UNIVERSIDAD
DE ANTIOQUIA

# Diseño e implementación de sistema de monitoreo de ETLs con enfoque en transformaciones de Big Data dentro de GCP y AWS.

Valentina Botero Vivas

Informe de práctica como requisito para optar al título de:
Ingeniera Electrónica

Asesor Interno
Hernán Felipe García Arias
Profesor Universidad de Antioquia, PhD

Asesor Externo
Christian David Moreno Uribe
SM Data Engineer, Ingeniero Electrónico

Universidad de Antioquia
Facultad de Ingeniería, Departamento de Ingeniería Electrónica
Medellín, Colombia
2023

# 1 RESUMEN

En este informe de práctica, se expone el diseño e implementación de un software que permite el monitoreo a los procesos de ETL de Source Meridian. Esta herramienta permite la trazabilidad de los procesos y ayuda con la identificación de posibles mejoras a los procesos internos de la compañía en particular al equipo de datos.

Source Meridian brinda a sus clientes soluciones para la implementación de software de alta calidad y la prestación de servicios de asesoría e implementación, en particular uno de sus clientes principales requiere del uso de diversas herramientas usadas para la transformación de datos a gran escala. En los procesos internos se generan registros que están siendo desaprovechados para la toma de decisiones que llevan a la optimización de los mismos.

El siguiente trabajo presenta el desarrollo de un software de monitoreo de ETLs con enfoque en transformaciones de Big Data dentro de GCP y AWS, permitiendo al equipo técnico tener una visión profunda de los procesos internos. El resultado obtenido brinda un punto de partida para el análisis de la cantidad de recursos usados y comportamientos en los diferentes procesos activos en el equipo de datos.

**Palabras clave**: Logs, GCP, AWS, python, monitoreo, métricas.

# CONTENIDO

# LISTA DE FIGURAS

# 2 INTRODUCCIÓN

Las innovaciones tecnológicas día a día son más relevantes dentro de las empresas y han generado que la mayoría de estas migren sus actividades al ámbito tecnológico, ya que brindan herramientas capaces de detectar problemas y tomar decisiones dentro de los procesos de negocio sin importar la industria. Esto ha permitido el aumento en la cantidad de datos que se procesan y por ende el continuo cambio de los paradigmas de trabajo convencional. Los datos son la base de las tecnologías del mercado actual como redes sociales, inteligencia de las cosas y domótica [1].

Todo lo anterior dio origen al término Big Data, el cual describe grandes volúmenes de datos que por su gran complejidad requieren de herramientas novedosas para su gestión, análisis, procesamiento y almacenamiento. Este término le permitió a muchas empresas tener un punto de referencia frente al cómo identificar sus mapas de acción para la cantidad de datos que requieren procesar en busca de dar soluciones a sus clientes [2].

Source Meridian (SM) es una empresa de software que busca brindar soluciones para construir nuevos productos y sacar el máximo provecho de la información de sus clientes, por medio de diversos métodos de análisis y disponibilidad de la misma. Para esto, manejan tecnologías de transformación de datos a gran escala (Big Data). Actualmente las transformaciones que se dan dentro de la empresa están generando información que no es aprovechada para la toma de decisiones, en particular aquellas que podrían llevar a la optimización en la selección de hardware o el uso adecuado de herramientas de orquestación internas. Como primer paso en busca de mejorar la gestión y el control de la información, dentro de la empresa se quiere una herramienta que permita el monitoreo y visualización de este tipo de eventos.

La monitorización de los logs dentro de las empresas es el paso inicial para implementar buenas prácticas en la gestión de los logs, lo cual aporta diversos beneficios tanto de funcionamiento como de objetivos de negocio, entre las cuales se resaltan la detección de comportamientos inadecuados con el uso de hardware o software, la toma de decisión de negocio y la facilidad de acceso y explotación de datos [3].

Para esto se va a desarrollar un software capaz de extraer toda la información de los ficheros de registro (logs) generados en la empresa a partir de los procesos de transformación, modelarlos, almacenarlos y realizar análisis relacionados con la vida útil de los procesos, tales como, duración promedio de subprocesos, qué tipo de recursos se usan, cantidad de procesos exitosos y fallidos, etc. Esto con el fin de brindar apoyo en la toma de decisiones del equipo técnico de SM.

La metodología a emplear consiste en una serie de fases secuenciales. Inicialmente se hará la extracción y preparación de los logs. Luego, se estudia la naturaleza de los mismos para diseñar un modelo de datos que permita el aprovechamiento de ellos y posteriormente se implementará el software cuyos resultados serán presentados a modo de reporte.

# 3 OBJETIVOS

## 3.1 General

- Desarrollar un software para el monitoreo de eventos generados dentro de procesos de transformación de Big Data utilizando modelamiento de datos multivariados apoyado con herramientas para acceso de información en la nube para la empresa Source Meridian.

## 3.2 Específicos

- Realizar la extracción y depuración de los logs generados en los procesos de transformación que actualmente se encuentran en el almacén de datos de la empresa SM con el lenguaje de programación python para obtener información precisa y de calidad.
- Diseñar un modelo de datos que contenga la información generada por cada uno de los procesos de transformación utilizando métodos cuantitativos.
- Desarrollar un software que use la información generada a partir de la extracción de logs, la modele y construya métricas y visualizaciones que apoyen la toma de decisiones dentro del equipo de datos, aplicando buenas prácticas de programación.
- Evaluar la funcionalidad y el desempeño del software desarrollado para el monitoreo de eventos utilizando pruebas de aplicación con la participación del usuario en escenarios reales.

# 4 MARCO TEÓRICO

A continuación se presentan los conceptos relacionados con la temática a tratar en el presente informe.

## 4.1 BigData

Las herramientas analíticas y bases de datos convencionales no soportan el tratamiento de volúmenes de datos grandes, el Big Data permite a sus usuarios crear y manipular enormes repositorios, facilitando la toma de decisiones [4].

## 4.2 Inteligencia empresarial (Business Intelligence - BI)

En Google Cloud definen la inteligencia empresarial como: “Proceso que consiste en aprovechar el potencial de las personas y la tecnología para recoger y analizar los datos que deben usar las organizaciones en la toma de decisiones estratégicas” [5].

## 4.3 Almacén de datos (Data Warehouse)

Es un sistema de gestión que permite potencializar todos los procesos realizados en tareas de inteligencia empresarial o analíticas. Ayuda a la toma de decisiones, ya que fueron diseñados para realizar consultas en grandes cantidades de datos [6].

## 4.4 Amazon Web Services (AWS)

AWS es una plataforma en la nube que ofrece servicios de computación e interfaces de programación (API). Las principales categorías son almacenamiento, bases de datos, análisis y machine learning [7]. Los servicios relevantes para este proyecto son los listados a continuación.

### 4.4.1 S3

AWS define S3 como "un servicio de almacenamiento de objetos que ofrece escalabilidad, disponibilidad de datos, seguridad y rendimiento líderes en el sector". Brinda un servicio de almacenamiento y protección de datos para pequeños, medianos y grandes sectores. Permitiendo a las organizaciones una mejor gestión de acceso y optimización de costos [8].

### 4.4.2 Lambda

AWS define las lambdas como "es un servicio informático sin servidor y basado en eventos que le permite ejecutar código para prácticamente cualquier tipo de aplicación o servicio backend sin necesidad de aprovisionar o administrar servidores" [9].

### 4.4.3 CloudWatch

AWS define CloudWatch como un servicio que "recopila y visualiza los registros, las métricas y los datos de evento en tiempo real en paneles automatizados para simplificar la infraestructura y el mantenimiento de aplicaciones" [10].

## 4.5 Google Cloud Platform (GCP)

GCP es un conjunto de servicios de computación que proporciona herramientas en la nube para almacenamiento, análisis y machine learning en la nube. Proporciona la infraestructura, plataformas y entornos sin servidores, brindando entornos de trabajo que aumenten el rendimiento de los negocios [11]. El servicio relevante para este proyecto se expone a continuación.

