# O uso das ferramentas Hadoop, Hive e Zeppelin para análise do dataset de veículos em circulação no RS

**Gabriel Neves**[1], **Henrique Schwab**[1], **Rodrigo Monteiro**[1], **Thiago Garbin**[1]

[1]Ciência de Dados – Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Porto Alegre – RS – Brazil

{gabrielneves1994,schwabhenrique,rsmonteiro82,tsgarbin}@gmail.com

***Abstract.*** *This paper demonstrates the process of using Big Data Hadoop, Hive and Zeppelin tools. The installation and configuration of the systems will be detailed and a dataset in CSV format will be ingested. Finally, the chosen dataset is explored with queries in the "SQL"format and a performance comparison between Hive storage formats is described.*

***Resumo.*** *O presente trabalho demonstra o processo de uso das ferramentas de Big Data Hadoop, Hive e Zeppelin. Será detalhado a instalação e configuração dos sistemas e realizado a ingestão de um dataset no formato CSV. Por fim, o dataset escolhido é explorado com consultas no formato "SQL" e é mostrado um comparativo de desempenho entre formatos de armazenamento do Hive.*

## 1. Introdução

Analisar um dataset com uma grande quantidade de dados é uma tarefa complexa e exige conhecimento de diversas ferramentas. O objetivo deste trabalho é demonstrar através da configuração e do uso de algumas soluções para Big Data, como o Hadoop, Hive e Zeppelin uma forma de ingestão e análise de dados com a utilização do conjunto de dados de veículos em circulação no estado do RS.

## 2. Dataset

O dataset escolhido foi o "Frota de Veículos em Circulação Dezembro 2020". Ele reúne dados de todos os veículos em circulação no Estado do Rio Grande do Sul no mês de dezembro de 2020.

Os dados estão disponibilizados em um arquivo CSV com tamanho de 1,77 GB, contendo 7.123.343 linhas e 27 colunas. No dataset é possível encontrar informações dos 7.123.342 veículos registrados no RS. Marca, ano de fabricação, categoria dos veículos, espécie, tipo, cor, idade, assim como informações relacionadas a restrições e licenciamento dos veículos estão entre os dados disponibilizados.

A fonte dos dados é o portal de dados abertos do governo do RS. Os dados de dezembro de 2020 disponibilizados no site do governo possuem última atualização em 08/12/2020 [RIO GRANDE DO SUL 2020].

## 3. Processo de configuração do Hadoop, Hive e Zeppelin

Para a instalação da plataforma Hadoop, foi utilizada uma máquina com sistema operacional Ubuntu 18.04.5 LTS que já continha o Java 8 instalado. Tal processo também pode ser feito em uma máquina virtual que contenha estes requisitos. A versão do Hadoop utilizada é a 3.2.1, que é a última versão estável. Através do terminal do Linux, inicialmente foi realizado o download da plataforma para posteriormente descompactar esta e configurar as variáveis de ambiente com o arquivo hadoop_vars.sh, configurar o arquivo hadoop_env.sh para que o Hadoop encontre o Java e configurar o diretório de logs, conforme os comandos abaixo:

```
wget -c https://downloads.apache.org/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz
tar -zxvf hadoop-3.2.1.tar.gz
mv hadoop-3.2.1 hadoop
```

```
cat > hadoop_vars.sh << EOL
export JAVA_HOME=$(dirname $(dirname $(readlink -f $(which javac))))
export HADOOP_HOME=$HOME/hadoop
export PATH=$PATH:$HOME/hadoop/bin:$HOME/hadoop/sbin
EOL

source hadoop_vars.sh
sed -i "\$aexport JAVA_HOME=$JAVA_HOME" $HADOOP_HOME/etc/hadoop/hadoopenv.sh
mkdir $HADOOP_HOME/logs
```

Na sequência, foi necessário alterar a configuração de arquivos que ficam na home do Hadoop. No arquivo *core-site.xml* foi informado onde está localizado o sistema de arquivos e onde devem ser gravados arquivos de sistema. No arquivo *hdfs-site.xml* foi indicado o número de réplicas para os blocos dos arquivos no sistema, que neste caso é apenas um, pois está sendo utilizada uma máquina. Por fim, no arquivo *yarn-site.xml* foi habilitado o serviço *mapreduce shuffle* e no arquivo mapred-site foi especificado que será utilizado o yarn como framework de escalonamento. Tais alterações foram realizadas com a inserção das seguintes propriedades:

core-site.xml:

```
<configuration>
<property>
<name>fs.defaultFS</name>
<value>hdfs://localhost:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/home/${user.name}/hadooptmp</value>
</property>
</configuration>
```

mapred-site.xml:

```
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
<property>
<name>mapreduce.application.classpath</name>
<value>$HADOOP_HOME/share/hadoop/mapreduce/*:
$HADOOP_HOME/share/hadoop/mapreduce/lib/*</value>
</property>
</configuration>
```

hdfs-site.xml:

```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
</configuration>
```

yarn-site.xml:

```
<configuration>
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
</configuration>
```

