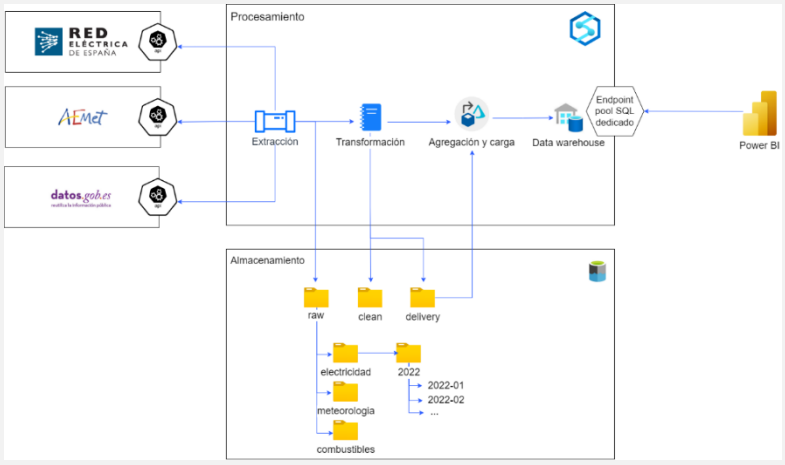
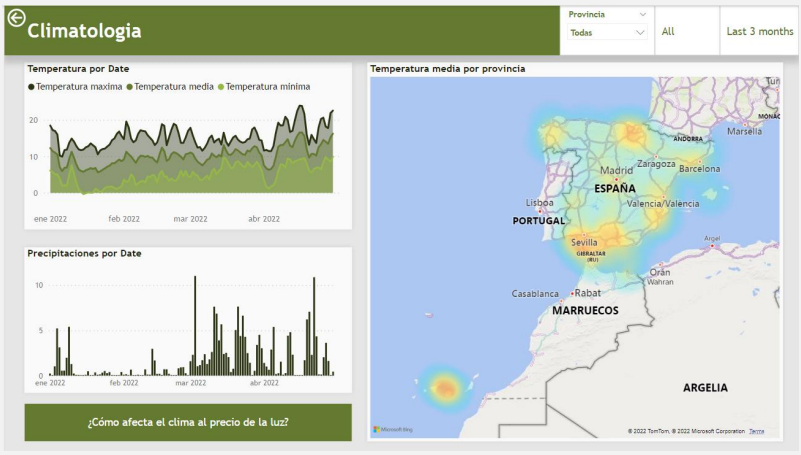
www.stratebi.com

91.788.34.10

info@stratebi.com

Trabajando con Synapse Analytics y PowerBI. Caso de Uso real





CONTENIDO

1. INTRODUCCIÓN

Synapse Analytics es un servicio cloud de Azure que proporciona una serie de funcionalidades que permiten la realización de un análisis completo de los datos. Para ello, Synapse cuenta con varias capacidades funcionales y técnicas de las que podemos destacar los pipelines para implementar procesos ETL, ELT o ELTL, los notebooks de Synapse o los dataflows para realizar transformaciones y asegurar la calidad del dato, los pools de SQL para almacenar los datos a analizar a modo de Data Warehouse, entre otras muchas funcionalidades.

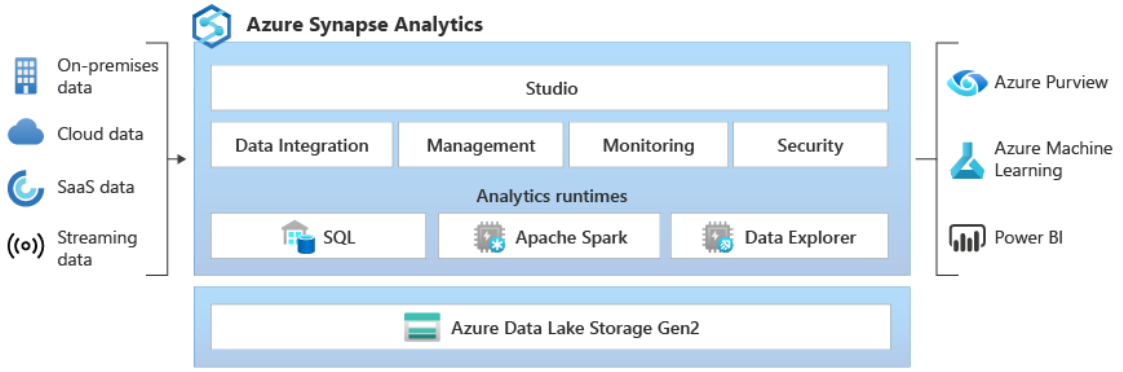


Figura 0. Arquitectura de Synapse Analytics

Toda esta suite de utilidades permite a los usuarios realizar un proceso de analítica de datos con perspectiva end-to-end. De esta forma, una vez que el usuario haya identificado las fuentes de datos que quiere analizar, puede emplear el servicio de Synapse Analytics para realizar la extracción del origen de datos, transformar estos datos con notebooks o dataflows y almacenarlos en un destino como una base de datos Azure SQL o un pool SQL dedicado de Synapse. Nos ha parecido interesante crear un ejemplo con Azure Synapse, pues se consolida como pieza clave en las arquitecturas modernas de datos basadas en Azure. A lo largo de este artículo, daremos un paseo por Synapse Analytics con una visión end-to-end, implementando procesos ELTL (extracción, carga, transformación y carga), hasta explotar el dato con Power BI.

Requisitos Previos

Para poder implementar procesos en Synapse Analytics es necesario cumplir varios requisitos:

- Es necesario tener una suscripción de Azure para poder crear los recursos.
- Necesitamos tener un grupo de recursos en el que crearemos los recursos necesarios de Azure.
- Si queremos visualizar los datos una vez terminado el proceso ELTL, debemos tener instalado Power BI Desktop.

Creación de los Recursos

Como se ha mencionado previamente, es necesario que tengamos un grupo de recursos creado en el portal de Azure donde podamos crear el resto de los recursos necesarios. Una vez definido el grupo, debemos crear los siguientes recursos:

Figura 1. Recursos necesarios en Azure

Se pueden distinguir varios tipos de recursos entre los cuales el principal es el Synapse workspace. Este recurso será el eje principal de todo el proceso pues gestiona todas las tareas del proceso ELTL y proporciona el acceso a los datos.

Además del propio recurso de Synapse, es necesario crear previamente un ADLS Gen 2 pues es el sistema de almacenamiento soportado por Synapse y es obligatorio emplearlo cuando queremos crear un workspace de Synapse.

En cuanto a los pools de SQL y Spark, estos son creados desde Synapse por dos motivos:

- El pool de Spark se crea para otorgar la capacidad de procesamiento a los notebooks y a los dataflows. Se le pueden asignar diferentes capacidades de cómputo e incluso la habilidad de escalar entre varios nodos.
- El pool de SQL dedicado se utiliza como destino en nuestro proceso ELTL. Este pool hace las funciones de Data Warehouse y, al ser de tipo “Dedicado”, solo incurre en costes cuando lo tenemos encendido.

Por último, se puede observar que se ha creado un Key Vault para almacenar algunos secretos que usa el pipeline.

2. PLANTEAMIENTO Y ARQUITECTURA

En el ejercicio que se plantea, se desea recabar información sobre el histórico de meses anteriores para poder analizar la evolución de los precios de la luz y de los combustibles, así como ver en qué grado afectan los valores climatológicos en estas variables.

Para ello, se emplean varias APIs públicas como orígenes de datos. Estas APIs son las siguientes:

- **Red Eléctrica de España**: permite la obtención de datos sobre el precio de la electricidad, la demanda y generación energética entre otros muchos datos.

- **AEMET**: proporciona datos sobre las mediciones de los valores climatológicos diarios como la temperatura o las precipitaciones, para cada una de sus estaciones meteorológicas.

- **Datos.gob.es**: este portal de datos abiertos pertenece al Gobierno de España y en él se proporcionan diferentes mediciones y datos proporcionados por los principales organismos públicos. En este caso, se ha empleado una de las APIs del Ministerio de Transición Ecológica y el Reto Demográfico que permite acceder a los datos de los precios de los combustibles en las estaciones de servicio españolas.

Una vez se ha expuesto el planteamiento inicial de nuestro proceso ELTL, se presenta la arquitectura del mismo en el siguiente diagrama:

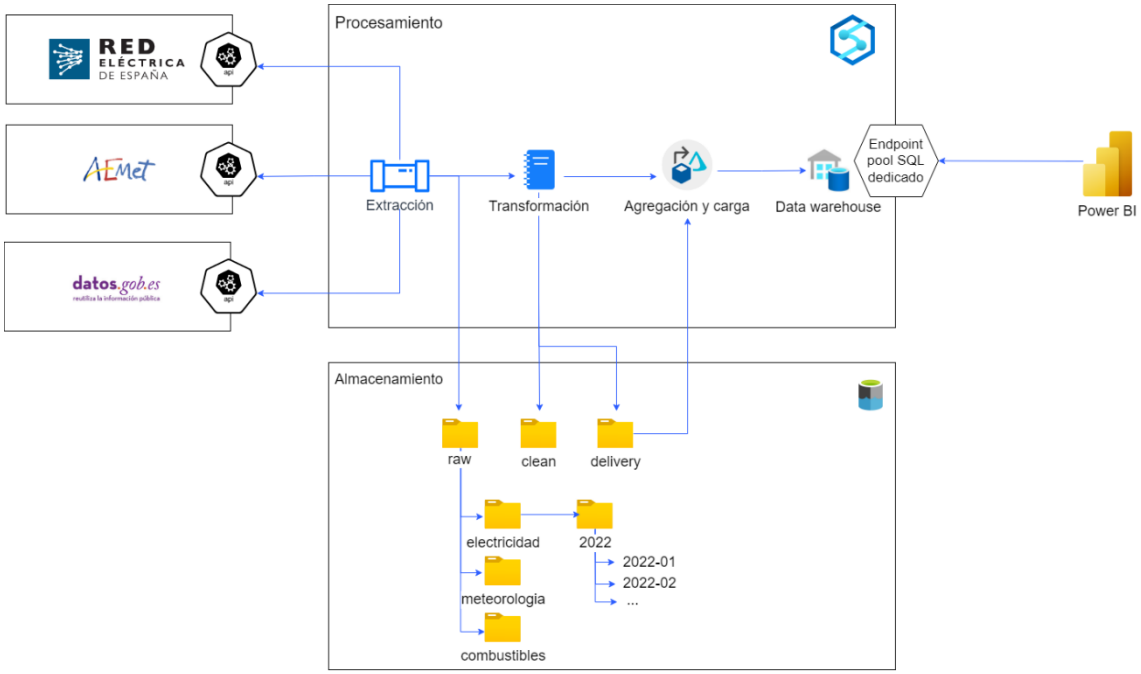


Figura 2. Arquitectura del proceso ELTL con Synapse

Como se puede observar en la figura 2, en la parte de la izquierda se encuentran los orígenes de datos del proceso ELTL. Para realizar la fase de extracción de datos, se emplean pipelines de Synapse que realizan diversas llamadas a los endpoints de las APIs.

