**Evidencias sistema de aplicación**

**Data Warehouse HVR**

# 

**Contenido**

[**Objetivo 2**](#_33j60uzdcc4f)

[**Alcance 2**](#_p8lbwo2o436j)

[Arquitectura 3](#_nu7ieggb6w0v)

[Origen de Datos 4](#_jl92r9pe71ht)

[Realizar pruebas de conexión desde el servidor local a GCP 4](#_jybgj3lcoa2a)

[Configuración de Buckets en Cloud Storage 7](#_ddqhh944f5lu)

[Cloud Functions para la carga de datos en BigQuery 11](#_kf5423uq17in)

[Cloud Function lector\_de\_bucket\_dev 11](#_z3u3nvez2jm7)

[Cloud Function carga\_csv\_bq\_bronze\_dev 12](#_3trh144lvkv5)

[Dataform para gestión de transformaciones en BigQuery 13](#_uwo4edbfs052)

[Configuración de git en Dataform para gestión de transformaciones en BigQuery 16](#_62xa1bfuqclg)

[BigQuery 18](#_i7ef28qqx92s)

[Looker Studio 20](#_2lh1ys6pk4jy)

[Seguridad y Gobernanza 20](#_nxnu7n30sjnt)

# 

# 

# Objetivo

Diseñar y desarrollar un Data Warehouse alojado en GCP que optimice la carga diaria de información, entrenar un modelo ML predictivo para un caso de negocio y la generación de informes y dashboards de la empresa HVR.

# Alcance

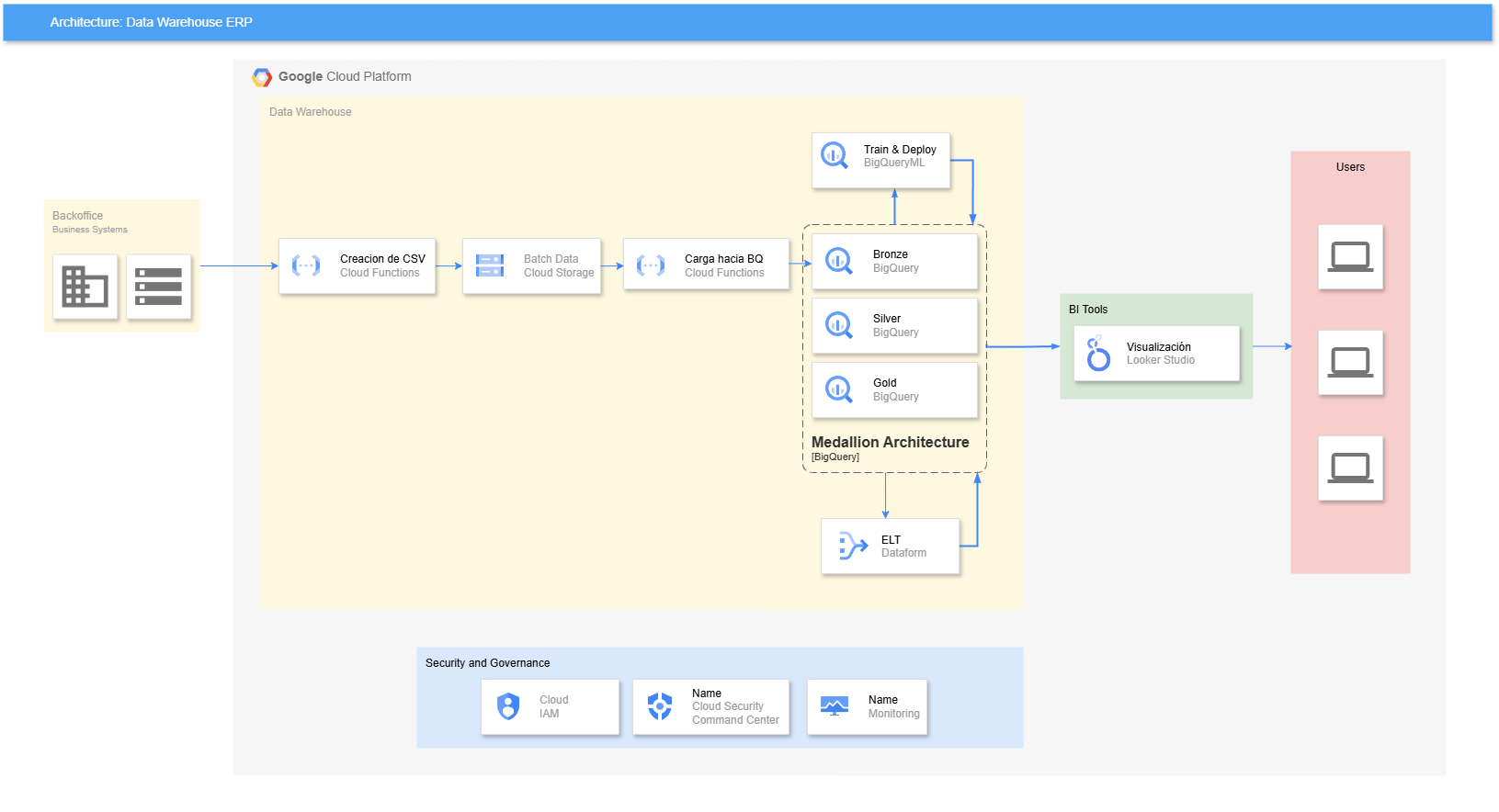
El proyecto busca construir un data warehouse en Google Cloud Platform (GCP) para la carga y transformación de datos a BigQuery. Incluirá la creación de un flujo de datos con capas raw, silver y gold, el uso de BigQuery ML para entrenar un modelo de regresión lineal, y la generación de informes de BI personalizados mediante Looker Studio.

Dentro de los objetivos se encuentran los siguientes puntos:

* Configuración de la infraestructura en GCP: Configuración de buckets, políticas de retención de almacenamiento,conjuntos de datos en BigQuery, permisos en BigQuery, Cloud Functions para la carga de datos, Configuración de git en Dataform, BigQuery ML para Machine Learning, Looker Studio para dashboards interactivos, roles y permisos (IAM), etc.
* Extracción de Datos en CSV: Extraer datos en formato CSV desde un servidor SQL Server externo hacia Google Cloud Storage.
* Optimización del Proceso de ETL: Cargar datos desde los CSV y transformarlos en BigQuery a través de distintas capas (raw, silver y gold).
* Creación de un Modelo Estrella: Organizar los datos en la capa gold usando un modelo estrella que optimice las consultas y el análisis.
* Entrenamiento de un Modelo de ML: Usar BigQuery ML para entrenar un modelo de regresión lineal sobre los datos transformados.
* Visualización y Reporting: Generar informes dinámicos en Looker Studio con roles de acceso controlados.
* Seguridad, Monitoreo y Alertas: Garantizar la protección de los datos y el monitoreo continuo de los flujos de trabajo, con alertas ante fallos o comportamientos anómalos.
* Gestión de Metadatos: Utilizar Data Catalog para organizar, clasificar y documentar los datasets y tablas, facilitando su búsqueda y gestión.
* Automatización y Orquestación: Implementar Google Cloud Workflows para gestionar y automatizar el flujo de trabajo de ETL.
* Almacenamiento Intermedio: Utilizar Google Cloud Storage (GCS) para almacenar temporalmente datos intermedios durante el proceso de ETL.

# 

# Arquitectura



Utilizaremos Google Cloud Platform para implementar un Data Warehouse escalable y seguro, diseñado para la integración de datos desde sistemas de negocio hacia capas de transformación en BigQuery, utilizando una arquitectura en medallones (bronze, silver, y gold). Esto permite una organización jerárquica y estructurada de los datos que facilita su procesamiento y análisis.

Para realizar el flujo de datos implementaremos diferentes servicios de GCP que nos ayudarán a automatizar y ejecutar cada etapa.

# Origen de Datos

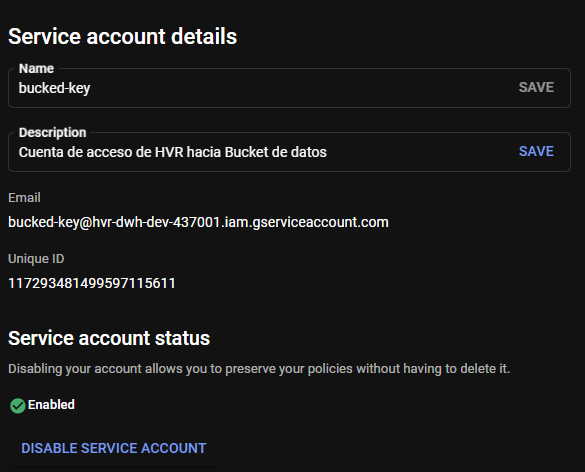
## Realizar pruebas de conexión desde el servidor local a GCP

Se realiza el proceso de configuración de una conexión segura y estable entre el servidor del cliente y el bucket de Google Cloud Platform (GCP) raw\_data\_hvr\_dev, con el objetivo de automatizar la carga diaria de archivos CSV. El proceso incluyó la creación de una cuenta de servicio en GCP con permisos específicos, la configuración de una key en el servidor del cliente para acceso seguro, y la verificación de la correcta transferencia de archivos.