Em seguida, para finalizar a configuração do Hadoop, foi realizada a formatação do HDFS e a inicialização dos serviços do Hadoop em modo daemon para a criação dos diretórios e alteração de permissões, de acordo com os comandos a seguir:

```
echo 'Y' | hdfs namenode -format

yarn --daemon start resourcemanager
yarn --daemon start nodemanager
yarn --daemon start timelineserver
hdfs --daemon start namenode
hdfs --daemon start datanode

hdfs dfs -mkdir -p /user/$USER
hdfs dfs -chown $USER:$USER /user/$USER
hdfs dfs -mkdir /tmp
hdfs dfs -chmod 777 /tmp
```

A próxima etapa é a instalação do Apache Hive, que permitirá a manipulação de bancos de dados e tabelas com comandos SQL. A versão utilizada será a 3.1.2, que pode ser baixada e descompactada conforme os seguintes comandos:

```
wget -c https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
tar -zxvf apache-hive-3.1.2-bin.tar.gz
```

Para facilitar o processo de encontrar o programa, foi alterado o arquivo hadoop_vars.sh, inserindo a linha referente ao Hive e alterando a linha do Path:

```
export HIVE_HOME=$HOME/apache-hive-3.1.2-bin
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$HIVE_HOME/bin:$PATH
```

Prosseguindo no terminal do Linux e já tendo carregado novamente o arquivo *hadoop_vars.sh* com as variáveis de ambiente, foi necessário realizar uma correção nas versões das bibliotecas guava e slf4j para um melhor funcionamento do programa:

```
rm $HIVE_HOME/lib/guava-19.0.jar
cp $HADOOP_HOME/share/hadoop/hdfs/lib/guava-27.0-jre.jar $HIVE_HOME/lib
rm $HIVE_HOME/lib/log4j-slf4j-impl-2.10.0.jar
```

Para configurar o usuário que acessará o Hive, foi preciso novamente alterar o arquivo *core-site.xml* incluindo as properties a seguir, nas quais username refere-se ao nome de usuário do sistema Linux:

```
<property>
<name>hadoop.proxyuser.username.groups</name>
<value>*</value>
</property>
<property>
<name>hadoop.proxyuser.username.hosts</name>
<value>*</value>
</property>
```

Na sequência, após reiniciar os serviços do Hadoop em modo daemon, a instalação do Hive foi concluída com a criação dos diretórios do Hive Metastore no HDFS e a inicialização do Hive Metastore utilizando o banco de dados Derby local:

```
hdfs dfs -mkdir -p /user/hive/warehouse
hdfs dfs -chmod g+w /user/hive/warehouse
mkdir $HIVE_HOME/hiveserver2
cd $HIVE_HOME/hiveserver2
$HIVE_HOME/bin/schematool -dbType derby -initSchema
```

Por fim, para possibilitar a utilização do Hive com uma interface gráfica mais amigável e que possibilita maior interação e criação de gráficos, foi realizada a instalação do Apache Zeppelin na versão 0.8.2:

```
wget -c https://downloads.apache.org/zeppelin/zeppelin-0.8.2/zeppelin-0.8.2-bin-all.tgz
tar -zxvf zeppelin-0.8.2-bin-all.tgz
```

Mais uma vez, foi realizada alteração do arquivo hadoop_vars.sh que contém a configuração das variáveis de ambiente, agora para incluir a linha referente ao Zeppelin e alterar a linha referente ao Path, carregando o arquivo com source após o processo:

```
export ZEPPELIN_HOME=$HOME/zeppelin-0.8.2-bin-all
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$HIVE_HOME/bin:$ZEPPELIN_HOME/bin:$PATH
```

Em seguida, fez-se necessária a alteração do arquivo zeppelin-env.sh que se encontra na pasta conf que está dentro da home do Zeppelin. Neste arquivo, a linha referente ao caminho de configuração do Hadoop foi descomentada para ficar da seguinte maneira (username refere-se ao usuário do sistema Linux):

```
export HADOOP_CONF_DIR=/home/username/hadoop/etc/hadoop
```

