CODERFLEX

Data Engineering
Consigna del Proyecto Final

CODERHOUSE

ETL completo en Airflow

Para finalizar el curso y poner en práctica todos los conocimientos adquiridos te proponemos **crear un pipeline que extraiga datos de una API pública** de forma constante combinándolos con información extraída de una base de datos y colocándolos en un Data Warehouse.

Objetivos

- Crear un pipeline que extraiga datos de una API pública de forma constante combinándolos con información extraída de una base de datos (mínimamente estas 2 fuentes de datos, pero pueden utilizarse hasta 4).
- Colocar los datos extraídos en un Data Warehouse.
- Automatizar el proceso que extraerá, transformará y cargará datos cuantitativos (ejemplo estos son: valores de acciones de la bolsa, temperatura de ciudades seleccionadas, valor de una moneda comparado con el dólar, casos de covid).
- Automatizar el proceso para lanzar alertas (2 máximo) por e-mail en caso de que un valor sobrepase un límite configurado en el código.

Requisitos

Este trabajo cuenta con dos instancias:

1. Pipeline
2. Tabla

1. Pipeline

El Pipeline debe ser entregado preferiblemente en un repositorio de GitHub, de caso contrario se puede compartir como una carpeta en Google Drive. En ambos casos, la solución provista deberá tener todo lo necesario para correr en el ambiente del evaluador.

- **Script**

  El script es facilitado desde Python y deberá cumplir con estándares de calidad, como no duplicar secciones y ser comprensible para cualquiera que lo vea por primera vez.

- **Extracción de datos de una API pública**

  El proceso de extracción de los datos deberá contar mínimamente con dos fuentes de datos pero puede llegar a tener hasta cuatro. Las fuentes pueden ser: una API pública, una base de datos que debería ser instanciada con la solución entregada o un archivo estático en formato tabular.

- **Carga de datos de un Datawarehouse**

  Los datos pueden ser de procesos cambiantes, como por ejemplo: valores de acciones de la bolsa, temperatura de ciudades seleccionadas, valor de una moneda comparado con el dólar, casos de covid.

- **Transformación de datos mediante Pandas**

  El proceso de transformación de los datos tendrá que seguir un sentido lógico, que ayude a la eficientización del proceso y permita un flujo adecuado de los mismos. Es recomendable añadir columnas que puedan proporcionar información valiosa y combinar los dataframes de Pandas de diferentes fuentes en uno.

1. Pipeline

- **Mecanismos de Alerting**

  El mecanismo de alerta tiene que ser manipulable, tiene que ser flexible para poder cambiar todo tipo de configuraciones de la misma, como pueden ser los valores límite (máximo y mínimo). También cambiar otro tipo de configuraciones de la alerta (Por ejemplo, modificar las ciudades, las acciones, o demás detalles que esta emita). El mensaje recibido es ampliamente descriptivo, incluyendo todos los detalles pertinentes.

- **DAG**

  El DAG de Airflow contiene todo lo necesario para correr el script de forma eficiente. Deberá utilizar preferentemente el operador PythonOperator o en segunda instancia el BashOperator. Cada tarea del DAG deberá tener un objetivo lógico y una tarea no debería hacer muchas cosas, es decir, el DAG debería tener una separación de tareas lógica.

  En DAG debería permitir un mecanismo de backfill.

- **Contenedor de Docker**

  El Dockerfile que contiene Airflow y el DAG posee los pasos mínimos y necesarios para ejecutar Airflow de forma correcta. No hay librerias, archivos o binarios que no son utilizados en la ejecución del script. No hay pasos extras, irrelevantes o ineficientes. La imagen que genera ese Dockerfile no supera los 4GB de espacio.

2. Tabla

Este documento se presenta en formato .pdf

Debe contener las siguientes características:

- **Modelado**

  Entregar el modelado de la tabla en un script de tipo SQL para que el evaluador pueda ejecutarlo antes de la ejecución del script. La tabla donde se ingestan los datos posee una buena separación de columnas que mínimamente incluyan: una columna de fecha que registre cuándo fue extraída esa información, otra columna que posea la fecha de cuándo esa información fue generada, una o varias columnas de string donde se incluya la información relevante.

ETL completo en Airflow

Contenidos adicionales

Estos contenidos adicionales no se incluyen en los criterios de evaluación, pero si te gustaría agregar valor a tu proyecto... ¡Súmalos! Solo ten en cuenta que aquello que incluyas debe funcionar correctamente.

- Mecanismo de backfill mediante "context"
- Seteo de cluster y sort keys en la tabla
- Extracción de datos de otra base de datos o de un archivo estático

Recomendaciones

- Para cada cosa que se quiera consultar de Airflow, Docker o Pandas, es recomendable acceder a su documentación oficial consultando la versión específica que se esté usando en caso de una librería de Python.
- Si tu trabajo estará relacionado con la bolsa de valores, te recomendamos evitar el uso de la librería yfinance.
- Tener cuidado con el envío de alertas por mail, ya que la cuenta de SendGrid a utilizar sólo permite 100 emails por día.
- Usar el comando “docker image ls” para ir viendo cuánto pesa nuestra imagen de Docker, no queremos que sea muy pesada!
- Les recomendamos usar Python versión 3.8 (pueden usar igual 3.6 o 3.7).

ETL completo en Airflow

Ejemplos

Para guiarte, te compartimos algunos ejemplos de trabajos finales:

- Ejemplo Proyecto Final

Criterios de evaluación

Para la evaluación de tu Proyecto Final, tendremos en cuenta los siguientes criterios de evaluación.