Proyecto Call 911 Baltimore

Paso 1: Alcance del Proyecto y Captura de Datos

Identificación y Captura de los Datos

Fuente de Datos

Los datos utilizados en este proyecto provienen de **Kaggle**, una plataforma reconocida por su amplio repositorio de datasets públicos. El conjunto de datos seleccionado contiene información sobre **llamadas de emergencia 911**, con más de **2.8 millones de registros**. Este volumen es adecuado para realizar pruebas de escalabilidad y procesamiento en un entorno distribuido.

Detalles del Dataset

- **Formato:** CSV
- **Columnas principales:**
    - callDateTime: Fecha y hora de la llamada.
    - priority: Prioridad del incidente (Baja, Media, Alta).
    - district: Distrito donde ocurrió el incidente.
    - description: Descripción del tipo de incidente.
    - latitude y longitude: Coordenadas geográficas del incidente.

Proceso de Captura

1. **Descarga:** El dataset se descargó desde Kaggle y se almacenó en un bucket de **Amazon S3** para integrarlo con los servicios de AWS.
2. **Automatización:** Se configuró un trigger de **AWS EventBridge** para activar una función Lambda cuando se suban nuevos archivos al bucket.
3. **Almacenamiento Inicial:** Los datos crudos se almacenan en S3 y se utilizan como fuente principal para el procesamiento ETL.

Casos de Uso Final de los Datos

Propósito Principal

El objetivo de este proyecto es **procesar, transformar y visualizar** datos de incidentes de emergencia para identificar patrones y tendencias geográficas y temporales. Esto permitirá tomar decisiones informadas y optimizar recursos.

Casos de Uso Específicos

1. **Visualización Geográfica:**
    - Representar los incidentes en un mapa interactivo en **Power BI** para identificar zonas con alta concentración de incidentes.
2. **Análisis de Prioridades:**
    - Analizar la distribución de incidentes según su prioridad para identificar los más críticos y determinar áreas que necesitan recursos adicionales.
3. **Tendencias Temporales:**
    - Evaluar patrones de incidentes en diferentes horarios y días para planificar horarios pico y optimizar la distribución de recursos.
4. **Optimización de Respuesta:**
    - Utilizar los datos como base para optimizar tiempos de respuesta y asignación de personal.
5. **Base de Datos de Verdad:**
    - Los datos transformados en **Amazon RDS** sirven como una base de datos confiable para consultas analíticas y reportes interactivos.

Paso 2: Explorar y Evaluar los Datos (EDA)

Exploración de los Datos

Problemas Identificados

Durante la exploración de los datos, se identificaron los siguientes problemas:

1. **Valores Perdidos:**
    - Columnas como latitude y longitude contienen valores nulos.

2. **Datos Duplicados:**
    - Registros duplicados identificados en la columna callNumber.
3. **Problemas de Formato:**
    - La columna location contenía valores inconsistentes, como paréntesis o espacios adicionales.

Análisis Realizado

Se utilizaron las siguientes técnicas para identificar estos problemas:

- Conteo de valores nulos por columna:

```
null_counts = spark_df.select([
    count(when(col(c).isNull(), c)).alias(c) for c in spark_df.columns
])
null_counts.show()
```

- Identificación de duplicados:

```
duplicates = spark_df.count() - spark_df.dropDuplicates().count()
print(f"Filas duplicadas: {duplicates}")
```

- Inspección de valores únicos para detectar inconsistencias en columnas clave:

```
spark_df.select("location").distinct().show(10, truncate=False)
```

Limpieza de los Datos

Pasos Implementados

1. **Relleno de Valores Nulos:**
    - Se reemplazaron valores nulos en latitude y longitude con 0 para garantizar consistencia geográfica:

```
spark_df = spark_df.fillna({"latitude": 0, "longitude": 0})
```

2. **Eliminación de Duplicados:**
    - Se eliminaron registros duplicados basados en la columna callNumber:

```
spark_df = spark_df.dropDuplicates(["callNumber"])
```

3. **Limpieza de Formato:**
    - Se corrigieron inconsistencias en la columna location eliminando paréntesis y espacios adicionales:

```
spark_df = spark_df.withColumn("location", regexp_replace(col("location"), "[(
```

4. **División de Columnas:**
    - La columna location se dividió en latitude y longitude:

```
spark_df = spark_df.withColumn("latitude", split(col("location"), ",").getItem
                   .withColumn("longitude", split(col("location"), ",").getIte
```

Diagrama del Proceso

El siguiente diagrama representa el flujo de limpieza de los datos:

1. **Carga de Datos Crudos (S3)** → 2. **Identificación de Problemas** → 3. **Relleno de Nulos y Limpieza de Formato** → 4. **Transformación Final (Glue)**

Esto asegura que los datos estén listos para la siguiente etapa del pipeline.

