Laboratório Prático: Construir Pipelines de Dados ETL com BashOperator usando Apache Airflow

Tempo estimado: **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 que visa descongestionar as rodovias nacionais analisando os dados de tráfego das diferentes praças de pedágio. Cada rodovia é operada por um operador de pedágio diferente com uma configuração de TI diferente que utiliza diferentes formatos de arquivo. Seu trabalho 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

Nota sobre capturas de tela

Ao longo deste laboratório, você será solicitado a tirar capturas de tela e salvá-las em seu dispositivo. Você precisará fazer o upload das capturas de tela para revisão por pares. Você pode usar várias ferramentas gratuitas de captura de tela ou as teclas de atalho do seu sistema operacional (Alt + PrintScreen no Windows, por exemplo) para capturar as capturas de tela necessárias. Você pode salvar as capturas de tela com a extensão `.jpg` ou `.png`.

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 tenha salvo em uma sessão anterior serão perdidos. Para evitar a perda de seus dados, por favor, planeje concluir estes laboratórios em uma única sessão.

Exercício 1: Configurar o ambiente de laboratório

1. Inicie o Apache Airflow.

Open Apache Airflow in IDE

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

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

3. Execute os seguintes comandos para conceder as permissões apropriadas às diretórias.

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

4. Baixe o conjunto de dados da fonte para o seguinte destino usando o comando `curl`.

```
sudo curl https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0250EN-SkillsNetwork/labs/Final%20Assignment/tolldata.tgz -o
```

Exercício 2: Criar imports, argumento do DAG e definição

Por favor, use o `BashOperator` para todas as tarefas nesta tarefa.

1. Crie um novo arquivo chamado `ETL_toll_data.py` no diretório `/home/project` e abra-o no editor de arquivos.
2. Importe todos os pacotes que você precisa para construir o DAG.
3. Defina os argumentos do DAG conforme os seguintes detalhes no arquivo `ETL_toll_data.py`:

Parâmetro	Valor
owner	<Você pode usar qualquer nome fictício>
start_date	hoje
email	<Você pode usar qualquer e-mail fictício>
email_on_failure	True
email_on_retry	True
retries	1
retry_delay	5 minutos

   Tire uma captura de tela do código da tarefa. Nomeie a captura de tela como `dag_args.jpg`.

4. Defina o DAG no arquivo `ETL_toll_data.py` usando os seguintes detalhes.

Parâmetro	Valor
DAG id	`ETL_toll_data`
Schedule	Uma vez por dia
default_args	Como você definiu no passo anterior
description	Tarefa Final do Apache Airflow

   Tire uma captura de tela do comando e da saída que você usou. Nomeie a captura de tela como `dag_definition.jpg`.

Ao final deste exercício, você deve ter as seguintes capturas de tela com extensão `.jpg` ou `.png`:
1. *dag_args.jpg*
2. *dag_definition.jpg*

Exercício 3: Crie as tarefas usando BashOperator

1. Crie uma tarefa chamada `unzip_data` para descompactar os dados. Use os dados baixados na primeira parte desta tarefa em `Configurar o ambiente de laboratório` e descompacte-os no diretório de destino usando `tar`.

   Tire uma captura de tela do código da tarefa. Nomeie a captura de tela como `unzip_data.jpg`.

   Você pode descompactar localmente e ler o arquivo `fileformats.txt` para entender os detalhes das colunas.

2. Crie uma tarefa chamada `extract_data_from_csv` para extrair os campos `Rowid`, `Timestamp`, `Número de veículo anonimizado` e `Tipo de veículo` do arquivo `vehicle-data.csv` e salve-os em um arquivo chamado `csv_data.csv`.

   Tire uma captura de tela do código da tarefa. Nomeie a captura de tela como `extract_data_from_csv.jpg`.

3. Crie uma tarefa chamada `extract_data_from_tsv` para extrair os campos `Número de eixos`, `ID da praça de pedágio` e `Código da praça de pedágio` do arquivo `tollplaza-data.tsv` e salve-o em um arquivo chamado `tsv_data.csv`.

   Tire uma captura de tela do código da tarefa. Nomeie a captura de tela como `extract_data_from_tsv.jpg`.

