

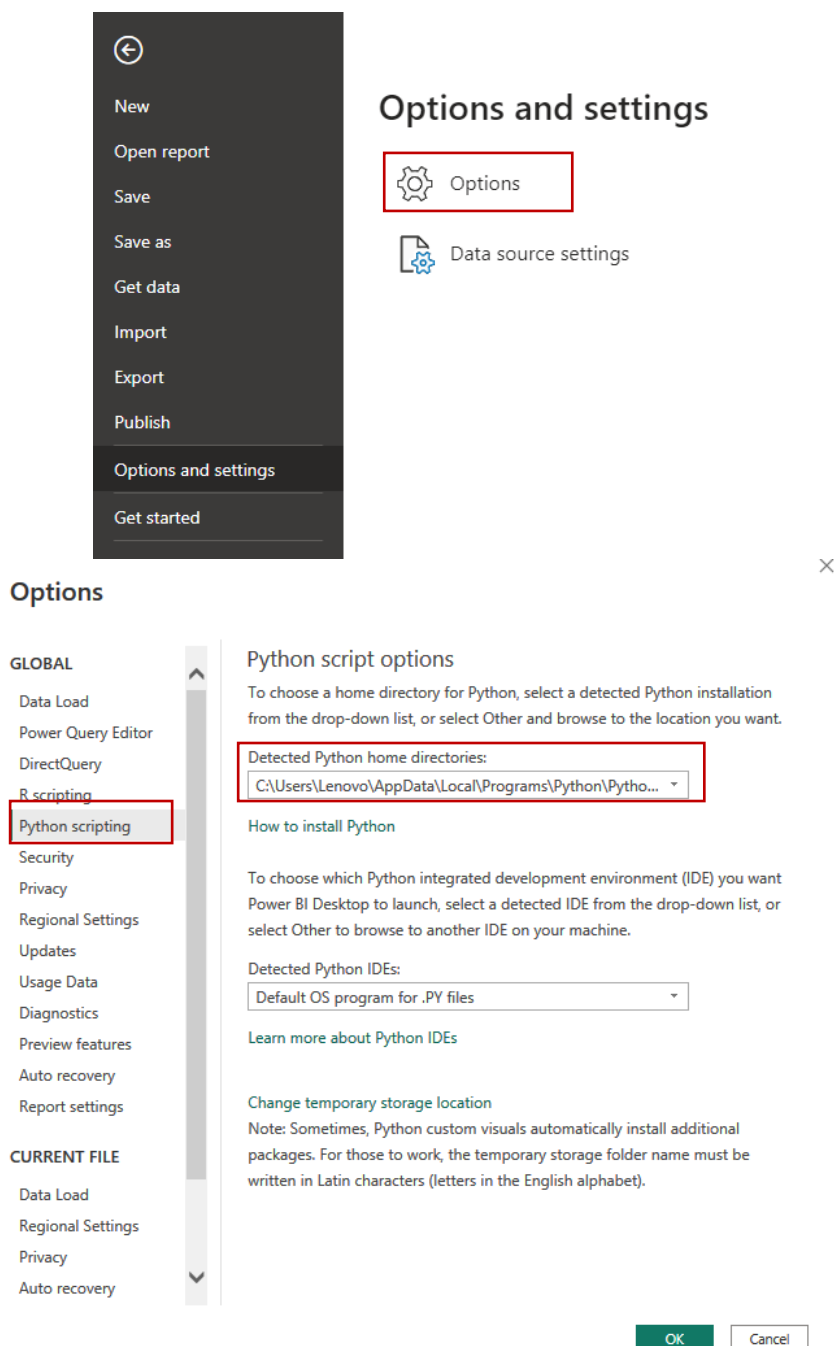
INDICE

INTRODUCCIÓN A PYTHON Y POWER BI	2
Activar Python en Power BI	2
Power BI - Ejercicio	3
1. Conjunto de datos	3
2. Limpieza, Transformación y Modelado	4
3. Visualización y DAX.....	6
a) Corporativo.....	6
b) Mapa de Calor	12
c) Tendencia	14
d) Correlación	21
d) Modelos.....	25
e) Predicción	29
OVER VIEW	34

INTRODUCCIÓN A PYTHON Y POWER BI

Activar Python en Power BI

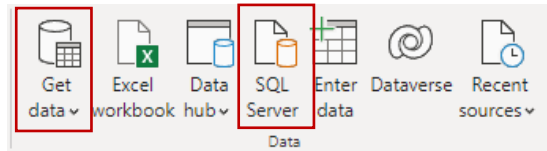
En Power BI, seleccione "Archivo" en la barra de menú y haga clic en "Opciones y configuración". En la ventana emergente, seleccione "Opciones" y luego ubicamos "Python scripting". Aquí, habilite la opción "Usar la versión de Python instalada en mi sistema" y luego seleccione la ruta de acceso a la instalación de Python (por defecto Power BI ubica la ruta general de usuario, en caso de no encontrarla se debe proporcionar la ruta en el cual fue instalado Python).



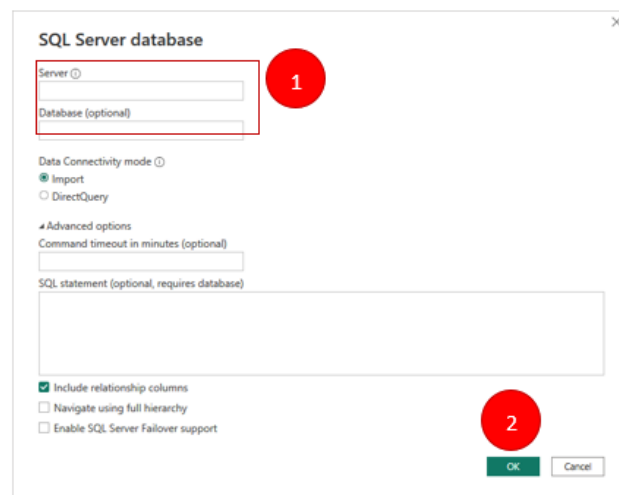
Power BI - Ejercicio

1. Conjunto de datos

Para importar un conjunto de datos desde SQL Server a Power BI, primero debemos tener acceso al servidor de la base de datos. Luego, en Power BI Desktop, seleccionamos "Obtener datos" en la pestaña de inicio y elegimos "SQL Server" como origen de datos; También podemos acortar este paso, desde la cinta de opciones principal seleccionando la opción de "SQL Server".



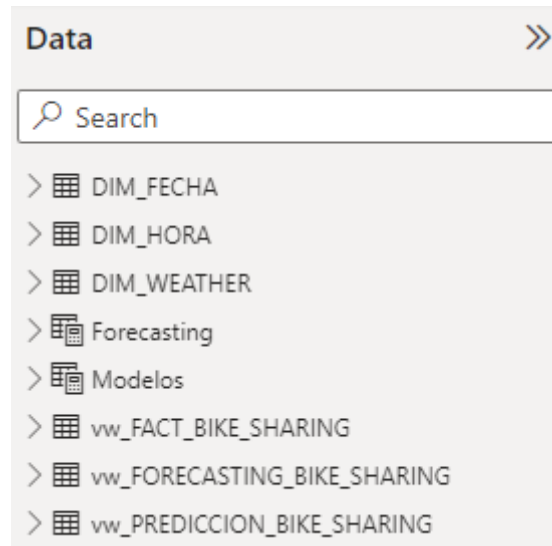
Luego ingresamos los detalles de la conexión; el servidor y colocamos el nombre de la base de datos "DMT_BIKE_SHARING", como dato adicional en las opciones avanzadas se puede colocar un query sql para realizar importaciones de tablas específicas.



A continuación, se nos muestra una ventana en la cual seleccionaremos las tablas a cargar; al realizar la primera carga de datos, en donde seleccionaremos las siguientes vistas y tablas.

Nombre	Tipo
DIM_FECHA	Table
DIM_WEATHER	Table
DIM_HORA	Table
vw_FACT_BIKE_SHARING	View
vw_PREDICCIÓN_BIKE_SHARING	View
vw_FORECASTING_BIKE_SHARING	View

Una vez realizada la carga del conjunto de datos (Dimensiones y Hechos), se nos mostraran las tablas en el panel “Data”

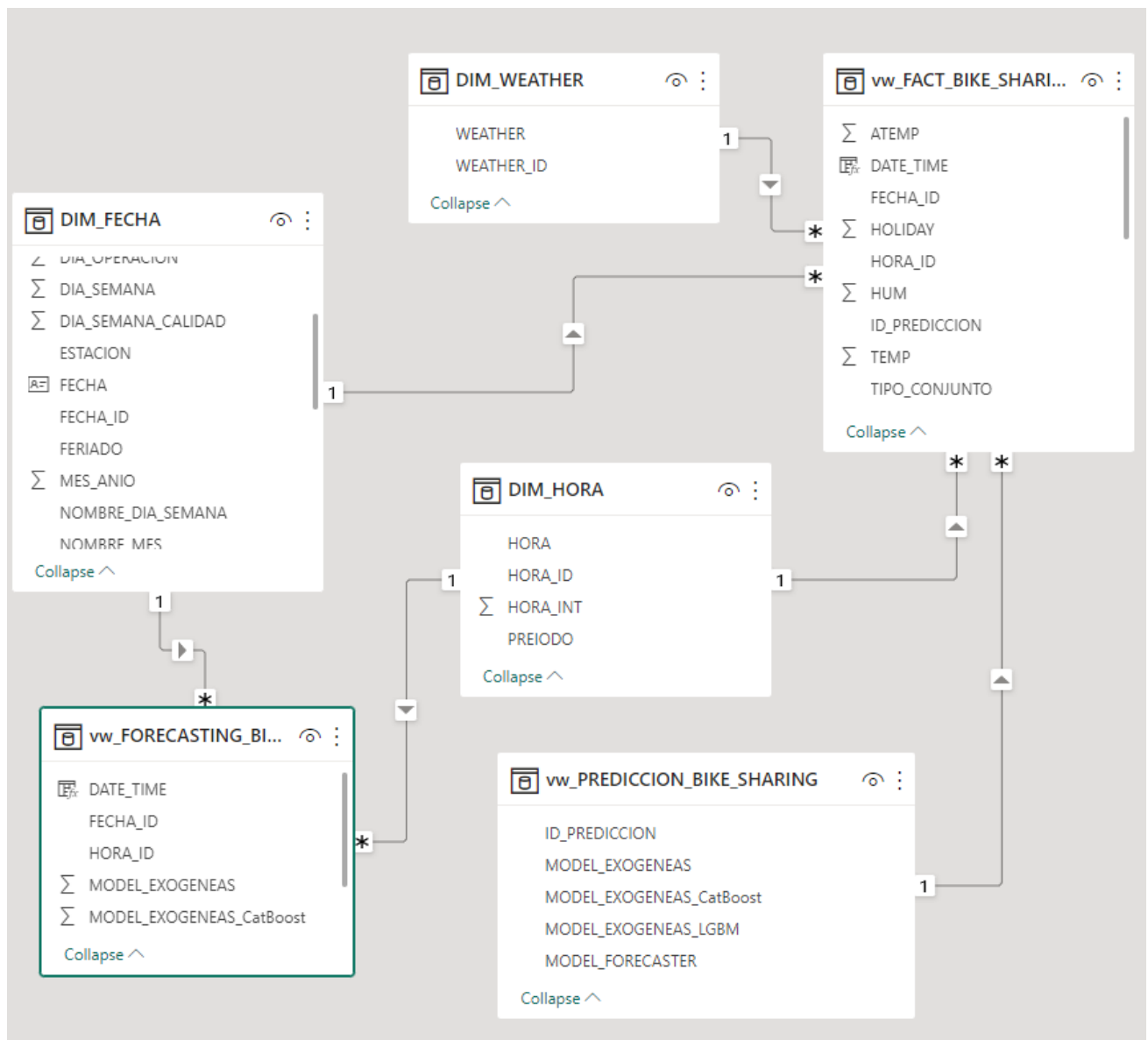


2. Limpieza, Transformación y Modelado

La presente fase se compone de los siguientes aspectos:

