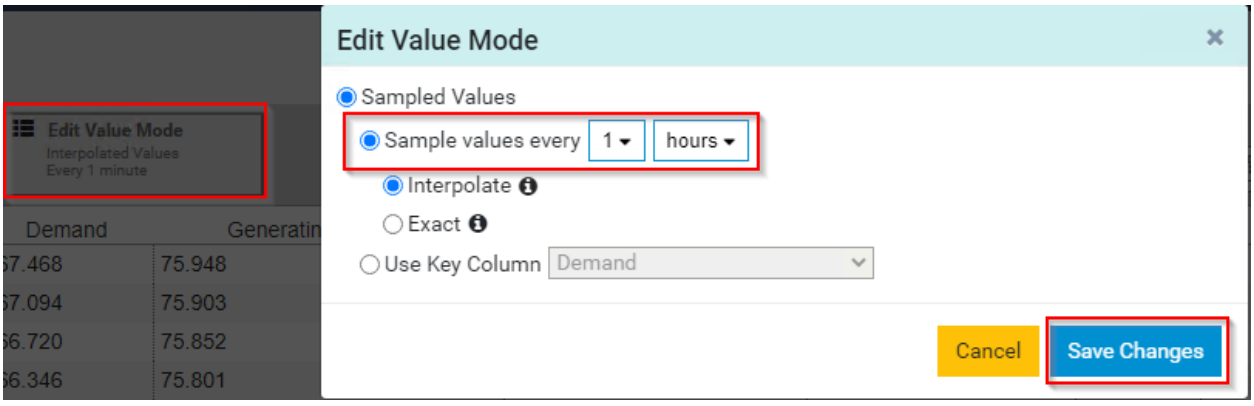


Edit the shape to match UNITs and Gas Turbines (add a second filter) as per the previous exercise:

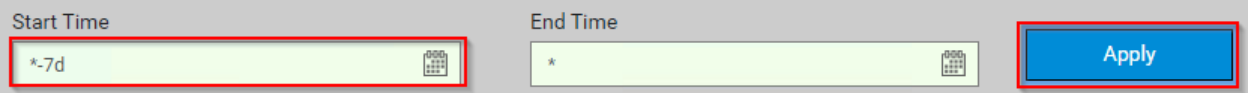


There should be 30 matches. Click **Next**.

Change the Value Mode to sample every 1 hour and Save Changes:

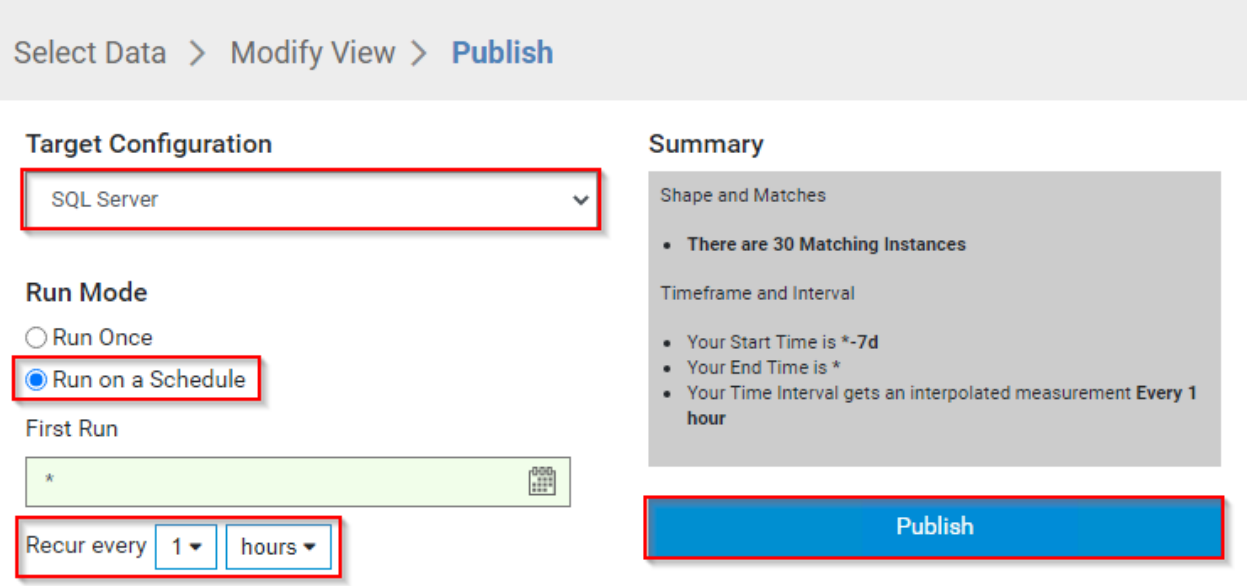


Change the **Start Time** to ***-7d** and click **Apply.**

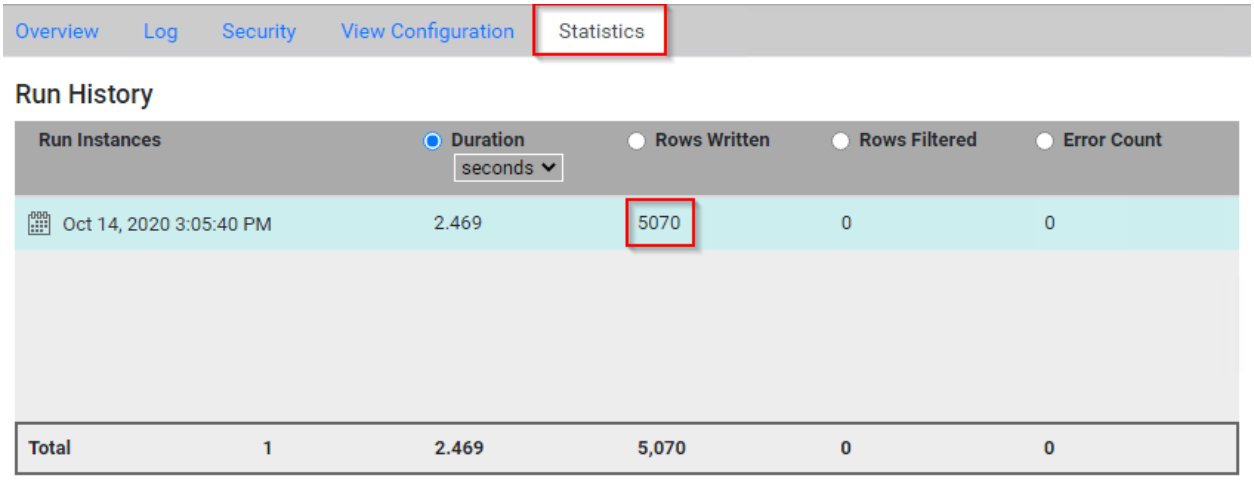


Click **Next**.

Select **SQL Server** as the Target. Run on an **hourly schedule** to keep the data current. Click **Publish**:



When publishing is finished, check the Statistics to confirm that 5070 rows were published:



Activity – Import the Unit Specifications and Unit Performance Tables

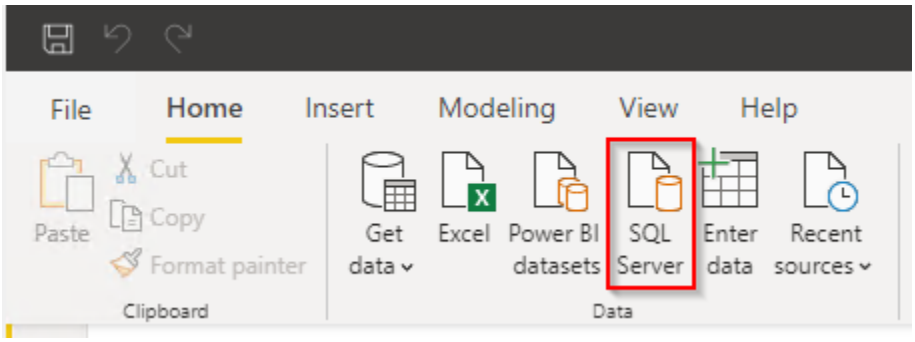
Objective:

- Import the SQL Server tables created in the previous exercises
- Rename the 'UNITGas Turbine' columns

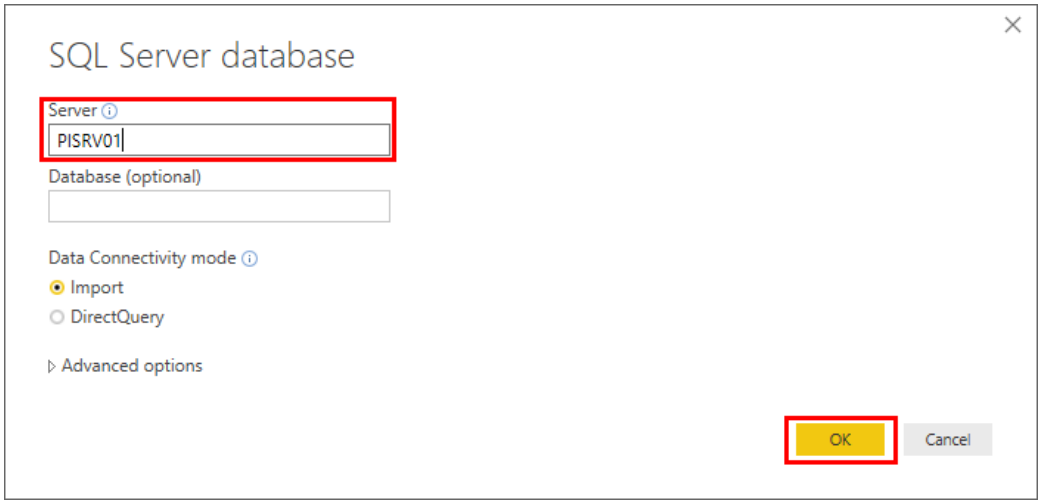
Approach:

Open a new Power BI report.

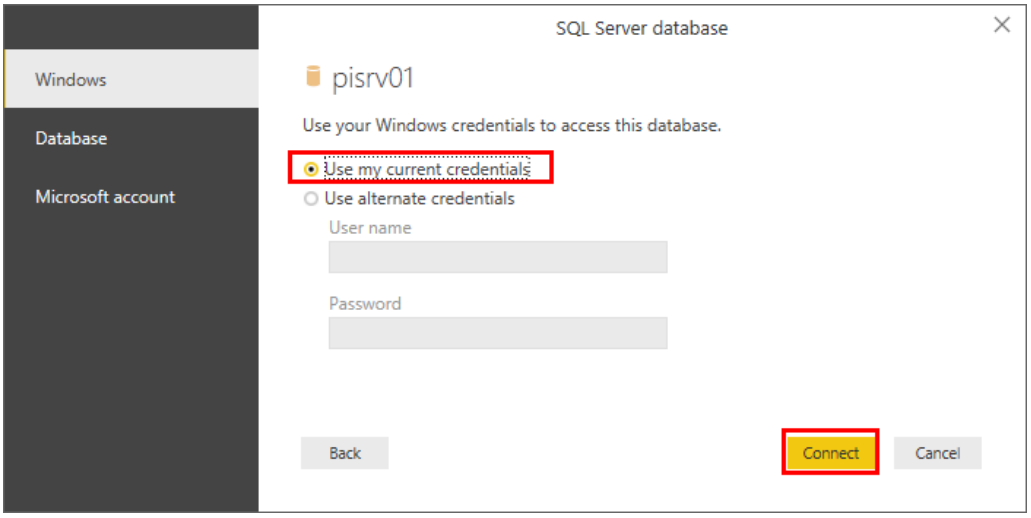
Import data from SQL Server:



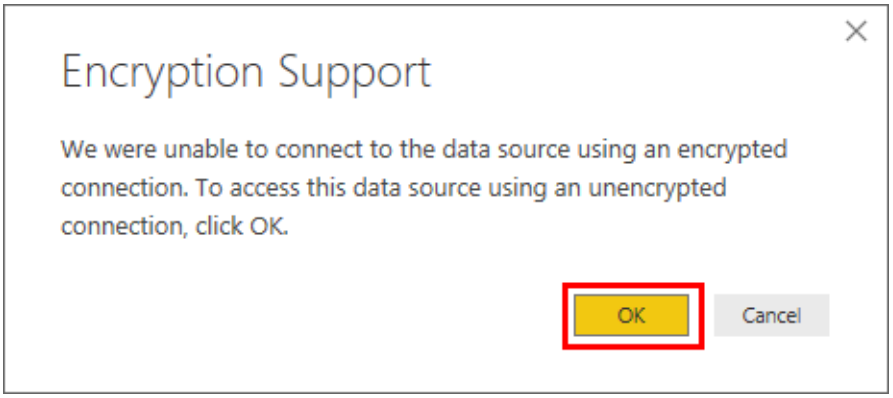
Enter **PISRV01** as the server name and click **OK**.



