INGENIERÍA INFORMÁTICA

3CO25 16-0619 Datos Masivos

Proyecto Final

“Solución en la Nube para Procesamiento de Datos Masivos”

Profesor

Msc. Esteban Vega Retana

Estudiantes

Hellen Méndez Rivas

Emanuel Morales Ávila

Jarod Ugalde Slak

Nathalie Garcia Rivera

Alejandro Mora Madrigal

IIICO 2025

Resumen Ejecutivo

Este proyecto implementa una arquitectura en Amazon Web Services orientada al procesamiento de datos masivos, utilizando un dataset público de accidentes de tránsito en Estados Unidos con un tamaño aproximado de 2,8 GB y más de 7,7 millones de registros. La solución se construyó como un flujo integral que inicia con la ingesta y almacenamiento de datos en Amazon S3 bajo un enfoque de *data lake*, continúa con la catalogación automática del esquema mediante AWS Glue Crawler, y permite la consulta y transformación a través de Amazon Athena. Con el fin de optimizar el rendimiento y reducir costos por escaneo, los datos se convirtieron a formato Parquet y se particionaron por año, habilitando consultas más eficientes sobre subconjuntos del dataset. Por último, se integró una capa de visualización mediante Power BI Desktop conectado a Athena por ODBC, construyendo un dashboard interactivo. Adicionalmente, se documentan decisiones técnicas y eventos relevantes durante la implementación, incluyendo los inconvenientes al intentar habilitar Amazon QuickSight y la elección de una alternativa funcional para cumplir los objetivos del proyecto.

Introducción

El procesamiento de datos masivos se ha convertido en un componente esencial para organizaciones que requieren extraer valor a partir de volúmenes elevados de información, con alta velocidad de generación y variedad de estructuras. En este contexto, los servicios en la nube ofrecen una ventaja significativa al permitir escalabilidad, pago por uso y acceso a herramientas administradas que reducen la complejidad operativa frente a infraestructuras tradicionales. El presente proyecto, desarrollado para el curso de Datos Masivos, tiene como propósito diseñar, implementar y documentar una solución en la nube que permita almacenar, transformar, analizar y visualizar un conjunto de datos real de gran tamaño, aplicando principios propios de arquitecturas Big Data.

Para esta implementación se seleccionó un dataset masivo de accidentes en Estados Unidos, debido a que su volumen y cantidad de registros representan un escenario realista para poner a prueba técnicas de ingesta, catalogación, transformación y consulta eficiente. La solución se construyó en AWS con una orientación serverless, evitando el aprovisionamiento de servidores dedicados y aprovechando servicios administrados como S3, Glue y Athena. El proyecto contempla una capa de visualización que permita comunicar resultados relevantes mediante gráficos y métricas, aspecto clave para demostrar el funcionamiento práctico de la arquitectura implementada.

Objetivo general

Diseñar e implementar una arquitectura funcional en AWS para almacenar, procesar, analizar y visualizar un dataset masivo, procurando mantenerse en niveles gratuitos o costos mínimos.

Objetivos específicos

Comprender servicios Big Data en cloud.

Diseñar arquitectura con almacenamiento, procesamiento, base analítica y visualización.

Implementar flujo de carga, transformación y análisis.

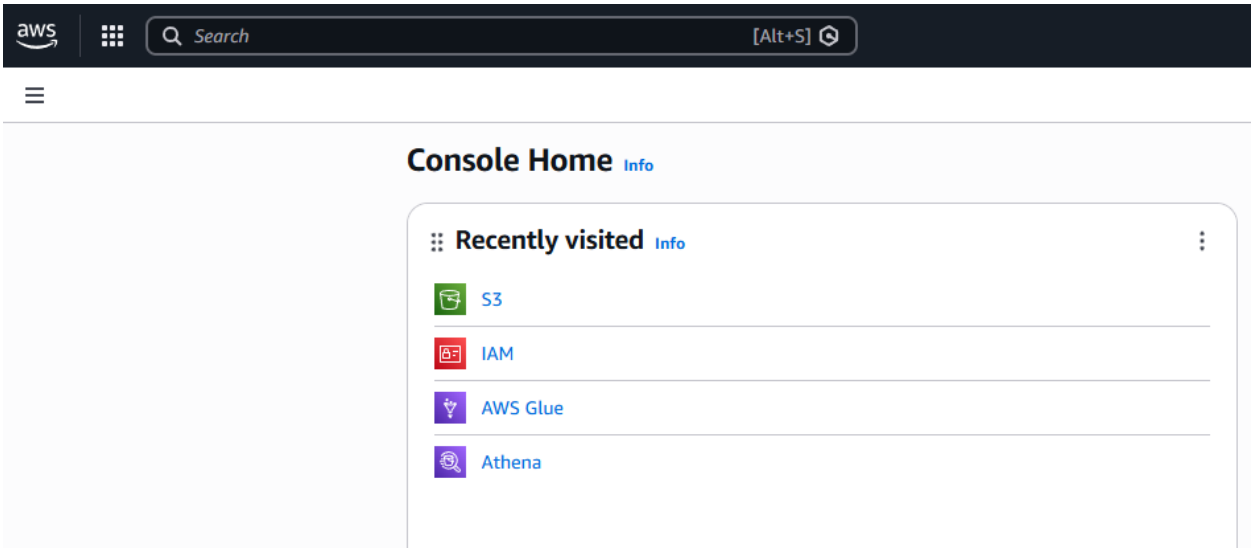
Evaluar costos y limitaciones del Free Tier.

Documentar con justificación técnica y conclusiones.