El objetivo principal fue asegurar un flujo de datos confiable y seguro para la carga de documentos CSV desde el servidor hacia el bucket en GCP. Para esto, se llevaron a cabo varias actividades, incluyendo la creación y configuración de una cuenta de servicio, la generación de una key de acceso, y la ejecución de pruebas de carga automática de archivos.

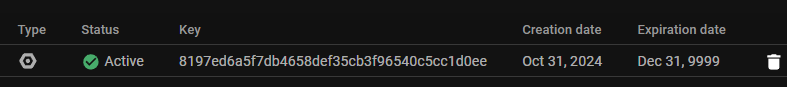
**Creación de la cuenta de servicio:**

Se crea una cuenta de servicio con permisos de administrador sobre el bucket raw\_data\_hvr\_dev.



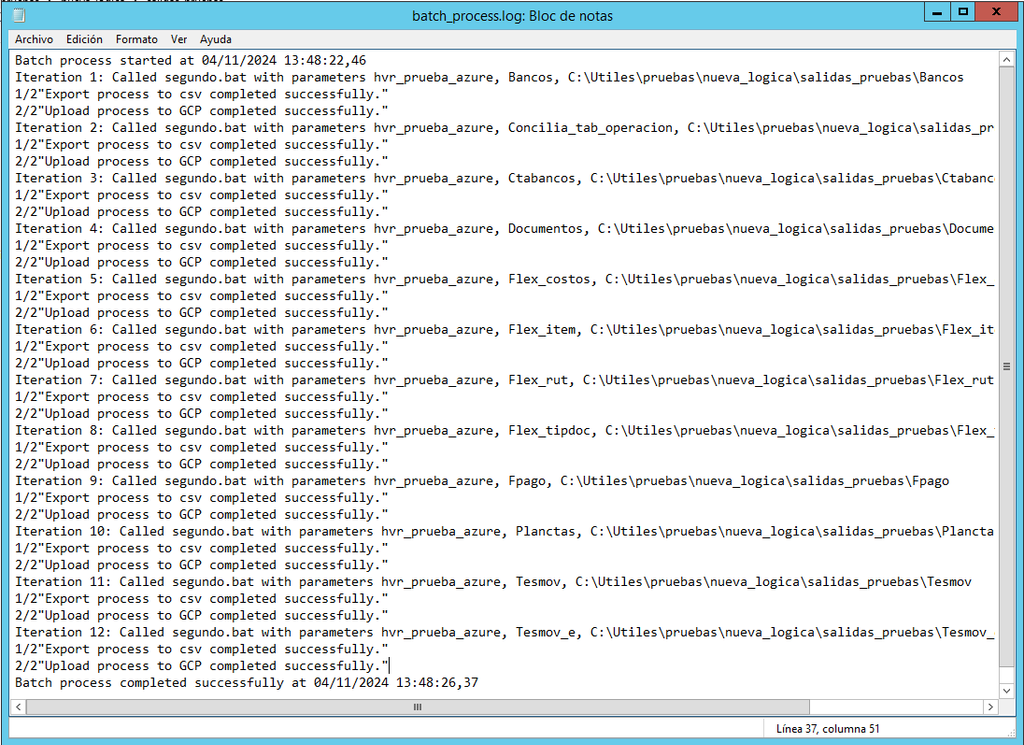
**Creación de Key de acceso**

Se crea una Key para conexión segura desde el servidor de cliente HVR hacia el bucket raw\_data\_hvr\_dev

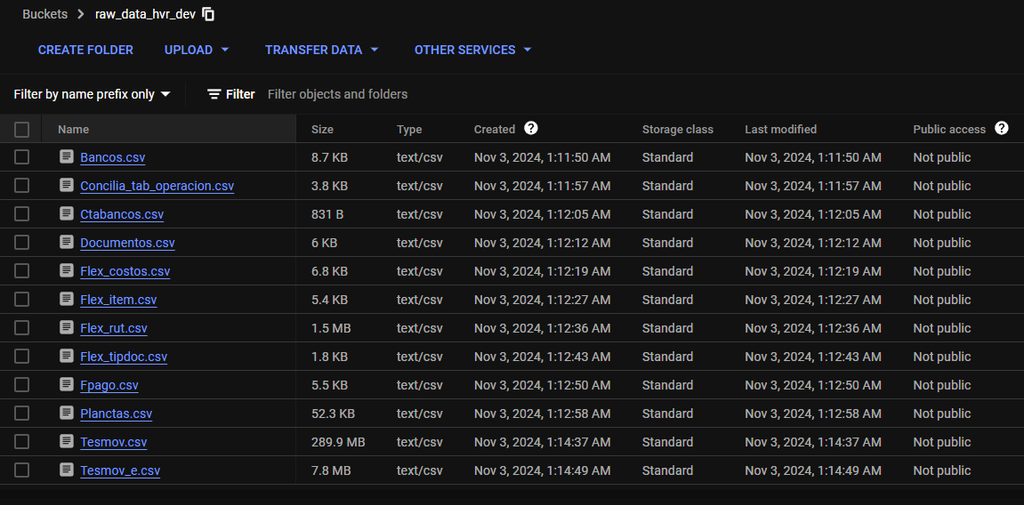


**Evidencia de log entregados por el cliente:**

Archivo log con la secuencia de carga de datos hacia el Bucket



Evidencia de documentos alojados en el bucket:



## **Configuración de Buckets en Cloud Storage**

Para realizar la creación de los Buckets en Cloud Storage daremos inicio al proyecto HVR en GCP.

Para el proyecto se implementarán 3 ambientes de trabajo con el objetivo de separar bien los roles asignados dentro del proyecto y proteger los datos.

**Ambiente de desarrollo(Dev)**

Aquí los desarrolladores podrán probar las modificaciones o integraciones sin el riesgo de afectar los datos reales del proyecto, realizando ajustes de códigos, configuraciones, etc.

**Ambiente QA**

En este ambiente se realizarán pruebas evaluando si los cambios realizados están bien y no generen errores a futuro.

**Ambiente Producción (Prod)**

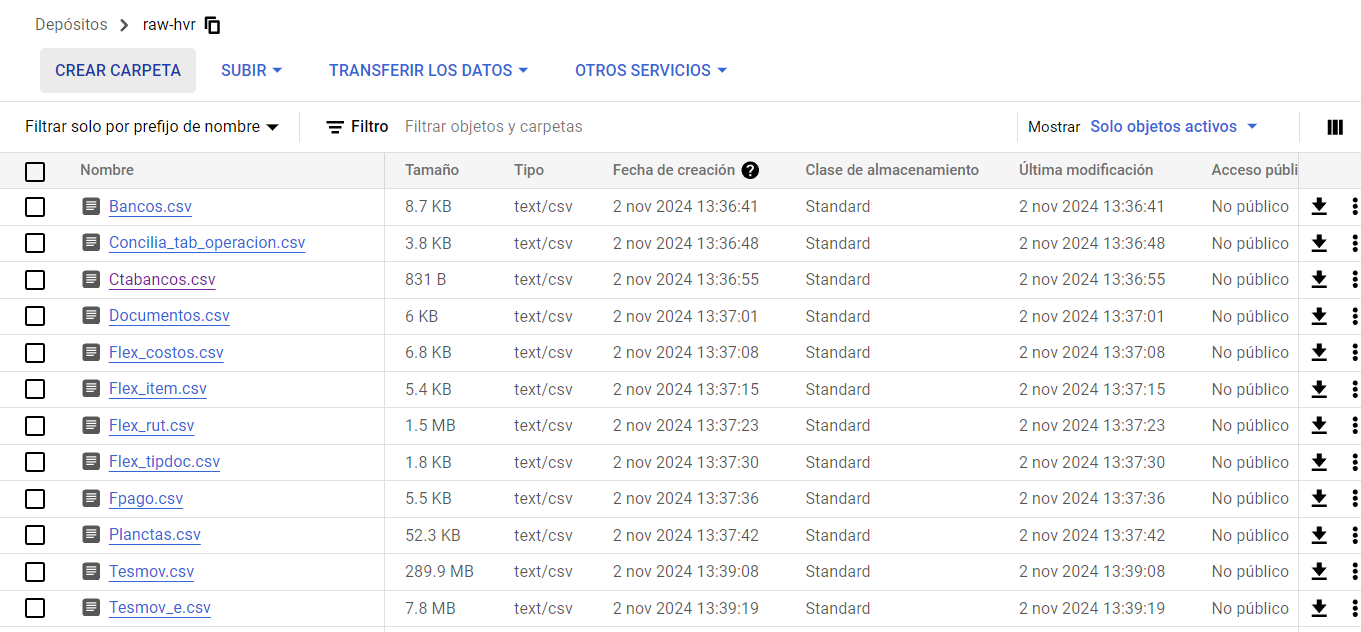
Es el paso final para implementar los cambios realizados en un código, una vez que hayan pasado por Dev y QA finalmente en producción se desplegarán las modificaciones sin riesgo de afectar los datos del proyecto.

