

Apache Kafka

Apache Kafka Installation

Kafka Came in 4 flavours and we can classify them as below:

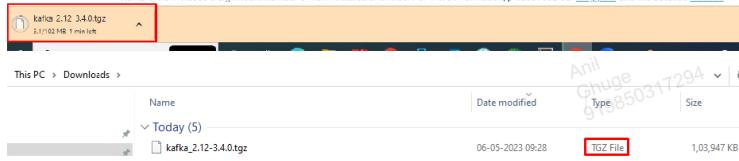
1. Open Source: Apache Kafka
2. Community Edition: Confluent.io
3. Commercial Distribution: confluent.io
4. Managed Service: confluent, amazon, aiven.io, azure kafka etc.

Download Apache Kafka

Step-1: Download the Apache Kafka, from <https://kafka.apache.org/downloads>



Step-2: when we click on the above link highlighted, it will start downloading the Kafka tar file as shown below:



Apache Kafka

Step-3: once the download of the tar file completed as shown above, copy and extract it in C:\ drive

Name	Date modified	Type	Size
kafka_2.12-3.4.0.tgz	06-05-2023 09:29	TGZ File	1,03,947 KB
kafka_2.12-3.4.0	06-05-2023 09:30	File folder	

Step-4: once extraction of the tar is done using the 7zip software or any , we can see the extracted folder. Now go inside in it and we can see one more tar file as shown below:

Name	Date modified	Type	Size
kafka_2.12-3.4.0	06-05-2023 09:31	File folder	
kafka_2.12-3.4.0.tar	06-05-2023 09:29	TAR File	1,07,900 KB

Now we can see the actual extracted the kafka binaries as shown below:

Name	Date modified	Type	Size
bin	31-01-2023 23:35	File folder	
config	31-01-2023 23:35	File folder	
libs	31-01-2023 23:35	File folder	
licenses	31-01-2023 23:35	File folder	
site-docs	31-01-2023 23:35	File folder	
LICENSE	31-01-2023 23:32	File	15 KB
NOTICE	31-01-2023 23:32	File	28 KB

Step-5: Once the download and install set up done, we can see the version of the kafka using the below command:

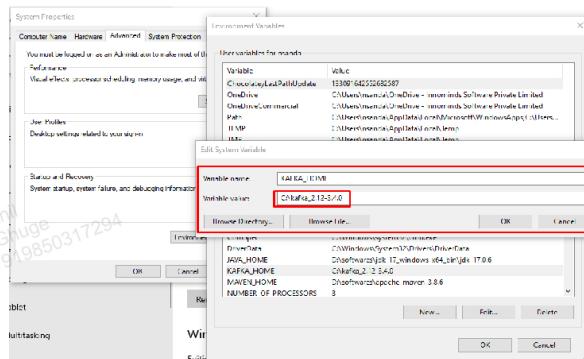
```
C:\kafka_2.12-3.4.0\bin\windows>kafka-configs.bat --version  
3.4.0 (Commit:2e1947d240607d53)
```