- La limpieza de datos implica identificar y corregir errores y valores atípicos en los datos
- La transformación de datos implica cambiar el formato o la estructura de los datos para que sean más útiles para el análisis.
- El modelado de datos implica la creación de modelos que representen los datos y permitan el análisis y la predicción

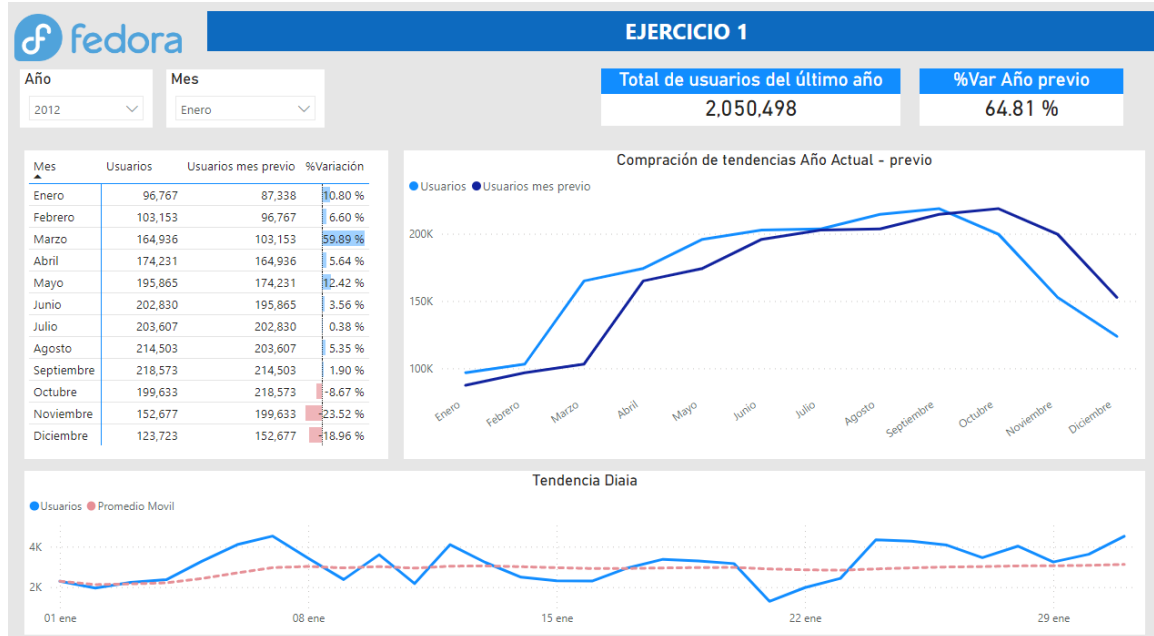
Como podemos notar, estos aspectos fueron realizados en la creación de la base de datos DMT_BIKE_SHARING, Tendiendo como resultado en el apartado “modelo” las tablas correctamente relacionadas.



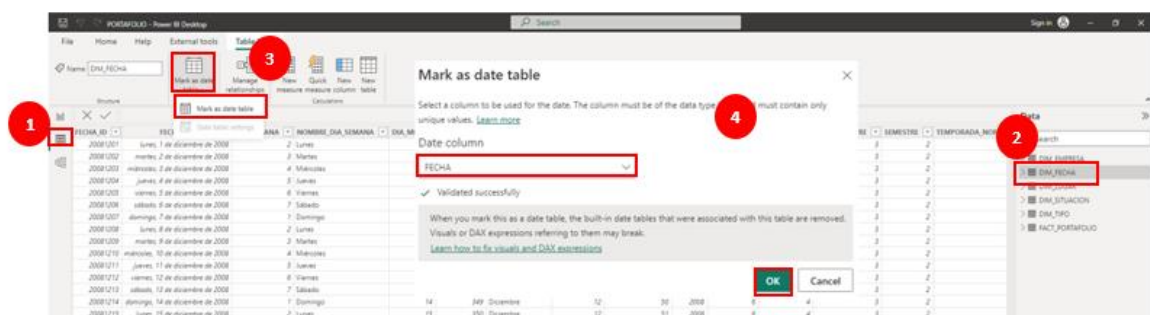
3. Visualización y DAX

a) Corporativo

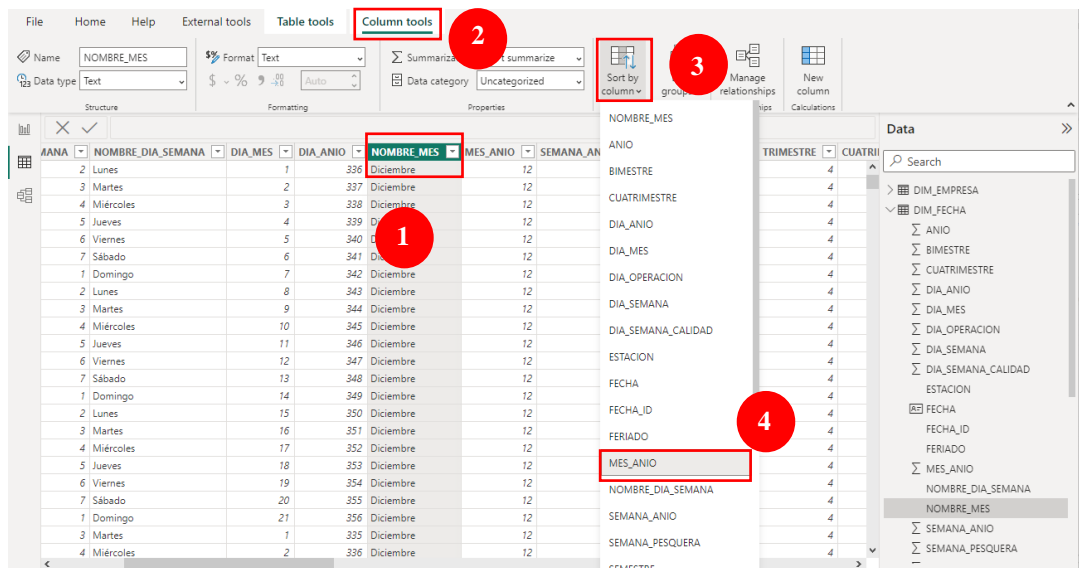
- ✓ Creamos una pestaña llamada “Corporativo”
- ✓ Creación de métricas basadas en fechas



- Lo primero que haremos será definir nuestra tabla FECHA, ya que en todo proyecto debe haber como mínimo una tabla FECHA. Para ello nos vamos a la capa de DATOS y seleccionamos nuestra tabla DIM_FECHA. Luego en la pestaña de herramientas de tablas, marcamos la tabla como tabla de fechas.



- También podemos crear un orden sobre las columnas, esto con el objetivo de graficar correctamente gráficos basados en las fechas (cronológicamente). Ordenamos la columna NOMBRE_MES en base a la columna MES_ANIO.
- Comenzamos seleccionando la columna NOMBRE_MES de la DIM_FECHA, nos posicionamos en la cinta de opciones Column tools y buscamos la opción de Sort by column, en donde seleccionaremos la columna MES_ANIO.



- Para tener control de los datos, comenzamos creando los filtros de Año (DIM_FECHA) y Mes (DIM_FECHA).
 - **Filtro de año;** se usa la columna ANIO de la tabla DIM_FECHA, se describen las configuraciones a usar:
 - En la sección de Visual,
 - desactivamos el Slicer header
 - En la opción Slicer setting, Option, seleccionamos el Style en “Dropdown”
 - En la opción Slicer setting, Selecion, aplicamos la opción “select all”
 - En la sección General
 - Activamos la opción de Title y definimos el título “Año”
 - **Filtro de Mes;** se hace uso de la columna NOMBRE_MES de la tabla DIM_FECHA, se describen las configuraciones a usar:
 - En la sección de Visual,
 - desactivamos el Slicer header
 - En la opción Slicer setting, Option, seleccionamos el Style en “Dropdown”
 - En la opción Slicer setting, Selecion, aplicamos la opción “select all”
 - En la sección General
 - Activamos la opción de Title y definimos el título “Mes”
 - Procedemos a usar el tipo de filtro Top N en cada uno de los filtros creados; la columna a usar será el ID que hace la unión entre la vw_FACT_BIKE_SHARING Y la respectiva DIM_FECHA

ANIO
top 100 by Count of F...

Filter type ⓘ
Top N

Show items
Top 100

By value
Count of FECHA_ID

Apply filter

NOMBRE_MES
top 15 by Count of FE...

Filter type ⓘ
Top N

Show items
Top 15

By value
Count of FECHA_ID


Apply filter

- **No olvide seleccionar la opción de *Apply filter* luego de haber colocado las opciones
- **el valor a mostrar dependerá del análisis previo realizado.


- Antes de empezar a desarrollar las gráficas basadas en Python, podemos agregar un logo, un *text box* el cual tendrá como título “EJERCICIO 1”; Teniendo preparada la tabla y los filtros, comenzaremos a crear las métricas las cuales deben ser creadas en la tabla vw_FACT_BIKE_SHARING.

