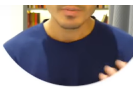
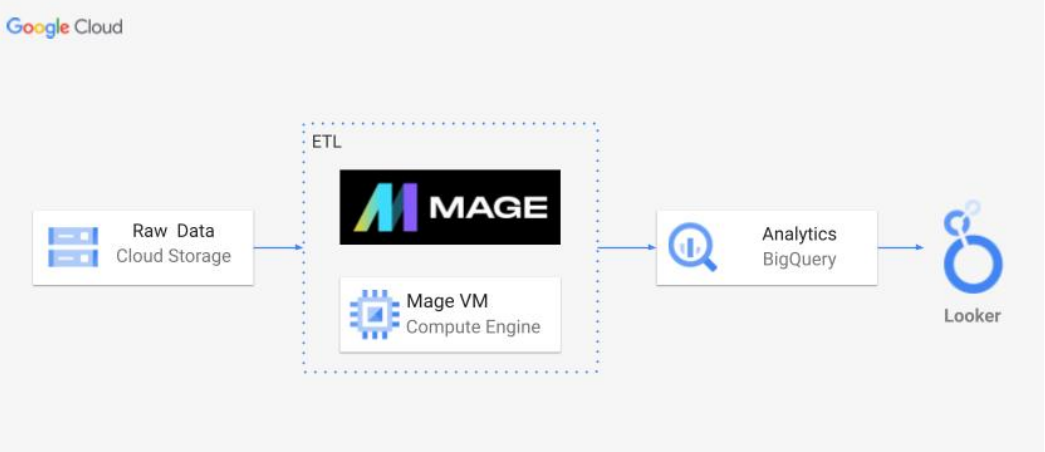
GCP Services



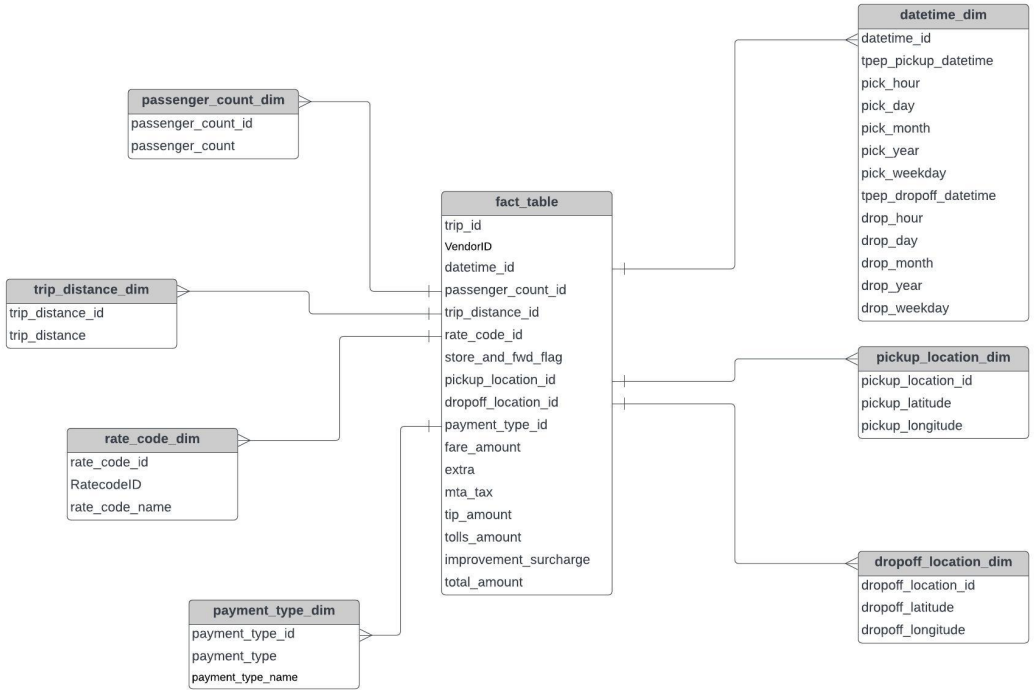
Cloud Storage	Google Cloud Storage is an online file storage service provided by Google as part of its cloud computing platform. It allows you to store and retrieve your data in the cloud, making it accessible from anywhere with an internet connection.
Compute Engine	Google Compute Engine is a cloud computing service that provides virtual machines for running applications and services. It allows you to easily create, configure, and manage virtual machines with various operating systems and hardware configurations.
BigQuery	BigQuery is a cloud-based data warehouse provided by Google Cloud Platform that allows you to store, analyze, and query large datasets using SQL-like syntax. It is a serverless, highly scalable, and cost-effective solution that can process and analyze terabytes to petabytes of data in real-time.
Looker	Looker Studio is a web-based data visualization and reporting tool that allows you to create interactive dashboards and reports from a variety of data sources, including Google Analytics, Google Sheets, and BigQuery. It enables you to turn your data into informative and engaging visualizations, which can be easily shared and collaborated on with others.

Arquitectura

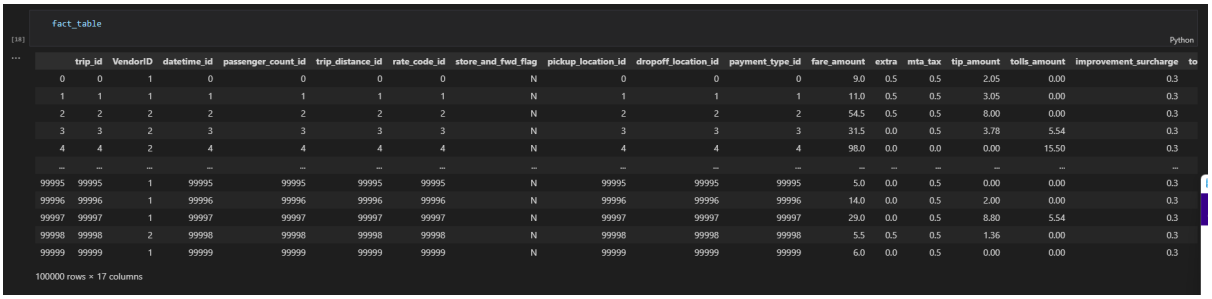


Extraemos los datos de tráfico de la página https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page

Modelado de los datos



Se manipula el dataset inicial para poder formar la estructura de la tabla fact_table. La transformación se puede ver en el notebook del proyecto.



