

Bootcamp Analista de Dados

Trabalho Prático

Módulo 2	Processamento de dados utilizando o Ecossistema
-----------------	--

Objetivos

Exercitar os seguintes conceitos trabalhados no Módulo:

- ✓ Manipular o HDFS (Hadoop Distributed FileSystem, formatando o sistema de arquivos e executando operações com arquivos e diretórios).
- ✓ Acessar as ferramentas de Gerenciamento do Ecossistema Hadoop.
- ✓ Alterar o código fonte de um programa, compilá-lo e executá-lo.

Enunciado

Para esta atividade, o aluno deverá assistir atentamente as seguintes aulas, disponíveis no ambiente virtual de aprendizagem, **exatamente na ordem em que se apresenta abaixo**:

1. Instalando a máquina virtual;
2. Executando os comandos básicos do Ecossistema Hadoop e do HDFS;
3. Criando, compilando e executando um Programa com o Hadoop/MapReduce;

Atividades

Os alunos deverão desempenhar as seguintes atividades:

1 – Instalando a Máquina Virtual

Deve-se assistir ao vídeo “Instalando a Máquina Virtual” e seguir todos os passos existentes nele. Ao final, deve-se verificar se a máquina virtual está inicializando normalmente e se o login está sendo realizado.

Após executar o login (*login: igt senha: igt*) na máquina virtual, inicie o Terminal do Linux, conforme demonstrado na Figura 1.

Figura 1 – Tela Inicial da Máquina Virtual.



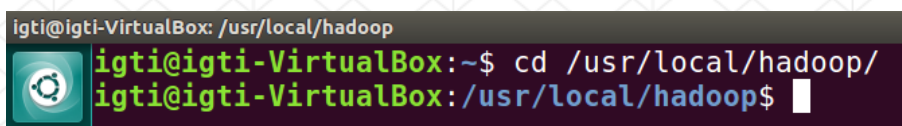
2 – Preparando o HDFS e iniciando os serviços do Hadoop

A ferramenta Hadoop já foi previamente instalada no ambiente virtual e encontra-se no diretório `/usr/local/hadoop`. Sendo assim, devemos ir até o diretório de instalação do Hadoop utilizando o seguinte comando:

```
cd /usr/local/hadoop
```

Após a digitação do comando acima, teremos a tela apresentada na Figura 2.

Figura 2 – Diretório de instalação do Hadoop.



Antes de tudo é necessário realizar a formatação do sistema de arquivos distribuídos do Hadoop (HDFS). Esse processo envolve dois passos, sendo: (i) eliminar os diretórios temporários; (ii) realizar a formatação do HDFS.

- (i) Delete todos os subdiretórios que estão dentro da pasta `tmp` localizada dentro do diretório padrão do Hadoop (`/usr/local/hadoop`). Utilize o comando abaixo:

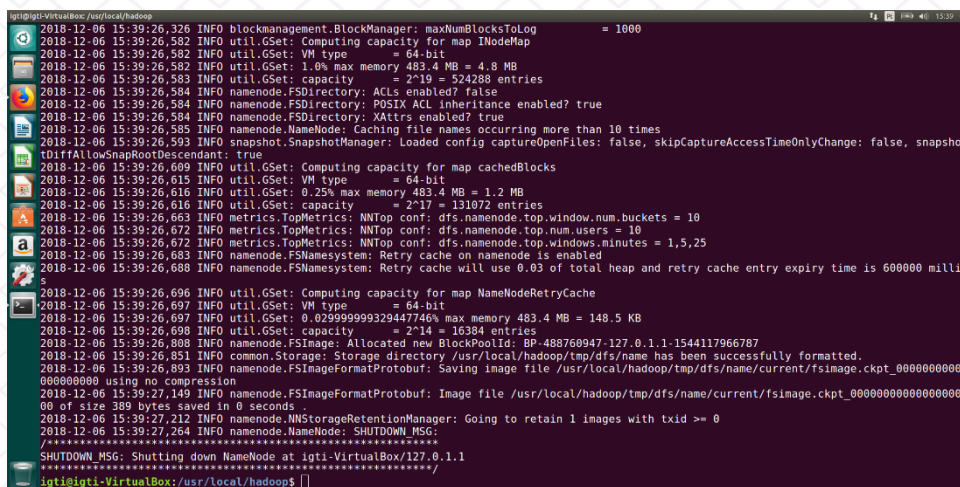
```
rm -r /usr/local/hadoop/tmp/*
```

- (ii) Para formatar o HDFS, execute o comando abaixo:

```
/usr/local/hadoop/bin/hdfs namenode -format
```