Paso 4: Ejecutar la ETL

Creación de Tuberías de Datos y Modelo de Datos

Tuberías de Datos

1. **Carga de Datos:**
    - Los datos son cargados desde Amazon S3 usando un DynamicFrame de AWS Glue.
2. **Transformación:**
    - Se aplican operaciones de limpieza y transformación como rellenar nulos, eliminar duplicados y dividir columnas.

3. **Almacenamiento:**
    - Los datos transformados se almacenan en Amazon RDS usando conexión JDBC.

Modelo de Datos

El modelo de datos está diseñado para ser analítico y eficiente, con las siguientes columnas principales:

- callDateTime (timestamp)
- priority (string)
- district (string)
- description (string)
- latitude (double)
- longitude (double)

Controles de Calidad

Controles Implementados

1. **Integridad en la Base de Datos Relacional:**
    - Las columnas tienen tipos de datos definidos en Amazon RDS:
        - callNumber es clave primaria para garantizar unicidad.
        - latitude y longitude son de tipo double.
2. **Comprobaciones de Fuente/Conteo:**
    - Se asegura que el número de registros cargados en RDS coincide con los datos transformados:
    - transformed_count = spark_df.count()
    - print(f"Registros transformados: {transformed_count}")
3. **Pruebas de Unidad:**

- Pruebas automatizadas verifican:
    - Que no haya valores nulos en latitude y longitude.
    - Que los duplicados hayan sido eliminados correctamente.

Diccionario de Datos

Campos Principales

Columna	**Descripción**	**Tipo de Dato**
callDateTime	Fecha y hora de la llamada	Timestamp
priority	Nivel de prioridad del incidente	String
district	Distrito donde ocurrió el incidente	String
description	Descripción breve del incidente	String
latitude	Coordenada geográfica (latitud)	Double
longitude	Coordenada geográfica (longitud)	Double

Criterio de Reproducibilidad

Requisitos

1. **Pipeline Automatizado:**
    - Los datos se procesan automáticamente al cargarse en S3 mediante AWS Glue.
2. **Script Reutilizable:**
    - El script Glue está parametrizado para ser ejecutado en diferentes entornos con ajustes mínimos.
3. **Versionamiento:**
    - Los cambios en los datos y el código son versionados para mantener el control.

Esto garantiza que los procesos puedan ser ejecutados nuevamente bajo las mismas condiciones, produciendo resultados consistentes.

Paso 5: Completar la Redacción del Proyecto

Objetivo del Proyecto

El objetivo del proyecto es **procesar y transformar un gran volumen de datos de llamadas de emergencia (911)** para generar visualizaciones interactivas que permitan identificar patrones geográficos y temporales, mejorar la asignación de recursos y optimizar la capacidad de respuesta ante incidentes críticos.

Preguntas Clave

1. ¿Cuáles son las zonas con mayor frecuencia de incidentes?
2. ¿Qué horarios tienen más llamadas de emergencia?
3. ¿Qué tipos de incidentes son más comunes según la prioridad?
4. ¿Cómo varía la distribución de incidentes entre diferentes distritos?

Elección del Modelo

Se eligió un modelo basado en una arquitectura de procesamiento distribuido con AWS Glue y Amazon RDS. Esto permite manejar grandes volúmenes de datos y transformarlos de manera eficiente para integrarlos con herramientas de visualización como Power BI.

Resultados Visualizados

Se desarrollaron dashboards en **Power BI**, como se muestra en la figura adjunta, que incluyen:

1. **Cantidad de Incidentes por Prioridad:**
    - Un gráfico de barras que muestra la distribución de incidentes según su nivel de prioridad (Medium, Low, High).
2. **Ubicación Geográfica de Incidentes:**
    - Un mapa interactivo con los puntos de incidentes registrados en Baltimore.
3. **Incidentes por Distrito:**
    - Gráfico de barras que desglosa el número de incidentes por distrito.
4. **Top Localidades con más Incidentes:**
    - Una tabla que lista las ubicaciones con mayor cantidad de incidentes.