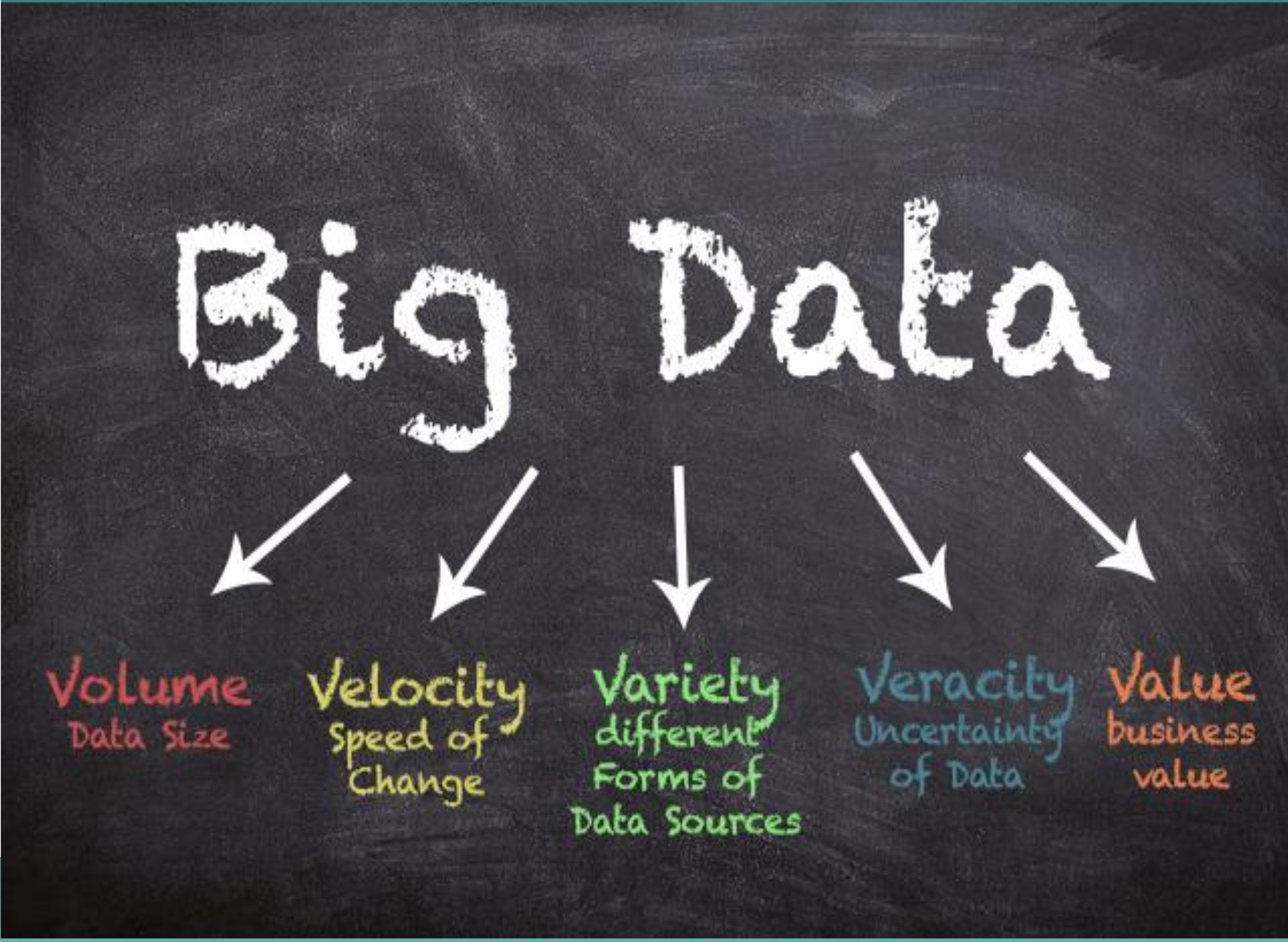
10/01/2018 | By: Ajit K Prasad



Big Data Engineering with Hadoop & Spark

Kafka Introduction

APACHE kafka
A distributed streaming platform

APACHE Spark

Acadgild

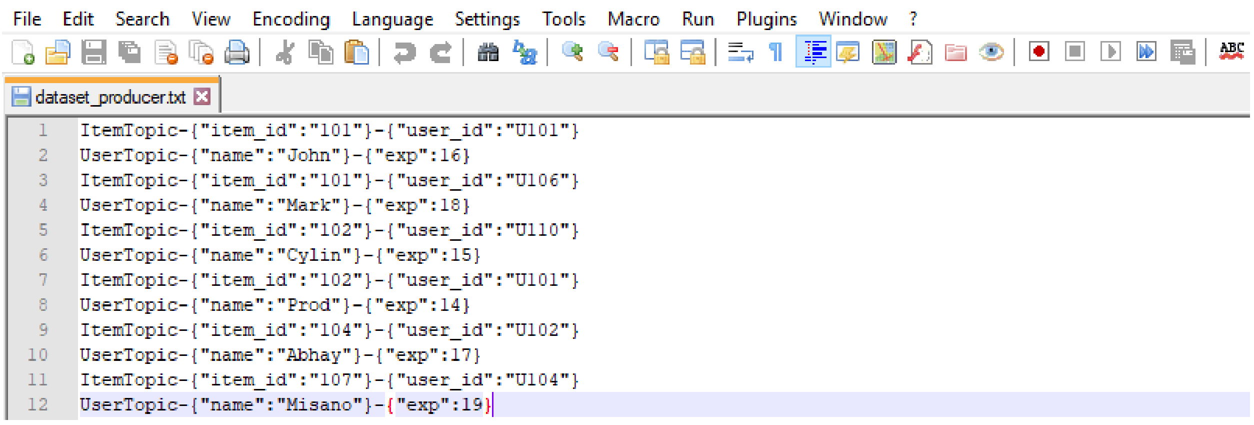
Session 23: Assignment 23.1

This assignment is aimed at consolidating the concepts that was learnt during the Apache Kafka session of the course.

Dataset:

Download the dataset from this *link*.

This is how the dataset looks:


File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
dataset_producer.txt

```
ItemTopic-{"item_id":"101"}-{"user_id":"U101"}
UserTopic-{"name":"John"}-{"exp":16}
ItemTopic-{"item_id":"101"}-{"user_id":"U106"}
UserTopic-{"name":"Mark"}-{"exp":18}
ItemTopic-{"item_id":"102"}-{"user_id":"U110"}
UserTopic-{"name":"Cylin"}-{"exp":15}
ItemTopic-{"item_id":"102"}-{"user_id":"U101"}
UserTopic-{"name":"Prod"}-{"exp":14}
ItemTopic-{"item_id":"104"}-{"user_id":"U102"}
UserTopic-{"name":"Abhay"}-{"exp":17}
ItemTopic-{"item_id":"107"}-{"user_id":"U104"}
UserTopic-{"name":"Misano"}-{"exp":19}
```

This file has two topics namely:

1. **ItemTopic**: It has item_id and user_id
2. **UserTopic**: It has user details like name and experience

Task 1:

Create a java program MyKafkaProducer.java that takes a file name and delimiter as input arguments. It should read the content of file line by line. Fields in the file are in following order:

1. Kafka Topic Name
2. Key
3. value

– For every line, insert the key and value to the respective Kafka broker in a fire and forget mode.
– After record is sent, it should print appropriate message on screen.