Concluída esta etapa e com os daemons do Hadoop ativos, foi possível inicializar os serviços do Hive e do Zeppelin, bem como inserir no HDFS o arquivo csv referente a frota de veículos em circulação em dezembro de 2020 no Rio Grande do Sul:

```
nohup $HIVE_HOME/bin/hive --service hiveserver2 \
--hiveconf hive.security.authorization.createtable.owner.grants=ALL \
--hiveconf hive.root.logger=INFO,console &

$ZEPPELIN_HOME/bin/zeppelin-daemon.sh start

hdfs dfs -mkdir -p veiculosrs
hdfs dfs -put VEI_DADOS_ABERTOS_20201201_1.CSV veiculosrs
```

Após isto, a última etapa de configuração consistiu em acessar a interface do Zeppelin através do http://localhost:8080 para criar o interpretador do Hive com conexão via jdbc, uma vez que este não é nativo do sistema. Isto consistiu em ir até a opção interpreter que está dentro de anonymous no canto superior direito e depois clicar em "Create", preenchendo em seguida o campo Interpreter name com "hive" e Interpreter Group com "jdbc". Além disto, os campos abaixo também precisaram ser preenchidos, clicando-se em Save após o preenchimento:

Properties

| *Property* | *Value* |
|---|---|
| default.driver | org.apache.hive.jdbc.HiveDriver |
| default.url | jdbc:hive2//localhost:10000 |
| default.user | username |
| defaul.splitQueries | true |

Dependencies

| *Property* |
|---|
| /home/username/apache-hive-3.1.2-bin/jdbc/hive-jdbc-3.1.2-standalone.jar |

Com as configurações finalizadas, foi possível gerar um novo notebook selecionando o Hive como interpretador. Dentro deste notebook, após inserir comandos para configurar o Map Reduce como engine de execução dos jobs e o Yarn como o framework dos jobs, prosseguiu-se com a criação da tabela dezembro2020 com base no arquivo csv que foi carregado no HDFS:

```
CREATE DATABASE veiculosrs;
USE veiculosrs;

CREATE EXTERNAL TABLE dezembro2020 (marca STRING, fabricacao INT, categoria STRING, especie STRING,
tipo STRING,carroceria STRING, cor STRING, combustivel STRING, cilindradas INT, potencia INT,
situacao STRING, munic_emplacamento STRING, lotacao STRING, cmt STRING, pbt STRING,
capacidade_carga STRING, tipo_proprietario STRING,sexo_proprietario STRING, idade INT,
data_aquisicao STRING, tipo_primeiro_registro STRING, dt_primeiro_reg STRING,
dt_ultimo_doc_licenciado STRING, situacao_licenciamento STRING,situacao_infracoes STRING,
restricoes STRING, tipo_restricao STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ';'
STORED AS TEXTFILE
LOCATION 'hdfs:///user/username/veiculosrs'
TBLPROPERTIES ("skip.header.line.count"="1");
```

## 4. Análise dos Resultados

Para validarmos todo o processo que foi criado anteriormente, realizamos algumas consultas ao conjunto de dados, respondendo às perguntas pré-definidas. Como critério de seleção de automóveis, utilizamos os tipos 'AUTOMOVEL', 'UTILITARIO', 'CAMINHOTE', 'CAMIONETA'.

### 4.1. Quais são os dez automóveis mais registrados no RS?

Para obtermos os 10 automóveis com mais registros no estado do RS, selecionamos todas as marcas de carros e a quantidade de ocorrência de cada uma. No processo de seleção de cada uma das marcas, utilizamos uma expressão regular que seleciona todo o conteúdo até o primeiro espaço em branco encontrado no registro de cada marca, ignorando o conteúdo após o espaço em branco encontrado.

```
SELECT REGEXP_EXTRACT(MARCA,'^\\S*',0) CARRO, COUNT(MARCA) QTD
FROM DEZEMBRO2020
WHERE TIPO IN('AUTOMOVEL','UTILITARIO','CAMINHONETE','CAMIONETA')
GROUP BY REGEXP_EXTRACT(MARCA,'^\\S*',0)
ORDER BY COUNT(MARCA) DESC
LIMIT 10;
```

### 4.2. Quais as 15 cores mais comuns em Porto Alegre em automóveis?

Para descobrirmos quais são as 15 cores de automóveis mais comuns em Porto Alegre, foi realizada uma consulta que seleciona todas as cores de automóveis e a quantidade de ocorrências de cada uma das 15 principais.

```
SELECT COR, COUNT(COR) QTD
FROM DEZEMBRO2020
WHERE TIPO IN('AUTOMOVEL','UTILITARIO','CAMINHONETE','CAMIONETA')
AND MUNIC_EMPLACAMENTO = 'PORTO ALEGRE'
GROUP BY COR
ORDER BY COUNT(COR) DESC
LIMIT 15;
```