Métrica	Formato	Descripción
.DIA = MAX(DIM_FECHA[DIA_MES])	Whole number	Obtener el máximo de la columna DIA_MES
.sum_users = SUM(vw_FACT_BIKE_SHARING[USERS])	Whole number	Obtener la suma de la columna USERS
.user_moth = TOTALMTD([.sum_users], DIM_FECHA[FECHA])	Whole number	Obtener el total acumulado del mes asociado a la métrica
.users_total_year = TOTALYTD([.sum_users], DIM_FECHA[FECHA])	Whole number	Obtener el total acumulado del año asociado a la métrica
.users_prev_month = TOTALMTD([.user_moth], PREVIOUSMONTH(DIM_FECHA[FECHA]))	Whole number	Obtener el total acumulado del mes previo asociado a la métrica
.users_prev_year = TOTALMTD([.users_total_year], PREVIOUSYEAR(DIM_FECHA[FECHA]))	Whole number	Obtener el total acumulado del año previo asociado a la métrica
.%var_users_yearr = DIVIDE([.users_total_year] - [.users_prev_year], [.users_prev_year])	Percentage	División de las métricas, resultando en el porcentaje de la varianza
.%var_users_month = DIVIDE([.user_moth] - [.users_prev_month], [.users_prev_month])	Percentage	División de las métricas, resultando en el porcentaje de la varianza
.promedio_movil = DIVIDE([.user_moth], [.DIA])	Whole number	División de las métricas obteniendo un promedio móvil en base a las posiciones


**No olvide aplicar la coma de miles en los datos numéricos

- Comenzamos creando una matriz con el icono  y a continuación, el detalle de su contenido
 - Métricas:

	Métrica	Alias
Columns	NOMBRE_MES	Mes
Values	1. [.user_moth] 2. [.users_prev_month] 3. [%var_users_month]	1. Usuarios 2. Usuarios mes previo 3. %Variación

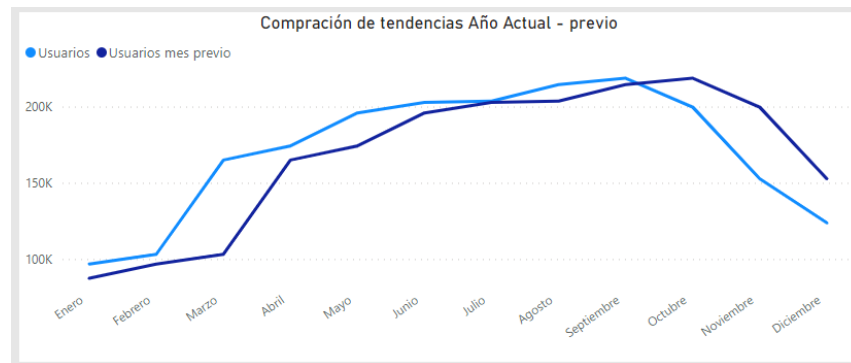
- En la sección de Visual,
 - Style presets, lo establecemos en Minimal.
 - Cell elements, seleccionamos en Apply setting la Series: %Variación.
 - Activamos la opción de Data bars.
 - Seleccionamos el icono .
 - Establecemos el color #A0D1FF en la opción Positive bar.
 - Establecemos el color ##EFB5B9 en la opción Negative bar.

Mes	Usuarios	Usuarios mes previo	%Variación
Enero	96,767	87,338	10.80 %
Febrero	103,153	96,767	6.60 %
Marzo	164,936	103,153	59.89 %
Abril	174,231	164,936	5.64 %
Mayo	195,865	174,231	12.42 %
Junio	202,830	195,865	3.56 %
Julio	203,607	202,830	0.38 %
Agosto	214,503	203,607	5.35 %
Septiembre	218,573	214,503	1.90 %
Octubre	199,633	218,573	-8.67 %
Noviembre	152,677	199,633	-23.52 %
Diciembre	123,723	152,677	-18.96 %

- Ahora crearemos un gráfico de líneas seleccionando el icono  y a continuación, el detalle de su contenido
 - Métricas:

	Métrica	Alias
X-axis	NOMBRE_MES	Mes
Y-axis	1. [.user_moth] 2. [.users_prev_month]	1. Usuarios 2. Usuarios mes previo

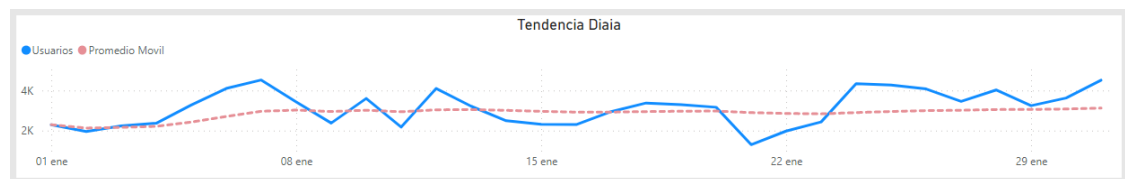
- En la sección de Visual,
 - X-axis, Desactivamos Title.
 - Y-axis, Desactivamos Title.
- En la sección General,
 - Title, colocamos el título “Compración de tendencias Año Actual - previo”




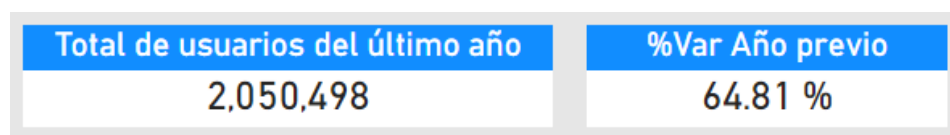
- Duplicamos este último grafico de líneas, le colocamos el título “Tendencia y promedio móvil” y cambiamos las métricas

- Métricas:





	Métrica	Alias
X-axis	FECHA	Mes
Y-axis	1. [.sum_users] 2. [.promedio_movil]	1. Usuarios 2. Promedio Movil



- Ahora crearemos una tarjeta seleccionando el icono  y a continuación, el detalle de su contenido
 - Usaremos la Metrica: [.users_total_year]
 - En la sección de Visual,
 - Desactivamos la opción de Category label
 - Callout value
 - Display units: None
 - Font: 20
 - En la sección General,
 - Title, colocamos el título “Total de usuarios del último año”
 - Text color: blanco
 - Background color: #118DFF
- Finalmente duplicamos la tarjeta y cambiamos la métrica por [.%var_users_yearr] y el titulo a “%Var Año previo”.



- Teniendo todos los gráficos creados, debemos deshabilitar la interacción del filtro Mes con los gráficos etiqueta, matriz, y grafico de líneas (concretamente el grafico comparación de tendencias); para ello seleccionamos el filtro mes, luego nos dirigimos a

la cinta de opciones, Format y seleccionamos la opción Edit Interactions. Esto hará que aparezca este icono   en todos los gráficos por lo que debe cambiar el estado a   en los graficos mencionados.



b) Mapa de Calor

Para el tercer ejercicio tendremos las siguientes consideraciones:

- ✓ Crearemos un mapa de calor en base a cantidad de usuarios por día y hora

fedora

EJERCICIO 2

Año


2011

Mes

Enero

HORA	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
12:00 a.m.	16	17	5	5	6	11	17	25	25	5	12	7	7	14	28	39	17	4	3	13	21	13	22	7	9	17	26	9	28	33	7
01:00 a.m.	40	17	2	2	6	4	7	16	12	1	3	6	2	5	20	23	16	4	7	5	6	12	13	1	5	5	26	3	20	29	7
02:00 a.m.	32	9	2	1	2	2	1	16	11	3	3	1	2	1	12	16	8	4	3	2	2	11	18	1	2	10	26	2	15	11	1
03:00 a.m.	13	6	2	1	2	2	1	7	4	1	3	1	3	1	8	15	2	4	3	1	1	7	5	1	2	10	26	1	8	8	2
04:00 a.m.	1	3	1	2	2	1	1	1	1	3	3	1	4	1	5	1	3	4	2	1	1	3	5	1	1	10	26	1	3	1	2
05:00 a.m.	1	3	3	4	3	4	5	5	1	3	6	5	3	8	1	2	1	4	7	6	5	3	3	5	9	1	26	4	3	3	8
06:00 a.m.	2	2	30	36	33	36	34	2	1	31	27	16	28	17	3	1	5	4	32	35	27	2	1	15	36	8	26	16	2	3	37
07:00 a.m.	3	1	64	94	88	95	84	9	6	77	99	54	72	70	10	3	13	4	90	101	68	8	2	84	108	30	26	60	5	3	72
08:00 a.m.	8	8	154	179	195	219	210	15	10	188	217	128	202	158	23	18	33	4	197	249	217	27	19	177	238	72	26	157	34	12	185
09:00 a.m.	14	20	88	100	115	122	134	20	19	94	130	81	139	117	33	32	47	4	109	143	166	40	28	102	144	58	26	101	34	38	112
10:00 a.m.	36	53	44	42	57	45	63	61	49	31	54	39	38	44	59	79	57	4	47	57	63	53	58	40	55	28	26	49	55	64	69
11:00 a.m.	56	70	51	57	46	59	67	62	49	30	35	35	37	53	72	93	64	4	52	68	59	63	99	46	61	41	26	30	64	59	48
12:00 p.m.	84	93	61	78	79	84	59	98	83	52	57	55	52	61	89	104	80	3	70	84	78	70	116	63	106	48	26	29	78	97	68
01:00 p.m.	94	75	61	97	71	67	73	102	75	54	52	49	83	77	101	118	93	22	78	98	73	84	87	60	93	47	26	31	65	84	54
02:00 p.m.	106	59	77	63	62	70	50	95	72	47	63	44	42	64	118	91	86	28	75	81	62	75	110	45	68	36	26	38	99	122	86
03:00 p.m.	110	74	72	65	62	72	74	82	45	47	49	60	68	129	113	93	35	82	70	65	103	77	57	84	43	26	41	120	109	44	
04:00 p.m.	93	76	76	83	89	86	87	76	92	74	76	68	78	90	128	99	82	61	104	91	97	83	65	70	116	36	24	80	107	123	86
05:00 p.m.	67	65	157	212	190	172	187	69	62	178	136	139	162	159	83	105	71	125	197	215	161	67	55	184	222	26	84	149	91	77	161
06:00 p.m.	35	53	157	182	169	163	123	55	48	155	95	137	144	139	84	67	92	133	161	185	120	54	49	153	225	26	104	109	68	65	156
07:00 p.m.	37	30	110	112	132	112	95	30	41	95	51	83	99	92	74	61	60	99	112	152	96	59	50	106	146	26	79	89	58	55	111
08:00 p.m.	36	22	52	54	89	69	51	28	38	74	32	56	64	68	41	57	33	83	76	126	53	45	35	81	119	26	59	62	43	33	78
09:00 p.m.	34	31	52	48	43	48	39	37	20	38	20	57	40	52	57	28	27	41	59	57	41	39	25	59	45	26	38	58	36	28	56
10:00 p.m.	28	9	20	35	42	52	36	34	15	24	29	33	30	36	26	21	13	33	59	56	34	30	28	35	53	26	27	26	32	21	34
Total	985	804	1353	1563	1602	1608	1511	959	822	1321	1269	1164	1406	1422	1248	1204	1000	731	1653	1927	1543	984	991	1417	1987	682	847	1168	1101	1099	1501

- Comenzamos duplicando la primera página, la renombramos “Mapa de Calor”, cambiamos el título a “EJERCICIO 2.