– Pass dataset_producer.txt as the input file and - as delimiter.

Solution:

1. Program to perform this task is as below:
   – *Imports required for the program is given below:*

```
import org.apache.kafka.clients.producer.KafkaProducer;
import org.apache.kafka.clients.producer.ProducerRecord;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.Properties;
```

- *This is the class called "MyKafkaProducer" which takes two arguments (Input File name and delimiter) in the command line*

```
public class MyKafkaProducerLive
{
	public static void main(String[] args) throws IOException
	{
		if (args.length != 2)
		{
			System.out.println("Please provide appropriate command line arguments");
			System.exit(-1);
		}
```

- *We configure the properties for KafkaProducer:*
    - *We create a new instance of Properties called props*
    - *Using this instance we add properties to kafkaProducer like, bootstrapserver/meta-data-brokerlist, key and value serializers*

```
		Properties props = new Properties();
		props.put("bootstrap.servers", "localhost:9092");
		props.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer");
		props.put("value.serializer", "org.apache.kafka.common.serialization.StringSerializer");
```

- *We then instantiate the KafkaProducer class called producer, we have mentioned string in <> because both key and value are String*
- *We add the properties instance (props)to KafkaProducer instance*
- *We also instantiate ProducerRecord as producerRecord*

```
		KafkaProducer<String, String> producer = new KafkaProducer<>(props);
		ProducerRecord<String, String> producerRecord = null;
```

