Live Case

AGENDA

1. Visão geral da stack iFood (5’)
2. Solução das tarefas (15’ cada tarefa)
3. Comentários finais (5’)

Visão Geral

Stack iFood (simplificada)

Tarefa 1

Tarefa 1 - Planejamento de ETL para a Empresa XYZ

Contexto

A Empresa XYZ deseja otimizar seu processo de ETL para lidar com o crescente volume de dados de vendas, dados de produtos obtidos de fontes externas (arquivos ou algoritmos criados para a coleta de dados), e interações de usuários nas redes sociais. Hoje, existem produtos de dados que utilizam todas essas fontes. Atualmente, o processo é lento e propenso a erros, afetando a disponibilidade e a qualidade dos dados para análise.

Como você descreveria, de forma conceitual, como planejar e estruturar o processo de ETL para o cenário apresentado?

Supondo que o processo de ETL esta lento porque tanto a coleta quanto a carga em si estao descentralizados, alem da falta de testes (tipagem dos dados, validacao de schema, etc) o que gera um custo de manutençao e mapeamento alto (falta de rastro/observaçao dos dados).

Creio que uma estrutura de ETL escalavel e que sane os erros apresentados seja:

Extraçao: utilizar funcoes/classes em python para abstrair ao maximo as fontes de dados, em que a funcao possua argumentos que possibilitem a troca mais simples de origem de dados. Junto ao coletor, é válda a criaçao de um schema/validacao de schema e tipagem para os dados que entram. Caso haja algum problema, o processo ja quebra nessa fase. Tambem nessa fase, preservaremos os dados brutos como estao, para acesso em caso de problema. A ferramenta utilizada é o databricks para as extraçoes e aws para armazenamento.

Transformaçao: pode ser feita tanto em sql quanto em python. Nessa etapa, sao feitos joins entre tabelas, tratativa de valores nulos, replaces, organizaçao de nomes de colunas (camel case ou snake case), alem de agregacoes dos dados para consumo do time de BI/analistas de negocios para dashboards. Aqui tambem podem ser feitos testes para cada tipo de transformaçao caso python seja usado (cada funcao do python teria seu teste). A ferramenta utilizada é o databricks para as extraçoes e aws para armazenamento.

Carga:

Tarefa 1 - Planejamento de ETL para a Empresa XYZ

Resposta

- xxx

Tarefa 2

Tarefa 2 - Otimização de Queries

Contexto

Suponha que você tenha um DataFrame PySpark df_vendas com as seguintes colunas: data_venda, produto_id, quantidade, preco_unitario, e categoria. Você deseja analisar o total de vendas por categoria para o último trimestre, mas a consulta está demorando para ser executada.

Você deve analisar a query e propor modificações para otimizá-la, explicando o raciocínio por trás de suas escolhas.

```
python
from pyspark.sql import functions as F

df_vendas \
    .filter(F.col("data_venda") >= "2023-10-01") \
    .groupBy("categoria") \
    .agg(F.sum(F.col("quantidade") * F.col("preco_unitario")).alias("total_vendas")) \
    .orderBy(F.desc("total_vendas")) \
    .show()
```

Tarefa 2 - Otimização de Queries

Resposta

```
python
from pyspark.sql import functions as F

df_vendas \
    .filter(F.col("data_venda") >= "2023-10-01") \
    .groupBy("categoria") \
    .agg(F.sum(F.col("quantidade") * F.col("preco_unitario")).alias("total_vendas")) \
    .orderBy(F.desc("total_vendas")) \
    .show()
```

Para otimizaçao da query, creio que ordenacao dos dados pode ser feita no final do ETL. Nesse momento, uma ordenacao/agregacao de valores pode pesar a query. Minha solucao seria algo nesse sentido: (escrito em sql).

Select

Qtd * precounit as total_vendas

Where data_venda >= ‘2023-10-01’

Tarefa 3

Tarefa 3 - Resolução de Problemas

Contexto

A Empresa XYZ identificou que, em sua base de dados de vendas, existem inconsistências nos registros de vendas devido a erros no processo de captura de dados. Alguns registros apresentam preco_unitario como negativo e quantidade maior que o estoque disponível no momento da venda, o que é logicamente impossível.

Como você propõe uma estratégia para identificar e corrigir esses problemas dentro do processo de ETL?

Julgando que o processo esta sendo falho na extraçao dos dados, acho interessante fazer testes nessa etapa do ETL, testes especificamente para validacao de tipos de dados (coluna x tem um int). Nesse caso, a validacao da coluna quantidade deveria esperar apenas um numero positivo de preco_unit, o que nao aconteceu. Por isso, o preço unitario ficou negativo e dependendo do metodo de movimentaçao do estoque, tambem negativo. Testes para dados faltantes tambem (contagem de linhs origem x extracao)

Tarefa 3 - Resolução de Problemas

Resposta

xxx

Comentários Finais