Após alguns segundos, a tela apresentada na Figura 3 será exibida:

Figura 3 – Tela de Formatação do Hadoop (resultado).



```
2018-12-06 15:39:26.326 INFO blockmanagement.BlockManager: maxNumBlocksToLog = 1000
2018-12-06 15:39:26.582 INFO util.GSet: Computing capacity for map INodeMap
2018-12-06 15:39:26.582 INFO util.GSet: VM type = 64-bit
2018-12-06 15:39:26.582 INFO util.GSet: 1.0% max memory 483.4 MB = 4.8 MB
2018-12-06 15:39:26.583 INFO util.GSet: capacity = 2^19 = 524288 entries
2018-12-06 15:39:26.584 INFO namenode.FSDirectory: ACLs enabled? false
2018-12-06 15:39:26.584 INFO namenode.FSDirectory: POSIX ACL inheritance enabled? true
2018-12-06 15:39:26.584 INFO namenode.FSDirectory: XAttrs enabled? true
2018-12-06 15:39:26.585 INFO namenode.NameNode: Caching file names occurring more than 10 times
2018-12-06 15:39:26.593 INFO snapshot.SnapshotManager: Loaded config captureOpenFiles: false, skipCaptureAccessTimeOnlyChange: false, snapshotDiffAllowSnapRootDescendant: true
2018-12-06 15:39:26.609 INFO util.GSet: Computing capacity for map cachedBlocks
2018-12-06 15:39:26.615 INFO util.GSet: VM type = 64-bit
2018-12-06 15:39:26.616 INFO util.GSet: 0.25% max memory 483.4 MB = 1.2 MB
2018-12-06 15:39:26.616 INFO util.GSet: capacity = 2^17 = 131072 entries
2018-12-06 15:39:26.663 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
2018-12-06 15:39:26.672 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
2018-12-06 15:39:26.672 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
2018-12-06 15:39:26.683 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
2018-12-06 15:39:26.688 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 milli
s
2018-12-06 15:39:26.696 INFO util.GSet: Computing capacity for map NameNodeRetryCache
2018-12-06 15:39:26.697 INFO util.GSet: VM type = 64-bit
2018-12-06 15:39:26.697 INFO util.GSet: 0.0299999999329447746% max memory 483.4 MB = 148.5 KB
2018-12-06 15:39:26.698 INFO util.GSet: capacity = 2^14 = 16384 entries
2018-12-06 15:39:26.808 INFO namenode.FSImage: Allocated new BlockPoolId: BP-488760947-127.0.1.1-1544117966787
2018-12-06 15:39:26.851 INFO common.Storage: Storage directory /usr/local/hadoop/tmp/dfs/name has been successfully formatted.
2018-12-06 15:39:26.893 INFO namenode.FSImageFormatProtobuf: Saving image file /usr/local/hadoop/tmp/dfs/name/current/fsimage.ckpt_0000000000
00000000 using no compression
2018-12-06 15:39:27.149 INFO namenode.FSImageFormatProtobuf: Image file /usr/local/hadoop/tmp/dfs/name/current/fsimage.ckpt_0000000000000000
00 of size 389 bytes saved in 0 seconds.
2018-12-06 15:39:27.212 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
2018-12-06 15:39:27.264 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
Shutting down NameNode at igti-VirtualBox/127.0.1.1
*****/
igti@igti-VirtualBox: /usr/local/hadoop$
```

Neste momento você apagou todos os arquivos e diretórios do sistema de arquivos distribuídos do Hadoop (HDFS). Esses comandos específicos do HDFS serão melhor trabalhados nas próximas atividades práticas e também durante nossas aulas. Por enquanto, iremos utilizar apenas a formatação. Agora já estamos prontos para iniciar os serviços do Hadoop.