### 4.5.1 Cloud Composer

Es un servicio que se basa en el proyecto de código abierto Apache Airflow desarrollado en Python, que brinda la completa organización del flujo de trabajo de manera programada, permitiendo la creación, programación, monitorización y administración centralizada de los mismos [12].

## 4.6 PostgreSQL (PG)

Es un gestor de bases de datos de código abierto con diversas funcionalidades diseñadas para facilitar a los usuarios la protección de datos, creación de aplicaciones y entornos sin importar el tamaño del conjunto de datos. El gestor admite características requeridas por el estándar SQL [13].

## 4.7 Extracción, transformación y carga (ETL)

La extracción, transformación y carga, ETL, por sus siglas en inglés, es el procesamiento de datos provenientes de diversas fuentes a un almacén de datos o repositorio. Se compone de tres etapas. Inicialmente, la extracción consiste en la copia de datos crudos desde una fuente a un espacio temporal para ser procesados en la etapa de transformación. El proceso depende del caso de estudio específico, dentro de los comunes están: cifrado, filtrado, formateo, protección, cálculos y comparaciones. Finalmente, los datos transformados son llevados al repositorio destino. Idealmente el proceso de carga está automatizado, buscando minimizar el tráfico en los sistemas [14].

## 4.8 Logs

También llamados registros, son un archivo de texto que almacena información reportada por un sistema operativo o aplicación. En algunos casos con ellos es posible identificar que está pasando indicando fecha y hora. Son muy importantes, ya que permiten identificar patrones, gestionar y tener control del acceso a los recursos. Pueden volverse esenciales dentro de una compañía, ya que brindan herramientas para el análisis de un sistema a un nivel que el usuario no puede identificar, en términos de seguridad, uso, rendimiento y detección de fallos. Cada aplicación o sistema operativo cuenta con su propio formato para el almacenamiento de la información [15].

### 4.8.1 Python logging library

El módulo logging de python contiene clases y funciones que permiten implementar sistemas flexibles de registros de eventos. Los eventos que se pueden monitorear con este módulo tienen cinco niveles que indican la severidad de los mismos: DEBUG, INFO, WARNING, ERROR y CRITICAL [16].

## 4.9 Modelo de datos

Un modelo de datos es la representación gráfica que define a sistemas de administración de información en cualquier campo. Permite crear una vista unificada de todos los datos de la compañía. Amazon expresa que el modelo "esboza los datos que recoge la empresa, la relación entre los distintos conjuntos de datos y los métodos que se usarán para almacenarlos y analizarlos" [17].

### 4.9.1 Modelo dimensional

La Ph.D, Elizabeth León Guzmán define el modelamiento dimensional como "una técnica para diseñar un modelo lógico de la bodega de datos, que permite alto rendimiento al momento de acceder a los datos (orientado a consultas)" [18] . Esta técnica se compone de:

- Hechos: Es una operación o actividad que ocurre en el tiempo. Los hechos son aquellas medidas que brindan información acerca del negocio o tema de estudio.
- Medidas: Son valores comúnmente numéricos que describen el hecho. Ayudan a medir el desempeño. Existen medidas básicas que se obtienen directamente del almacén de datos y las medidas derivadas que se construyen a partir de las básicas.
- Dimensiones: Son el contexto en el que se están analizando las medidas. Estas son las que definen los niveles de análisis dentro del modelo y son las que proveen el orden de la información.

- Atributos: Son todos los elementos dentro de una dimensión, todos deben tener el mismo nivel lógico.Estos definen el quién, por qué, y para qué del modelo.
- Relaciones: Los atributos en diferentes dimensiones se relacionan mediante las medidas del negocio.

# 5 METODOLOGÍA

## 5.1 Información de procesos

Dentro de la empresa Source Meridian, específicamente el equipo de datos Datacore maneja diversos servicios para el desarrollo de su propia arquitectura y utilidades. Un ejemplo de estos servicios se muestran en la Figura 1, los cuales son: AWS Lambdas para la ejecución de lógica de ejecución efímera en los procesos de ETL, S3 es el principal sumidero de datos, CloudWatch para recopilar los logs en las ejecuciones, PG como motor de bases de datos y Airflow actúa como orquestador entre todas las partes.

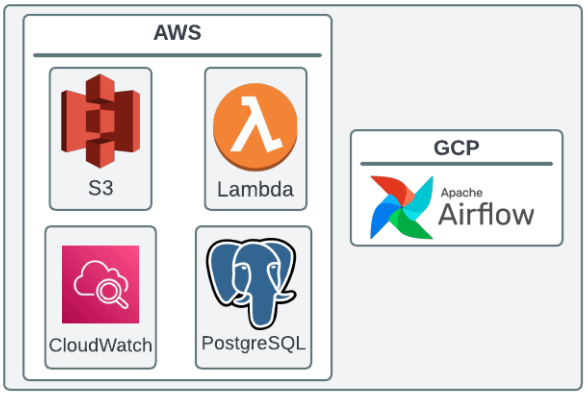


Figura 1: Servicios requeridos

Las distintas ETLs que se corren en la empresa generan múltiples registros, los cuales se almacenan en CloudWatch sin ningún tipo de formato o monitoreo como se ve en la Figura 2.

```
START RequestId: 7a115959-fcd4-4af0-be82-e67037ec6e61 Version: $LATEST
[ERROR] Runtime.ImportModuleError: Unable to import module 'lambda_function': No module named 'lambda_function' Traceback (most recent call l…
END RequestId: 7a115959-fcd4-4af0-be82-e67037ec6e61
REPORT RequestId: 7a115959-fcd4-4af0-be82-e67037ec6e61 Duration: 1.78 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 38 MB Ini…
```

Figura 2: Logs originales en CloudWatch.

## 5.2 Diseño de solución

Al entender la necesidad y profundizar en cómo funcionaba el proceso de logging, se plantearon una serie de pasos para diseñar una solución que no afectara el funcionamiento normal de los procesos.

1. Definir un formato general para los logs, el cual se adaptara a las diferentes herramientas o servicios.

2. Identificación de los parámetros representativos para ser monitoreados.

3. Desarrollo de una funcionalidad genérica que pudiera ser compartida entre los diferentes procesos y que permitiera el modelado de los logs.

4. Centralización de los logs: Establecer la mejor estrategia para extraer y centralizar los registros que cumplan el formato configurado en las diferentes herramientas para su posterior uso como fuente de información.

5. Modelado de Datos: Diseñar e implementar un modelo dimensional que facilite la visualización y comprensión de los procesos.

6. Reporte de información: Generar reportes que brinden información acerca de los diferentes procesos en términos de tiempos de ejecución y costos.

## 5.3 Desarrollo

### 5.3.1 Definición e implementación de formato

Se eligió un formato que directamente o por medio de transformaciones brindaran información valiosa para la toma de decisiones en el uso de recursos e historial de las herramientas. El formato elegido consta de siete variables como se ve en la figura 3.

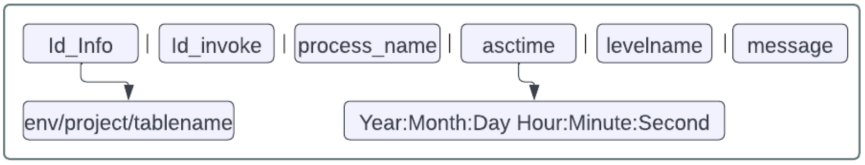


Figura 3: Formato para registros.

- Identificador de información (Id_Info): Identificador que permite saber dónde y qué se está trabajando, se compone de 3 parámetros:
  - Entorno de trabajo: Desarrollo o producción.
  - Proyecto: Nombre del proyecto sobre el que se está trabajando.
  - Tabla: Nombre de la tabla a la que se le están haciendo transformaciones.
- Identificador de llamado (Id_Invoke): Contiene un número de identificación que se genera automáticamente en AWS, este será útil para identificar cuáles registros son producto de un mismo llamado.
- Nombre del proceso (process_name): Contiene el nombre del proceso que se está usando en el momento.

