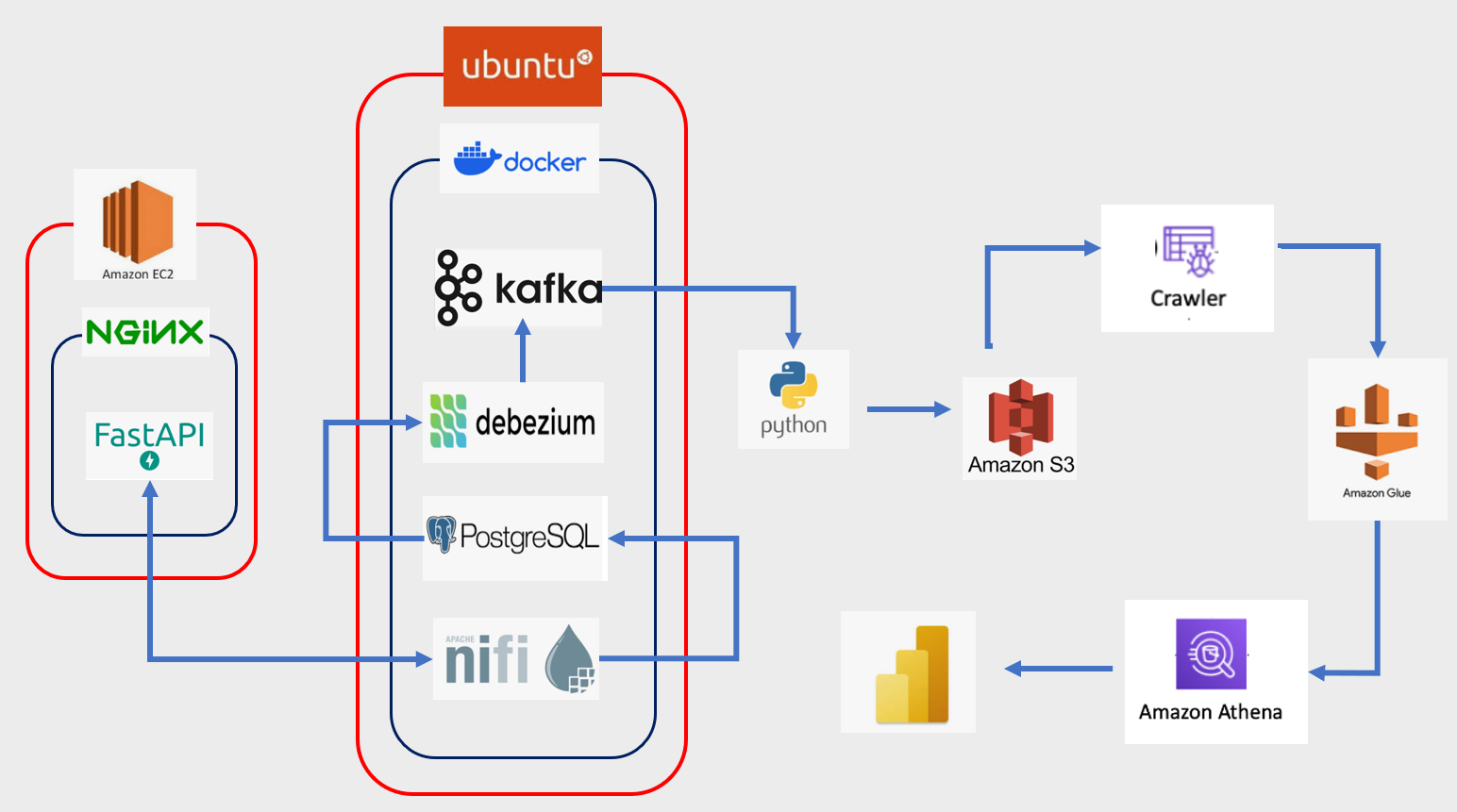
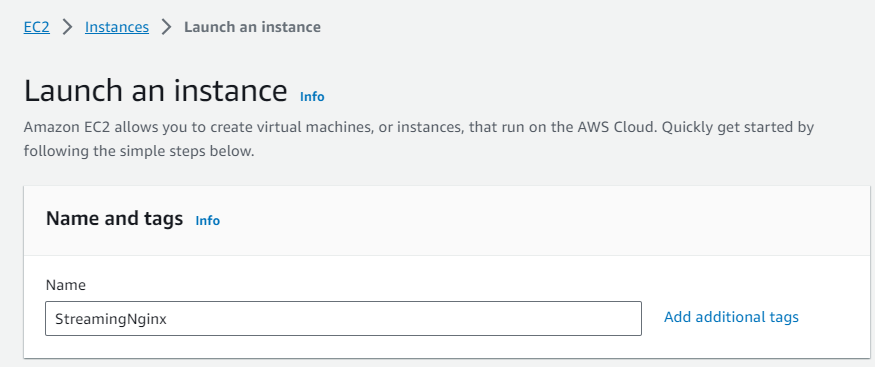
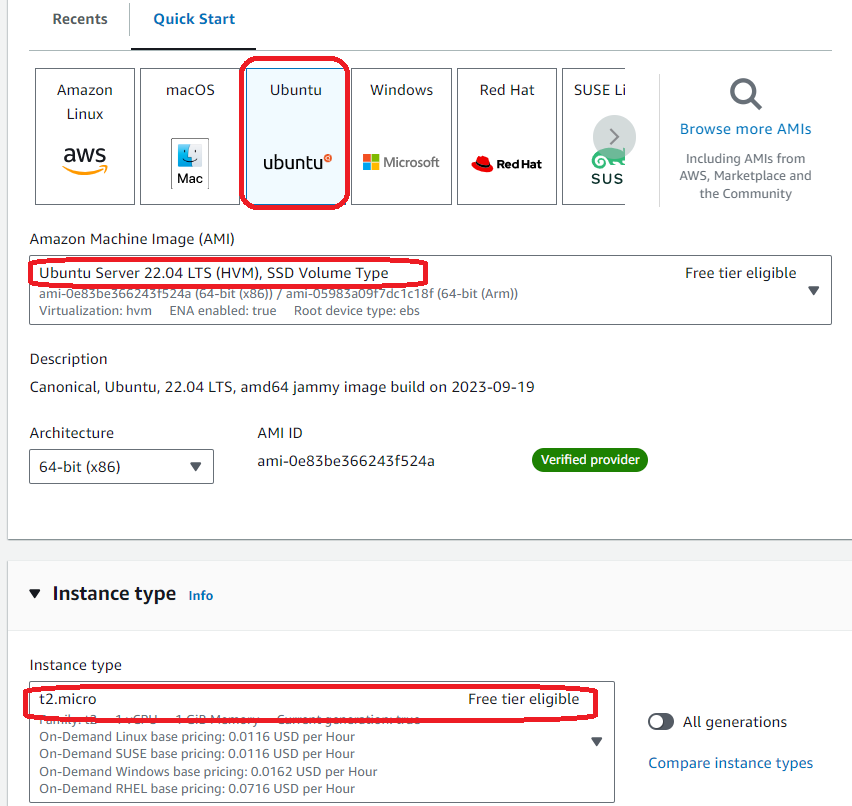
**Projet de streaming de données**

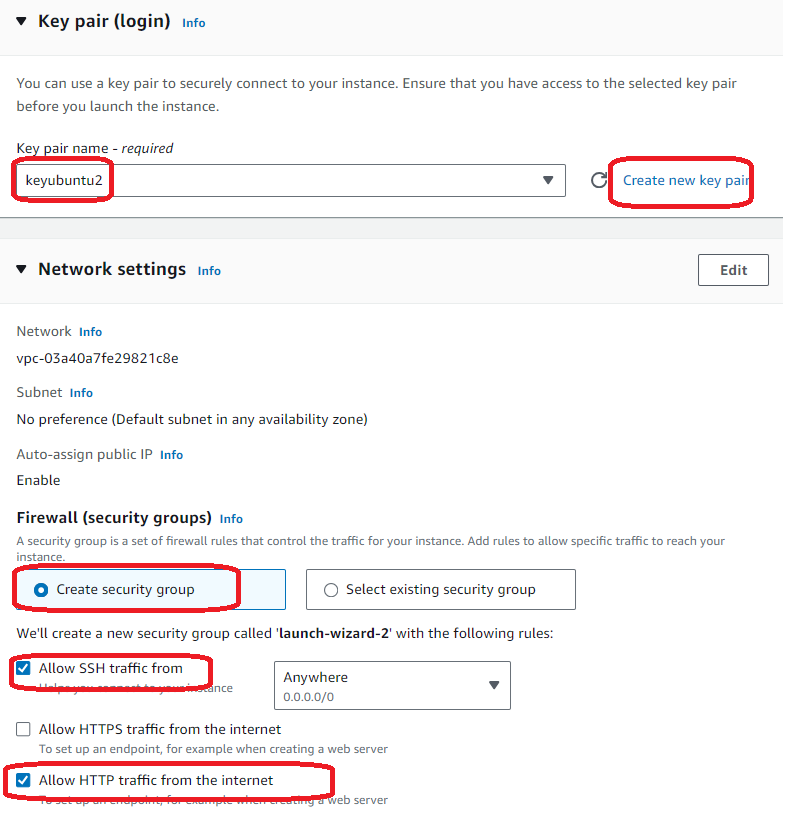
****

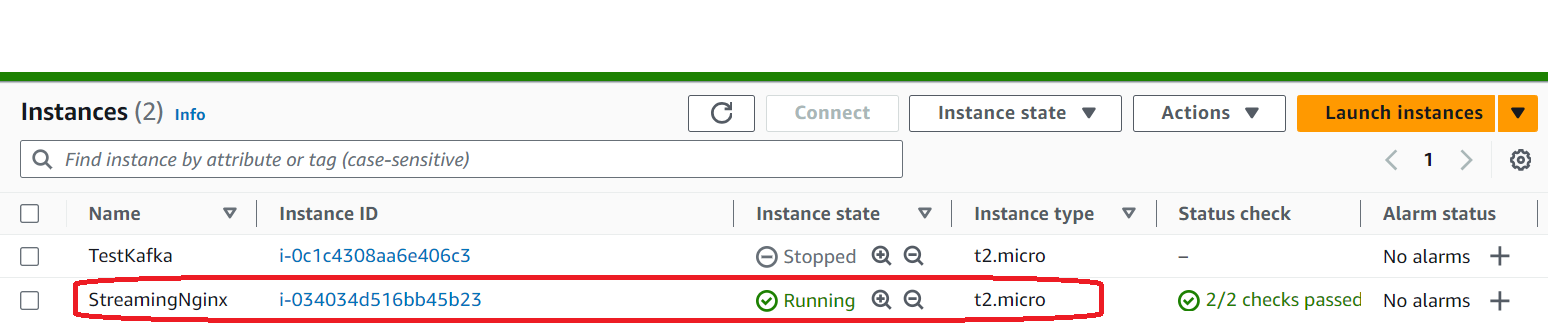
**Instalación de FASTPI en NGINX**

Creamos una instancia en EC2 con Ubuntu para instalar NGINX



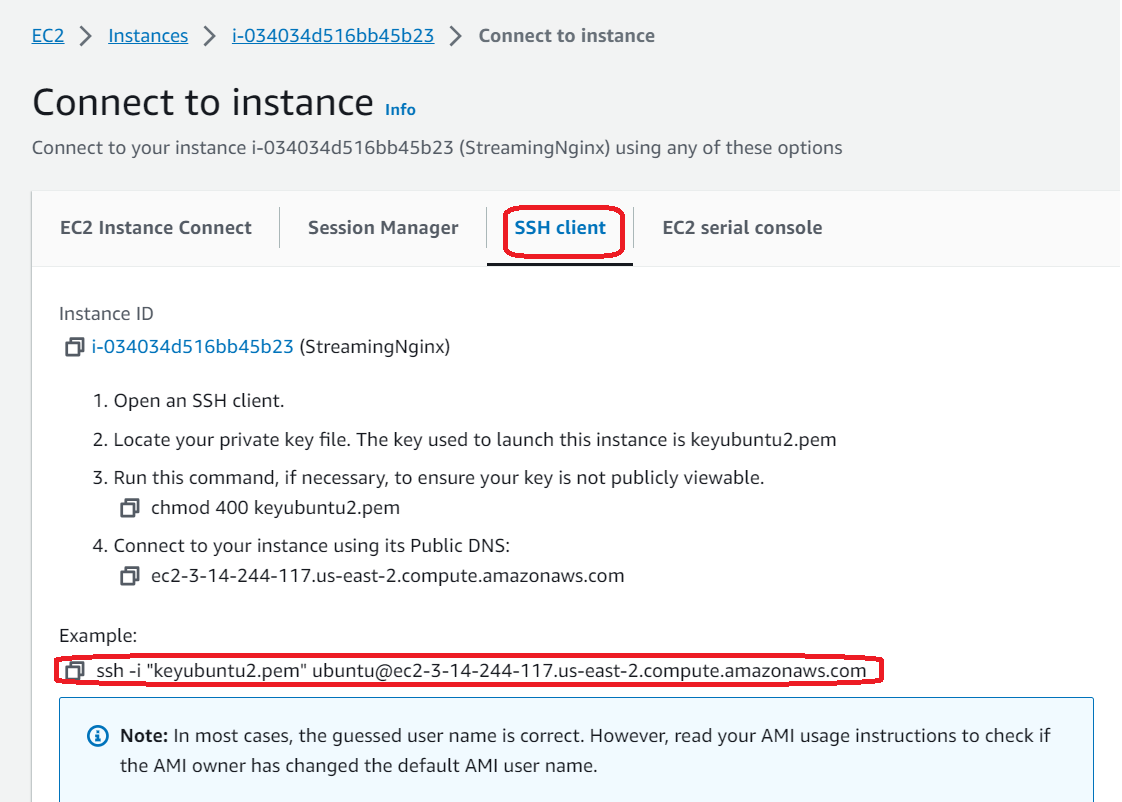


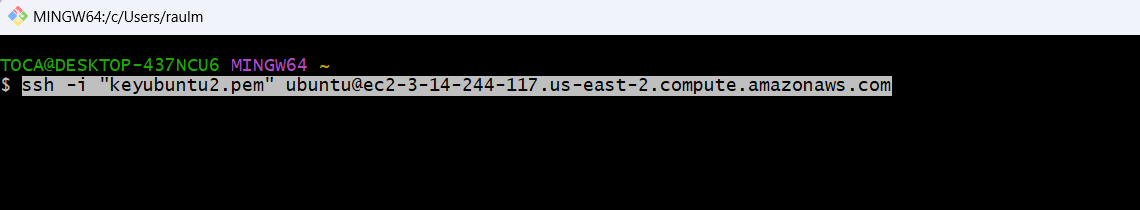




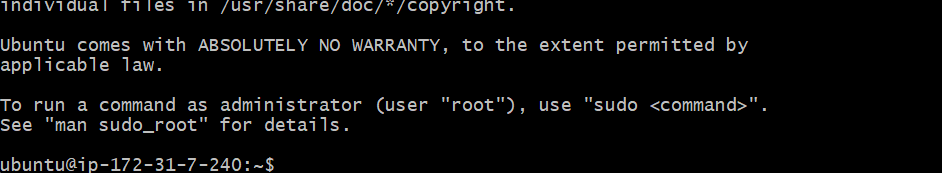
Nos conectamos a la instancia creada mediante un terminal.