Investigación y elección de proveedor

Si bien existen múltiples proveedores capaces de soportar flujos de Big Data, incluyendo Microsoft Azure, Google Cloud Platform y AWS, se seleccionó AWS por la disponibilidad de servicios integrados y ampliamente documentados que permiten construir una solución completa con un enfoque modular. En particular, AWS facilita la construcción de arquitecturas tipo data lake mediante Amazon S3, permitiendo almacenar grandes volúmenes de información de forma durable y con costos controlables. Además, la existencia de un catálogo de metadatos administrado con AWS Glue Data Catalog reduce el esfuerzo de definir esquemas manualmente, lo cual resulta especialmente valioso cuando se trabaja con datasets extensos y con múltiples columnas.

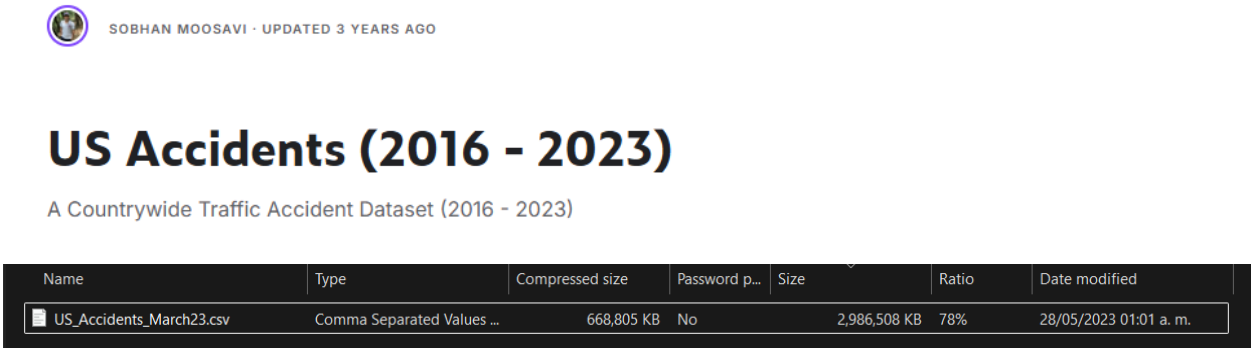
Amazon Athena permite ejecutar consultas SQL directamente sobre datos almacenados en S3 sin desplegar infraestructura de base de datos, lo cual se alinea con el objetivo académico de comprender enfoques modernos de análisis a escala y modelos de pago por uso. Finalmente, la integración con herramientas de inteligencia de negocios externas como Power BI mediante controladores ODBC ofrece una alternativa práctica para visualización, especialmente cuando herramientas nativas como QuickSight presentan restricciones administrativas o de suscripción en la cuenta utilizada. En conjunto, estas razones justifican la elección de AWS como plataforma principal para el proyecto, procurando mantenerse dentro de limitaciones del nivel gratuito y minimizando el riesgo de costos inesperados



Dataset utilizado

El dataset seleccionado corresponde a US Accidents (2016–2023), publicado en Kaggle por Moosavi (2023), y se relaciona con el área de transporte. El archivo original se encuentra en formato CSV, con un tamaño aproximado de 2,8 GB y una cantidad estimada de 7,7 millones de registros, lo cual cumple ampliamente el requisito mínimo de 100 MB establecido en el enunciado del proyecto. La información contenida incluye identificadores de eventos, variables de severidad, componentes temporales (hora de inicio y fin), atributos de localización (estado, ciudad y coordenadas), así como variables contextuales como condiciones climáticas y características del entorno vial.

Su volumen lo convierte en un caso apropiado para evaluar estrategias de procesamiento y consulta eficientes, ya que operar el archivo completo en herramientas locales típicamente resulta lento o inviable. Por esta razón, el dataset es adecuado para demostrar la utilidad de un enfoque en la nube, donde el almacenamiento, el procesamiento distribuido y la optimización de formatos pueden mejorar significativamente el desempeño y la capacidad de análisis.



Arquitectura propuesta

Componentes implementados

La arquitectura implementada se diseñó bajo un enfoque de data lake en AWS, donde Amazon S3 funciona como repositorio central para almacenar tanto los datos en estado bruto como los datos procesados. Este patrón permite separar claramente la zona de ingesta o *raw*, de la zona analítica o *processed*, lo cual facilita la trazabilidad del flujo y evita pérdida de información original. Para habilitar el descubrimiento del esquema y la organización de metadatos, se utilizó AWS Glue Crawler, que inspecciona los objetos en S3 y genera automáticamente la definición de tablas en el Glue Data Catalog.

Luego, Amazon Athena se utilizó como motor de consulta y transformación, permitiendo ejecutar SQL sobre los datos en S3 sin necesidad de administrar clústeres o servidores. Como práctica de optimización típica en Big Data, el dataset fue convertido a formato Parquet y particionado por año, lo cual reduce el volumen de datos escaneados en consultas y mejora tiempos de respuesta. Finalmente, la capa de consumo y visualización se implementó mediante Power BI Desktop, conectado a Athena mediante ODBC, con el fin de construir un dashboard interactivo que muestre hallazgos relevantes.

Flujo lógico

CSV bruto → S3 /raw/

Glue Crawler → tabla raw en datosmasivos_db

Athena → consultas y validaciones

Athena CTAS → Parquet particionado → S3 /processed/accidents_parquet/

Athena → consultas optimizadas

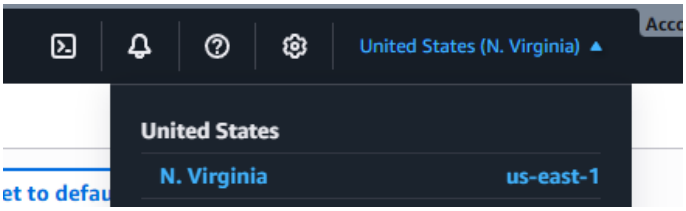
Power BI → conexión ODBC → dashboard

Implementación paso a paso

Creación de cuenta AWS y región

Se creó cuenta AWS (Free Tier).

Se seleccionó región us-east-1 para Glue/Athena y compatibilidad de drivers.



IAM: creación de usuario y credenciales

Se creó un usuario IAM dedicado (ej. powerbi-athena-user).

