

## Tutorial 5 – Primeiros passos com o Apache Kafka

Neste tutorial veremos como instalar o Apache Kafka na máquina Cloudera e faremos um teste de execução. A versão da CDH 5.13 não possui o Apache Kafka instalado, por isso precisamos começar com a instalação.

Esse tutorial está dividido em duas partes:

1. Instalação do Kafka
2. Criando um tópico

Vale lembrar que o Kafka é um sistema de mensagens baseado na lógica publish-subscribe. Isto é, temos um publicador e um consumidor dos dados. Um sistema de mensagens permite que você envie dados entre aplicações.

Na lógica do Kafka toda interação entre produtores (producer) e consumidores (consumer) é realizada através de tópicos. O produtor cria a mensagem e envia ao tópico especificado. Todos os consumidores vinculados aquele tópico recebem a mensagem publicada.

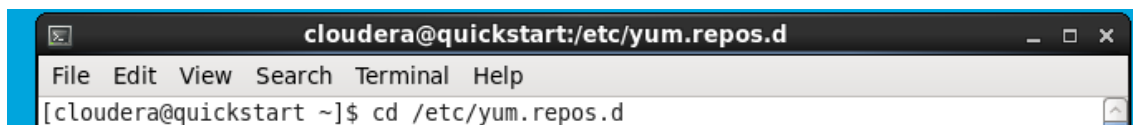
### Parte 1 – Instalação do Kafka na CDH

Primeiro vamos instalar o Kafka por meio do Yum.

O Yum é uma ferramenta para gerenciamento de pacotes no Linux que permite instalar, atualizar e remover programas.

1. No terminal digite:

```
cd /etc/yum.repos.d
```



2. Vamos fazer o download do Kafka. Digite no terminal

```
sudo wget http://archive.cloudera.com/kafka/redhat/6/x86\_64/kafka/
```

```
[cloudera@quickstart yum.repos.d]$ sudo wget http://archive.cloudera.com/kafka/redhat/6/x86_64/kafka/
--2019-12-03 02:57:55-- http://archive.cloudera.com/kafka/redhat/6/x86_64/kafka/
Resolving archive.cloudera.com... 151.101.92.167
Connecting to archive.cloudera.com|151.101.92.167|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5385 (5.3K) [text/html]
Saving to: "index.html"

100%[=====] 5,385 --.-K/s in 0s

2019-12-03 02:57:55 (110 MB/s) - "index.html" saved [5385/5385]
```

3. Antes de prosseguir com a instalação vamos limpar a cache do yum, digite:

`sudo yum clean all`

```
[cloudera@quickstart yum.repos.d]$ sudo yum clean all
Loaded plugins: fastestmirror, security
Cleaning repos: base cloudera-cdh5 cloudera-gplextras5 cloudera-kafka
               : cloudera-manager epel extras updates
Cleaning up Everything
```

4. Agora vamos proceder com a instalação. Digite no terminal:

`sudo yum install kafka`

```
[cloudera@quickstart yum.repos.d]$ sudo yum install kafka
Loaded plugins: fastestmirror, security
Setting up Install Process
Determining fastest mirrors
epel/metalink | 47 kB 00:00
* base: ftp.unicamp.br
* epel: mirror.compevo.com
* extras: centos.xpg.com.br
* updates: espejito.fder.edu.uy
base | 3.7 kB 00:00
base/primary_db | 4.7 MB 00:00
cloudera-cdh5 | 951 B 00:00
cloudera-cdh5/primary | 43 kB 00:00
cloudera-cdh5 153/153
cloudera-gplextras5 | 951 B 00:00
cloudera-gplextras5/primary | 2.5 kB 00:00
```

5. Instale o Kafka-server, para isso digite

`sudo yum install kafka-server`

```
[cloudera@quickstart yum.repos.d]$ sudo yum install kafka-server
Loaded plugins: fastestmirror, security
Setting up Install Process
Loading mirror speeds from cached hostfile
* base: ftp.unicamp.br
* epel: mirror.compevo.com
* extras: centos.xpg.com.br
* updates: espejito.fder.edu.uy
Resolving Dependencies
--> Running transaction check
---> Package kafka-server.noarch 0:0.10.2+kafka2.2.0-1.2.2.0.p0.92.el6 will be installed
```