Cabe destacar que una vez se han extraído los datos del origen, se almacenan en la capa raw del ADLS Gen 2 que está conectado con el recurso de Synapse. El ADLS se puede estructurar de diferentes formas, sin embargo, gracias a la estructura de almacenamiento de ficheros jerárquica introducida en la Gen 2 de los data lakes, podemos crear varias capas en función del estado del dato. Esto se materializa en la división del data lake en tres capas:

- Raw: contiene los datos en el formato de origen.
- Clean: contiene los datos tras aplicar transformaciones y un proceso de limpieza.
- Delivery: contiene los datos que van a utilizar las aplicaciones.

Una vez se han extraído los datos y se han almacenado en la capa raw, se proceden a realizar las transformaciones con los notebooks de Synapse y se almacenan los datos en las capas clean y delivery.

En este caso, dado que los datos se encuentran a diferentes niveles de detalle (algunos se proporcionan por las APIs a nivel diario y otros a nivel horario), se ha decidido tener un pool de SQL dedicado donde almacenar los datos agregados al mismo nivel de detalle y usar este en vez de la capa delivery para que lo utilice nuestro informe de Power BI. Para realizar esta agregación, se han empleado transformaciones mediante dataflows que toman como origen la capa delivery y como destino el pool SQL dedicado.

Una vez se ha creado el Data Warehouse con el pool SQL dedicado, podemos usar el endpoint de Synapse para importar estos datos desde Power BI y realizar los informes correspondientes.

3. ELTL CON PIPELINES

La implementación de los procesos ELTL con Synapse se materializa con los pipelines. Estos pipelines son una serie de actividades encadenadas una tras otra. En el ejercicio planteado, se pueden observar que hay tres fuentes diferentes y por ello, se ha decidido dividir el proceso ELTL en tres pipelines diferentes, uno para cada fuente. Cabe destacar que sobre cada origen de datos se tiene una forma diferente de realizar las peticiones. Sin embargo, todas las APIs admiten peticiones basadas en rangos de fechas por lo que, para facilitar el proceso de creación de pipelines, se van a crear unos parámetros que serán comunes en todos ellos.

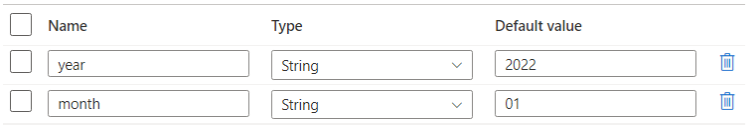


Figura 3. Parámetros de los pipelines

En primer lugar, trataremos el pipeline que extrae información de la API de la Red Eléctrica de España.

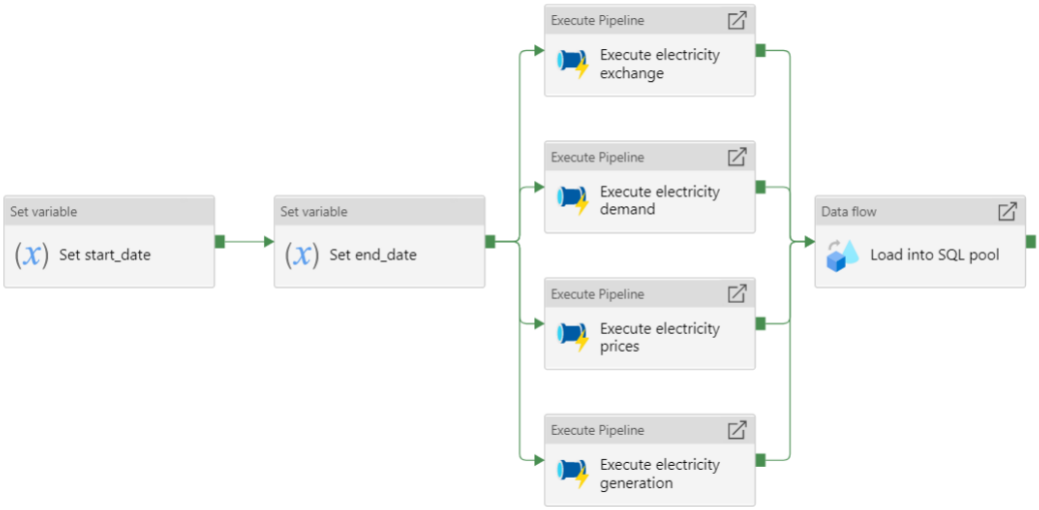


Figura 4. Pipeline de electricidad

Como se puede apreciar en la figura 4, el pipeline comienza estableciendo una serie de variables. Estas variables son necesarias para establecer el rango de fechas del que se quiere extraer la información y son requisito obligatorio para las peticiones de la API de la REE. Estos valores se calculan teniendo en cuenta el parámetro que se ha mencionado previamente.

Una vez se ha definido el rango de fechas que se quiere extraer, se ejecutan una serie de pipelines que tienen como objetivo hacer varias peticiones según la información que se quiera obtener. En este caso, se quiere extraer información de:

- Precios (€/MWh): Spot y PVPC. Son los dos tipos de precios de la electricidad del sistema eléctrico español:
    - El precio spot de electricidad es uno de los principales indicadores del mercado eléctrico y señal de su condición de adaptación entre oferta y demanda. Refleja el coste de suministrar un kilo watt hora (kWh) adicional al sistema eléctrico.
    - El PVPC (Precio de Voluntario Pequeño Consumidor) es una tarifa de discriminación horaria y precio variable. Es decir, el precio del kWh de electricidad cambia para cada hora de cada día. Por lo tanto, su precio está sometido a la volatibilidad del mercado eléctrico, según la oferta y la demanda entre las compañías generadoras de energía y las comercializadoras que venden esa energía al consumidor.
- Demanda (MWh): Real, programada y prevista.
- Generación (MWh): Datos de generación de energías renovables y no renovables.
- Intercambios (MWh): Importaciones y exportaciones de España con los países con los que existe una frontera física (Marruecos, Portugal, Francia y Andorra).

Como se puede ver en el flujo, cada uno de estos pipelines se ejecuta en paralelo, lo que permite realizar solicitudes simultáneas a la API en cuestión. Todos estos pipelines tienen una estructura similar, que se presenta en la siguiente imagen:

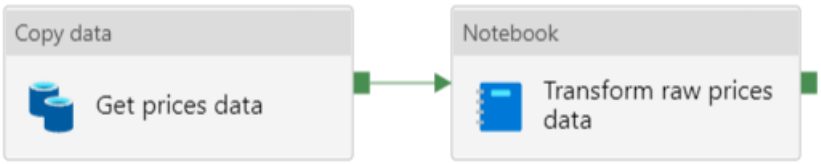


Figura 5. Pipeline precios de la electricidad

El pipeline se divide en dos actividades:

- Copy data: como su propio nombre indica, consiste en copiar datos de una fuente de origen a un destino. En este caso, la fuente de origen es la API de la Red Eléctrica y la fuente de destino es la capa raw de nuestro datalake.

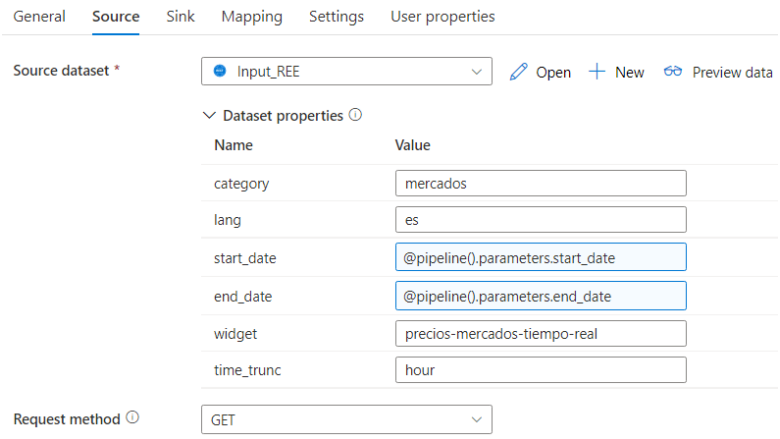


Figura 6. Origen de la actividad copy data

Sobre la imagen anterior, hay varios factores a tener en cuenta como el método HTTP que se emplea (GET) o el dataset que se está empleando. El dataset funciona como una interfaz entre nuestra actividad del pipeline y los datos. En este caso, el dataset está parametrizado, lo que permite usar el mismo dataset para hacer el resto de las llamadas a la API. Desde Synapse, podemos crear los integration datasets desde la pestaña "Data". En la siguiente imagen se muestra la configuración realizada para este dataset en particular:

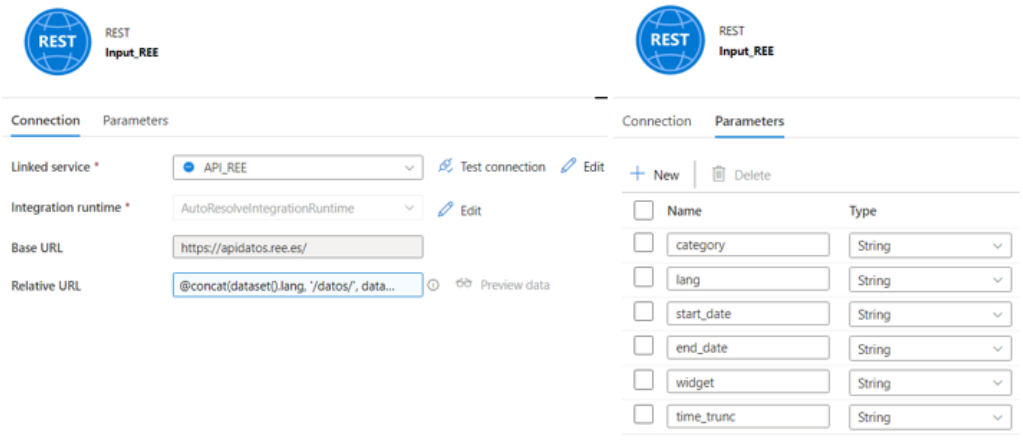


Figura 7. Configuración del dataset

Este dataset es de tipo REST ya que es el servicio que vamos a emplear en la consulta a la API. Los parámetros que se han establecido son los que están definidos en la documentación de la API, y se emplean en la URL relativa para usar este mismo dataset en todas las peticiones a la REE.

Por otra parte, en la actividad de copia también hay que definir el destino de los datos extraídos. Como mencionamos previamente, los datos se almacenan en primera instancia en la capa raw del ADLS Gen 2.