- Tiempo (asctime): Contiene toda la información relacionada con el tiempo en el que se está ejecutando el proceso (Año:Mes:Día Hora:Minuto:Segundo) en estándar UTC.
- Nivel (levelname): Los registros se pueden clasificar de acuerdo a su nivel de severidad, para este caso se desea almacenar los registros desde el nivel de información. Por esto solo se verán logs de eventos con severidad de información, error, advertencia y crítico.
- Mensaje (message): Este campo lleva información referente a lo sucedido en el proceso para el registro específico.El mensaje se configura a lo largo del código fuente de la herramienta.

### 5.3.2 Configuración de logger

Con la librería de python logging se desarrolló un módulo que contiene la lógica necesaria para establecer el formato definido previamente en las herramientas.

```python
import logging


def get_logger(name, id_info, id_invoke):
    root = logging.getLogger()
    if root.handlers:
        for handler in root.handlers:
            root.removeHandler(handler)
    logging.basicConfig(level=logging.INFO)
    logger = logging.getLogger(name)  # Create a custom logger
    logger.propagate = False
    logger.handlers = []
    extra = {"id_info": id_info, "id_invoke": id_invoke, "process_name": name}
    formatter = logging.Formatter(
        "%(id_info)s | %(id_invoke)s | %(process_name)s | %(asctime)s | %(levelname)s | %(message)s",
        datefmt="%Y-%m-%d %H:%M:%S",
    )
    c_handler = logging.StreamHandler()
    c_handler.setLevel(logging.INFO)
    c_handler.setFormatter(formatter)
    logger.addHandler(c_handler)
    logger = logging.LoggerAdapter(logger, extra)
    return logger
```

Figura 4: Configuración logger.

En la figura 4 se muestra lo desarrollado. La función recibe tres parámetros:

- Nombre (name): Especifica el nombre del proceso que se está trabajando.
- Identificador de información (id_info): Contiene el entorno, proyecto y tabla.
- Identificador de llamado (id_invoke).

Después, se define el logger a nivel de INFO por tanto los eventos tipo DEBUG no quedarán registrados. Se tomó esta decisión porque este tipo de eventos generarían demasiados registros que no brindan información valiosa para la toma de decisiones. Teniendo en cuenta que muchas de las herramientas funcionan en la nube se estableció el parámetro de tiempo en estándar UTC.

AWS separa los registros por grupos (logs groups) y son almacenados en CloudWatch según la herramienta a la que pertenezcan.

Al hacer pequeñas pruebas para evaluar la configuración y el formato elegido, fue necesario agregar un parámetro al logger para evitar registros duplicados en CloudWatch. El parámetro de propagación (propagate) en la configuración permite que el controlador local del proceso no herede los registros a los controladores de nivel superior.

**Lambda layer**

Para disponibilizar la configuración del logger en AWS, se creó una capa para el servicio de lambdas. La capa contiene el módulo mostrado en la figura 4 para generar los logs en el formato deseado y es compatible con intérpretes de Python versiones 3.7, 3.8 y 3.9.

### 5.3.3 Extracción, filtrado y centralización de registros.

Se desarrolló una ETL que extrae los datos de diversas fuentes, los filtra y los guarda centralizados en un bucket de s3.

**Extracción** Por medio de una lambda se extraen los diferentes grupos de CloudWatch para un bucket de s3 como se ve en la figura 5.

```python
import boto3
from modules.modules import get_time


def lambda_handler(event, context):
    client = boto3.client("logs", region_name="us-east-1")
    extraction_date = event.get("extraction_date")
    log_group = event.get("log_group")
    env = event.get("env", "dev")

    if not extraction_date or not log_group:
        raise ValueError(
            "Missing log_group or extraction_date log_group: %s extraction_date: %s "
            % (log_group, extraction_date)
        )

    ti = event.get("start_timestamp")
    tf = event.get("final_timestamp")

    if ti and tf:
        ti = int(ti) * 1000
        tf = int(tf) * 1000
    else:
        ti = get_time(month=1)
        tf = get_time()

    response = client.create_export_task(
        taskName="Export Cloudwatch logs",
        logGroupName=f"/aws/lambda/{log_group}",
        fromTime=int(ti),
        to=int(tf),
        destination="purplelab-datacore-landing-zone-us-east-1",
        destinationPrefix=f"logs/{env}/rdl/lambda/source_dt={extraction_date}",
    )
    return response
```

```python
import datetime


def get_time(month=0):
    today = datetime.datetime.now(datetime.timezone.utc)
    process_month = today.month - month
    process_year = today.year
    if process_month == 0:
        process_month = 12
        process_year -= 1
    process_ts = (
        datetime.datetime.timestamp(
            datetime.datetime(
                process_year,
                process_month,
                today.day,
                0,
                0,
                0,
                tzinfo=datetime.timezone.utc,
            )
        )
        * 1000
    )
    return process_ts
```

Figura 5: Lambda de extracción.

Esta lambda se encarga de extraer los logs de CloudWatch recibiendo como parámetro el grupo de logs al que pertenece y una fecha de extracción para almacenarlos en un bucket de s3. Se extraen todos los logs generados en un mes, previo a la fecha que se ingrese, si no se recibe el parámetro se utiliza la hora y día local. En el bucket los registros se guardan con un identificador de tiempo para llevar una mejor trazabilidad. Este proceso se muestra en la Figura 6.

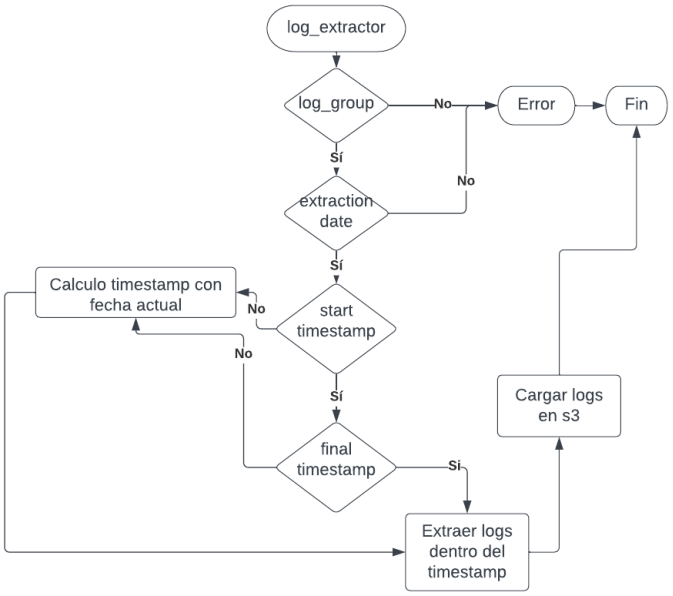


Figura 6: Diagrama de flujo lambda extracción.

**Filtrado** Se configura una lambda para seleccionar los logs que cumplen con el formato previamente establecido que están almacenados en el bucket de s3, que sigue el flujo mostrado en la Figura 7. Para esto se hace un filtrado con la expresión regular basada en reglas sintácticas que permita describir la secuencia de caracteres del formato:

(^[0-9\-T:. ]*[\w\/ ]+[\|]+[ \w\d\-]+[\|][ \w\-]+[\|]+[0-9\-:. ]+[\|]+[ \w\-]+[\|]+.*)

Se le adiciona un primer grupo de captura ya que el almacenamiento en CloudWatch tiene una columna extra que define el tiempo de generación de los logs llamada timestamp que tiene el formato YYYY-MM-DDTH:M:S.

Los registros que cumplan con la expresión se escriben en un archivo formato csv en el siguiente orden:

**stream_id, entorno, proyecto, tabla, invoke_id, nombre_proceso, tiempo, dia, evento, mensaje**

El stream_id es un parámetro dentro de CloudWatch que permite identificar todos los registros que comparten la misma fuente. Por otro lado, el invoke_id son todos los registros que pertenecen a una misma ejecución.