Se asignaron permisos mínimos necesarios:

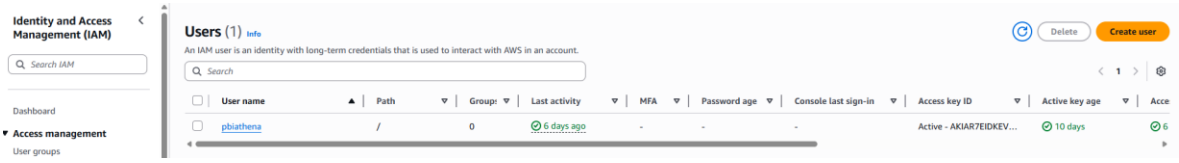
S3 (lectura/escritura en bucket del proyecto y output de Athena)

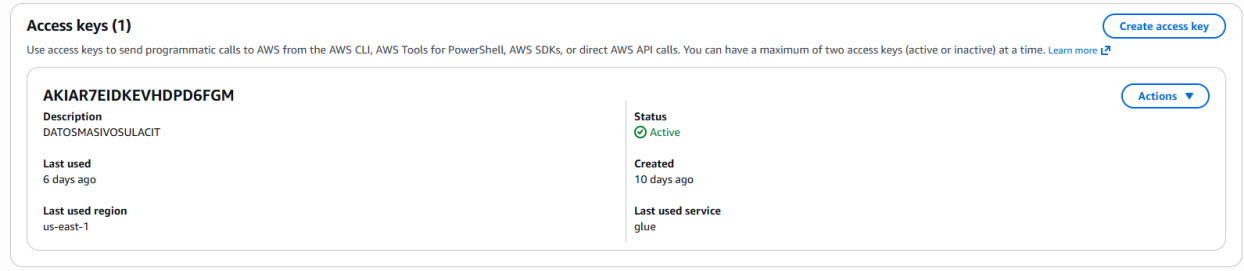
Athena (ejecución de queries)

Glue (lectura del Data Catalog)

Se creó Access Key (para ODBC y Power BI).

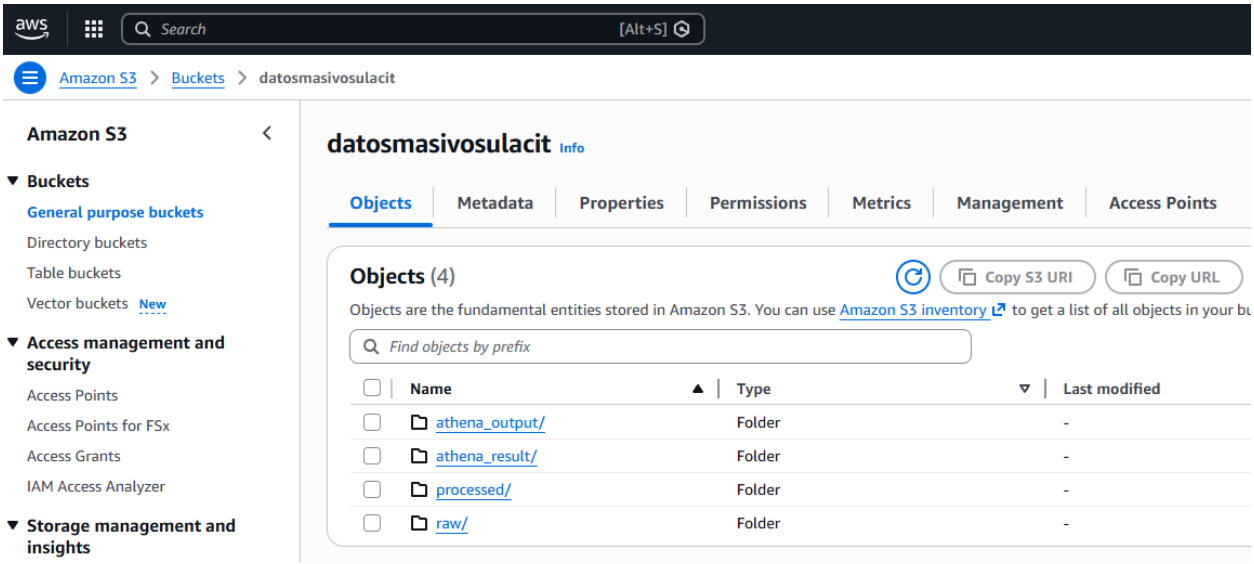
Se registraron: Access Key ID y Secret Access Key.





Amazon S3: creación del bucket y estructura

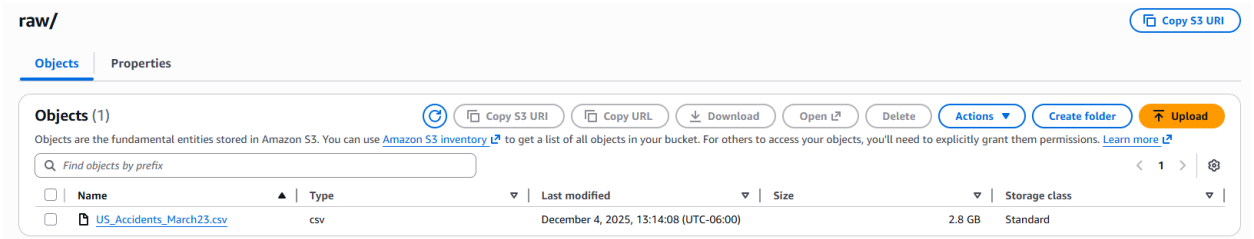
Amazon S3 se utilizó como capa de almacenamiento principal y como base del data lake. En el proyecto, el bucket se organizó en rutas que separan datos brutos y datos procesados, permitiendo preservar el dataset original y generar derivados optimizados sin sobrescribir la fuente. Esta práctica es común en entornos Big Data debido a la durabilidad del servicio y su capacidad de escalar con bajo esfuerzo operativo. Además, centralizar los datos en S3 habilita que servicios como Glue y Athena consuman la información de forma directa, eliminando la necesidad de mover archivos entre sistemas.