- Creamos una matriz con el icono , en donde asignaremos las siguientes columnas:

- HORA, DIM_HORA
- DIA_MES, DIM_FECHA
- USERS, vw_FACT_BIKE_SHARING

Rows	
HORA	✓ X
Columns	
DIA_MES	✓ X
Values	
Sum of USERS	✓ X

- Agregaremos un “Background color” con un estrilo “Gradient”; para ello seleccionamos la pestaña de “Sum of USERS”, conditional Formating, Background color.

Background color - Sum of USERS ✕

Format style: Gradient ▾ Apply to: Values only ▾

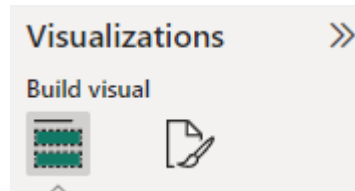
What field should we base this on?: Sum of USERS ▾ Summarization: Sum ▾ How should we format empty values?: As zero ▾

Minimum: Lowest value ▾ ▢ ▾ Enter a value Maximum: Highest value ▾ ▢ ▾ Enter a value

☐ Add a middle color

Learn more about conditional formatting OK Cancel

- Podemos dejar la configuración por defecto o cambiar el color dependiendo del impacto visual que queramos generar en el usuario.
- En ocasiones debemos ajustar el tamaño del lienzo debido a que es imprescindible ver la información visual completa, teniendo esto en cuenta seleccionaremos cualquier espacio en blanco del área o podemos presionar la tecla ESC hasta que el panel de visualizaciones aparezca de la siguiente forma:



- En el apartado de formatos de página, iremos a la sección Canvas settings y colocaremos los siguientes datos:

▼ Canvas settings

Type: Custom ▾

Height: 800 px ▴ ▾

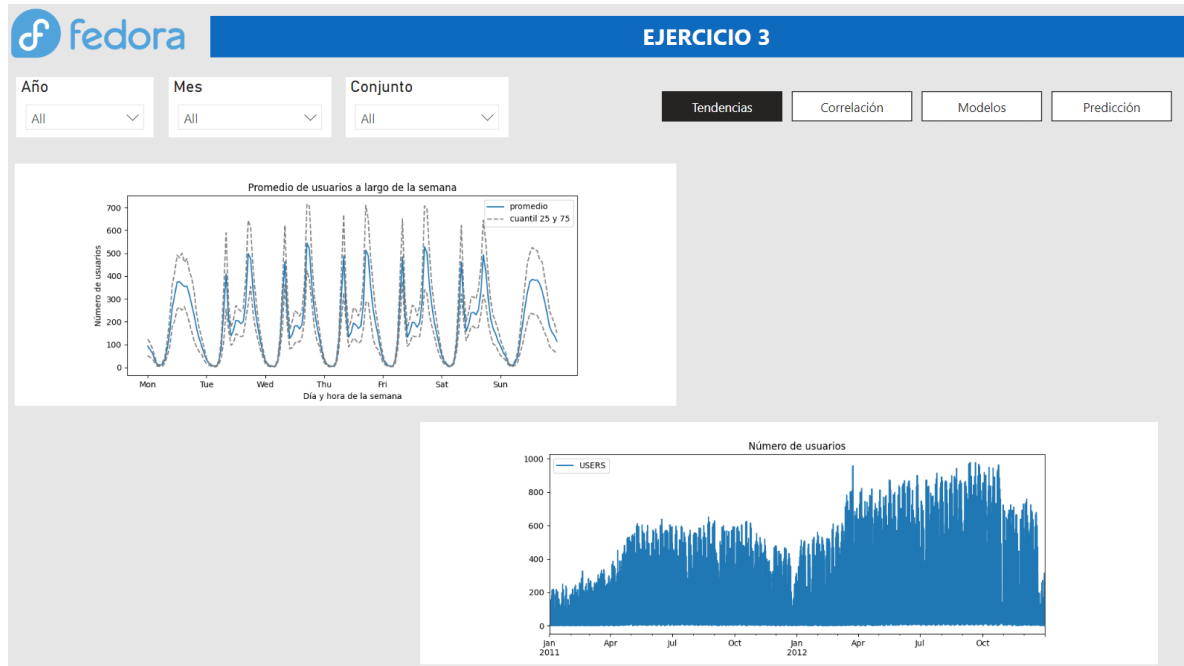
Width: 1340 px ▴ ▾

Vertical alignment: Top ▾

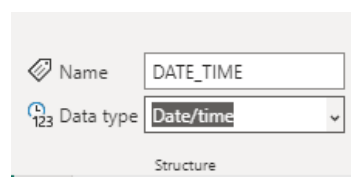
↺ Reset to default

c) Tendencia

- ✓ Creamos una pestaña llamada “Tendencia”
- ✓ Preparamos la tabla vw_FACT_BIKE_SHARING
 - Creación de columna calculada DATE_TIME
- ✓ Creación de graficas de tendencias basadas en Python
- ✓ Creación de un navegador personalizado



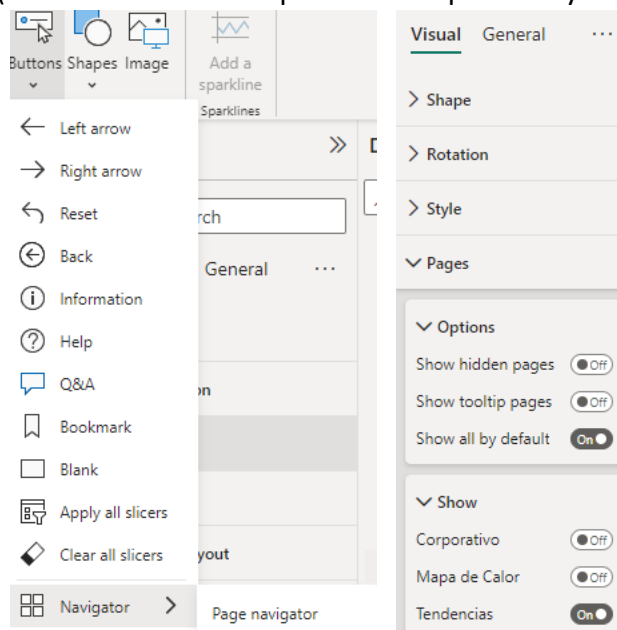
- A continuación, preparamos la vista vw_FACT_BIKE_SHARING en donde crearemos 1 columna calculada. Tenga en cuenta que, para el cambio de formato de las columnas calculadas, se debe ir al apartado “structure” el cual aparece cuando seleccionamos la misma.



Columna calculada	Formato	Descripción
DATE_TIME = CONCATENATE(LASTNONBLANK(DIM_FECHA[FECHA], 1) & " ", LASTNONBLANK(DIM_HORA[HORA],1))	Date/time	Concatena la fecha tiempo de las tablas

- Adicionalmente creamos un nuevo filtro de datos, para segmentar los conjuntos a evaluar.
 - **Filtro de Conjunto**; se hace uso de la columna TIPO_CONJUNTO de la tabla vw_FACT_BIKE_SHARING, se describen las configuraciones a usar:
 - En la sección de Visual,
 - desactivamos el Slicer header

- En la opción Slicer setting, Option, seleccionamos el Style en “Dropdown”
 - En la opción Slicer setting, Seleccion, aplicamos la opción “select all”
- En la sección General
 - Activamos la opción de Title y definimos el título “Conjunto”
- Ahora agregaremos el navegador de páginas (cinta de opciones Insert, sección de elements, Buttons, Navigator, Page navigator), este objeto ira creciendo conforme agreguemos más paginas (en las opciones visuales se puede limitar las páginas que podrá visualizar el usuario(deshabilitaremos las pestañas Corporativo y Mapa de calor)).



****Vista general**

- Teniendo preparado el entorno procedemos a crear un gráfico basado en Python seleccionando el icono **Py** ubicado en el panel de visualizaciones. Al momento de agregar este objeto emergerá una ventana la cual pedirá que se habilite los scripts, seleccionamos Enable

Enable script visuals

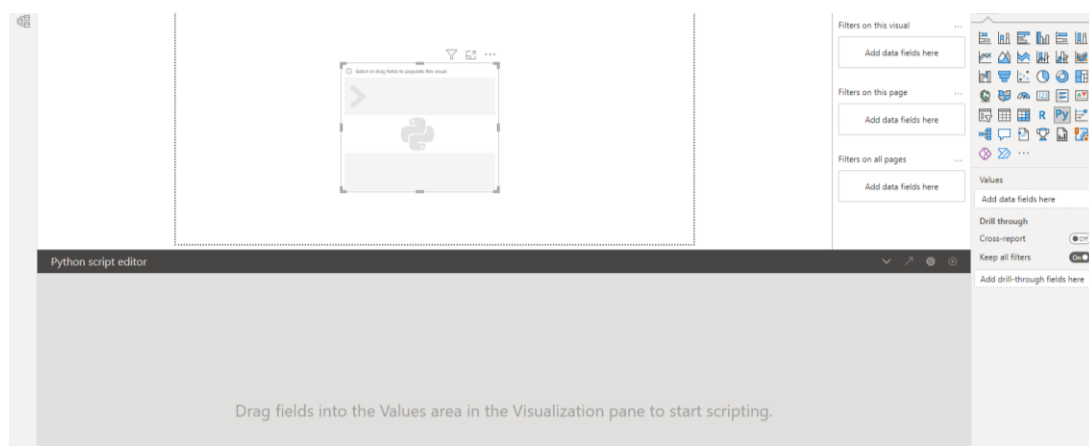
You need to enable script visuals to begin creating Python script.
Script visuals can execute script code that may contain security or privacy risks.

Enable

Cancel

- Una vez creado este objeto notaremos 2 cosas al tenerlo seleccionado;

- La primera, donde se nos muestra un apartado o área del script editor (deshabilitada porque aun no se han ingresado columnas o valores con los cuales trabajar)
- La segunda, en el panel de visualizaciones solo tenemos una entrada de valores “Values”, esto se debe a que Python en Power BI recibe como entrada un dataframe (estructura de datos en forma tabular) teniendo esto como premisa debemos considerar los siguientes aspectos:
 - La interpretación de dataframe en Power BI es un conjunto de valores únicos, por lo que todo valor duplicado será eliminado, siendo esto un problema dado que perdemos datos; es por ello por lo que siempre debemos colocar una columna que tenga la granularidad mínima o sobre la cual convierta la tupla o fila en un valor único.
 - El rendimiento de Python puede verse perjudicado dependiendo de la lógica agregada en la codificación de los gráficos usando Python (también aplica para R)



- Ahora agregaremos las siguientes columnas:
 - DATE_TIME, vw_FACT_BIKE_SHARING
 - DIA_SEMANA, DIM_FECHA
 - USERS, vw_FACT_BIKE_SHARING
 - HORA, DIM_HORA

<div> <div>Values</div> <div> <div>DATE_TIME</div> <div>Year</div> <div>Quarter</div> <div>Month</div> <div>Day</div> </div> <div> <div>DIA_SEMANA</div> <div>USERS</div> <div>HORA</div> </div> </div>	<p>Como podrá notar, la columna DATE_TIME aparece desglosada, para solucionar ello seleccionamos la pestaña indicada y seleccionamos la opción que DATE_TIME</p>
---	--

Values		Así debería quedar la relación de columnas a usar
DATE_TIME	▼ ×	
DIA_SEMANA	▼ ×	
USERS	▼ ×	
HORA	▼ ×	

- Ahora notaremos que el apartado de Python script editor nos muestra contenido:

Python script editor

⚠ Duplicate rows will be removed from the data.