Además, para almacenar parámetros relevantes en temas de costos por ejecución, se seleccionan los registros que inicien con 'REPORT:'. Estos registros son generados por AWS y contienen varios parámetros. Primero, el Request Id que es el identificador de invocación. Este coincide con el id que está en el formato de los logs definido anteriormente. También, la duración del proceso y la duración factura en milisegundos(ms). Además, brinda información acerca de la memoria. Tanto la configurada para el proceso como la máxima usada en la invocación específica, ambas en MegaBytes. Por último, la duración en ms de lo que le tomó al servicio inicializarse para realizar sus funciones.

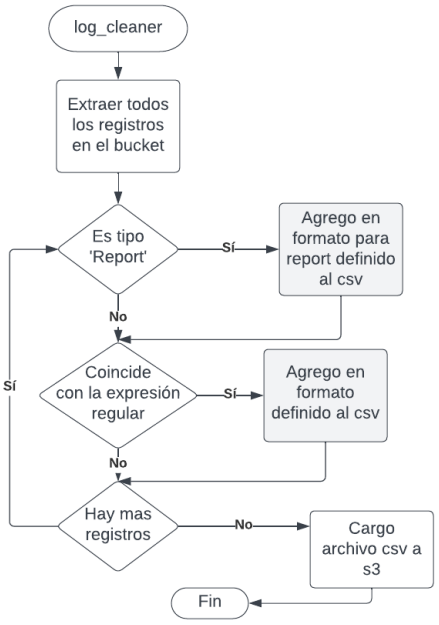


Figura 7: Diagrama de flujo lambda limpieza.

Ahora, los registros que coinciden con el filtrado del reporte generado por AWS son escritos en el mismo archivo csv generado en la selección de los registros por expresión, pero, de la siguiente manera:

**None, None, None, None, RequestId, report, duración, facturación, memoria, memoria usada**

Fue necesario agregar los campos a la izquierda para conservar el orden entre los dos tipos de registro, así todos los identificadores quedan en la misma columna y se facilita la búsqueda.

**Carga** Durante las etapas de procesamiento de registros se realizan cargas de archivos a s3 que siguen el flujo y los diferentes procesos. En cada etapa se almacenan los archivos indicando el día de ejecución de la tarea.

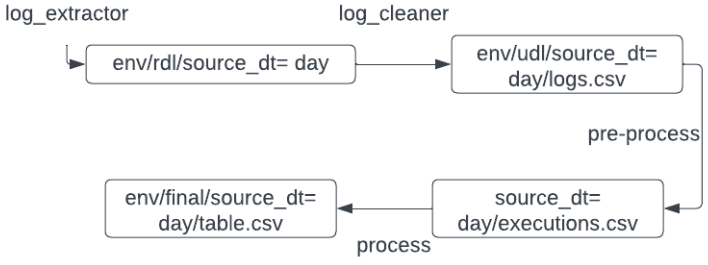


Figura 8: Diagrama de flujo carga.

Como se ve en la figura 8 inicialmente la lambda de extracción carga los archivos en la carpeta rdl según el entorno de la herramienta (dev o prod) y la fecha actual (day) de extracción. Luego,

la lambda de limpieza parte de esta ubicación, selecciona los registros a trabajar y carga los datos identificando la carpeta udl. Partiendo de este archivo se hace un pre-procesamiento de los datos, el cual se describe detalladamente en la siguiente sección y se genera el archivo que contiene la tabla de hechos (executions.csv), este se almacena según el día de extracción. Finalmente, se parte de este archivo y se cargan en la carpeta env/final/source_dt=day/ los archivos que contienen los reportes encontrados.

### 5.3.4 Preprocesamiento de datos

Para iniciar con el modelado se hizo un preprocesamiento de los datos que define la tabla de hechos. Las columnas elegidas para la tabla son:

- Id de invocación: Es un string de 32 caracteres entre letras y números que identifica la invocación del proceso en AWS.
- Entorno, Proyecto y Tabla: Brinda información sobre el entorno de trabajo, a qué proyecto pertenecen y cuales datos se están moviendo en esa ejecución.
- Nombre del proceso: El nombre de la herramienta se está invocando en la ejecución.
- Tiempo de inicio: 'YYYY-MM-DD HH:MM:SS' del momento de la ejecución en estándar UTC.
- Día: Por facilidad de búsqueda se extrae el año, mes y día del tiempo de inicio (YYYY-MM-DD).
- Estado: Indica si se presentó una excepción o fue un proceso exitoso. Puede ser SUCCESS o ERROR.
- Duración: Duración de la ejecución en segundos.
- Duración Facturada: Cantidad de segundos facturados por la ejecución.
- Memoria usada en la ejecución en [MB].

El procesamiento realizado consiste en identificar todos los registros que tienen el mismo identificador de invocación. Una vez se tienen definidos los registros que pertenecen a la misma ejecución se determina cual es el registro con la fecha menor para el tiempo de inicio y por medio del evento se determina el estado del proceso. El resto de los campos deseados se extraen directamente del archivo almacenado en el bucket de s3 en la etapa de carga. Posteriormente, todos estos datos se escriben en otro archivo csv en el orden deseado.

### 5.3.5 Modelado

Partiendo de la naturaleza de los datos recogidos por el logger en las diferentes ETL's y teniendo en cuenta la estructura definida en el preprocesamiento de datos, se establecen 4 dimensiones en las cuales los datos recogidos hasta el momento toman sentido:

**Proceso** La dimensión de procesos esta conformada por información puntual de las diversas ETLs existentes en el equipo. Permite complementar e identificar los recursos necesarios y por tanto el costo por ejecución. Los campos definidos para cada proceso son:

- Nombre (name): Nombre del proceso en la nube.
- Memoria (memory_MB): Memoria máxima configurada en Megabytes (MB).
- Almacenamiento (storage_MB): Almacenamiento configurado en MB.
- Tiempo máximo (time_out_min): Tiempo máximo de ejecución configurado en minutos (min).
- Entorno (environment): Es el entorno al cual pertenece la ETL. Comúnmente, un mismo proceso tiene su versión en desarrollo y en producción.
- Lenguaje (runtime): El lenguaje de ejecución en el cual se correrá el proceso configurado.
- Tipo (type): Servicio en la nube usado.
- Costo (cost_usd_ms): El costo que tiene el proceso en dólares por milisegundo (ms) según las características previamente configuradas.

Esta dimensión tiene una relación muchos a uno con la tabla de hechos del modelo. Es decir, muchos registros en la columna process_name coinciden con un valor activo registrado en la dimensión de procesos.

**Proyecto** Cada proyecto está gestionado y centralizado en un repositorio específico y además, idealmente su ejecución debe estar programada y automatizada. Para tener toda esta información los campos que componen la dimensión proyecto son:

- Nombre (name): Nombre del proyecto.
- Repositorio (repository): Nombre del repositorio en el cual se gestiona el proyecto.
- Tipo de ejecución (run): Su ejecución es manual o por automatizada por medio de airflow.
- Día (day): Día del mes en el cual se ejecuta el proyecto.
- Periodicidad (periodicity): Cada cuanto se ejecuta. Mensual, trimestral, anual,etc.

Esta dimensión tiene una relación muchos a uno con la tabla de hechos del modelo. Es decir, muchos registros en la columna project coinciden con un valor activo registrado en la dimensión de proyectos.

**Tablas** Contiene información detallada de las tablas existentes. Los parámetros son:

- Nombre (name): Nombre de la tabla.
- Fuente (source): Fuente de información de los datos.
- Partición (partition): Fecha de último procesamiento de la tabla.
- Esquema (schema): Nombre del esquema definido para los datos de la tabla.

Esta dimensión tienen una relación muchos a uno con la tabla de hechos del modelo. Es decir, muchos registros en la columna tables coinciden con un valor activo registrado en la dimensión de tablas.