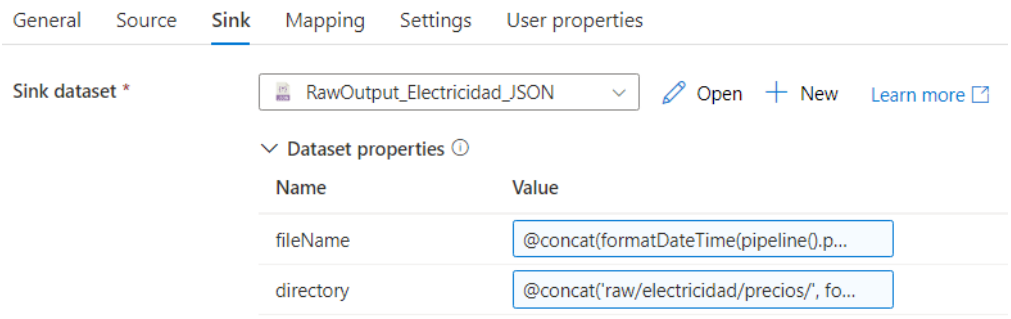


Figura 8. Destino de la actividad copy data

En el caso del dataset destino también está parametrizado para poder mantener la estructura jerárquica vista en la figura 3, de forma que el nombre del fichero tenga el siguiente formato [año]-[mes].json y el directorio en el que se almacena también está parametrizado para tener separados los años en diferentes carpetas. El formato de los ficheros es de tipo JSON ya que en la capa raw se almacenan los datos tal y como vienen del origen. De esta forma se puede mantener una trazabilidad y permite detectar el origen de los posibles errores cometidos en las diferentes transformaciones.

Los datasets de Synapse requieren de un conector para su creación. Estos conectores se denominan linked services y existen varios tipos.

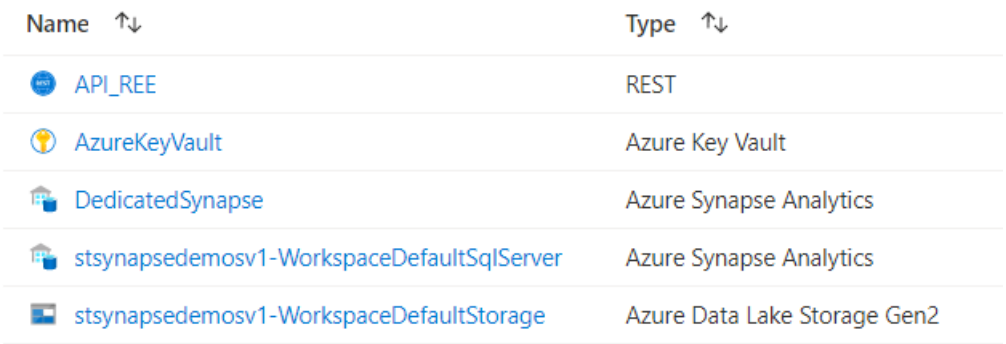


Figura 9. Linked Services

- Notebook. Los notebooks de Synapse siguen la misma filosofía que los Jupyter notebooks mezclando celdas de texto y celdas de código. En las celdas de código se pueden emplear varios lenguajes como Scala, SQL o PySpark, siendo este último el empleado en todo el ejercicio. A través de Python se pueden leer los datos de la capa raw, almacenarlos temporalmente en una estructura de datos como un dataframe, realizar las transformaciones pertinentes y posteriormente guardarlo en la capa clean o delivery del datalake.

El procedimiento es el mismo para el resto de pipelines que obtienen información de electricidad con la diferencia de que se deben cambiar los parámetros para hacer la llamada pertinente en la API.

Por último, una vez que se han almacenado los datos en las correspondientes capas, se deben agregar los datos para que estén al mismo nivel de detalle, es decir, transformar los datos para que estén a nivel diario. Si recordamos la figura 4, podemos ver que la última actividad consiste en un dataflow el cual veremos en detalle posteriormente. Este dataflow realiza una agregación para que los datos estén a nivel diario como mencionamos previamente.

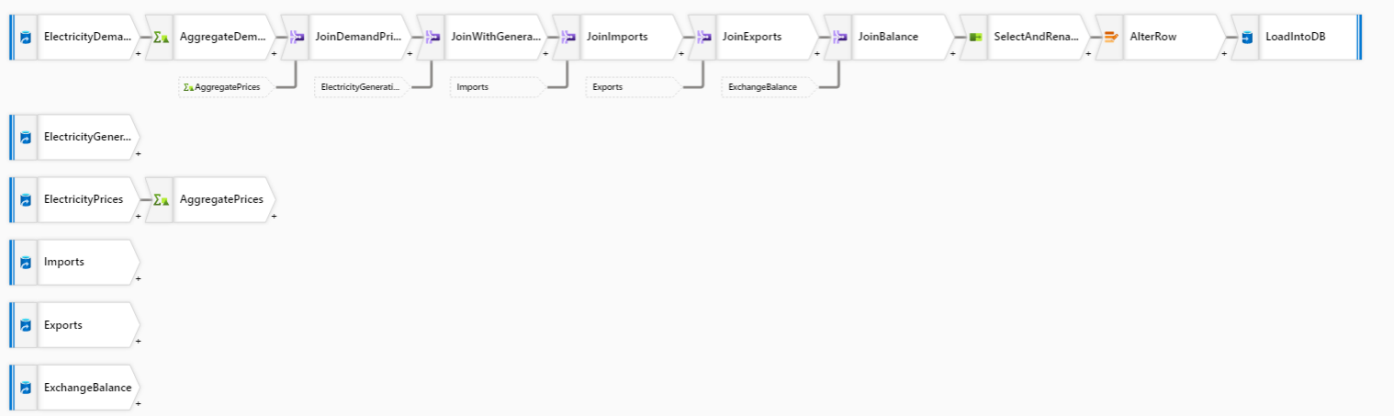


Figura 10. Dataflow electricidad

En este dataflow, se utilizan varios orígenes de datos, que se corresponden con los diferentes datasets creados para los ficheros almacenados en la capa de delivery. Estos ficheros están almacenados en formato parquet. Para poder trabajar al mismo nivel de detalle, hay que realizar una agregación a los precios de la electricidad y a la demanda energética, ya que la API proporciona estos datos a nivel horario mientras que el resto está a nivel diario.

Una vez se han agregado a nivel diario, se pueden unir todos los ficheros con el fin de crear una sola tabla que contenga todos los datos de la electricidad. Esto depende del modelo que se quiera implementar posteriormente podría decidirse hacer una tabla para los precios, otra para la generación, etc. pero para este ejercicio, por simplicidad, se ha decidido usar una sola

tabla que será una de las tablas de hechos de nuestro modelo final. Para realizar estas uniones, se utiliza la operación INNER JOIN que une los diferentes ficheros.

Por otra parte, también se realiza una operación de SELECT para eliminar las columnas duplicadas y renombrarlas, y se realiza un ALTER ROW para poder realizar una operación de tipo UPSERT en la base de datos de destino. Esto permite insertar los datos que no existían en la base de datos y actualizar los datos ya existentes.

Por último, se ha de destacar que el destino es una tabla perteneciente al pool SQL dedicado. Por ello, es necesario crear previamente la tabla con un script SQL.

```
CREATE TABLE [electricity]
(
    [year] [INT] NOT NULL,
    [month] [INT] NOT NULL,
    [day] [INT] NOT NULL,
    [real_demand] [FLOAT],
    [programmed_demand] [FLOAT],
    ...
)
```

A partir de lo anterior, también se deben crear los pipelines para extraer información sobre las condiciones climatológicas y sobre los combustibles. Estos dos pipelines son muy similares entre sí pues tienen las mismas actividades. Además, en ambos casos solo se realiza una petición a la API ya que, a diferencia de los datos proporcionados por la REE, toda la información que nos interesa se puede recabar con una sola petición. A continuación, los veremos con más detalle:

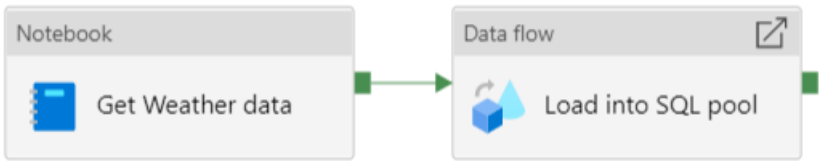


Figura 11. Pipeline de meteorología

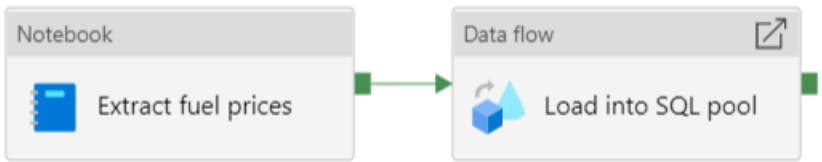


Figura 12. Pipeline de combustibles

Como se puede observar en las figuras 11 y 12, los pipelines constan tan solo de dos actividades que realizan el proceso ELTL.

- Notebook. En este caso, se ha preferido hacer la extracción directamente desde el notebook de Synapse debido a los formatos que devuelven las APIs. Dentro de estos notebooks, además de realizar las peticiones a las diferentes APIs, también se realizan diversas transformaciones para adaptar el formato de los datos correctamente. Además, la información se almacena en las diferentes capas del datalake según el estado del dato.
- Data Flow. De manera similar a lo que ocurre con la electricidad, los datos extraídos de las APIs tienen un nivel de detalle horario

Figura 13. Dataflow meteorología

Figura 14. Dataflow combustibles

Como se puede observar en las imágenes, tras obtener los datos de la capa delivery en formato parquet, se agregan los datos a nivel diario. Además, para que se pueda realizar la operación de UPSERT sobre la base de datos, se inserta un paso intermedio denominado ALTER ROW. Este paso permite actualizar las filas en caso de que se modifiquen los valores para una determinada clave. En este caso, la clave está formada por la fecha y por la provincia ya que, a diferencia de los valores de la electricidad que son a nivel nacional, los valores del precio del combustible y los datos climatológicos son diferentes dependiendo de la provincia. Por último, se define la base de datos destino, el cuál es un pool SQL dedicado. Al igual que en el dataflow de la electricidad, es necesario que se creen las tablas con sentencias SQL previo a la ejecución del dataflow.

Una vez que se han creado los tres pipelines, podemos crear otro pipeline adicional que los ejecute en paralelo.