Carga del dataset masivo a S3

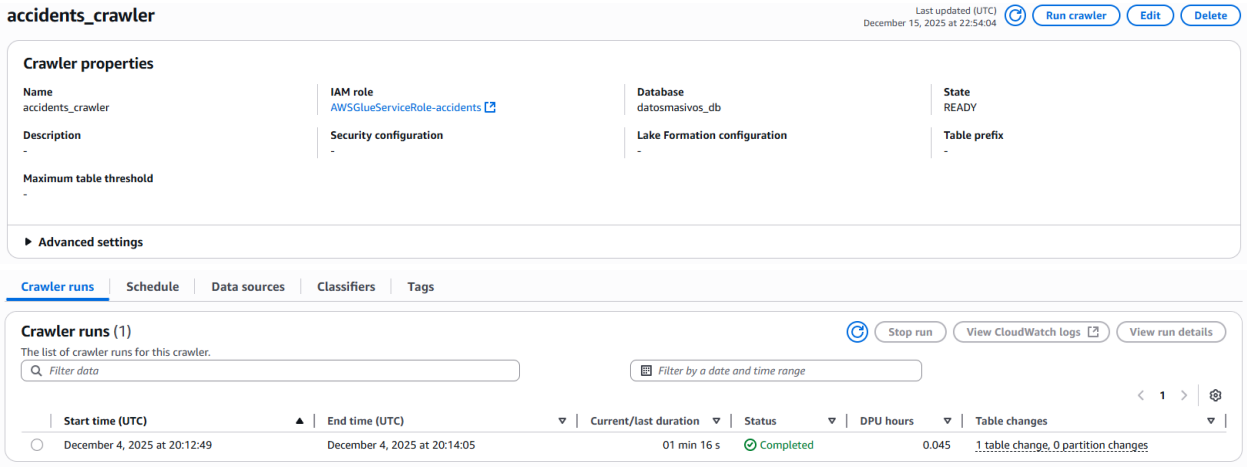
Se subió el CSV de 2.8 GB a raw/.

Durante la carga se verificó el progreso desde la consola.



AWS Glue: Crawler para inferir esquema y crear tabla RAW

AWS Glue se utilizó para automatizar el descubrimiento del esquema del dataset. El Crawler inspecciona los archivos en S3 e infiere columnas y tipos de datos para registrar una tabla en el Glue Data Catalog. Esto es especialmente útil cuando se trabaja con datasets grandes, donde definir manualmente el esquema puede resultar propenso a errores y consume tiempo. El Data Catalog funciona como una “capa de metadatos” que permite que Athena y otros servicios consulten las tablas sin necesidad de conocer detalles físicos del almacenamiento.

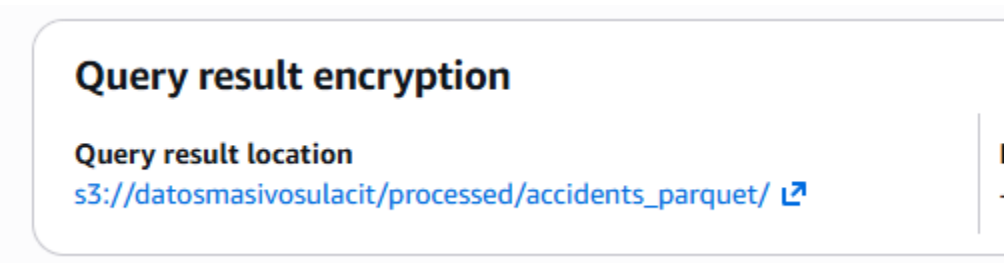


Amazon Athena: configuración inicial

Amazon Athena se utilizó como motor de consulta y transformación bajo un modelo serverless. Su principal ventaja es que permite ejecutar SQL directamente sobre datos en S3, evitando la administración de servidores o bases de datos tradicionales. En este proyecto se utilizó Athena tanto para validar la tabla RAW como para ejecutar transformaciones mediante CTAS (Create Table As Select), generando salidas en Parquet. Athena cobra por cantidad de datos escaneados, por lo que el diseño del proyecto incorporó optimizaciones para reducir escaneo y controlar costos.

Se abrió Athena y se configuró el Query result location:

s3://datosmasivosulacit/processed/accidents_parquet/

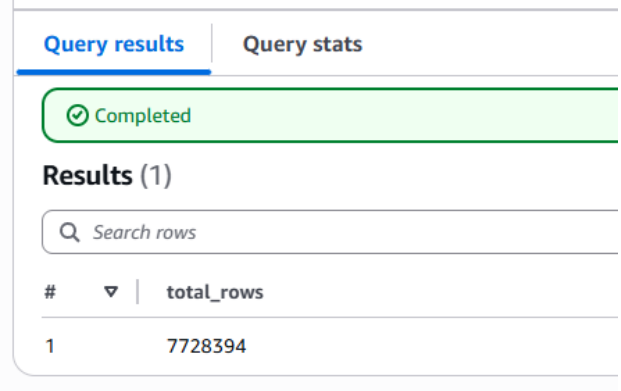


Athena: validación de tabla RAW

Se ejecutaron consultas para validar la carga.