Copiamos el comando de conexión



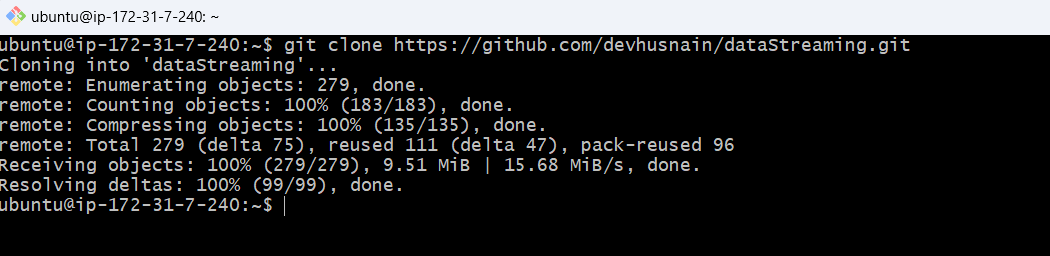


Ya estamos conectados



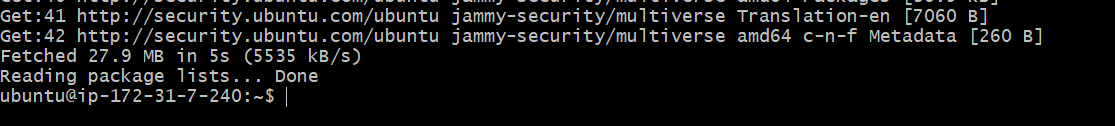
Vamos a clonar un repositorio de github

git clone <https://github.com/devhusnain/dataStreaming.git>

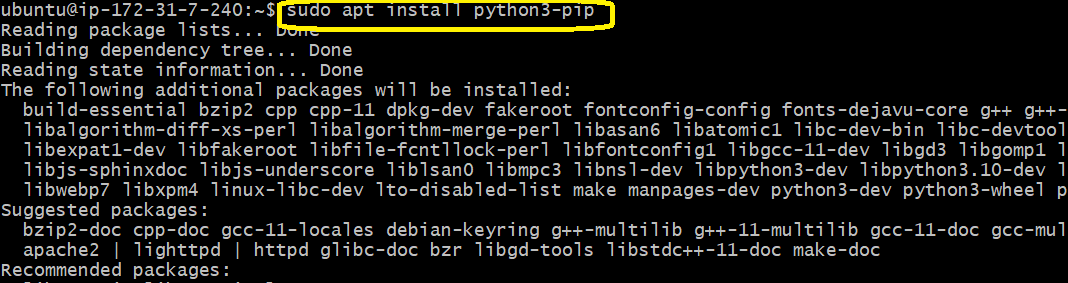


use estos comandos para instalar python-pip y los requisitos para fastapi

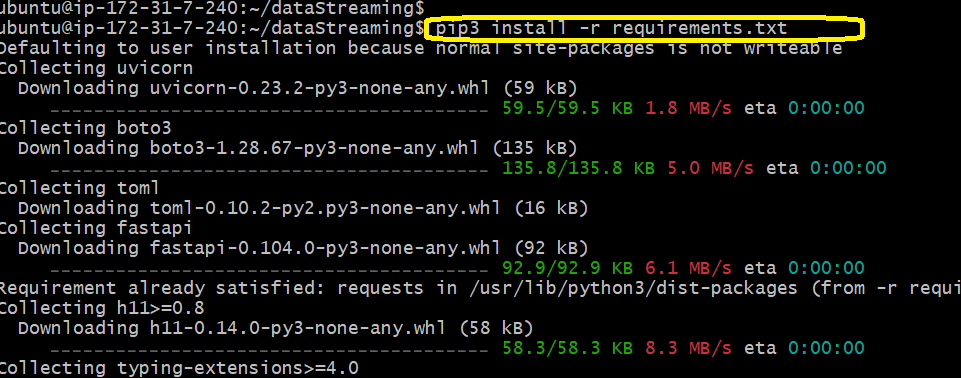
sudo apt-get update



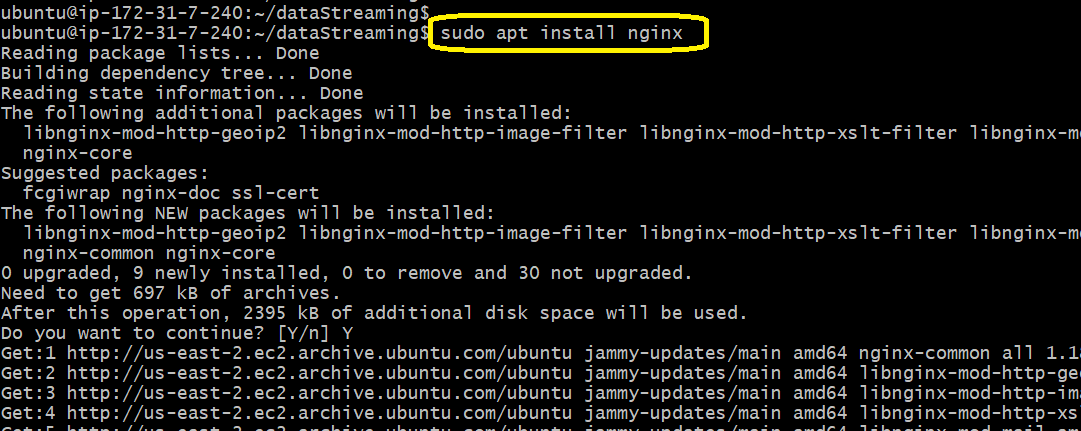
sudo apt install python3-pip



pip3 install -r requirements.txt

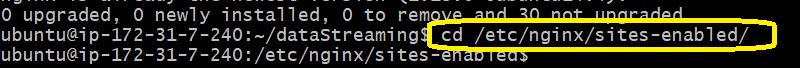


sudo apt install nginx



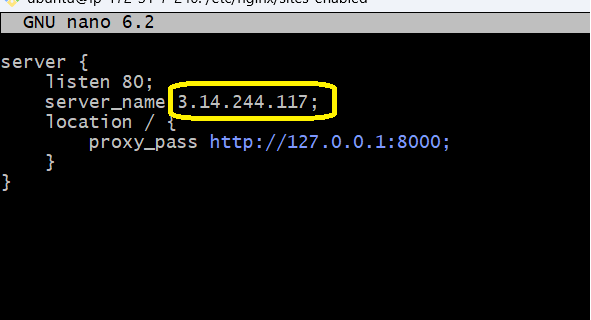
Cambiamos de carpeta con el comando

cd /etc/nginx/sites-enabled/

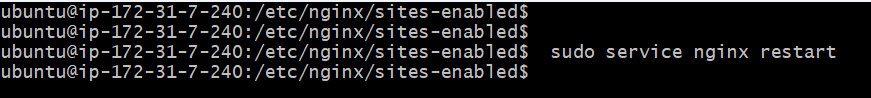


Luego ceamos el archivo fastapi\_nginx con el editor NANO con el siguiente comando

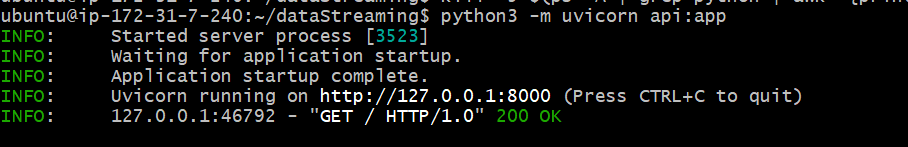
sudo nano fastapi\_nginx



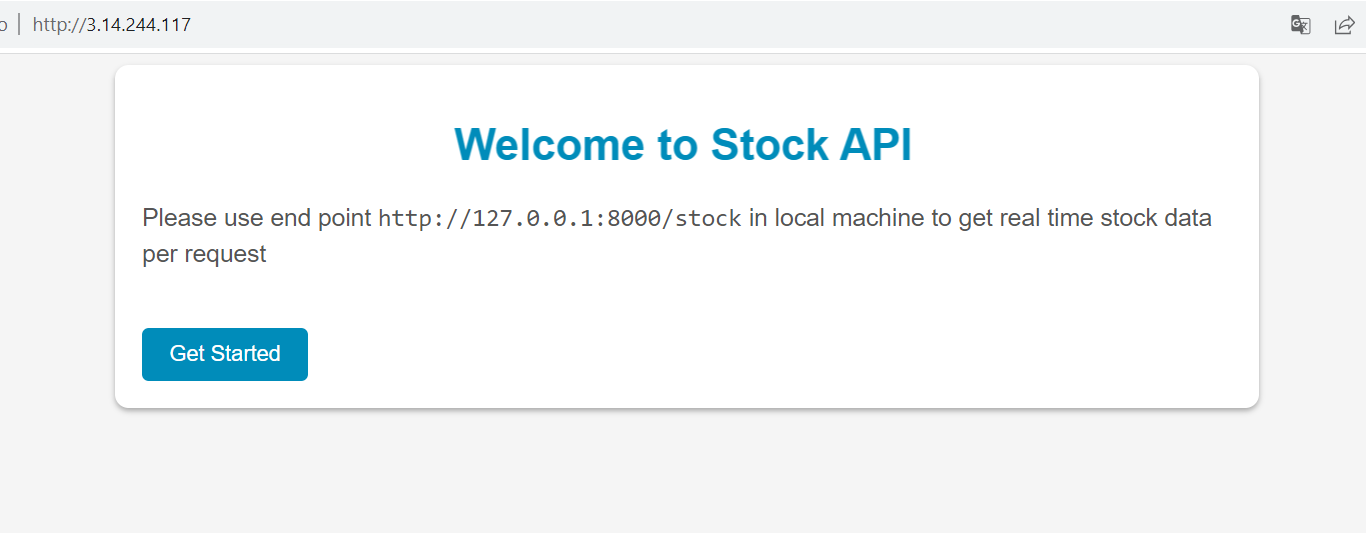
Ejecutamos lo siguiente para reiniciar le servicio nginx

sudo service nginx restart

Luego regresamos a la carpeta dataStreaming que clonamos de github y ejecutamos el fastapi creado

python3 -m uvicorn api:app

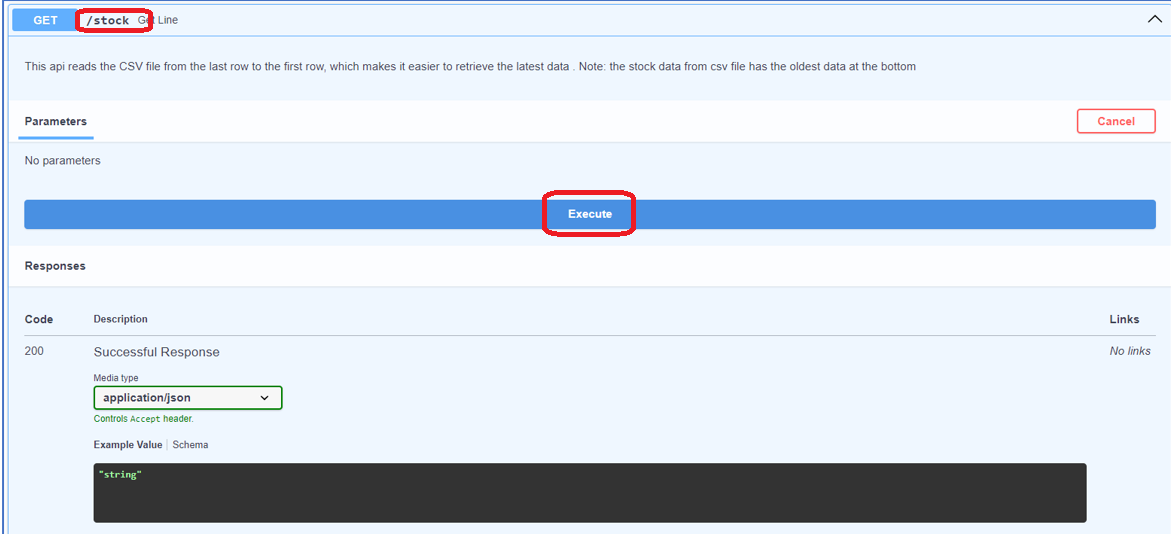
El servicio esta activo y abrimos la IP publica del la instancia EC2



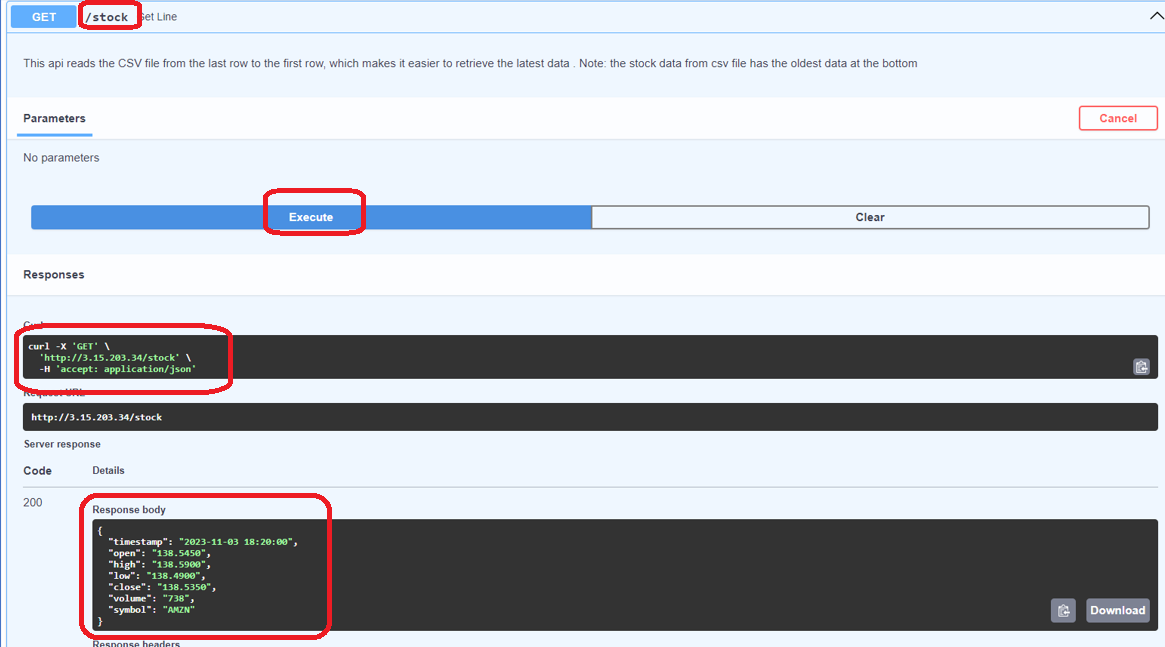
Probamos los procesos de FASTAPI



Seleccionamos STOCK y ejecutamos



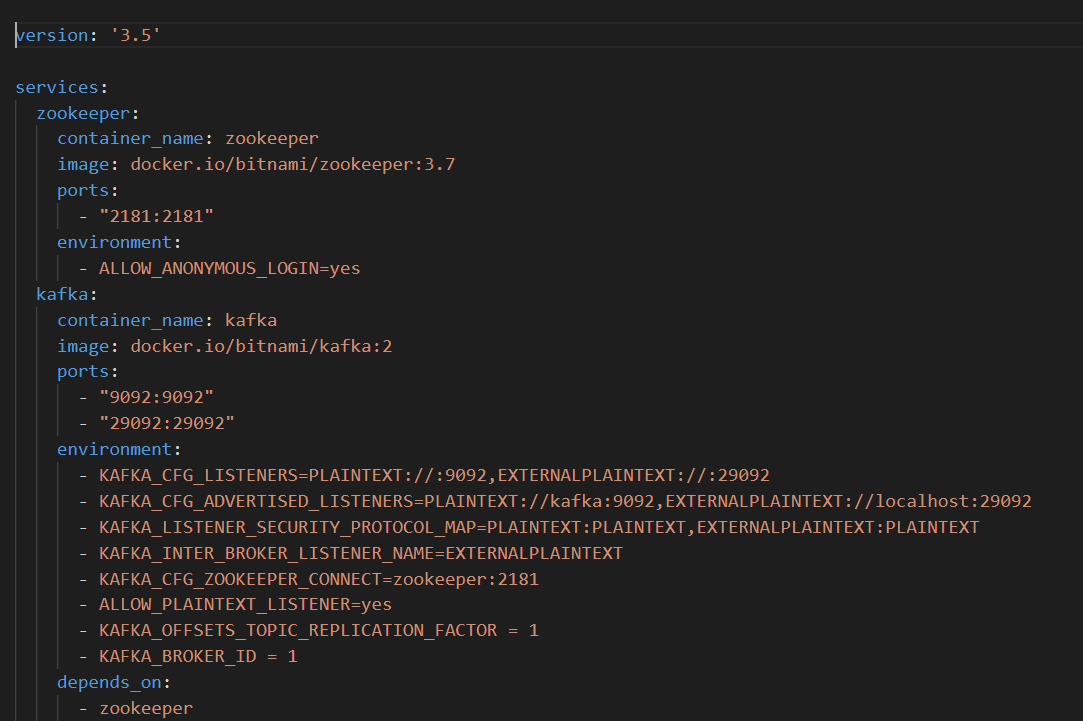
El API nos muestra el resultado en formato json