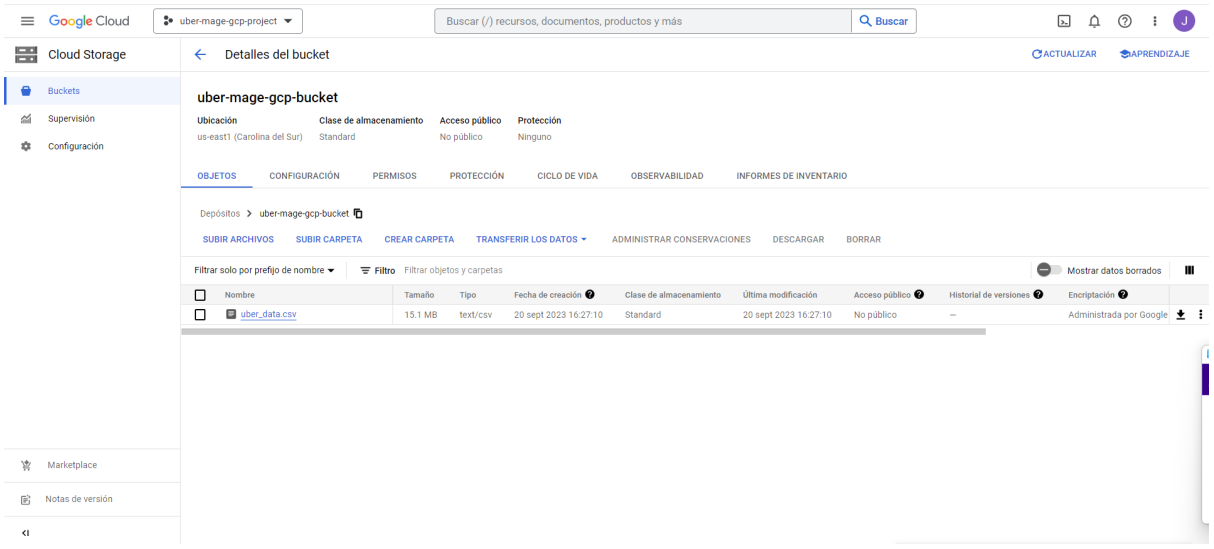
fact_table

[18] Python

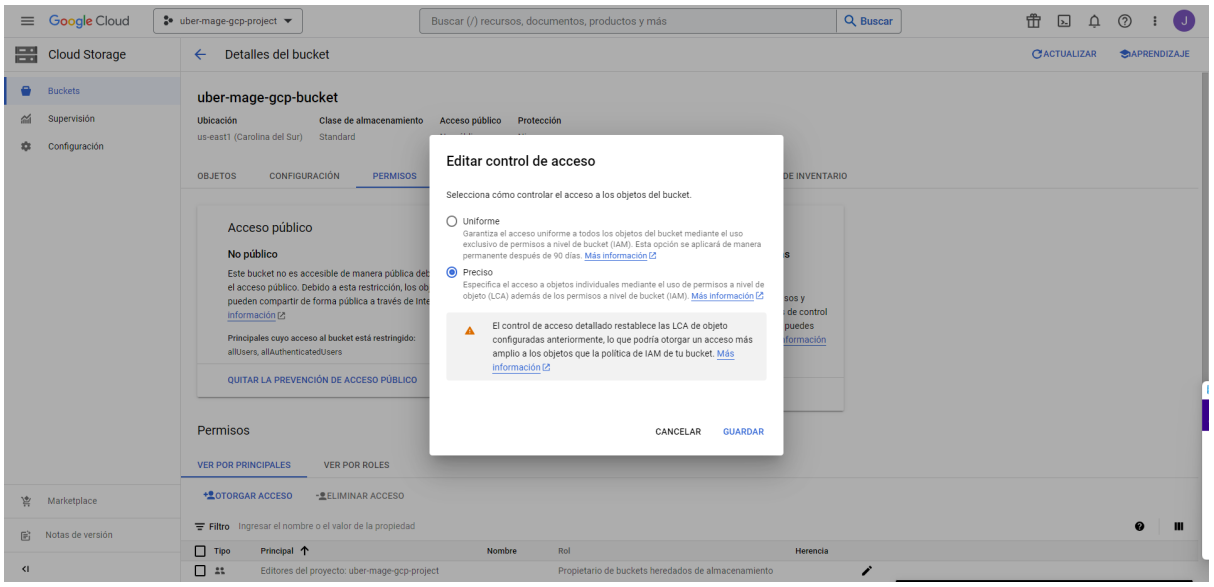
	trip_id	VendorID	datetime_id	passenger_count_id	trip_distance_id	rate_code_id	store_and_fwd_flag	pickup_location_id	dropoff_location_id	payment_type_id	fare_amount	extra	mta_tax	tip_amount	tolls_amount	improvement_surcharge	to
0	0	1	0	0	0	0	N	0	0	0	9.0	0.5	0.5	2.05	0.00	0.3	
1	1	1	1	1	1	1	N	1	1	1	11.0	0.5	0.5	3.05	0.00	0.3	
2	2	2	2	2	2	2	N	2	2	2	54.5	0.5	0.5	8.00	0.00	0.3	
3	3	2	3	3	3	3	N	3	3	3	31.5	0.0	0.5	3.78	5.54	0.3	
4	4	2	4	4	4	4	N	4	4	4	98.0	0.0	0.0	0.00	15.50	0.3	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
99995	99995	1	99995	99995	99995	99995	N	99995	99995	99995	5.0	0.0	0.5	0.00	0.00	0.3	
99996	99996	1	99996	99996	99996	99996	N	99996	99996	99996	14.0	0.0	0.5	2.00	0.00	0.3	
99997	99997	1	99997	99997	99997	99997	N	99997	99997	99997	29.0	0.0	0.5	8.80	5.54	0.3	
99998	99998	2	99998	99998	99998	99998	N	99998	99998	99998	5.5	0.5	0.5	1.36	0.00	0.3	
99999	99999	1	99999	99999	99999	99999	N	99999	99999	99999	6.0	0.0	0.5	0.00	0.00	0.3	

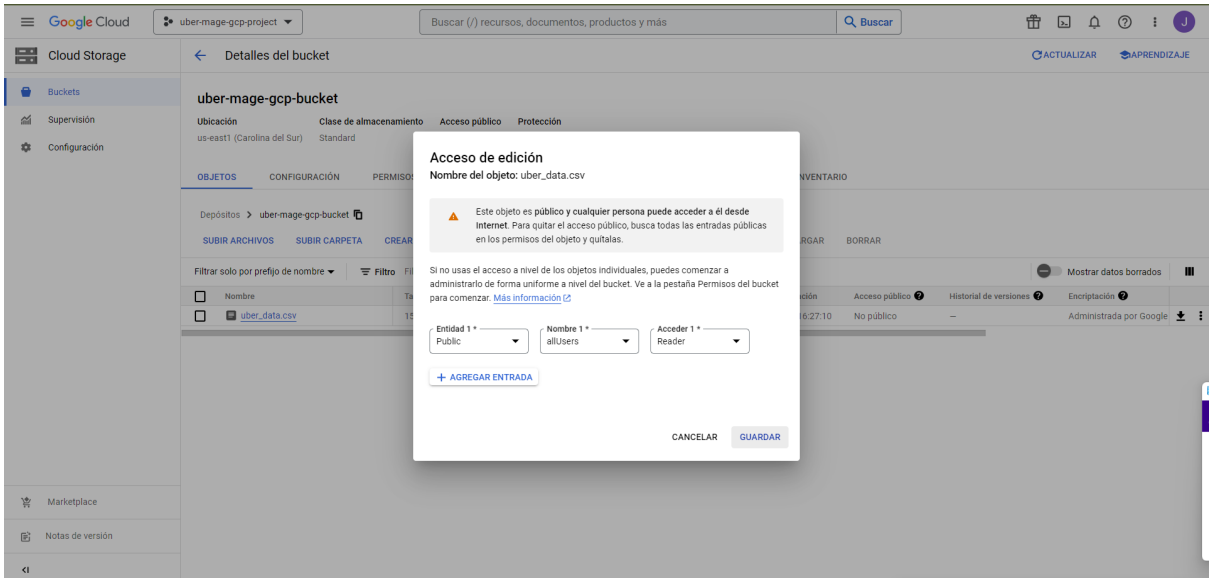
100000 rows × 17 columns

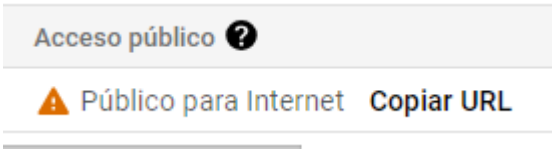
Iniciamos creando un bucket en el servicio Cloud Storage de GCP, y cargamos la data de Uber.



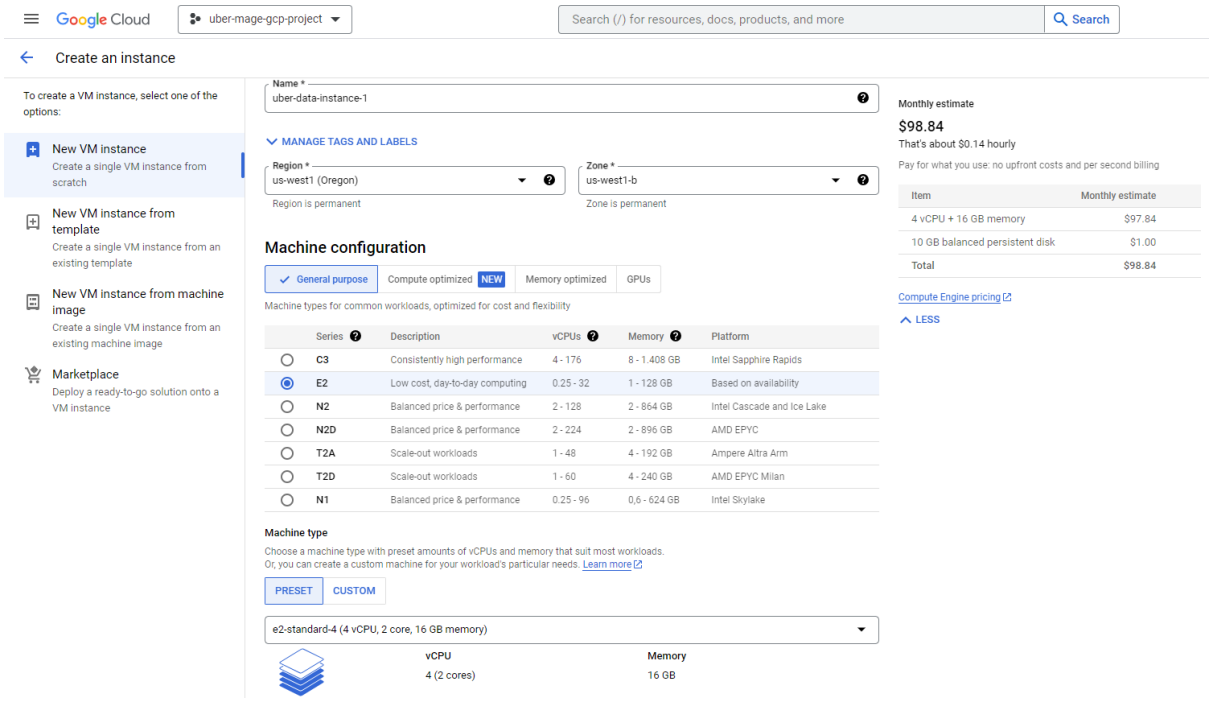
Editamos los permisos del archivo para que cualquiera pueda acceder.