Anil
Ghuge
919850317294

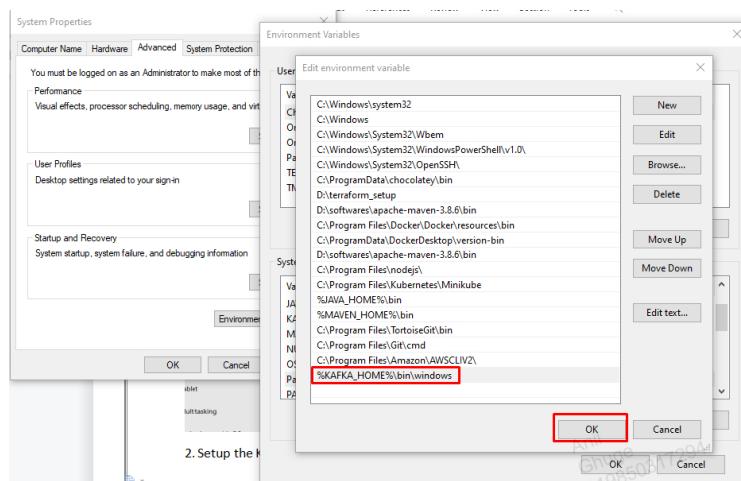
Apache Kafka

setup kafka windows directories as an environment variable

1. Setup KAKA_HOME env variable as shown below:



2. Setup the KAKA_HOME in the path directory as shown below:



3. After the KAKA_HOME setup done as shown above, go to any of the directory like D:\ or E:\ and the use the bat files directly as shown below usually which were there in \bin\windows directory.

```
D:\personal\my_trainings>kafka-configs --version  
3.4.0 (Commit:2e1947d240607d53)
```

Setup Single Node Kafka Cluster

Step-1: Spin-up a zookeeper instance. Zookeeper comes with the kafka distribution that we download

Step-2: Spin up the Kafka Broker. Once the broker is up, it register it self in the zookeeper

Step-3: zookeeper monitor and manages the health of the kafka broker.



Step-1: Spin-up a zookeeper instance

go inside the bin/windows as shown below and start the zookeeper:

Name	Date modified	Type	Size
bin	31-01-2023 23:32	Windows Batch File	1 KB
kafka	31-01-2023 23:32	Windows Batch File	1 KB
kafka-broker-api-versions	31-01-2023 23:32	Windows Batch File	1 KB
kafka-contigs	31-01-2023 23:32	Windows Batch File	1 KB
kafka-console-consumer	31-01-2023 23:32	Windows Batch File	1 KB
kafka-console-producer	31-01-2023 23:32	Windows Batch File	1 KB
kafka-consumer-groups	31-01-2023 23:32	Windows Batch File	1 KB
kafka-consumer-perf-test	31-01-2023 23:32	Windows Batch File	1 KB
kafka-delegation-tokens	31-01-2023 23:32	Windows Batch File	1 KB
kafka-delete-records	31-01-2023 23:32	Windows Batch File	1 KB
kafka-dump-log	31-01-2023 23:32	Windows Batch File	1 KB
kafka-get-offsets	31-01-2023 23:32	Windows Batch File	1 KB
kafka-leader-election	31-01-2023 23:32	Windows Batch File	1 KB
kafka-log-dir	31-01-2023 23:32	Windows Batch File	1 KB
kafka-mactodo-quorum	31-01-2023 23:32	Windows Batch File	1 KB
kafka-new-node	31-01-2023 23:32	Windows Batch File	1 KB
kafka-producer-perf-test	31-01-2023 23:32	Windows Batch File	1 KB
kafka-reassign-partitions	31-01-2023 23:32	Windows Batch File	1 KB
kafka-rebalance-evaluation	31-01-2023 23:32	Windows Batch File	1 KB
kafka-run-class	31-01-2023 23:32	Windows Batch File	6 KB
kafka-server-start	31-01-2023 23:32	Windows Batch File	2 KB
kafka-server-stop	31-01-2023 23:32	Windows Batch File	1 KB
kafka-storage	31-01-2023 23:32	Windows Batch File	1 KB
kafka-streams-application-reset	31-01-2023 23:32	Windows Batch File	1 KB
kafka-topics	31-01-2023 23:32	Windows Batch File	1 KB
kafka-transactions	31-01-2023 23:32	Windows Batch File	2 KB
zookeeper-server-start	31-01-2023 23:32	Windows Batch File	1 KB
zookeeper-server-stop	31-01-2023 23:32	Windows Batch File	2 KB
zookeeper-shell	31-01-2023 23:32	Windows Batch File	2 KB

If Kafka_home setup done, then directly go to kafka_home directory and start the zookeeper as shown below:

```
C:\kafka_2.12-3.4.0>zookeeper-server-start config/zookeeper.properties
```