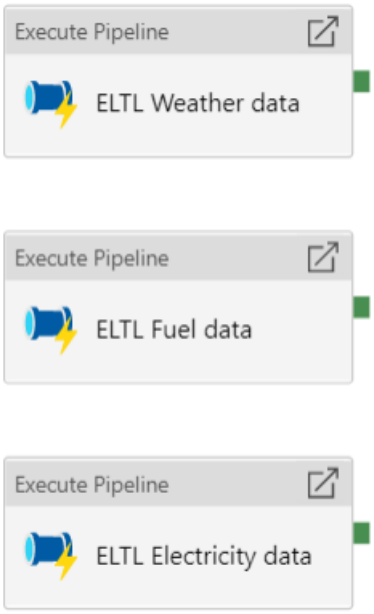


Figura 15. Pipeline ELTL

Este pipeline está parametrizado de forma que el usuario puede introducir el mes y el año del que quiere obtener el histórico de datos. Cada pipeline ajusta internamente los valores de los parámetros para adaptarlos al formato exigido por cada una de las APIs.

4. POOLS SQL

A lo largo de la implementación hemos hecho referencia al pool SQL dedicado, pero ¿qué es realmente un pool SQL y por qué es dedicado?

Los pools SQL son el servicio cloud que proporciona Azure que cumple con las labores de un Data Warehouse tradicional. Los Data Warehouse son plataformas destinadas a almacenar datos que permiten realizar el análisis de estos. Son un elemento fundamental dentro del mundo de Data Analytics y del Business Intelligence, y los pools SQL de Synapse son la solución cloud que proporciona Microsoft.

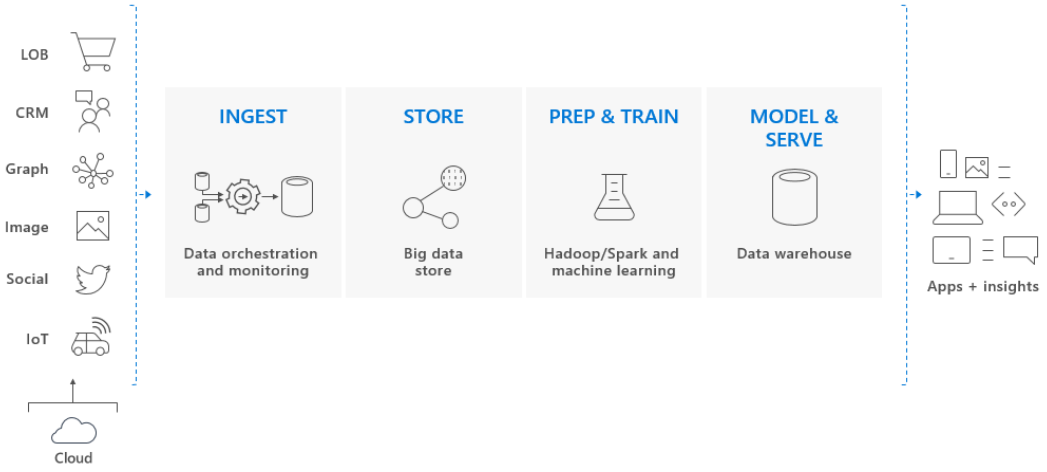


Figura 16. Solución Big Data end-to-end

Dentro de las opciones que proporciona Synapse a la hora de crear los pools SQL, encontramos dos categorías:

- Pools SQL dedicados. Funcionan como un servidor de base de datos SQL tradicional, pero en la nube. Se puede definir su potencia y rendimiento a la hora de crearlo, aunque hay que tener en cuenta que el coste variará según la configuración que se realice.

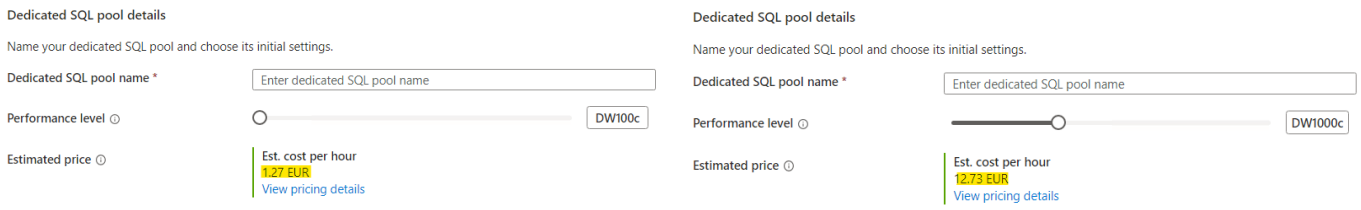


Figura 17. Comparación de precios pool SQL dedicado

Como se puede observar, si se incrementa el rendimiento del pool SQL dedicado, el coste por hora también aumenta. El usuario tiene que decidir qué nivel de rendimiento va a requerir dependiendo del rendimiento que necesite. El rendimiento se mide en DWU que representan una combinación de ciertos recursos como CPU, memoria o velocidad de escritura y lectura. Puede obtener más información aquí.

- Pools SQL serverless. Los pools sin servidor o serverless son más sencillos de cara al usuario ya que los costes se producen por TB de datos procesado y no hay costes por horas a diferencia del pool dedicado. Sin embargo, a diferencia del pool SQL dedicado que permite trabajar con tablas relacionales, el pool serverless está enfocado en realizar consultas a los ficheros del datalake.

En nuestro caso, utilizaremos el pool SQL dedicado ya que queremos tener un modelo relacional para acceder a él desde Power BI y generar nuestro informe. De todas maneras, Synapse crea por defecto un pool serverless que permite explorar los ficheros del datalake.

Name	Type	Status	Size
Built-in	Serverless	Online	Auto
DedicatedDemo	Dedicated	Paused	DW100c

Figura 18. Pools SQL

Como hemos mencionado, el pool serverless nos permite realizar consultas a nuestros ficheros del datalake. Vamos a realizar una prueba con los datos extraídos del precio de la luz.

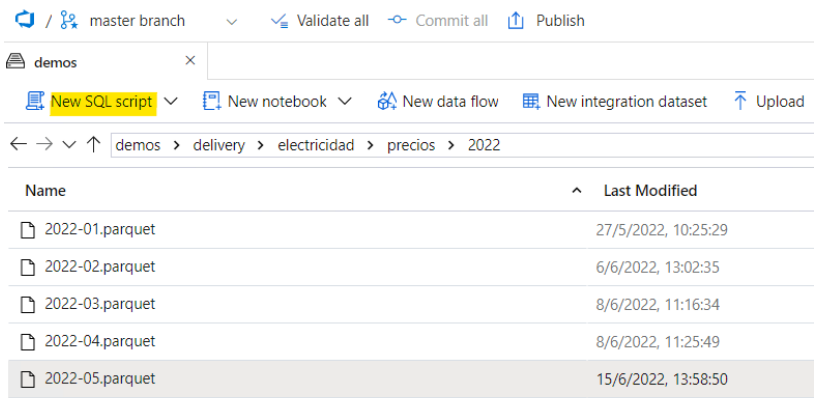


Figura 19. Ficheros de ADLS Gen 2

Si seleccionamos uno de los ficheros y creamos un script SQL, podemos hacer una operación de SELECT para ver los datos que hay en dicho archivo.

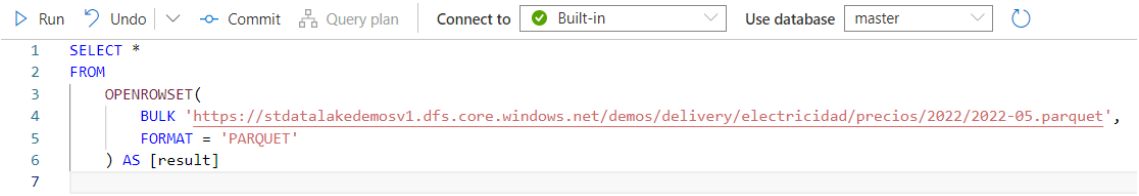
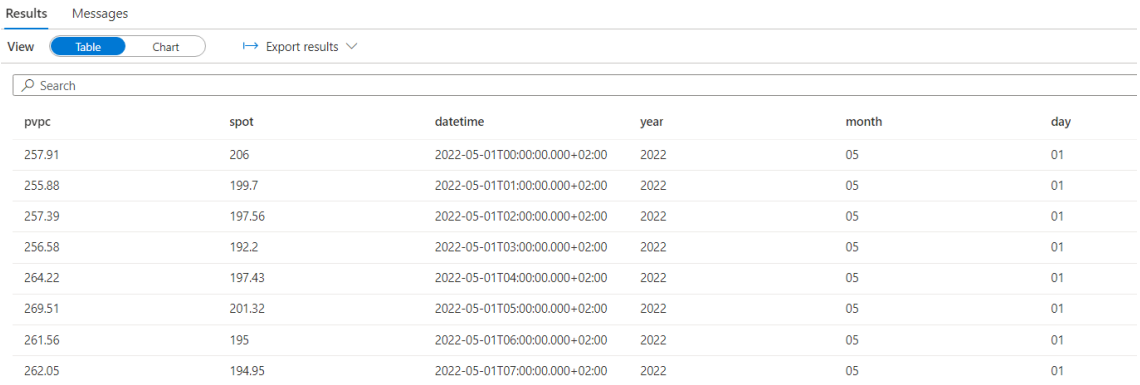


Figura 20. Consulta SQL

Podemos observar que Synapse está utilizando el pool serverless (Built-in) para realizar la operación de consulta. Una vez ejecutamos la operación, podemos observar los resultados en formato de tabla e incluso como un gráfico.



pvpc	spot	datetime	year	month	day
257.91	206	2022-05-01T00:00:00.000+02:00	2022	05	01
255.88	199.7	2022-05-01T01:00:00.000+02:00	2022	05	01
257.39	197.56	2022-05-01T02:00:00.000+02:00	2022	05	01
256.58	192.2	2022-05-01T03:00:00.000+02:00	2022	05	01
264.22	197.43	2022-05-01T04:00:00.000+02:00	2022	05	01
269.51	201.32	2022-05-01T05:00:00.000+02:00	2022	05	01
261.56	195	2022-05-01T06:00:00.000+02:00	2022	05	01
262.05	194.95	2022-05-01T07:00:00.000+02:00	2022	05	01

Figura 21. Resultados en formato de tabla

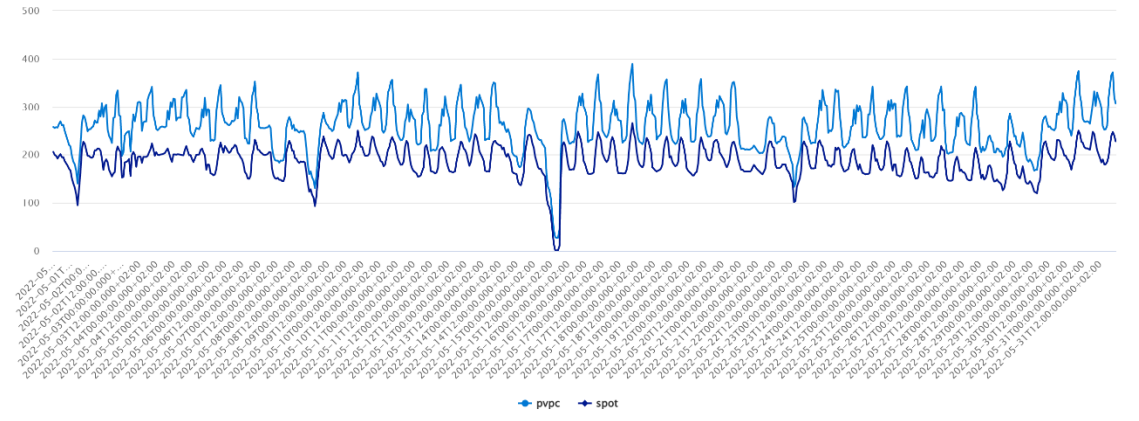


Figura 22. Resultados en formato gráfica

La gráfica que proporciona Synapse es interactiva, pudiendo añadir o quitar las series que queremos mostrar e incluso permite destacar los valores cuando pasamos el cursor por encima como se muestra en el siguiente ejemplo.