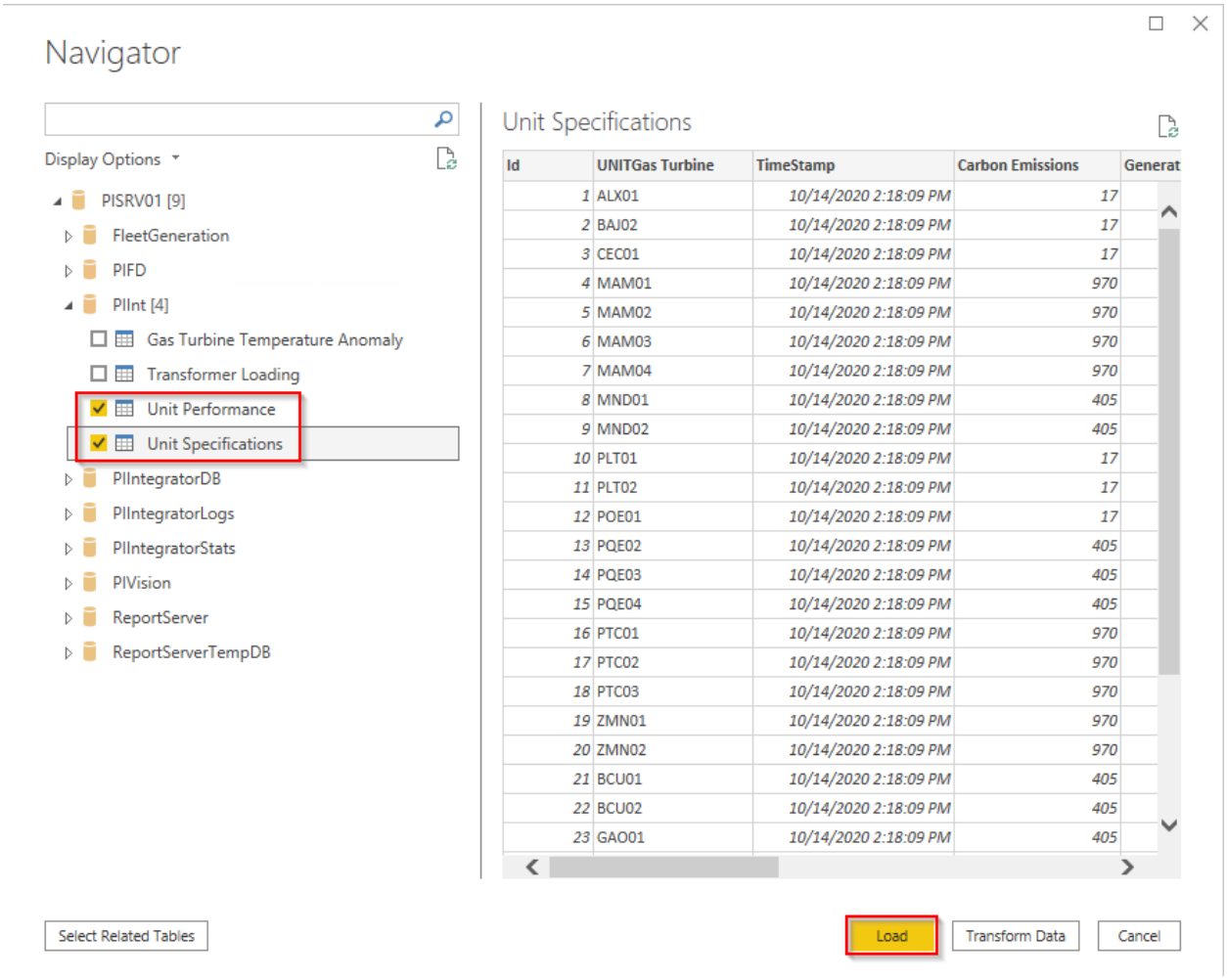
If Prompted, Leave “use my current credentials” selected and click **Connect**:



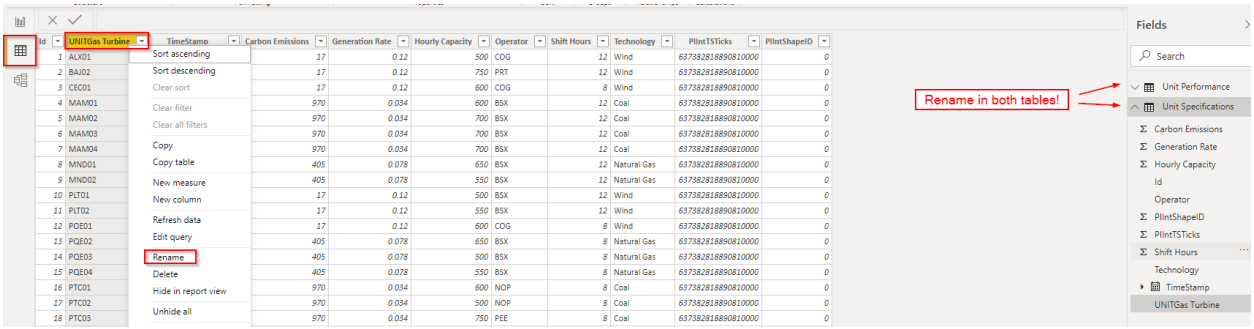
There may be a warning that the connection is not encrypted, this can be safely ignored, **click OK if prompted:**



Expand the PIInt database and Select the **Unit Performance** and **Unit Specifications** tables, click **Load:**



Once the tables have loaded, **right-click -> Rename** the "**UNITGas Turbine**" column to **UNIT for both tables**.



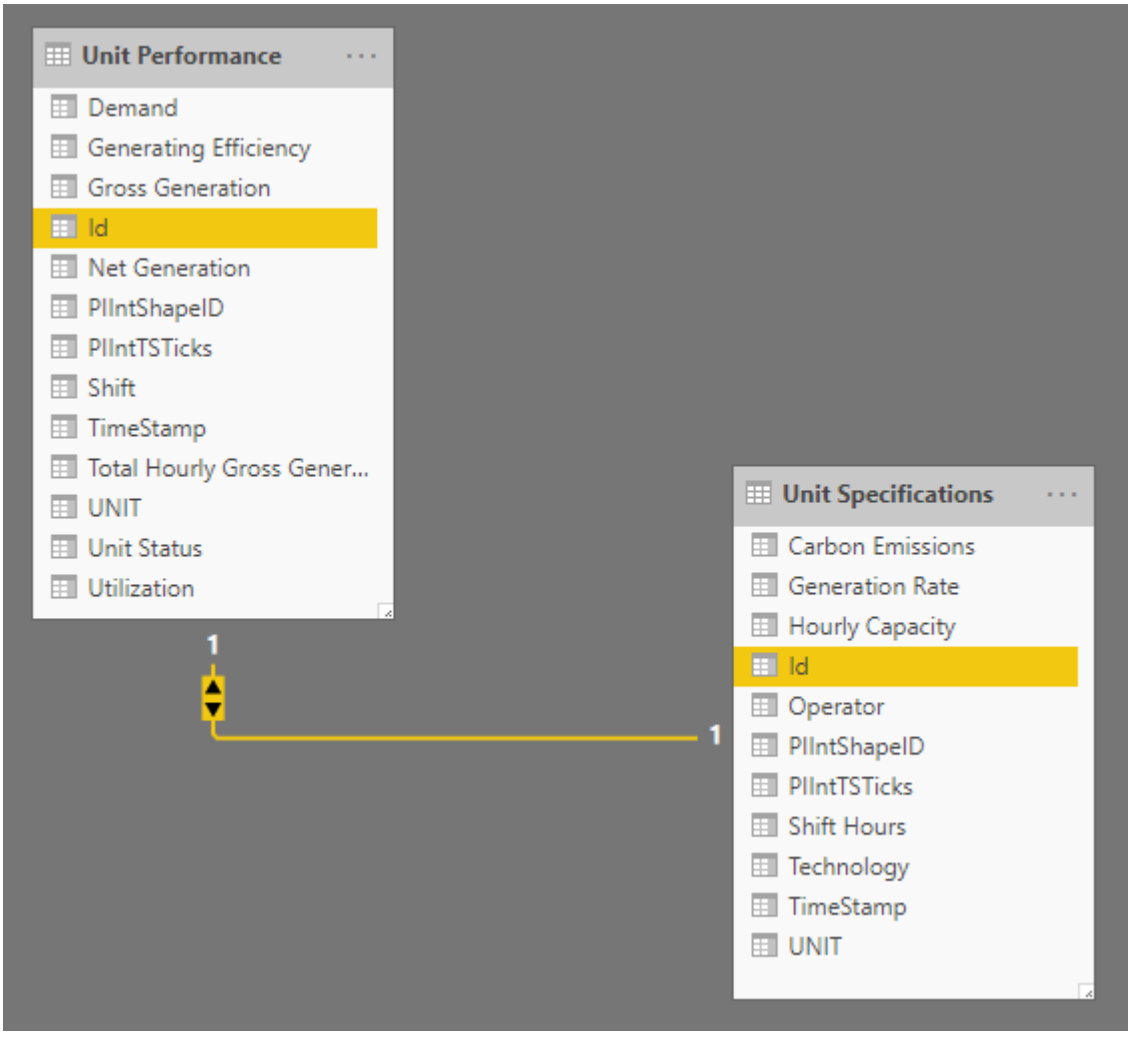
Activity – Create Table Relationships

Objective:

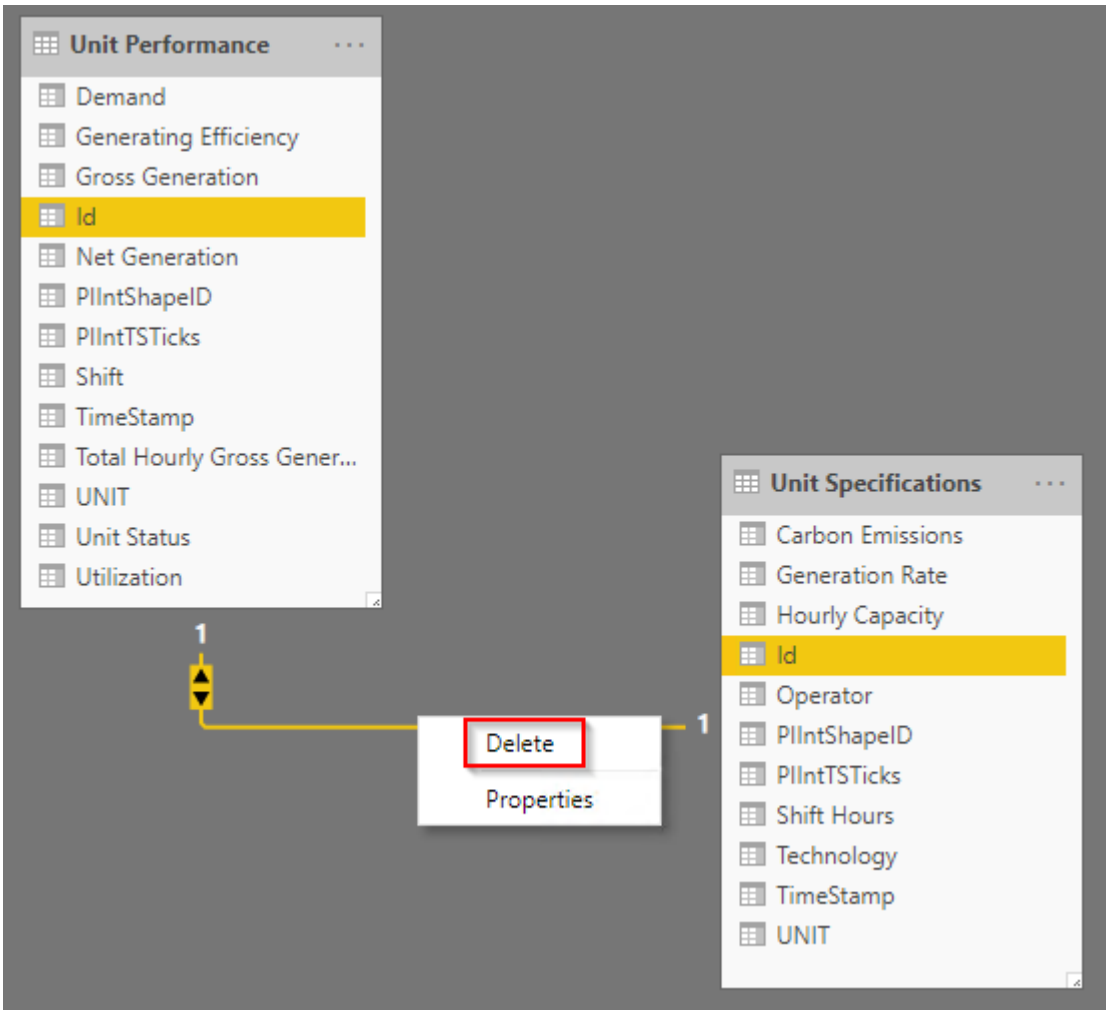
- Delete the automatically created table relationship between Id columns.
- Create a relationship between UNIT columns.

Approach:

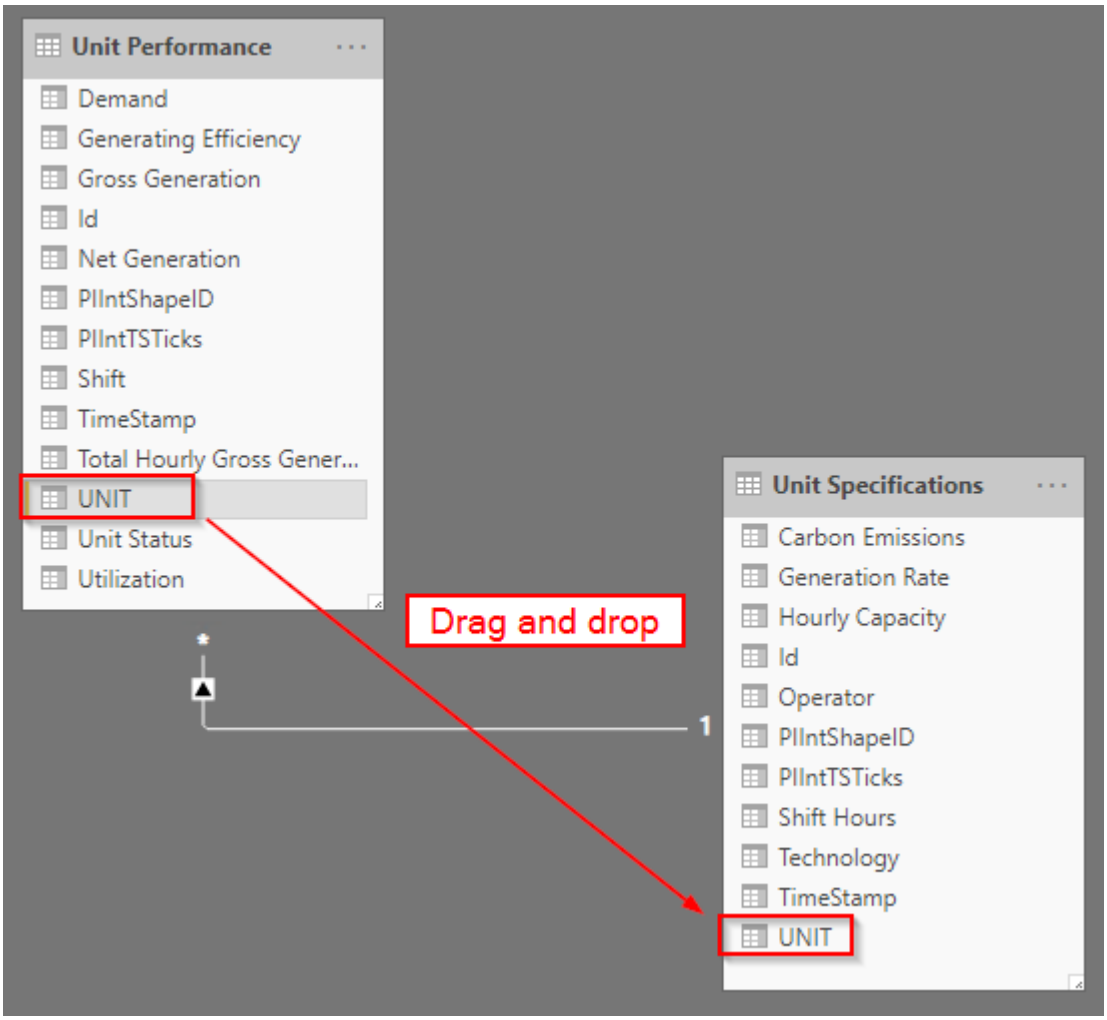
In Power BI, Go to the **Relationships** tab, and click on the line:



This relationship was automatically detected by Power BI and is purely coincidental. The Id is simply the row number. **Right-click -> Delete** the relationship.



Form the correct relationship by **dragging UNIT from one table to UNIT on the other table**. Recall UNITGas Turbine was renamed to UNIT for both tables in a previous step.



Now the Unit Performance table can be filtered based on selections from the Unit Specifications table, for example using a slicer to filter based on Operator.

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.

DAX is a collection of functions, operators, and constants that can be used in a formula, or expression, to calculate and return one or more values. Stated more simply, DAX helps you create new information from data already in your model.

For more information, consult the Microsoft Documentation.

Activity – Calculate the amount of CO2 produced every hour

Objective:

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

Approach:

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.

Activity – Calculate the Generation Cost

Objective:

- Add the cost calculation column to your **Unit Performance** table

Approach:

- Add and additional column named **Cost** to the Unit Performance table with the dollar cost per hour.
- Take note of the input units of measure. Cost should be in dollars. Therefore Cost = Total Hourly Gross Generation (from the Unit Performance table) * Generation Rate (from the Unit Specifications table) * 1000

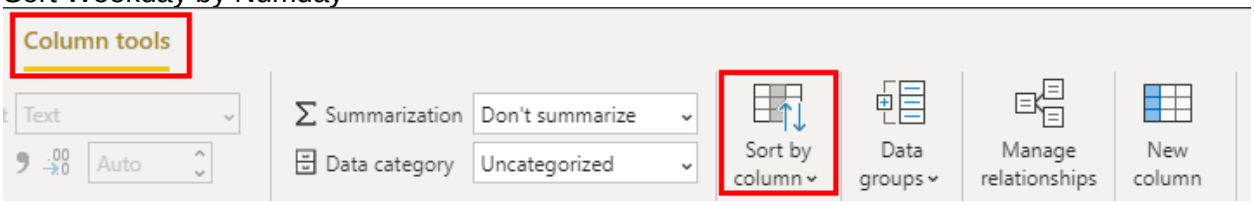
Activity – Add Column for Day of the Week and sort

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

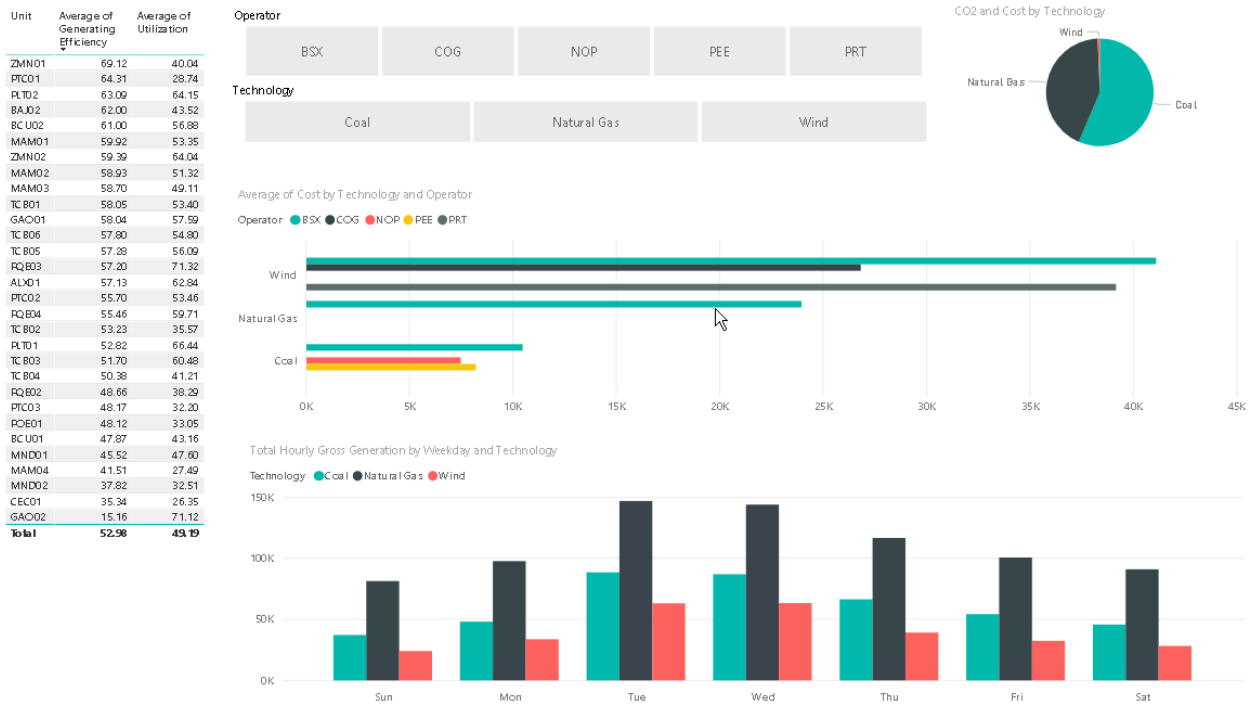


Configuring the Visualizations

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

Activity – Build the Report

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



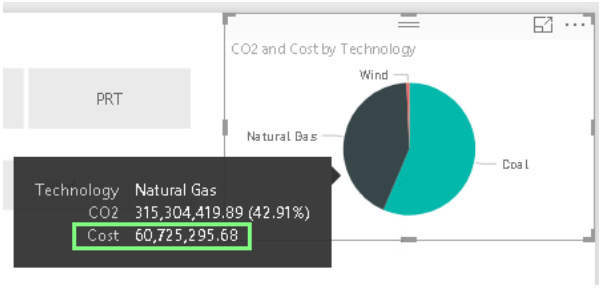
Unit	Average of Generating Efficiency	Average of Utilization
ZMN01	69.12	40.04
PTC01	64.31	28.74
PLT02	63.09	64.15
BAJ02	62.00	43.52
BCU02	61.00	56.88
MAM01	59.92	53.35
ZMN02	59.39	64.04
MAM02	58.93	51.32
MAM03	58.70	49.11
TCB01	58.05	53.40
GAO01	58.04	57.59
TCB06	57.80	54.80
TCB05	57.28	56.09
FQE03	57.20	71.32
ALX01	57.13	62.84
PTC02	55.70	53.46
FQE04	55.46	59.71
TCB02	53.23	35.57
PLT01	52.82	66.44
TCB03	51.70	60.48
TCB04	50.38	41.21
FQE02	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:

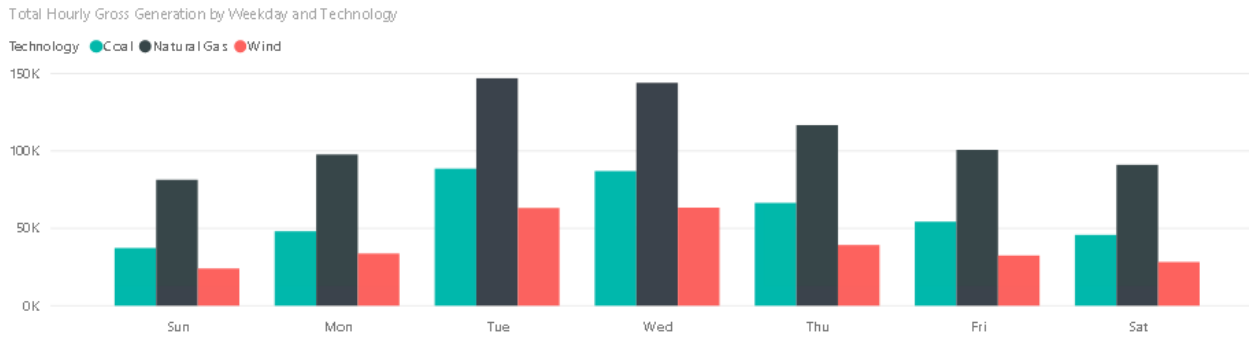
- Add a **Table** showing Average Generating Efficiency and Average Utilization by Unit

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
PQB03	57.20	71.32
ALX01	57.13	62.84
PTC02	55.70	53.46
PQB04	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
PQB02	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**

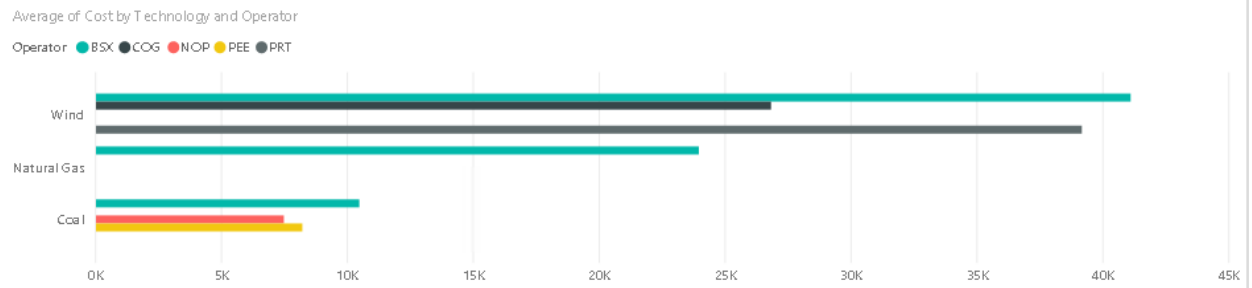
- Add a **Pie Chart** showing how the **C02 emissions** from each generation technology contribute to the whole. Add a **Tooltip** that shows the **Cost** when the user hovers over the Pie Chart



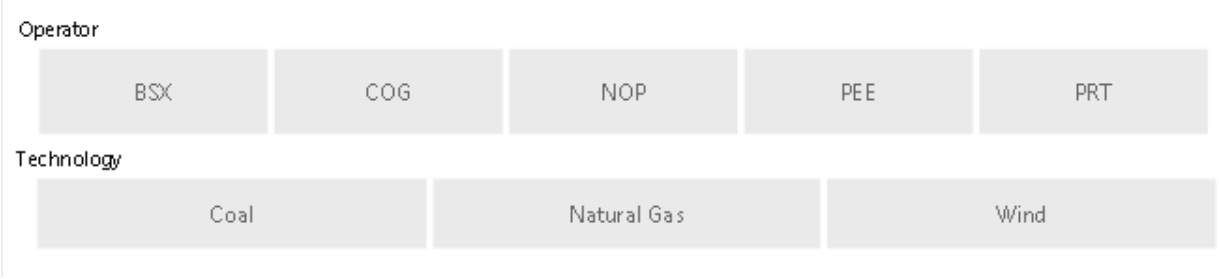
- Add a **Clustered Column Chart** showing the Sum of Total Hourly Gross Generation with Technology as the Legend and Weekday as the Axis



- 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

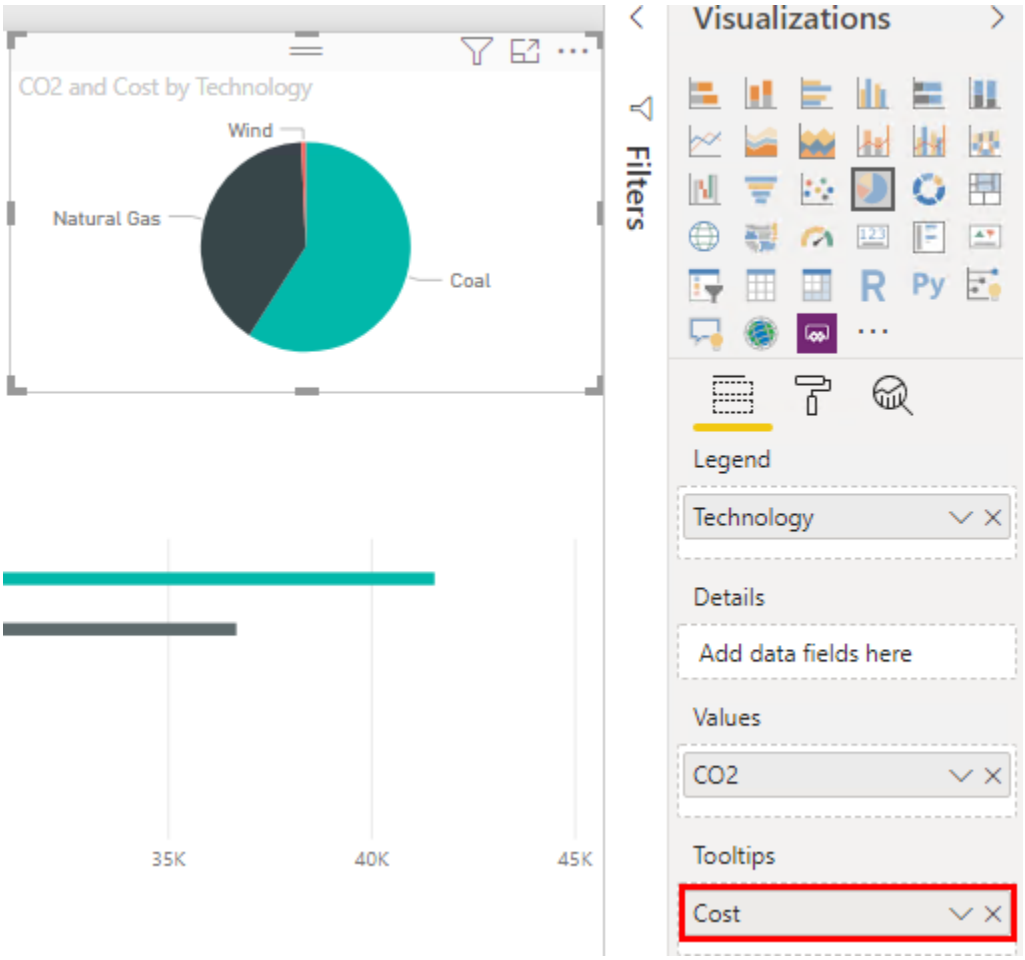


- Optionally improve the look and feel of the report through the use of formatting. Bump up the font sizes, adjust column names and titles, etc.

Solution – Build the Report

Solution file is in **C:\Class\Part 2 - PI OLEDB Enterprise\Solutions**. There should be enough screenshots in the exercise itself.

The only part that hasn't been explicitly covered is adding the tooltip, which can be accomplished by adding the Cost (summarize as Sum) from the Unit Performance table to the Tooltips field:



Final Challenge: Fleet Generation Report with Event Frames

The intent of the final challenge is to test your understanding of the course material by going through the entire report development process on your own but first let's get some practice configuring Event Frame views.

Event frames for the last week may need to be backfilled depending on when the virtual machines were deployed and how long they've been running for.

Activity – Backfill Event Frames in the Online Fleet Generation database

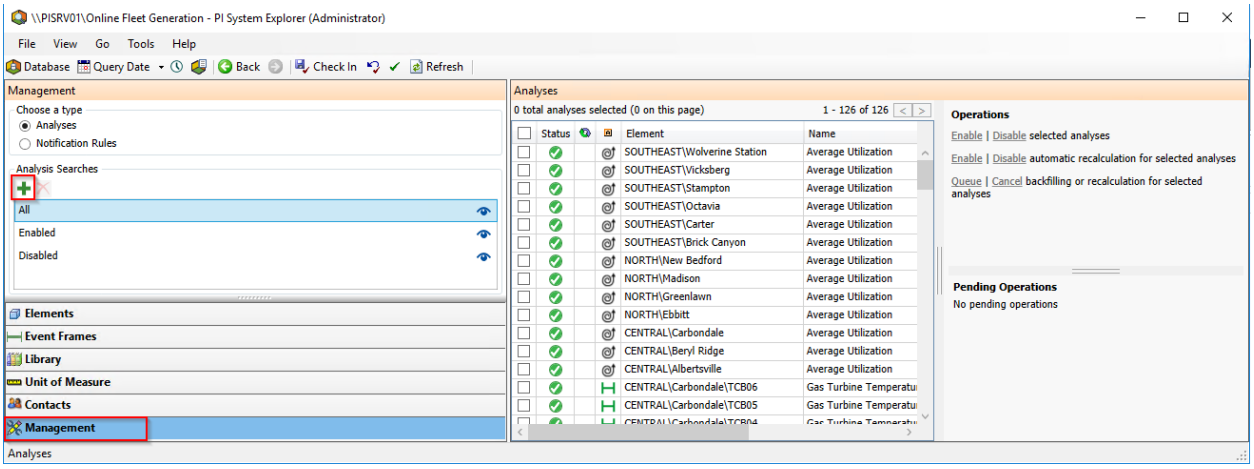
Objective:

- Generate the Gas Turbine Temperature Anomaly and Inactivity Event Frames for use in subsequent exercises.

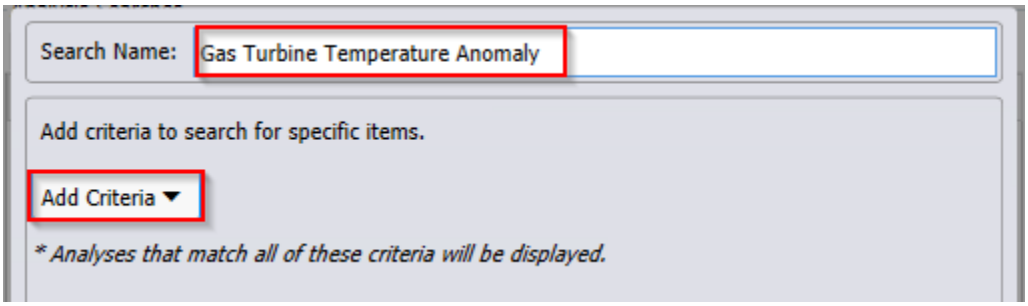
Approach:

Open PI System Explorer and go to the **Online Fleet Generation** database.