1 # The following code to create a dataframe and remove duplicated rows is always executed and acts as a preamble for your script:

2

3 # dataset = pandas.DataFrame(DATE_TIME, DIA_SEMANA, USERS, HOUR)

4 # dataset = dataset.drop_duplicates()

5

6 # Paste or type your script code here:

***el icono resaltado hace referencia al botón de ejecución de script, el cual debe ser seleccionado en cada cambio del script.*

- En este apartado colocaremos el siguiente código:

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

#Transformación
dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'], format='%Y-%m-%d
%H:%M:%S')
dataset = dataset.set_index('DATE_TIME')
dataset = dataset.asfreq('H')
dataset = dataset.sort_index()

#Diseño de la grafica
fig, ax = plt.subplots(figsize=(10, 4))
promedio_dia_hora = dataset.groupby(["DIA_SEMANA", "HORA"])["USERS"].mean()
q25_dia_hora = dataset.groupby(["DIA_SEMANA", "HORA"])["USERS"].quantile(0.25)
q75_dia_hora = dataset.groupby(["DIA_SEMANA", "HORA"])["USERS"].quantile(0.75)

promedio_dia_hora.plot(ax=ax, label='promedio')
q25_dia_hora.plot(ax=ax, linestyle='dashed', color='gray', label='')
q75_dia_hora.plot(ax=ax, linestyle='dashed', color='gray', label='cuantil 25 y 75')

#Formato Visual
ax.set(
    title="Promedio de usuarios a largo de la semana",
    xticks=[i * 24 for i in range(7)],
    xticklabels=["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"],
    xlabel="Día y hora de la semana",
    ylabel="Número de usuarios"
)
ax.legend()
plt.show()
```

1. Se importan las librerías pandas y matplotlib.
2. Se utiliza la función "pd.to_datetime" de pandas para convertir la columna "DATE_TIME" del conjunto de datos a un objeto datetime. El formato de la fecha y la hora se especifica con el parámetro "format".
3. Se establece la columna "DATE_TIME" como el índice del conjunto de datos.
4. Se utiliza la función "asfreq" de pandas para establecer la frecuencia de los datos a cada hora (se establece la frecuencia a 'H').
5. Se ordena el conjunto de datos por el índice.
6. Se crea una figura y un objeto de ejes para la visualización.
7. Se calcula el promedio, el cuantil 25 y el cuantil 75 de los usuarios para cada día de la semana y cada hora del día.
8. Se grafica el promedio de los usuarios en función del día y la hora de la semana.
9. Se grafican los cuantiles 25 y 75 como líneas punteadas de color gris.
10. Se establece el título de la figura y las etiquetas de los ejes.
11. Se muestra la leyenda y se muestra la figura.

- Duplicamos el grafico y solo nos quedamos con las columnas DATE_TIME y HORA:

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

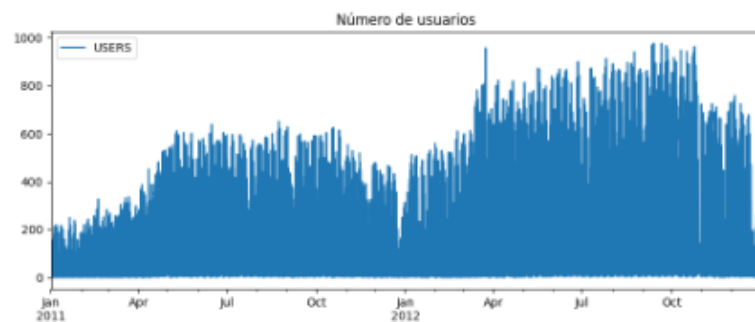
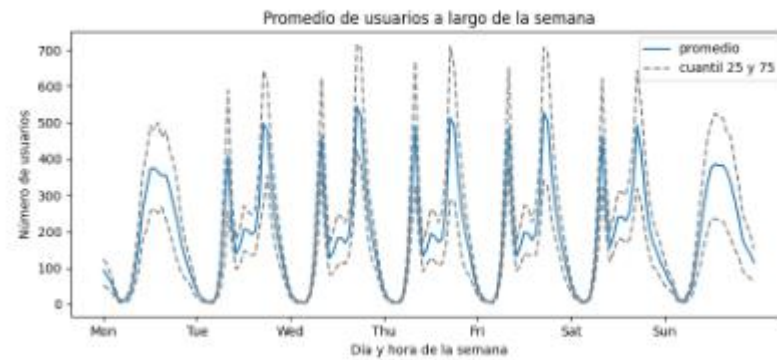
```
#Transformación
dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'],
format='%Y-%m-%d %H:%M:%S')
dataset = dataset.set_index('DATE_TIME')
dataset = dataset.asfreq('H')
dataset = dataset.sort_index()
```

```
#Diseño de la grafica
fig, ax = plt.subplots(figsize=(11, 4))
dataset['USERS'].plot(ax=ax)
```

```
#Formato Visual
ax.set(
    title='Número de usuarios'
)
ax.legend()
plt.show()
```

1. Se importan las librerías pandas y matplotlib.
2. Se utiliza la función "pd.to_datetime" de pandas para convertir la columna "DATE_TIME" del conjunto de datos a un objeto datetime. El formato de la fecha y la hora se especifica con el parámetro "format".
3. Se establece la columna "DATE_TIME" como el índice del conjunto de datos.
4. Se utiliza la función "asfreq" de pandas para establecer la frecuencia de los datos a cada hora (se establece la frecuencia a 'H').
5. Se ordena el conjunto de datos por el índice.
6. Se crea una figura y un objeto de ejes para la visualización.
7. Se grafica la columna "USERS" del conjunto de datos en función del tiempo.
8. Se establece el título de la figura.
9. Se muestra la leyenda y se muestra la figura.

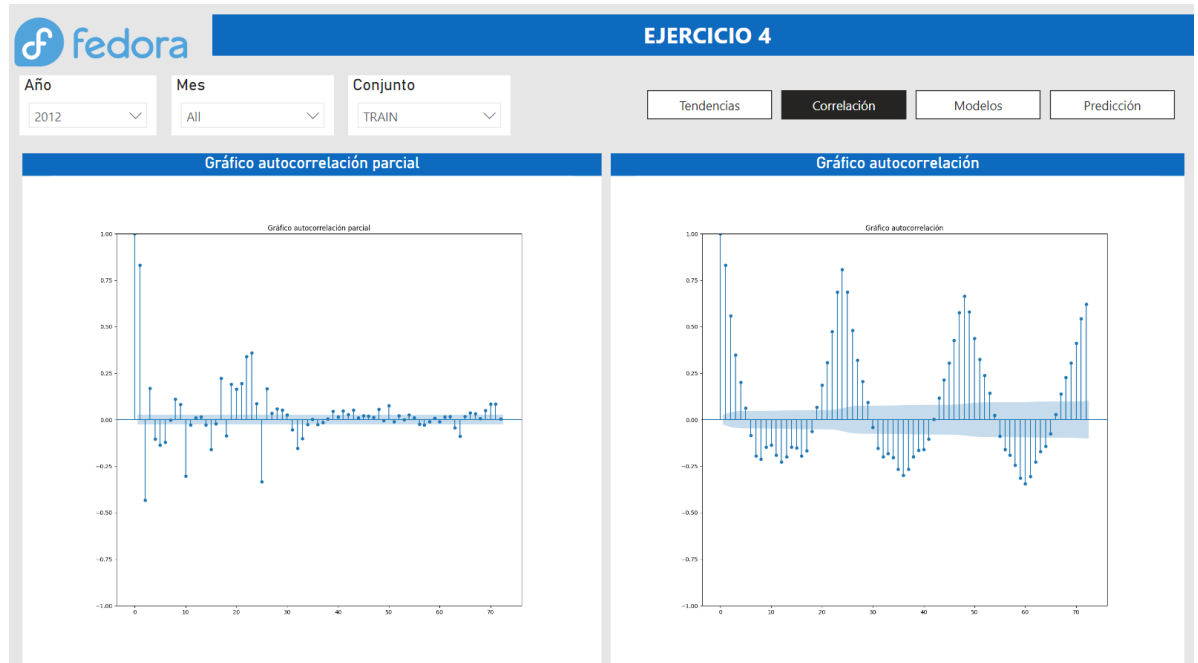
- Podrá notar la facilidad con la que se crean los gráficos, además que podemos realizar el proceso de tratamiento a los datos (Recuerde que este ligado al rendimiento del informe), también notara que dentro del mismo grafico se puede incluir el titulo; es por ello por lo que deshabilitaremos el título del objeto yendo al panel de visualizaciones.



d) Correlación

En este segundo ejercicio, tendremos las siguientes consideraciones:

- ✓ Crearemos una nueva pestaña llamada “Correlación”
- ✓ Creación de dos gráficos de autocorrelación entre una serie de tiempo



- Comenzamos duplicando la primera página, la renombramos “Correlación” y de título “EJERCICIO 4”.
- Continuaremos usando solo las columnas DATE_TIME y USERS; en los siguientes gráficos notaremos que los títulos no se ajustan bien al gráfico, por lo que activaremos el título del objeto visual (en el panel de visualizaciones, apartado general) en donde colocaremos un fondo de color azul (o a su elección) con letras blancas.
- Otro punto que debe considerar es el crecimiento del navegador de páginas, en este punto debería tener 2 opciones habilitadas (en referencia a las 2 pestañas agregadas y 2 pestañas desactivadas).

- A continuación, comenzamos con el desarrollo del primer grafico “Gráfico autocorrelación parcial”:

```
#Librerias
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.graphics.tsaplots import plot_pacf

#Transformación
dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'],
format='%Y-%m-%d %H:%M:%S')
dataset = dataset.set_index('DATE_TIME')
dataset = dataset.asfreq('H')
dataset = dataset.sort_index()

#Diseño de la grafica
fig, ax = plt.subplots(figsize=(13, 12))
plot_pacf(dataset['USERS'], ax=ax, lags=72, method='ywm')