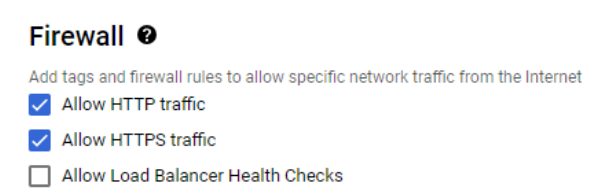




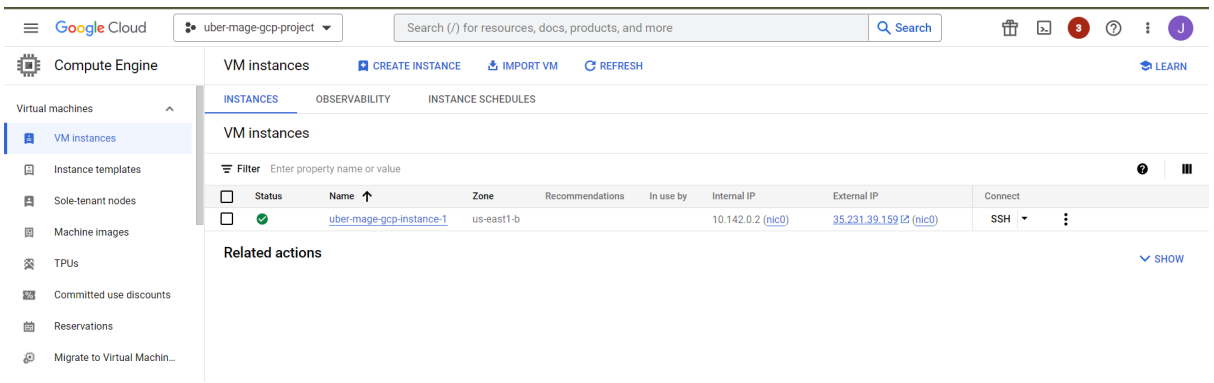


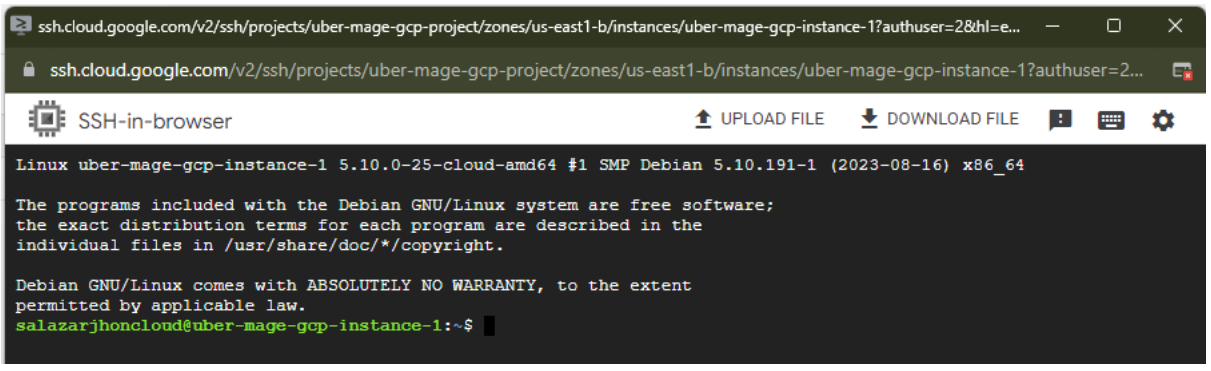
Creamos una VM instance en el servicio Compute Engine de GCP para poder utilizar nuestra máquina virtual.





Se puede ver la instancia creada. Hacemos click en la opción de SSH para conectarnos.





Al ingresar a la consola, corremos los siguientes comandos para poder utilizar python3.

sudo apt-get update -y
sudo apt-get install python3-distutils
sudo apt-get install python3-apt
wget https://bootstrap.pypa.io/get-pip.py
sudo python3 get-pip.py

Luego, instalamos las dependencias.

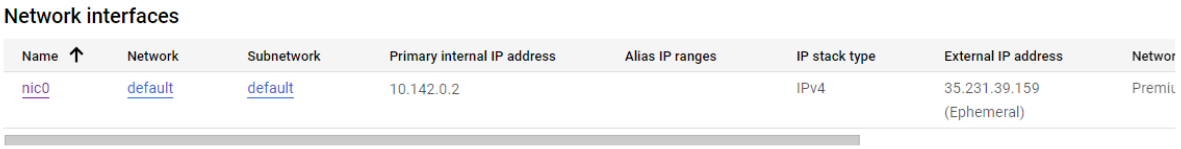
sudo pip3 install pandas
sudo pip3 install google-cloud
sudo pip3 install google-cloud-bigquery
sudo pip3 install mage-ai (Entorno para manejar el workflow)

Empezamos a utilizar **mage**.

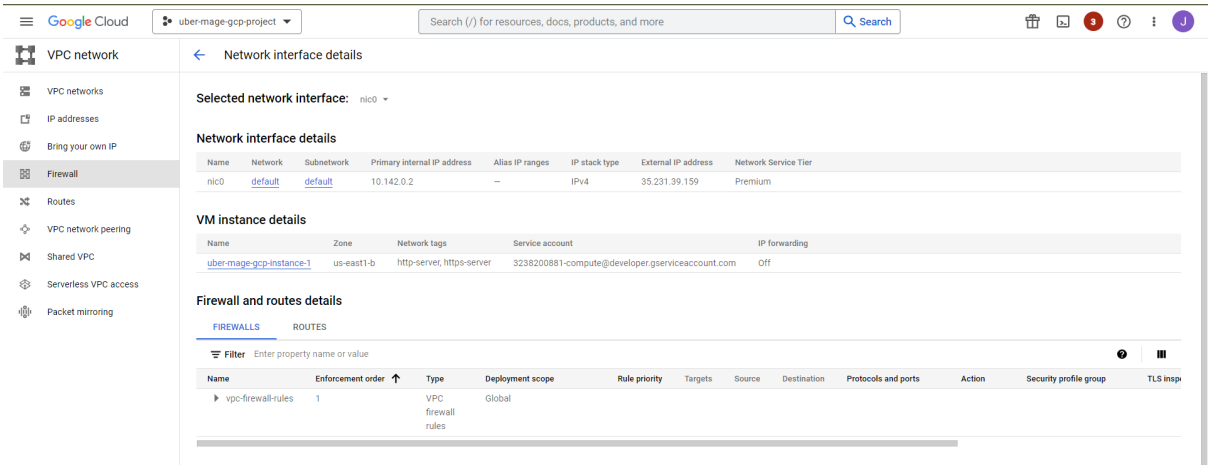
mage start uber_gcp_project

```
salazarjhoncloud@uber-mage-gcp-instance-1:~$ mage start uber_gcp_project
WARNING:traitlets:Message signing is disabled.  This is insecure and not recommended!
WARNING:traitlets:Message signing is disabled.  This is insecure and not recommended!
INFO:mage_ai.server.scheduler_manager:Start scheduler.
INFO:mage_ai.server.active_kernel:Switch active kernel: python3
INFO  [alembic.runtime.migration] Context impl SQLiteImpl.
INFO  [alembic.runtime.migration] Will assume non-transactional DDL.
INFO  [alembic.runtime.migration] Running upgrade  -> c07a23ff782b, Add initial tables.
INFO  [alembic.runtime.migration] Running upgrade c07a23ff782b -> 52ab80005742, Add variables to pipeline schedule.
INFO  [alembic.runtime.migration] Running upgrade 52ab80005742 -> 053ee2c10d85, Add completed_at.
INFO  [alembic.runtime.migration] Running upgrade 053ee2c10d85 -> 3fafd217efa7, Add event_matcher model.
INFO  [alembic.runtime.migration] Running upgrade 3fafd217efa7 -> 067326f43bc3, Add variables to PipelineRun.
INFO  [alembic.runtime.migration] Running upgrade 067326f43bc3 -> b01be687e537, Add started_at to block run
INFO  [alembic.runtime.migration] Running upgrade b01be687e537 -> 84de4cdd6126, Add sla to pipeline schedule
INFO  [alembic.runtime.migration] Running upgrade 84de4cdd6126 -> 5cd59ec4cf1d, Add passed_sla to pipeline run
INFO  [alembic.runtime.migration] Running upgrade 5cd59ec4cf1d -> 2266370f589b, Add indexes to pipeline run.
INFO  [alembic.runtime.migration] Running upgrade 2266370f589b -> 8971d4cd5b39, add event_variables and metadata to pipeline_runs
INFO  [alembic.runtime.migration] Running upgrade 8971d4cd5b39 -> ec5df57a1c60, Add metrics to block_runs
INFO  [alembic.runtime.migration] Running upgrade ec5df57a1c60 -> 7ac6fed06918, Add token column to pipeline_schedule table
INFO  [alembic.runtime.migration] Running upgrade 7ac6fed06918 -> 6aecc9bc451c, Update schedule type enum
INFO  [alembic.runtime.migration] Running upgrade 6aecc9bc451c -> 26305e46df52, Create backfills table
INFO  [alembic.runtime.migration] Running upgrade 26305e46df52 -> 1d8e65aeefdd, Add settings to pipeline schedule
INFO  [alembic.runtime.migration] Running upgrade 1d8e65aeefdd -> 97ff9f55a3c0, Create users, oauth2 applications, and oauth2 access tokens
INFO  [alembic.runtime.migration] Running upgrade 97ff9f55a3c0 -> 643b6e65e814, Add unique indexes on authentication tables
Checking port 6789...
INFO:mage_ai.server.server:Mage is running at http://localhost:6789 and serving project /home/salazarjhoncloud/uber_gcp_project
INFO:mage_ai.server.server:Initializing block cache.
INFO:mage_ai.server.server:Initializing tag cache.
INFO  [alembic.runtime.migration] Running upgrade 643b6e65e814 -> 1f9353eddbc6, Add secrets table
```

Vamos a la sección de interfaces de red de nuestra instancia. Seleccionamos la interfaz *nic0* para agregar una regla de firewall que nos permite utilizar el puerto 6789 de mage.