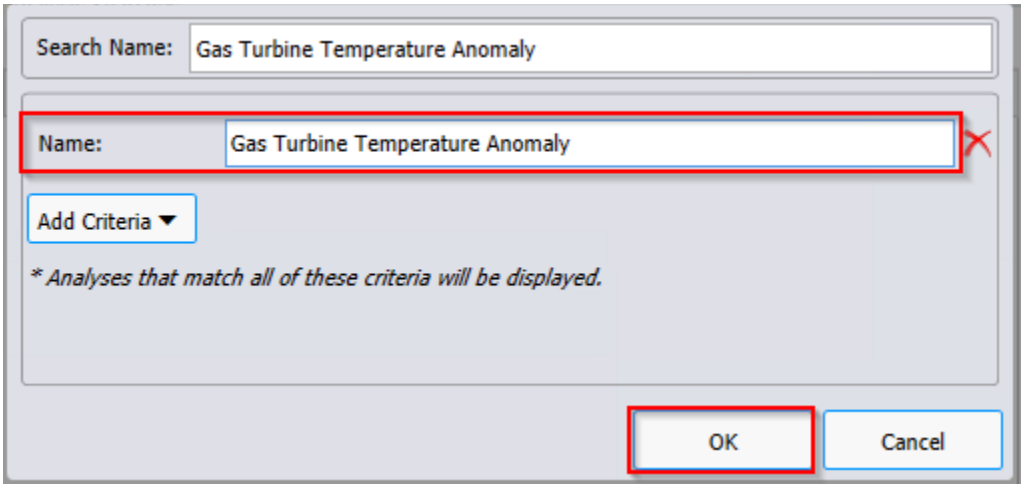
In the **Management** area, add a new **Search**:



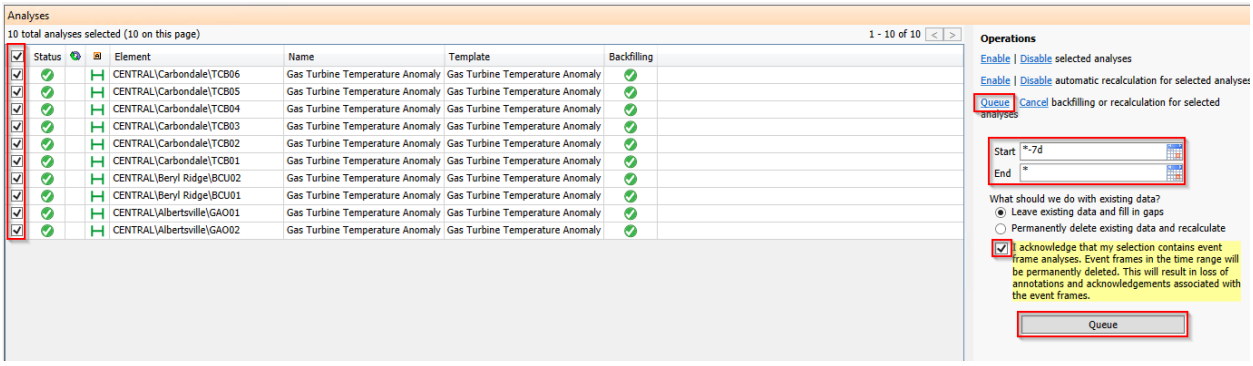
Name it **Gas Turbine Temperature Anomaly**, then **Add Criteria**:



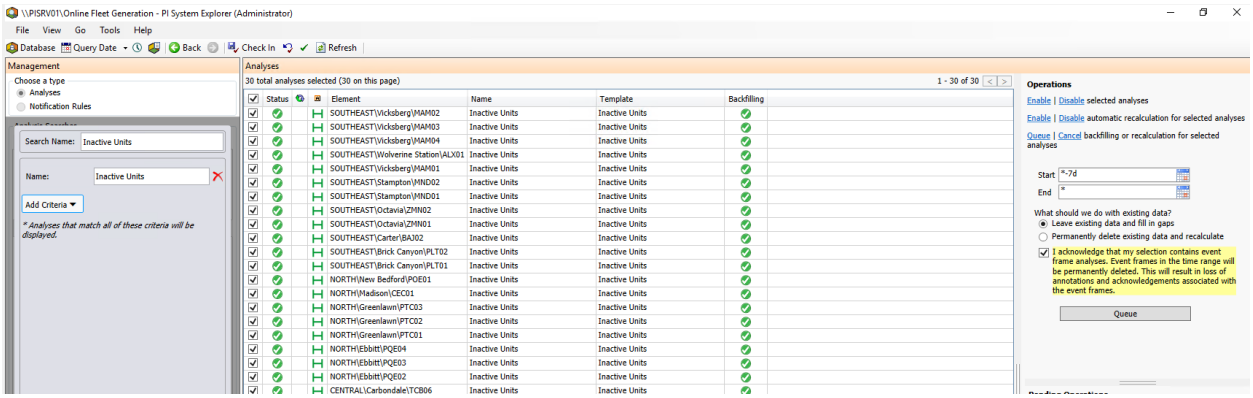
Search on **Name: Gas Turbine Temperature Anomaly** and click OK**:**



Select all analyses, click Queue, enter *-7d as the Start time, check the acknowledgment box, and start the backfill:



Repeat the above process for the Inactive Units Event Frame Analyses:



Activity – Create an Event View with PI Integrator for BA

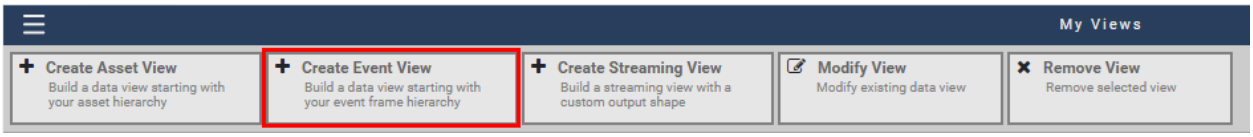
Objective:

- Create an Event View with PI Integrator for BA

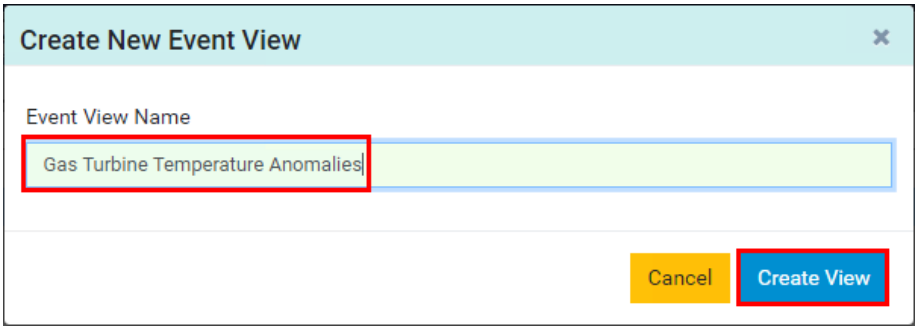
Approach:

We'll create an event view for Gas Turbine Temperature Anomaly events. Open Chrome and navigate to https://pisrv01.pischool.int:444/.

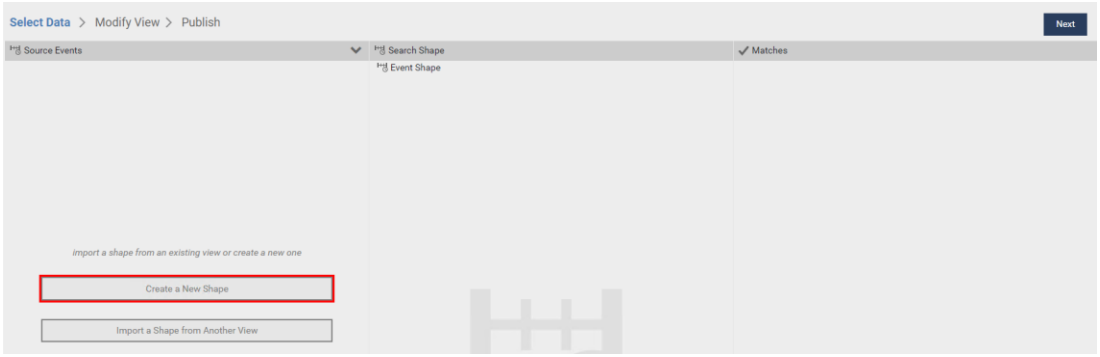
Create an Event View:



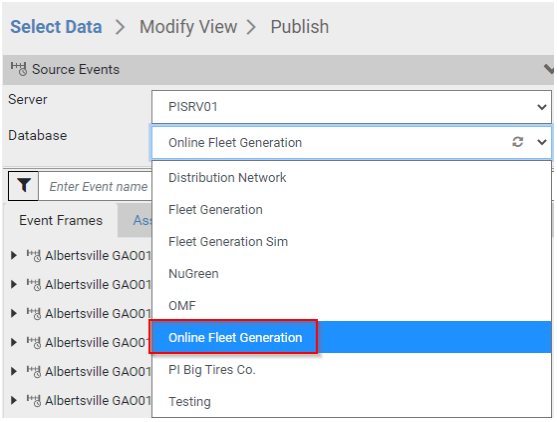
Name the event view **Gas Turbine Temperature Anomalies** and click Create View:



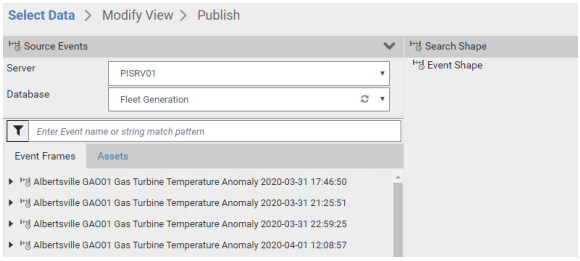
Create a new Shape:



Point at the Online Fleet Generation database, since that's where the Event Frames are:

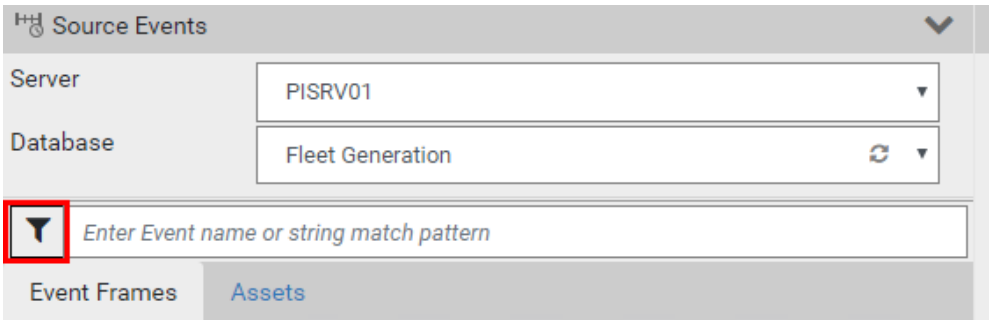


Recent Event Frames should show up in the preview. **Everyone's data is random, so everyone's preview will look different**:

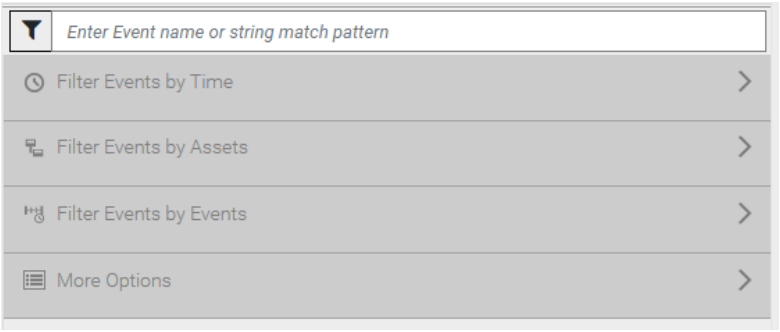


You just need to find an event frame of the proper type in order to start building the shape, but let's look at the filtering options which will allow you to narrow down the search.