4. Crie uma tarefa chamada `extract_data_from_fixed_width` para extrair os campos `Código do tipo de pagamento` e `Código do veículo` do arquivo de largura fixa `payment-data.txt` e salve-o em um arquivo chamado `fixed_width_data.csv`.

   Tire uma captura de tela do código da tarefa. Nomeie a captura de tela como `extract_data_from_fixed_width.jpg`.

5. Crie uma tarefa chamada `consolidate_data` para consolidar os dados extraídos das tarefas anteriores. Esta tarefa deve criar um único arquivo csv chamado `extracted_data.csv` combinando dados dos seguintes arquivos:

    - `csv_data.csv`
    - `tsv_data.csv`
    - `fixed_width_data.csv`

    O arquivo csv final deve usar os campos na ordem abaixo:

    - `Rowid`
    - `Timestamp`
    - `Número de veículo anonimizado`
    - `Tipo de veículo`
    - `Número de eixos`
    - `ID da praça de pedágio`
    - `Código da praça de pedágio`
    - `Código do tipo de pagamento`, e
    - `Código do veículo`

    Dica: Use o comando bash `paste` que mescla as colunas dos arquivos passados como parâmetro de linha de comando e envia a saída para um novo arquivo especificado. Você pode usar o comando `man paste` para explorar mais.

Exemplo: `paste file1 file2 > newfile`

Tire uma captura de tela do código da tarefa. Nomeie a captura de tela como `consolidate_data.jpg`.

6. Crie uma tarefa chamada `transform_data` para transformar o campo vehicle_type no `extracted_data.csv` em letras maiúsculas e salve-o em um arquivo chamado `transformed_data.csv` no diretório de staging.

    Dica: Você pode usar o comando `tr` dentro do BashOperator no Airflow.

Tire uma captura de tela do código da tarefa. Nomeie a captura de tela como `transform.jpg`.

7. Defina o pipeline de tarefas conforme os detalhes abaixo:

Tarefa	**Funcionalidade**
Primeira tarefa	`unzip_data`
Segunda tarefa	`extract_data_from_csv`
Terceira tarefa	`extract_data_from_tsv`
Quarta tarefa	`extract_data_from_fixed_width`
Quinta tarefa	`consolidate_data`
Sexta tarefa	`transform_data`

Tire uma captura de tela da seção do pipeline de tarefas do DAG. Nomeie a captura de tela como `task_pipeline.jpg`.

Ao final deste exercício, você deve ter as seguintes capturas de tela com extensão `.jpg` ou `.png`:

1. *unzip_data.jpg*
2. *extract_data_from_csv.jpg*
3. *extract_data_from_tsv.jpg*
4. *extract_data_from_fixed_width.jpg*
5. *consolidate_data.jpg*
6. *transform.jpg*
7. *task_pipeline.jpg*

Exercício 4: Colocando o DAG em operação

1. Envie o DAG. Use a CLI ou a interface da Web para mostrar que o DAG foi enviado corretamente.
   Tire uma captura de tela mostrando que o DAG que você criou está na lista de DAGs. Nomeie a captura de tela como `submit_dag.jpg`.

   Nota: Se você não encontrar seu DAG na lista, pode verificar se há erros usando o seguinte comando no terminal:

   ```
   airflow dags list-import-errors
   ```

2. Retome e acione o DAG através da CLI ou da interface web.

3. ***Tire uma captura de tela*** do DAG retomado na CLI ou na GUI. Nomeie a captura de tela como `unpause_trigger_dag.jpg`.

4. ***Tire uma captura de tela*** das tarefas na execução do DAG através da CLI ou da interface web. Nomeie a captura de tela como `dag_tasks.jpg`.

5. ***Tire uma captura de tela*** das execuções do DAG no console do Airflow através da CLI ou da interface web. Nomeie a captura de tela como `dag_runs.jpg`.

Lista de verificação de capturas de tela

Você deve ter as seguintes capturas de tela com extensão `.jpg` ou `.png`:

1. *submit_dag.jpg*
2. *unpause_trigger_dag.jpg*
3. *dag_tasks.jpg*
4. *dag_runs.jpg*

Essa ação conclui a tarefa.

Authors

Lavanya T S
Ramesh Sannareddy

Other Contributors

Rav Ahuja