#Formato Visual
ax.set(
    title="Gráfico autocorrelación parcial",
)

plt.show()
```

1. Se importan las librerías pandas, matplotlib y statsmodels.
2. Se utiliza la función "pd.to_datetime" de pandas para convertir la columna "DATE_TIME" del conjunto de datos a un objeto datetime. El formato de la fecha y la hora se especifica con el parámetro "format".
3. Se establece la columna "DATE_TIME" como el índice del conjunto de datos.
4. Se utiliza la función "asfreq" de pandas para establecer la frecuencia de los datos a cada hora (se establece la frecuencia a 'H').
5. Se ordena el conjunto de datos por el índice.
6. Se crea una figura y un objeto de ejes para la visualización.
7. Se utiliza la función "plot_pacf" de statsmodels para graficar la autocorrelación parcial de la columna "USERS" del conjunto de datos. Se establece el parámetro "lags" a 72 y el parámetro "method" a 'ywm'.
8. Se establece el título de la figura.
9. Se muestra la figura.

- Se muestra y detalla el segundo grafico “Gráfico autocorrelación”

```
#Librerías
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.graphics.tsaplots import plot_acf

#Transformación
dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'],
format='%Y-%m-%d %H:%M:%S')
dataset = dataset.set_index('DATE_TIME')
dataset = dataset.asfreq('H')
dataset = dataset.sort_index()

#Diseño de la grafica
fig, ax = plt.subplots(figsize=(13, 12))
plot_acf(dataset['USERS'], ax=ax, lags=72)

#Formato Visual
ax.set(
    title="Gráfico autocorrelación",
)

plt.show()
```

1. Primero se importan las librerías pandas, matplotlib.pyplot y statsmodels.graphics.tsaplots.plot_acf. Luego, se realiza una transformación en los datos para que la columna "DATE_TIME" sea interpretada como una fecha y hora y se ajusta la frecuencia a horas.
2. Después, se crea una figura con una subparcela de tamaño (13, 12). Se utiliza la función plot_acf de statsmodels para trazar el gráfico de autocorrelación para la serie de tiempo de la columna "USERS" del dataset, con 72 retrasos (lags) y se muestra en la subparcela.
3. Finalmente, se establece el título del gráfico en "Gráfico de autocorrelación" y se muestra la figura. El gráfico de autocorrelación se utiliza para detectar la autocorrelación en una serie de tiempo, es decir, la correlación entre valores retrasados en el tiempo. En el gráfico, los puntos más alejados de la línea de puntos en el eje x indican la magnitud de la correlación en cada retraso (lag), lo que puede ser útil para determinar el orden de los modelos ARIMA y para detectar patrones estacionales en la serie de tiempo.

Gráfico autocorrelación parcial

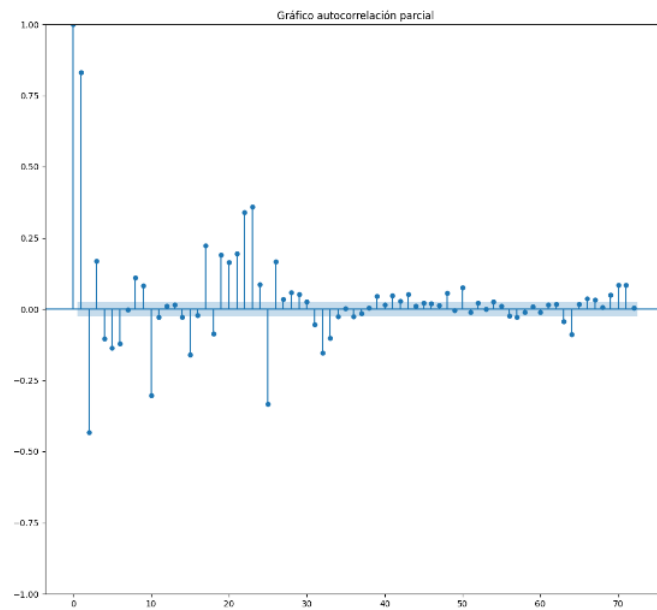
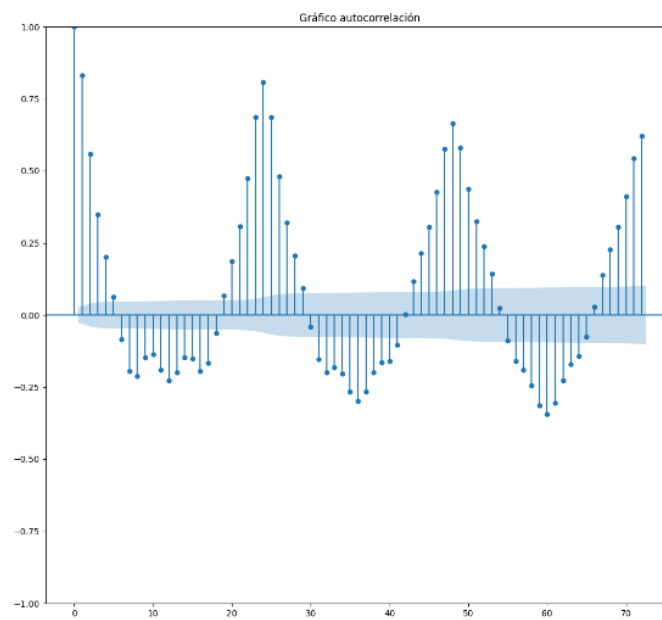
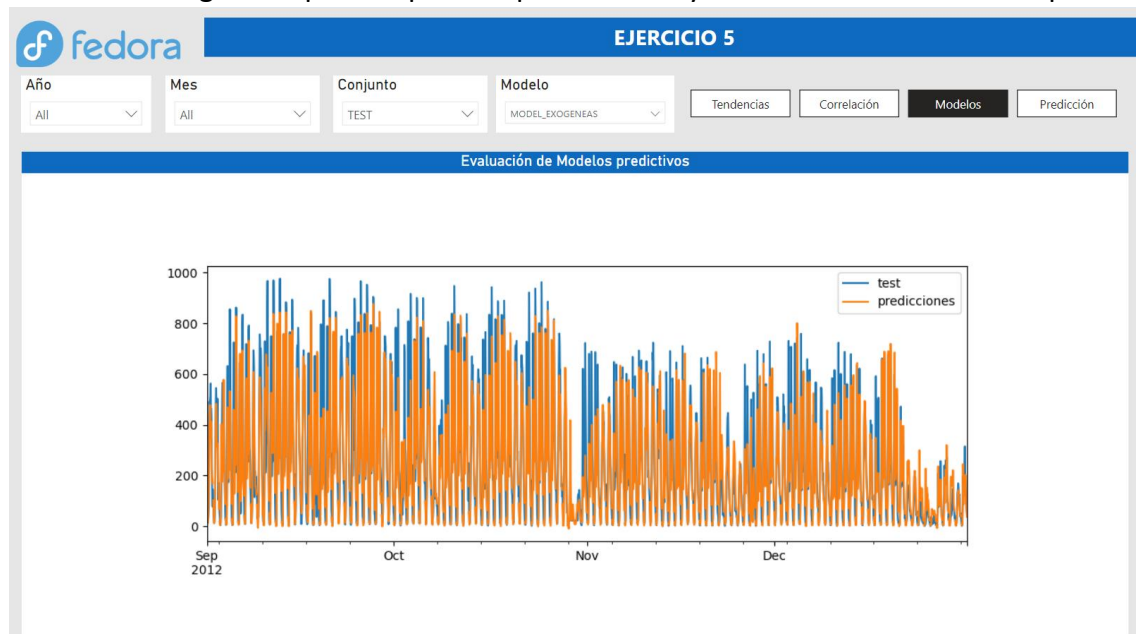


Gráfico autocorrelación

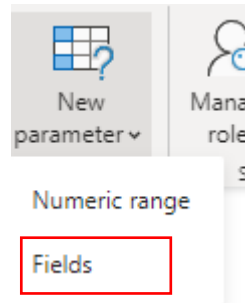


d) Modelos

- ✓ Crearemos una nueva pestaña llamada “Modelos”
- ✓ Reutilizaremos los filtros.
- ✓ Crearemos un filtro basado en parámetros desde columnas.
- ✓ Crearemos un grafico que compara las predicciones y los reales en serie de tiempo.



- Iniciamos duplicando la Primera pestaña, renombramos la página “Modelos”, Cambiamos el título a “EJERCICIO 5”.
- Nos dirigimos a la cinta de opciones Modeling, y buscamos la opción de new parameter, en donde seleccionaremos la sub-opción Fields.



- A continuación, en la ventana emergente colocaremos en name “Modelos”, y en el apartado de Fields, buscaremos la tabla vw_FORECASTING_BIKE_SHARING en donde seleccionaremos todas las columnas que comiencen con MODEL.

Parameters



Add parameters to visuals and DAX expressions so people can use slicers to adjust the inputs and see different outcomes. [Learn more](#)

What will your variable adjust?

Fields

Name

Modelos

Add and reorder fields

MODEL_EXOGENEAS X
MODEL_EXOGENEAS_CatBoost X
MODEL_EXOGENEAS_LGBM X
MODEL_FORECASTER X

Fields

Search

> DIM_FECHA
> DIM_HORA
> DIM_WEATHER
> vw_FACT_BIKE_SHARING
vw_PREDICCION_BIKE_SHARING
 ID_PREDICCION
 MODEL_EXOGENEAS
 MODEL_EXOGENEAS_CatBoost
 MODEL_EXOGENEAS_LGBM
 MODEL_FORECASTER

☒ Add slicer to this page

Create

Cancel

- De igual forma a como realizamos la creación de estos parámetros en el ejercicio anterior tenemos una nueva tabla de parámetros la cual podemos editar (agregar o quitar valores) seleccionando la misma; y adicionalmente se crea un filtro en el área de trabajo (dado que se dejó activa la opción).