Click the filter icon:

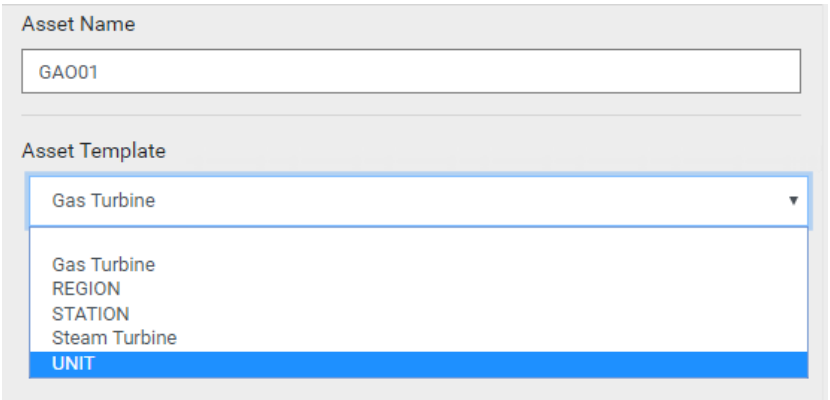


These settings allow filtering the preview and will help you find the event you're looking for. On a production system there could be over a million Event Frames spanning many different types.

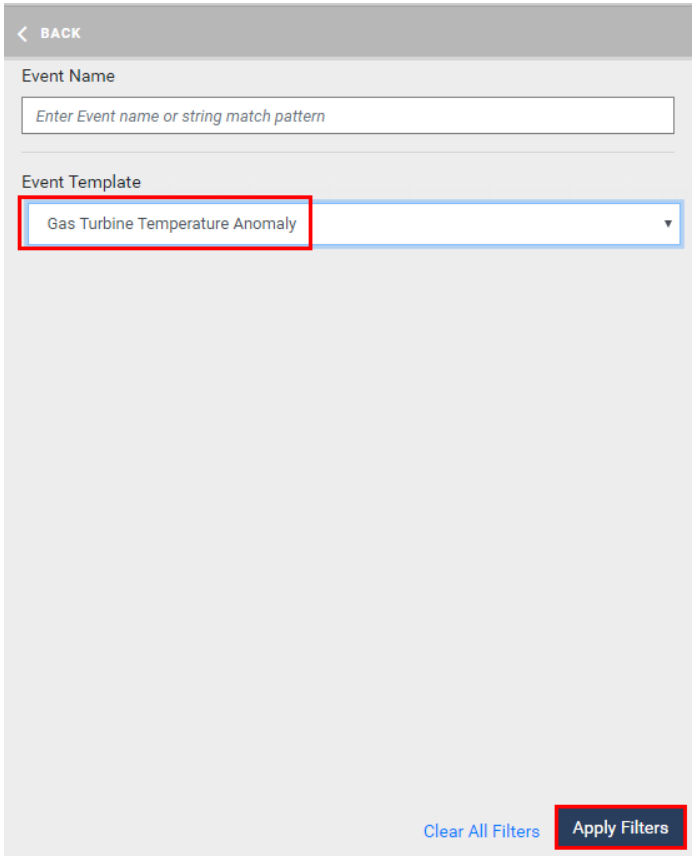


Filter Events by Time is pretty straightforward.

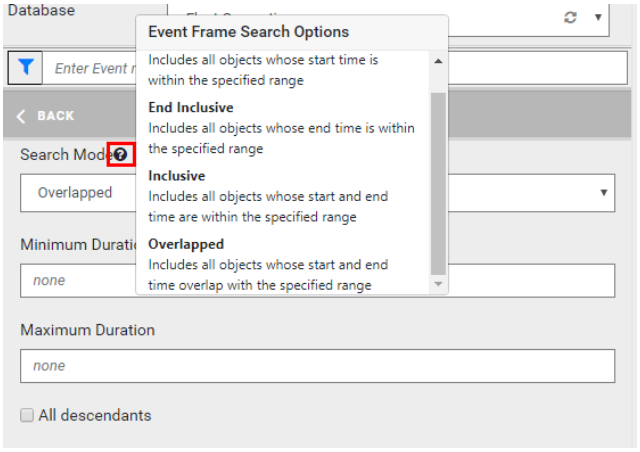
Filter Events by Asset allows you to filter by primary referenced Element (the Element in AF whose Analysis generated the event). **No need to set anything here, this is just an example.**



Filter Events by Events allows you to filter by Event Frame name or Event Frame Template. **Select Gas Turbine Temperature Anomaly** as the Event Template and click **Apply Filters** to filter out the Inactivity Events:

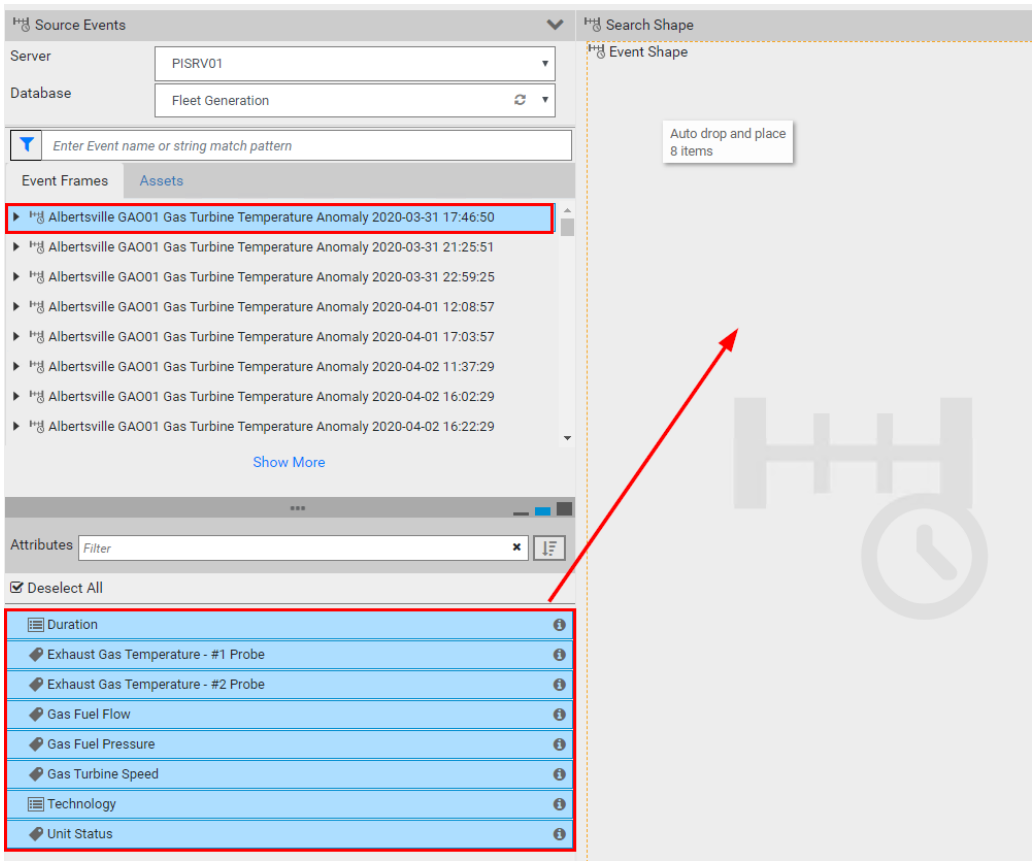


And while we won't set anything, let's take a look at **More Options**. Click the question mark to see explanations for the **Search Mode**. Minimum Duration and Maximum Duration are self-explanatory. **All descendants** applies to hierarchical event frames, which we don't have.

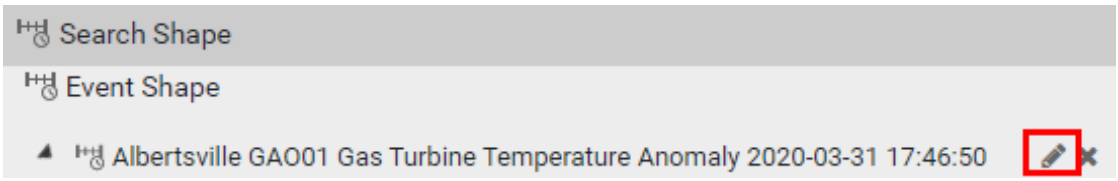


If you can't find Gas Turbine Temperature Anomaly events, it's possible that they weren't backfilled. Check to see if they exist in the Online Fleet Generation database through PI System Explorer and troubleshoot as necessary.

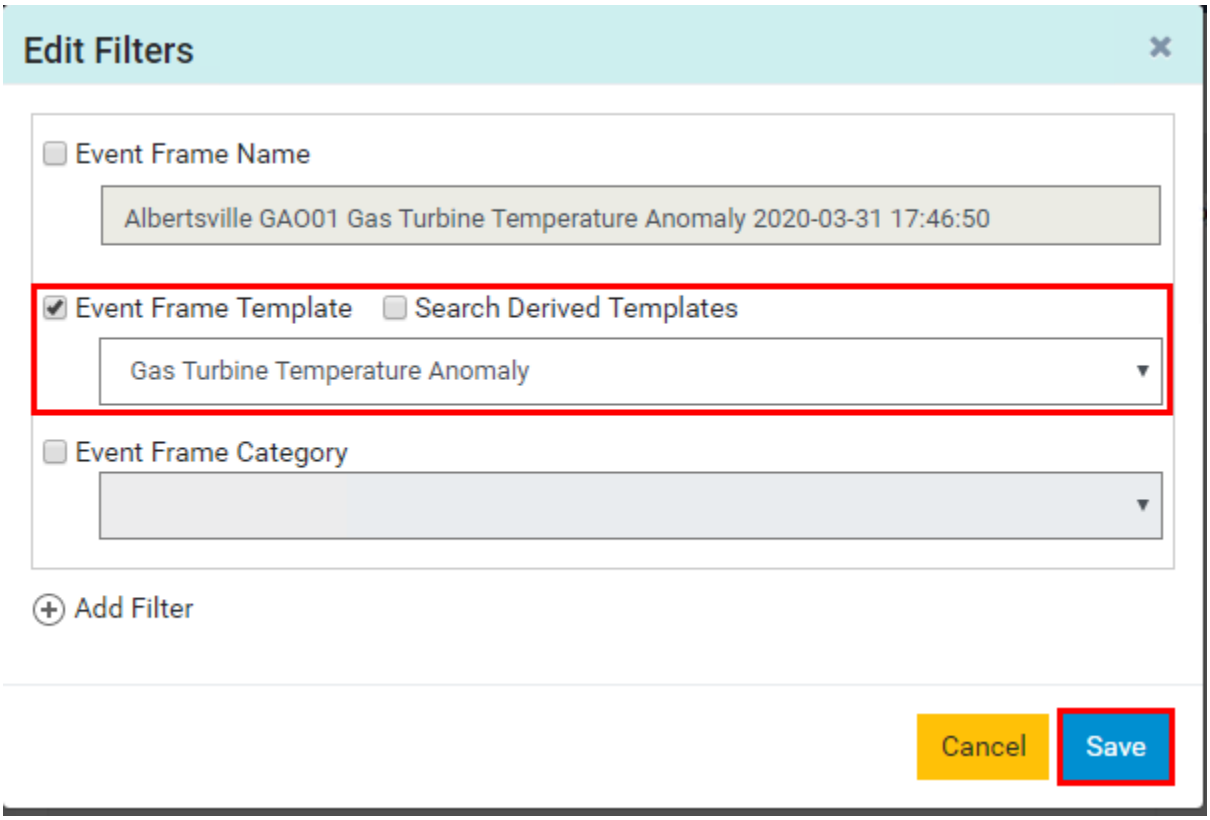
Once you see some events, you can start to configure the Shape. Click one of the Events, then drag and drop all Attributes:



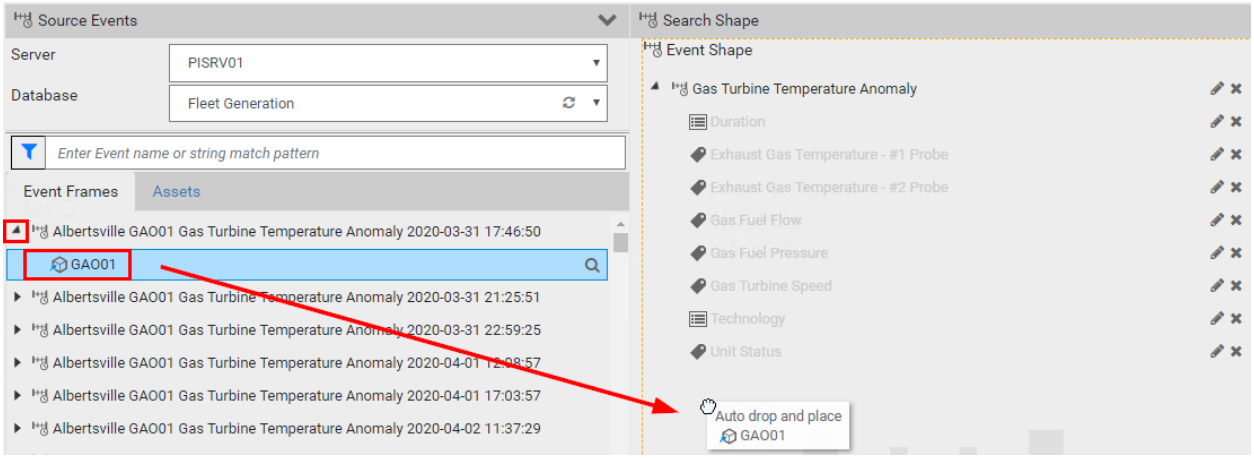
Edit the Shape:



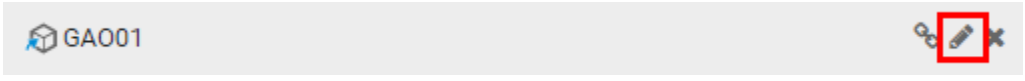
Uncheck the box next to Event Frame Name and match Event Frames by Template then Save.



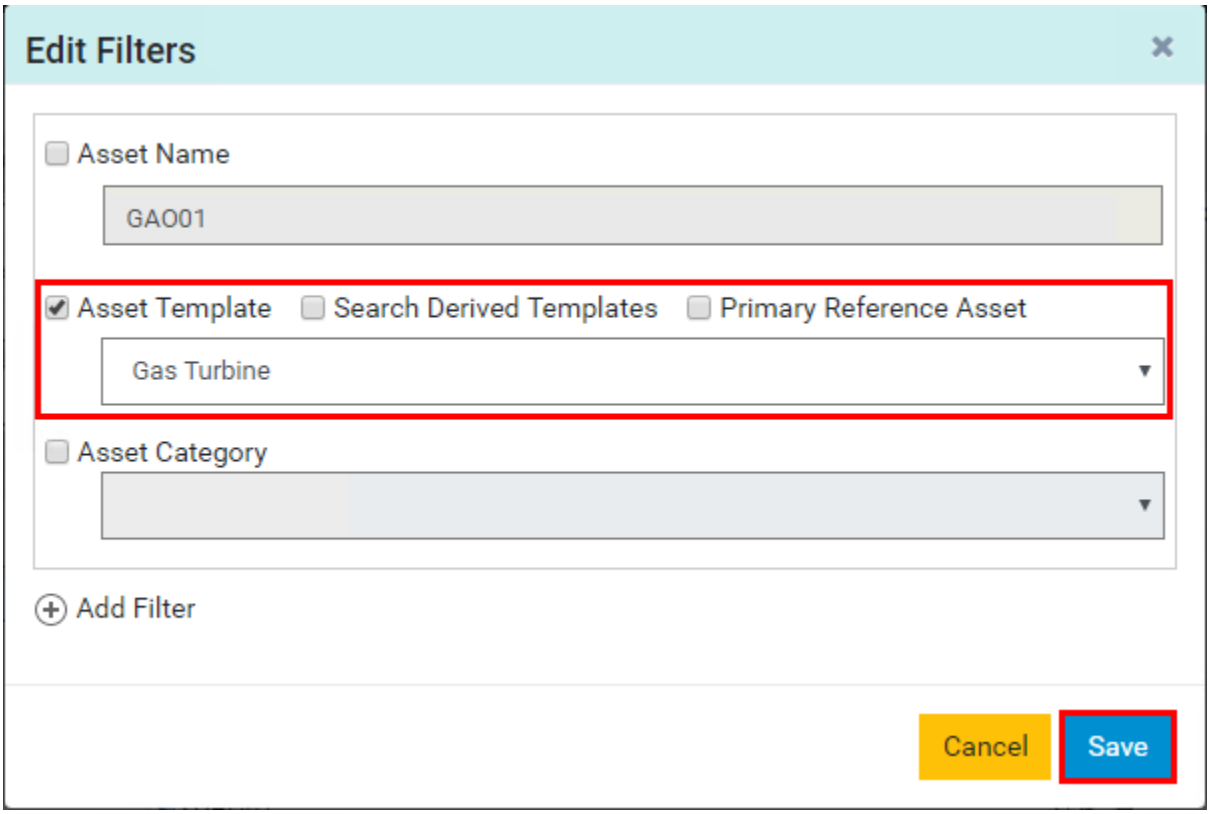
Expand the Event Frame and drag and drop the Gas Turbine to the shape configuration:



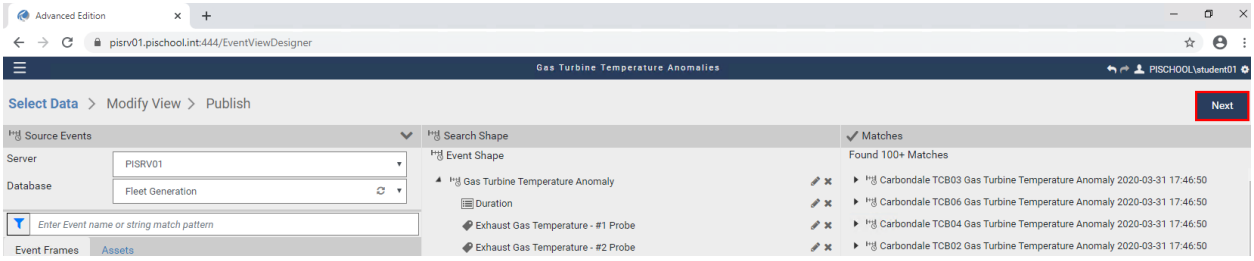
Edit the Gas Turbine object:



Filter by Asset Template, click Save:

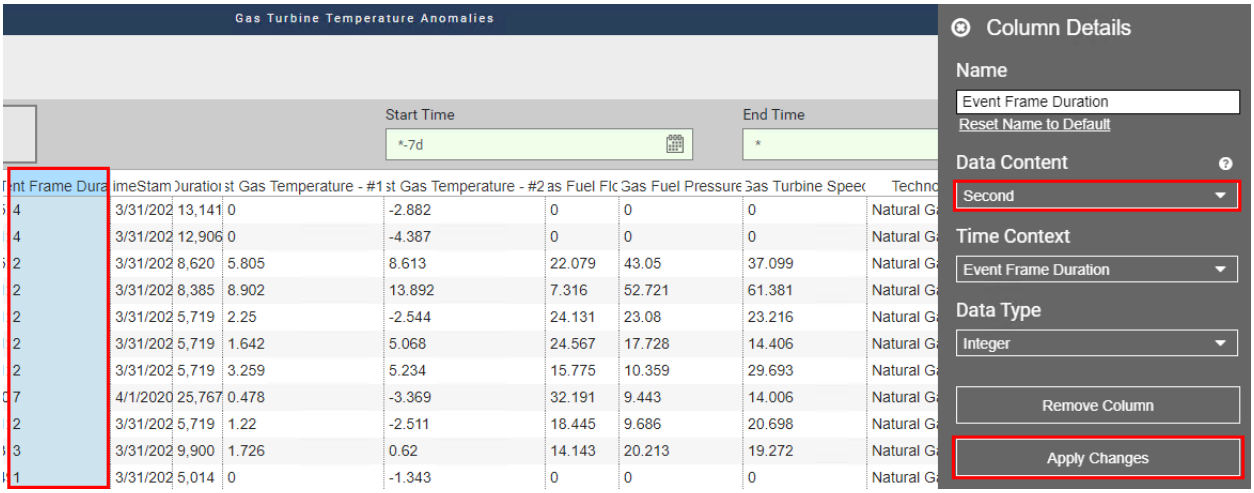


You should see a bunch of matches. Go to the Next screen:

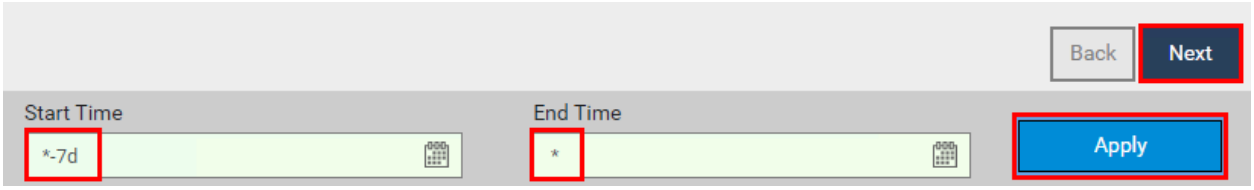


Change the Event Frame Duration Data Content to Second otherwise it will be displayed as a round number of hours.

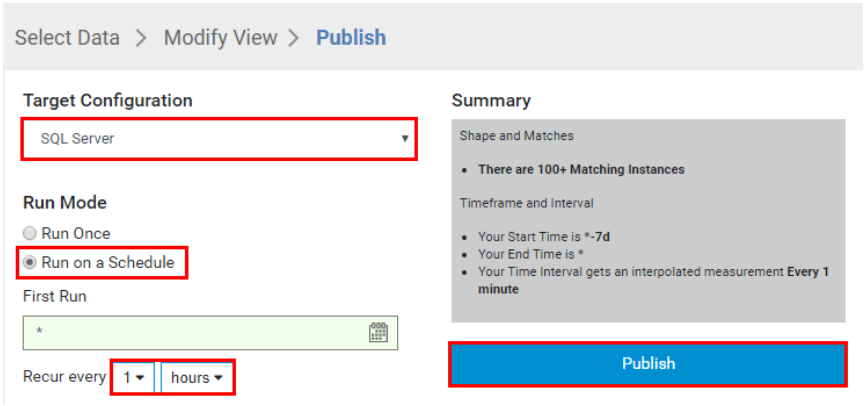
Be sure to click Apply Changes.