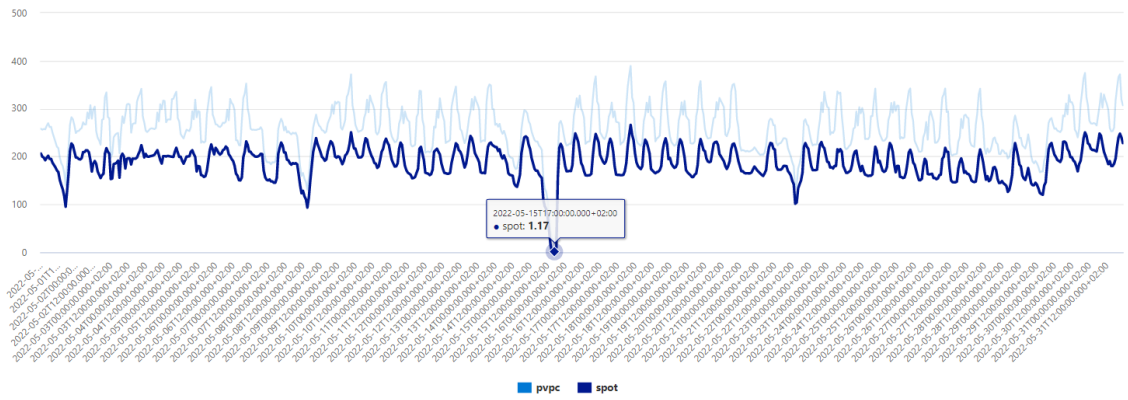


Figura 23. Filtrando la gráfica

De esta forma, podemos realizar un análisis exhaustivo de nuestros datos. Además de todo esto, el pool SQL sin servidor nos permite analizar no solo uno sino varios ficheros de una carpeta. Esto nos permite, por ejemplo, realizar una consulta para obtener todos los datos de los precios de la electricidad del 2022 que están almacenados en la carpeta 2022. Para ello, debemos modificar ligeramente la consulta SQL.

```
SELECT
    [pvpc],
    [spot],
    [datetime]
FROM
    OPENROWSET(
        BULK 'https://stdatalakedemosv1.dfs.core.windows.net/demos/delivery/electricidad/precios/2022/*.parquet',
        FORMAT = 'PARQUET'
    )  with ( [pvpc] FLOAT, [spot] FLOAT, [datetime] VARCHAR(800) )AS [result]
    ORDER BY [datetime]
```

Figura 24. Consulta SQL para todos los ficheros de una carpeta

Como se ha podido ver en los resultados, los datos de los precios de la luz se mantienen a nivel horario en el datalake. Sin embargo, para hacer nuestro informe de Power BI queremos que todos los datos estén a nivel horario. Para ello, mediante los dataflows se agregan estos datos y se almacenan en el pool dedicado.

El pool dedicado está enfocado en trabajar con tablas SQL y no con los ficheros del ADLS. Por este motivo, este pool permite la realización de operaciones CRUD sobre las tablas SQL. Existe una gran diferencia en el funcionamiento de este pool respecto al pool serverless, ya que cuando estamos empleando el pool SQL dedicado, tenemos que encenderlo y apagarlo cuando

dejemos de usarlo. Esto lo podemos hacer desde el portal de Azure o desde el workspace de Synapse, salvo que su inicio y pausa se automatice dentro del pipeline

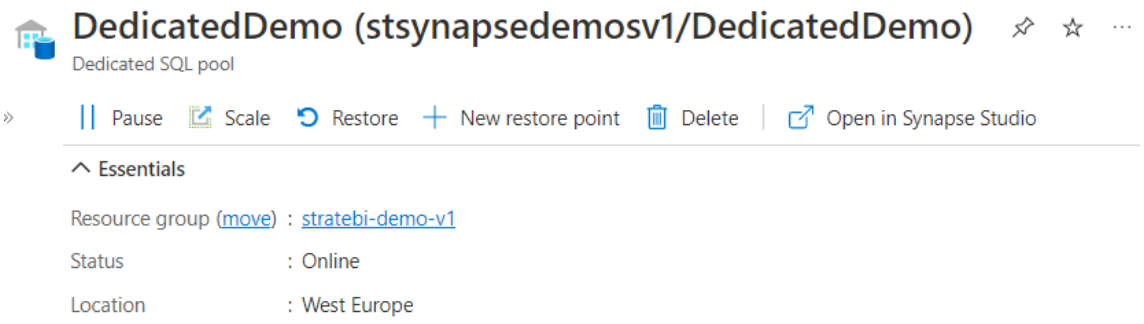


Figura 25. Pool SQL dedicado desde el portal de Azure

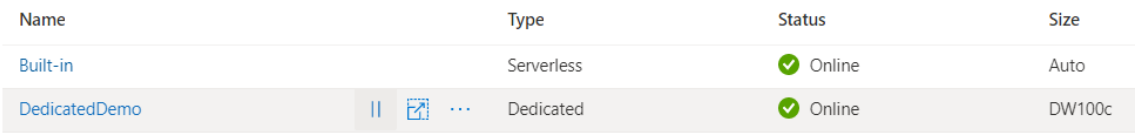


Figura 26. Pool SQL dedicado desde Synapse

Una vez encendido el pool dedicado, podemos empezar a utilizarlo. En la siguiente imagen, se puede observar cómo están las tablas creadas con los datos obtenidos tras haber realizado el proceso ELTL con los pipelines.

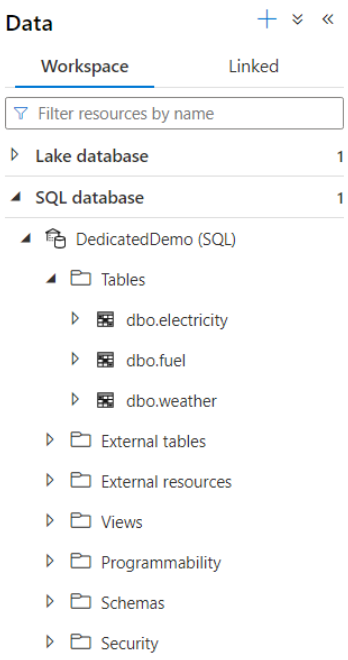


Figura 27. Pool SQL dedicado

Al igual que con el pool serverless, se pueden realizar consultas sobre las diferentes tablas:

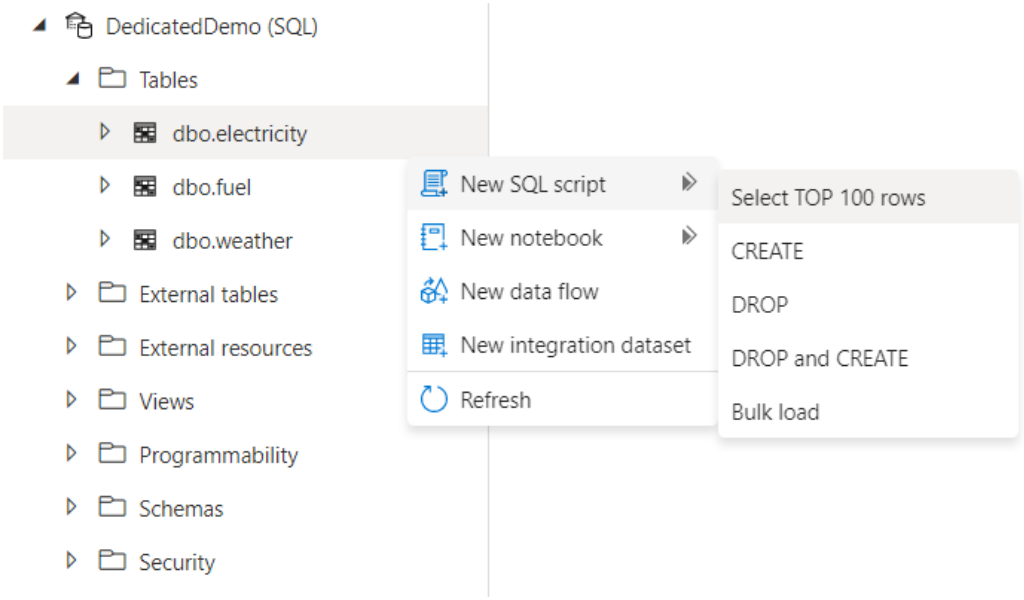


Figura 28. Operaciones con el pool SQL dedicado

Hemos visto cómo podemos emplear los diferentes tipos de pools SQL que ofrece Azure con Synapse. Sin embargo, el detalle de tener que encender y apagar nuestro pool dedicado cuando hacemos el proceso ELTL con los pipelines puede llegarnos a causar que incurramos en costes innecesarios por no apagar el pool cuando terminan de ejecutar las actividades del pipeline. Para ello, vamos a realizar algunas modificaciones en nuestro pipeline para que se inicie el pool antes de ejecutar el ELTL y para que se pause una vez se haya terminado.

Figura 29. Pipeline ELTL con control del pool SQL

Como podemos observar en la imagen, hay una actividad que ejecuta el pipeline que realiza los tres procesos ELTL de las diferentes APIs, y otras dos actividades de tipo “Web” que controlan el pool SQL. Estas actividades realizan llamadas a la API REST de Azure.

La URL que debemos emplear para iniciar el pool SQL es la siguiente:

https://management.azure.com/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Synapse/workspaces/{workspace-name}/sqlPools/{database-name}/resume?api-version=2019-06-01-preview

Para la actividad de parar el pool SQL es similar:

https://management.azure.com/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Synapse/workspaces/{workspace-name}/sqlPools/{database-name}/pause?api-version=2019-06-01-preview

Además, debemos realizar algunas configuraciones en la actividad para que Synapse pueda tener permisos suficientes a la hora de modificar el estado del pool SQL dedicado.