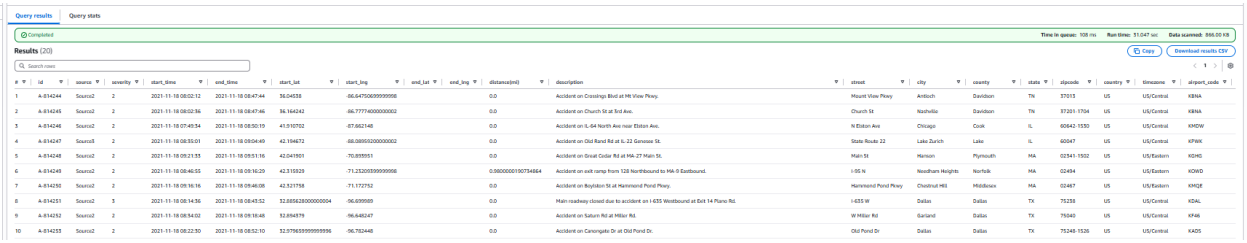
Consulta 1 – conteo total

```
SELECT COUNT(*) AS total_rows
FROM datosmasivos_db.raw;
```



Consulta 2 – muestra

```
SELECT *
FROM datosmasivos_db.raw
LIMIT 20;
```



Transformación (ETL) y optimización del dataset

¿Qué transformación se hizo?

Conversión de CSV → Parquet (mejor performance)

Limpieza básica: manejar errores de timestamp con try_cast

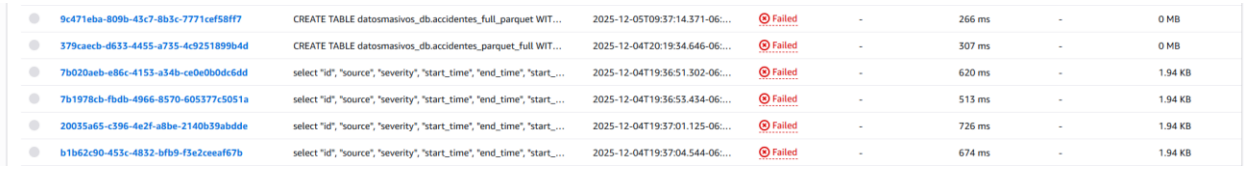
Enriquecimiento: crear columna year

Particionamiento: partitioned_by = ARRAY['year']

Problemas encontrados durante la transformación

Error de timestamp (“Invalid format…”)
Solución: usar try_cast(start_time AS timestamp) y filtrar nulos.

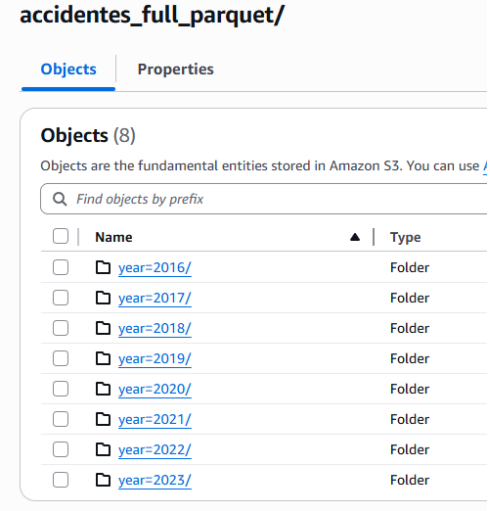
Columnas con caracteres especiales (paréntesis, %)
Solución: usar SELECT * para evitar referenciar nombres complejos.



9c471eba-809b-43c7-8b3c-7771cef58ff7	CREATE TABLE datosmasivos_db.accidentes_full_parquet WIT…	2025-12-05T09:37:14.371-06:…	Failed	-	266 ms	-	0 MB
379caecb-d633-4455-a735-4c9251899b4d	CREATE TABLE datosmasivos_db.accidentes_parquet_full WIT…	2025-12-04T20:19:34.646-06:…	Failed	-	307 ms	-	0 MB
7b020aeb-e86c-4153-a34b-ce0e0b0dc6dd	select "id", "source", "severity", "start_time", "end_time", "start_…	2025-12-04T19:36:51.302-06:…	Failed	-	620 ms	-	1.94 KB
7b1978cb-fbdb-4966-8570-605377c5051a	select "id", "source", "severity", "start_time", "end_time", "start_…	2025-12-04T19:36:53.434-06:…	Failed	-	513 ms	-	1.94 KB
20035a65-c396-4e2f-a8be-2140b39abdde	select "id", "source", "severity", "start_time", "end_time", "start_…	2025-12-04T19:37:01.125-06:…	Failed	-	726 ms	-	1.94 KB
b1b62c90-453c-4832-bfb9-f3e2ceeaf67b	select "id", "source", "severity", "start_time", "end_time", "start_…	2025-12-04T19:37:04.544-06:…	Failed	-	674 ms	-	1.94 KB

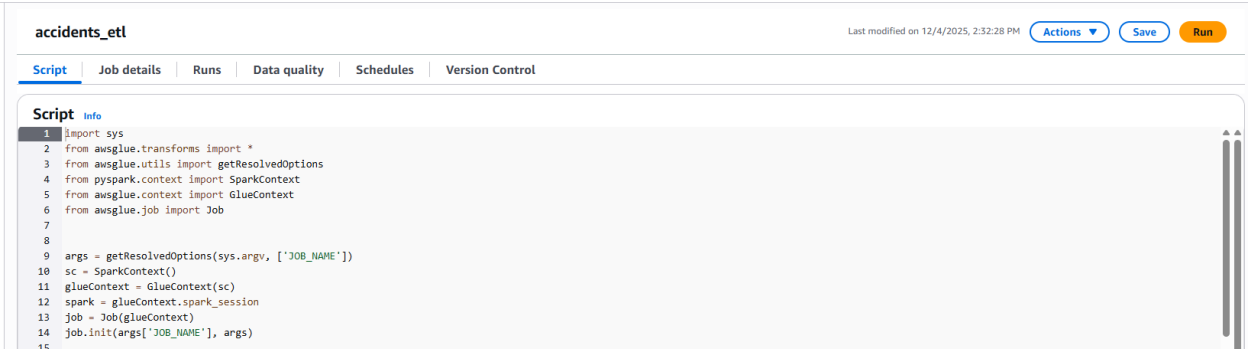
Creación del dataset Parquet FULL particionado

```
DROP TABLE IF EXISTS datosmasivos_db.accidentes_parquet;

CREATE TABLE datosmasivos_db.accidentes_parquet

WITH (

    format = 'PARQUET',

    external_location =
's3://datosmasivosulacit/processed/accidents_parquet/',

    partitioned_by = ARRAY['year']

) AS

SELECT

    *,

    year(try_cast(start_time AS timestamp)) AS year

FROM datosmasivos_db.raw

WHERE try_cast(start_time AS timestamp) IS NOT NULL;
```