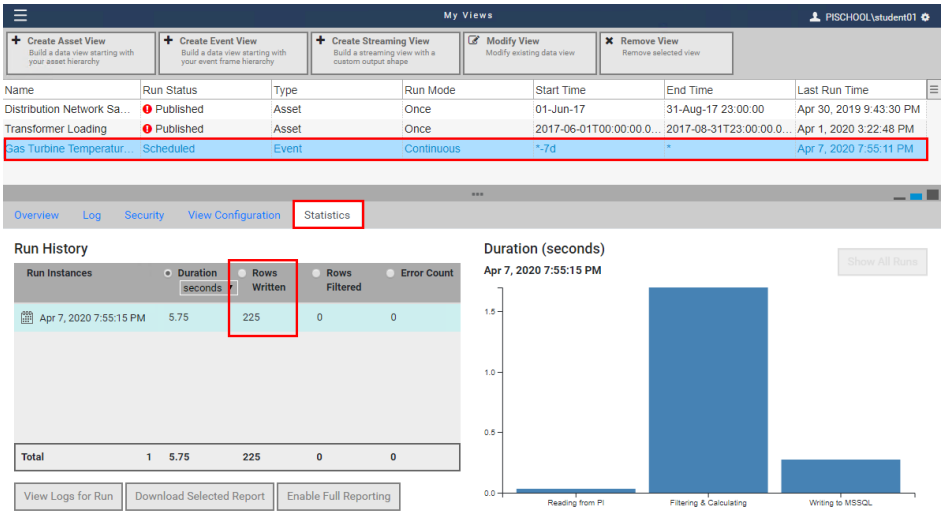
Change the Start Time to *-7d and end time to *, click Apply, then move to the Next Screen:



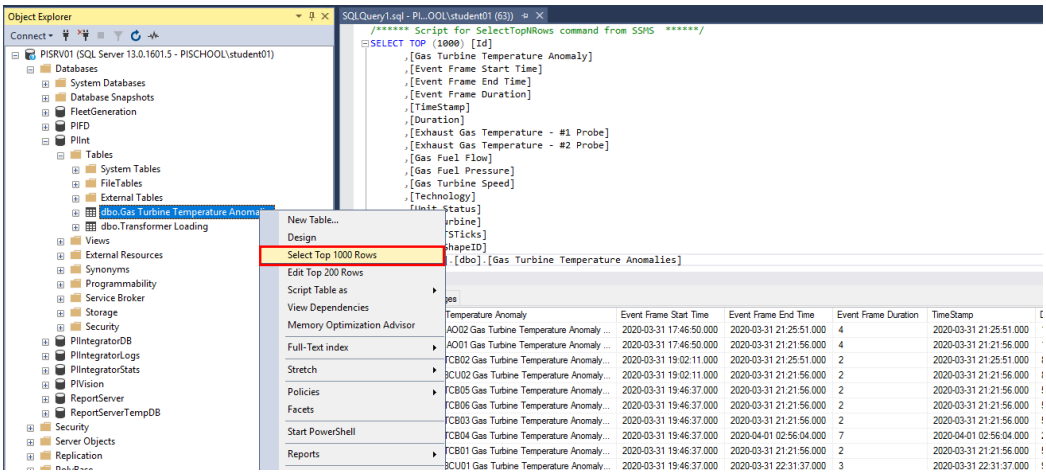
Select SQL Server as the target and have it run on an hourly schedule to keep the Event Frames current, then click Publish.



Check the statistics to confirm that Rows were written:



Optionally use **SQL Server Management Studio** to confirm that Event Frames were written to the SQL Server table:



Activity – Build the Final Report

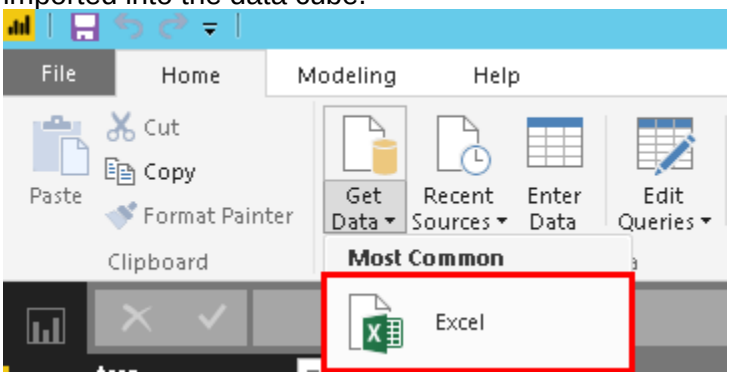
This is your chance to build a report from scratch using the concepts learned earlier in the course. Let's see if you can remember how to do everything!

Objectives:

- Incorporate downtime (Inactivity) events
- Determine the carbon footprint of each unit and display on a US map.

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.



.

- Use PI Integrator for BA to publish an Event View for Inactivity events
  - 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).
  - Configure a continuous Event View that updates hourly. The initial publication should include the last 7 days

- Import the Inactivity events for the last 7 days using the above Event View

- 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 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.

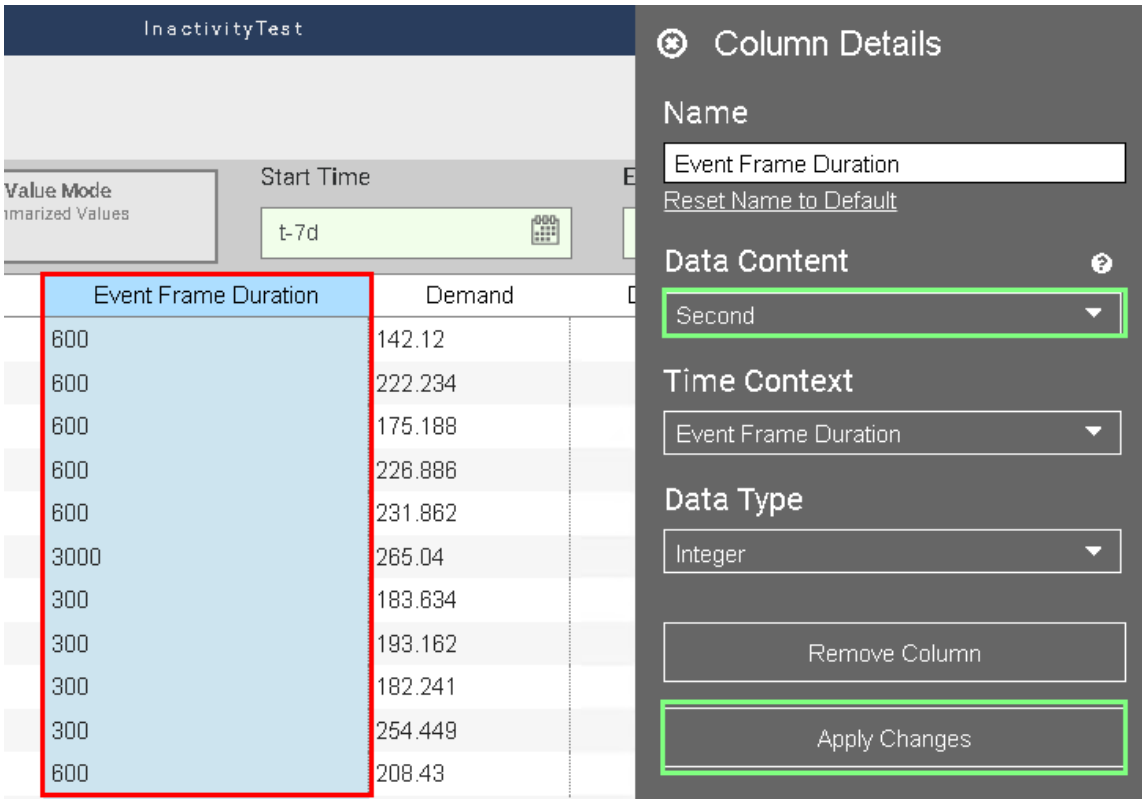
- Insert a map within the client to display the location of each of the units and the associated total hourly carbon emissions.

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

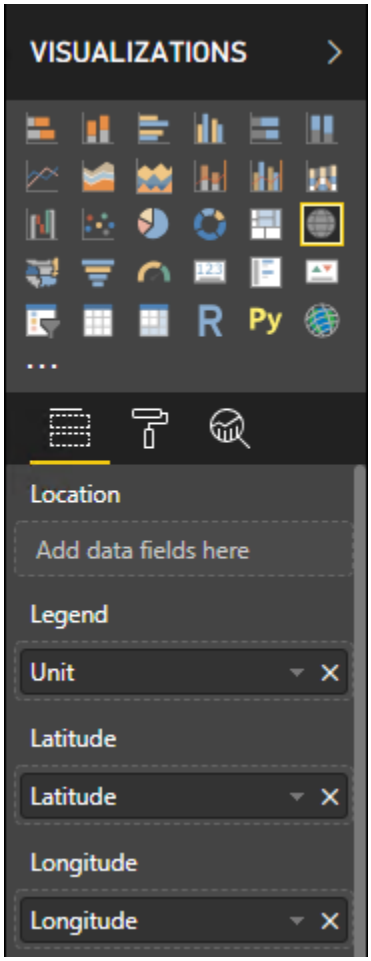
- Customize the display to make it more user friendly by improving the formatting and adding slicers.

Hints:

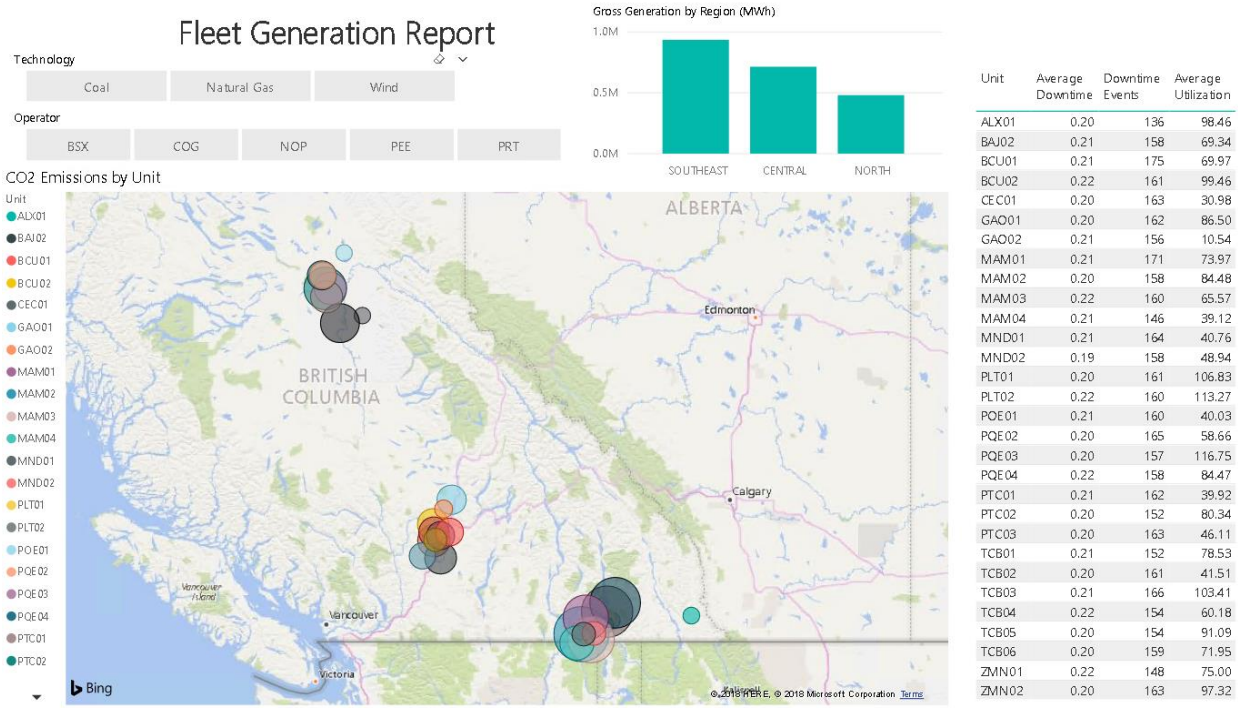
- When configuring the Event View, the Data Context must be set to Second or else it will round to the nearest whole hour (which will always be zero).



- Use the ordinary map, not the ESRI one. Drag and drop latitude and longitude from the table that was imported from the Unit Coordinates.xlsx spreadsheet.



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**