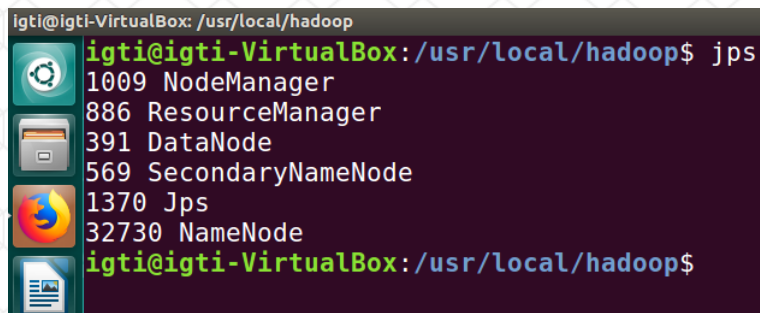
Estamos dentro do diretório de instalação do Hadoop, e a próxima tarefa será iniciar os serviços da ferramenta. Iremos utilizar um arquivo que encontra-se dentro do diretório `/usr/local/hadoop/sbin` chamado `start-all.sh`. Para isso, execute o comando abaixo:

```
/usr/local/hadoop/sbin/start-all.sh
```

Observe que ao utilizar o comando **start-all**, o Hadoop emite uma mensagem informando que esse comando foi descontinuado. Apesar disso, o comando funciona perfeitamente e, caso queira, você pode utilizar os comandos **start-yarn.sh** e **start-dfs.sh**. O efeito será exatamente o mesmo.

Neste momento é esperado que todos os serviços do Hadoop tenham sido iniciados. Para conferir se todos os serviços foram devidamente iniciados (DataNode, ResourceManager, NameNode, SecondaryNameNode e NodeManager), digite o comando *jps* no Terminal. Esse comando lista os serviços Java que estão sendo executados na máquina. Após a execução do comando *jps*, se tudo estiver correto, a tela da Figura 4 será apresentada. Observe que todos os cinco serviços necessários para a execução do Hadoop encontram-se listados:

Figura 4 – Tela com os cinco serviços do Hadoop inicializados.



```
igti@igti-VirtualBox: /usr/local/hadoop$ jps
1009 NodeManager
886 ResourceManager
391 DataNode
569 SecondaryNameNode
1370 Jps
32730 NameNode
igti@igti-VirtualBox: /usr/local/hadoop$
```

Importante: se ao listar os processos com o comando *jps*, não aparecerem como ativos os processos DataNode, ResourceManager, NameNode, SecondaryNameNode e NodeManager, você deverá seguir os seguintes passos:

- 1) Pare o serviço do Hadoop: `/usr/local/hadoop/sbin/stop-all.sh`
- 2) Delete novamente os arquivos temporários: `rm -r /usr/local/hadoop/tmp/*`
- 3) Veja se os arquivos e diretórios temporários foram realmente excluídos, usando o comando:

```
ls /usr/local/hadoop/tmp
```

- 4) Formate novamente o HDFS: `/usr/local/hadoop/bin/hdfs namenode -`

format.

- 5) Reinicie os serviços do Hadoop: `/usr/local/hadoop/sbin/start-all.sh`
- 6) Consulte novamente os serviços do Hadoop com o comando `jps`.

Obs.: você não deverá avançar no tutorial caso os cinco processos do Hadoop não estejam em execução.

3 – Compilando o programa no Hadoop/MapReduce

Nesse momento já estamos com os serviços do Hadoop executando plenamente e já podemos escrever e compilar o nosso primeiro programa. Inicialmente, dentro do diretório `/usr/local/hadoop` foi criado um diretório chamado `ExemploIGTI`. Você deverá navegar nesse diretório e verificar que o mesmo possui um arquivo e um outro diretório.

O arquivo `build_ExemploIGTI.xml` armazena os dados de compilação do nosso programa, e o arquivo `ExemploIGTI.java`, que se encontra dentro do diretório `src`, possui o código fonte em Java referente ao nosso programa de exemplo.

Vá para o diretório `ExemploIGTI` utilizando o seguinte comando:

```
cd /usr/local/hadoop/ExemploIGTI/
```

Liste o conteúdo do diretório `ExemploIGTI`:

```
ls
```

Em seguida, vá para o diretório `src` utilizando o seguinte comando:

```
cd /usr/local/hadoop/ExemploIGTI/src
```

Liste o conteúdo do diretório `src`:

```
ls
```

Para compilar o nosso primeiro programa Hadoop iremos utilizar o Apache Ant, que já foi previamente instalado na máquina. O comando para compilação é o seguinte:

```
ant -f /usr/local/hadoop/ExemploIGTI/build_ExemploIGTI.xml makejar
```