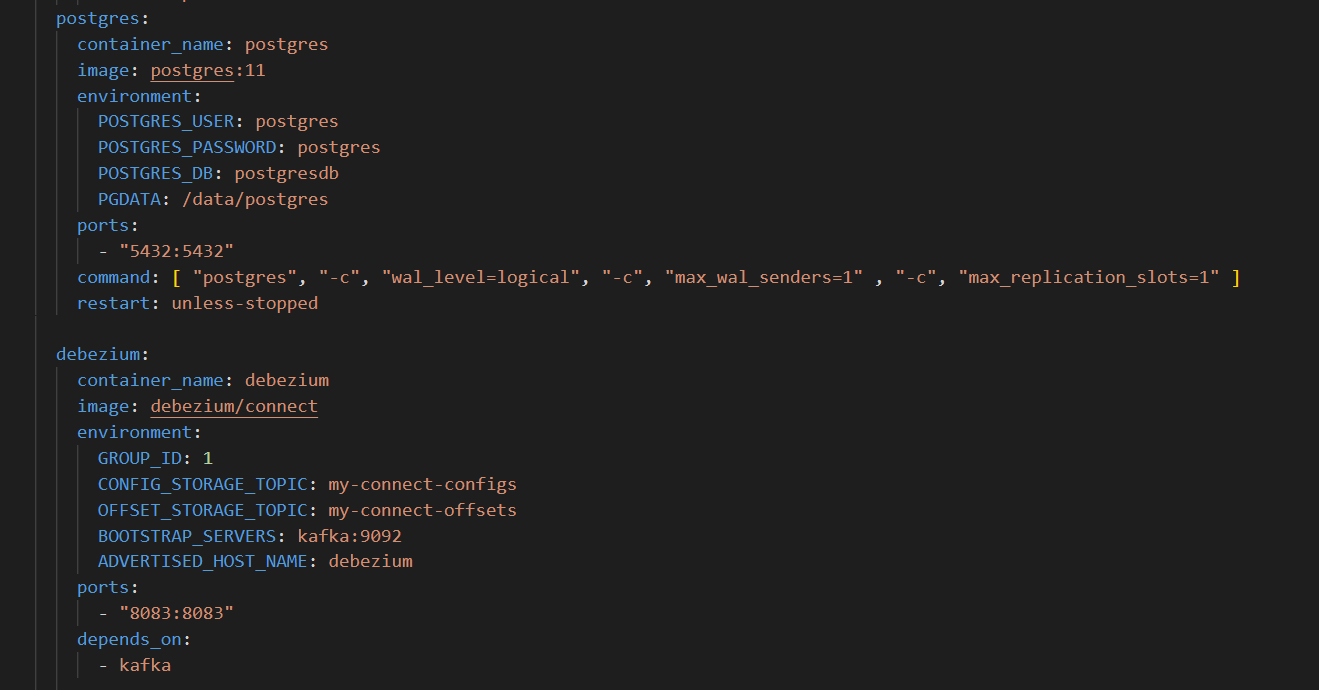


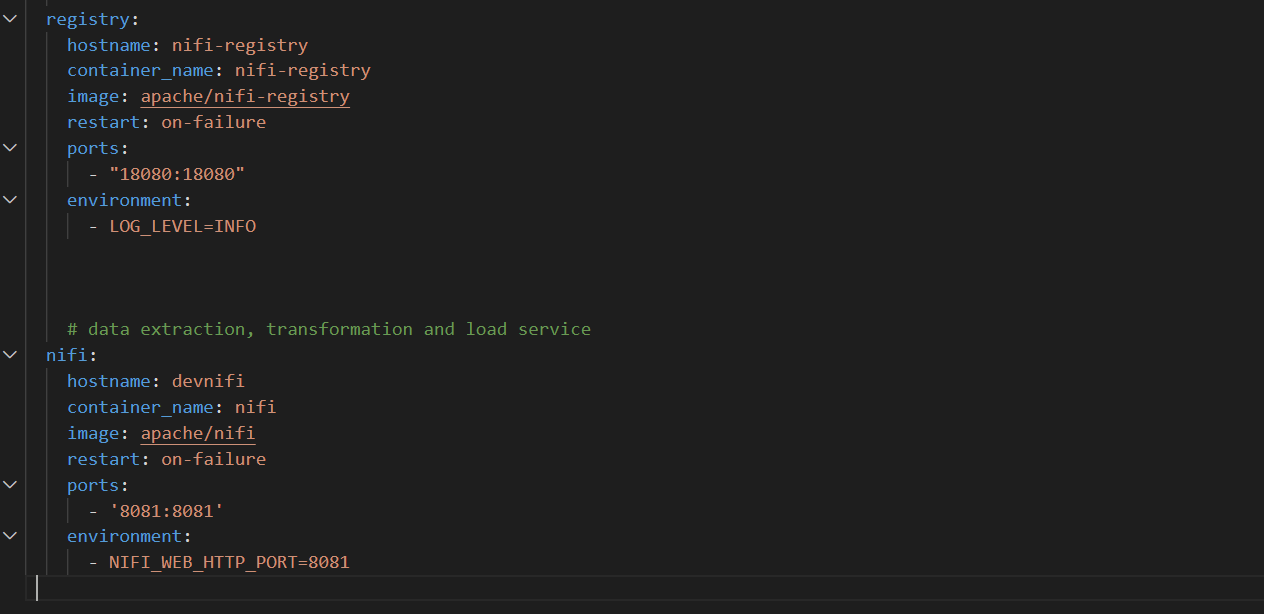


**Instalación de NIFI, Debezium , Postgres y Kafka con Dockers en Ubuntu en maquina virtual**

Instalaremos los productos mediante Docker compose y para esto tenemos un archivo Docker-compose.yml

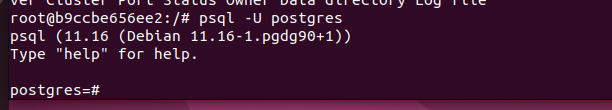




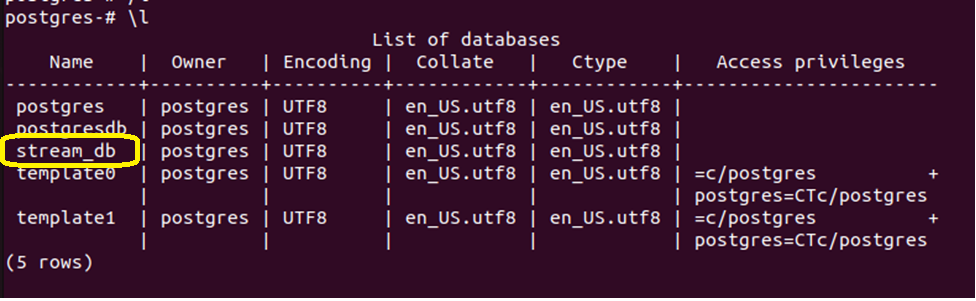


Ejecutamos Docker compose -up y tendremos ejecutanto Kafka, Debezium, Postgres y Nifi

Vamos a crear la tabla stock sobre Postgres



Listamos la base de datos

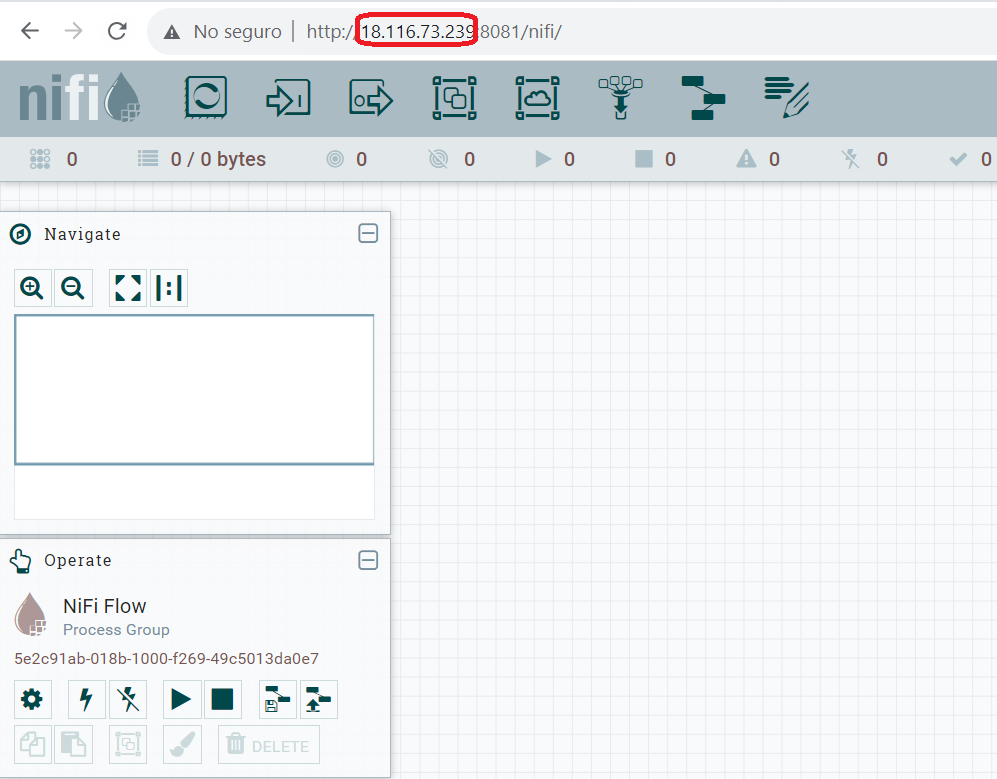


Creamos la tabla stock



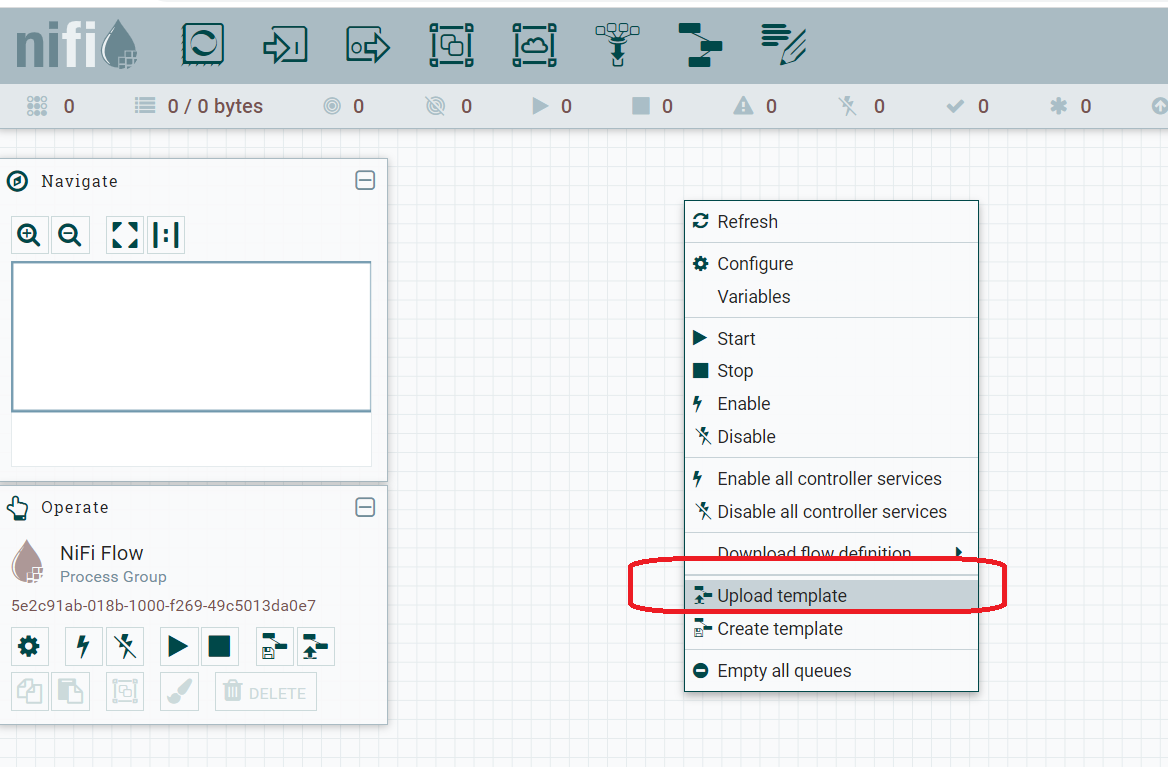
**Vamos a probar Apache NIFI.**

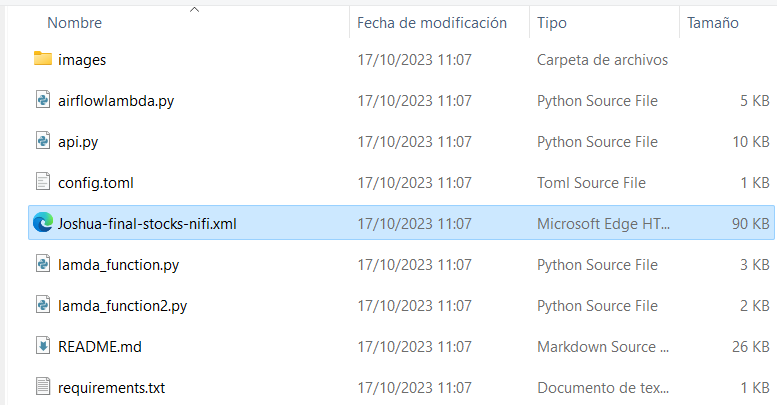
Apache NIFi se conecta con FASTAPI que esta instalado en una instancia EC2 con NGINX