```
Forecasting = {  
    ("MODEL_EXOGENEAS", NAMEOF('vw_FORECASTING_BIKE_SHARING'[MODEL_EXOGENEAS]),  
    0),  
    ("MODEL_EXOGENEAS_CatBoost",  
    NAMEOF('vw_FORECASTING_BIKE_SHARING'[MODEL_EXOGENEAS_CatBoost]), 1),  
    ("MODEL_EXOGENEAS_LGBM",  
    NAMEOF('vw_FORECASTING_BIKE_SHARING'[MODEL_EXOGENEAS_LGBM]), 2),  
    ("MODEL_FORECASTER", NAMEOF('vw_FORECASTING_BIKE_SHARING'[MODEL_FORECASTER]),  
    3)  
}
```

- En el caso del filtro, definiremos el slicer settings:
 - Options, Dropdown
 - Selection, Single select
- Ahora creamos (o reutilizamos) un objeto visual Python, en donde agregamos las siguientes columnas
 - DATE_TIME, vw_FACT_BIKE_SHARING
 - USERS, vw_FACT_BIKE_SHARING
 - Modelos, Ubicada en Modelos (ultima tabla creada por los parámetros)

- Luego agregamos el siguiente código en Python:

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

dataset['DATE_TIME'] =
pd.to_datetime(dataset['DATE_TIME'], format='%Y-%m-%d
%H:%M:%S')
dataset = dataset.set_index('DATE_TIME')
dataset = dataset.asfreq('H')
dataset = dataset.sort_index()

#Columna dinamica
col = dataset.columns

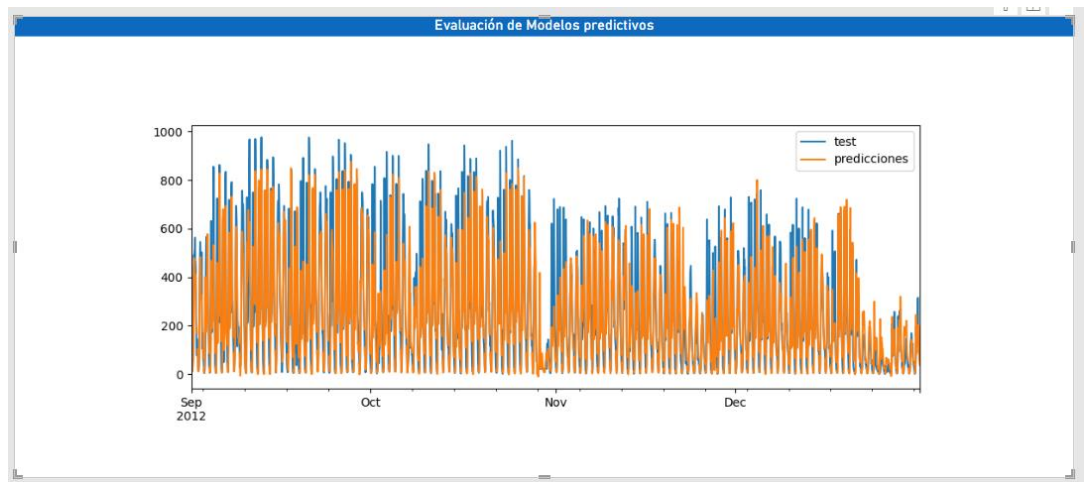
for elemento in col:
    if 'MODEL_' in elemento:
        columna = elemento

# Imprimir el DataFrame resultante
fig, ax = plt.subplots(figsize=(15, 6))
dataset[columna].plot()

plt.show()
```

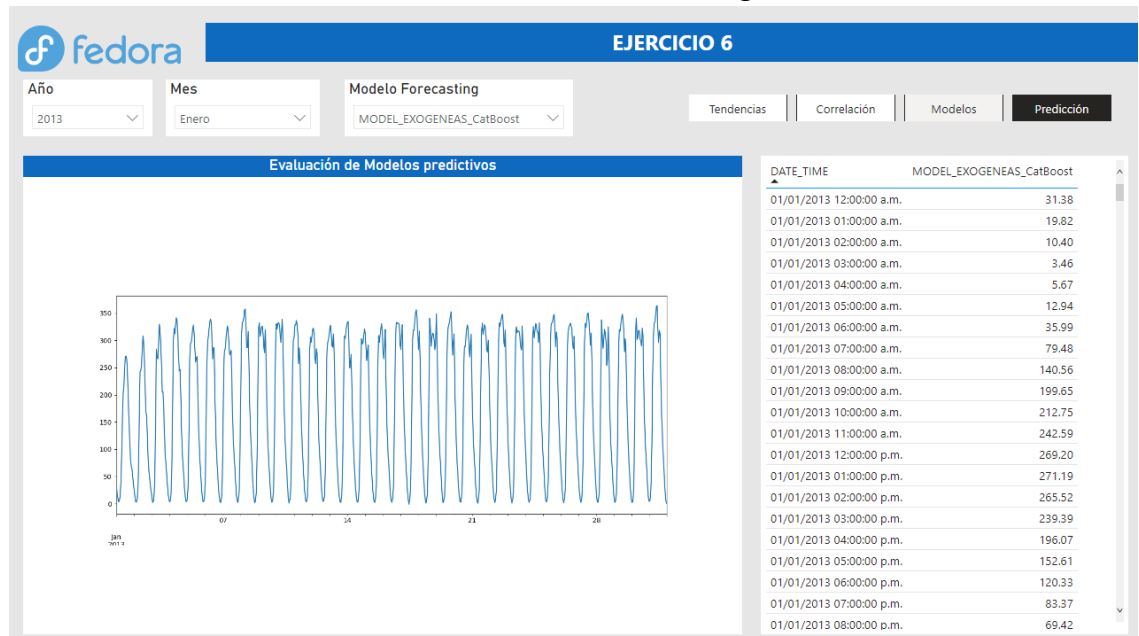
1. Primero, el código convierte la columna "DATE_TIME" en formato datetime y la establece como índice del dataframe. Luego, se rellena el índice con un intervalo horario ('H') y se ordena el dataframe por índice.
2. A continuación, el código busca una columna que contenga la subcadena "MODEL_" en su nombre y la guarda en una variable llamada "columna".
3. Finalmente, el código traza dos líneas: una para los datos de prueba de la columna "USERS" y otra para las predicciones del modelo de la columna guardada en "columna". El gráfico resultante muestra las dos líneas superpuestas y con una leyenda que las diferencia.

- Dado que la columna modelos contiene datos dinámicos debido a la parametrización, al seleccionar alguna opción del filtro modelo podremos obtener diferentes resultados en la serie de tiempo logrando identificar la exactitud de los modelos entrenados.
- Adicionalmente podemos agregarle un título al grafico el cual seria “Evaluación de Modelos predictivos”, con texto blanco y fondo azul.

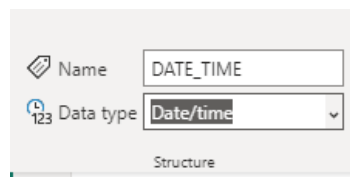


e) Predicción

- ✓ Crearemos una nueva pestaña llamada “Predicción”
- ✓ Ajustaremos los filtros Año y Mes.
- ✓ Crearemos un filtro basado en parámetros desde columnas.
- ✓ Crearemos un gráfico que muestre la serie de tiempo predicha.
- ✓ Crearemos una matriz donde se detalle el los valores, según el modelo seleccionado.

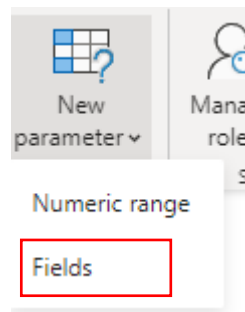


- Iniciamos duplicando la Primera pestaña, renombramos la página “Predicción”, Cambiamos el título a “EJERCICIO 6”.
- preparamos la vista vw_FORECASTING_BIKE_SHARING en donde crearemos 1 columna calculada. Al igual que en el ejercicio 3 se debe cambiar el formato de la columna calculada desde el apartado “structure” el cual aparece cuando seleccionamos la misma.



Columna calculada	Formato	Descripción
DATE_TIME = CONCATENATE(LASTNONBLANK(DIM_FECHA[FECHA], 1) & " ", LASTNONBLANK(DIM_HORA[HORA],1))	Date/time	Concatena la fecha tiempo de las tablas

- Nos dirigimos a la cinta de opciones Modeling, y buscamos la opción de new parameter, en donde seleccionaremos la sub-opción Fields.



- A continuación, en la ventana emergente colocaremos en name “Forecasting”, y en el apartado de Fields, buscaremos la tabla vw_FORECASTING_BIKE_SHARING en donde seleccionaremos todas las columnas que comiencen con MODEL.

Parameters ×

Add parameters to visuals and DAX expressions so people can use slicers to adjust the inputs and see different outcomes. [Learn more](#)

What will your variable adjust?
Fields

Name
Forecasting

Add and reorder fields

MODEL_EXOGENEAS	×
MODEL_EXOGENEAS_CatBoost	×
MODEL_EXOGENEAS_LGBM	×
MODEL_FORECASTER	×

☒ Add slicer to this page

Fields

Search

- > Forecasting
- > Modelos
- > vw_FACT_BIKE_SHARING
- > vw_FORECASTING_BIKE_SHARING
 - > DATE_TIME
 - ☐ FECHA_ID
 - ☐ HORA_ID
 - ☒ Σ MODEL_EXOGENEAS
 - ☒ Σ MODEL_EXOGENEAS_CatBoost
 - ☒ Σ MODEL_EXOGENEAS_LGBM
 - ☒ Σ MODEL_FORECASTER
- > vw_PREDICCION_BIKE_SHARING

Create **Cancel**

- Al momento de seleccionar créate, se creará una nueva tabla de parámetros la cual podemos editar (agregar o quitar valores) seleccionando la misma; adicionalmente se crea un filtro en el área de trabajo.

```
Modelos = {
    ("MODEL_EXOGENEAS", NAMEOF('vw_PREDICCION_BIKE_SHARING'[MODEL_EXOGENEAS]), 0),
    ("MODEL_EXOGENEAS_CatBoost",
    NAMEOF('vw_PREDICCION_BIKE_SHARING'[MODEL_EXOGENEAS_CatBoost]), 1),
    ("MODEL_EXOGENEAS_LGBM", NAMEOF('vw_PREDICCION_BIKE_SHARING'[MODEL_EXOGENEAS_LGBM]), 2),
    ("MODEL_FORECASTER", NAMEOF('vw_PREDICCION_BIKE_SHARING'[MODEL_FORECASTER]), 3)
}
```

- En el caso del filtro, definiremos el slicer settings:
 - Options, Dropdown

- Selection, Single select
- Debemos tener en cuenta que los filtros Año y Mes vienen preconfigurados (filtro TOP N interno) en base a la tabla vw_FACT_BIKE_SHARING, por ello debemos acceder a estos filtros y cambiar la condición (columna) del filtro TOP N usando la columna de la tabla vw_FORECASTING_BIKE_SHARING que en este caso también es FECHA_ID.

