Relatório

Carlos Martins, 18836

João Azevedo, 18845

Barcelos, Fevereiro 2023

Contents

# Introduction

Uma empresa que comercializa bicicletas numa cadeia de lojas de mobilidade sustentável, está a mudar o seu sistema de dados de base de dados relacionais on-premise, para Big Data. Além dos dados novos pretende integrara no seu novo sistema os dados históricos de vendas, que foram exportados dos sistemas atuais em formato csv. O armazenamento de dados será feito em HDFS, e será utilizado o Hive para gestão de dados.

## Project objectives

Dada a introdução feita no tópico anterior (1) pretendemos na elaboração deste projeto ter em conta os seguintes aspetos:

* Modelação dos dados
* Garantir o melhor formato para armazenamento dos dados
* Na implementação em Hive garantir a melhor performance
* Implementar análises de dados em Zeppelin

# Modelação de Dados

Nesta secção será apresentado o modelo de dados escolhido após a análise dos dados.

Após a análise do csv principal onde continha os dados todos agrupados foi decidido separar os dados em vários csv’s. Cada cvs representa uma tabela como mostra a figura 1. Com base na figura 1 podemos visualizar que temos uma tabela product que representa os produtos existentes e que podem ser comprados, a tabela de customer onde temos os clientes, a tabela de salesorder é onde temos as compras feitas pelos clientes e a salesorderdetail onde temos detalhes ao nível da compra de cada cliente como por exemplo: o produto que comprou e o seu custo.

Figura - Modelo de dados

Em suma, no modelo apresentado acima é possível identificar 4 entidades:

* Product
* Customer
* SalesOrder
* SalesOrderDetail

Para garantir um bom desempenho do sistema foi definido guardar as strings com o tipo de dados string, tinyint para armazenar números inteiro pequenos, com uma faixa de valores -128 a 127 como é o caso das colunas \*+\* da tabela \*, smallint para armazenar números inteiros com uma faixa de valores de -32768 a 32767 como por exemplo: coluna \*+ tabela +, float para armazenar números flutuantes com precisão de até 7 dígitos decimais como por exemplo: coluna \*\* tabela \*\*, em relação a dados que envolvam datas e horas foi escolhido o tipo de dados timestamp uma vez que é o único projetado para esse efeito.

# Armazenamento de dados em Hive

O Hive suporta vários formatos de arquivos, cada um com suas próprias características, vantagens e usos adequados. Um dos formatos suportados é o **Text File Format**. este é o formato de arquivo mais simples suportado por Hive. Os arquivos de texto são armazenados como arquivos de texto simples no HDFS. Este formato é mais adequado para projetos de pequena escala ou para dados não estruturados. O **SequenceFile Format** é usado para armazenar séries de pares chave-valor. É mais adequado para dados com grande quantidade de dados, pois é mais eficiente em termos de espaço de armazenamento e desempenho de leitura. O **RCFile** (Record Columnar File) é semelhante ao SequenceFile, mas é otimizado para colunas em vez de linhas. Este formato é mais adequado para dados estruturados com muitas colunas, pois é mais eficiente em termos de armazenamento e desempenho de consulta. O **ORC** (Optimized Row Columnar) File é o formato de arquivo mais avançado suportado por Hive. Este formato é otimizado para consultas de colunas e tem muitas outras melhorias de desempenho, como compressão de dados e suporte para dados nulos. Este formato é mais adequado para projetos de grande escala com dados estruturados. Por fim, temos o **Parquet File** é outro formato de arquivo avançado suportado por Hive. É semelhante ao ORC, mas é projetado para ser utilizado com vários sistemas de base de dados, incluindo Hive, Impala, Spark, etc. Este formato é mais adequado para projetos que exigem colaboração entre sistemas de base de dados.

Após a análise dos formatos indicados acima foi escolhido como tipo de formato final o ORC por garantir um bom desempenho do sistema.

# Implementação do Modelo

# Upload dos Dados

Para efetuar o upload de um arquivo csv no Hive seguimos os seguintes passos:

1. Criar uma tabela no Hive de forma temporária com colunas correspondentes aos dados do arquivo csv.
2. Copiar o arquivo para o HDFS
3. Utilizar o comando LOAD DATA INPATH no Hive para carregar os dados do arquivo csv para a tabela temporária.
4. Criar a uma tabela com o formato ORC
5. Inserir os dados da tabela temporária para a tabela final com um INSERT INTO TABLE

O código \* mostra a criação da tabela temporária este processo foi efetuado para 4 as entidades, após isto criamos duas pastas na diretoria …. E fizemos upload dos ficheiros csv para a diretoria … a criação das duas pastas e não só uma deveu-se ao facto de querermos ter uma separação entre tabelas finais e tabelas temporárias, as tabelas finais encontram-se na pasta \*\* enquanto que na pasta \*\* estão as tabelas temporárias. Através do código \* procedemos ao import dos dados do csv para as tabelas temporárias, após isto criamos as tabelas finais com o formato de ORC, por fim fizemos um insert dos dados das tabelas temporárias para as tabelas finais.

# Implementação e análise das querys

# Implementação e análise das querys em Zeppelin