Dentro de cada ambiente se crearán los buckets correspondientes a “raw\_data\_hvr", "error\_raw\_data\_hvr", “temp\_data\_hvr“, “clean\_data\_hvr” y “history\_data“ dentro de Cloud Storage

**Bucket raw\_data\_hvr\_dev**

En este bucket se almacenarán todos los datos que llegan desde el servidor de la empresa HVR, los archivos que se cargarán serán en formatos csv.

Al ser cargados automáticamente una cloud function realizará el proceso de limpieza y carga de datos a Big Query dentro del dataset Bronze.



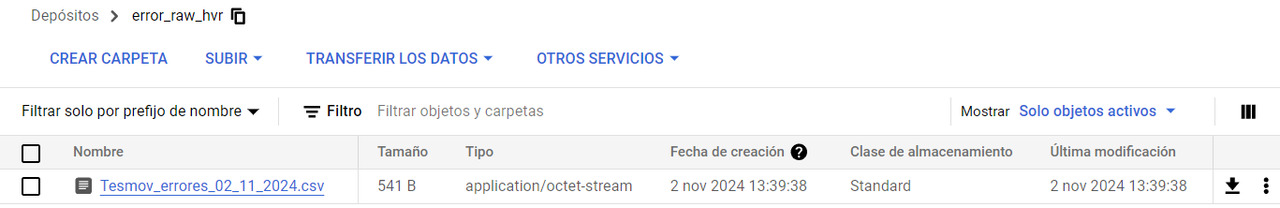
**Bucket error\_raw\_data**

En este bucket se almacenarán los datos erróneos de carga de datos.

Si un archivo tiene un error en el proceso de limpieza automáticamente se genera un archivo csv, el cual es guardado en este bucket con el nombre del documento más la fecha de ejecución.

Dentro de este archivo se registra la fila donde se debería encontrar el dato erróneo que no se pudo cargar a la tabla correspondiente.

En la imagen podemos visualizar que el archivo “Tesmov\_errores\_02\_11\_2024.csv” tiene errores registrados.

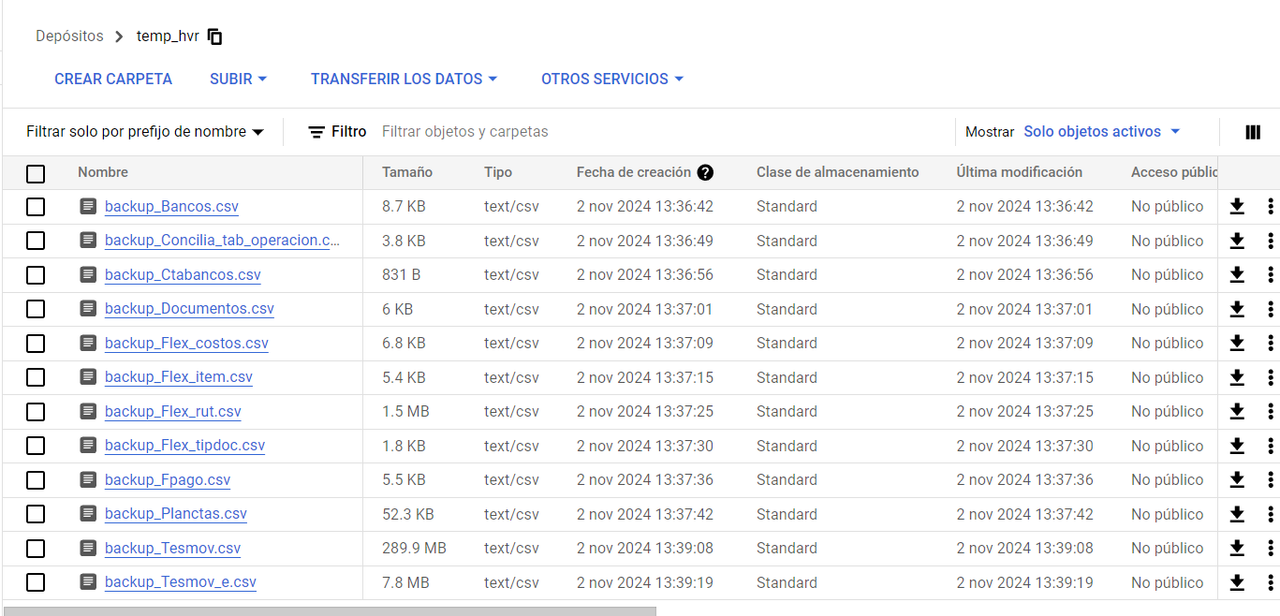


El error registrado se visualiza con los datos de la fila completa, el cual se debería analizar de forma manual para que se solucione y vuelva a pasar por todo el proceso de limpieza y carga de datos.



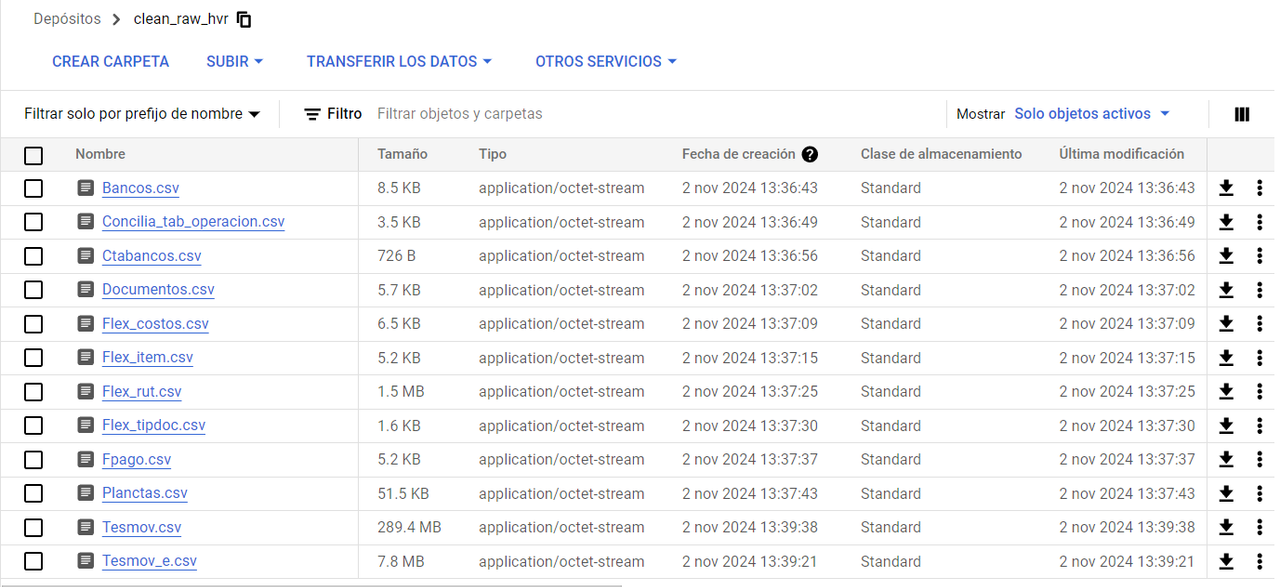
**Bucket temp\_data**

En este Bucket se guardan archivos temporales mientras se va realizando el proceso de limpieza y carga de datos, se crea una tabla temporal en el dataset Bronze “ctabancos\_bronze\_temp“, el cual se elimina cuando está listo el proceso de carga de datos quedando con el nombre “ctabancos\_bronze”.