ETL Alternativo con AWS Glue Job (PySpark / Spark)

Como complemento al enfoque basado en Athena, se implementó un Glue Job utilizando PySpark para demostrar procesamiento distribuido con Apache Spark, una tecnología ampliamente utilizada en Big Data. Este job ejecuta un ETL clásico, lee la tabla RAW desde el catálogo, convierte campos de tiempo, elimina registros inválidos, deriva campos como año/mes y genera un dataset agregado por estado y año. Aunque este resultado no reemplaza el Parquet FULL (detalle completo), sí sirve como dataset resumido para análisis de tendencias y KPIs, evidenciando habilidades prácticas de transformación a escala y uso de Python en la nube.



Extract:

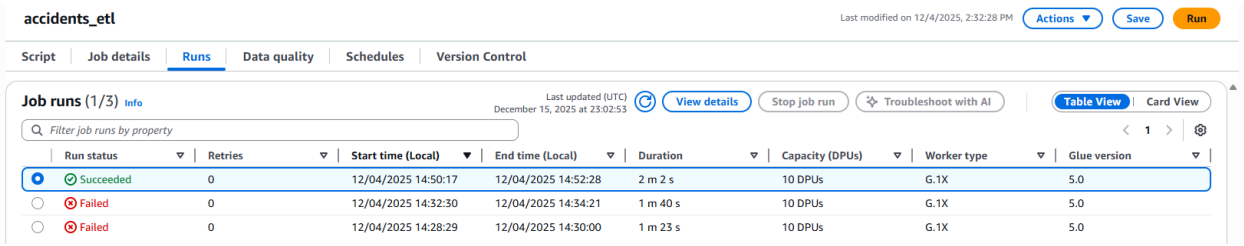
El job leyó los datos desde el AWS Glue Data Catalog, consumiendo la tabla RAW creada por el Crawler (CSV en S3) sin necesidad de definir manualmente el esquema.

Transform:

Se aplicaron transformaciones en PySpark:

- Conversión de Start_Time (texto) a timestamp (start_time_ts)
- Filtrado de filas inválidas (sin fecha válida o sin estado)

- Creación de columnas derivadas year y month
- Agregación: accidentes totales por State y year



Load:
El resultado agregado se guardó en Amazon S3 en formato Parquet, particionado por year:

s3://datosmasivosulacit/processed/accidents_full_parquet/

Rol del Glue Job en la arquitectura:
Este job generó un dataset resumido que complementa el dataset full generado con Athena. La versión agregada es útil para KPIs rápidos y análisis por tendencias anuales o geográficas sin cargar millones de registros a la herramienta de visualización.

Código del Glue Job (PySpark)

```
import sys

from awsglue.transforms import *

from awsglue.utils import getResolvedOptions

from pyspark.context import SparkContext

from awsglue.context import GlueContext

from awsglue.job import Job


args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()

glueContext = GlueContext(sc)

spark = glueContext.spark_session

job = Job(glueContext)
```

```
job.init(args['JOB_NAME'], args)

DATABASE_NAME = "datosmasivos_db"
TABLE_NAME = "raw"

datasource0 = glueContext.create_dynamic_frame.from_catalog(
    database=DATABASE_NAME,
    table_name=TABLE_NAME
)
df = datasource0.toDF()

from pyspark.sql.functions import col, to_timestamp, year, month,
count as spark_count

df = df.withColumn(
    "start_time_ts",
    to_timestamp(col("start_time"))
)

df = df.filter(col("start_time_ts").isNotNull())
df = df.filter(col("state").isNotNull())
df = df.withColumn("year", year(col("start_time_ts")))
df = df.withColumn("month", month(col("start_time_ts")))

agg_df = df.groupBy("state", "year").agg(
    spark_count("*").alias("total_accidents"))
```

```
OUTPUT_PATH = "s3://datosmasivosulacit/processed/accidents_parquet/"
(
    agg_df
    .write
    .mode("overwrite")
    .partitionBy("year")
    .parquet(OUTPUT_PATH))
job.commit()
```

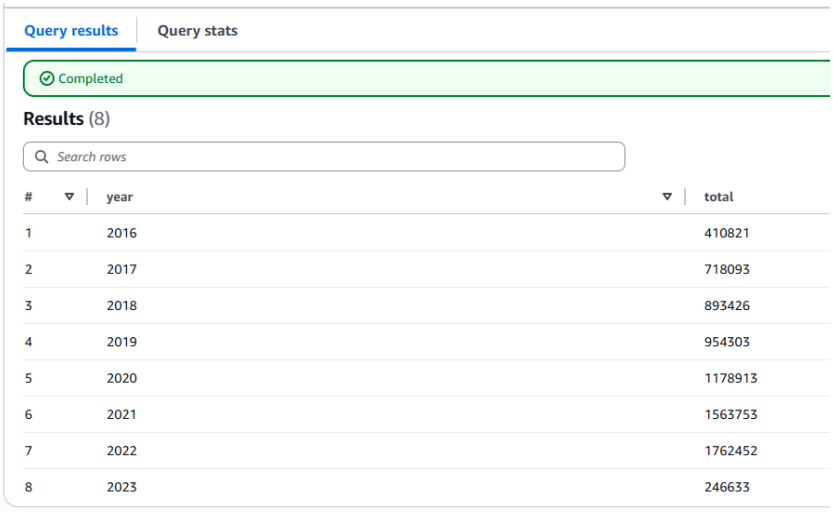
Validación de particiones y conteo en Parquet