[2023-05-06 09:59:39,712] INFO Reading configuration from: config/zookeeper.properties (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,733] WARN config/zookeeper.properties is relative. Prepend .\ to indicate that you're sure! (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,742] WARN \tmp/zookeeper is relative. Prepend .\ to indicate that you're sure! (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,743] INFO clientPortAddress is 0.0.0.0:2181 (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,743] INFO secureClientPort is not set (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,744] INFO observerMasterPort is not set (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,744] INFO metricsProvider.className is org.apache.zookeeper.metrics.impl.DefaultMetricsProvider (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,745] INFO autoPurge.snapRetentionCount set to 3 (org.apache.zookeeper.server.DelimiterCleanupManager)

[2023-05-06 09:59:39,746] INFO autoPurge.purgeInterval set to 60 (org.apache.zookeeper.server.DelimiterCleanupManager)

[2023-05-06 09:59:39,746] INFO Purge task is not scheduled. (org.apache.zookeeper.server.BootstrapCleanupManager)

[2023-05-06 09:59:39,746] WARN Either no config or no quorum defined in config, running in standalone mode (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,747] INFO log4j 1.2 jmx support not found; jmx disabled. (org.apache.zookeeper.jmx.ManagedIt)

[2023-05-06 09:59:39,747] INFO Reading configuration from: config/zookeeper.properties (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,747] WARN config/zookeeper.properties is relative. Prepend .\ to indicate that you're sure! (org.apache.zookeeper.server.QuorumPeerConfig)

[2023-05-06 09:59:39,750] WARN \tmp/zookeeper is relative. Prepend .\ to indicate that you're sure! (org.apache.zookeeper.server.QuorumPeerConfig)

```
C:\kafka_2.12-3.4.0>zookeeper-server-start config/zookeeper.properties
```

Once the zookeeper server stated, we can see the zookeeper data directory in /tmp/zookeeper as shown below:

Name
zookeeper

This directory is creating in the /tmp location because in the zookeeper.properties files it has been configured as shown below:

```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements. See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to you under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License. You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
```

```
 8 #           http://www.apache.org/licenses/LICENSE-2.0
 9 #
10 # Unless required by applicable law or agreed to in writing, software
11 # distributed under the License is distributed on an "AS IS" BASIS,
12 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13 # See the License for the specific language governing permissions and
14 # limitations under the License.
15 #
16 # The snapshot is stored.
17 #dataDir=/tmp/.snapshot
18 #
19 # the port at which the clients will connect
20 clientPort=2283
21 #
22 # Set the max-connections limit on the number of connections since this is a bind
23 # socket
24 maxClients=50
25 #
26 # Disable the adminserver by default to avoid port conflicts.
27 #
28 # Set the port to something non-conflicting if choosing to enable this
29 admin.enableServer=false
30 #
31 # admin.serverPort=8080
```

Apache Kafka

Step-2: Spin up the Kafka Broker

To Setup Kafka broker, we need to run the “kafka-server-start.bat” file by supplying the server.properties file.

To do this, go to KAFKA_HOME directory, and issue the below command:

```
C:\kafka_2.12-3.4.0>.\bin\windows\kafka-server-start.bat .\config\server.properties  
(OR)  
C:\kafka_2.12-3.4.0>kafka-server-start config/server.properties
```

As soon as the Kafka server started, it is called as the broker-1, which starts listening to the consumers and producers on the port 9092, also a kafka-logs directory will be created in /tmp directory as shown below:

Name	Date modified	Type
kafka-logs	06-05-2023 10:14	File folder
zookeeper	06-05-2023 09:59	File folder

This directory has been created, because it was mentioned in the server properties file as shown below:

```
59 ##### Log Basics #####
60
61 # A comma separated list of directories under which to store log files
62 log.dirs=/tmp/kafka-logs
63
```