- Tenga en cuenta que, de no realizar correctamente el paso anterior, tendrá 2 filtros exentos de la lógica de esta pestaña; para validar que se hizo correctamente, se muestra los valores que deben salir en ambos filtros

- Ahora creamos (o reutilizamos) un objeto visual Python, en donde agregamos las siguientes columnas
 - DATE_TIME, vw_FORECASTING_BIKE_SHARING
 - Forecasting, Ubicada en Forecasting (ultima tabla creada por los parámetros)

- Luego agregamos el siguiente código en Python:

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

dataset['DATE_TIME'] =
pd.to_datetime(dataset['DATE_TIME'], format='%Y-%m-%d
%H:%M:%S')
dataset = dataset.set_index('DATE_TIME')
dataset = dataset.asfreq('H')
dataset = dataset.sort_index()

#Columna dinamica
col = dataset.columns

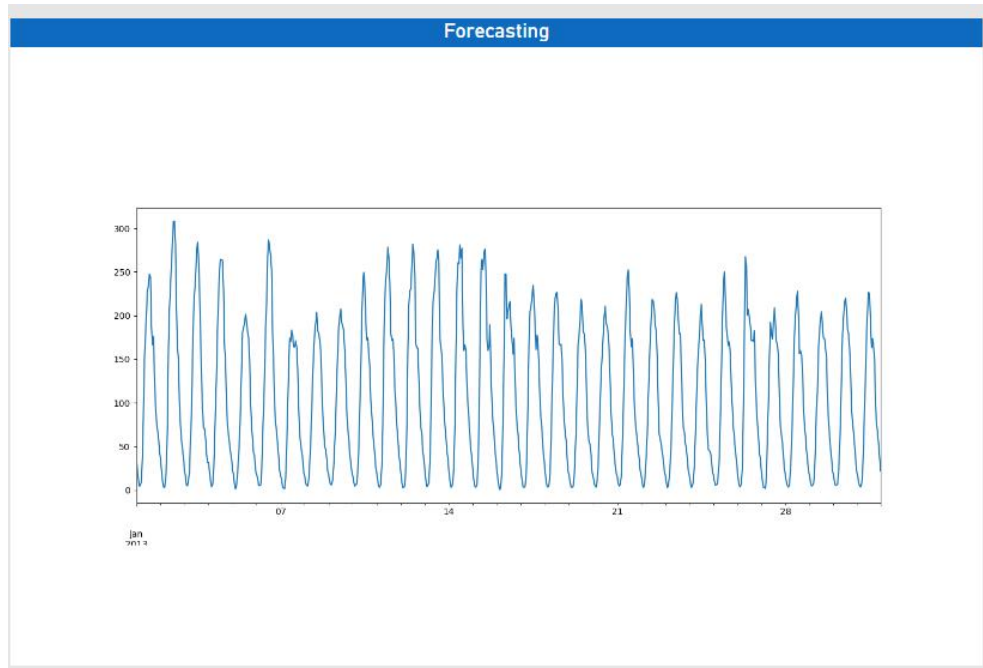
for elemento in col:
    if 'MODEL_' in elemento:
        columna = elemento


# Imprimir el DataFrame resultante
fig, ax = plt.subplots(figsize=(11, 4))
dataset['USERS'].plot(ax=ax, label='test')
dataset[columna].plot(ax=ax, label='predicciones')

ax.legend()
plt.show()
```

1. Carga un conjunto de datos y realiza algunas operaciones de series de tiempo para preparar los datos.
2. Busca en el conjunto de datos una columna que contenga la cadena 'MODEL_' en su nombre y la asigna a la variable 'columna'.
3. Crea una figura con un objeto de eje y un tamaño determinado.
4. Traza la columna 'columna' del conjunto de datos en el objeto del eje.
5. Muestra el gráfico generado.

- Dado que la columna modelos contiene datos dinámicos debido a la parametrización, al seleccionar alguna opción del filtro modelo podremos obtener diferentes resultados en la serie de tiempo logrando identificar la exactitud de los modelos entrenados.
- Adicionalmente podemos agregarle un título al grafico el cual seria “Evaluación de Modelos predictivos”, con texto blanco y fondo azul.



- Para concluir, crearemos una tabla con el icono  y a continuación, el detalle de su contenido

- Métricas:

	Métrica	Origen
Columns	1. DATE_TIME 2. Forecasting	1. vw_FORECASTING_BIKE_SHARING 2. Forecasting

- En la sección de Visual,
 - Style presets, lo establecemos en Minimal.

DATE_TIME	MODEL_EXOGENEAS
01/01/2013 12:00:00 a.m.	35.40
01/01/2013 01:00:00 a.m.	22.39
01/01/2013 02:00:00 a.m.	11.94
01/01/2013 03:00:00 a.m.	4.68
01/01/2013 04:00:00 a.m.	5.56
01/01/2013 05:00:00 a.m.	10.59
01/01/2013 06:00:00 a.m.	36.79
01/01/2013 07:00:00 a.m.	85.37
01/01/2013 08:00:00 a.m.	148.58
01/01/2013 09:00:00 a.m.	175.29
01/01/2013 10:00:00 a.m.	202.75
01/01/2013 11:00:00 a.m.	228.48
01/01/2013 12:00:00 p.m.	234.10
01/01/2013 01:00:00 p.m.	247.46
01/01/2013 02:00:00 p.m.	244.66
01/01/2013 03:00:00 p.m.	188.56
01/01/2013 04:00:00 p.m.	166.41
01/01/2013 05:00:00 p.m.	176.24

OVER VIEW

1. Columnas Calculadas

Orden	Métrica	Tabla
1	DATE_TIME = CONCATENATE(LASTNONBLANK(DIM_FECHA[FECHA], 1) &" ", LASTNONBLANK(DIM_HORA[HORA],1))	vw_FACT_BIKE_SHARING
2	DATE_TIME = CONCATENATE(LASTNONBLANK(DIM_FECHA[FECHA], 1) &" ", LASTNONBLANK(DIM_HORA[HORA],1))	vw_FORECASTING_BIKE_SHARING

2. Métricas

Orden	Métrica	Tabla
1	.DIA = MAX(DIM_FECHA[DIA_MES])	vw_FACT_BIKE_SHARING
2	.sum_users = SUM(vw_FACT_BIKE_SHARING[USERS])	vw_FACT_BIKE_SHARING
3	.user_moth = TOTALMTD([.sum_users], DIM_FECHA[FECHA])	vw_FACT_BIKE_SHARING
4	.users_total_year = TOTALYTD([.sum_users], DIM_FECHA[FECHA])	vw_FACT_BIKE_SHARING
5	.users_prev_month = TOTALMTD([.user_moth], PREVIOUSMONTH(DIM_FECHA[FECHA]))	vw_FACT_BIKE_SHARING
6	.users_prev_year = TOTALMTD([.users_total_year], PREVIOUSYEAR(DIM_FECHA[FECHA]))	vw_FACT_BIKE_SHARING
7	.%var_users_yearr = DIVIDE([.users_total_year] - [.users_prev_year], [.users_prev_year])	vw_FACT_BIKE_SHARING
8	.%var_users_month = DIVIDE([.user_moth] - [.users_prev_month], [.users_prev_month])	vw_FACT_BIKE_SHARING
9	.promedio_movil = DIVIDE([.user_moth], [.DIA])	vw_FACT_BIKE_SHARING

3. Librerías Necesarias

Código de instalación	Descripción de la librería
pip install numpy	se utiliza para trabajar con matrices o arrays multidimensionales y realizar operaciones matemáticas en ellos
pip install pandas	se utiliza principalmente para la manipulación y análisis de datos

pip install <code>matplotlib</code>	permite crear gráficos y visualizaciones de alta calidad. Se utiliza comúnmente en el análisis de datos y en la creación de informes visuales
pip install <code>statsmodels</code>	proporciona una amplia gama de herramientas y técnicas estadísticas y econométricas para analizar y modelar datos. Esta librería es muy útil para el análisis de series de tiempo y la regresión lineal

4. Script Python

Orden	Código
1	<pre> #Librerias import pandas as pd import matplotlib.pyplot as plt #Transformación dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'], format='%Y-%m-%d %H:%M:%S') dataset = dataset.set_index('DATE_TIME') dataset = dataset.asfreq('H') dataset = dataset.sort_index() #Diseño de la grafica fig, ax = plt.subplots(figsize=(10, 4)) promedio_dia_hora = dataset.groupby(["DIA_SEMANA", "HORA"])["USERS"].mean() q25_dia_hora = dataset.groupby(["DIA_SEMANA", "HORA"])["USERS"].quantile(0.25) q75_dia_hora = dataset.groupby(["DIA_SEMANA", "HORA"])["USERS"].quantile(0.75) promedio_dia_hora.plot(ax=ax, label='promedio') q25_dia_hora.plot(ax=ax, linestyle='dashed', color='gray', label='') q75_dia_hora.plot(ax=ax, linestyle='dashed', color='gray', label='cuantil 25 y 75') #Formato Visual ax.set(title="Promedio de usuarios a largo de la semana", xticks=[i * 24 for i in range(7)], xticklabels=["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"], xlabel="Día y hora de la semana", ylabel="Número de usuarios") ax.legend() plt.show() </pre>
2	<pre> #Librerias import pandas as pd import matplotlib.pyplot as plt from statsmodels.graphics.tsaplots import plot_acf #Transformación dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'], format='%Y-%m-%d %H:%M:%S') dataset = dataset.set_index('DATE_TIME') dataset = dataset.asfreq('H') dataset = dataset.sort_index() #Diseño de la grafica </pre>

	<pre> fig, ax = plt.subplots(figsize=(11, 4)) dataset['USERS'].plot(ax=ax) #Formato Visual ax.set(title='Número de usuarios') ax.legend() plt.show() </pre>
3	<pre> #Librerias import pandas as pd import matplotlib.pyplot as plt from statsmodels.graphics.tsaplots import plot_pacf #Transformación dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'], format='%Y-%m-%d %H:%M:%S') dataset = dataset.set_index('DATE_TIME') dataset = dataset.asfreq('H') dataset = dataset.sort_index() #Diseño de la grafica fig, ax = plt.subplots(figsize=(13, 12)) plot_pacf(dataset['USERS'], ax=ax, lags=72, method='ywm') #Formato Visual ax.set(title="Gráfico autocorrelación parcial",) plt.show() </pre>
4	<pre> #Librerias import pandas as pd import matplotlib.pyplot as plt from statsmodels.graphics.tsaplots import plot_acf #Transformación dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'], format='%Y-%m-%d %H:%M:%S') dataset = dataset.set_index('DATE_TIME') dataset = dataset.asfreq('H') dataset = dataset.sort_index() #Diseño de la grafica fig, ax = plt.subplots(figsize=(13, 12)) plot_acf(dataset['USERS'], ax=ax, lags=72) #Formato Visual ax.set(title="Gráfico autocorrelación",) plt.show() </pre>
5	<pre> #Librerias import pandas as pd import matplotlib.pyplot as plt </pre>

	<pre> #Transformación dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'], format='%Y-%m-%d %H:%M:%S') dataset = dataset.set_index('DATE_TIME') dataset = dataset.asfreq('H') dataset = dataset.sort_index() #Columna dinámica col = dataset.columns for elemento in col: if 'MODEL_' in elemento: columna = elemento #Diseño de la grafica fig, ax = plt.subplots(figsize=(11, 4)) dataset['USERS'].plot(ax=ax, label='test') dataset[columna].plot(ax=ax, label='predicciones') ax.legend() plt.show() </pre>
6	<pre> #Librerias import pandas as pd import matplotlib.pyplot as plt dataset['DATE_TIME'] = pd.to_datetime(dataset['DATE_TIME'], format='%Y-%m-%d %H:%M:%S') dataset = dataset.set_index('DATE_TIME') dataset = dataset.asfreq('H') dataset = dataset.sort_index() #Columna dinamica col = dataset.columns for elemento in col: if 'MODEL_' in elemento: columna = elemento # Imprimir el DataFrame resultante fig, ax = plt.subplots(figsize=(15, 6)) dataset[columna].plot() plt.show() </pre>