**Fecha** Cada tabla contiene ciertos parámetros que brindan información más amplia, estos son:

- Fecha (date): Año, mes y día de la ejecución.
- Año (year): Año de la ejecución.
- Mes (month): Mes de la ejecución en formato año mes. Ejemplo: 202212
- Semestre (semester): Semestre del año al cual pertenece la fecha de ejecución, en formato año semestre. Ejemplo: 20221.
- Trimestre (quarter): Semestre del año al cual pertenece la fecha de ejecución, en formato año trimestre. Ejemplo: 20224.

Esta dimensión tiene una relación muchos a uno con la tabla de hechos del modelo. Es decir, muchos registros en la columna date coinciden con un valor en la dimensión de tablas.

**Tratamiento de dimensiones lentamente cambiantes (SCD)** Al realizar las dimensiones se presentó un problema con la variación de los datos a través del tiempo. Se eligió una estrategia que permita mantener el historial de los datos adicionando columnas en las dimensiones con fechas que establecen la vigencia de los datos y un campo que establece la validez de los mismos según la fecha actual. Los parámetros son:

- Válido desde (valid_from): Fecha de inicio de vigencia.
- Válido hasta (valid_until): Fecha de finalización de vigencia.
- Válido actual (valid_now): Booleano que indica si el registro está vigente.

Estos valores se agregaron a las dimensiones cuyos datos pueden cambiar con el paso del tiempo, estas son proceso,proyecto y tablas.

Teniendo todo esto en cuenta en la figura 9 se muestra el resultado del modelo dimensional para los registros. El resultado del procesamiento de datos mencionado en el apartado 5.3.4 es la tabla de hechos llamada executions.

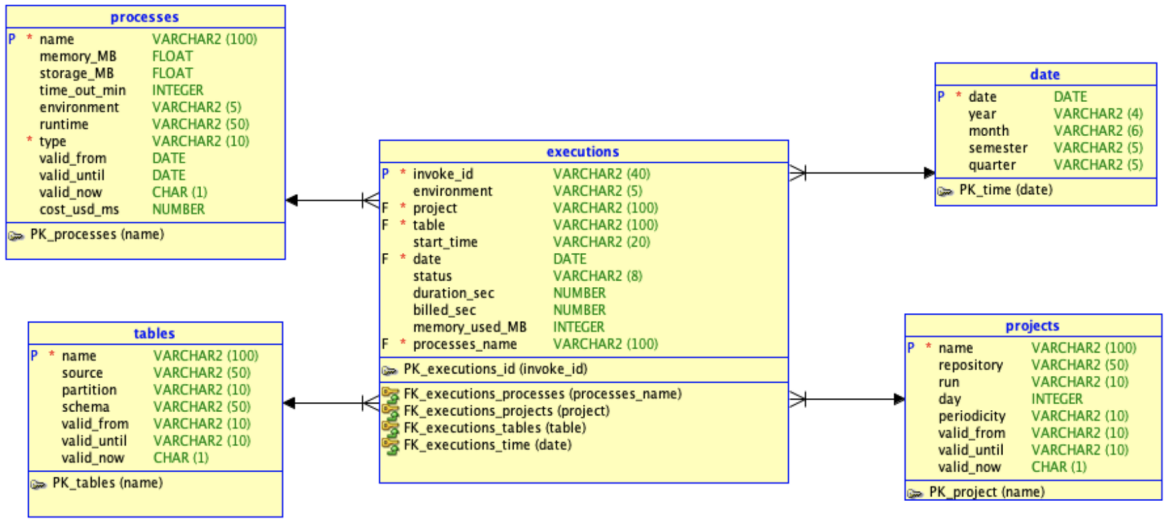


Figura 9: Diagrama modelo.

### 5.3.6 Métricas

Partiendo del modelo generado en el ítem anterior se desarrolló un script que genera una serie de reportes consecuentes con la información registrada en la tabla de hechos con el fin de responder preguntas claves para el equipo de datos.

Por medio de un código fuente modular se generaron reportes enfocados en análisis de estado de procesos y ejecuciones por entorno de trabajo. Ambas visiones tienen valores de tiempos, porcentajes y costos por mes para procesos, proyectos y tablas.

**Reporte por estado** El módulo para realizar el reporte por estado de ejecuciones recibe tres parámetros:

- Vector que contiene los diferentes procesos/proyectos/tablas que están registrados en la tabla de hechos del modelo.
- Un string que contiene el nombre de la columna que se quiere extraer y analizar de la tabla de hechos. Se espera que sea “process_name”, “project” o “tables” para analizar procesos, proyectos o tablas respectivamente.
- El nombre del archivo csv en el que se guardará el reporte. Este quedará env_name_report.csv.

Este módulo acumula todos las ejecuciones registradas clasificadas por mes y separadas en archivos por entorno de trabajo, los parámetros que se muestran son:

- Cantidad de ejecuciones exitosas.
- Cantidad de ejecuciones fallidas.
- Porcentaje de ejecuciones exitosas.
- Tiempo promedio de ejecución total.

- Tiempo promedio de ejecución para ejecuciones exitosas.
- Tiempo promedio de ejecución para ejecuciones fallidas.
- Tiempo total de ejecuciones.
- Tiempo total de ejecuciones exitosas.
- Tiempo total de ejecuciones fallidas.
- Costo promedio total.
- Costo promedio de ejecuciones exitosas.
- Costo promedio de ejecuciones fallidas.
- Costo total.
- Costo total de ejecuciones exitosas.
- Costo total de ejecuciones fallidas.

Toda esta información se almacena en archivos csv donde cada fila corresponde a un proceso/tabla/proyecto en un mes en específico. Las columnas del archivo son:

"'name','date','success','error','%_success','avg_time','avg_sucess','avg_error','total_time','total_time_success','total_time_error','avg_cost','avg_cost_sucess','avg_cost_error','total_cost','total_cost_success','total_cost_error'".

Este módulo se llama desde un módulo principal tres veces:

1. Estado de procesos: Se hace una petición a la base de datos para obtener todos los procesos registrados y se almacenan en un vector llamado processes. Luego se invoca el módulo de estados de ejecución con los siguientes argumentos processes, "process_name" y "process_status".
2. Estado de proyectos: Se hace una petición a la base de datos para obtener todos los proyectos registrados y se almacenan en un vector llamado projects. Luego se invoca el módulo de estados de ejecución con los siguientes argumentos projects, "project" y "project_status".
3. Estado de tablas: Se hace una petición a la base de datos para obtener todas las tablas registradas y se almacenan en un vector llamado tables. Luego se invoca el módulo de estados de ejecución con los siguientes argumentos tables, "tables" y "table_status".

# 6 RESULTADOS Y ANÁLISIS

El producto final diseñado, desarrollado y entregado al grupo de datos de la empresa SM es un proceso organizado y funcional que cumple con los objetivos planteados por ambas partes. A continuación se muestras los resultados y análisis de los mismos, por motivo de confidencialidad, no se revelan datos o información sensible a los procesos de la empresa.

## 6.1 Arquitectura final

La arquitectura final del proceso se muestra en la Figura 10 y sigue la siguiente secuencia. Primero la lambda de extracción toma todos los logs almacenados desde CloudWatch y los guarda en un bucket de s3, los cuales se filtran de acuerdo al formato definido por la lambda de limpieza y son almacenados en otra ubicación. Estos dos procesos están automatizados para hacerse una vez al mes, por medio de Airflow.

Posteriormente, un script desarrollado en python, toma los logs filtrados, los procesa y los convierte en la tabla de hechos del modelo.