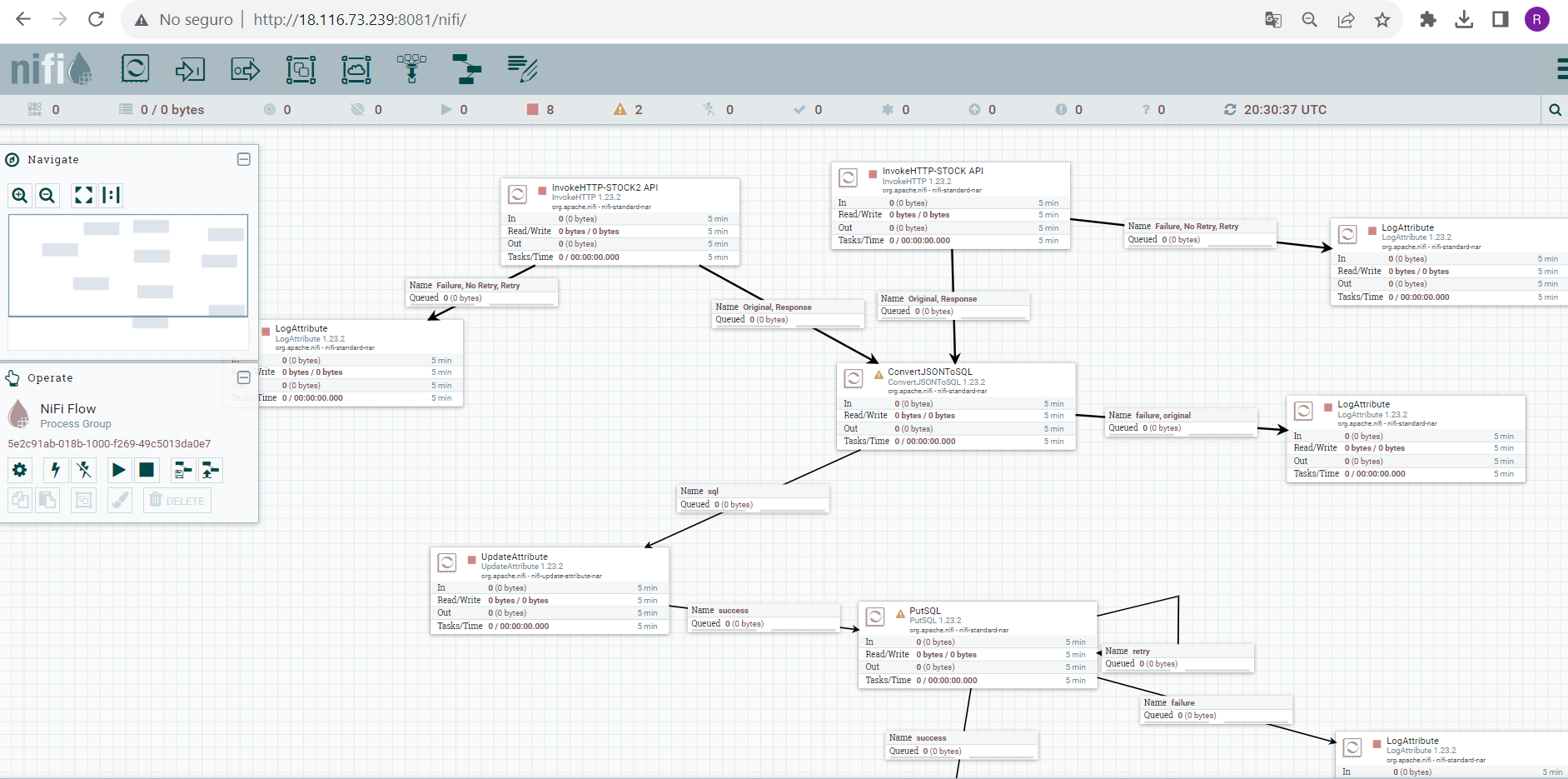




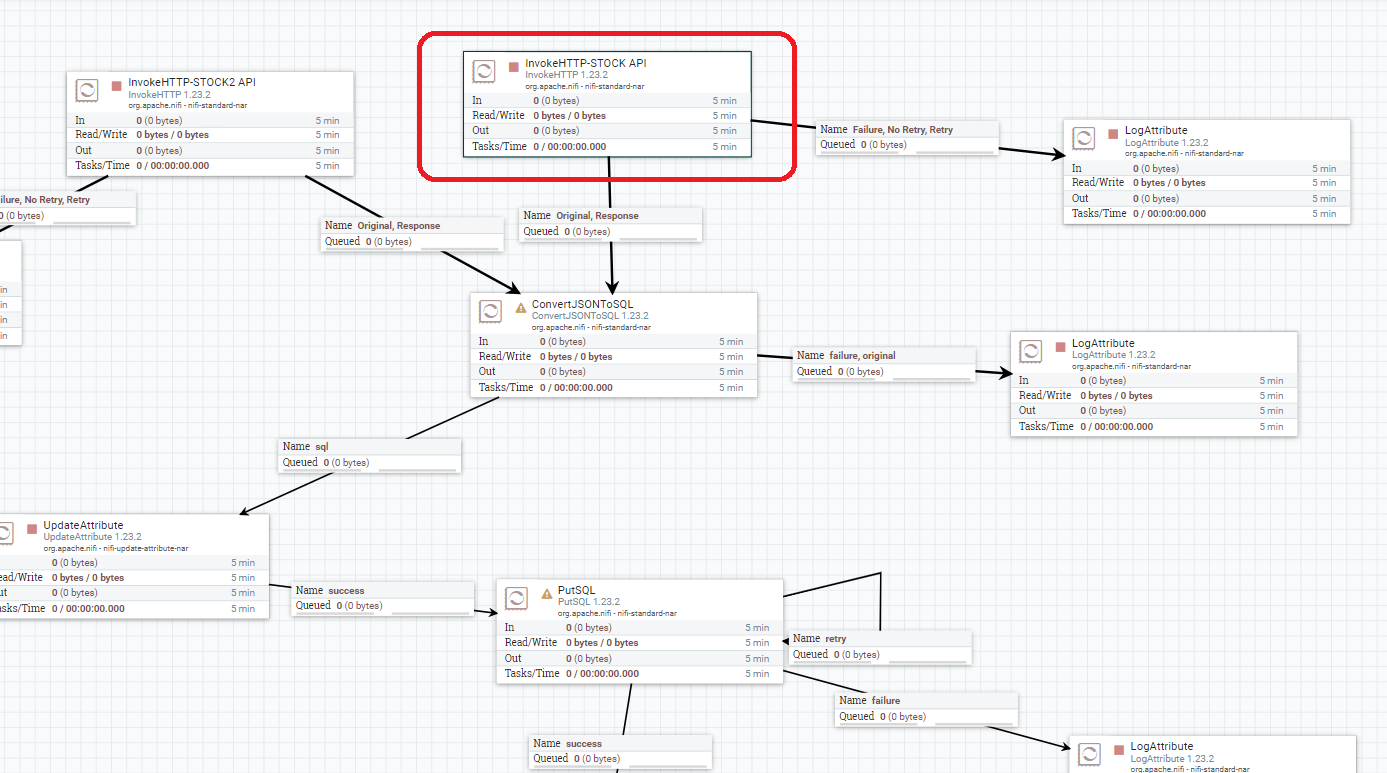
Cargamos el template

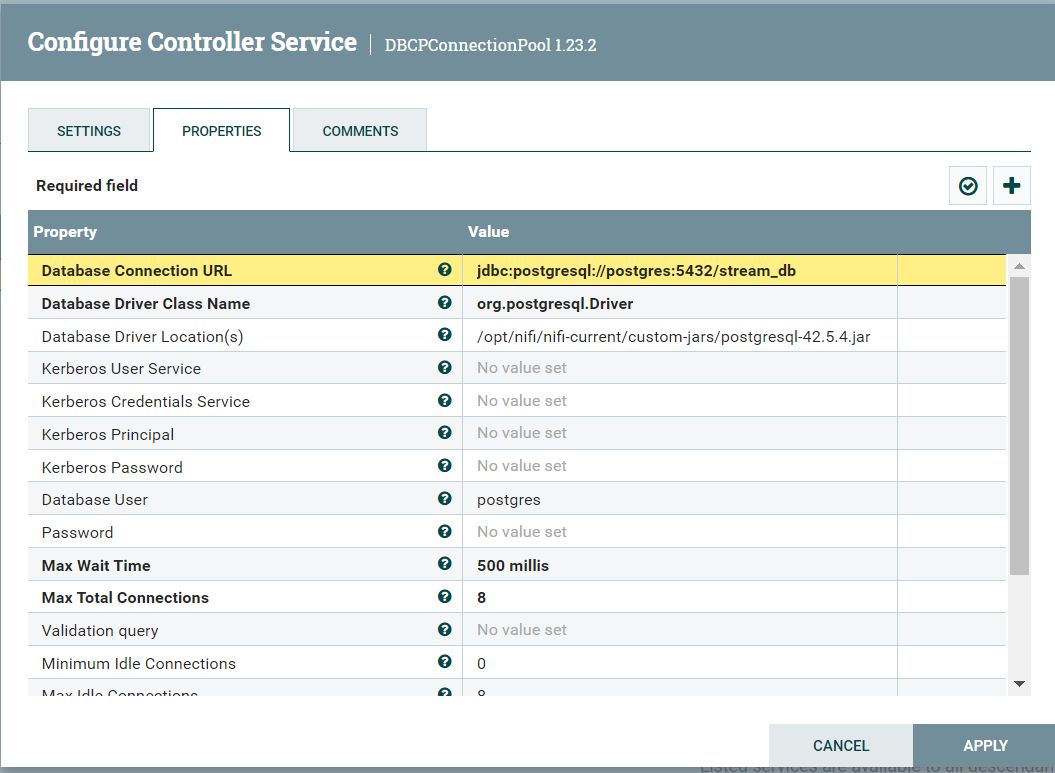


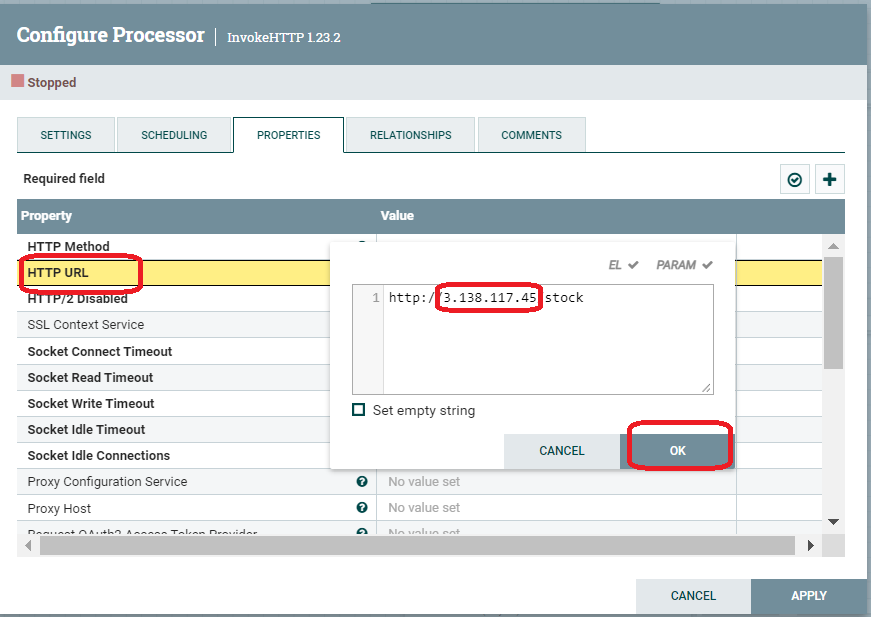




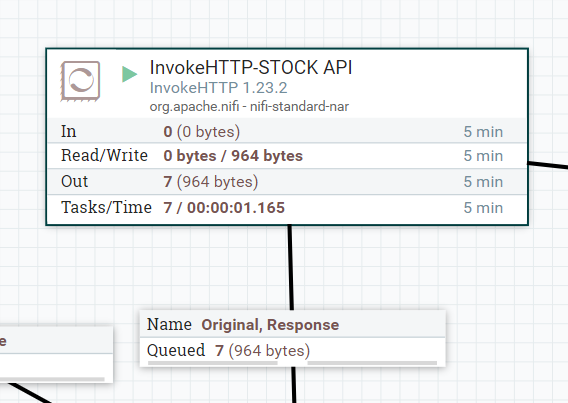
Cambiaremos la IP del proceso que obtiene la informacion desde el API de NGINX



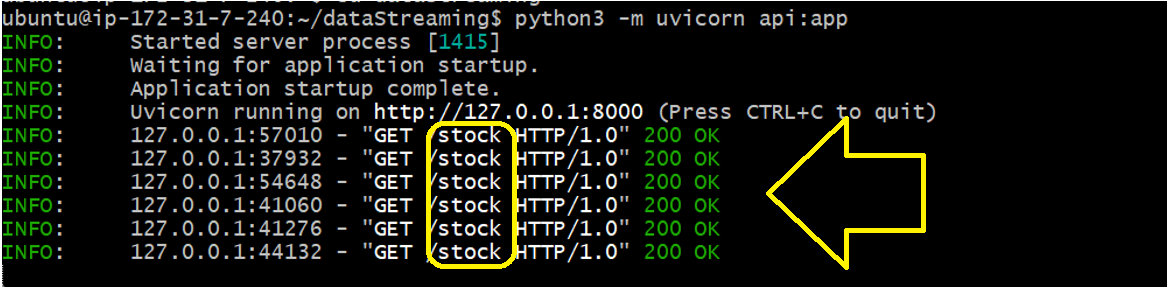




Ahora ejecutaremos el proceso InvokeHTTP-STOCK API. Este proceso llama al API Stock para obtener los datos de la bolsa de valores de las compañías Amazon y Apple. EL API obtiene los datos desde un bucket en S3



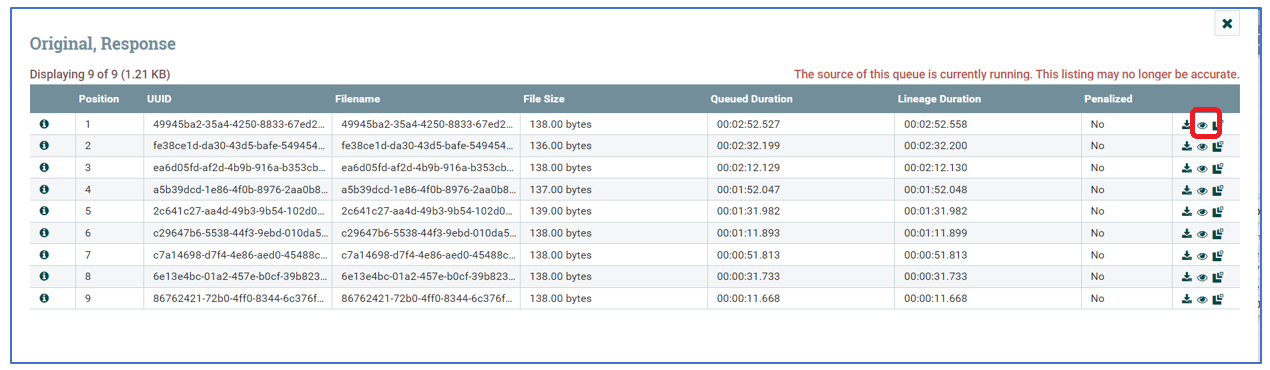
En la pantalla siguiente vemos como por cada acceso al API en al consola en NGINX se adiciona una ejecución



Ahora veremos la cola de registros en NIFI



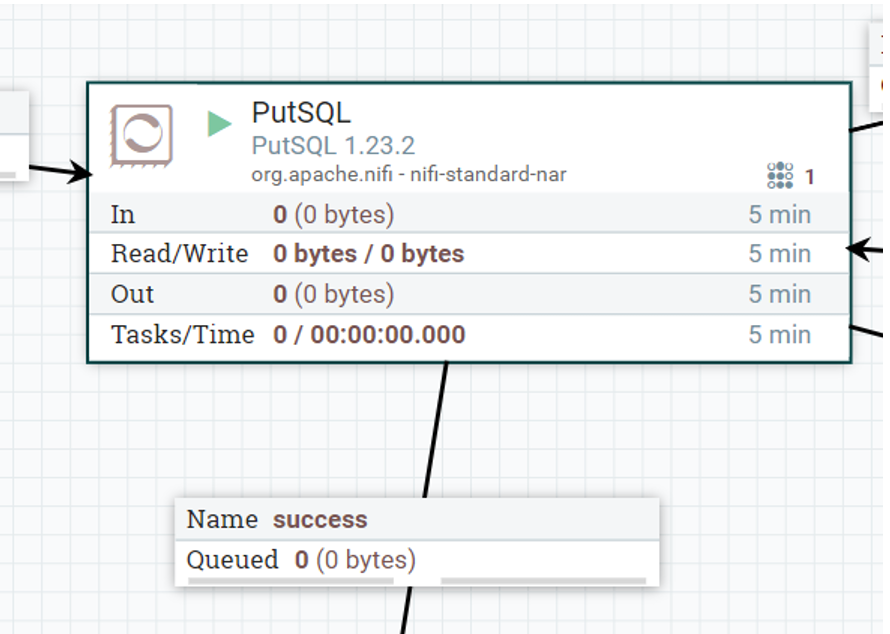
Veremos los registros obtenidos desde el API



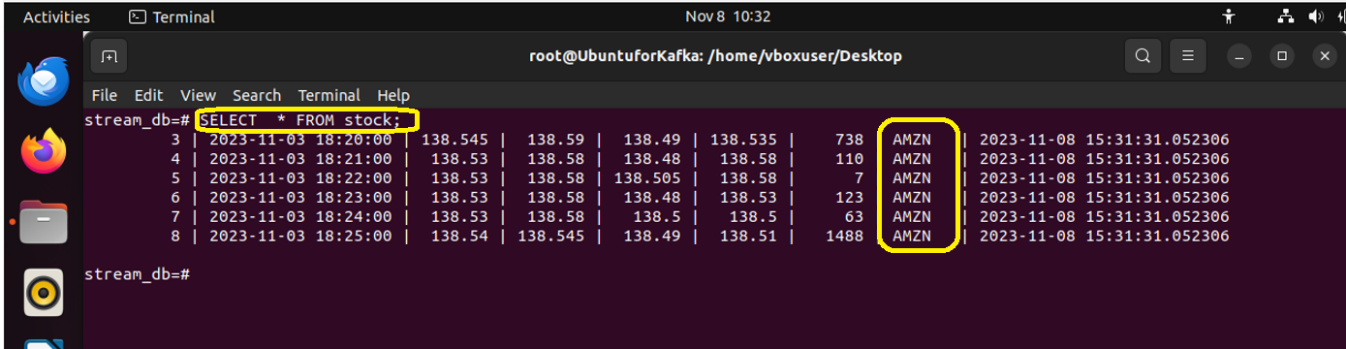
Ahora veremos el detalle de cada registro. Son los datos que se insertara en la base de datos Postgres.



El proceso PUTSQL insertara los datos a la tabla stock de la base de datos stream\_db que están alojados en Postgres



Verificamos si se insertaron los datos en la base de datos



**Configuramos el conector Debezium con Postgres**

curl -X POST -H "Accept:application/json" -H "Content-Type:application/json" localhost:8083/connectors/ -d '

{

"name": "test-connector",

"config": {

"connector.class": "io.debezium.connector.postgresql.PostgresConnector",

"database.hostname": "postgres",

"database.port": "5432",

"database.user": "postgres",

"database.password": "postgres",

"database.dbname" : "stream\_db",

"database.server.name": "dbserver1",

"topic.prefix": "dbserver1",

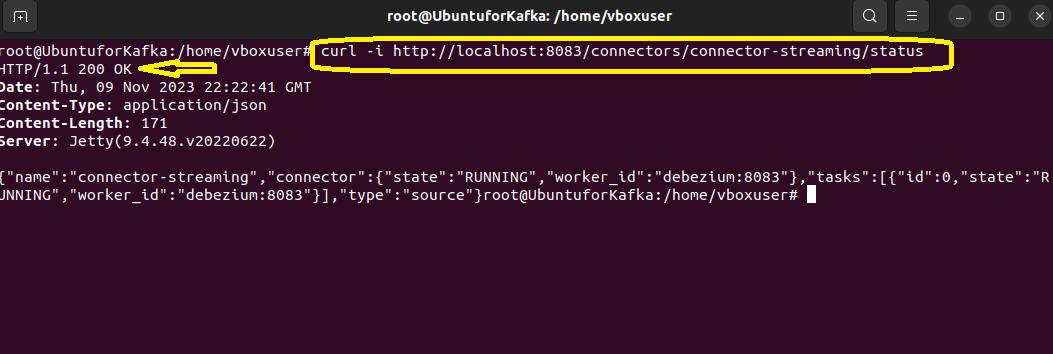
"plugin.name": "pgoutput"

}

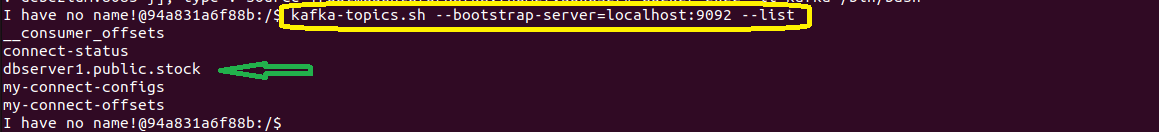
}'



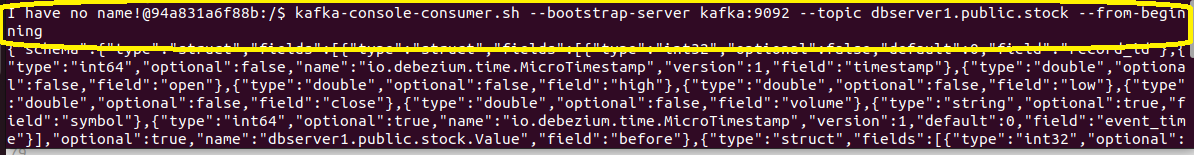
Verificamos que el conector este activo



Entramos al conteiner kafka y listamos los topics creados. Se creo automáticamente el topic que señala la flecha verde

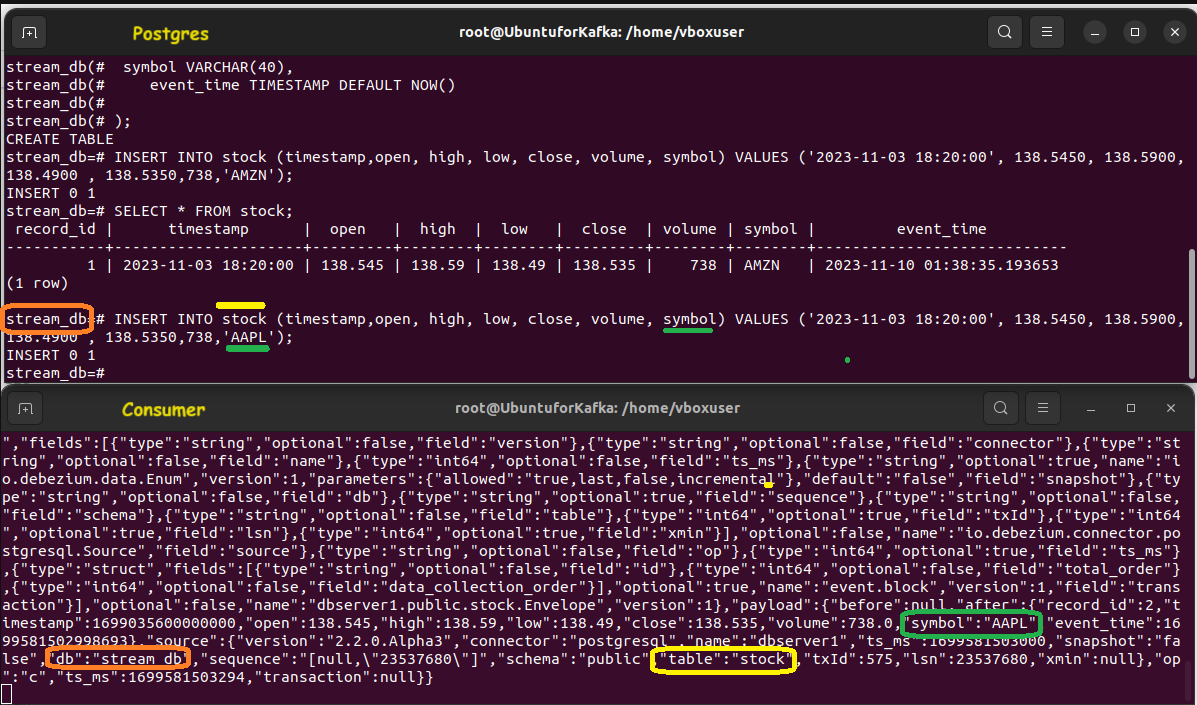


Ahora arrancaremos el consumer de Kafka con el topic creado



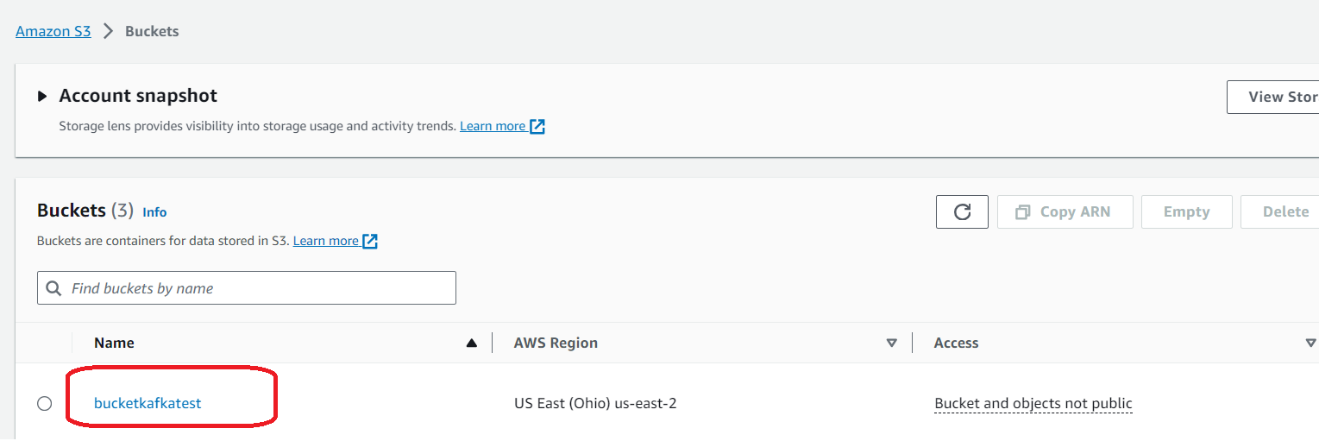
Ahora hacemos una prueba del conector.