- *Now we take the data provided in the command line i.e. file name and delimiter and save them in the array of string variables called filename and delimiter*

```
		String fileName = args[0];
		String delimiter = args[1];
```

- *We read the contents of the input file, and save their contents arrays in different variables:*
    - *We save the topic name, i.e. first part of array(0th index elements) in String variable topic and similarly we save key and value variables too*

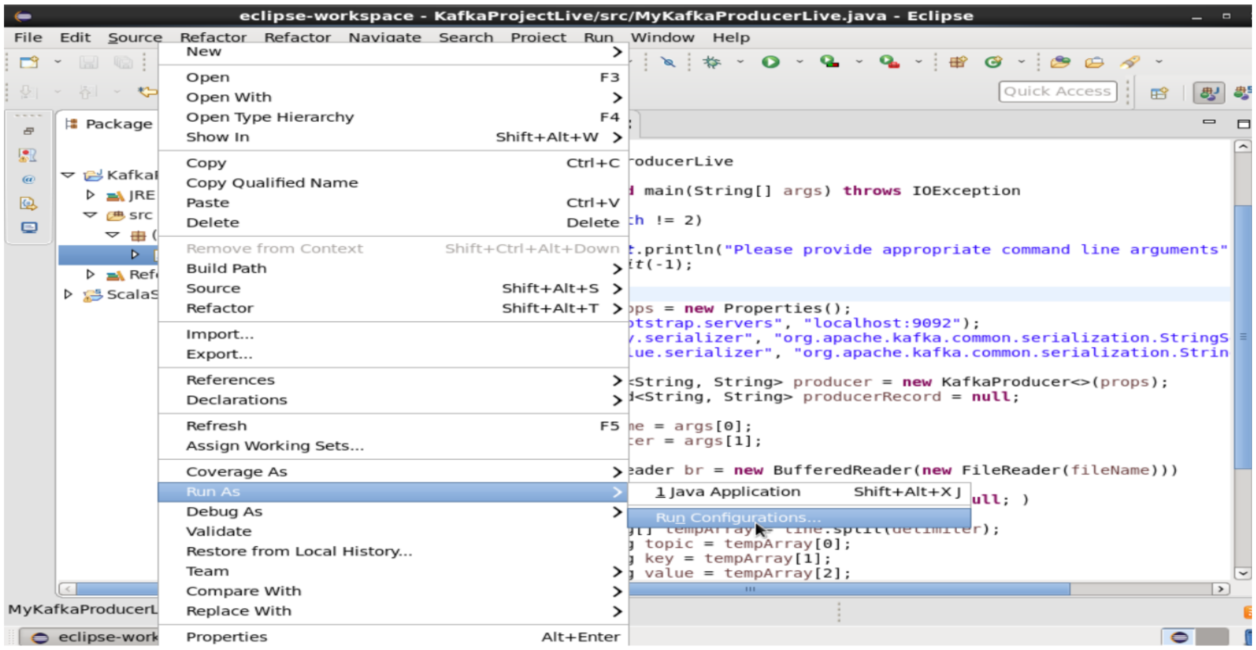
```
		try(BufferedReader br = new BufferedReader(new FileReader(fileName)))
		{
			for(String line; (line = br.readLine()) != null; )
			{
				String[] tempArray = line.split(delimiter);
				String topic = tempArray[0];
				String key = tempArray[1];
				String value = tempArray[2];
```

- *Now, we pass the variables topic, key and value to producer record*
- *We also print appropriate message which shows the topics, key and value contents*
- *We finally, close the producer*

