Taller: Procesamiento y Análisis de Datos para una Aerolínea

Consigna para el Taller de Alumnos

Título: Análisis de Retrasos de Vuelos y Satisfacción de Clientes

Descripción:

En esta parte del laboratorio, aprenderemos cómo ingresar datos por evento desde una función Lambda a Redshift, teniendo en cuenta los tipos de datos que se visualizarán en el proceso usando Pandas.

Partiendo del laboratorio No. 2, que genera el archivo s3://s3-training-activity-03/staging/flight_feedback_summary_lab.csv, se deberá **crear la siguiente tabla en Redshift desde la consola del editor de consultas:**

```
CREATE TABLE public.flight_feedback_summary_lab (
    flight_number VARCHAR(10),
    origin VARCHAR(10),
    destination VARCHAR(10),
    average_delay FLOAT,
    average_rating FLOAT,
    date_insert TIMESTAMP
);
```

Contexto:

En una fase previa del pipeline se ha generado el archivo **flight_feedback_summary_lab.csv** y ahora queremos agregarle una marca de tiempo para que posteriormente el equipo de análisis de datos pueda explotar los datos junto con esta nueva métrica. Dispones del archivo:

Datos de Entrada:

- **flight_feedback_summary_lab.csv**: Contiene información sobre los retrasos en los vuelos y cómo estos afectan la satisfacción de los pasajeros.

Objetivo:

Desarrollar un proceso automatizado que procese un evento en S3 a través de una función Lambda. Este proceso debe leer el evento recibido como un DataFrame utilizando awswrangler y escribir los datos procesados en una tabla de Amazon Redshift.

Es importante tener en cuenta que la tabla ya ha sido creada previamente en Redshift por el equipo de analítica, con tipos de datos definidos de antemano.

El objetivo es analizar el nuevo evento recibido incorporando **una marca de tiempo TIMESTAMP** en el proceso, esto con el fin de generar un log de inserción en la tabla. Este nuevo dataframe debe ser almacenado en la base de datos Amazon Redshift y cumplir con las siguientes condiciones:

1. **Ejecución por evento:**
    - La inserción de datos en Redshift no debe realizarse manualmente desde la consola o AWS CLI. En su lugar, se debe configurar un trigger en AWS Lambda para que el proceso se active automáticamente cuando se cree un objeto en el bucket de S3 **s3-training-activity-03.**
    - El evento debe ser de tipo **s3:ObjectCreated:*.**

2. **Generación y escritura de datos procesados en S3 y Redshift:**
    - El DataFrame leído desde el archivo debe escribirse en la tabla **flight_feedback_summary_lab en** Redshift.
    - Debe **intentarse ajustar el código** para que los tipos de datos que maneja Pandas y Redshift sean homogéneos, permitiendo que los datos se escriban correctamente en Redshift utilizando awswrangler.

Código:
Este código debes copiarlo en tu lambda:

```
import os
import time
import boto3
import pandas as pd
from datetime import datetime
import awswrangler as wr

# Variables de entorno
S3_BUCKET = os.environ["S3_BUCKET"]
SUMMARY_KEY = os.environ["SUMMARY_KEY"]
FEEDBACK_KEY = os.environ["FEEDBACK_KEY"]
REDSHIFT_CLUSTER = os.environ["REDSHIFT_CLUSTER"]
REDSHIFT_DATABASE = os.environ["REDSHIFT_DATABASE"]
REDSHIFT_USER = os.environ["REDSHIFT_USER"]
REDSHIFT_PASSWORD = os.environ["REDSHIFT_PASSWORD"]
REDSHIFT_TABLE = os.environ["REDSHIFT_TABLE"]
REDSHIFT_SCHEMA = os.environ.get("REDSHIFT_SCHEMA", "public")

def lambda_handler(event, context):
    try:
        # Rutas de los archivos
        flight_feedback_summary_lab = f"s3://{S3_BUCKET}/{SUMMARY_KEY}"
```

```
    # Leer datos desde S3
    flights_df = wr.s3.read_csv(flight_feedback_summary_lab)
    flights_df["date_insert"] = pd.to_datetime(datetime.now())


    print(flights_df.dtypes)

    con = wr.redshift.connect_temp(
        cluster_identifier=REDSHIFT_CLUSTER, database="dev", user="awsuser"
    )

    wr.redshift.copy(
        df=flights_df,
        path=f"s3://{S3_BUCKET}/processed/temp",
        con=con,
        schema=REDSHIFT_SCHEMA,
        table=REDSHIFT_TABLE,
        iam_role=os.environ["IAM_ROLE"],
        mode="overwrite",
    )

    return {
        "statusCode": 200,
        "body": "Archivos procesados exitosamente",
    }
except Exception as e:
    return {"statusCode": 500, "body": f"Error durante el procesamiento: {str(e)}"}
```

Variables:

Variable	Valor
S3_BUCKET	s3-training-activity-03
SUMMARY_KEY	flight_feedback_summary_lab.csv
REDSHIFT_CLUSTER	redshift-cluster-training
REDSHIFT_DATABASE	dev
REDSHIFT_USER	awsuser
REDSHIFT_PASSWORD	
REDSHIFT_TABLE	flight_feedback_summary_lab
REDSHIFT_SCHEMA	public
IAM ROLE	arn:aws:iam::933263644347:role/myRedshiftRole