CondoManage

Manual de intruções de como instalar o projeto em máquina local

Authors

- @IDevKennedyMoreira

🔗 Links

MY PORTFOLIO

LINKEDIN

Local Deployment

Para deploy deste projeto em máquina local é necessário efetuar algumas ressalvas, este projeto foi desenvolvido em sistema operacional MacOs e os passos de instalação podem mudar de acordo com sistema operacional, aqui basearei apenas a instalação em MacOs pois não possuo muita familiaridade com outros sistemas operacionais e não saberia como descrever o passo a passo nesse momento.

Instalando o Java versão 11.

```
brew install openjdk@11
```

Instalando o Apache Spark.

```
brew install apache-spark
```

Instalando o Python.

```
brew install python3
```

Instalar o virtualenv na máquina local.

```
pip install virtualenv
```

Abrir projeto e criar seu ambiente virtual.

```
virtualenv venv
```

Abrir projeto e iniciar o ambiente virtual.

```
source env/bin/activate
```

Instalar os requirements.

```
pip install -r requirements.txt
```

Definir variável de ambiente do airflow.

```
export AIRFLOW_HOME=~/Desktop/case_super_logica
```

Para executar o projeto em modo local sem airflow via terminal.

```
cd dags;python3 superlogica_data_pipeline_local.py
```

OBS! O motivo de rodar a partir da pasta de dags é que no início do projeto
não pensei que teria problemas em rodar a pipe com o airflow com scripts de

execução paralela em modo local, porém após consultar a documentação vi essa restrição mas o projeto já estava todo montado a partir desse ponto de montagem.

Para executar o projeto em modo local usando o airflow

```
airflow standalone
```

logue com usuário admin e senha presente no arquivo standalone_admin_password

OBS! Você não conseguirá rodar scripts em paralelo como o presente em nossa DAG sem instalar um banco de dados local e alterar o airflow.cgf para sua configuração de banco escolhida. Devido a esse trabalho recomendo fortemente o uso da execução via terminal presente no passo anterior.

Questão 1

A arquitetura de datalake escolhida para a criação dessa projeto foi a medalhão aqui temos:

Camada landing apenas recebendo os arquivos csv para processamento
camada raw apenas transforma os arquivos da camada landing em formato parquet;

camada refined aplica regras de negócio e gera a OBT (One Big Table) modalagem escolhida por mim para este projeto;

camada trusted é uma cópia da camada refined com dados prontos para análise.

Questão 2

A ingestão de dados desse projeto foi criada a partir de uma simulação da criação de arquivos csv a classe responsavel por isso é a DataGenerator presente no arquivo dags/models/datagenerator.py ela é orquestrada a partir de dags/landing_read_data_from_external.py e por sua vez os dados entram na camada raw através de dags/raw_read_data_from_landing.py a explicação de como cada arquivo funciona está em seus comentários internos.

Questão 3

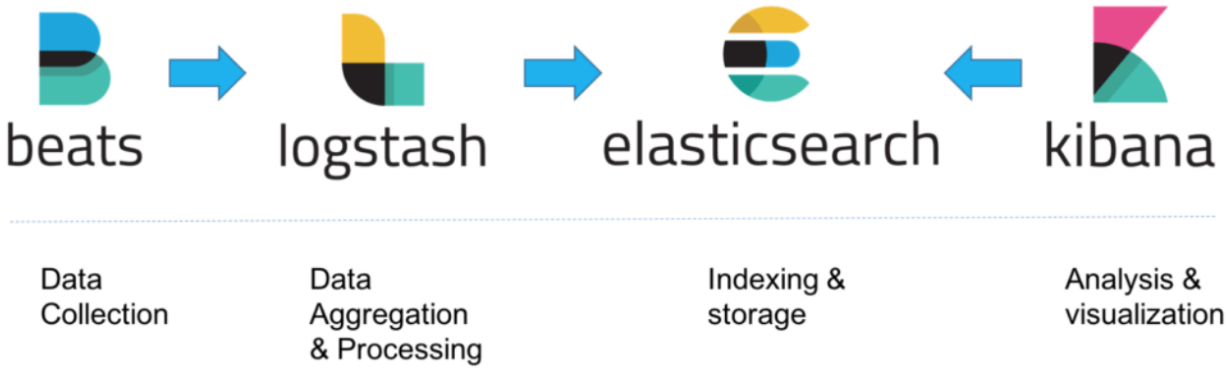
Questão 4

Vide comentários internos do arquivo dags/raw_streaming_read_data_from_landing_townhouse.py lá é feita a primeira parte da ingestão usando streaming, porém pipeline exposta na DAG e no arquivo dags/superlogica_data_pipeline_local.py é do tipo batching.

Questão 5

Para definicão de documentação usaria um repositório de documentação como o confluence e também geraria comentário seguindo a PEP8 e com a ferramenta mkdocs conseguiria fazer a documentação

automática
a partir dos cometários nos jobs.
Sobre as questões envolvendo logging e monitoramento decidiria por usar a ELK stack tendo assim o Beatsfile
efetuando pooling em arquivos de dados e logs presentes na ferramenta de orquestração como o Airflow por sua vez o Beatsfile aciona o Logstash com a mensageria para que ele definir o melhor índice do ElasticSearch para inserir essa mensagem, por fim teriamos a liberdade de definir boards no Kibana com os logs propostos.
Para enriquecimento dos logs poderiamos criar uma classe de log com um decorator de logging onde este por sua vez persistiria a mensagem em arquivos de logs de cada job e também colocaria o ELK stack para enxergar estes mesmos logs criando assim uma solução de monitoramento robusta.



Badges

Licença:

License MIT