Após compilar o programa, a mensagem da Figura 5 irá aparecer na tela:

Figura 5 – Tela do resultado da compilação.

```
Terminal
igti@igti-VirtualBox: /usr/local/hadoop
igti@igti-VirtualBox: /usr/local/hadoop$ ant -f /usr/local/hadoop/ExemploIGTI/build_ExemploIGTI.xml makejar
Buildfile: /usr/local/hadoop/ExemploIGTI/build_ExemploIGTI.xml

compile:
[javac] /usr/local/hadoop/ExemploIGTI/build_ExemploIGTI.xml:25: warning: 'in
cludeantruntime' was not set, defaulting to build.sysclasspath=last; set to fals
e for repeatable builds
[javac] Compiling 1 source file to /usr/local/hadoop/ExemploIGTI/classes

makejar:
[jar] Building jar: /usr/local/hadoop/ExemploIGTI/ExemploIGTI.jar

BUILD SUCCESSFUL
Total time: 1 second
igti@igti-VirtualBox: /usr/local/hadoop$
```

Em seguida, vamos verificar como ficou o conteúdo do nosso diretório ExemploIGTI, dando um comando ls. A Figura 6 apresenta o resultado.

```
cd /usr/local/hadoop/ExemploIGTI

ls
```

Figura 6 – Novo conteúdo do diretório ExemploIGTI.

```
Terminal
igti@igti-VirtualBox: /usr/local/hadoop/ExemploIGTI
igti@igti-VirtualBox: /usr/local/hadoop/ExemploIGTI$ ls
build_ExemploIGTI.xml  classes  ExemploIGTI.jar  src
igti@igti-VirtualBox: /usr/local/hadoop/ExemploIGTI$
```

Observe que foi criado o arquivo ExemploIGTI.jar. Esse é o arquivo compilado que será enviado para o Hadoop durante a execução.

4 - Alterando o programa no Hadoop/MapReduce

Nesse momento devemos realizar as alterações no código fonte do programa para que possamos executá-lo.

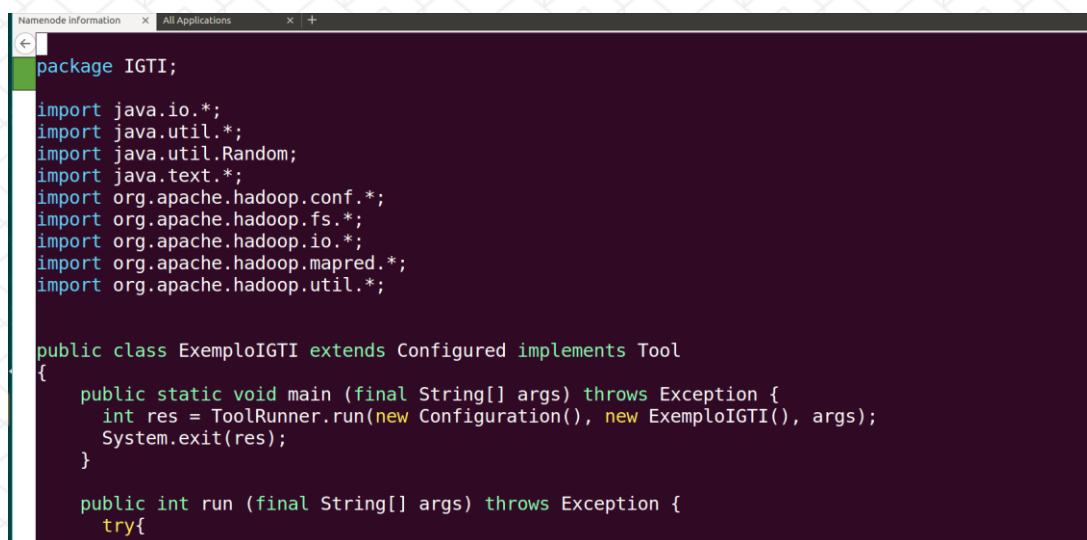
Primeiramente, observe que já temos algum código escrito no arquivo ExemploIGTI.java. Para verificar o conteúdo desse arquivo, dirija-se até a pasta src e abra o arquivo utilizando o aplicativo Vim.

```
cd /usr/local/hadoop/ExemploIGTI/src
```

```
vim ExemploIGTI.java
```

Ao abrir o arquivo com o Vim, a seguinte tela da Figura 7 deverá ser apresentada. Observe que já existe uma parte do código fonte implementada. O que temos ali é somente a estrutura do programa Hadoop, que explicamos em nossas aulas gravadas. Temos a classe `ExemploIGTI`, que possui o método `main`, a classe `MapIGTI` que implementa o nosso método `Map` e a classe `ReduceIGTI`, que implementa o nosso método `Reduce`. Observe que o método `run` da classe `ExemploIGTI` se encontra vazio, somente a estrutura foi criada.