### 4.3. Quais os tipos de combustíveis mais utilizados no RS?

Para visualizarmos quais são os tipos de combustíveis mais utilizados no estado do Rio do Grande do Sul, realizamos uma consulta que seleciona todos os tipos de combustíveis, exceto os registrados como 'SEM COMBUSTIVEL' e 'VIDE CAMPO OBSERVAÇÃO', e a quantidade de ocorrências de cada um deles.

```
SELECT COMBUSTIVEL, COUNT(COMBUSTIVEL) QTD
FROM DEZEMBRO2020
WHERE COMBUSTIVEL NOT IN('SEM COMBUSTIVEL','VIDE CAMPO OBSERVACAO')
GROUP BY COMBUSTIVEL
ORDER BY COUNT(COMBUSTIVEL) DESC;
```

### 4.4. Quais são as marcas dos automóveis elétricos que rodam no RS?

Para obtermos os modelos/marcas dos automóveis elétricos que rodam no estado do Rio Grande do Sul, fizemos uma consulta selecionando todas as marcas de automóveis e suas ocorrências e com os tipos de combustíveis 'ELÉTRICO/FONTE INTERNA' e 'ELÉTRICO/FONTE EXTERNA'.

```
SELECT MARCA, COUNT(MARCA) QTD
FROM DEZEMBRO2020
WHERE COMBUSTIVEL IN ('ELETRICO/FONTE INTERNA','ELETRICO/FONTE EXTERNA')
AND TIPO IN('AUTOMOVEL','UTILITARIO','CAMINHONETE','CAMIONETA')
GROUP BY MARCA
ORDER BY COUNT(MARCA) DESC
```

### 4.5. Qual a média de idade, desvio padrão e o menor ano de fabricação da frota de automóveis em Porto Alegre, Caxias do Sul, Alvorada, Dom Pedrito, Santa Vitória do Palmar e Mostardas?

Para calcularmos a média de idade da frota de automóveis nas cidades de Porto Alegre, Caxias do Sul, Alvorada, Dom Pedrito, Santa Vitória do Palmar e Mostardas, fizemos uma consulta que seleciona o município de emplacamento dos automóveis e a média de idade dos automóveis. Adicionalmente, obtivemos o menor ano de fabricação de um automóvel naquela cidade e o desvio-padrão das idades dos automóveis.

```
SELECT MUNIC_EMPLACAMENTO, AVG(2020 - FABRICACAO) AS MEDIA_IDADE_AUTOMOVEL
     , STDDEV_POP(2020 - FABRICACAO) AS DESVP_IDADE
     , MIN(FABRICACAO) AS  MENOR_ANO_FABRICACAO
FROM DEZEMBRO2020
WHERE TIPO IN('AUTOMOVEL','UTILITARIO','CAMINHONETE','CAMIONETA')
AND MUNIC_EMPLACAMENTO IN ('PORTO ALEGRE','ALVORADA','CAXIAS DO SUL','DOM PEDRITO'
                          ,'SANTA VITORIA DO PALMAR','MOSTARDAS')
GROUP BY MUNIC_EMPLACAMENTO
ORDER BY MEDIA_IDADE_AUTOMOVEL
```

## 5. Análise de desempenho

As consultas demonstradas na seção anterior foram executadas em uma tabela externa ao Hive, ou seja o conteúdo não está sob gerenciamento da plataforma. Os tempos de execução via Zeppelin foram 59, 54, 57, 55 e 58 segundos. Quando executadas as mesmas consultas para uma tabela gerenciada, armazenada no formato ORC, os tempos foram reduzidos para 43, 39, 37, 40 e 42 segundos. Neste caso há uma melhora de mais de 25% no desempenho em função do uso do formato otimizado de acesso aos dados.

## 6. Conclusão

Concluímos que os sistemas apresentados neste trabalho oferecem um ótimo suporte para as tarefas de ingestão e análise de datasets. Após a configuração do ambiente, o uso do Hive em conjunto com o Zeppelin assemelha-se as ferramentas tradicionais de banco de dados, o que facilita a extração e análise dos dados. Observamos ainda que novas análises de desempenho podem ser realizadas com outras configurações das ferramentas. Existem alternativas como, por exemplo, testar o número de *mappers* usados no Hive ou ainda com a utilização de outros *engines*, como o Tez e o Spark.

## Referências

RIO GRANDE DO SUL (2020). Frota de veiculos em circulação dezembro 2020. https://dados.rs.gov.br/dataset/frota-veiculos-em-circulacao/resource/85ea91f9-943b-43e0-bc1c-6570b4fcd565.