Haremos un insert en la tabla STOCK de la base de datos STREAM\_DB y automáticamente se visualizara en la pantalla del consumer de Kafka la modifcacion de la tabla.



Enviamos los archivos json generados por el conector de Debezium a un bucket de S3 mediante un programa en Python

Creamos un bucket en S3 de AWS

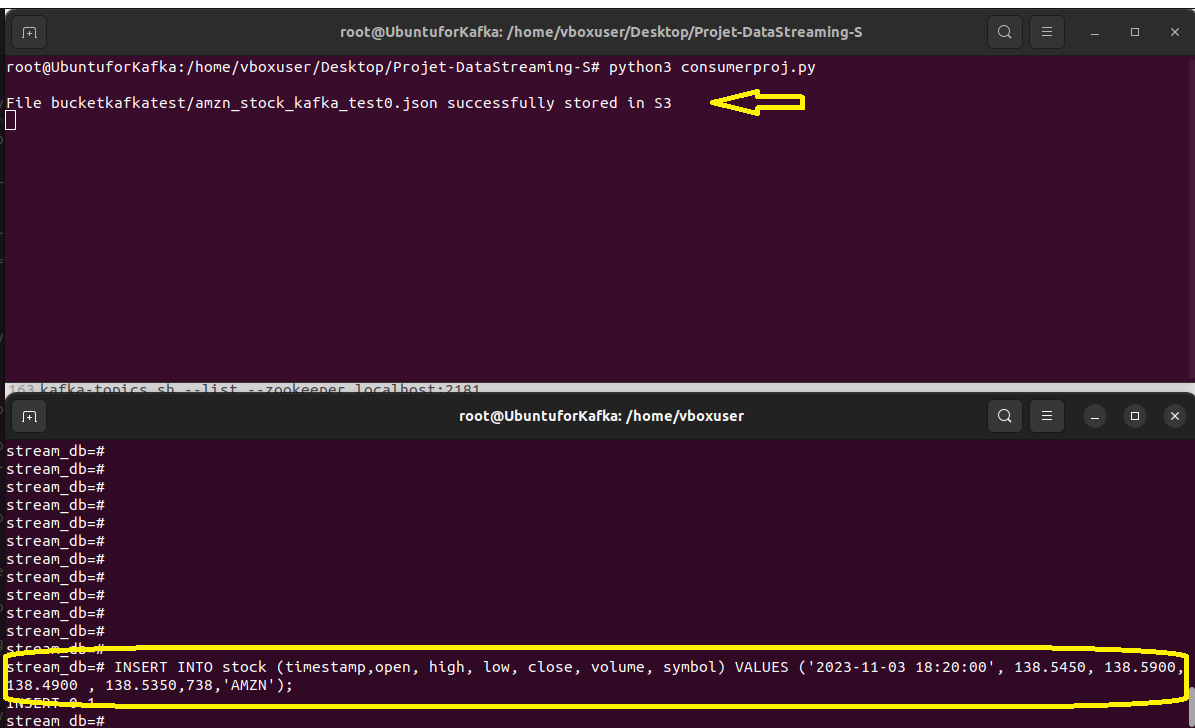


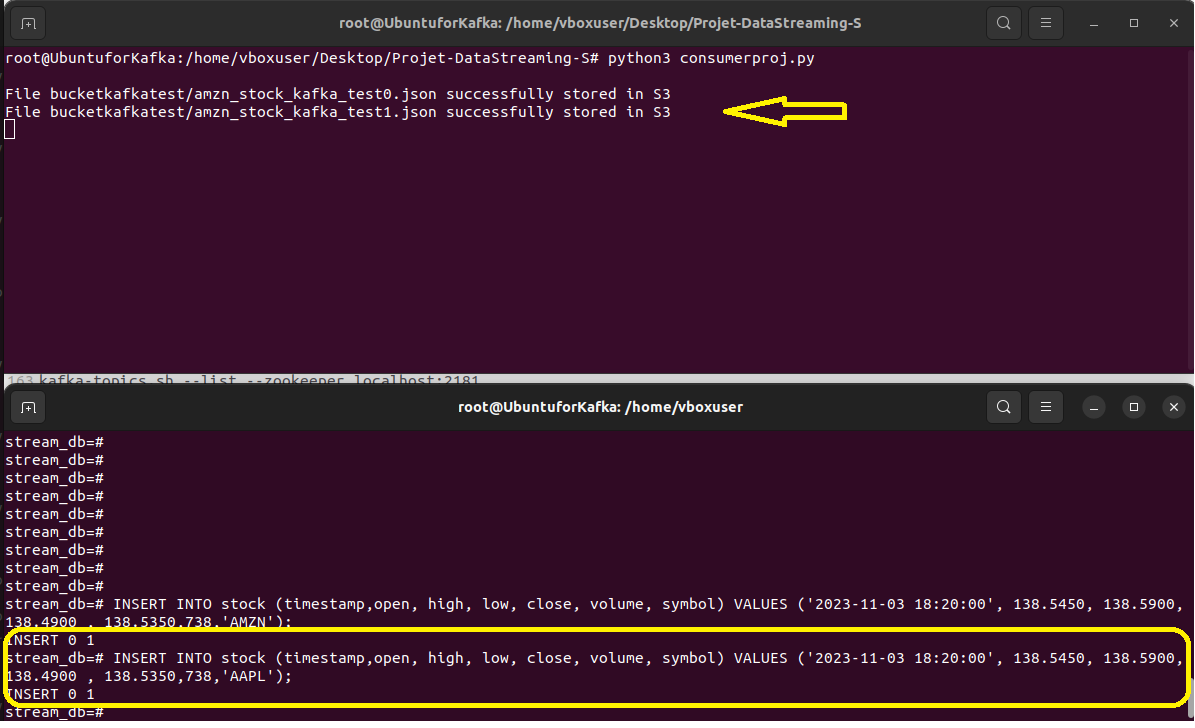
Luego ejecutamos un programa Python que hemos creado que se llama consumerproj.py

En la siguiente pantalla insertaremos un registro en la tabla stock y el programa de Python lo enviara al bucket S3



Presionamos enter en el editor sql de postgres

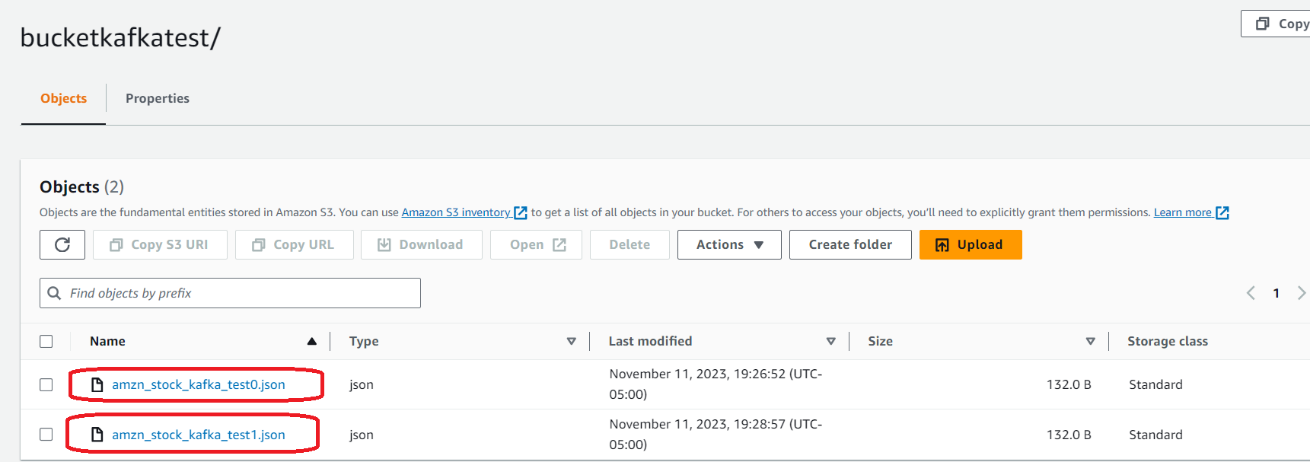




El programa nos dice que envio los archivos json a S3

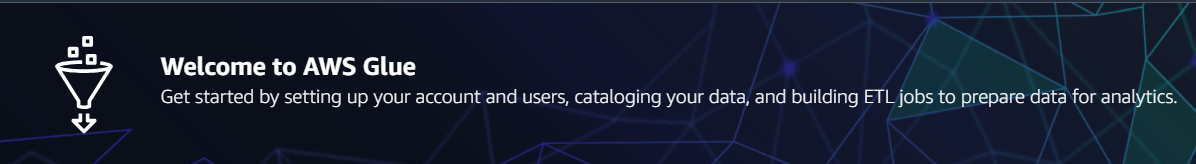
Vamos al bucket y verificamos si llegaron los archivos.