Also this will be treated as the first broker or broker-1 because of the broker id configured in server.properties as shown below:

Apache Kafka

```
21 ##### Server Basics #####
22
23 # The id of the broker. This must be set to a unique integer for each broker.
24 broker.id=0
25
```

Broker connects to the zookeeper as it was mentioned in the server.properties file as shown below:

```
118 ##### Zookeeper #####
119
120 # Zookeeper connection string (see zookeeper docs for details).
121 # This is a comma separated host:port pairs, each corresponding to a zk
```

```

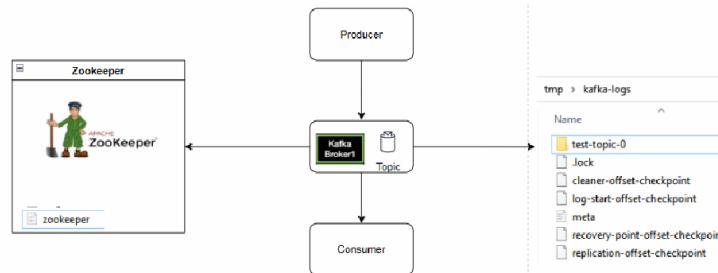
122 # server. e.g. "127.0.0.1:5000,127.0.0.1:5001,127.0.0.1:5002".
123 # You can also append an optional chroot string to the urls to specify the
124 # root directory for all kafka znodes.
125 zookeeper.connect=localhost:2181
126
127 # Timeout in ms for connecting to zookeeper
128 zookeeper.connection.timeout.ms=18000
...

```

By this now we completed

- ✓ Zookeeper server start
- ✓ Kafka-server start

Hence as only one broker started and connected to zookeeper, we can say it as the single-node cluster.



Step-3: Create a topic

A. we need a topic inside the broker to create the topic use the below command:

Create a Topic

Apache Kafka

```
kafka-topics.bat --bootstrap-server localhost:9092 --create --topic test-topic --partitions=1 --replication-factor 1
```

```
C:\kafka_2.12-3.4.0>kafka-topics --bootstrap-server localhost:9092 --create --topic test-topic --partitions=1 --replication-factor 1
Created topic test-topic.
```

Once the topic is created we can see it as a directory, inside the /tmp/kafka-logs dir.

Name	Date modified	Type	Size
test-topic-0	06-05-2023 10:44	File folder	
.lock	06-05-2023 10:08	LOCK File	0 KB
cleaner-offset-checkpoint	06-05-2023 10:08	File	0 KB
log-start-offset-checkpoint	06-05-2023 10:46	File	1 KB
meta	06-05-2023 10:43	Properties Source ...	1 KB
recovery-point-offset-checkpoint	06-05-2023 10:46	File	1 KB
replication-offset-checkpoint	06-05-2023 10:46	File	1 KB

When we go inside the test-topic-0 directory, we can see the following files:

Local Disk (C:) > tmp > kafka-logs > test-topic-0			
Name	Date modified	Type	Size
00000000000000000000000000000000	06-05-2023 10:44	INDEX File	10,240 KB
00000000000000000000000000000000	06-05-2023 10:44	Text Document	0 KB
00000000000000000000000000000000.timeindex	06-05-2023 10:44	TIMEINDEX File	10,240 KB
leader-epoch-checkpoint	06-05-2023 10:44	File	1 KB
partition	06-05-2023 10:44	METADATA File	1 KB

Here the highlighted file is called as a log file, which will store the data sent by the producers and makes this data available for the consumers.