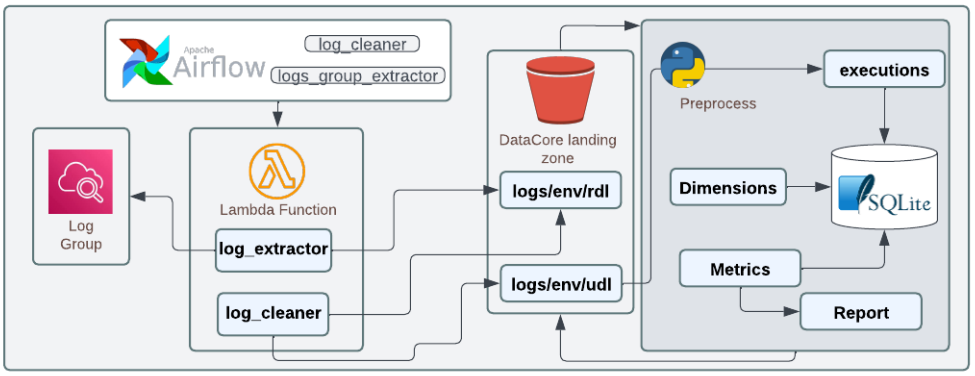


Figura 10: Arquitectura sistema monitoreo.

El modelo lógico que representa los datos se carga en una base de datos y se generan las dimensiones. A partir de este, un script en python genera las métricas y los reportes de la información.

## 6.2 Extracción y filtrado de datos

A continuación se muestra el sistema de monitoreo desarrollado para un proceso de transformación paso a paso. En la figura 11 se muestran los logs originales en CloudWatch para el proceso A.

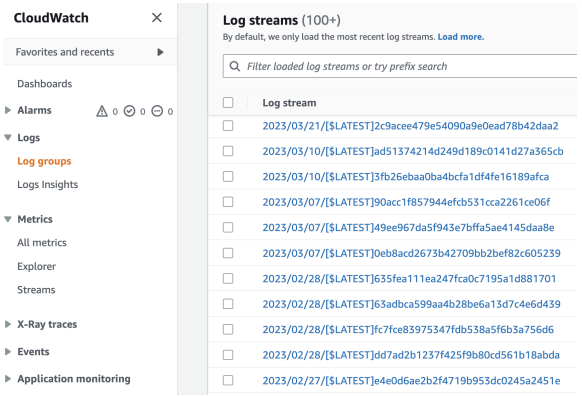


Figura 11: Logs originales para proceso A.

Para mostrar el funcionamiento del parámetro de extracción por fechas se va a extrajeron los logs para este proceso entre el 28 de Febrero y el 11 de Marzo de 2023. Los parámetros para la lambda de extracción en la esquina superior derecha y el resultado de la misma en S3, se muestran en la figura 12.

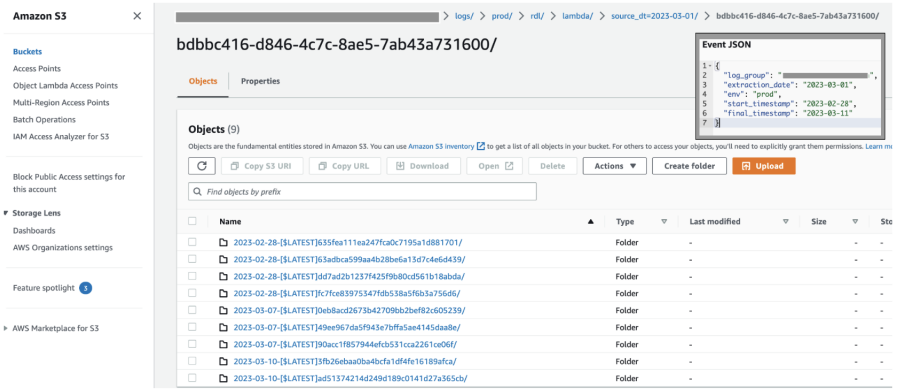


Figura 12: Extracción de logs para proceso A.

Es evidente que el resultado de la extracción sólo fueron los registros que estaban dentro del rango de tiempo establecido en los parámetros. También, se resalta que el almacenamiento de los mismos está en una carpeta identificada por la fecha de extracción definida en este caso (2023-03-01).

En la figura 13 se muestra la estructura dentro de los archivos que contienen los logs sin filtrar y los parámetros usados en la lambda de limpieza, buscando filtrar todos los registros extraídos el primero de Marzo de 2023.

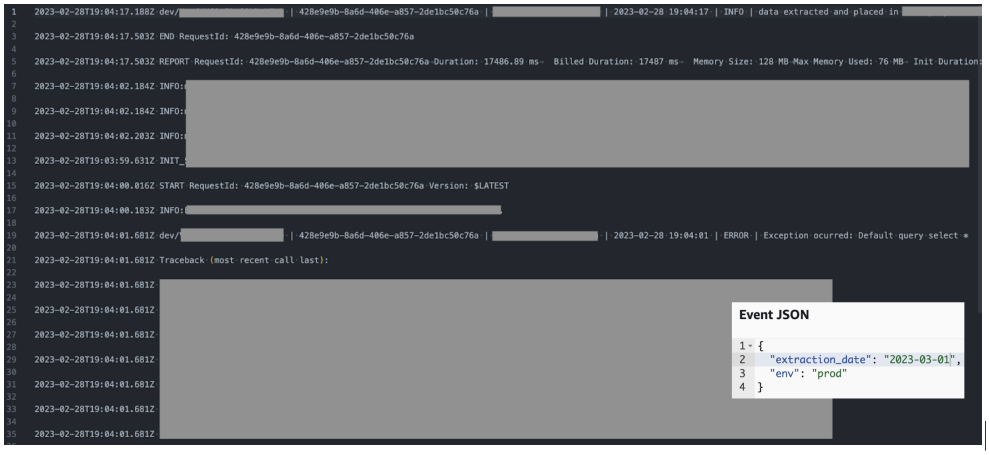


Figura 13: Logs en s3 sin filtrar.

Luego de ejecutar la lambda de filtrado se obtuvo un archivo en formato .csv que contiene todos los logs que cumplen con el formato elegido y los del tipo reporte. Note como si se compara la figura 13 con la 14 sólo quedan los logs en el formato y un registro de reporte por cada 'invoke_id' extraído.


Figura 14: Logs en s3 filtrados.

## 6.3 Preprocesamiento de datos

Con el script de preprocesamiento que se mencionó en la sección 5.3.4 se procesan los logs filtrados y se generó la tabla de hechos. En la figura 15 se muestra el archivo .csv generado en el filtrado de los datos cargado en la base de datos, en la parte inferior se ve que para este caso particular hay 45 registros.