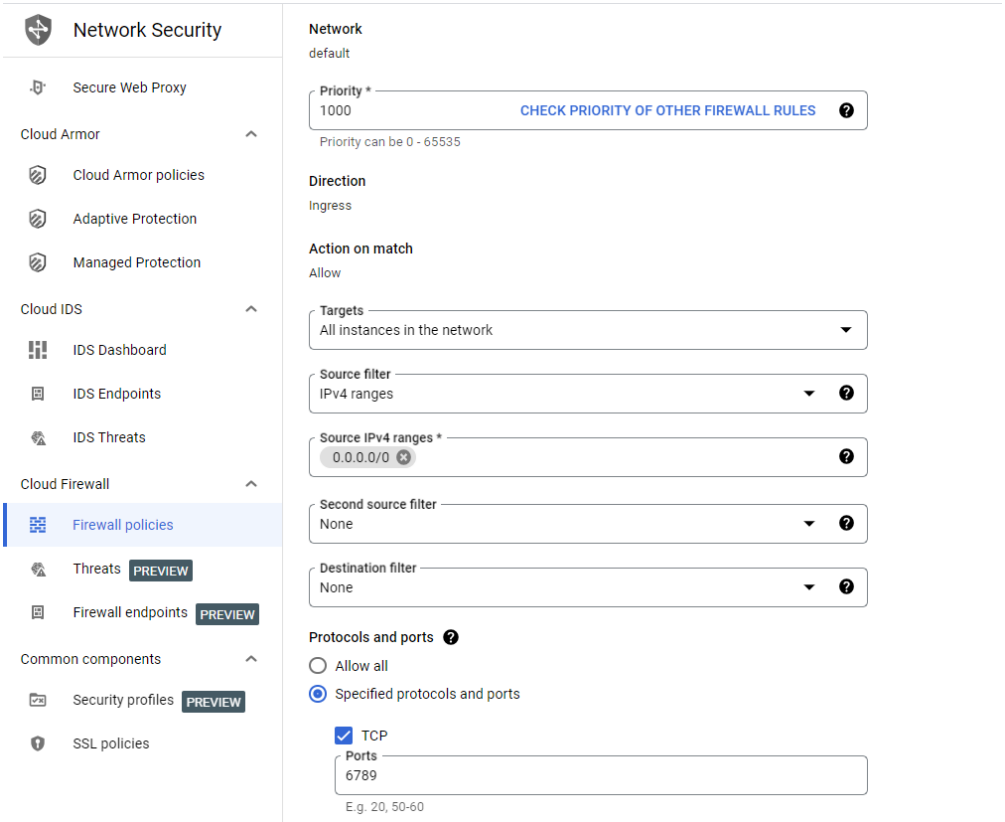


En el menú vertical seleccionamos la opción **Firewall**.

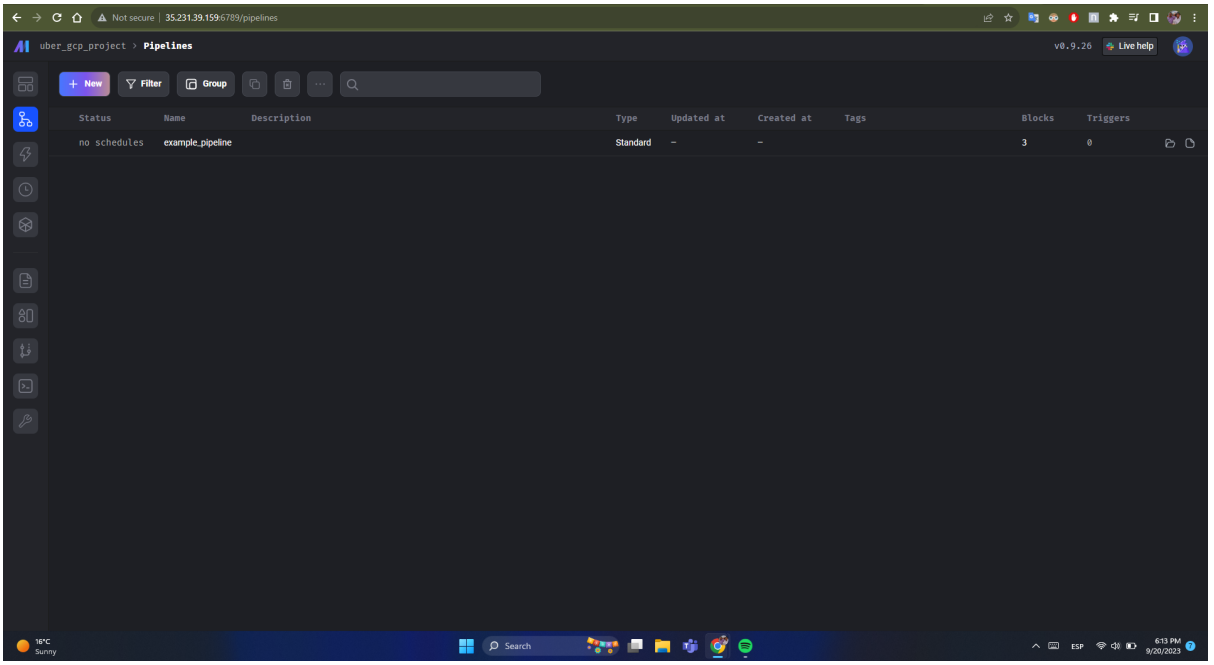


En este punto seleccionamos la opción de crear una nueva regla de firewall.

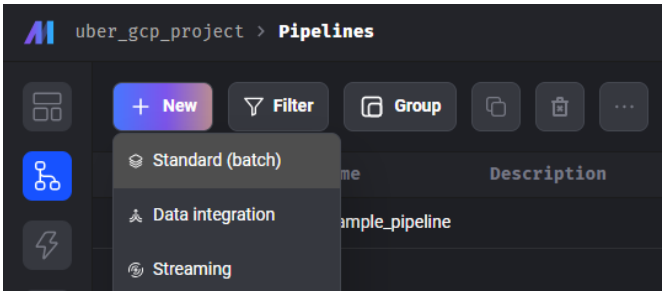
La configuración es la siguiente.



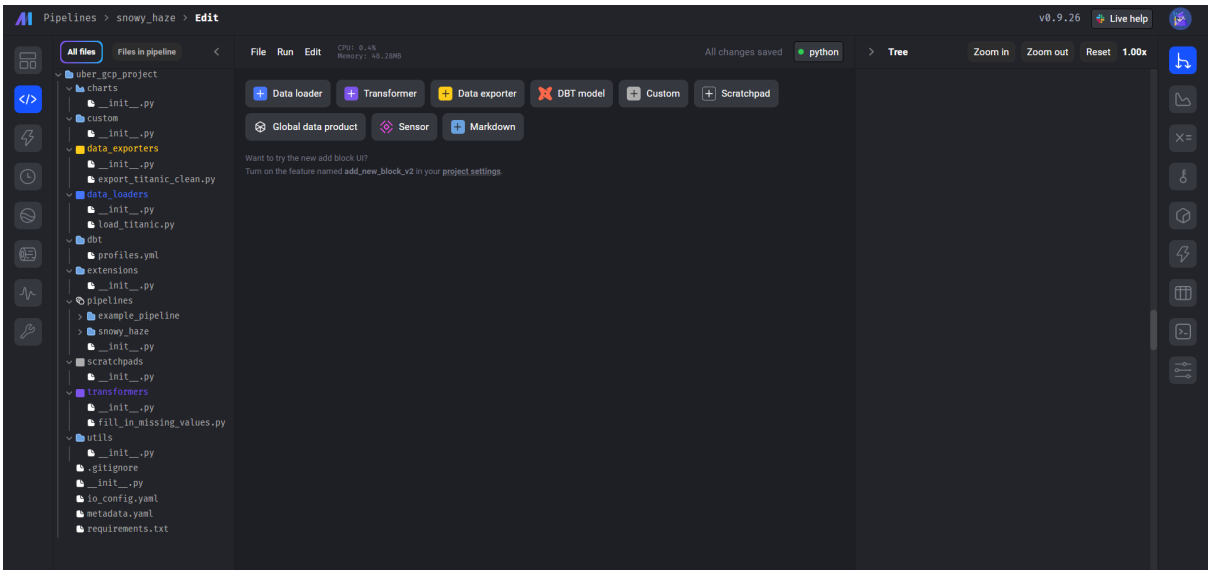
Al intentar acceder nuevamente a **mage**, tomando la dirección IP externa de la instancia y el puerto proporcionado por mage, podemos ver la siguiente interface.



