

Documentação - Teste Técnico Minsait - Engenheiro de Dados Pleno

Introdução

Este documento descreve a solução desenvolvida para o Teste Técnico de Engenheiro de Dados Pleno na Minsait. Com o objetivo de processar dados em formato JSON no padrão HL7 FHIR, armazená-los em um banco de dados ClickHouse e permitir consultas rápidas através do Superset. O ambiente foi preparado para processar milhões de dados em segundos.

Tecnologias Utilizadas (Somente Open-Source)

- **Python** (Processamento de dados)
- **PySpark** (ETL e manipulação dos dados)
- **ClickHouse** (Banco de dados para armazenamento e consulta)
- **JDBC** (Conexão entre PySpark e ClickHouse)
- **Superset** (Interface de visualização e consultas SQL)

Estrutura do Projeto

1. Importação de bibliotecas e configuração da sessão Spark
2. Leitura e transformação dos arquivos JSON
3. Processamento dos dados
 - Pacientes
 - Medicamentos
 - Condições médicas
4. Armazenamento no banco de dados ClickHouse
5. Integração com Superset para consultas e visualizações
6. Materialized Views para otimização de consultas
7. Visualização e consultas realizadas via Superset e ClickHouse

Seção 1: Importação de Bibliotecas e Configuração da Sessão Spark

A primeira parte do código configura uma sessão Spark para realizar o ETL dos arquivos JSON e conexão com o banco de dados ClickHouse via JDBC.

```
spark = SparkSession.builder \  
    .appName("ETL para ClickHouse via JDBC") \  
    .config("spark.sql.shuffle.partitions", "100") \  
    .config("spark.executor.memory", "4g") \  
    .config("spark.driver.memory", "4g") \  
    .config("spark.sql.execution.arrow.pyspark.enabled", "true") \  
    .config("spark.jars", "/opt/spark/jars/clickhouse-jdbc-0.4.6-shaded.jar") \  
    .getOrCreate()
```

A sessão Spark é configurada para otimizar o uso de memória e processamento, além de habilitar o uso de Arrow para melhor desempenho.

Seção 2: Leitura e Transformação dos Arquivos JSON

Os arquivos JSON são lidos do diretório especificado e carregados em um DataFrame Spark, sendo cacheados para melhor performance.

```
DATA_PATH = "data/*.json"
df = spark.read.option("multiline", "true").json(DATA_PATH).cache()
df = df.withColumn("entry_exploded", explode_outer(col("entry"))) \
    .select("entry_exploded.resource.*")
```

A coluna entry_exploded é expandida para normalizar os dados e facilitar o processamento.

Seção 3: Processamento dos Dados

O script Python é dividido em três funções principais para processamento dos dados:

3.1 process_patients()

- Processa os dados de pacientes, extraindo informações como nome, gênero, telefone, endereço e SSN.
- Normaliza os nomes e limpa dados desnecessários para garantir integridade.
- Usa explode_outer() para manipular arrays contidos no JSON.

3.2 process_medications()

- Identifica medicamentos prescritos, dosagens, rotas de administração e frequência de uso.
- Limpa strings e normaliza dados para facilitar a busca e agregação.
- Inclui lógica para capturar dados não estruturados e padronizá-los.

3.3 process_conditions()

- Processa informações sobre condições médicas, normalizando nomes e mantendo o status clínico associado.
- Os nomes das condições são padronizados usando regex e funções de normalização de strings.

Seção 4: Modelagem do Banco de Dados

A modelagem do banco de dados no ClickHouse foi projetada para otimizar o desempenho das consultas.

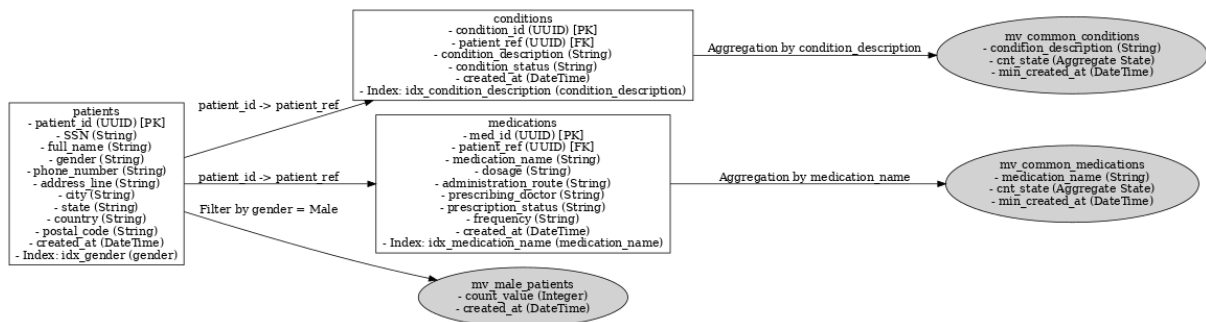
Particionamento e Ordenação

- **Particionamento:** Uso de PARTITION BY toYYYYMM(created_at) para distribuição dos dados por mês.
- **Ordenação (ORDER BY):** Chaves primárias (patient_id, condition_id, med_id) utilizadas para otimizar operações de leitura e escrita.

