Análisis de LOGS

Profesor: Walter Elias

Alumnos: Luis Diego Rodriguez, Luis Alberto Rodriguez

Análisis Detallado del Proceso de ETL para Logs de Apache

Este documento detalla el proceso de Extracción, Transformación y Carga (ETL) implementado para el análisis de logs de Apache del servidor web de EPRE (Ente Provincial Regulador de la Energía). El proyecto involucra la creación de un cubo de datos dimensional utilizando Python para el procesamiento y MySQL como base de datos de destino.

1. Contexto y Objetivos del Proyecto

1.1 Fuente de Datos

- **Archivo**: `access_ssl_20230404.log`
- **Formato**: Combined Log Format de Apache

1.2 Objetivo Principal

Crear un ETL para la transformación de logs que permita:

> Análisis de patrones de tráfico web: Cantidad de logs, Bytes enviados, Dominios con más logs, Errores según el código de estado.

2.1 Herramientas Utilizadas

- **Python**: Lenguaje principal de desarrollo
- **Pandas**: Manipulación y análisis de datos
- **MySQL**: Base de datos dimensional
- **Regex**: Procesamiento de patrones de texto
- **SQLAlchemy**: Para conexión a base de datos

2.2 Estructura Dimensional para el analisis

El diseño es un esquema estrella con:

- **1 Tabla de Hechos**: tabla_hechos
- **7 Dimensiones**: tiempo, método, ruta, código_estado, dirección_ip, agente_usuario, referer

3. Proceso de Extracción

3.1 Lectura del Archivo de Log

se definió el siguiente patrón regex *r'(\S+) (\S+) (\S+) \[(.?)\] "([^"]+)" (\d{3}) (\S+) "([^"]*)" "([^"]*)"'* para parsear el archivo que contiene los logs.

3.2 Problemas Identificados y Soluciones

Problema 1: Líneas Malformadas, se tienen logs que no cumplian con el patron regex definido, se omite al saber que solo son 100 líneas aproximadamente

Problema 2: Parsing del Timestamp, el formato de esta no era compatible para hacerle una transformación a través del método datetime, por lo cual se optó por definir una función para parsear el timestamp capturando el patrón y luego reemplazandolo por un patrón que si pueda ser compatible con el método datetime un ejemplo sería el siguiente:

Entrada: "29/Jan/2023:03:50:28"
Salida: "2023-01-29 03:50:28"

4. Creación de Dimensiones y tabla de hechos

Dimensión Tiempo: Año, mes, día, Hora, Fecha

Se mantuvo la granularidad por minuto para análisis detallado

Dimensión Ruta: id_ruta, Url

Dimensión Metodo: id_metodo y metodo

Dimension Codigo_Estado: id_codigo, Estado

Dimension Direccion_ip: id_direccion_ip, direccion_ip

Dimension Agente_Usuario: id_agente_usuario, agente_usuario

Dimension Refer: id_referer, referer

Tabla de Hechos: id_tiempo, id_metodo, id_ruta, id_codigo_estado, id_direccion_ip, id_agente_usuario, id_referer; Métricas: bytes_enviados, conteo_solicitudes

5. Proceso de Carga

Se generó el DDL en el motor MySQL dado que conocemos la sintaxis en SQL para la creación del modelado del data warehouse.

5.2 Problemas de Carga y Soluciones

Problema 1: Límites de Columna, truncamiento de datos con muchos caracteres, en código Python

Problema 2: Transacciones Fallidas, Separamos la carga de datos correspondientes a cada dimensión y tabla de hechos en celdas diferentes dentro de una notebook de python para evitar perder todos los datos insertados, si falla la carga de alguna de las dimensiones o tabla de hechos.

6. Dashboard

Metabase fue seleccionado como herramienta dado que permite hacer consultas SQL además de tener una interfaz de usuario sencilla para el cálculo de métricas, tiene integración con el motor de base de datos MySQL, facilitando la extracción de los datos

7. Conclusiones

Se pudo procesar correctamente la amplia mayoría de los logs, pudiendo ser parseados y separados para su posterior análisis, se encontraron problemas durante la elaboración a la cual se le pudieron encontrar soluciones, dando como resultado un ETL que permite procesar archivos logs que tengan la misma estructura que el archivo original ser procesado y finalmente cargado dentro del datawarehouse para su posterior análisis usando la herramienta Metabase.