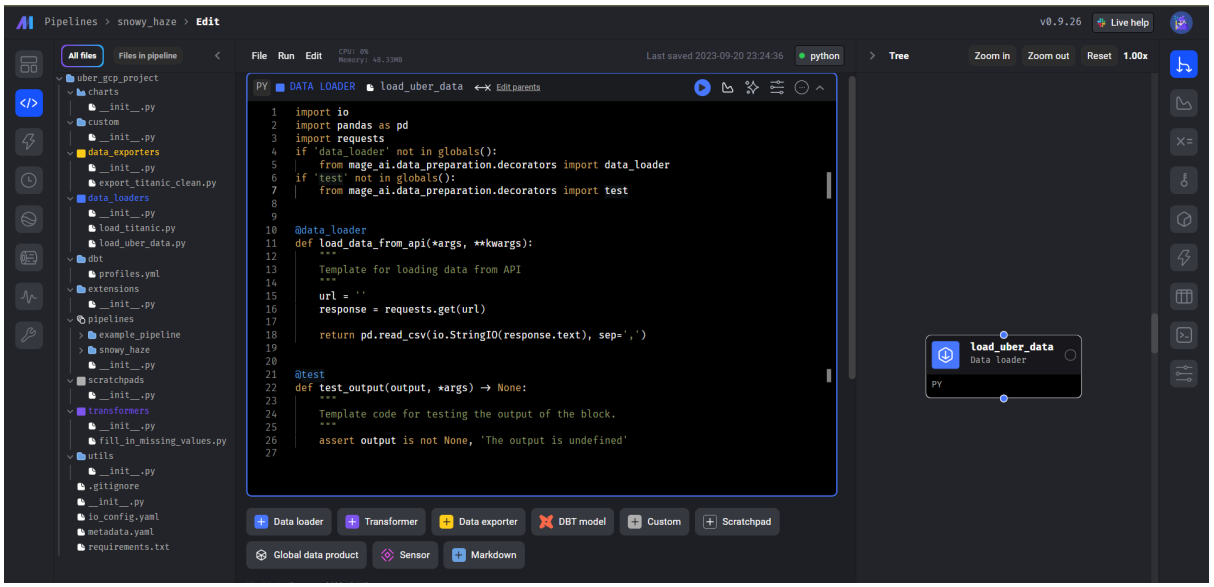
Creamos un escenario Standard.



Mage ofrece template para apoyarse y generar scripts más fácilmente.

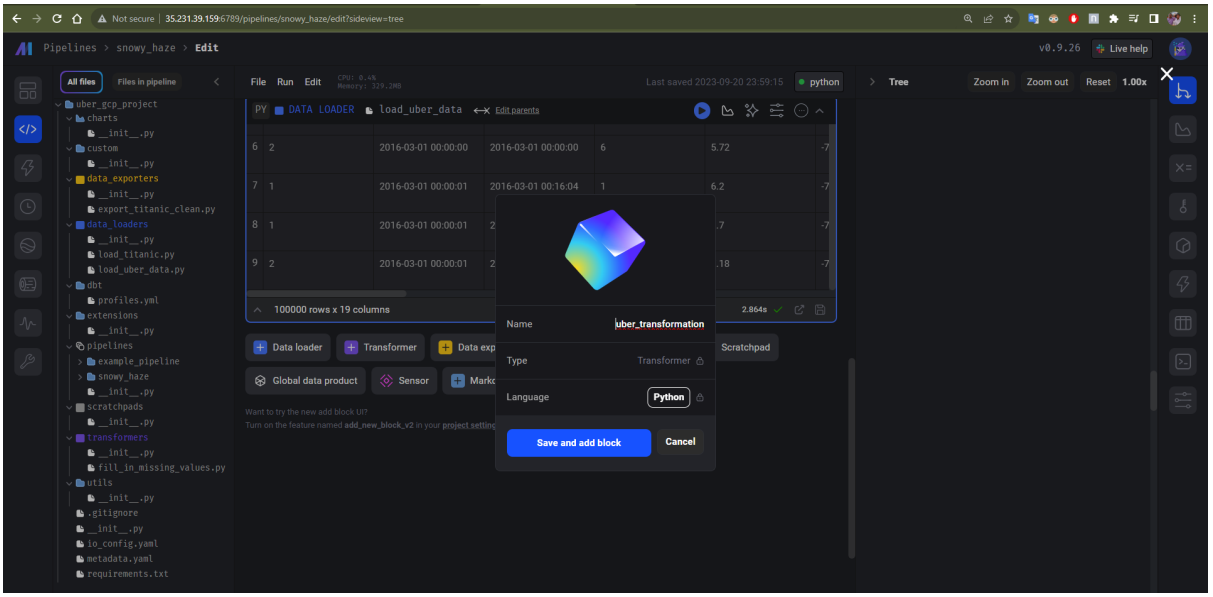


Creamos un Data loader de tipo API, ya que podemos acceder a nuestro storage de GCP de forma sencilla a través de la url pública.

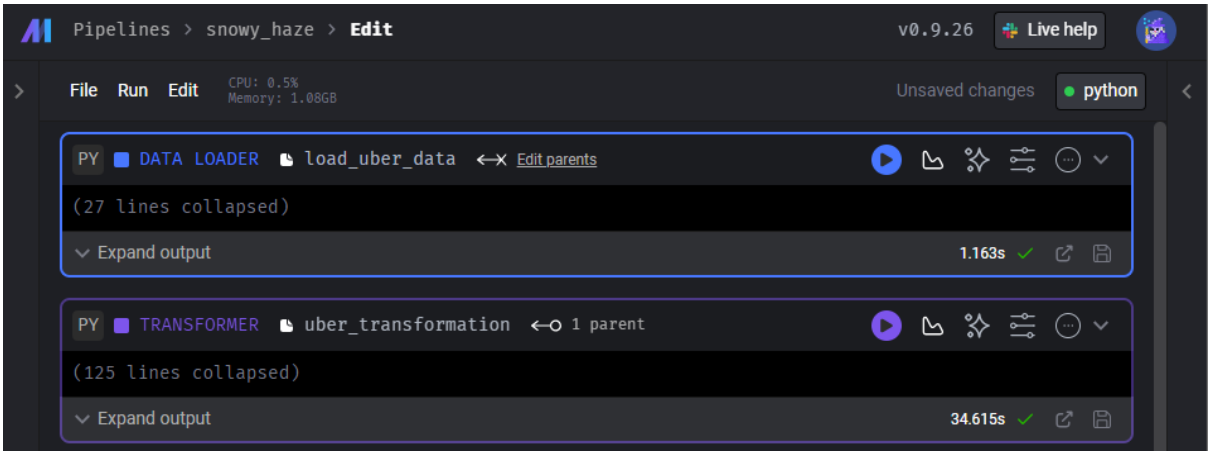


Corremos este bloque y podemos ver los datos del csv de uber.

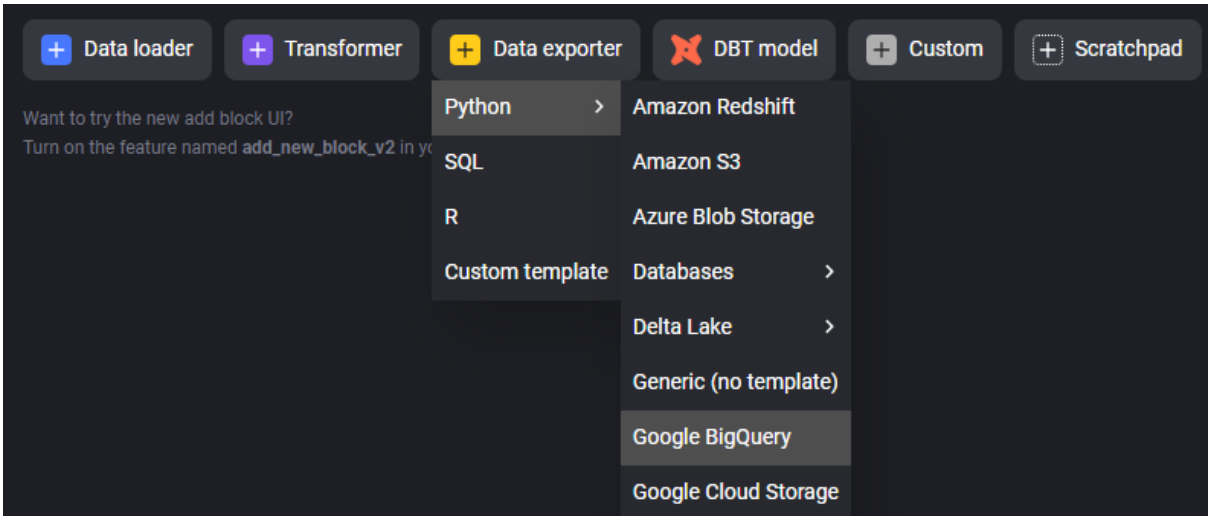
Luego creamos un Transformer (Generic) para manipular estos datos y obtener el modelo mencionado anteriormente.