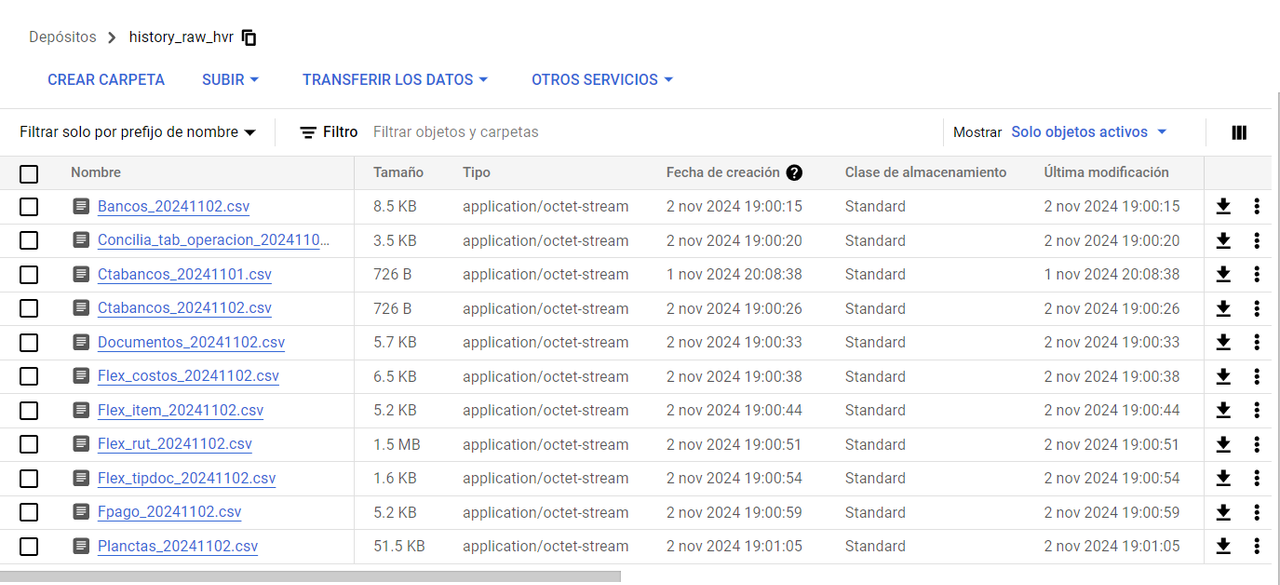
**Bucket clean\_data**

Se almacenan los archivos que hayan pasado por el proceso de limpieza en csv.



**Bucket history\_data**

Se almacenan los csv procesados y almacenados en el dataset bronze con el nombre y fecha de ejecución.

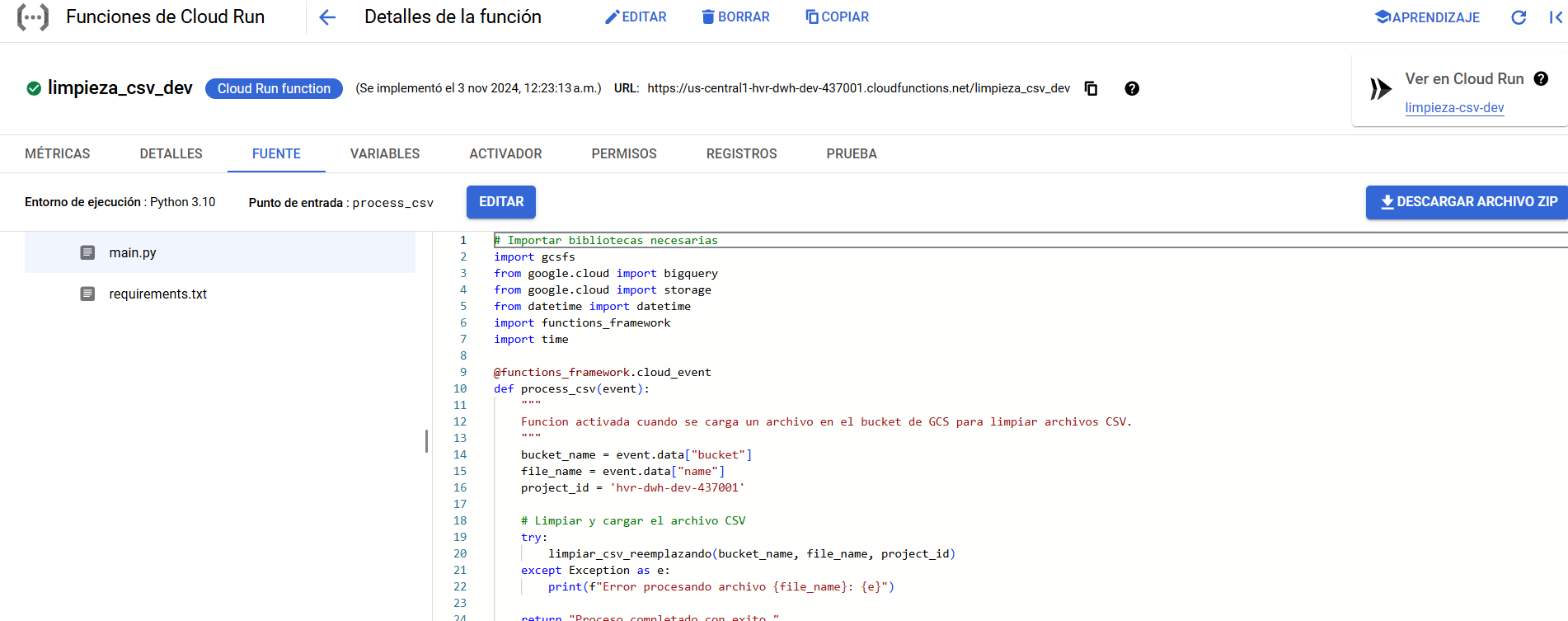


## **Cloud Functions para la carga de datos en BigQuery**

Para la carga de datos se crearon 3 Cloud Function, cada una con una función distinta, las cuales se explicarán a continuación:

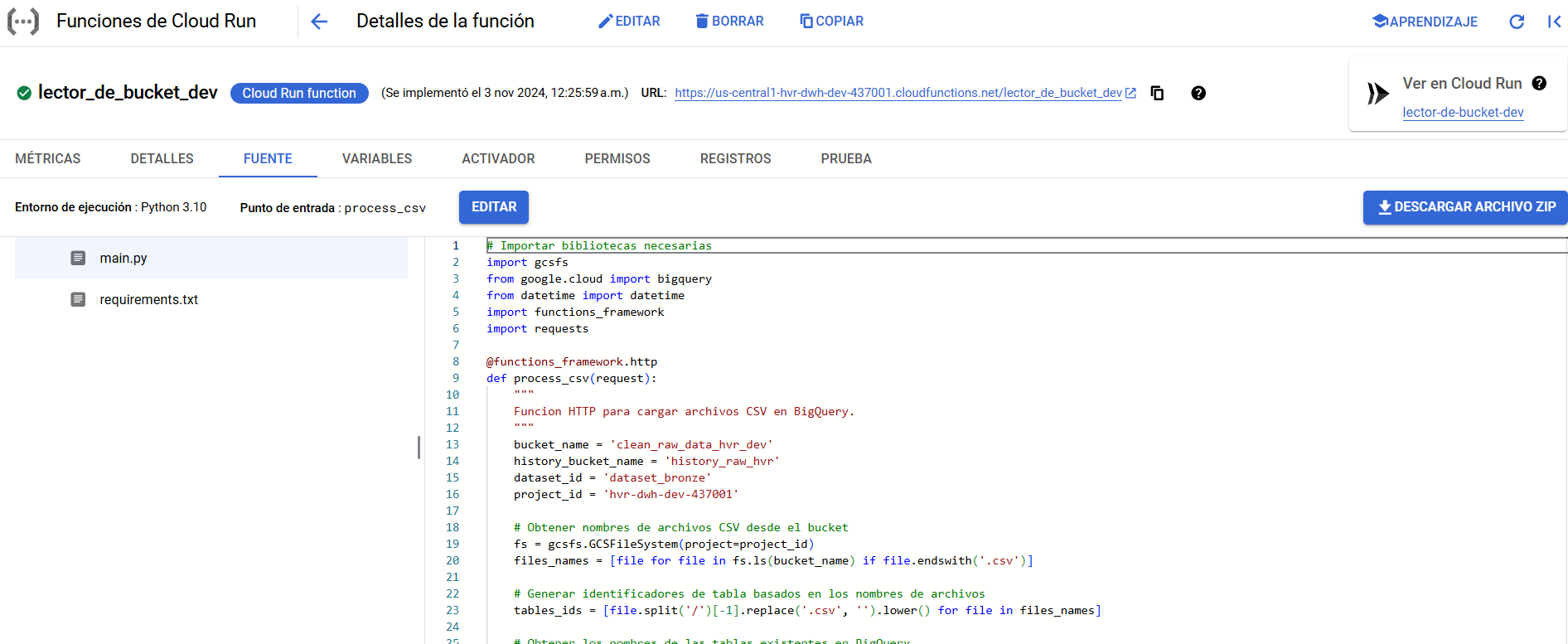
**Cloud Function limpieza\_csv\_dev**

Esta Cloud Function se activa automáticamente cuando un archivo CSV se carga en un Bucket específico de Google Cloud Storage (GCS). Su propósito principal es realizar una limpieza y validación de los datos en el archivo y guardar el archivo ya procesado en un Bucket final clean\_raw\_data\_hvr\_dev. Además, Almacena todas las filas inválidas detectadas que no cumpla con los requisitos de formato en un archivo separado alojado en el Bucket error\_raw\_data\_hvr\_dev.