Step-4: Create a Producer and send the data as shown below:

```
C:\kafka_2.12-3.4.0>kafka-console-producer --broker-list localhost:9092 --topic test-topic < sample1.csv
```

```
C:\kafka_2.12-3.4.0>kafka-console-producer --broker-list localhost:9092 --topic test-topic < sample1.csv
```

Once the producer sent the data to the topic then we can see the logfile size increased as shown below:

Anil
Ghuge
919850317294

Apache Kafka				
Name	Date modified	Type	Size	
00000000000000000000000000000000	06-05-2023 10:44	INDEX File	10,240 KB	
00000000000000000000000000000000	06-05-2023 10:52	Text Document	185 KB	
00000000000000000000000000000000.timeindex	06-05-2023 10:44	TIMEINDEX File	10,240 KB	
leader-epoch-checkpoint	06-05-2023 10:44	File	1 KB	
partition	06-05-2023 10:44	METADATA File	1 KB	

So this is indicating that the data produced by the producer is storing in the log file at the topic of the broker.

We can see the data of the log file only with the kafka-dump-log utility as shown below:

```
C:\tmp\kafka-logs\test-topic-0>kafka-dump-log --files 00000000000000000000.log
```

```
c:\tmp\kafka-logs\test-topic-0>kafka-dump-log --files 00000000000000000000.log
Dumping 00000000000000000000.log
log starting offset: 0
baseOffset: 169 count: 163 baseSequence: 0 lastSequence: 169 producerId: 0 partitionLeaderEpoch: 0 isTransactional: false idControl: false deleteHorizonMs: OptionalLong.empty position: 0 createTime: 160330577663 size: 1605 words: 2 compressCodecs: none crc: 7d1008f1
isvalid: true
baseOffset: 163 lastOffset: 315 count: 163 baseSequence: 325 producerId: 0 producerEpoch: 0 partitionLeaderEpoch: 0 isTransactional: false idControl: false deleteHorizonMs: OptionalLong.empty position: 16303 createTime: 160330577663 size: 1620 magic: 2 compressCodecs: none crc: 7d1008f1
baseOffset: 325 lastOffset: 482 count: 167 baseSequence: 325 lastSequence: 482 producerId: 0 producerEpoch: 0 partitionLeaderEpoch: 0 isTransactional: false idControl: false deleteHorizonMs: OptionalLong.empty position: 16443 createTime: 160330577663 size: 1628 magic: 2 compressCodecs: none crc: 5d2957685 isvalid: true
baseOffset: 482 lastOffset: 655 count: 163 baseSequence: 483 lastSequence: 655 producerId: 0 producerEpoch: 0 partitionLeaderEpoch: 0 isTransactional: false idControl: false deleteHorizonMs: OptionalLong.empty position: 16012 createTime: 160330577696 size: 1625 magic: 2 compressCodecs: none crc: 5d2957685 isvalid: true
baseOffset: 655 lastOffset: 820 count: 163 baseSequence: 656 lastSequence: 820 producerId: 0 producerEpoch: 0 partitionLeaderEpoch: 0 isTransactional: false idControl: false deleteHorizonMs: OptionalLong.empty position: 16377 createTime: 160330577699 size: 1628 magic: 2 compressCodecs: none crc: 5d2957685 isvalid: true
baseOffset: 820 lastOffset: 984 count: 164 baseSequence: 821 lastSequence: 984 producerId: 0 producerEpoch: 0 partitionLeaderEpoch: 0 isTransactional: false idControl: false deleteHorizonMs: OptionalLong.empty position: 161623 createTime: 160330577704 size: 1630 magic: 2 compressCodecs: none crc: 5d2957685 isvalid: true
```

Step-4: Create Consumer

Using kafka-console-consumer

```
C:\kafka_2.12-3.4.0>kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test-topic --from-beginning
```

```
\kafka_2.12-3.4.0\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test-topic --from-beginning
ZYLOG,BE,0.8,0.9,0.8,0.85,0.85,2316,1993.8,05-NOV-2018,11,INE225I01026,
Processed a total of 1967 messages
terminate batch job (Y/N)?
```