Aquí tenemos los archivos creados

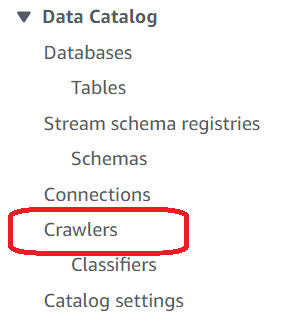


Ahora vamos a trabajar con la información del bucket

Nos vamos a AWS Glue

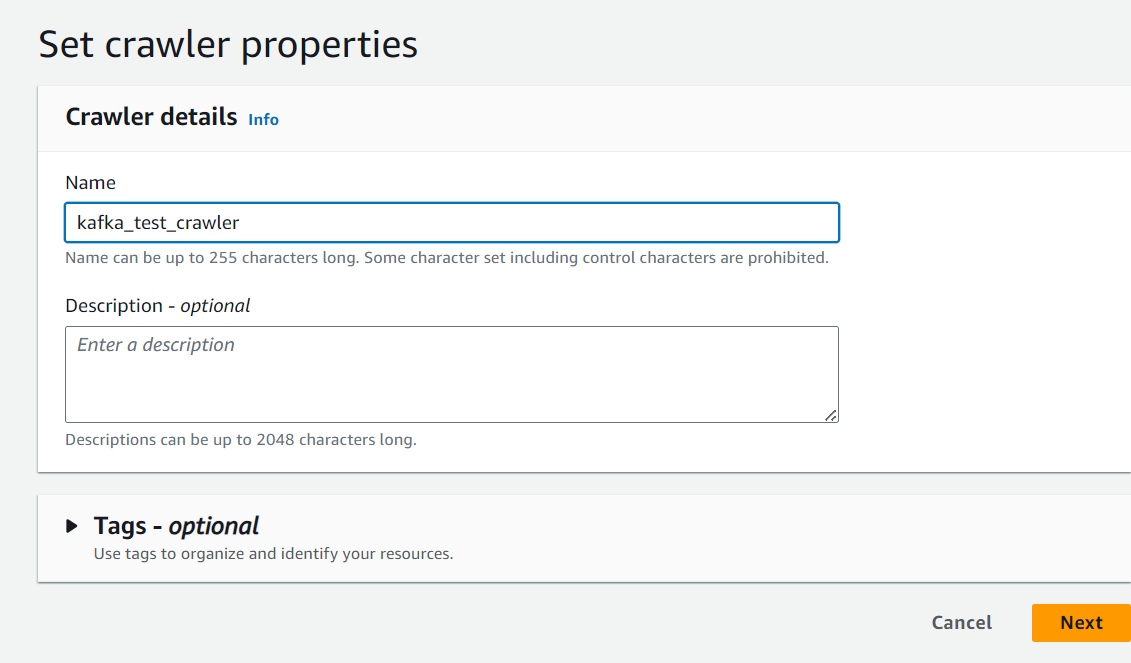


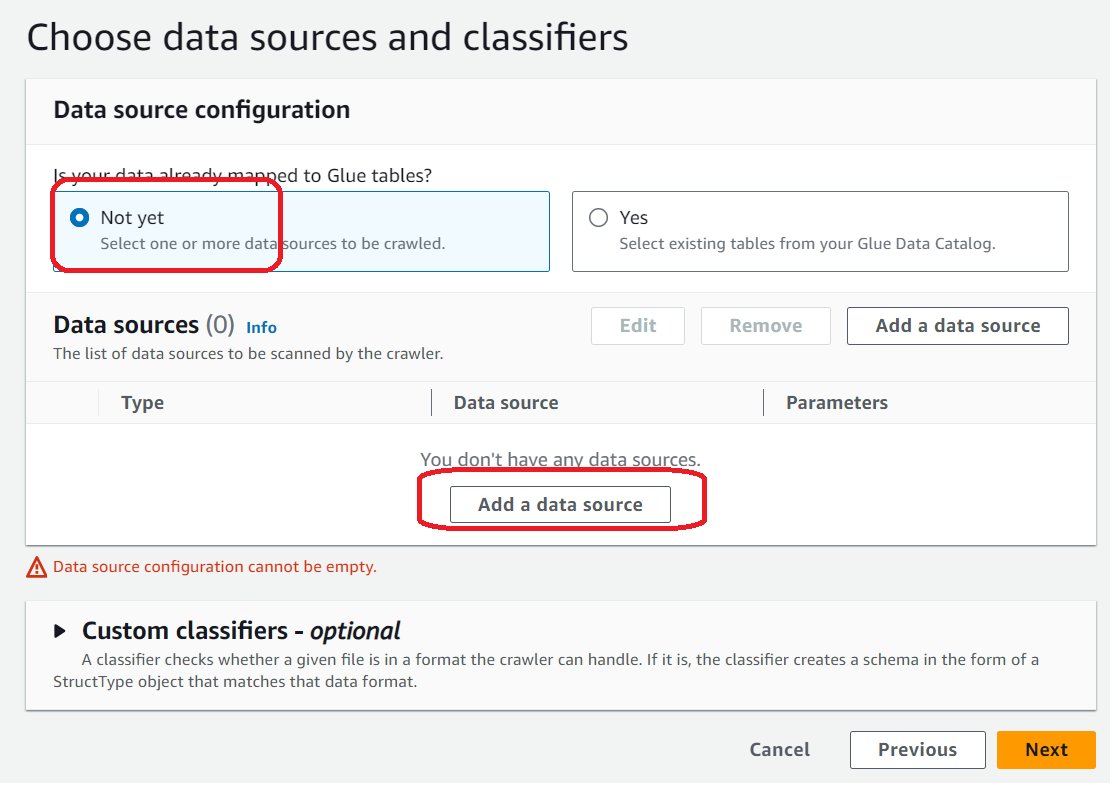
Y en el menú de la izquierda elegimos

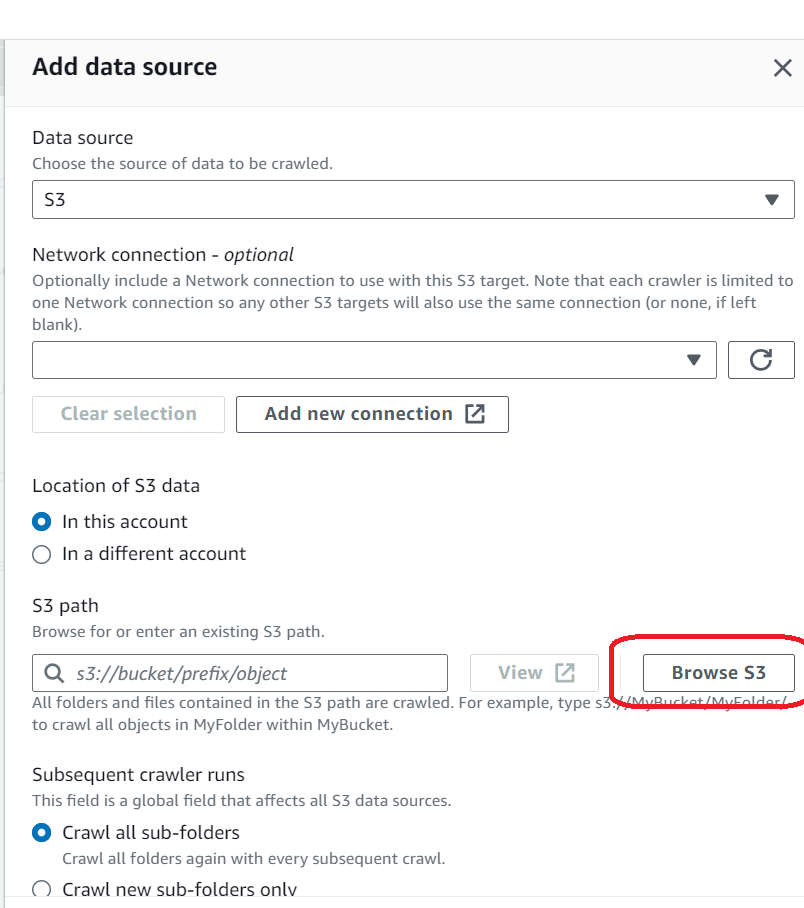


Create Crawlers

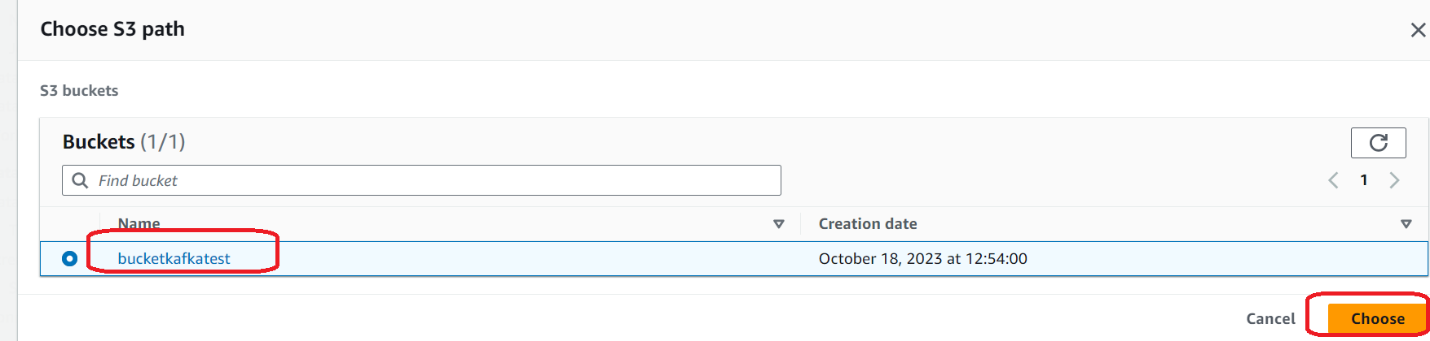


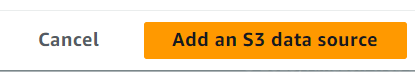


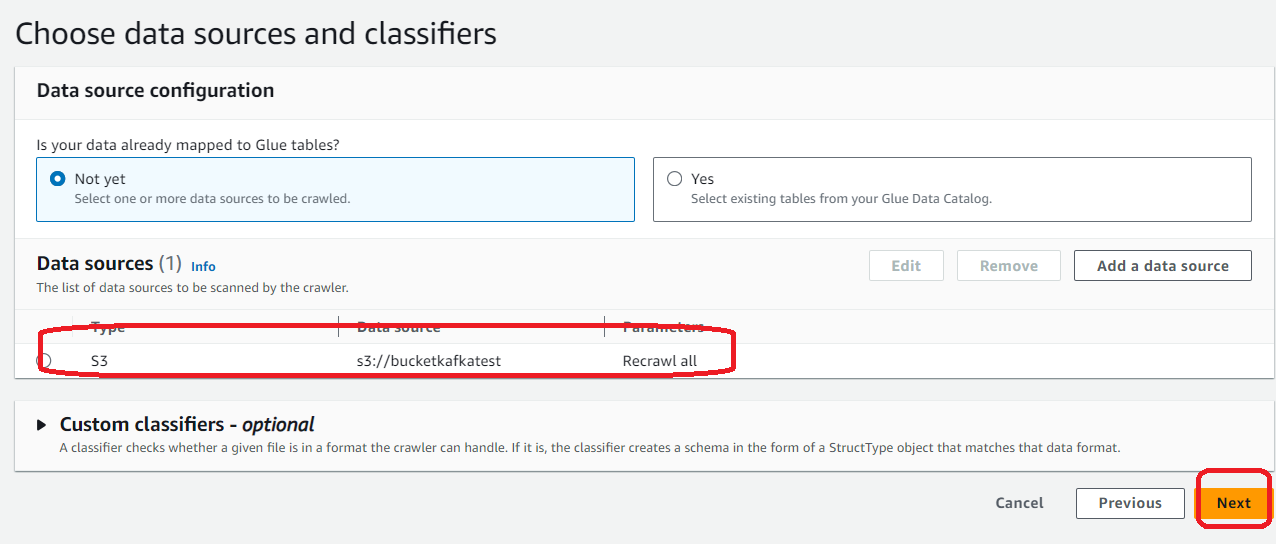




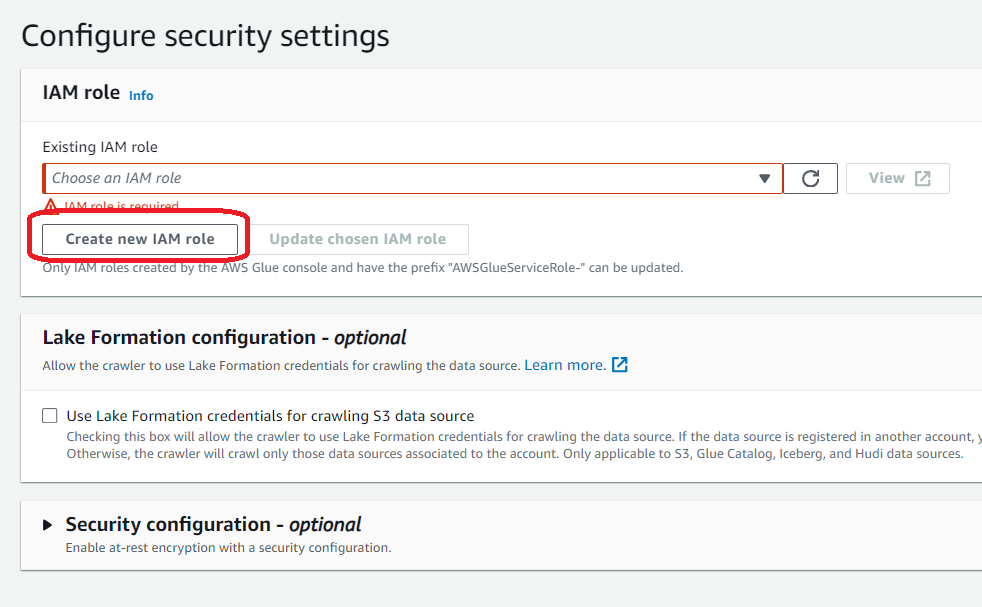
Elegir el bucket

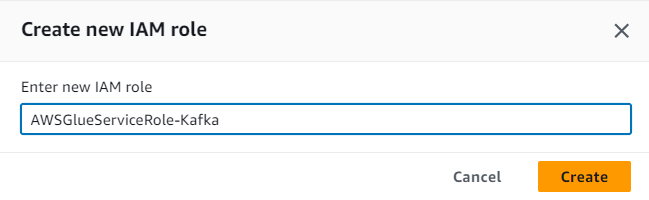


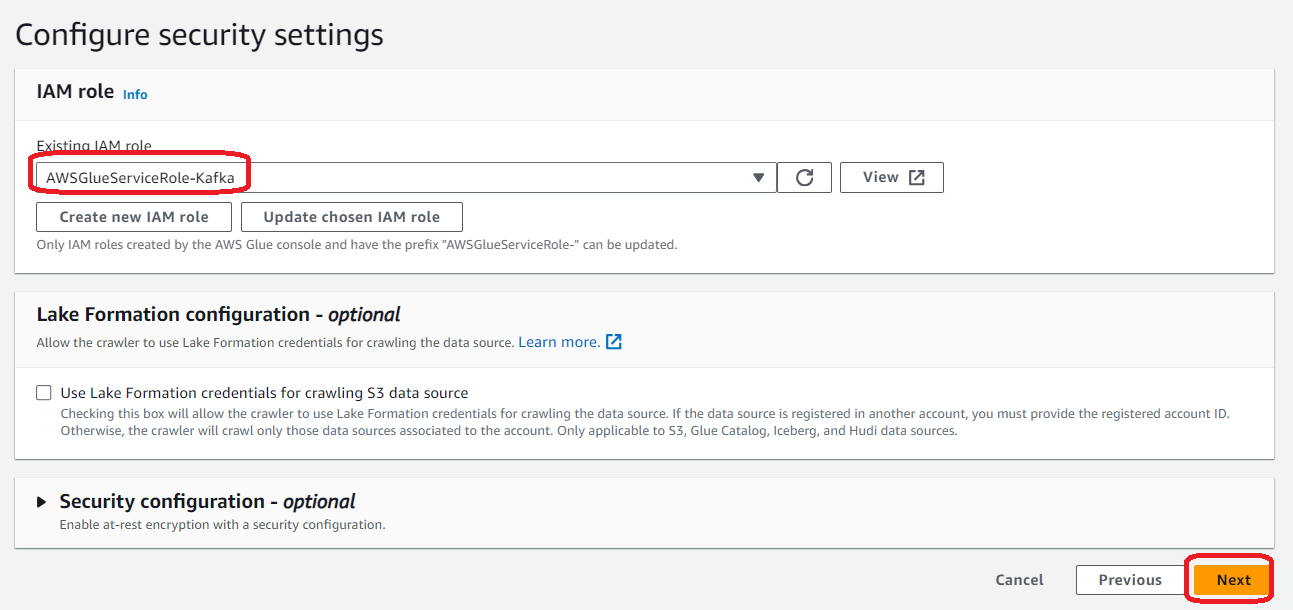




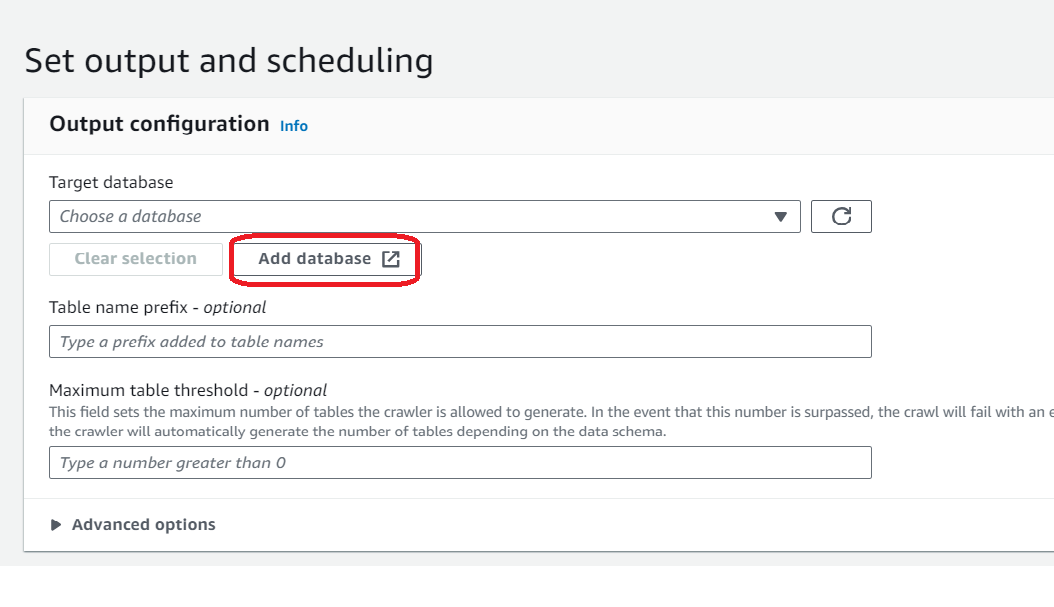
Crear un nuevo rol



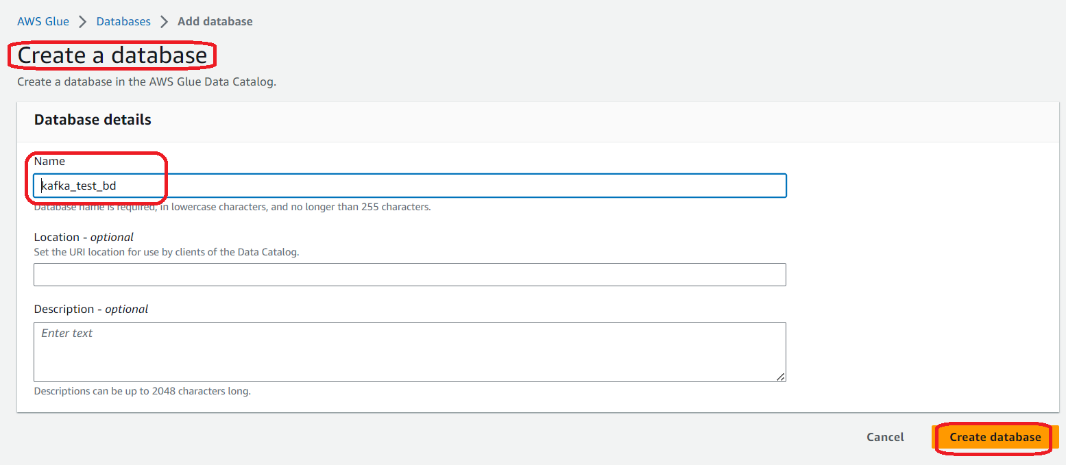


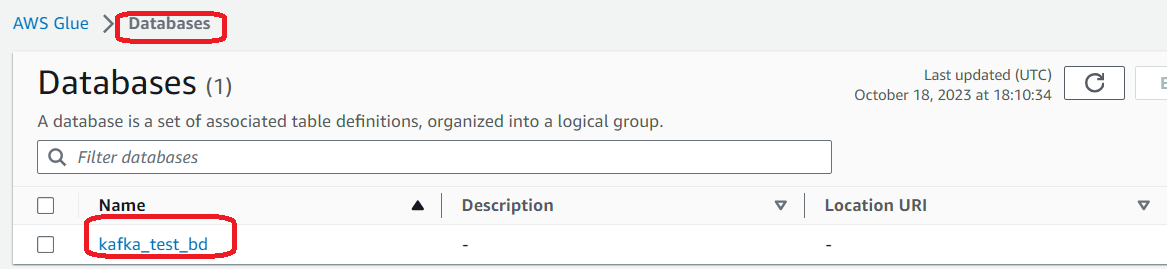


Crear una base de datos

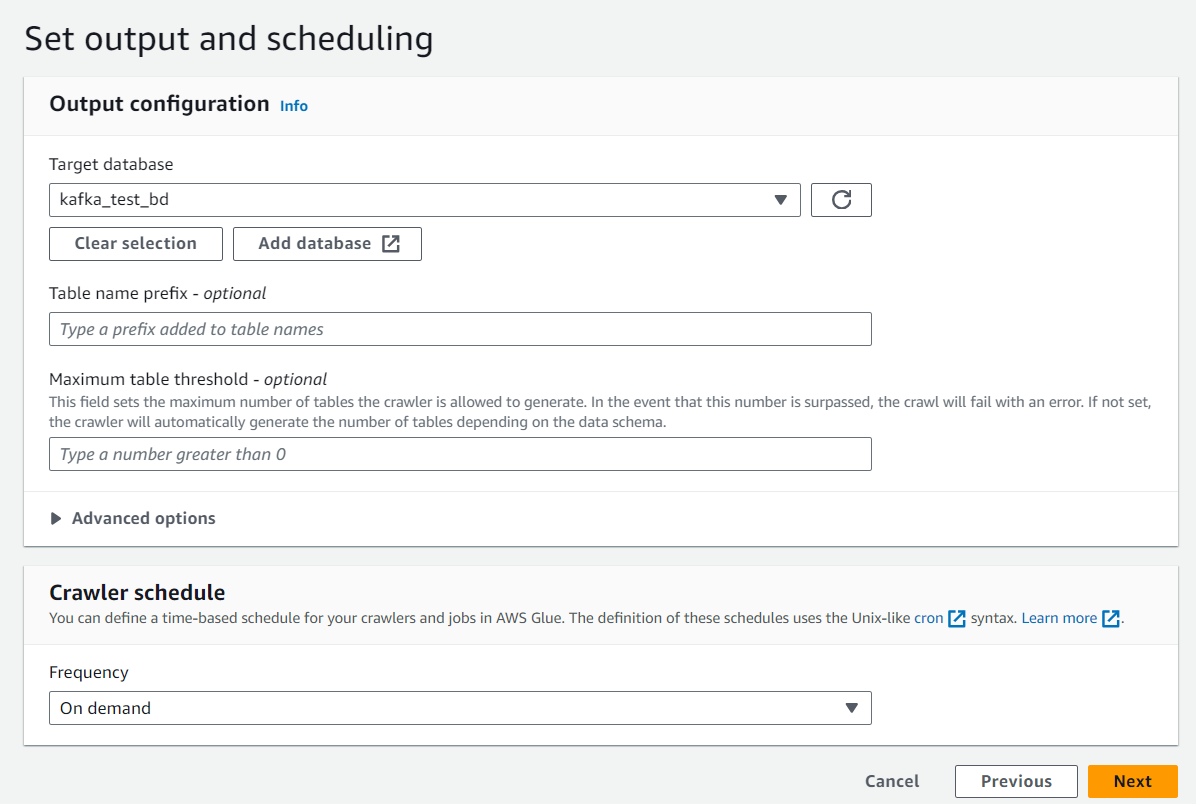


Ingresar el nombre y crear la base de datos

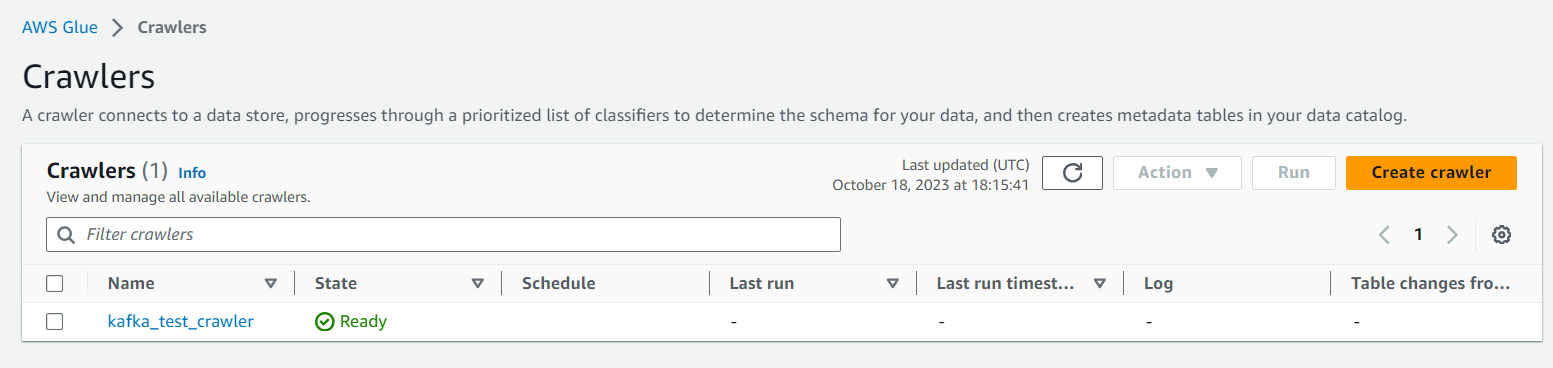