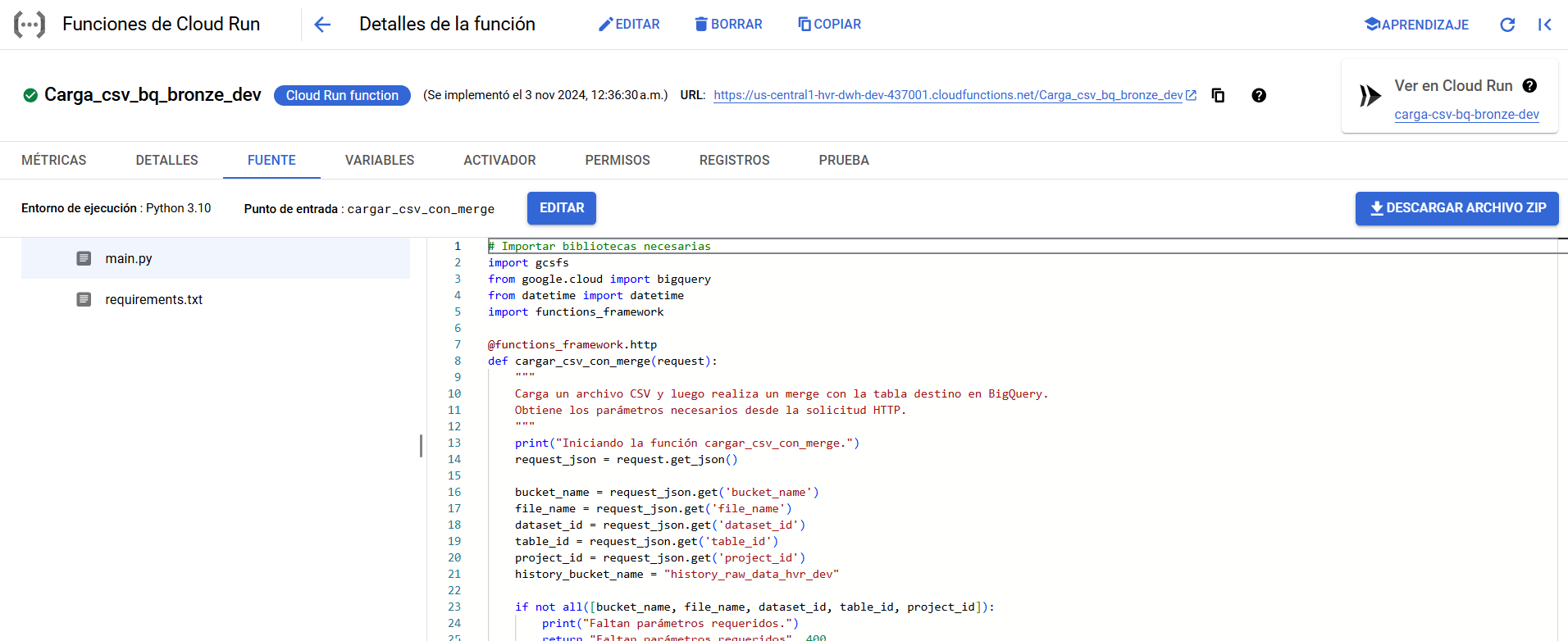
## Cloud Function lector\_de\_bucket\_dev

La Cloud Function es una función HTTP que se utiliza para obtener una lista de archivos en el bucket especificado, verifica si existe una tabla en BigQuery correspondiente para cada archivo, y si existe, procede a invocar la cloud function carga\_csv\_bq\_bronze con sus parámetros correspondientes, una vez procesado se guarda un archivo histórico del csv procesado en el bucket history\_raw\_hvr. Si no existe una tabla correspondiente, el archivo se omite.



## Cloud Function carga\_csv\_bq\_bronze\_dev

La función es una Cloud Function HTTP diseñada para cargar un archivo CSV desde Google Cloud Storage (GCS) en una tabla temporal de BigQuery y luego realizar una operación de merge con una tabla de destino en BigQuery. Esta función permite la actualización o inserción de registros en la tabla de destino en función de una columna de identificación.



## **Dataform para gestión de transformaciones en BigQuery**

El objetivo de implementar Dataform es automatizar las transformaciones de los datos en bigQuery, optimizando el flujo de trabajo del proceso de datos, estructurando el manejo de transformaciones en las capas Bronze, Silver y Gold, para mejorar la calidad y consistencia de los datos.

**Capas de almacenamiento:**

* Bronze: Capa de datos sin procesar.
* Silver: Capa de datos limpios, con transformaciones estándar.
* Gold: Capa de datos enriquecidos para análisis avanzado.

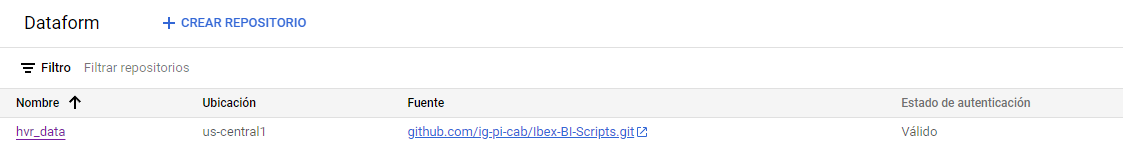
En Dataform se realizarán los siguiente:

* Se emplean scripts SQLX para gestionar las transformaciones en cada capa.
* Particiones: Se implementa la partición diaria en las tablas de la capa silver para optimizar el rendimiento en consultas.
* Monitoreo: Dataform genera reportes de ejecución y errores, facilitando la revisión de calidad y ajuste de procesos.

**Creación de repositorio**

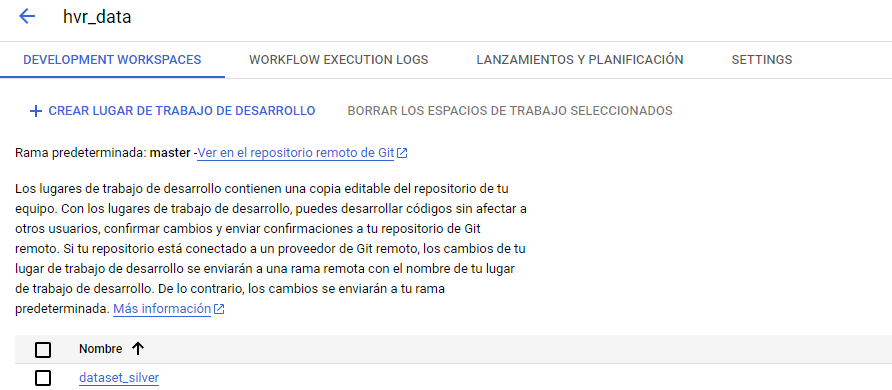
Se crea un repositorio dentro de Dataform

* Nombre: hvr\_data
* Ubicacion: us-central
* Fuente : [github.com/ig-pi-cab/Ibex-BI-Scripts.git](http://github.com/ig-pi-cab/Ibex-BI-Scripts.git)

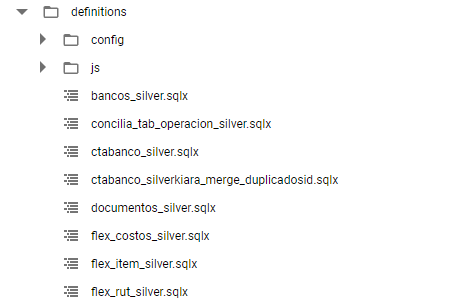


**Workspace**

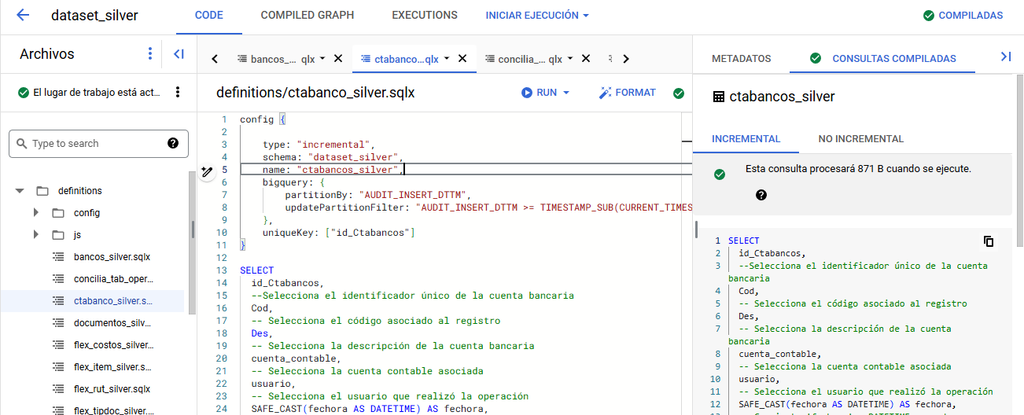
Se crea el workspace dataset\_silver dentro del repositorio hvr\_data, donde se realizarán las transformaciones de los datos.