Figura 7 – Estrutura do programa Hadoop/MapReduce.



```
package IGTI;

import java.io.*;
import java.util.*;
import java.util.Random;
import java.text.*;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;

public class ExemploIGTI extends Configured implements Tool
{
    public static void main (final String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new ExemploIGTI(), args);
        System.exit(res);
    }

    public int run (final String[] args) throws Exception {
        try{
            // 1 0 1
        }
    }
}
```

Você deverá assistir a aula “Criando, compilando e executando um Programa com o Hadoop/MapReduce” e realizar as seguintes implementações aqui no nosso código fonte:

1. Criar o objeto `JobConf`;
2. Criar os objetos para manipular os diretórios de Entrada e Saída do HDFS, com os objetos `Path` e `FileSystem` e o método `mkdirs`;
3. Copiar um arquivo do sistema de arquivos do Linux para o HDFS, utilizando o método `copyFromLocalFile`;
4. Configurar o diretório de Entrada e de Saída do seu job MapReduce, utilizando `FileInputFormat` e `FileOutputFormat`;
5. Atribuir os tipos de Key e Value, por meio dos métodos `setOutputKeyClass` e `setOutputValueClass`;

6. Atribuir as classes Mapper e Reducer, por meio dos métodos `setMapperClass` e `setReducerClass`;
7. Implementar a chamada ao método `RunJob`;
8. Implementar o conteúdo do método `Map` da classe `MapIGTI` e do método `Reduce` da classe `ReduceIGTI`.

Atenção: todos esses passos são detalhadamente explicados na aula gravada “Criando, compilando e executando um Programa com o Hadoop/MapReduce”.

Após a conclusão da sua implementação, feche o Vim (ESC + :wq) e compile novamente o seu programa, utilizando o comando abaixo:

```
ant -f /usr/local/hadoop/ExemploIGTI/build_ExemploIGTI.xml makejar
```

5 – Executando o programa no Hadoop/MapReduce

Após compilar nosso programa e gerar o nosso jar, iremos submetê-lo para a execução em um *job* Hadoop/MapReduce. A linha de comando que iremos utilizar está destacada abaixo:

```
/usr/local/hadoop/bin/hadoop                                     jar  
/usr/local/hadoop/ExemploIGTI/ExemploIGTI.jar IGTI.ExemploIGTI
```

O primeiro parâmetro que utilizamos é o `/usr/local/hadoop/bin/hadoop`. O arquivo Hadoop que fica dentro da pasta `bin` é o responsável por enviar os programas para a execução do *framework*. Em seguida é passada a palavra `jar`, indicando que iremos enviar um arquivo compilado do tipo jar para execução. O terceiro parâmetro é a localização desse jar no sistema de arquivos do Linux. Lembre-se que compilamos esse arquivo na pasta `ExemploIGTI`. Por último, estamos informando a classe que possui o método *main* (`IGTI.ExemploIGTI`).

A Figura 8 apresenta o comando sendo enviado e o job sendo executado.

