Sumário

O teste prático consiste na implementação de um pipeline de ETL simplificado no envolvendo dois bancos de dados (Fonte e Alvo), uma API isolando o banco de dados fonte e um script de ETL para transferir dados de um para o outro.

Deverão ser criados dois bancos de dados, Fonte e Alvo, ambos em `postgresql`. O banco de dados Fonte deverá ser acessado através de uma API escrita em `fastapi`. O script de ETL deve acessar a API usando a biblioteca `httpx` e escrever os resultados no banco de dados Alvo utilizando a biblioteca `sqlalchemy` (e, opcionalmente, `pandas`).

Banco de Dados Fonte

Criar um banco de dados `postgresql`. banco deverá conter as seguintes tabelas e colunas:

- data
  - timestamp
  - wind_speed
  - power
  - ambient_temprature

Inserir dados aleatórios nela com frequência 1-minutal e intervalo de 10 dias. Especificar na entrega do teste o período de dados contido no banco.

Banco de Dados Alvo

Criar outro banco de dados `postgresql`. O banco deverá conter as seguintes tabelas e colunas:

- signal
  - id
  - name
- data
  - timestamp
  - signal_id
  - value

Esse banco deverá ser criado utilizando-se a biblioteca `sqlalchemy`. Cabe ao candidato estabelecer relações apropriadas entre as tabelas, bem como inserir dados auxiliares necessários a execução do processo de ETL.

Conector

Criar uma API em `fastapi` para expor dados do banco de dados Fonte.

Deverá ser implementada uma rota que permita a consulta aos dados da tabela `data`, filtrados por intervalo de tempo. A rota deverá permitir a seleção de uma ou mais variáveis a serem retornadas.

A API poderá conter rotas-extras.

ETL

Escrever um script em python para executar o processo de ETL:

- Recebe uma recebe uma data como input,
- Consulta dados para variáveis wind_speed e power via API para o dia daquela data. O script deverá se consultar a API utilizando a biblioteca `httpx`.
- Agrega o dado 10-minutal com agregações de média, mínimo, máximo e desvio padrão. A transformação de dados pode ser implementada com qualquer biblioteca, desde que ela seja executada de forma eficiente. Recomenda-se a utilização do `pandas` ou similar.
- Salva o dado no banco de dados Alvo. A escrita no banco de dados deverá utilizar a biblioteca `sqlalchemy` para se conectar ao banco. A escrita do dado no banco pode ser feita com qualquer tecnologia, mas recomenda-se o uso do `pandas` em conjunto com o `sqlalchemy`.

Bonus: Dagster

Orquestrar o script de ETL utilizando o `dagster`. Implementar:

- Recurso para acessar banco de dados Fonte.
- Recurso para acessar banco de dados Alvo.
- Asset particionado diário para executar o processo de ETL.
- Job e Schedule.

Essa etapa **não é obrigatória**.

Entrega

A entrega deverá ser feita através de um repositório no `github` com instruções de como .rodar os bancos de dados, a API e o script. Recomenda-se a utilização de `docker` e `docker-compose`. O repositório deverá conter um `README.md` mínimo descrevendo quais comandos devem ser executados para rodar o programa.