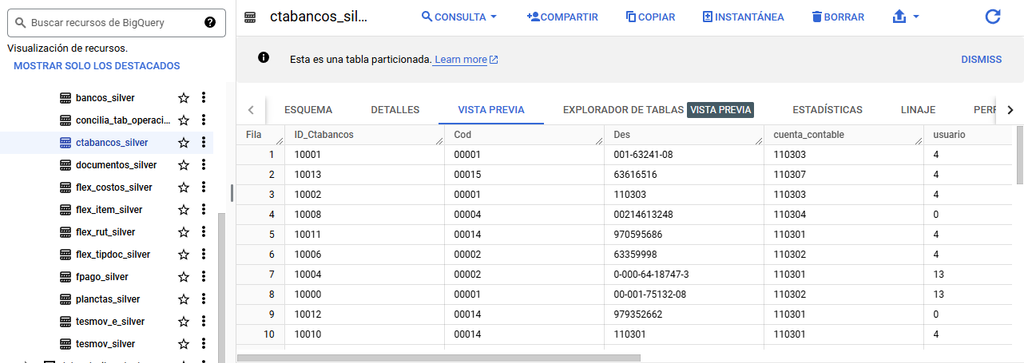
Creación de archivos en formato sqlx de las tablas silver



**Aprobación de datos transformados correctamente**

****

**Implementación de transformación de datos**

****

## 

## **Configuración de git en Dataform para gestión de transformaciones en BigQuery**

El objetivo es configurar GitHub en Dataform para facilitar la gestión y versionamiento de scripts de transformación de datos en BigQuery, permitiendo un desarrollo colaborativo y control de versiones.

Los beneficios que nos da es controlar cambios, nos permite revisiones de código, y asegura una sincronización efectiva en el equipo de desarrollo.

**Conexión con Github**

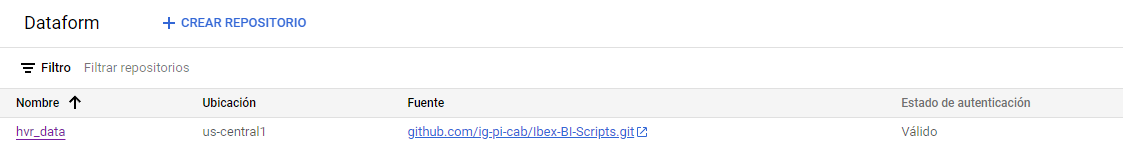
Se creó un repositorio en GitHub y se genera un token de acceso

* <https://github.com/ig-pi-cab/Ibex-BI-Scripts>

**Configuración de conexión en Dataform**

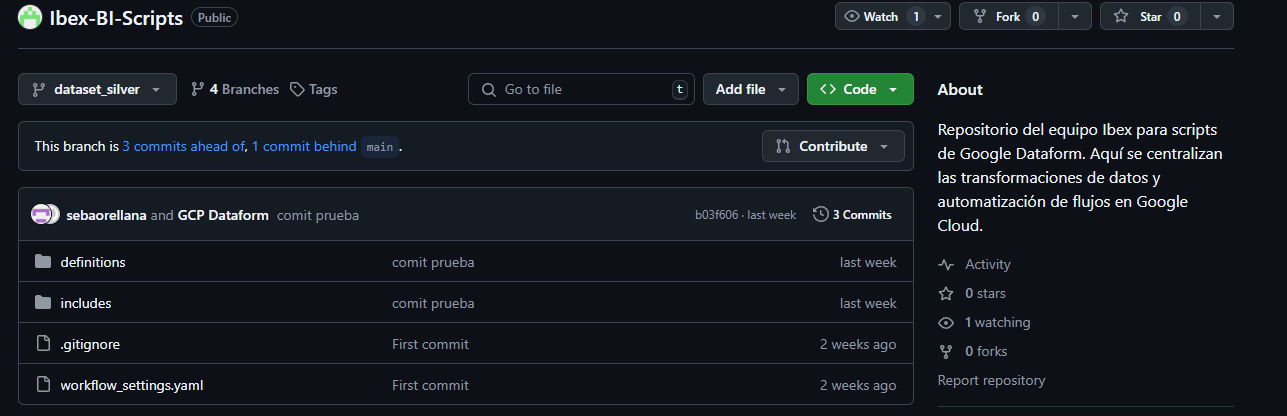
Se crea un repositorio dentro de Dataform y se da acceso con el token creado desde github

* Nombre: hvr\_data
* Ubicacion: us-central
* Fuente : [github.com/ig-pi-cab/Ibex-BI-Scripts.git](http://github.com/ig-pi-cab/Ibex-BI-Scripts.git)

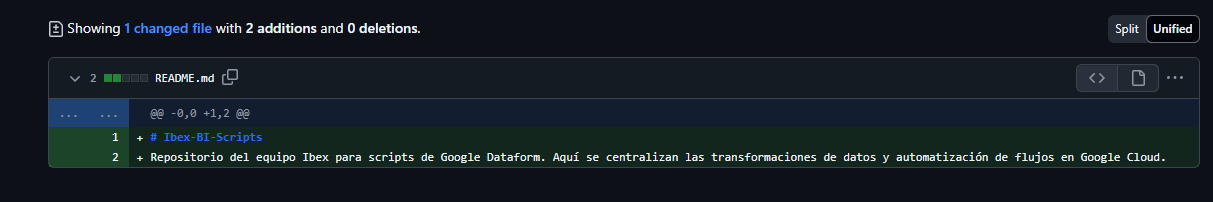


**Pruebas de commits**

Se realizan commits de prueba desde Dataform para compartir los scripts realizados



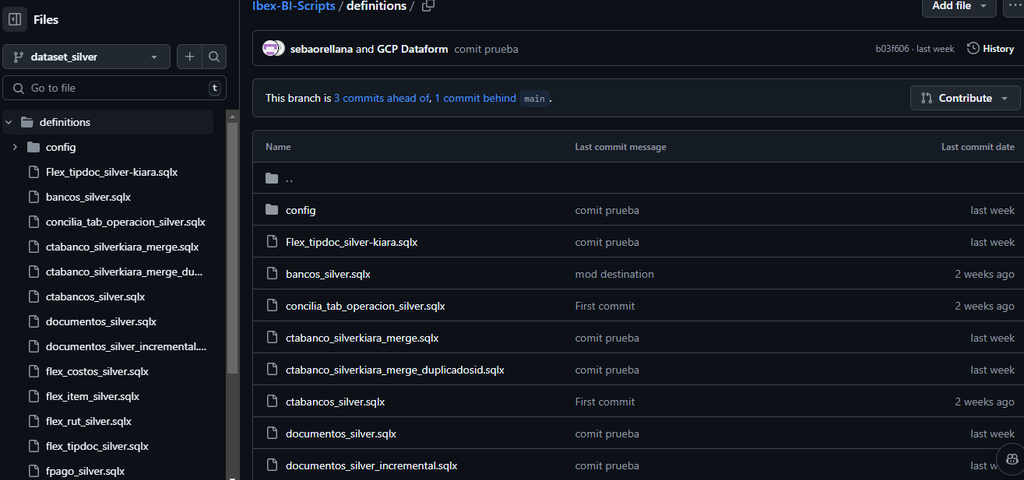
Creación de archivo README.md



Creación de Branch



**Registro de commits desde Branch dataset\_silver**

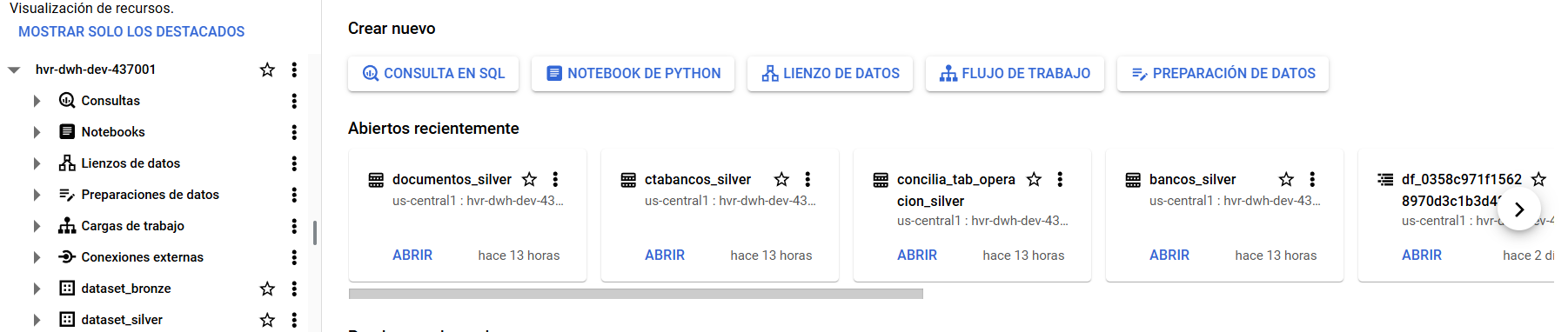
****

## BigQuery

Esta arquitectura organiza los datos en tres capas dentro de BigQuery:

* Bronze: Datos crudos o poco procesados directamente desde los sistemas de negocio.
* Silver: Datos transformados y limpiados, listos para análisis exploratorio y procesamiento intermedio.
* Gold: Datos altamente procesados y refinados para análisis avanzados y reportes.

Su función es facilitar la transición y limpieza de los datos a través de una estructura organizada en etapas para mejorar su calidad y utilidad.

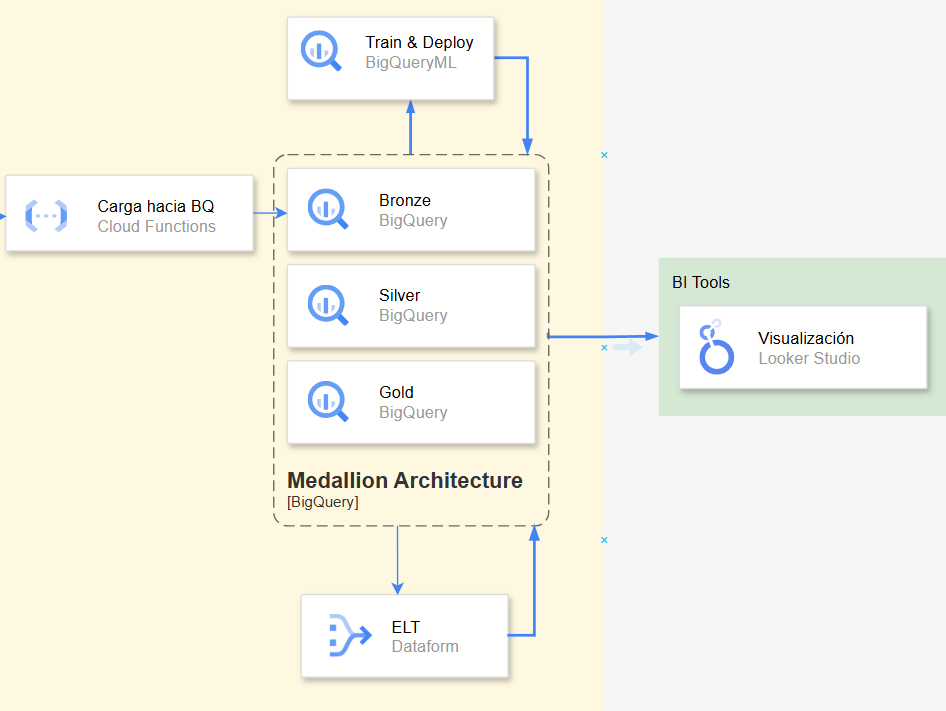


|  |  |
| --- | --- |

**BigQuery ML**

Servicio de machine learning de BigQuery que permite entrenar y desplegar modelos directamente dentro del Data Warehouse.

Su función es permitir la creación de modelos de aprendizaje automático sobre los datos en BigQuery, usando los datos de la capa “Gold”.



## 

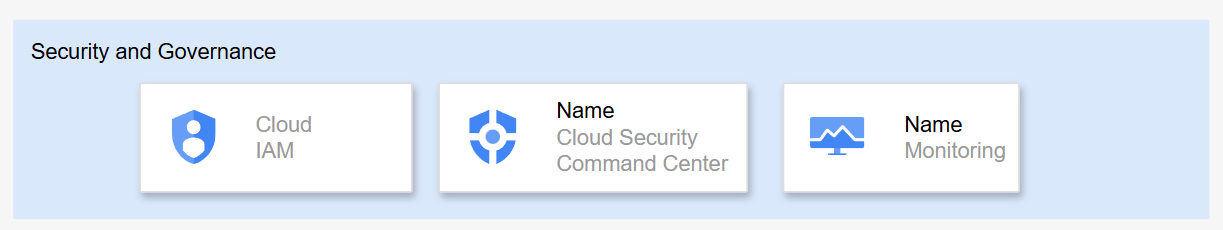
## Looker Studio

Herramienta de visualización de datos que permitirá a los usuarios acceder a reportes e insights basados en los datos procesados en BigQuery.

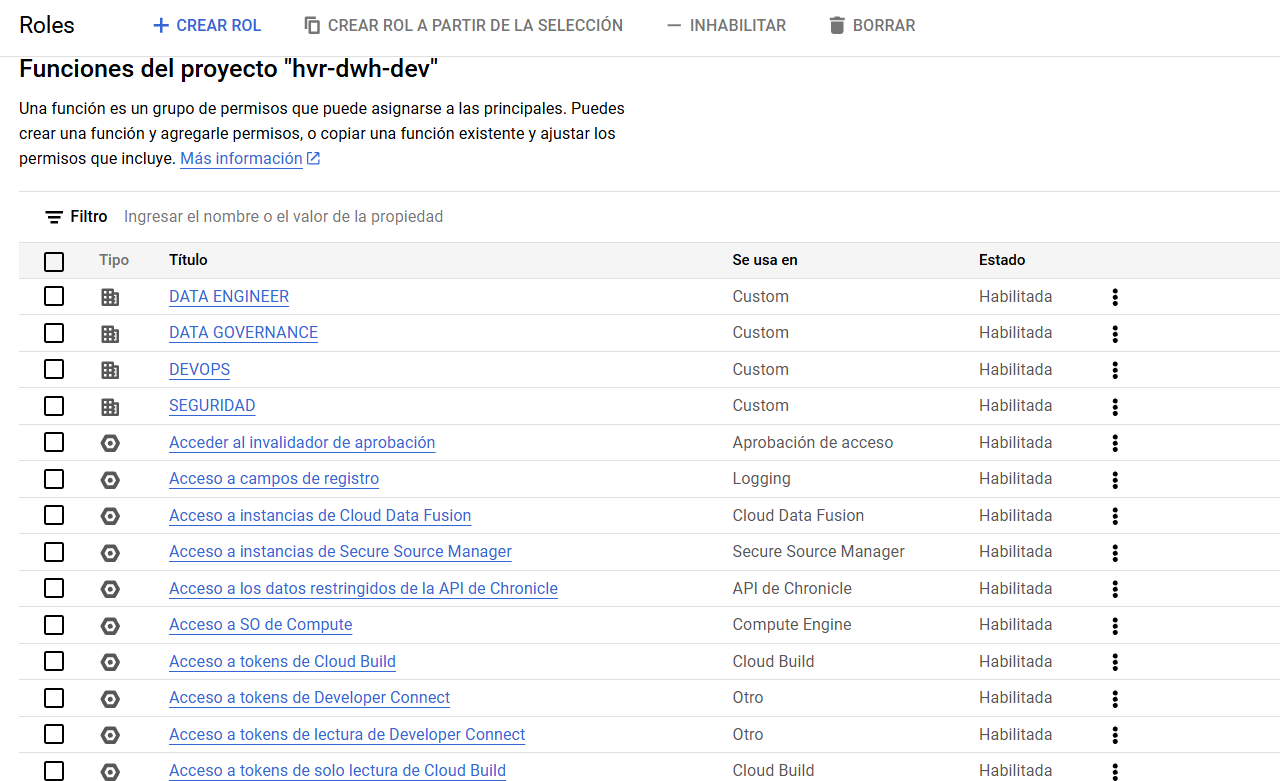
Su función es proveer un medio interactivo para explorar y analizar los datos finales, presentando dashboards y reportes a los usuarios.