Figura 7 – Logs de execução de um *job* Hadoop/MapReduce.


```

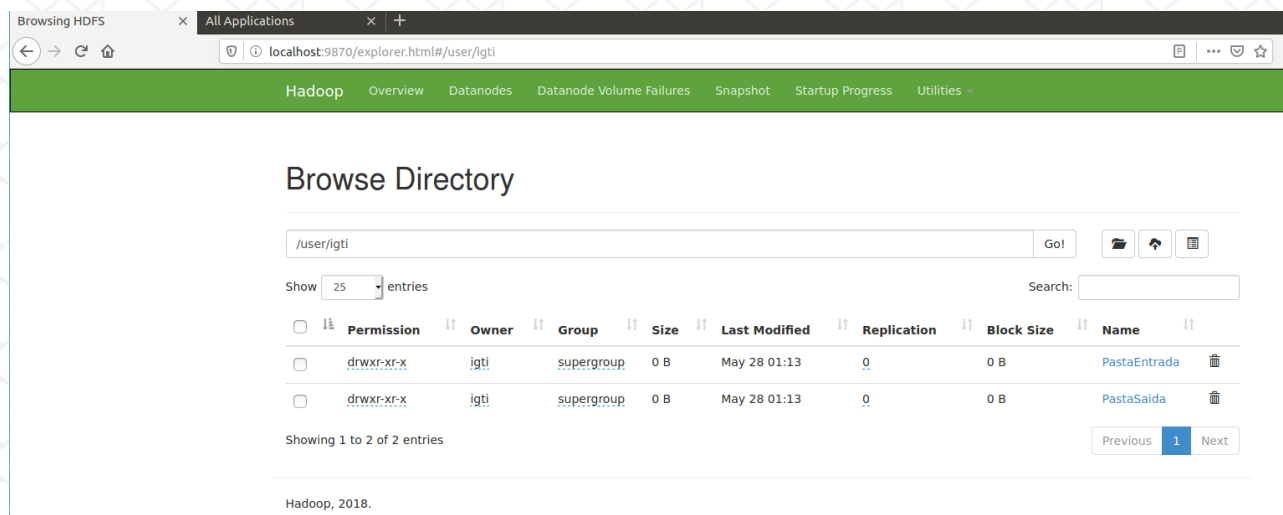
Terminal
igti@igti-VirtualBox: /usr/local/hadoop$ /usr/local/hadoop/bin/hadoop jar /usr/local/hadoop/ExemploIGTI/ExemploIGTI.jar IGTI.ExemploIGTI
2019-01-02 14:06:11,186 INFO client.RMPProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-01-02 14:06:11,794 INFO client.RMPProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-01-02 14:06:12,580 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/igti/.staging/job_1546445115443_0001
2019-01-02 14:06:12,860 INFO mapred.FileInputFormat: Total input files to process : 1
2019-01-02 14:06:13,473 INFO mapreduce.JobSubmitter: number of splits:2
2019-01-02 14:06:13,660 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
2019-01-02 14:06:14,412 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1546445115443_0001
2019-01-02 14:06:14,418 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-01-02 14:06:14,946 INFO conf.Configuration: resource-types.xml not found
2019-01-02 14:06:14,947 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-01-02 14:06:15,866 INFO impl.YarnClientImpl: Submitted application application_1546445115443_0001
2019-01-02 14:06:16,030 INFO mapreduce.Job: The url to track the job: http://igti-VirtualBox:8088/proxy/application_1546445115443_0001/
2019-01-02 14:06:16,035 INFO mapreduce.Job: Running job: job_1546445115443_0001
2019-01-02 14:06:36,841 INFO mapreduce.Job: Job job_1546445115443_0001 running in uber mode : false
2019-01-02 14:06:36,928 INFO mapreduce.Job: map 0% reduce 0%
2019-01-02 14:06:58,974 INFO mapreduce.Job: map 100% reduce 0%
2019-01-02 14:07:15,780 INFO mapreduce.Job: map 100% reduce 100%
2019-01-02 14:07:18,824 INFO mapreduce.Job: Job job_1546445115443_0001 completed successfully
2019-01-02 14:07:19,026 INFO mapreduce.Job: Counters: 53
File System Counters
FILE: Number of bytes read=306
FILE: Number of bytes written=605744
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=3306
HDFS: Number of bytes written=71

```

6 – Consultando os resultados

Para consultar os resultados de sua execução, acesse os diretórios do HDFS por meio do browser da sua máquina virtual, no endereço <http://localhost:9870>. Você deverá ir até o menu Utilities → Browse the file System. Clique no diretório user e depois em IGTI. O resultado do seu trabalho se encontra no diretório PastaSaida. A Figura 8 apresenta o resultado dessa operação:

Figura 8 – Conteúdo do HDFS após a execução.



Dentro do diretório PastaSaida há o arquivo part-00000. Esse é o resultado da execução da sua aplicação. Esse arquivo se encontra no HDFS, você deverá salva-lo no sistema de arquivos do Linux para conseguir visualizar o seu conteúdo. Clique sobre o arquivo, depois em download e salve-o. Ele será salvo em sua pasta Downloads.

Para acessá-lo, digite no terminal:

```
sudo vim /home/igti/Downloads/part-00000
```