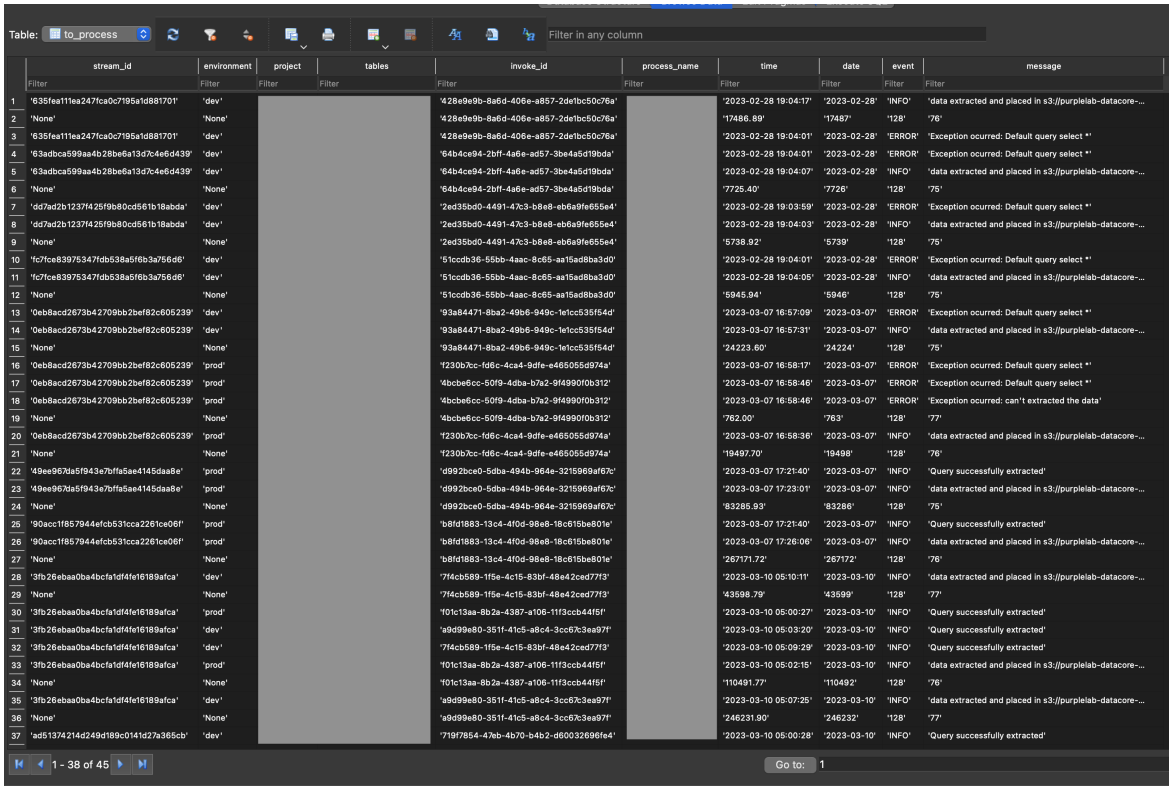


Figura 15: Logs filtrados sin preprocesamiento.

Por otro lado, en la figura 16 se ven los mismos datos procesados y ordenados. Se resalta como el formato y el procedimiento elegido, permitió disminuir la cantidad de registros de 45 a 15 sin eliminar información relevante.

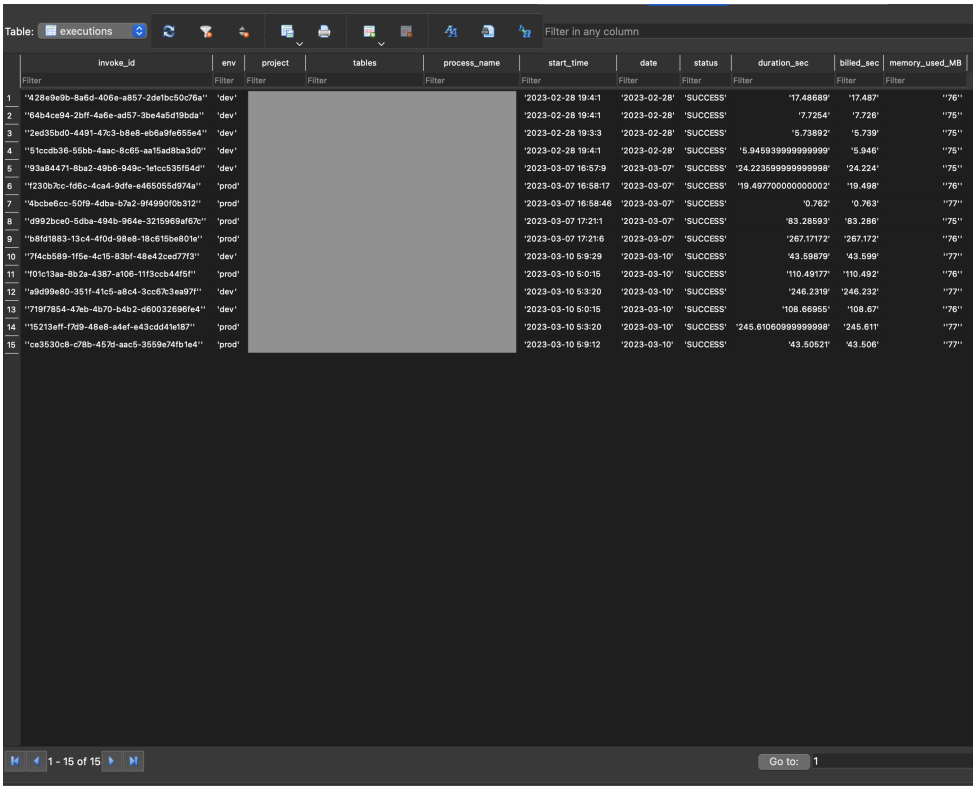


Figura 16: Logs en s3 filtrados y preprocesados.

## 6.4 Modelado

Partiendo de la tabla de hechos mostrada en la sección anterior (Figura 16) se hace el modelado de la datos como se explicó detalladamente en la sección 5.3.5. A continuación, sólo se mostrarán los resultados obtenidos para dos de las dimensiones explicadas, fecha y proyecto. Esto debido a que las otras dimensiones proveen información detallada del proceso monitoreado.

### 6.4.1 Fecha

Teniendo en cuenta que la extracción de los datos se hizo en el periodo de tiempo entre 28 de Febrero y el 11 de Marzo del 2023 y en CloudWatch se tenían registros para este intervalo del 28 de Febrero, 07 de Marzo y 10 de Marzo los resultados obtenidos que se ven en la figura 17 son congruentes, al contar con tres registros que coinciden con las fechas esperadas.

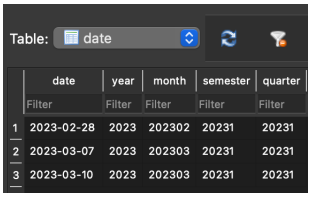


Figura 17: Dimensión fecha para proceso A.

### 6.4.2 Proyecto

El proceso A que se esta moniterando en este caso y en la fecha estipulada solo realizó procesos de transformación a dos proyectos como se ve en la figura 18.

Table: projects

| | name | repository | run | day | periodicity | valid_from | valid_until | valid_now |
|---|---|---|---|---|---|---|---|---|
| | Filter | Filter | Filter | Fil... | Filter | Filter | Filter | Filter |
| 1 | | ares | airflow | 10 | monthly | 2023-01-01 | 9999-01-01 | 1 |
| 2 | | theseus | airflow | 19 | monthly | 2023-01-01 | 9999-01-01 | 1 |

Figura 18: Dimensión proyectos para proceso A.

En esta dimensión se observan los campos para tratar las SCD que, como se observa en la figura 18 ambos registros son válidos. Este resultado es lo esperado ya que el día en el que se realizó la extracción pertenece al rango de validez establecido para la información de los proyectos.

## 6.5 Métricas

Finalmente, se calculan las mêtricas para el proceso A. Como se mencionó anteriormente el sistema de monitoreo genera 6 archivos de reporte enfocados en proyectos, proceso y tablas del monitoreo en el intervalo de tiempo establecido. En la figura 19 se muestra el resultado de las métricas resultantes sólo en el enfoque de proceso A en los dos entorno de desarrollo y en la figura 20 se ven las métricas de producción.

| date | success | error | % success | avg time | avg success | avg error | total time | total time success | total time error | avg cost | avg cost sucess | avg cost error | total cost | total cost success | total cost error |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| '2023-02' | '4' | '0' | '100.0' | '9.22' | '9.22' | '0' | '36.9' | '36.9' | '0' | '9.22' | '9.22' | '0' | '36.9' | '36.9' | '0' |
| '2023-03' | '4' | '0' | '100.0' | '105.68' | '105.68' | '0' | '422.72' | '422.72' | '0' | '105.68' | '105.68' | '0' | '422.73' | '422.73' | '0' |

Figura 19: Métricas de proceso A en desarrollo.

| date | success | error | % success | avg time | avg success | avg error | total time | total time success | total time error | avg cost | avg cost sucess | avg cost error | total cost | total cost success | total cost error |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| '2023-03' | '7' | '0' | '100.0' | '110.05' | '110.05' | '0' | '770.32' | '770.32' | '0' | '110.05' | '110.05' | '0' | '770.33' | '770.33' | '0' |

Figura 20: Métricas de proceso A en producción.