Multi Node Cluster Setup

UseCase-2: Creating the Topic with 4 partitions-without message key

1. Create topic:

```
kafka-topics.bat --create --topic multi-partitions-topic --replication-factor 1 --partitions 4 --bootstrap-server localhost:9092
```

2. Instantiate Console Producer without key

```
kafka-console-producer.bat --broker-list localhost:9092 --topic multi-partitions-topic
```

```
C:\kafka_2.12-3.1.0\bin\windows>kafka-console-producer.bat --broker-list localhost:9092 --topic multi-partitions-topic
```

Publish the data as shown below:

```
C:\kafka_2.12-3.1.0\bin\windows>kafka-console-producer.bat --broker-list localhost:9092 --topic multi-partitions-topic
>APPLE
>ALPHA
>ADAM
>ANGE
```

```
>2  
>3  
>4  
>5  
>6  
>7  
>89  
>
```

3. Instantiate Console Consumer without key

```
kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic multi-partitions-topic --from-beginning
```

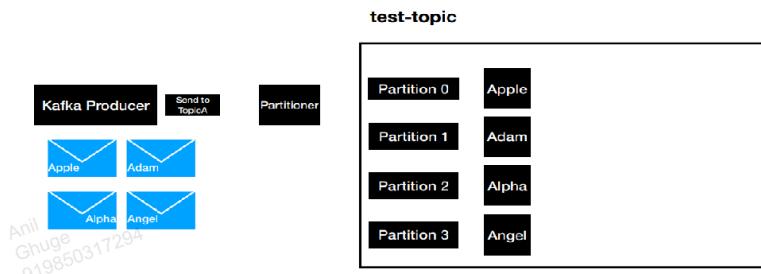
```
C:\kafka_2.12-3.1.0\bin\windows>kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic multi-partitions-topic --from-beginning
```

Now it will consume the data without any order as shown below:

```
C:\kafka_2.12-3.1.0\bin\windows>kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic multi-partitions-topic --from-beginning  
ALPHA  
ANGEL  
4  
APPLE  
5  
6  
7  
ADAM  
8  
9  
99
```

Apache Kafka

This is because as the messages are sending without keys, the messages are assigned all available partitions using the round robin algorithm as shown below:



UseCase-3: Creating the Topic with 4 partitions-with message key

1. Create topic with 4 partitions

```
kafka-topics.bat --create --topic multi-partitions-topic --replication-factor 1 --partitions 4 --bootstrap-server localhost:9092
```

```
C:\kafka_2.12-3.1.0\bin\windows>kafka-topics.bat --create --topic multi-partitions-topic --replication-factor 1 --partitions 4 --bootstrap-server localhost:9092  
Created topic multi-partitions-topic.
```

2. Instantiate Console Producer with key

```
kafka-console-producer.bat --broker-list localhost:9092 --topic multi-partitions-topic --property "key.separator=-" --property "parse.key=true"
```

```
C:\kafka_2.12-3.1.0\bin\windows>kafka-console-producer.bat --broker-list localhost:9092 --topic multi-partitions-topic --property "key.separator=-" --property "parse.key=true"
```

Publish the data as shown below:

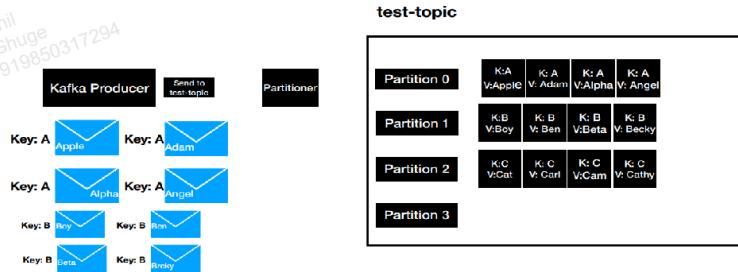
```
C:\kafka_2.12-3.1.0\bin\windows>kafka-console-producer.bat --broker-list localhost:9092 --topic multi-partitions-topic --property "key.separator=-" --property "parse.key=true"  
4  
APPLE  
5  
ANGEL  
6  
CAT  
7  
BALL
```

3. Instantiate the consumer with key

```
kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic multi-partitions-topic --from-beginning --property "key.separator=-" --property "print.key=true"
```

Apache Kafka

Now we can see the data populated in an order in consumer this is because, if the key is same, the message will be sent to the same partition. Partitioner will apply hashing technique on key of each message coming to the partition and identify the same partition with the same key for different values and send the data to only one partition.