Conteo del Parquet

```
SELECT COUNT(*) AS total_rows
FROM datosmasivos_db.accidentes_parquet;
```

Distribución por año

```
SELECT year, COUNT(*) AS total
FROM datosmasivos_db.accidentes_parquet
GROUP BY year
ORDER BY year;
```



#	year	total
1	2016	410821
2	2017	718093
3	2018	893426
4	2019	954303
5	2020	1178913
6	2021	1563753
7	2022	1762452
8	2023	246633

Visualización

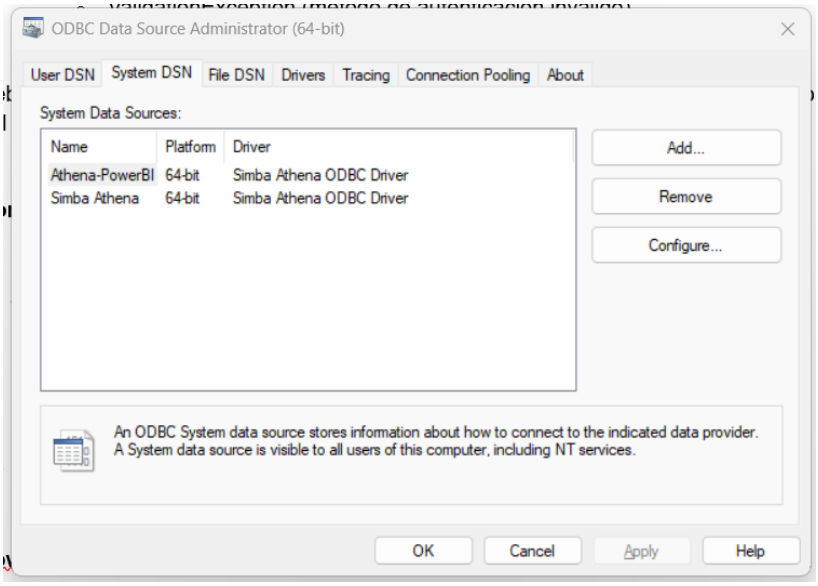
Intento con Amazon QuickSight

Durante la etapa de visualización se evaluó el uso de Amazon QuickSight como herramienta nativa de AWS. Sin embargo, al intentar activar la suscripción desde la consola, el proceso no finalizaba correctamente (la interfaz recargaba sin completar el registro), y al intentar habilitarlo mediante AWS CLI se presentaron excepciones relacionadas con validación y precondiciones de suscripción. Dado que la resolución de este tipo de incidentes depende en gran medida de configuraciones administrativas de la cuenta (suscripciones habilitadas, métodos de autenticación permitidos y políticas asociadas), se decidió adoptar una alternativa equivalente para cumplir con el requisito de visualización: Power BI Desktop. Esta decisión permitió mantener el avance del laboratorio y completar el objetivo de demostrar la solución en funcionamiento, documentando además una limitación realista que puede ocurrir en implementaciones cloud

Conexión Power BI - Athena (ODBC)

Para la visualización se empleó Power BI Desktop conectado a Athena mediante ODBC. Esta integración permite construir dashboards consumiendo datos directamente desde AWS, sin exportaciones manuales. Dado el volumen del dataset, se aplicó muestreo aleatorio desde Athena para traer un subconjunto de 1 millón de filas a Power BI, lo que equilibra representatividad con viabilidad en un entorno local. Con ello se construyeron visualizaciones que comunican tendencias temporales, distribución de severidad y concentración geográfica.

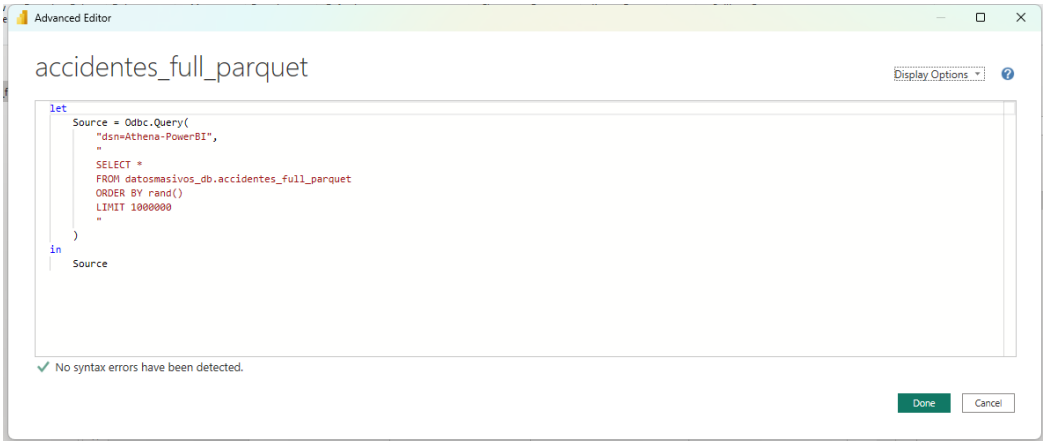
1. Se instaló el driver ODBC de Athena (Simba).
2. Se configuró DSN: Athena-PowerBI con:
    - Region: us-east-1
    - S3 Output: s3://datosmasivosulacit/athena-results/
    - Authentication: Access Key / Secret Key (IAM)



Power Query: muestreo aleatorio de 1M para no cargar 7.7M

En Advanced Editor se utilizó una consulta directa para traer un sample aleatorio:

```
let
    Source = Odbc.Query(
        "dsn=Athena-PowerBI",
        "
        SELECT *
        FROM datosmasivos_db.accidentes_parquet
        ORDER BY rand()
        LIMIT 1000000
        "
    )
in
    Source
```