Los resultados obtenidos evidencian patrones de comportamiento esperados dentro de los procesos de transformación de datos. Cuando se corren las ejecuciones en producción se espera que sean versiones estables, por tanto, se procesan mayor cantidad de datos, lo cual se evidencia en un costo mayor.

Por otro lado, si se compara el mes de Marzo, en producción se hicieron casi el doble de llamados respecto a desarrollo. Este comportamiento permite tener trazabilidad de los efectos y decisiones que se toman internamente al realizar pruebas y establecer versiones de los procesos.

En la figura 21 se ve el resultado de las métricas para las tablas procesadas en el intervalo de tiempo elegido. Es importante resaltar como el sistema brinda métricas que permiten identificar los comportamientos dentro de las transformaciones con una alta granularidad de los procesos.


Figura 21: Métricas de tablas de proceso A en desarrollo.

Ahora, al analizar estas métricas en paralelo con los indicadores de tablas y proyectos en cada uno de los entornos, se obtienen respuestas acerca de los tiempos de ejecución, consumo de recursos y alternativas de mejora dentro de los procesos. La ejecución del sistema a largo plazo, permitirá la acumulación de métricas que permitirán identificar claramente los cuellos de botella en los procesos del equipo que serán determinantes en la toma de decisiones sustentadas siguiendo el tipo de negocio.

## 6.6 Entregables

Se hizo entrega de una serie de documentos y manuales en los cuales se describe la solución diseñada, el paso a paso para ejecutar el proceso y cómo funciona, permitiendo que futuros desarrolladores entiendan y puedan hacer uso del mismo.

### 6.6.1 Manual técnico

Descripción detallada de los módulos necesarios para completar todo el proceso, con el objetivo para cada objeto que se construyó y para cada acción dentro de ese objeto, enumerar los parámetros de entrada y salida, para permitir que cualquier desarrollador interesado en el proceso tenga la capacidad de entenderlo o modificarlo si es necesario. Es un documento dinámico que debe ser revisado continuamente a medida que se apliquen cambios.

### 6.6.2 Manual operabilidad

El manual de operabilidad es un documento que se entregó al equipo de datos con el propósito de proporcionar instrucciones de uso e información sobre la ejecución del proceso automatizado para el

monitoreo de los registros.

### 6.6.3 Código fuente

Todos los módulos desarrollados fueron almacenados en un repositorio organizado que cumple con las buenas prácticas establecidas por la compañía.

### 6.6.4 Reportes con métricas

Se entregan seis archivos que contienen las métricas obtenidas en el desarrollo del proceso en formato csv.

# 7 CONCLUSIONES

- La configuración establecida para la extracción de los registros desde servicios en la nube permite el dinamismo y adaptación del sistema a los requerimientos del usuario final. Ya que, brinda la libertad de elegir los procesos e intervalos de tiempo que desean ser monitoreados. Además, el almacenamiento ordenado y centralizado de los recursos durante todo el proceso de extracción y filtrado facilita las labores de los ingenieros de datos al crear para ellos núcleos de procesamiento para la toma de decisiones. Finalmente, la automatización de estos sistemas mejoran el flujo de trabajo del equipo.

- El modelo de datos diseñado brinda una visión detallada de los diferentes procesos activos en el equipo. Al ser un modelo enfocado en las dimensiones de los procesos logra cuantificar y confirmar por medio de él configuraciones, comportamientos, tiempos y recursos requeridos y usados en las transformaciones de datos.

- El sistema de monitoreo entregado permitió desde un fuente de datos en la nube, extraer, estandarizar, filtrar, centralizar, modelar y reportar registros de procesamiento de datos. Con los cuales se lograron detectar problemas dentro de las ejecuciones, funcionalidades e incluso estudiar requerimientos de optimatización y mejora de los procesos activos dentro de la compañía.

- El desempeño del software para el monitoreo de procesos dentro del equipo de datos logra confirmar tendencias en el comportamiento de los mismos y responder a necesidades de entendimiento a nivel de negocio. Su máximo valor se verá con el paso del tiempo y acumulación de métricas para identificar cuellos de botella y tomar decisiones fundamentales y críticas partiendo de estos.

# 8 TRABAJO A FUTURO

El desarrollo de este sistema se hizo partiendo de la hipótesis de que la fuente de datos es ideal. Como trabajo a futuro se plantea la validación de los datos de entrada y no limitar el monitoreo a funciones lambda. Con esto se busca brindar una reporte de todos los procesos activos en la compañía, permitiendo tener una visión global en términos de recursos usados.

# 9 AGRADECIMIENTOS

Agradezco profundamente la confianza y oportunidad brindada por la empresa Source Meridian y en especial el acompañamiento permanente por parte del equipo DataCore en la formación, crecimiento y fortalecimiento de mis conocimientos en el desarrollo de este proyecto. El trabajar con un equipo altamente capacitado y colaborativo me enseñó herramientas claves para afrontar retos en mi carrera profesional, siguiendo la ética y buenas prácticas para entregar resultados de calidad.

# REFERENCIAS


[1] ThreePoints (2021, Marzo 03) Usos del Big Data en las empresas. [Online]. Disponible en: https://www.threepoints.com/blog/usos-del-big-data-en-las-empresas

[2] Power Data Solutions (s.f) Glosarios de Términos: Big Data. [Online]. Disponible en: https://www.powerdata.es/big-data

[3] A. Diaz (2017, Enero 19) ¿Qué son los Logs y por qué deben interesarte?. [Online]. Disponible en: https://dbibyhavas.io/es/blog/que-son-los-logs/

[4] E. Dans. (2011, Octubre 19). Big Data: una pequeña introducción. [Online]. Disponible en: https://www.enriquedans.com/2011/10/big-data-una-pequena-introduccion.html

[5] Google Cloud (2022). ¿Qué es la inteligencia empresarial?. [Online]. Disponible en: https://cloud.google.com/learn/what-is-business-intelligence?hl=es

[6] Oracle Cloud (2022). ¿Qué es un almacén de datos?. [Online]. Disponible en: https://www.oracle.com/co/database/what-is-a-data-warehouse/

[7] Amazon Web Services (2022) ¿Qué es aws? [Online]. Disponible en: https://aws.amazon.com/es/what-is-aws/

[8] Amazon Web Services (2022) Amazon S3. [Online]. Disponible en: https://aws.amazon.com/es/s3/

[9] Amazon Web Services (2022) Amazon Lambda [Online]. Disponible en: https://aws.amazon.com/es/lambda/

[10] Amazon Web Services (2022) Amazon CloudWatch [Online]. Disponible en: https://aws.amazon.com/es/lambda/

[11] Google Cloud (2022) Descripción general de Google Cloud. [Online].Disponible en: https://cloud.google.com/why-google-cloud

[12] Google Cloud (2022) Cloud Composer. [Online]. Disponible en: https://cloud.google.com/composer

[13] The PostgreSQL Global Development Group (2023) PostgreSQL: The World's Most Advanced Open Source Relational Database. [Online]. Disponible en: https://www.postgresql.org/

[14] IBM Cloud Education (2020, Abril 28) ETL. [Online]. Disponible en: https://www.ibm.com/cloud/learn/etltoc-etl-vs-elt-goFgkQcP

[15] L. Sancho (2022, Septiembre 29) Logs: qué son y por qué monitorizarlos. [Online]. Disponible en: https://pandorafms.com/blog/es/logs/

[16] Python (2021) 16.6. Logging - Logging facility for Python. [Online]. Disponible en: https://docs.python.org/3.6/library/logging.html

[17] Amazon Web Services (2022) ¿Qué es el modelado de datos?. [Online]. https://aws.amazon.com/es/what-is/data-modeling/: :text=El%20modelado%20de%20datos%20aporta,sistema%20en%20toda%20la%20organizaci%C3%B3n

[18] E. León Guzmán (.s.f) Modelamiento Dimensional. Ingeniería de sistemas. Universidad Nacional de Colombia. [Online]. Disponible en: https://disi.unal.edu.co/ eleonguz/cursos/bda/presentaciones/S3-modelamiento.pdf