6. Agora podemos iniciar o Kafka com o comando:

```
sudo service kafka-server start
```

```
[cloudera@quickstart yum.repos.d]$ sudo service kafka-server start
Starting Kafka Server (kafka-server): [ OK ]
Starting (kafka-server):
Starting Kafka Server (kafka-server): [ OK ]
```

7. Executando o jps você verá que o Kafka está rodando na JVM.

```
sudo jps
```

```
[cloudera@quickstart yum.repos.d]$ sudo jps
```

Além disso, podemos ver que o zookeeper, necessário para execução do Kafka, também está executando.

```
zookeeper → [cloudera@quickstart yum.repos.d]$ sudo jps
5160 QuorumPeerMain
8199 Bootstrap
6363 RESTServer
6540 ThriftServer
8233
5288 JournalNode
5667 Bootstrap
5219 DataNode
5975 ResourceManager
7131 Bootstrap
14915 Jps
5486 SecondaryNameNode
7165 HistoryServer
8273
6729 RunJar
5788 NodeManager
5714 JobHistoryServer
6647 RunJar
Kafka → 14284 Kafka
```

## Parte 2 – Criando nosso primeiro tópico.

Agora vamos ver o Kafka na prática. Nossa sequência será:

- Criar um tópico
- Inicializar o produtor (producer)
- Inicializar o consumidor (consumer)
- Enviar mensagens

Teremos ao final dois terminais, em um escreveremos uma mensagem e no outro visualizaremos a mensagem. Por padrão o Apache Kafka roda na porta 9092 e o Apache Zookeeper roda na porta 2181.

1. Acesse a pasta do Kafka, digite no terminal:

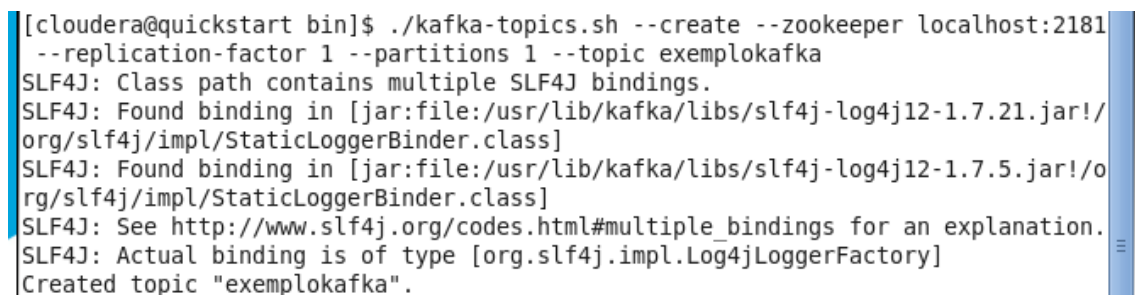
```
cd /usr/lib/kafka/bin/
```



```
[cloudera@quickstart ~]$ cd /usr/lib/kafka/bin/
```

2. Agora vamos criar um tópico chamado **exemplokafka**

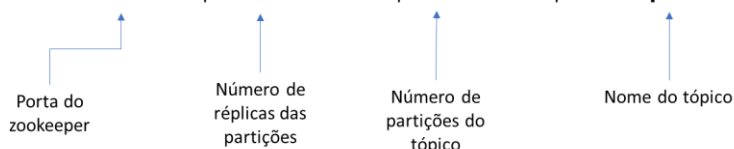
```
./kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1  
--partitions 1 --topic exemplokafka
```



```
[cloudera@quickstart bin]$ ./kafka-topics.sh --create --zookeeper localhost:2181  
--replication-factor 1 --partitions 1 --topic exemplokafka  
SLF4J: Class path contains multiple SLF4J bindings.  
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.21.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.  
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]  
Created topic "exemplokafka".
```

Entendendo nosso comando:

```
./kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic exemplokafka
```



3. Você pode listar todos os tópicos criados. Neste caso veremos o **exemplokafka**

`./kafka-topics.sh --list --zookeeper localhost:2181`

```
[cloudera@quickstart bin]$ ./kafka-topics.sh --list --zookeeper localhost:2181
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.21.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
exemplokafka
```

4. Agora vamos inicializar o producer para o tópico exemplokafka para que as mensagens sejam enviadas

`./kafka-console-producer.sh --broker-list localhost:9092 --topic exemplokafka`

```
[cloudera@quickstart bin]$ ./kafka-console-producer.sh --broker-list localhost:9092 --topic exemplokafka
```