Volvemos a crawlers

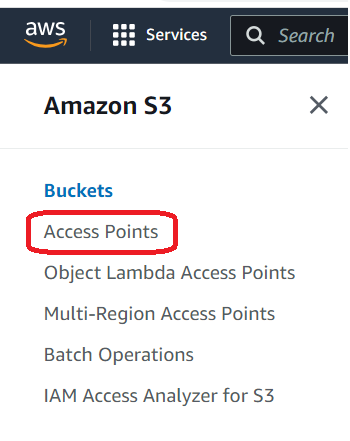


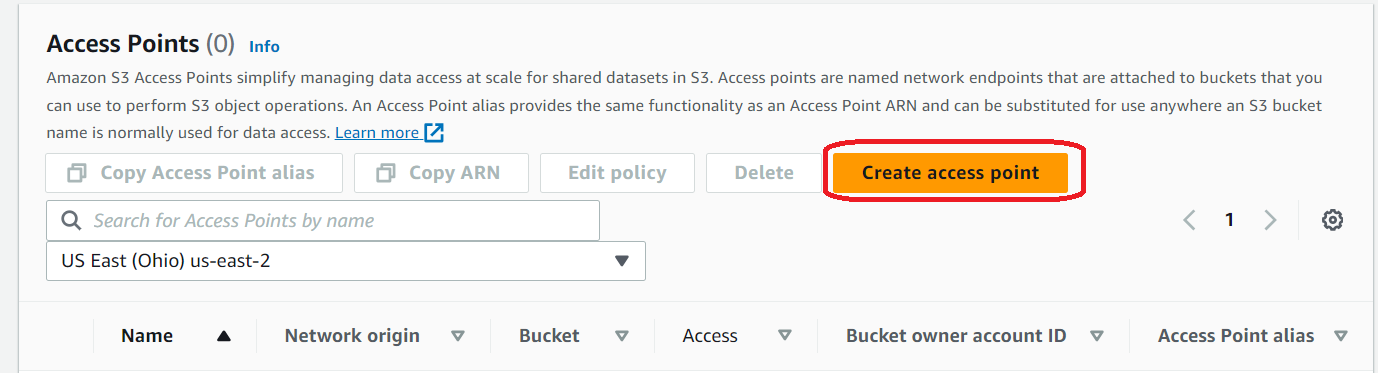


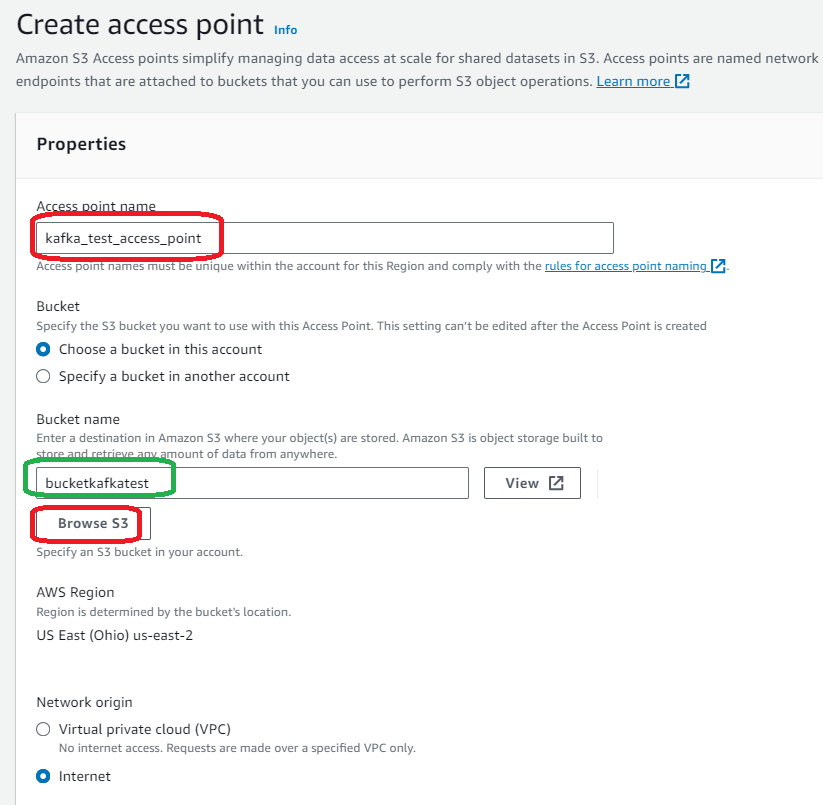


Creamos un punto de acceso

Nos vamos a S3 y elegimos en el menu de la izquierda la opción Access Points

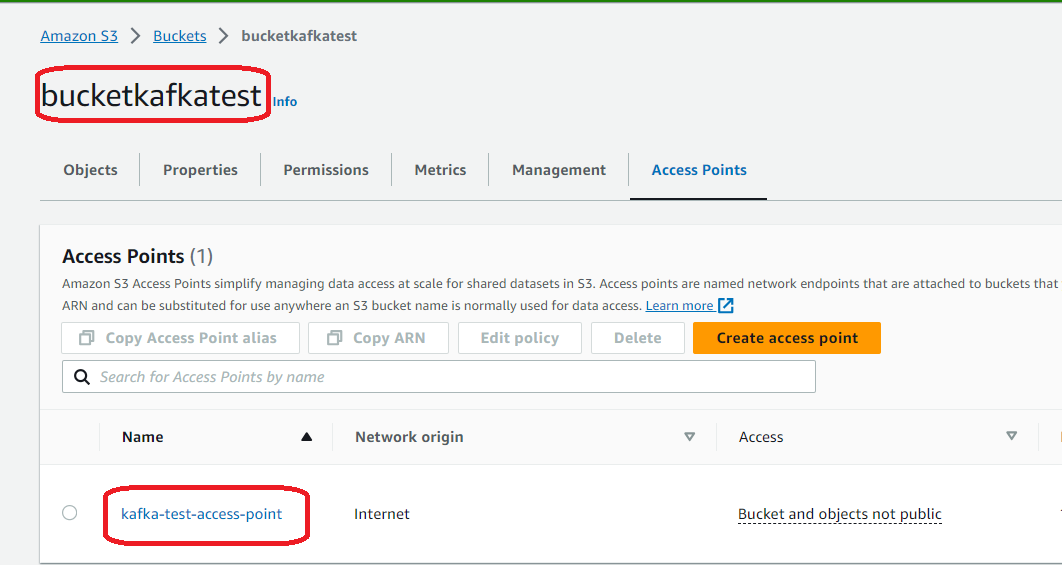








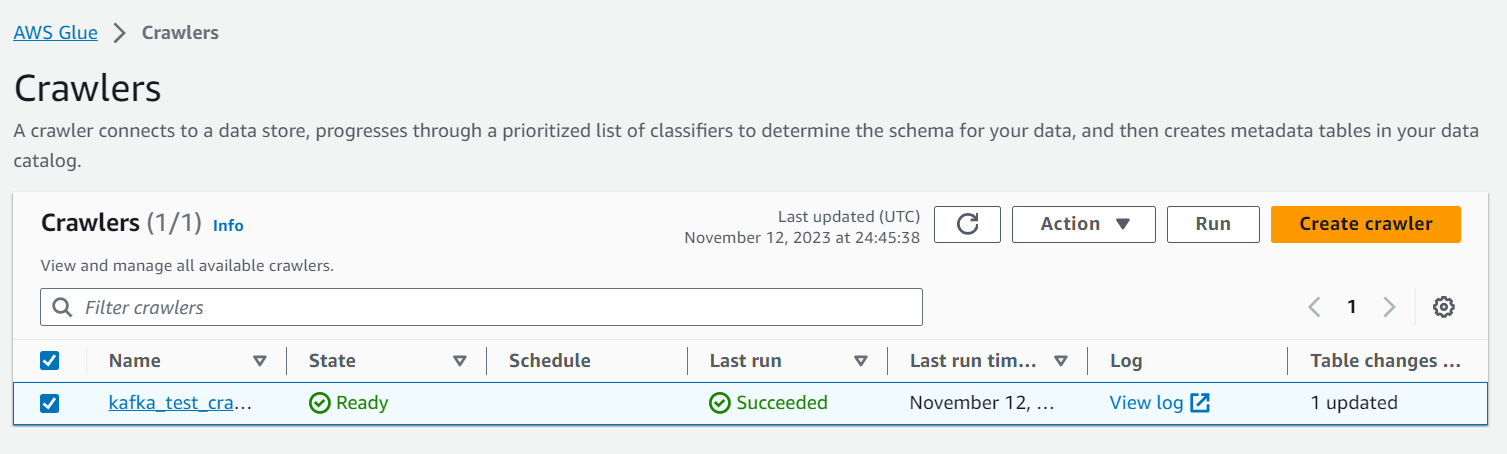
El bucket ya esta relacionado con el Access point creado.



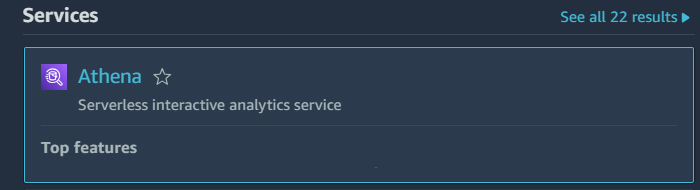
Luego ejecutamos el crawlers

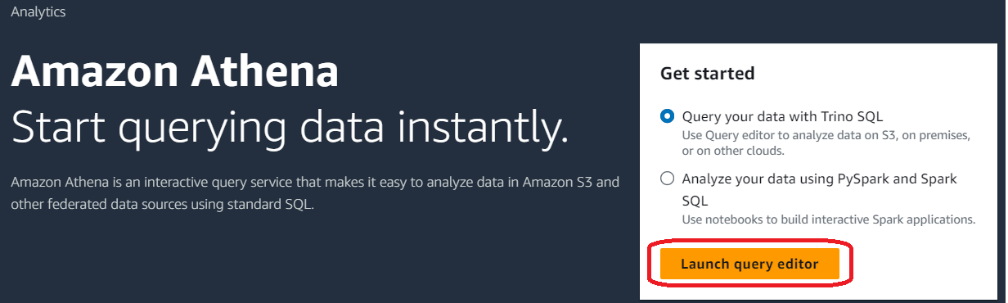


Nos indica que se creo una tabla

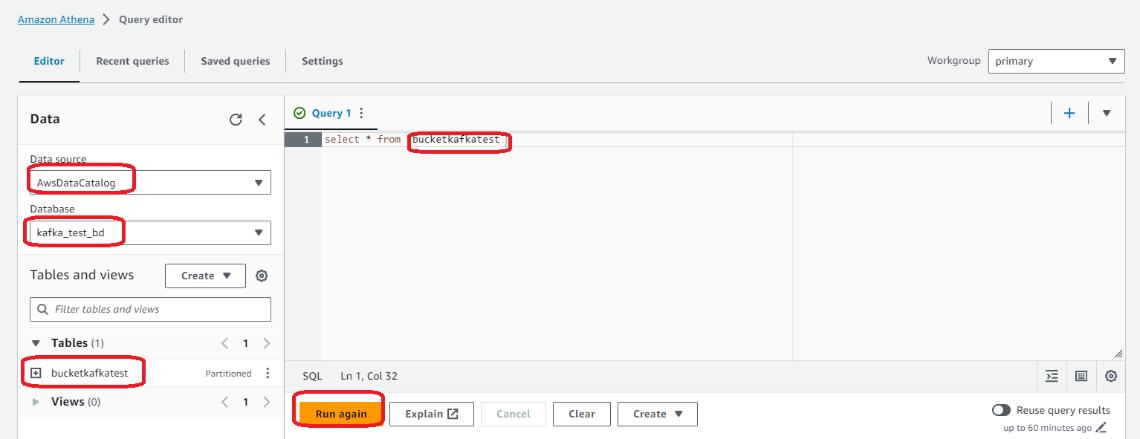


Ahora abriremos Athena

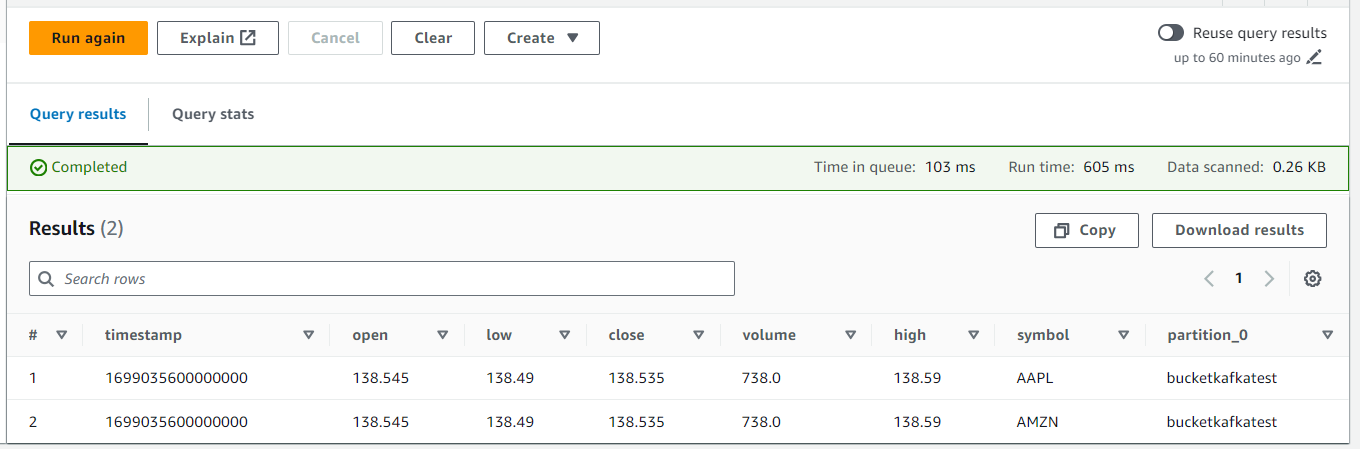




Se ingresa un nuevo query y se ejecuta



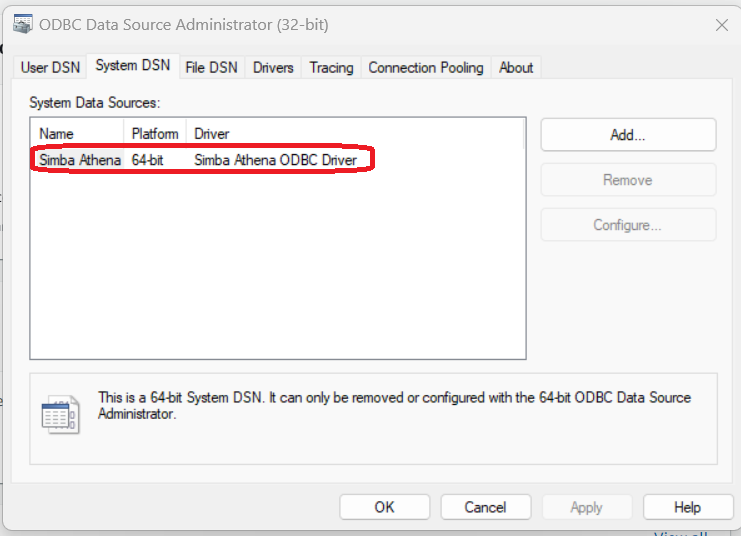
El resultado de query



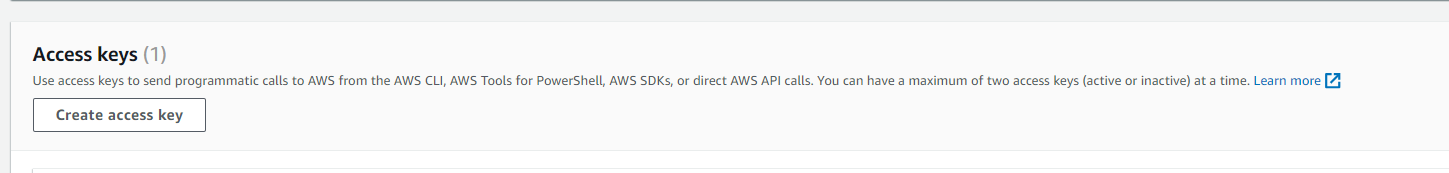
Conexión con Power BI

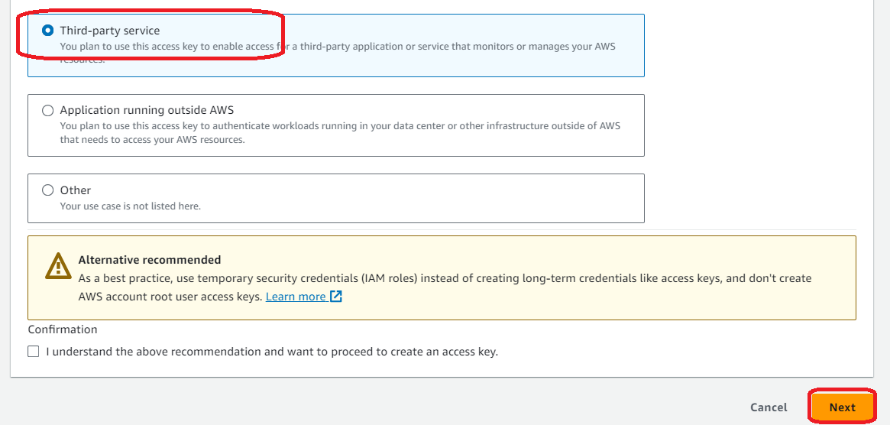
Descarga e Instalar ODBC driver para Athena