Índices Bloom Filter

- Índices criados para as tabelas principais (patients, conditions, medications) permitem filtragem rápida de dados através de partições.

Diagrama da Modelagem



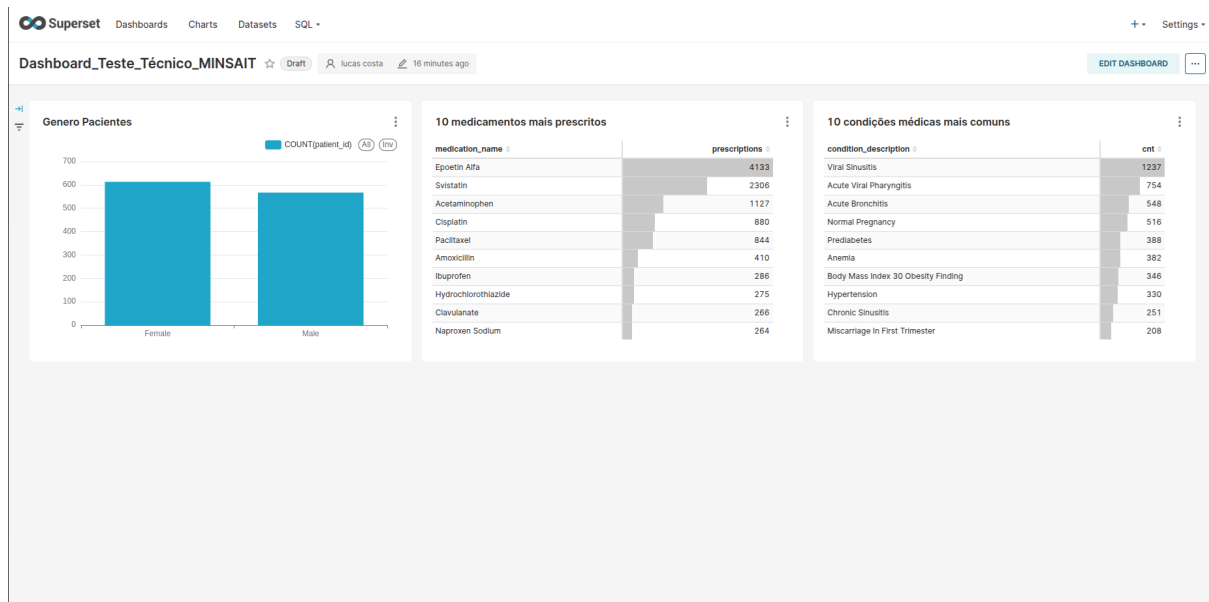
Seção 5: Materialized Views

Três Materialized Views foram criadas para acelerar consultas:

- **mv_common_conditions**: Calcula as condições médicas mais comuns.
- **mv_male_patients**: Conta o número de pacientes do gênero masculino.
- **mv_common_medications**: Calcula os medicamentos mais prescritos.

Seção 6: Integração com Superset

O Superset foi utilizado para criar dashboards e executar consultas SQL nas Materialized Views. Abaixo estão os resultados obtidos:



- Consultas SQL no Apache Superset:

- Pacientes masculinos:

The screenshot shows the Apache Superset SQL editor interface. The left sidebar contains the 'DATABASE' dropdown set to 'clickhouse', 'ClickHouse', and the 'SCHEMA' dropdown set to 'Select schema or type to search schemas'. The main editor area displays the following SQL query:

```
1 SELECT sum(count_value) AS male_patients_count
2 FROM healthcare.mv_male_patients;
```

Below the query editor, the 'RESULTS' tab is active, showing a single row of results:

male_patients_count
567

The interface also includes a 'QUERY HISTORY' tab, a 'CREATE CHART' button, a 'DOWNLOAD TO CSV' button, and a 'COPY TO CLIPBOARD' button. The 'LIMIT' is set to 1000, and the execution time is 00:00:00.28.

