Construir Pipelines de Dados ETL com PythonOperator usando Apache Airflow

Tempo estimado necessário: **90** minutos.

Cenário do Projeto

Você é um engenheiro de dados em uma empresa de consultoria em análise de dados. Você foi designado para um projeto para descongestionar as rodovias nacionais analisando os dados de tráfego rodoviário de diferentes praças de pedágio. Cada rodovia é operada por um operador de pedágio diferente com uma configuração de TI distinta que utiliza diferentes formatos de arquivo. Sua tarefa é coletar dados disponíveis em diferentes formatos e consolidá-los em um único arquivo.

Objetivos

Nesta tarefa, você desenvolverá um DAG do Apache Airflow que irá:

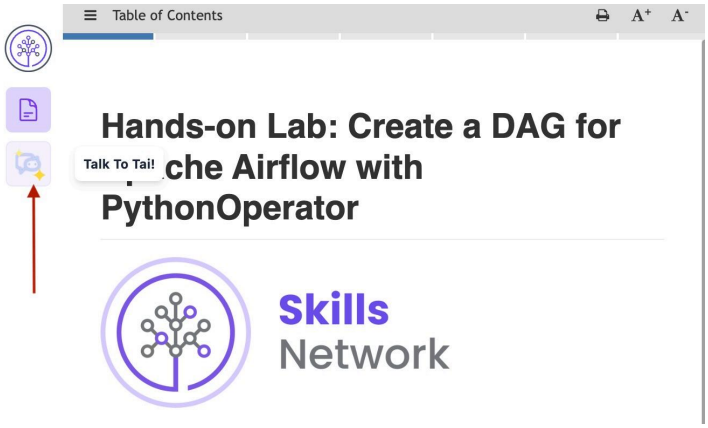
- Extrair dados de um arquivo csv
- Extrair dados de um arquivo tsv
- Extrair dados de um arquivo de largura fixa
- Transformar os dados
- Carregar os dados transformados na área de preparação

Sobre o Skills Network Cloud IDE

O Skills Network Cloud IDE (baseado no Theia e Docker) fornece um ambiente para laboratórios práticos relacionados a cursos e projetos. O Theia é um IDE (Ambiente de Desenvolvimento Integrado) de código aberto que pode ser executado em um desktop ou na nuvem. Para completar este laboratório, você usará o Cloud IDE baseado no Theia, executando em um contêiner Docker.

Aviso importante sobre este ambiente de laboratório

Por favor, esteja ciente de que as sessões para este ambiente de laboratório não são persistentes. Um novo ambiente é criado para você toda vez que você se conecta a este laboratório. Quaisquer dados que você possa ter salvo em uma sessão anterior serão perdidos. Para evitar a perda de seus dados, planeje completar esses laboratórios em uma única sessão. Você pode usar o assistente de IA `Tai` para completar essa tarefa.



Exercício 1: Preparar o ambiente de laboratório

1. Inicie o Apache Airflow.

Open Apache Airflow in IDE

   Por favor, aguarde até que o Airflow inicie completamente e esteja ativo antes de prosseguir. Se houver um erro ao iniciar o Airflow, por favor, reinicie-o.

2. Abra um terminal e crie uma estrutura de diretório para a área de preparação da seguinte forma: **/home/project/airflow/dags/python_etl/staging**.

```
sudo mkdir -p /home/project/airflow/dags/python_etl/staging
```

3. Execute os seguintes comandos para evitar problemas de permissão ao escrever nos diretórios.

```
sudo chmod -R 777 /home/project/airflow/dags/python_etl
```

```
::page{title="Exercise 2: Add imports, define DAG arguments, and define DAG"}
1. Create a file named `ETL_toll_data.py` in `/home/project` directory and add the necessary imports and DAG arguments to it.
    | Parameter | Value |
    | ----------------- | ------- |
    | owner |  <You may use any dummy name> |
    | start_date |  today |
    | email | <You may use any dummy email> |
    | retries | 1|
    | retry_delay |  5 minutes|
2. Create a DAG as per the following details.
    | Parameter | Value |
    | ----------------- | ------- |
    | DAG id  | `ETL_toll_data` |
    | Schedule | Daily once |
    | default_args | as you have defined in the previous step |
    | description | Apache Airflow Final Assignment |
::page{title="Exercise 3: Create Python functions"}
1. Create a Python function named `download_dataset` to download the data set from the source to the destination. You will call this function fr
    **Source:** https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0250EN-SkillsNetwork/labs/Final%20Assignment/tolldata.t
    **Destination:** `/home/project/airflow/dags/python_etl/staging`
2. Create a Python function named `untar_dataset` to untar the downloaded data set.
3. Create a function named `extract_data_from_csv` to extract the fields `Rowid`, `Timestamp`, `Anonymized Vehicle number`, and `Vehicle type` f
4. Create a function named `extract_data_from_tsv` to extract the fields `Number of axles`, `Tollplaza id`, and `Tollplaza code` from the `tollp
5. Create a function named `extract_data_from_fixed_width`to extract the fields `Type of Payment code` and `Vehicle Code` from the fixed width f
6. Create a function named `consolidate_data` to create a single csv file named `extracted_data.csv` by combining data from the following files:
    - `csv_data.csv`
    - `tsv_data.csv`
    - `fixed_width_data.csv`
    The final csv file should use the fields in the order given below:
    `Rowid`, `Timestamp`, `Anonymized Vehicle number`, `Vehicle type`, `Number of axles`, `Tollplaza id`, `Tollplaza code`, `Type of Payment cod
7. Create a function named `transform_data` to transform the `vehicle_type` field in `extracted_data.csv` into capital letters and save it into
::page{title="Exercise 4: Create a tasks using PythonOperators and define pipeline"}
1. Create 7 tasks using Python operators that does the following using the Python functions created in Task 2.
    1. download_dataset
      2. untar_dataset
     3. extract_data_from_csv
     4. extract_data_from_tsv
    5. extract_data_from_fixed_width
    6. consolidate_data
    7. transform_data
2. Define the task pipeline based on the details given below:
    | Task | Functionality |
    | ----------------- | ------- |
    |First task         | `download_data` |
    |Second task        | `unzip_data` |
    |Third task | `extract_data_from_csv` |
    |Fourth task        | `extract_data_from_tsv` |
    |Fifth task | `extract_data_from_fixed_width` |
    |Sixth task         | `consolidate_data` |
    |Seventh task       | `transform_data` |
::page{title="Exercise 5: Save, submit, and run DAG"}
1. Save the DAG you defined.
2. Submit the DAG by copying it into `$AIRFLOW_HOME/dags` directory.
<details>
<summary>
Click here if your DAG does not get submitted properly.
</summary>
    There might be some errors, which could stop the submission of your DAG. You can view the errors by running the following command:
```python
    erros de importação de dags do airflow list
```

3. Use CLI or Web UI to unpause the task.

4. Observe the outcome of the tasks in DAG on the Airflow console.

Solution

▶ Click here for the solution

Authors
```

Lavanya T S
Ramesh Sannareddy

Other Contributors

Rav Ahuja