Consumer Offset

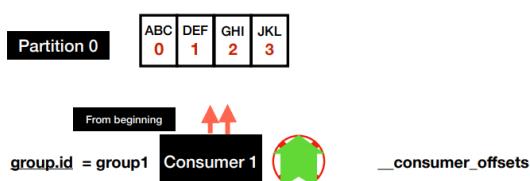
Every Kafka message within a single partition is uniquely identified by the offset. For example, the offset for the first message in the partition would be 0000, the offset for the second message would be 0001 and so on.

Any message that produced into the topic will have a unique ID called offset. Consumers have the 3 options when it comes to reading the topic:

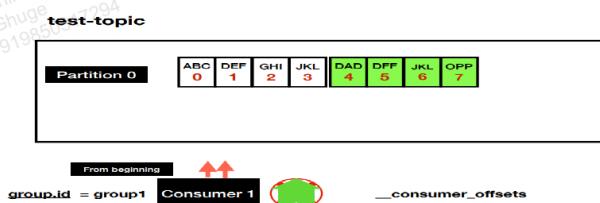
1. Read the message from the begining
 2. Latest: meaning the message only read after the consumer started
 3. Specific Offset: meaning the read the messages in the topic, by passing a specific off set value from the consumer.

Lets say we have one partition and it have some records in it. Lets say we have a consumer, which is going to read the messages from the begining for any kafka consumer it is required that the consumer to provide the group id. Consumer can poll and retrieve the multiple records at the same time. as a process, each message it moves the consumer read off set one by one.

Apache Kafka



lets say some reason the consumer is crashed, while the consumer was down, the producer of the topic some more messages.



now the consumer is brought up after some time, so how does it know that it needs to read from off set -4. the consumer off set in general stores in an internal topic called consumer_offsets

Consumer offsets used by the consumer to start reading the messages from the point it left off.

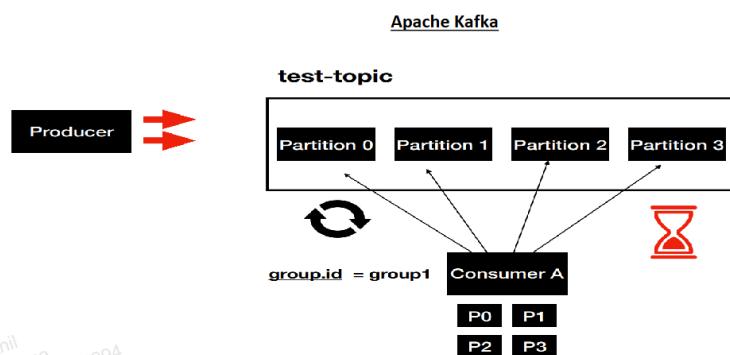
```
C:\kafka_2.12-3.1.0\bin\windows>kafka-topics.bat --bootstrap-server localhost:9092 --list  
_consumer_offsets  
multi-partitions-topic
```

```
multi-partitions-topic1  
multi-partitions-topic2  
test-topic
```

Consumer Groups

GroupID in Consumer Group plays a key role in Consumer Group. Lets have a test-topic with the four partitions, now we have the consumer-A with the groupID group-1, we have only one single consumer pulling all the 4 partitions in the topic and processing them in this case a single thread trying to poll all the partitions.