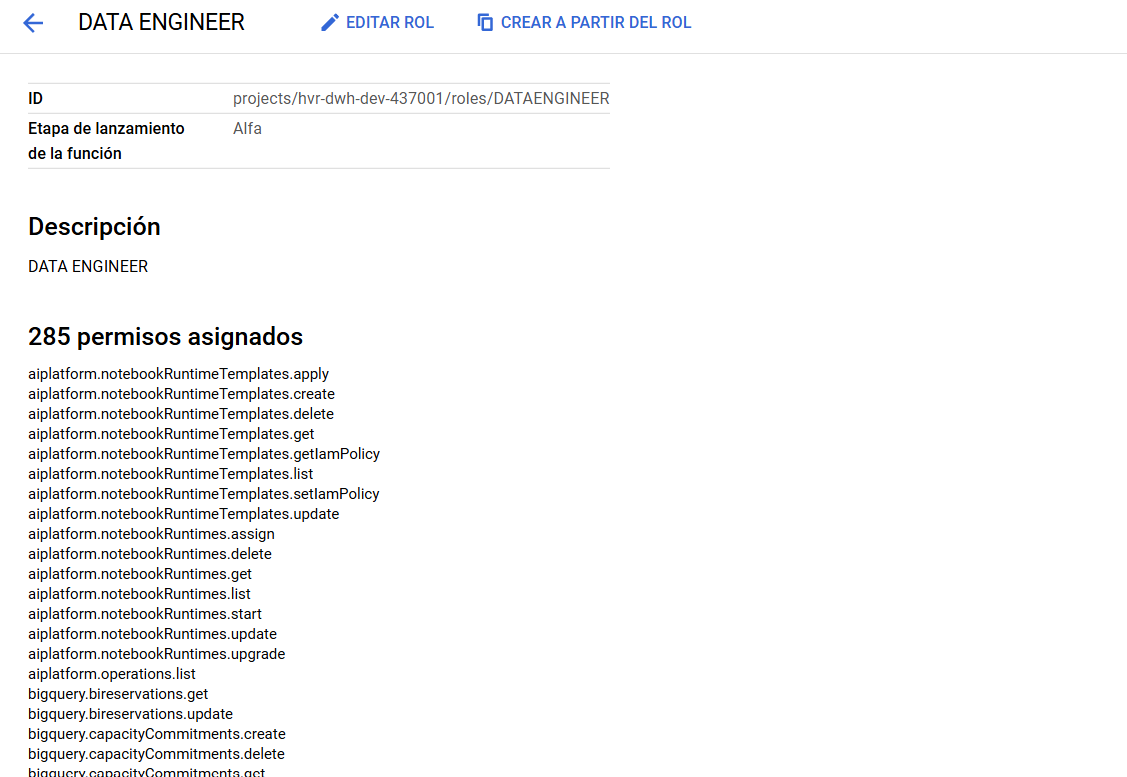
## Seguridad y Gobernanza

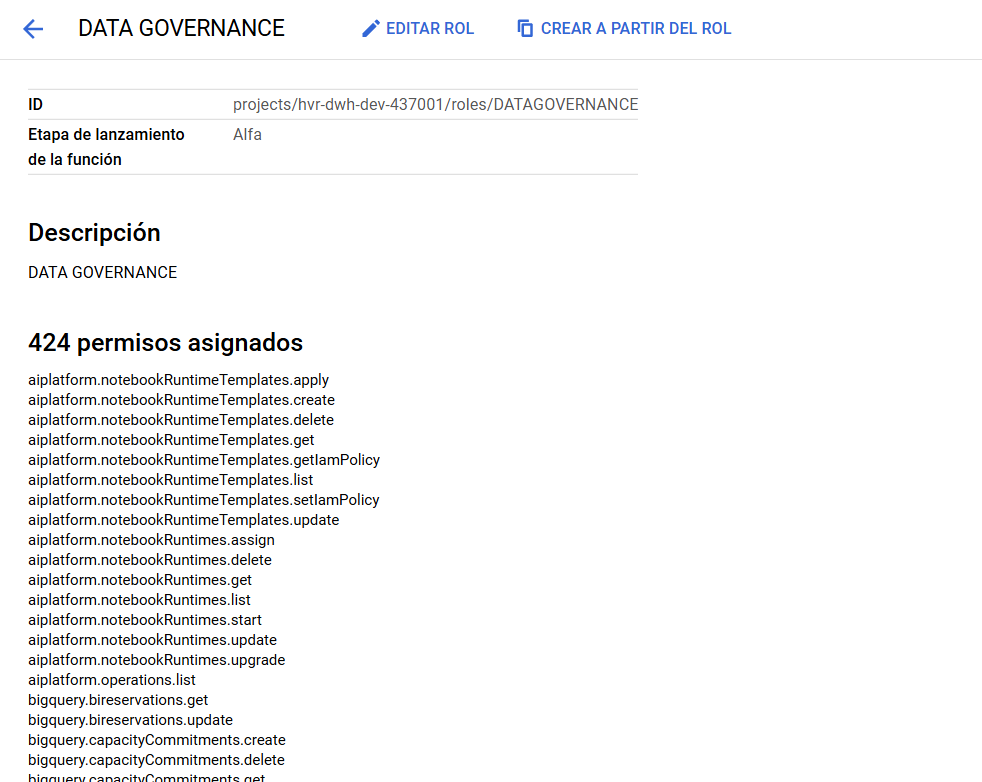
Para asegurar la integridad y privacidad de los datos, se integran varios servicios de seguridad y gobernanza en la arquitectura



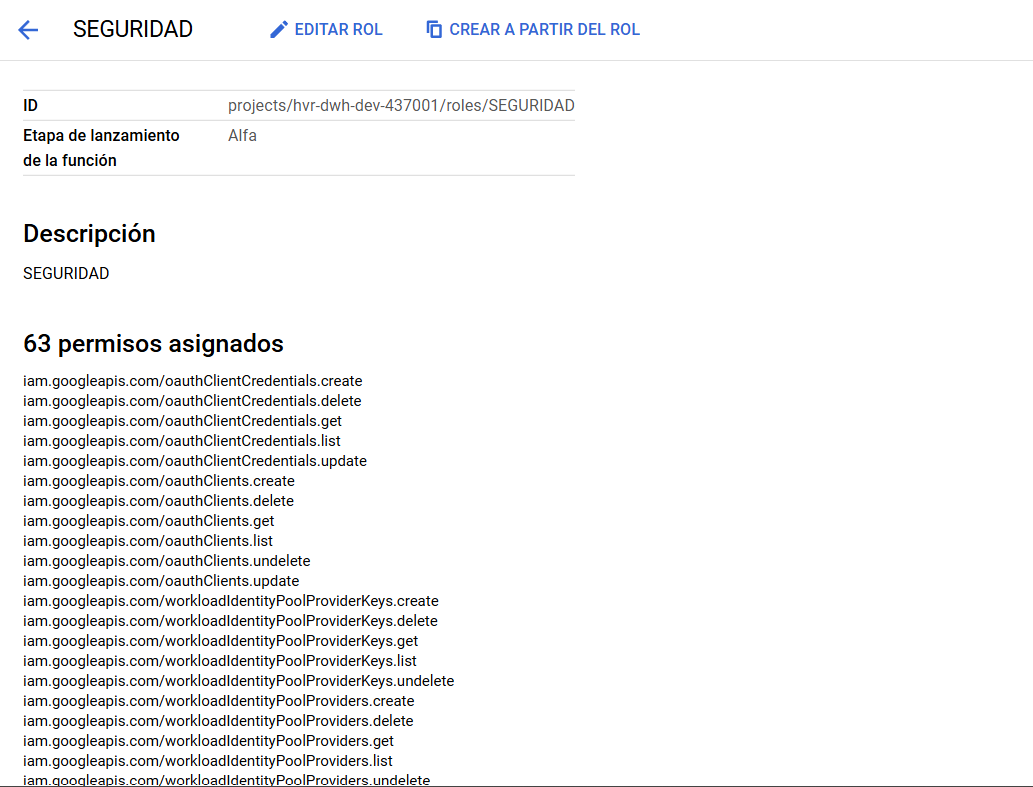
**Cloud IAM:** Control de acceso a nivel de identidad, asegurando que solo los usuarios y servicios autorizados puedan acceder a los datos.

****

****

****

****

****

**Cloud Security Command Center:** Plataforma de monitoreo de seguridad que ayuda a detectar y responder a amenazas.

**Monitoring:** Servicio de monitoreo para rastrear el rendimiento y estado de los componentes de la arquitectura, asegurando su funcionamiento óptimo.

**Documentación completa del proyecto aquí:** <https://hvrinformes.atlassian.net/wiki/spaces/F1PDT/pages/29491201/Proyecto+Data+Warehouse+HVR>