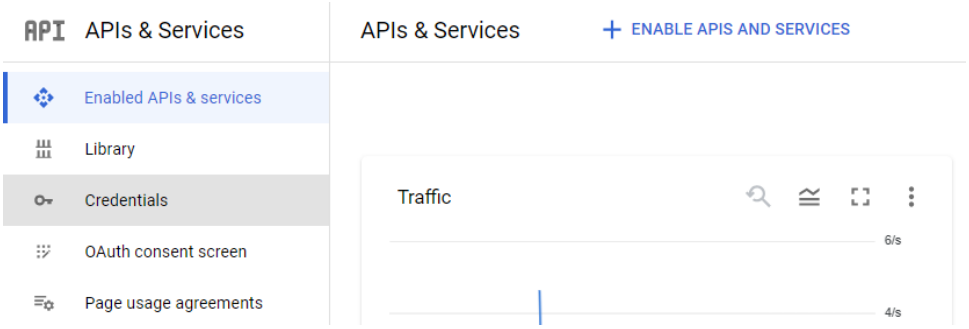
En el transformer simplemente pegamos el código realizado en el notebook. Hasta el momento tenemos estos 2 bloques. Ahora cargaremos la información en BigQuery.



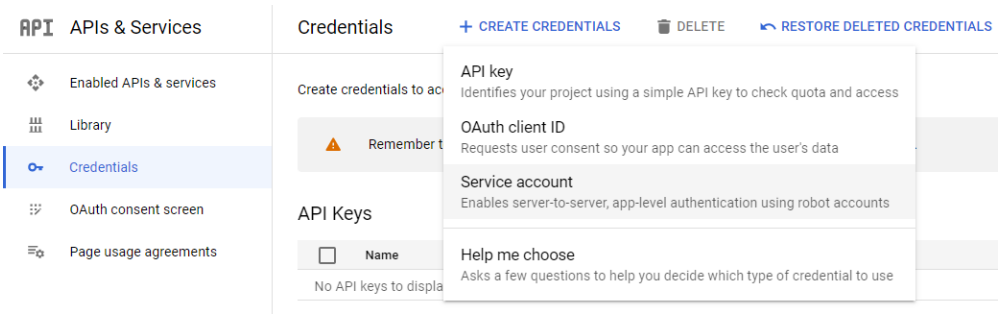
Creamos el bloque Data exporter.



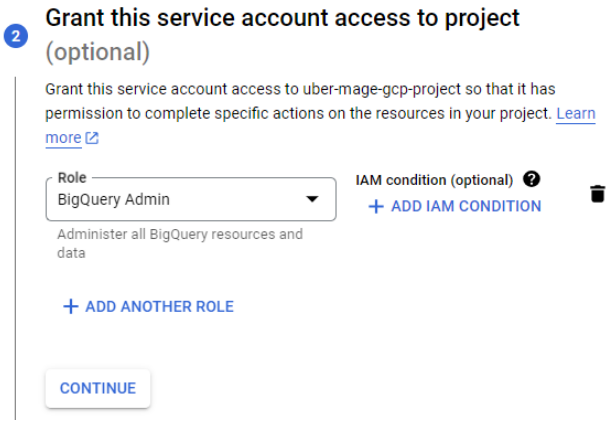
Para poder conectarnos desde la VM a cualquier servicio de GCP necesitamos unas credenciales. Vamos a la sección de APIs & Services y seleccionamos la opción Credentials.



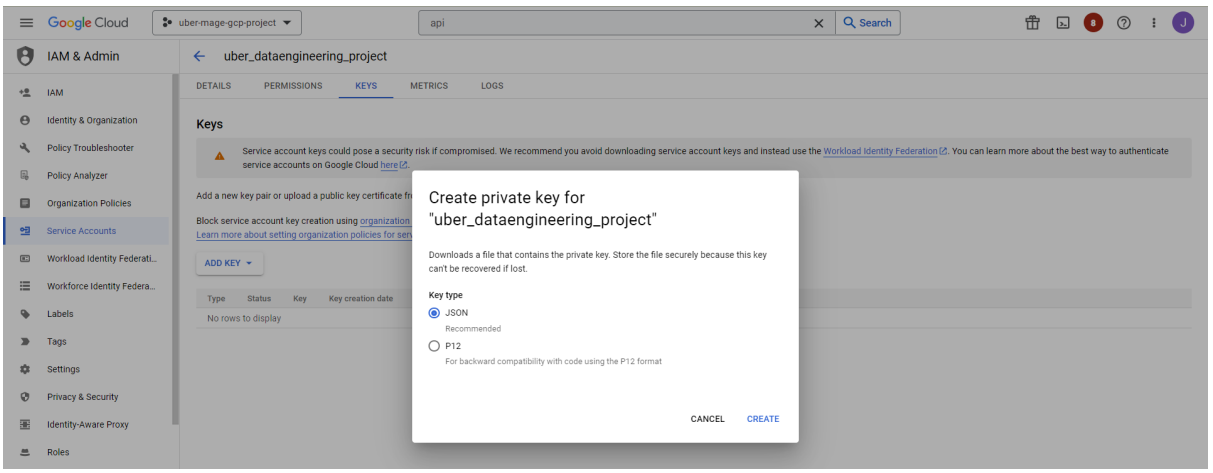
Seleccionamos crear credenciales y elegimos Service account.



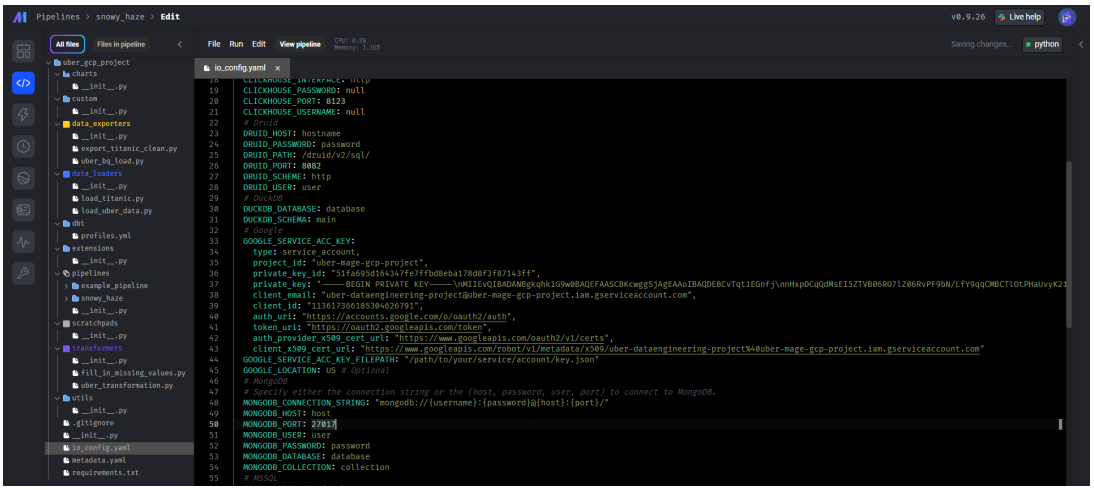
Escogemos el rol de BigQueryAdmin para poder crear tablas, editar, leer, etc.



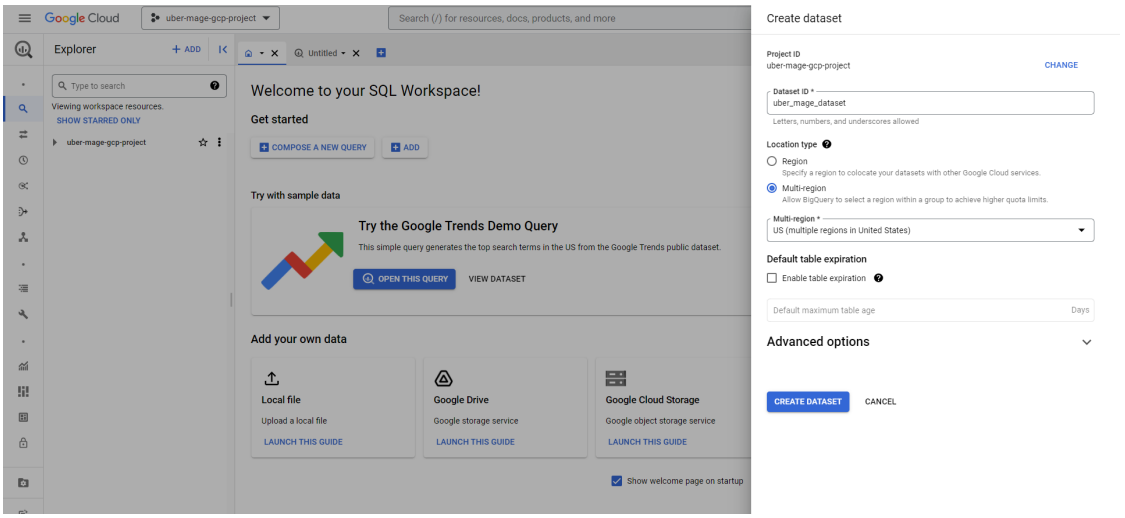
Seleccionamos la cuenta creada y generamos una key en formato JSON.