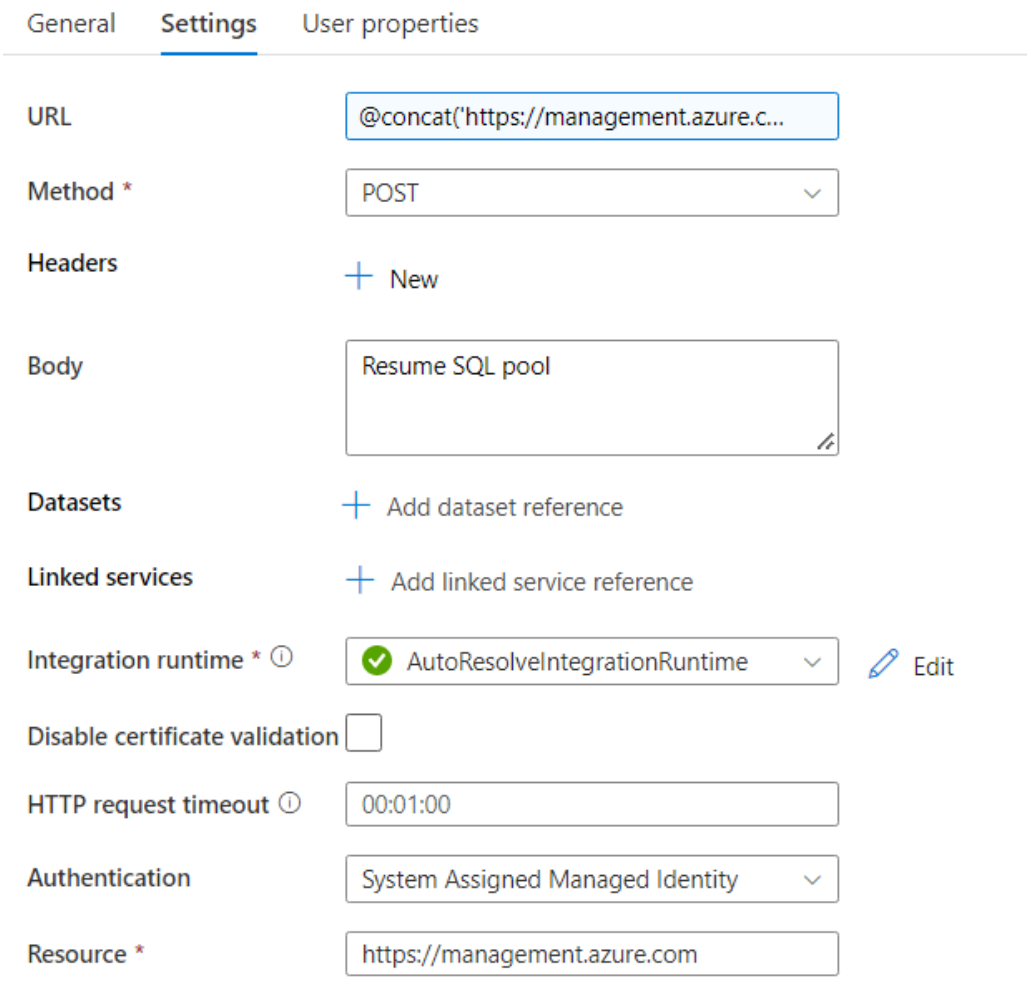


Figura 30. Configuración de las actividades web

También se deben asignar permisos de "Contributor" desde el portal de Azure al propio service principal de Synapse. Esto se puede realizar en el portal web en el apartado de "Access Control (IAM)":

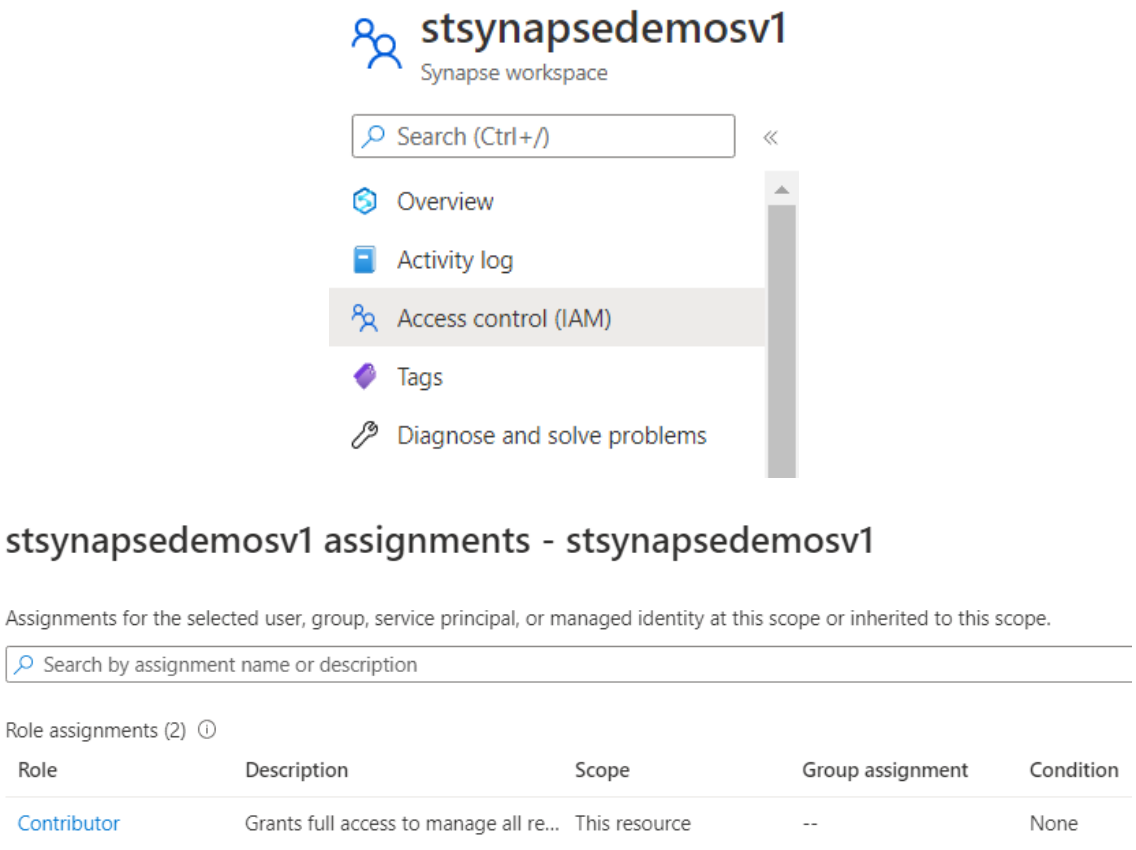


Figura 31. Permisos del workspace de Synapse en el portal de Azure

Una vez otorgados los permisos, se pueden ejecutar correctamente las actividades web.

5. VISUALIZACIÓN CON POWER BI

Una vez se han definido los pipelines asociados al proceso ELTL, podemos visualizar los datos mediante un informe de Power BI. Para ello, desde Power BI Desktop, seleccionamos la opción de "Obtener datos" y buscamos la opción de Synapse:

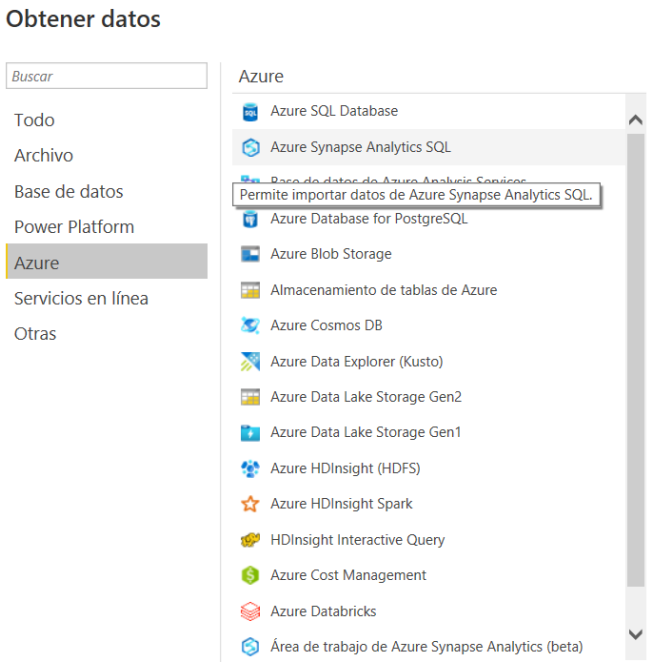


Figura 32. Orígenes de datos en Power BI

Una vez hemos elegido esta opción, previo a seguir con la configuración del origen de datos, es necesario que se inicie el pool SQL dedicado ya que, de lo contrario, la conexión entre Power BI Desktop y Synapse será fallida. Con el pool iniciado, se debe configurar el acceso al servidor desde Power BI.

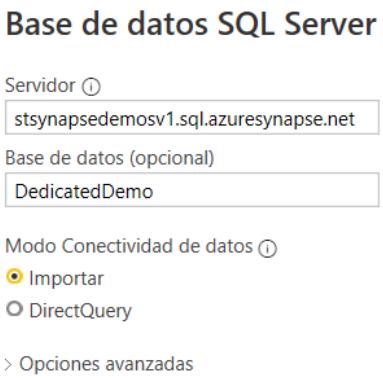


Figura 33. Configuración conexión Synapse SQL

La dirección del servidor se puede localizar desde el portal de Azure:

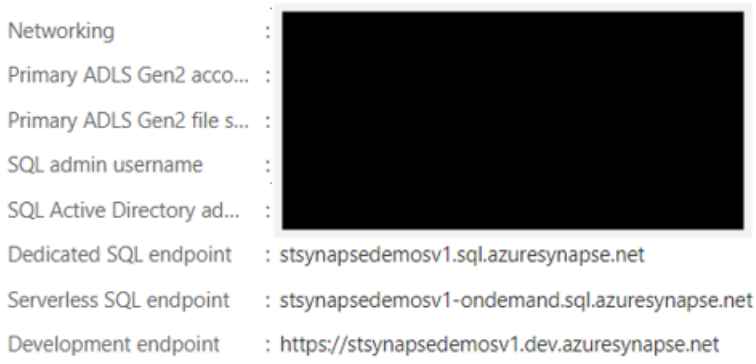


Figura 34. Configuración del endpoint de Synapse

En este caso, debemos utilizar el endpoint dedicado, sobre el que se han creado todas las tablas resultantes del proceso ELTL. Tras identificarnos con nuestro usuario, podremos cargar las tablas de nuestro modelo:

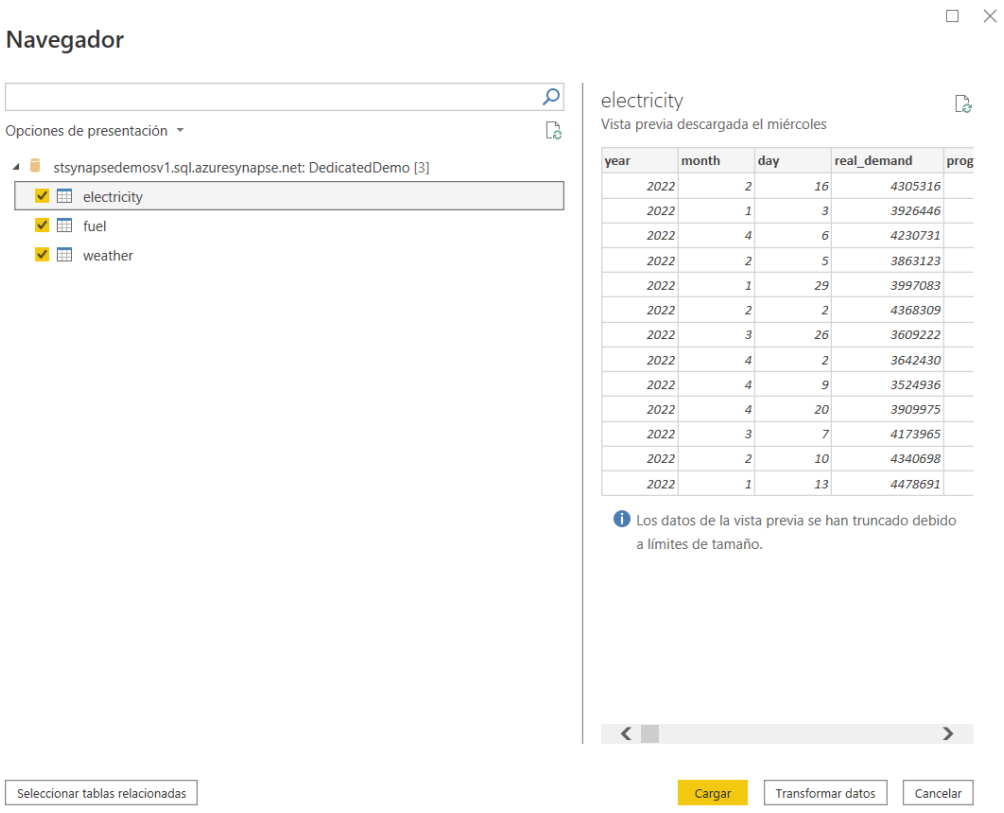


Figura 35. Tablas de Synapse en Power BI

Una vez hemos cargado los datos, podemos realizar las transformaciones que consideremos pertinentes para nuestro modelo como la obtención de dimensiones para nuestra tabla de hecho. Es importante tener en cuenta que, dado que el modo de carga establecido para Power BI es “Import”, una vez se hayan cargado los datos, se puede apagar el pool SQL dedicado para no incurrir en costes innecesarios.

Con el modelo preparado, podemos realizar nuestro informe. A modo demostrativo, se muestran los siguientes ejemplos:

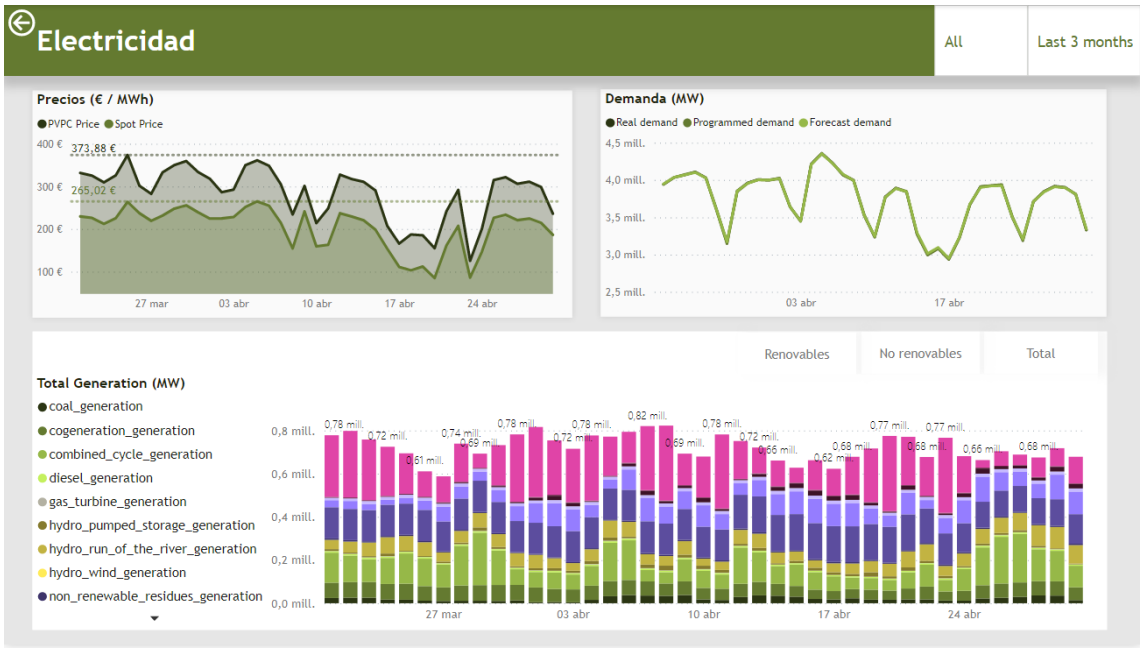


Figura 36. Análisis electricidad en Power BI

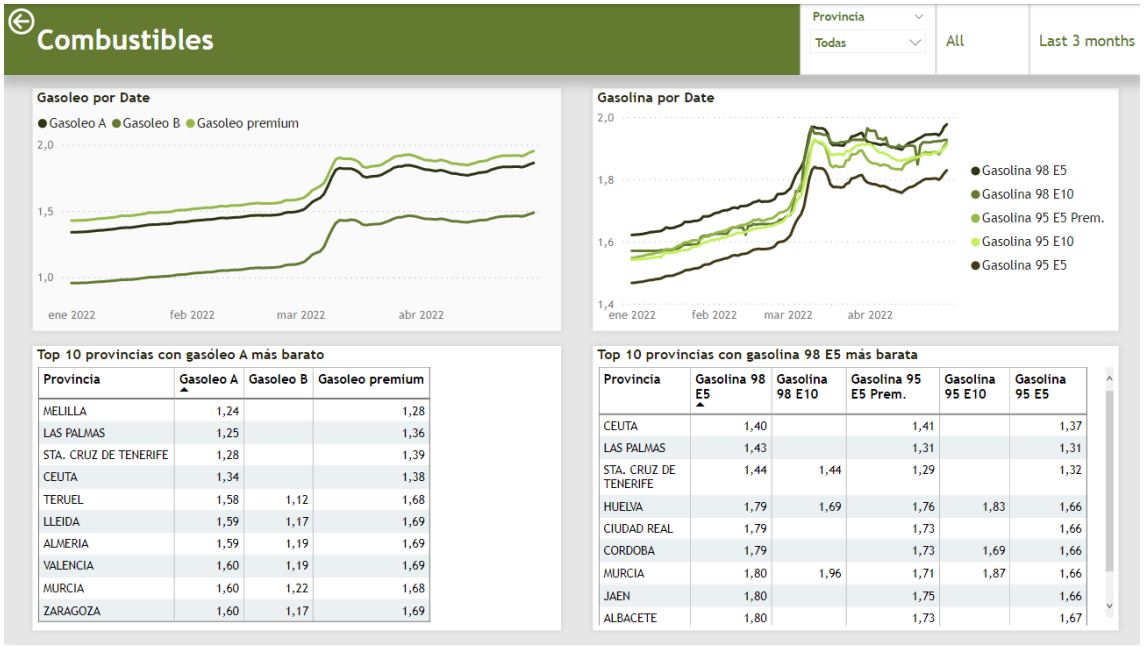


Figura 37. Análisis del precio del combustible en Power BI

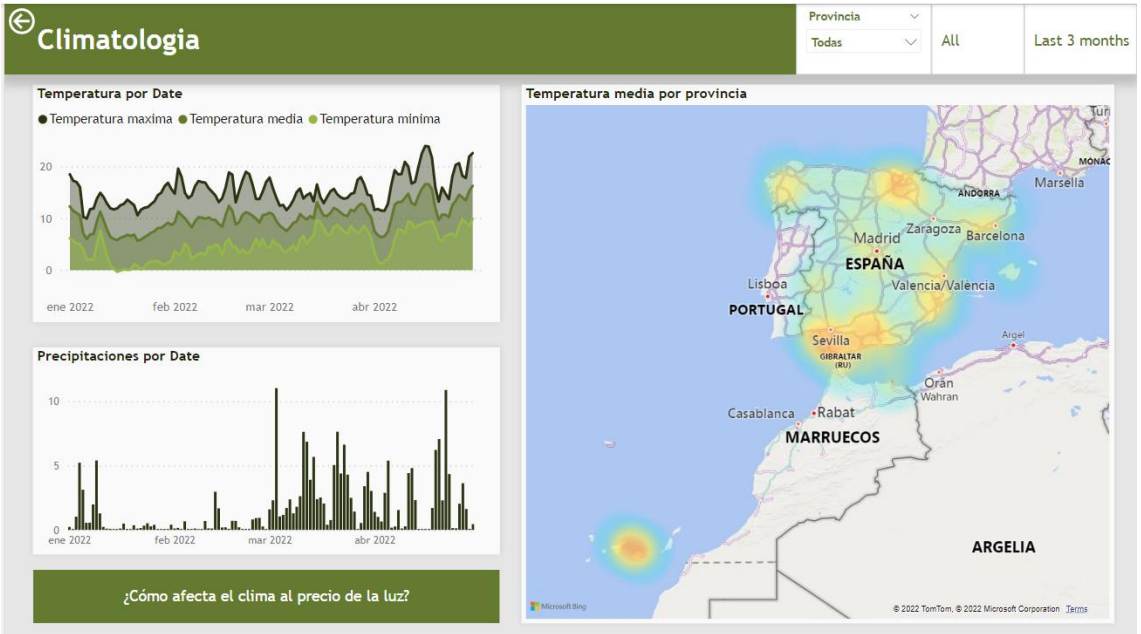


Figura 38. Análisis del clima en Power BI

Por otra parte, una vez hemos creado el informe lo podemos publicar en un área de trabajo o workspace.

Figura 39. Workspace de Power BI

Gracias a la API del servicio de Power BI, podemos actualizar el dataset con nuevos datos. Esto permite actualizar el dataset desde el pipeline que teníamos en Synapse con una Web Activity.