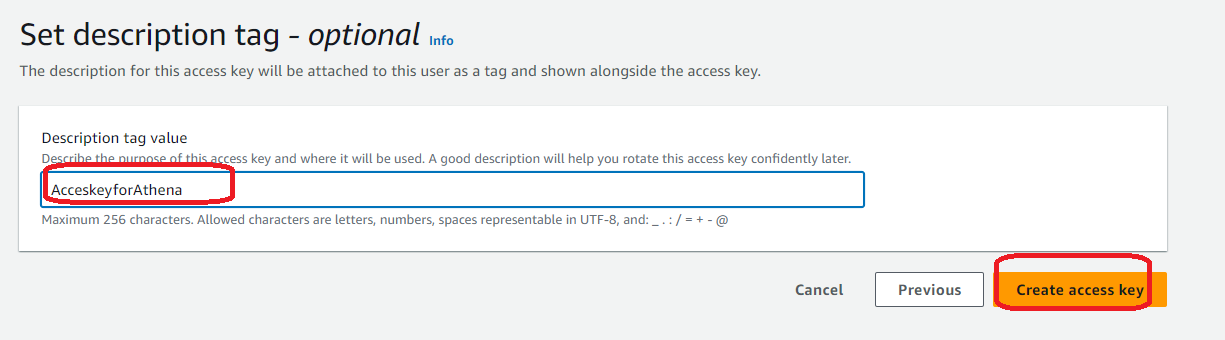




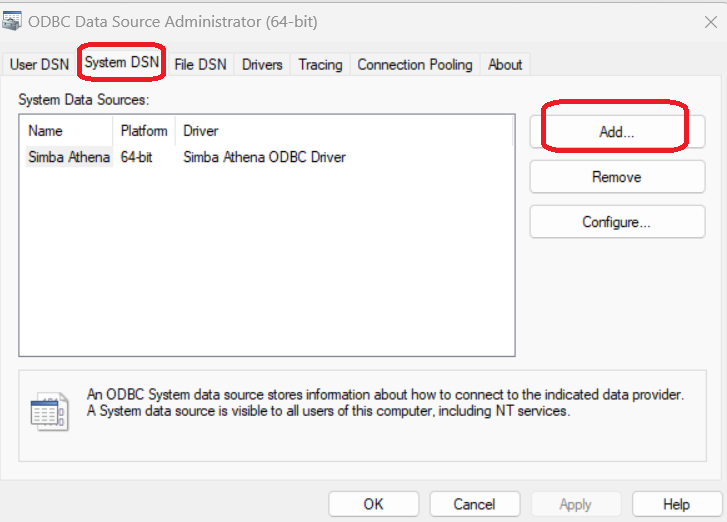
Luego crear un Access key en IAM



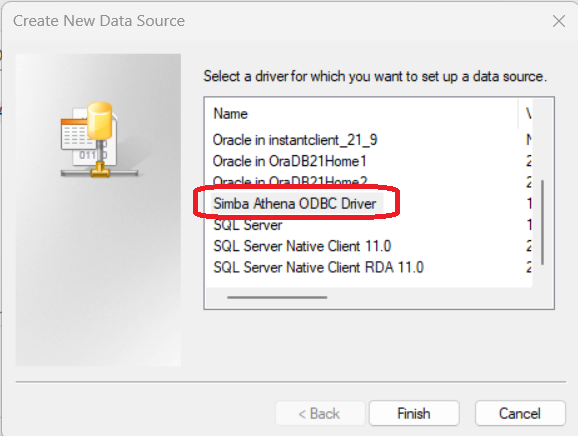




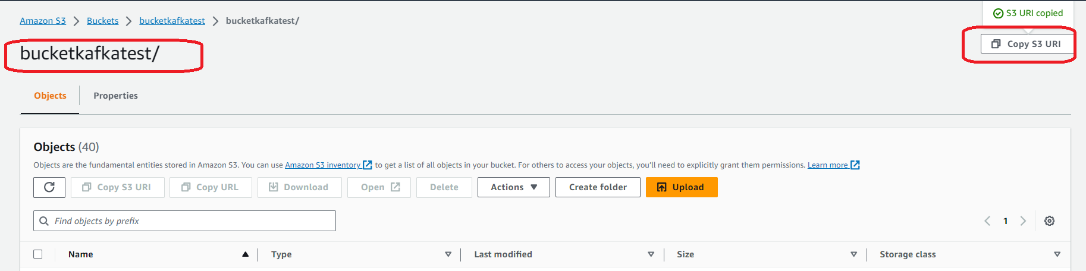
Luego creamos un nuevo DNS en ODBC Data Source Adminitrator



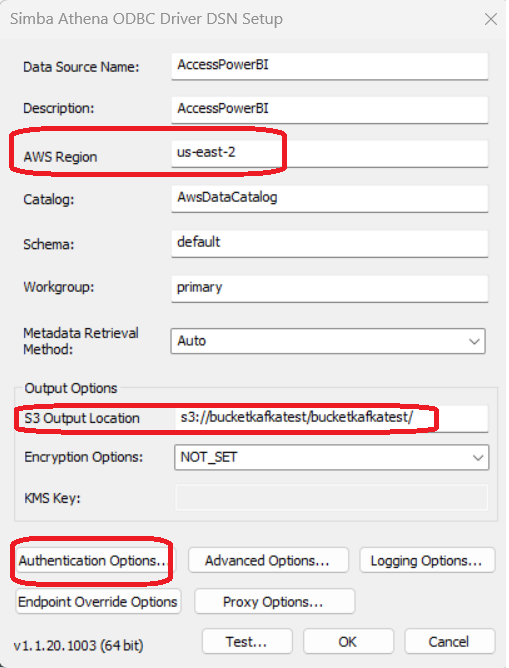
Seleccionamos Athena ODBC Driver



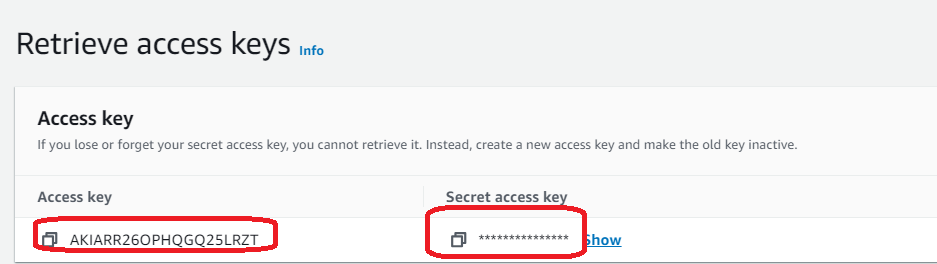
Se copia la información S3 Url para luego agregarlo en la configuración ODBC

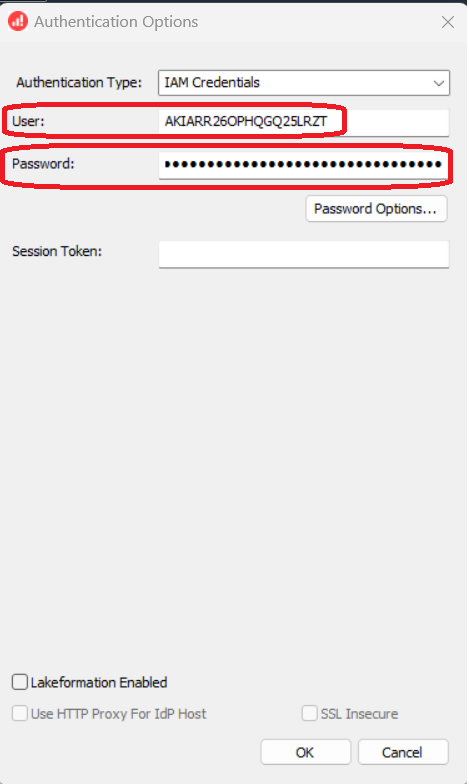


Pegas el S3 Url copiado y luego vas a Authentication Options

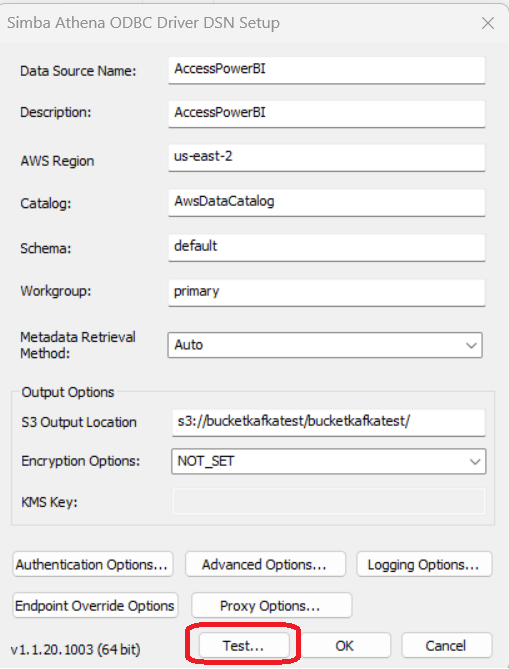


Vamos a IAM y copiamos el Access Key

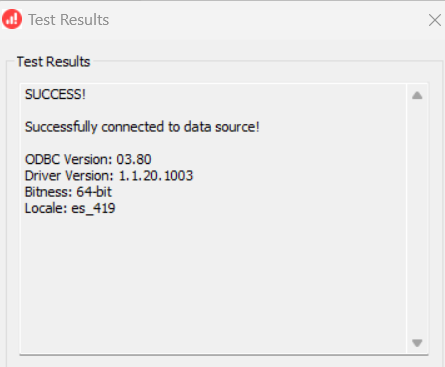




Luego realizas el test de conexión

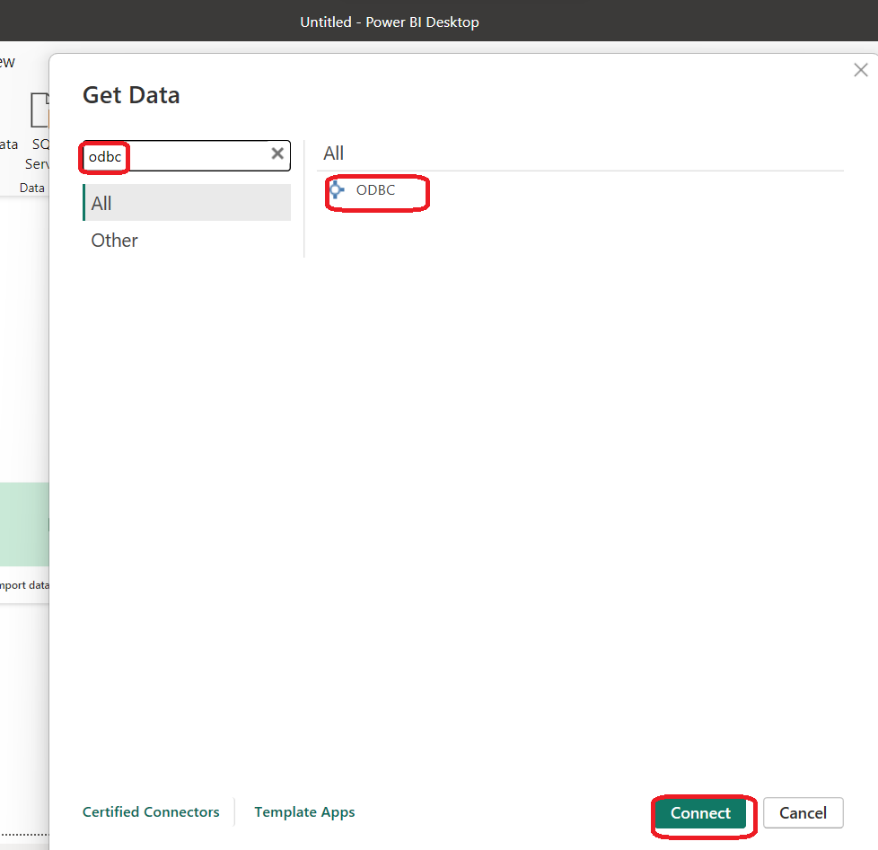


Test exitoso

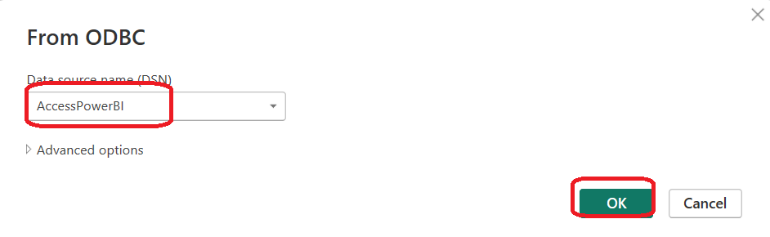


Abrimos Power BI

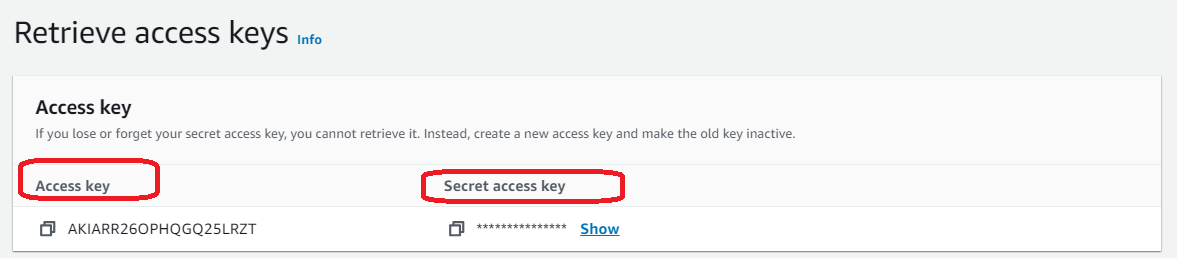
Para la conexión de datos elegimos la opción ODBC

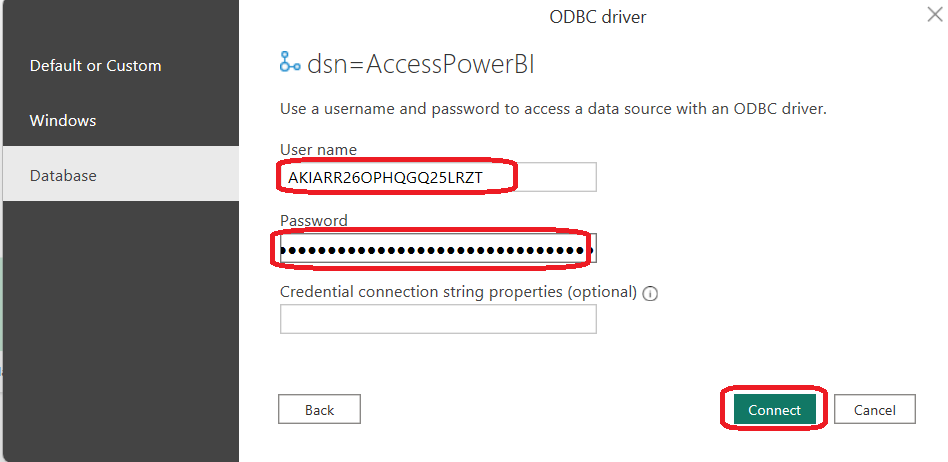


Elegimos el obdc creado y hacemos click en OK



Power BI nos solicita ingresar las credenciales de seguridad la cual debemos copiarla de IAM





Nos pudimos conectar con Athena



Creamos un grafico con los datos de Athena