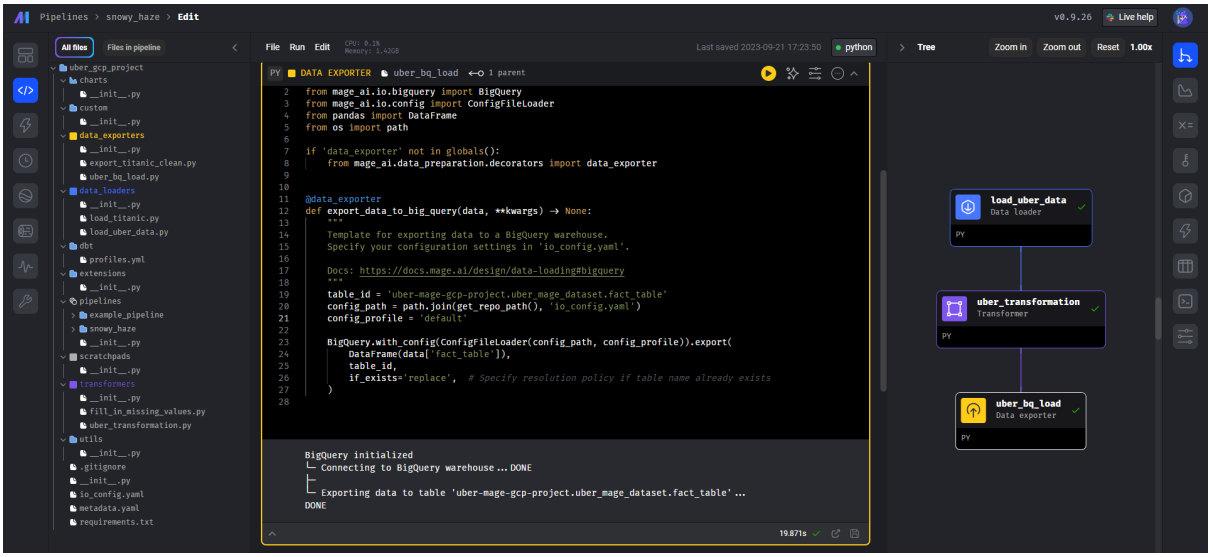
Nos dirigimos al archivo io_config.yaml e ingresamos nuestras credenciales generadas para la sección GOOGLE_SERVICE_ACC_KEY.

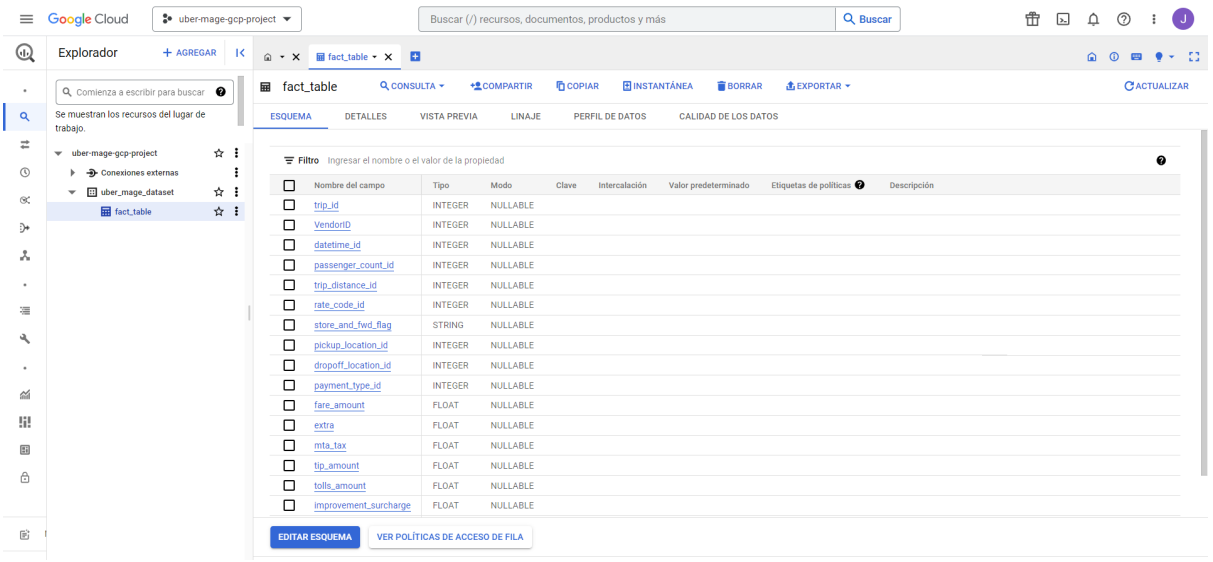


Ingresamos al servicio BigQuery (Datawarehouse) y creamos nuestro dataset dentro del proyecto correspondiente.

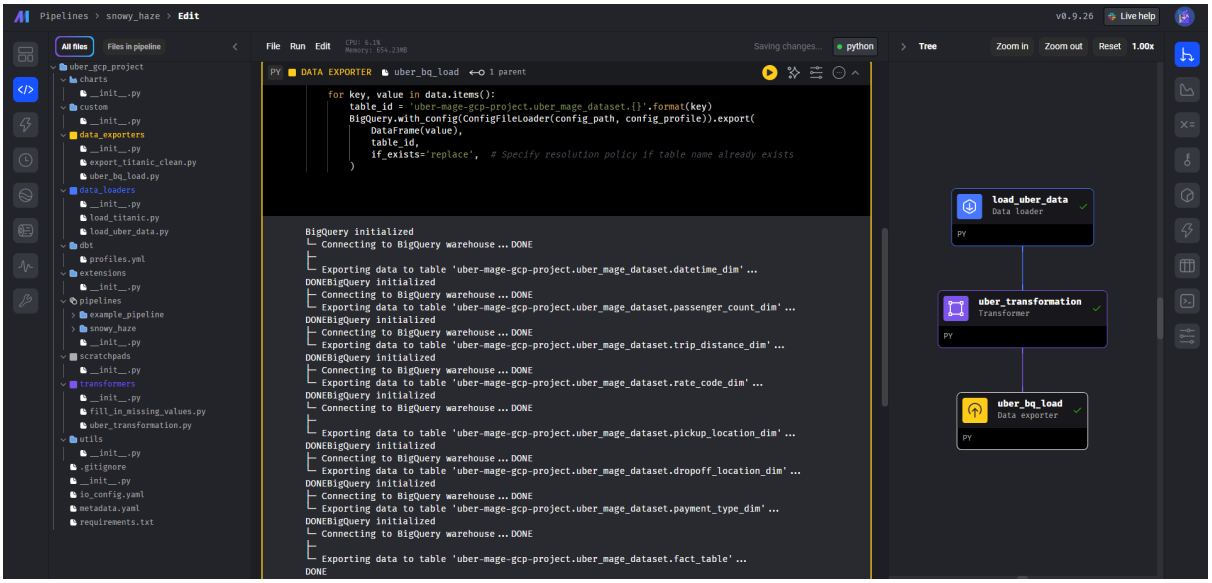


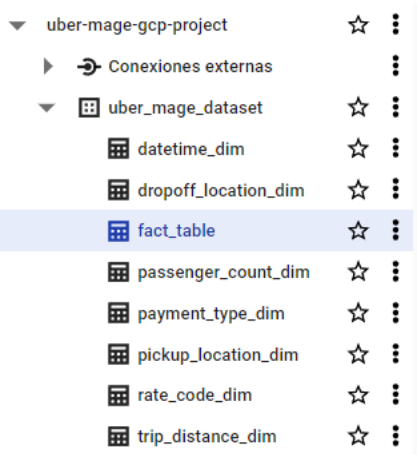
En el bloque de Data Exporter ingreamos los datos de nuestro dataset en BigQuery y lo ejecutamos. Como se puede ver, la conexión fue exitosa y se creó la nueva tabla.



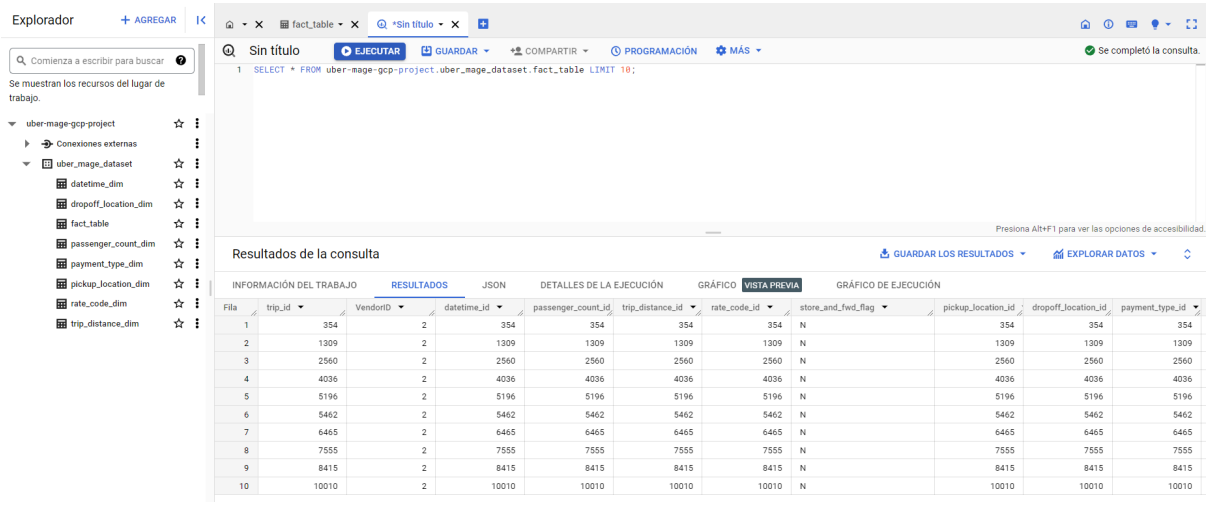


Ejecutamos nuevamente este proceso pero para todas las tablas.