- Medicamentos mais prescritos:

The screenshot shows the Apache Superset SQL editor interface. The left sidebar contains the 'DATABASE' dropdown set to 'clickhouse', 'ClickHouse', and the 'SCHEMA' dropdown set to 'Select schema or type to search schemas'. The main editor area displays the following SQL query:

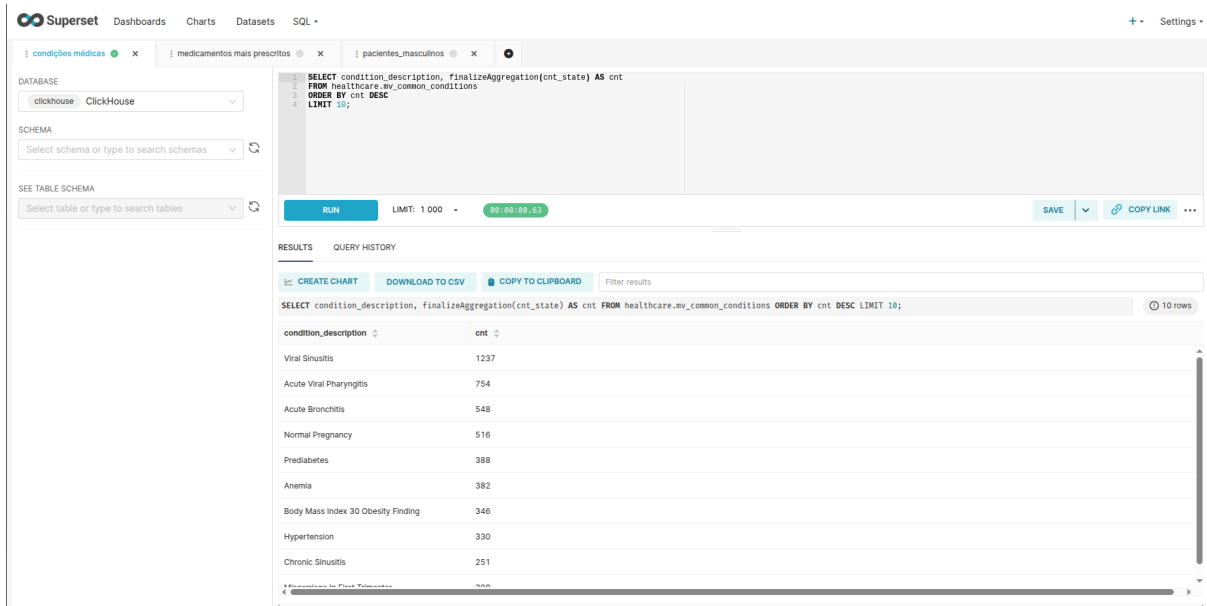
```
1 SELECT medication_name, finalizeAggregation(cnt_state) AS prescriptions
2 FROM healthcare.mv_common_medications
3 ORDER BY prescriptions DESC
4 LIMIT 10;
```

Below the query editor, the 'RESULTS' tab is active, showing 10 rows of results:

medication_name	prescriptions
Epoetin Alfa	4133
Statins	1862
Acetaminophen	883
Cisplatin	762
Pacitaxel	737
Amoxicillin	331
Statins	325
Ibuprofen	245
Hydrochlorothiazide	221
Clavulanate	217

The interface also includes a 'QUERY HISTORY' tab, a 'CREATE CHART' button, a 'DOWNLOAD TO CSV' button, and a 'COPY TO CLIPBOARD' button. The 'LIMIT' is set to 1000, and the execution time is 00:00:00.46.

- Condições médicas mais comuns:



The screenshot shows the Apache Superset web interface. On the left, there are navigation tabs for 'condições médicas', 'medicamentos mais prescritos', and 'pacientes masculinos'. The 'condições médicas' tab is active. Below the tabs, there are dropdown menus for 'DATABASE' (set to 'clickhouse'), 'SCHEMA' (with a search input), and 'SEE TABLE SCHEMA' (with a search input). The main area displays a SQL query in a text editor:

```
1 SELECT condition_description, finalizeAggregation(cnt_state) AS cnt
2 FROM healthcare_mv_common_conditions
3 ORDER BY cnt DESC
4 LIMIT 10;
```

Below the query editor, there are buttons for 'RUN', 'LIMIT: 1 000', 'SAVE', 'COPY LINK', and a status indicator '00:00:00.63'. The 'RESULTS' tab is selected, showing a table with 10 rows. The table has two columns: 'condition_description' and 'cnt'. The data is as follows:

condition_description	cnt
Viral Sinusitis	1237
Acute Viral Pharyngitis	754
Acute Bronchitis	548
Normal Pregnancy	516
Prediabetes	388
Anemia	382
Body Mass Index 30 Obesity Finding	346
Hypertension	330
Chronic Sinusitis	251

Conclusão

A adoção do ClickHouse, com um particionamento bem estruturado, ordenação otimizada e uso eficiente de Materialized Views, garantiu um processamento ágil e eficaz de grandes volumes de dados. A integração com o Superset proporcionou uma interface intuitiva e eficiente, permitindo a criação de dashboards e a execução de consultas rápidas. Essa arquitetura foi projetada para oferecer uma experiência robusta para analistas de BI e cientistas de dados, facilitando a análise agregada e a visualização de informações de forma clara e acessível.