Dashboard final

1. **Accidents by Year**

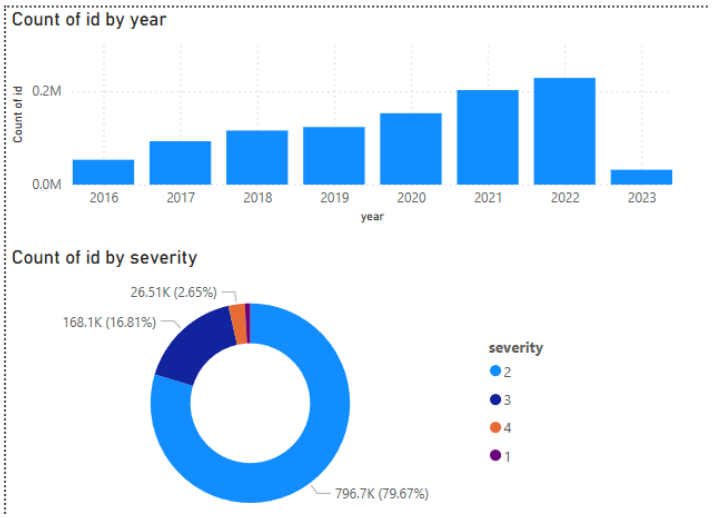
- X: year
- Y: Count(id)

2. **Accidents by Severity**

- Legend: severity
- Values: Count(id)

3. **Accidents by State**

- Location: state
- Color: Count(id)



Análisis de costos y limitaciones Free Tier

Desde la perspectiva de costos, la solución fue diseñada para minimizar consumo y mantenerse dentro de parámetros razonables del Free Tier y del modelo de pago por uso. En Amazon S3, el principal factor de costo es el almacenamiento total (dataset bruto + outputs procesados + resultados de consultas), por lo que se recomendó estructurar el bucket y evitar duplicaciones innecesarias de archivos. A nivel práctico, el dataset de 2,8 GB es manejable, pero se debe monitorear el crecimiento si se generan múltiples versiones del Parquet o si se conservan demasiados resultados de Athena.

En Amazon Athena, el costo se basa en la cantidad de datos escaneados por consulta, lo cual puede crecer rápidamente si se consulta repetidamente un CSV grande. Por ello, una decisión central del proyecto fue convertir a Parquet y aplicar particionamiento por año, de manera que consultas filtradas escaneen solo una fracción del total. Esta optimización es una buena práctica estándar en arquitecturas Big Data y reduce tanto tiempos como costos. En cuanto a AWS Glue, el uso del crawler fue acotado a ejecuciones puntuales, y el Glue Job se ejecutó de manera controlada para generar un dataset agregado, evitando ejecuciones recurrentes innecesarias.

Respecto a herramientas de visualización, aunque QuickSight puede integrarse muy bien con Athena, su activación puede implicar configuraciones adicionales o costos si se habilita en ediciones no gratuitas. Por ello, se utilizó Power BI Desktop como alternativa sin costo directo, manteniendo el objetivo de presentar un dashboard funcional conectado a la nube.

Conclusiones

El proyecto permitió implementar exitosamente un pipeline serverless completo para el procesamiento de datos masivos en la nube, integrando servicios clave del ecosistema de Amazon Web Services como Amazon S3, AWS Glue, Amazon Athena y una herramienta de visualización externa. Esta arquitectura demostró que es posible construir una solución funcional de Big Data sin la necesidad de administrar infraestructura dedicada, alineándose con las prácticas modernas de procesamiento de datos en la nube.

Asimismo, se trabajó con un dataset masivo real, con un tamaño aproximado de 2,8 GB y más de 7,7 millones de registros, lo cual representó un reto técnico significativo en términos de carga, consulta, transformación y visualización. El volumen de datos obligó a aplicar buenas prácticas propias de entornos Big Data, alejándose de enfoques tradicionales basados en herramientas locales o archivos planos.

Uno de los aprendizajes más relevantes del proyecto fue la importancia de la optimización del almacenamiento y las consultas. La conversión del dataset desde formato CSV a Parquet, junto con el particionamiento por año, permitió mejorar

notablemente el rendimiento de las consultas en Amazon Athena y reducir el volumen de datos escaneados, lo que se traduce directamente en un mejor control de costos bajo el modelo de pago por uso.

Durante el desarrollo también se evidenciaron limitaciones prácticas del entorno cloud, como los problemas presentados al intentar habilitar Amazon QuickSight en la cuenta utilizada. No obstante, esta situación permitió justificar técnicamente la adopción de Power BI Desktop como alternativa de visualización, demostrando flexibilidad en la arquitectura y la capacidad de integrar herramientas externas sin comprometer los objetivos del proyecto.

El proyecto cumple de manera integral con los requisitos planteados en el curso, incluyendo almacenamiento de datos en la nube, procesamiento y transformación de un dataset real, análisis mediante consultas, visualización de resultados relevantes y documentación técnica detallada del proceso, decisiones y dificultades encontradas.

Recomendaciones

Como trabajo futuro, se recomienda automatizar completamente el proceso ETL mediante el uso de AWS Glue Jobs o AWS Lambda, de forma que la carga y transformación de nuevos datasets pueda realizarse de manera periódica y sin intervención manual. Esto permitiría escalar la solución hacia un entorno más cercano a un sistema productivo.

Adicionalmente, es recomendable implementar mecanismos de control de costos en Amazon Athena, tales como Workgroups con límites de gasto y políticas de consulta, con el fin de prevenir consumos innecesarios y mantener el proyecto dentro de los márgenes del Free Tier o de presupuestos controlados.

Desde el punto de vista analítico, la solución podría enriquecerse mediante la creación de dashboards adicionales, incorporando análisis por mes, día de la semana, condiciones climáticas u horarios, lo cual permitiría obtener una visión más profunda del comportamiento de los accidentes y generar insights de mayor valor.

Por último, si se requiere una solución de visualización completamente integrada dentro del ecosistema AWS, se sugiere reintentar la implementación de Amazon QuickSight en un entorno con la suscripción y configuraciones adecuadas, lo que permitiría comparar su desempeño y facilidad de uso frente a herramientas externas como Power BI.