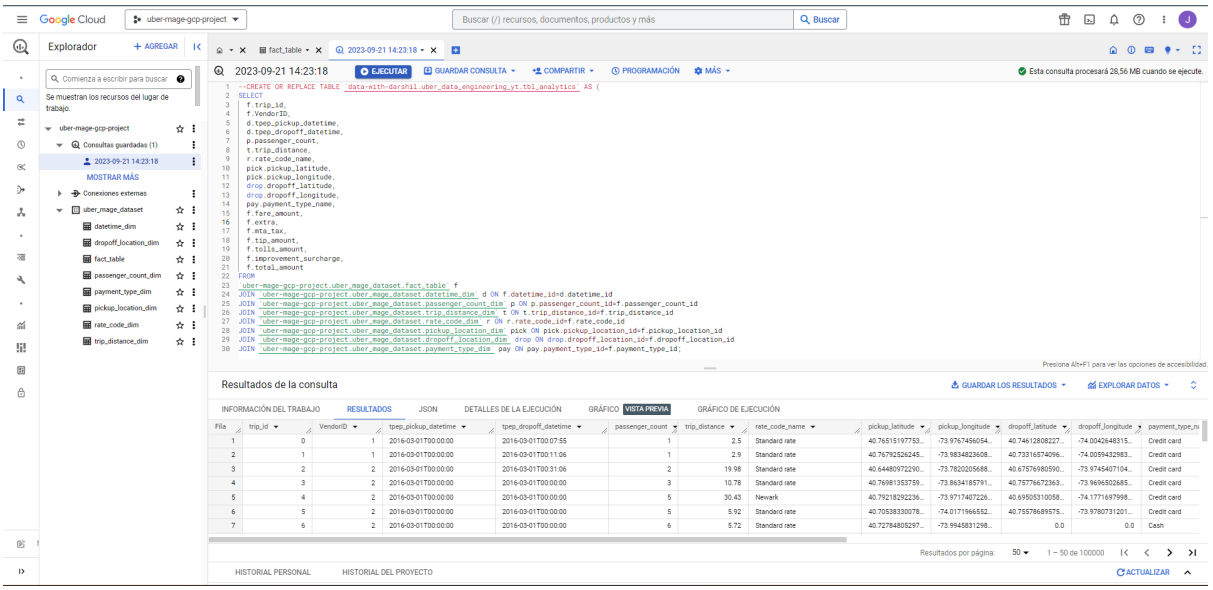




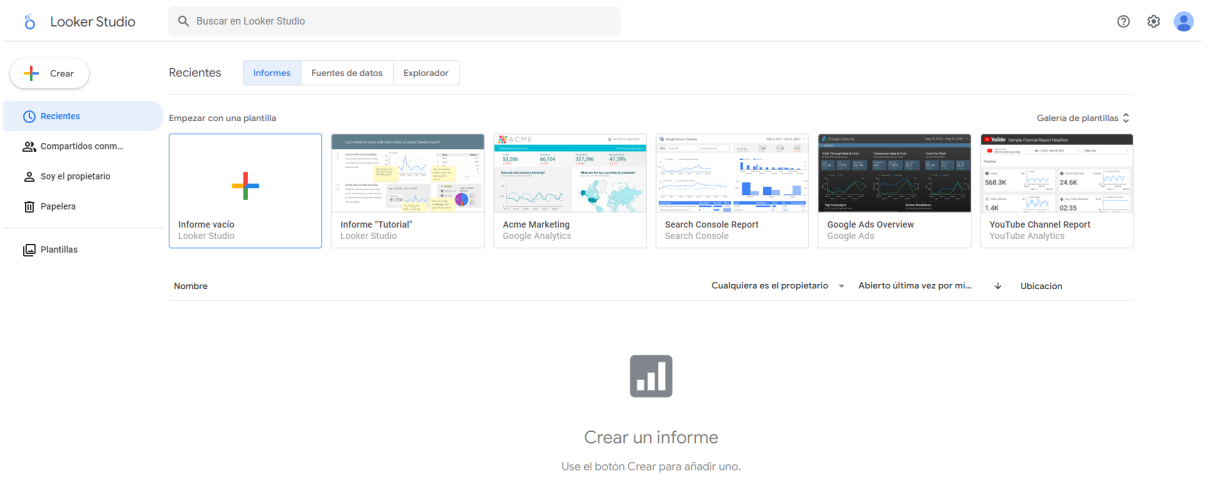
Ejecutamos una sentencia sencilla.



En este punto, queremos formar una nueva tabla con columnas provenientes de todas las tablas creadas anteriormente. El query final es el siguiente.

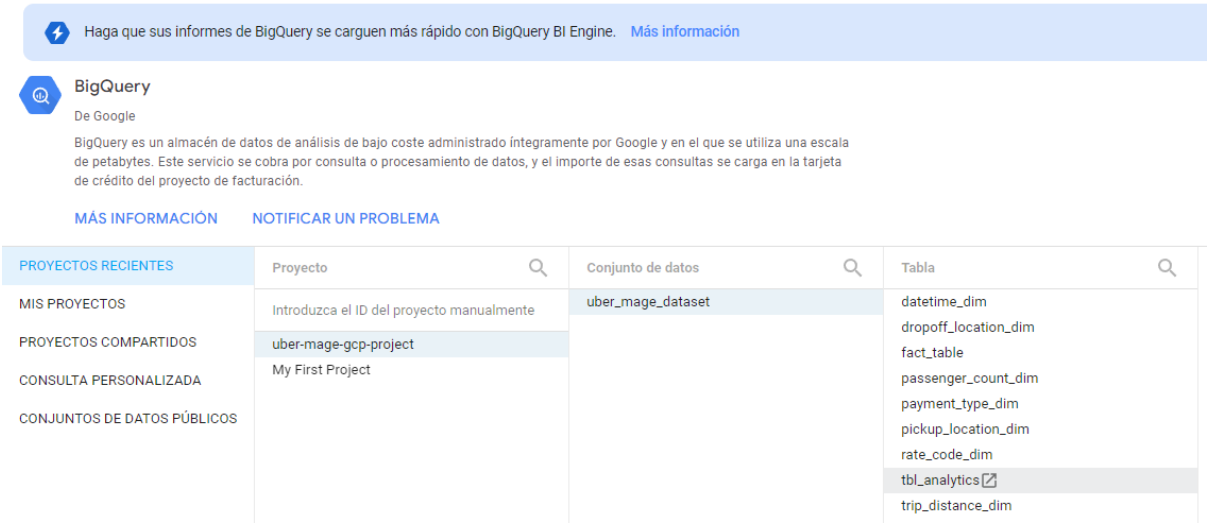


Creamos la tabla y queda almacenada en nuestro proyecto. Es la que usaremos para armar el dashboard. Para ellos ingresamos a Looker Studio y creamos un nuevo informe vacío.



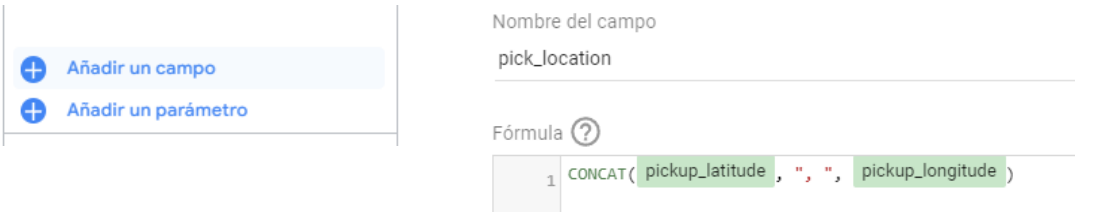
Elegimos la opción de conectarnos a BigQuery.

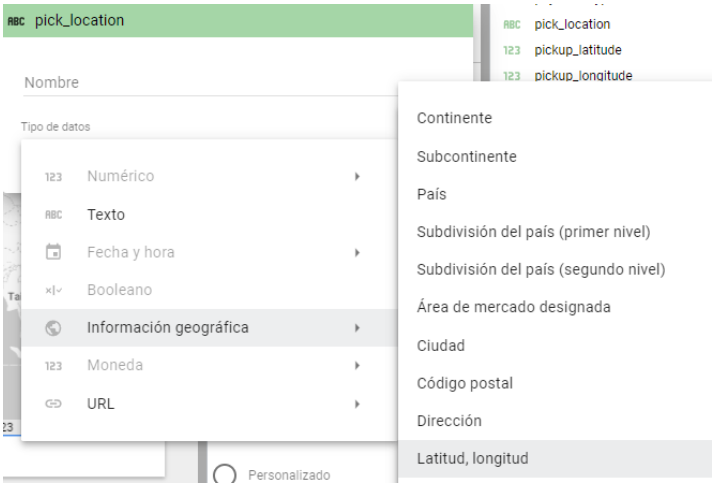
Escogemos nuestra tabla.



Dentro del informe podemos manejar diferentes elementos para ilustrar nuestra información de la mejor manera. En principio agregamos filtros, luego unas cifras relevantes, un mapa con las rutas de los viajes realizados y un par de gráficos de barras.

Para agregar el mapa, debemos crear un nuevo campo basado en la latitud y longitud del pickup.





Finalmente el dashboard nos queda de la siguiente forma.