5. Abra um novo terminal para iniciar o consumer e associando-o ao tópico exemplokafka. Acesse a pasta do Kafka

`cd /usr/lib/kafka/bin/`

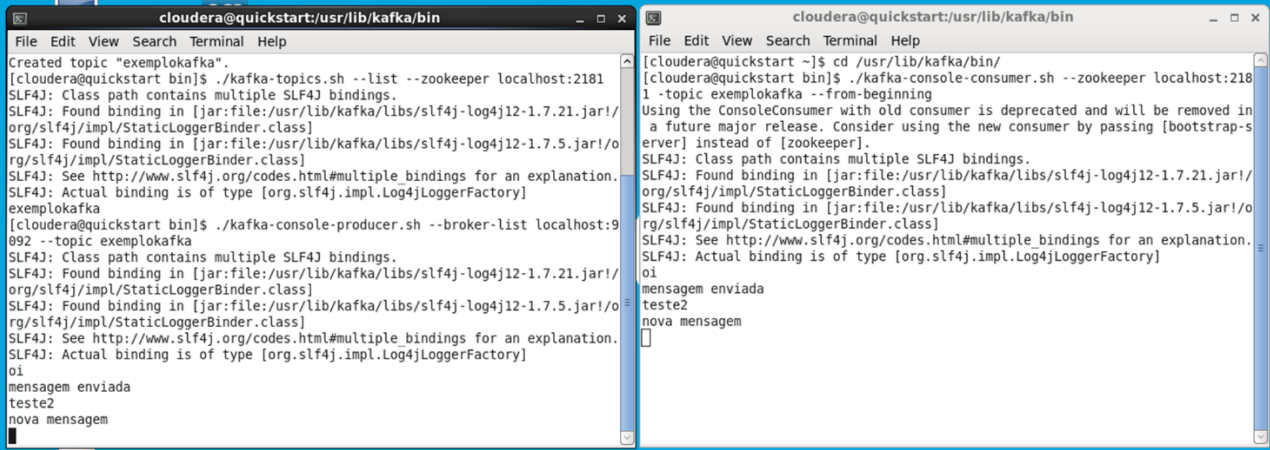
```
[cloudera@quickstart ~]$ cd /usr/lib/kafka/bin/
```

6. Agora vamos inicializar o consumer. Vamos passar por parâmetro o `--from-beginning` indicando que queremos tudo que já foi enviado ao tópico exemplokafka desde o começo.

`./kafka-console-consumer.sh --zookeeper localhost:2181 -topic exemplokafka --from-beginning`

```
[cloudera@quickstart bin]$ ./kafka-console-consumer.sh --zookeeper localhost:2181 -topic exemplokafka --from-beginning
```

7. Agora você pode escrever mensagens no terminal com o *producer* e essas mensagens serão exibidas no *consumer*.



The image shows two terminal windows side-by-side. The left window is titled 'cloudera@quickstart:usr/lib/kafka/bin' and shows the execution of 'kafka-topics.sh' to create a topic named 'exemplokafka'. It then runs 'kafka-console-producer.sh' to send messages to the topic. The output shows SLF4J warnings and the messages 'oi', 'mensagem enviada', 'teste2', and 'nova mensagem' being sent. The right window is also titled 'cloudera@quickstart:usr/lib/kafka/bin' and shows the execution of 'kafka-console-consumer.sh' to receive messages from the 'exemplokafka' topic. The output shows the same messages being received: 'oi', 'mensagem enviada', 'teste2', and 'nova mensagem'.

```
cloudera@quickstart:usr/lib/kafka/bin
File Edit View Search Terminal Help
Created topic "exemplokafka".
[cloudera@quickstart bin]$ ./kafka-topics.sh --list --zookeeper localhost:2181
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.21.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
exemplokafka
[cloudera@quickstart bin]$ ./kafka-console-producer.sh --broker-list localhost:9092 --topic exemplokafka
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.21.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
oi
mensagem enviada
teste2
nova mensagem

cloudera@quickstart:usr/lib/kafka/bin
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ cd /usr/lib/kafka/bin/
[cloudera@quickstart bin]$ ./kafka-console-consumer.sh --zookeeper localhost:2181 -topic exemplokafka --from-beginning
Using the ConsoleConsumer with old consumer is deprecated and will be removed in a future major release. Consider using the new consumer by passing [bootstrap.servers] instead of [zookeeper].
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.21.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/kafka/libs/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
oi
mensagem enviada
teste2
nova mensagem
```

Producer

Consumer