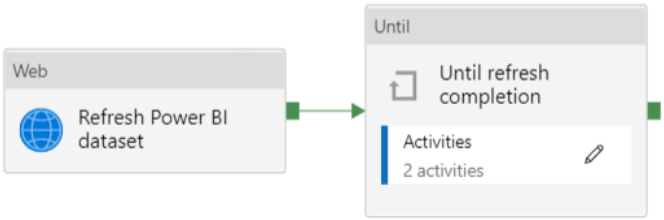


Figura 40. Pipeline actualización dataset

A continuación, vamos a analizar en detalle el pipeline. En primer lugar, se encuentra la web activity que, como hemos mencionado, se encarga de realizar una petición a la API de Power BI. En la siguiente imagen, se puede observar la configuración de la actividad:

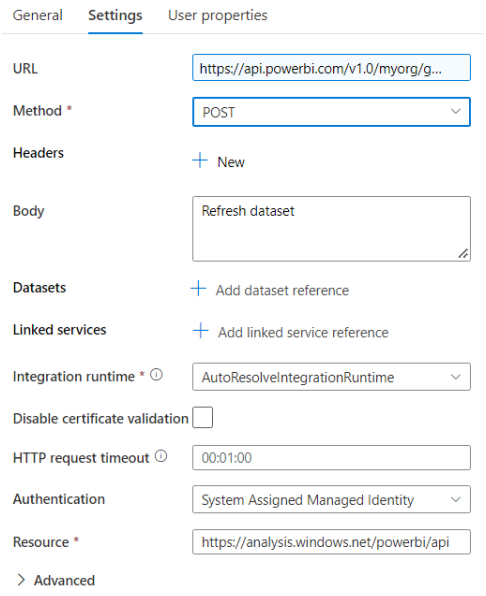


Figura 41. Configuración de la actividad

La URL que se emplea es la siguiente:

`https://api.powerbi.com/v1.0/myorg/groups/{group_id}/datasets/{dataset_id}/refreshes`

Tanto el group_id como el dataset_id se pueden obtener desde el servicio de Power BI.

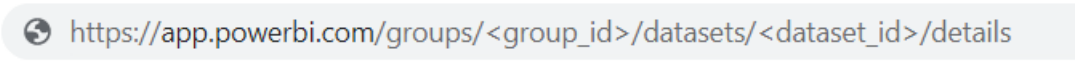


Figura 42. Dataset id y group id en Power BI Service

Hay que tener en cuenta que para que Synapse pueda actualizar el dataset de Power BI, el service principal de Synapse debe tener asignados los permisos necesarios al workspace de Power BI. Para ello, debemos incluir el service principal de Synapse en uno de los grupos de seguridad de Active Directory y otorgar permiso de *Contributor* al workspace donde se encuentra el dataset.

Una vez configurada la actualización desde Synapse, nos queda un último paso que realizar. Si recordamos nuestro pipeline, después del ELTL, se apaga el pool SQL dedicado. La actividad de refresco del dataset, se debe añadir al pipeline principal antes de apagar el pool SQL. Por otra parte, como se observa en la figura 40, es conveniente añadir un bucle que espere hasta que el dataset esté actualizado para que no haya problemas con la actualización si apagamos el pool SQL antes. Este bucle tiene la siguiente forma:

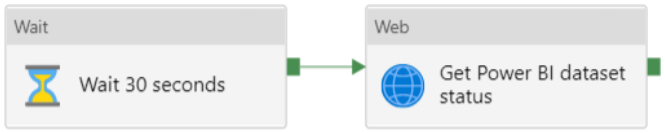


Figura 43. Bucle de espera actualización del dataset

En este bucle, se espera un tiempo determinado, establecido de forma arbitraria y tras ese periodo, se obtiene el estado de la actualización. Esta acción se repite hasta que se haya completado la actualización. Una vez configurada la actualización del dataset podemos incluir la actividad en el pipeline principal.

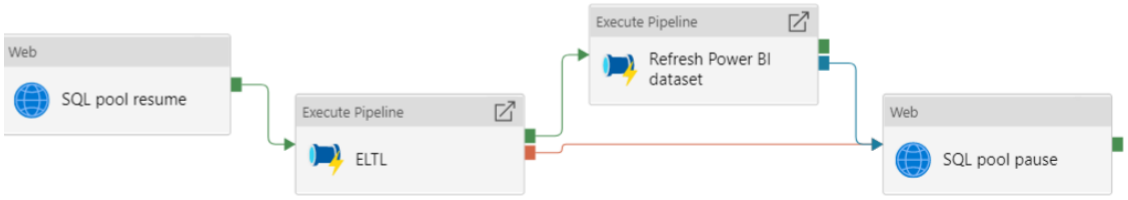


Figura 44. Pipeline de Synapse

6. AUTOMATIZACIÓN CON TRIGGERS

Por último, vamos a automatizar el pipeline para que se ejecute cada mes obteniendo los datos del mes anterior. Para lograrlo, se van a emplear los triggers. Existen diferentes tipos de triggers que se activan por eventos personalizados, eventos relacionados con el almacenamiento o eventos temporales. En este caso, el trigger va a ser temporal y se va a activar el día 1 de cada mes a las 12:00. En la siguiente imagen, se puede observar cómo es la configuración de este trigger:

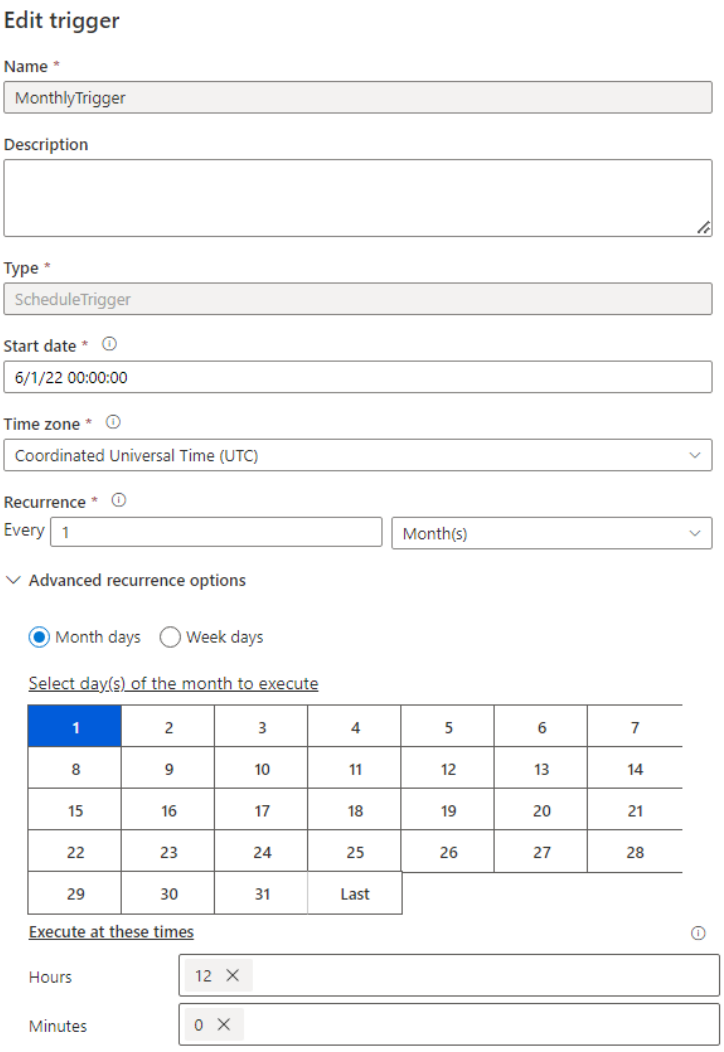


Figura 45. Trigger mensual

Al realizar la configuración del trigger, también se deben establecer los parámetros a emplear.

Name	Type	Value
subscriptionId	string	Value
poolName	string	Value
resourceGroup	string	Value
start_time	string	@trigger.startTime

Figura 46. Parámetros del trigger

En este caso, solo se asigna uno de los parámetros y el resto se dejan por defecto. El parámetro start_time sirve para calcular el mes anterior del que se quiere extraer la información. Para obtener los datos del mes y año que se quieren extraer, se utilizan variables. Para asignar estas variables tenemos que modificar el pipeline una última vez añadiendo dos actividades de tipo "Set variable".

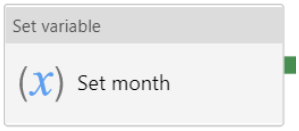


Figura 47. Set variable month

Mediante las expresiones de Synapse, podemos calcular el mes anterior:

```
@formatDateTime(subtractFromTime(formatDateTime(pipeline().parameters.start_time, 'yyyy/MM/ddThh:mm'), 1, 'MONTH'), 'MM')
```

De manera análoga, calculamos el año correspondiente al mes anterior:

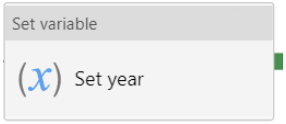


Figura 48. Set variable year

```
@formatDateTime(subtractFromTime(formatDateTime(pipeline().parameters.start_time, 'yyyy/MM/ddThh:mm'), 1, 'MONTH'), 'yyyy')
```

Finalmente, el pipeline queda de la siguiente forma:

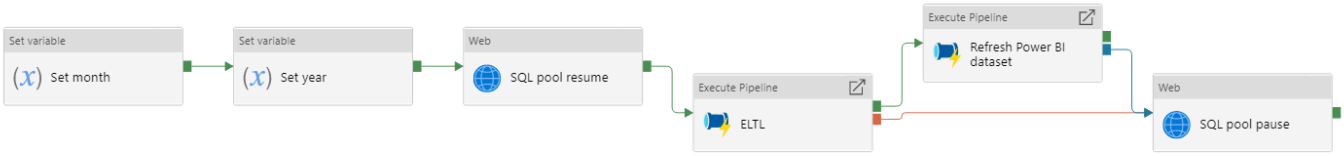


Figura 47. Pipeline parametrizado

7. CONCLUSIONES

Hemos visto a lo largo de este documento cómo utilizar Azure Synapse Analytics, con el que hemos hecho referencia a capacidades de almacenamiento, exploración y procesamiento de datos. En lo que respecta al procesamiento, hemos visto que con los pipelines podemos obtener datos diferentes orígenes y cómo tratarlos para que se adapten al mismo formato con notebooks y dataflows. Además, hemos visto cómo gestionar los ADLS Gen 2 en diferentes capas para mantener el dato en los diferentes estados desde su extracción pasando por diferentes transformaciones.

Por otra parte, hemos demostrado cómo integrar Power BI con Synapse Analytics mediante los pools SQL dedicados. Además, para un correcto uso de los servicios de Azure, hemos podido aprovechar las APIs de Azure y de Power BI para automatizar muchos de los procesos como el refresco de datos o la gestión del estado del pool dedicado.

En definitiva, se ha llevado a cabo un paseo por Azure Synapse Analytics, como pieza clave en las arquitecturas modernas de datos cuando hablamos de ecosistema Azure de Microsoft.