**Exemplo de Parquet com Pandas e Spark**



Apache Parquet é um formato de armazenamento disponível em projetos do ecossistema Hadoop.

A primeira versão é de 2013, parte de um projeto de parceria entre o Twitter e a Cloudera. Desde 2015 o projeto faz parte da Apache Software Foundation.

O Parquet armazena os dados de forma colunar plana. Esse modelo possui alguns benefícios:

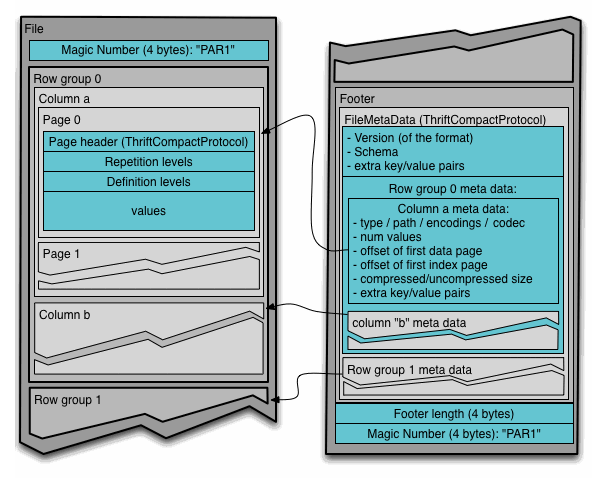
* É mais fácil compreender dados por colunas
* Algoritmo de compressão pode ser especificado por coluna
* Queries que usam várias colunas são mais eficientes

Parquet é excelente para trabalhar com estruturas de dados complexas em massa. Quem já atuou com outros formatos, de cara sentem a diferença em trabalhar com Parquet quando falamos em teras de dados. Isso graças ao algorítimo record shredding and assembly algorithm descrito no Dremel paper da Google e implementado como parte do *core* do Parquet.

Basicamente a divisão do Parquet está em:

* Dados
  + Row group
  + Column chunk
  + Page
* Metadados
  + Versão
  + Columns
  + Other

Abaixo temos um exemplo da estrutura de um arquivo Parquet:



Parquet reduz o espaço de armazenamento em 87%. É possível utilizá-lo em diversas ferramentas do ecossistema Hadoop, Spark, Hive, Impala, Pig...

**Prática**

**Pandas**

Com essa biblioteca para análise e manipulação de dados em Python, vamos converter uma lista de dicionários, salvar no formato Parquet e carregar o arquivo gerado para conferência.

# Importação das bibliotecas

**import** pandas **as** pd

# Criação de DataFrame com dados

pessoas\_df **=** pd**.**DataFrame([

{"nome":"Patricia", "idade": 15},

{"nome":"Letícia", "idade":30},

{"nome":"Marcelo", "idade":19},

{"nome":"Adriane", "idade":18},

{"nome":"Renato", "idade":38},

{"nome":"Claudia", "idade":44},

{"nome":"Monica", "idade":23},

{"nome":"Viviane", "idade":12},

{"nome":"Ester", "idade":24},

{"nome":"Gisele", "idade":32},

{"nome":"Luciane", "idade":56},

{"nome":"Sheyla", "idade":60},

])

# Exibição dos dados

**print**(pessoas\_df)

nome idade

0 Patricia 15

1 Leticia 30

2 Marcelo 19

3 Adriane 18

4 Renato 38

5 Claudia 44

6 Monica 23

7 Viviane 12

8 Ester 24

9 Gisele 32

10 Luciane 56

11 Sheyla 60

# Salva DataFrame como um arquivo Parquet

pessoas\_df**.**to\_parquet("./pessoas.pq")

# Carrega o arquivo Parquet

parquet\_df **=** pd**.**read\_parquet("./pessoas.pq")

Conferir os dados salvo

**print**(parquet\_df)

nome idade

0 Patricia 15

1 Leticia 30

2 Marcelo 19

3 Adriane 18

4 Renato 38

5 Claudia 44

6 Monica 23

7 Viviane 12

8 Ester 24

9 Gisele 32

10 Luciane 56

11 Sheyla 60

**Spark**

É uma ***engine*** para processamento e análise de dados. Vamos usar o Spark para salvar um DataFrame no formato Parquet, carregar o arquivo gerado e executar queries SQL em cima dos dados.

# Importação de bibliotecas

**from** pyspark.sql **import** SparkSession, Row

# Sessão com Spark

spark **=** SparkSession**.**builder**.**appName("demo-app")**.**getOrCreate()

# Criação DataFrame com dados

pessoas\_df **=** spark**.**createDataFrame([

Row(nome**=**"Patricia", idade**=**15),

Row(nome**=**"Leticia", idade**=**30),

Row(nome**=**"Marcelo", idade**=**19),

Row(nome**=**"Adriane", idade**=**18),

Row(nome**=**"Renato", idade**=**38),

Row(nome**=**"Claudia", idade**=**44),

Row(nome**=**"Monica", idade**=**23),

Row(nome**=**"Viviane", idade**=**12),

Row(nome**=**"Ester", idade**=**24),

Row(nome**=**"Gisele", idade**=**32),

Row(nome**=**"Luciane", idade**=**56),

Row(nome**=**"Sheyla", idade**=**60),

])

# Salvar DataFrame como Parquet

pessoas\_df**.**write**.**parquet("pessoas.parquet")

# Carregar o Parquet

parquet\_df **=** spark**.**read**.**parquet("pessoas.parquet")

# Cria uma View temporária para comandos SQL

parquet\_df**.**createOrReplaceTempView("pessoasView")

Listaremos todos os itens salvos no Parquet para conferir com o que criamos inicialmente.

In [7]:

todas\_pessoas **=** spark**.**sql("SELECT nome, idade FROM pessoasView")

todas\_pessoas**.**show()

+--------+-----+

| nome|idade|

+--------+-----+

| Ricardo| 23|

| Marcelo| 18|

| Marlei| 32|

|Marilene| 56|

| Judite| 60|

| Pedro| 15|

| Camila| 12|

| Alice| 24|

| Alex| 38|

| Otavio| 44|

| João| 30|

| Maria| 19|

+--------+-----+

Podemos efetuar outras pesquisas, como busca todas pessoas maiores de idade.

In [8]:

maiores\_de\_idade **=** spark**.**sql("SELECT nome, idade FROM pessoasView WHERE idade >= 18")

maiores\_de\_idade**.**show()

+--------+-----+

| nome|idade|

+--------+-----+

| Ricardo| 23|

| Marcelo| 18|

| Marlei| 32|

|Marilene| 56|

| Judite| 60|

| Alice| 24|

| Alex| 38|

| Otavio| 44|

| João| 30|

| Maria| 19|

+--------+-----+

Por fim, encerramos a sessão com o Spark.

In [9]:

spark**.**stop()

**Conclusão**

O artigo de hoje foi mais para explicar o que é Parquet e porque ele pode ser uma melhor alternativa de armazenamento em comparação a CSV e JSON.

Se você já faz uso de ferramentas como Spark ou outras da Apache, será inevitável uma migração no futuro para um ganho em desempenho.

A única desvantagem que eu vejo em comparação a outros formatos é que Parquet é um formato binário, sendo assim, é necessária alguma ferramenta externa para visualizar os dados, no caso de CSV ou JSON basta abrir em algum editor de texto, ou planilha, mas tecnicamente isso irá funcionar apenas para arquivos pequenos.

Dúvidas ou sugestões? Deixe um comentário.