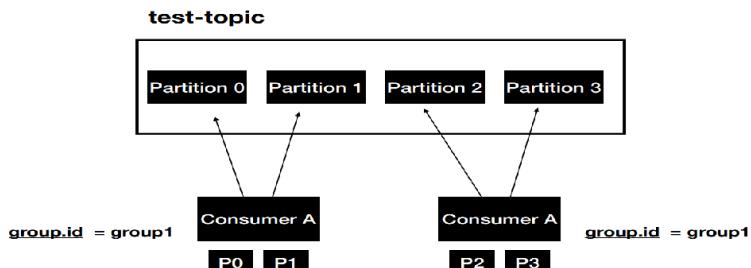
Anil
Ghuge
919850317294



Lets say the producer of the topic is producing messages at a faster way, then then consumer consuming.

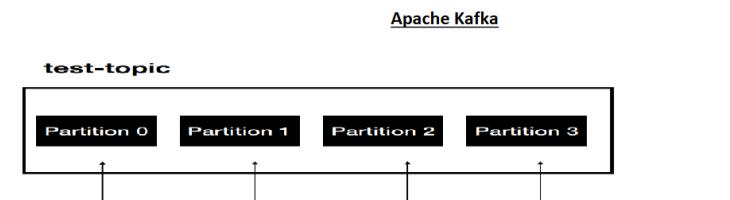
in that case, there will lag we can see at the consumer side and end up with the not processing the real time data. this is where the consumer group comes into the use.

lets spin up another consumer A with the same group id. now the partitions of the topic split between the two consumers, partition-0 and partition-1 are taking care of the fist consumer instance and p2, p3 are taking care of the second instance. thus we scaled our message consumption. this will help process the records little faster.



lets make it more faster by spinning up 2 more instances with the same grop id. now we have 5 consumer instances but only 4 partitions are available in that case one of the instance will be idle. this will leads to insufficient usage of the resources.

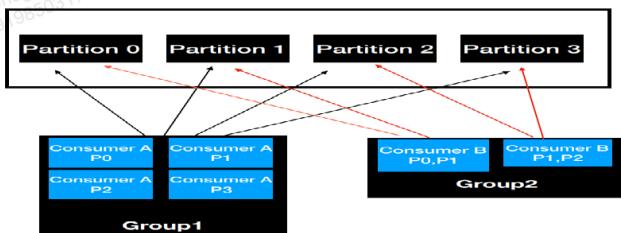
Anil
Ghuge
919850317294





Consumer Group with Multiple Groups

test-topic



Consumer Groups are used for scalable message consumption

- Each different application will have a unique consumer group
- Who manages the consumer group?
- Kafka Broker manages the consumer-groups
- Kafka Broker acts as a Group Co-ordinator

Instantiating the Console Consumer with Group Id

```
kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test-topic --group <group-name>
```

Execute the same command again to have one more instance of kafka consumer.

Here group name we have to use as the result of view consumers group command.

view consumer groups

```
kafka-consumer-groups.bat --bootstrap-server localhost:9092 --list
```

Ghuge
919850317294

Apache Kafka

Consumer Groups and their Offset

```
kafka-consumer-groups.bat --bootstrap-server localhost:9092 --describe --group console-consumer-27773
```

Instantiate the console producer:

```
kafka-console-producer.bat --broker-list localhost:9092 --topic test-topic
```

And stat giving the messages on console as shown below: 1 2 3 4

Then see 1and 4 are comes to consumer1 and 2 and 3 goes to consumer 2

Multi Node Cluster Setup

Install the confluent Kafka

1. Go to the location: https://docs.confluent.io/platform/current/installation/installing_cp/zip-tar.html
2. Copy the confluent platform zip

Confluent Platform

ZIP

```
curl -o http://packages.confluent.io/archive/7.1/confluent-7.1.1.zip
```

3. Open the command prompt and enter the copied text as below:

Anil
Ghuge
919850317294