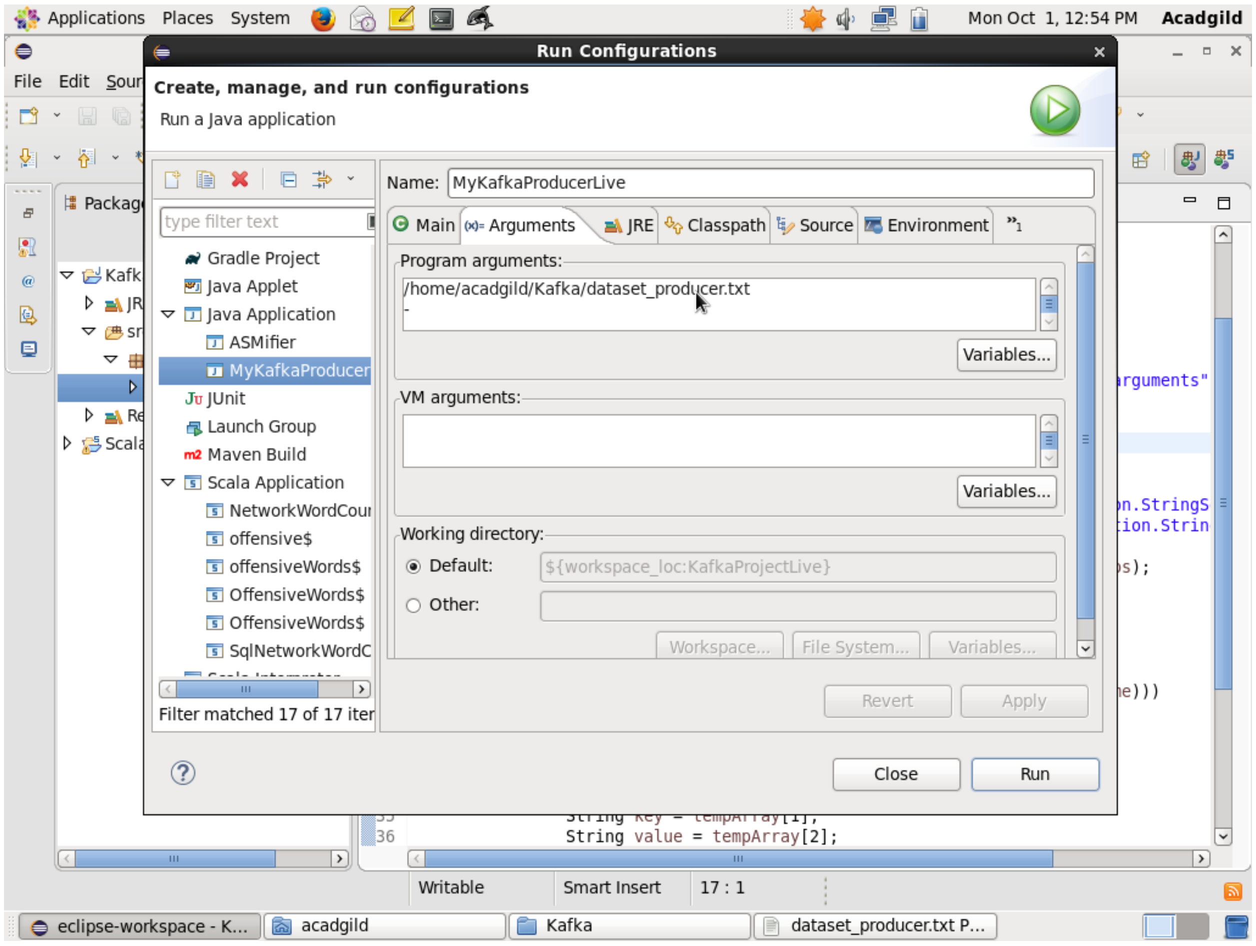
```
				producerRecord = new ProducerRecord<String, String>(topic, key, value);
				producer.send(producerRecord);
				System.out.printf("Record sent to topic:%s. Key:%s, Value:%s\n", topic, key, value);
			}
		}

		producer.close();
	}
}
```

2. Next, we start Zookeeper and Kafka Server using the following commands:
    - *$ $KAFKA_HOME/bin/zookeeper-server-start.sh $KAFKA_HOME/config/zookeeper.properties*
    - *$ $KAFKA_HOME/bin/kafka-server-start.sh $KAFKA_HOME/config/server.properties*

3. Next, we pass the arguments in "Run Configurations", and click on "Run" as shown below:



```
roducerLive
d main(String[] args) throws IOException
th != 2)
t.println("Please provide appropriate command line arguments"
it(-1);
ops = new Properties();
otstrap.servers", "localhost:9092");
y.serializer", "org.apache.kafka.common.serialization.StringS
lue.serializer", "org.apache.kafka.common.serialization.Strin
<String, String> producer = new KafkaProducer<>(props);
d<String, String> producerRecord = null;
me = args[0];
ter = args[1];
eader br = new BufferedReader(new FileReader(fileName)))
ull; )
g topic = tempArray[0];
g key = tempArray[1];
g value = tempArray[2];
```



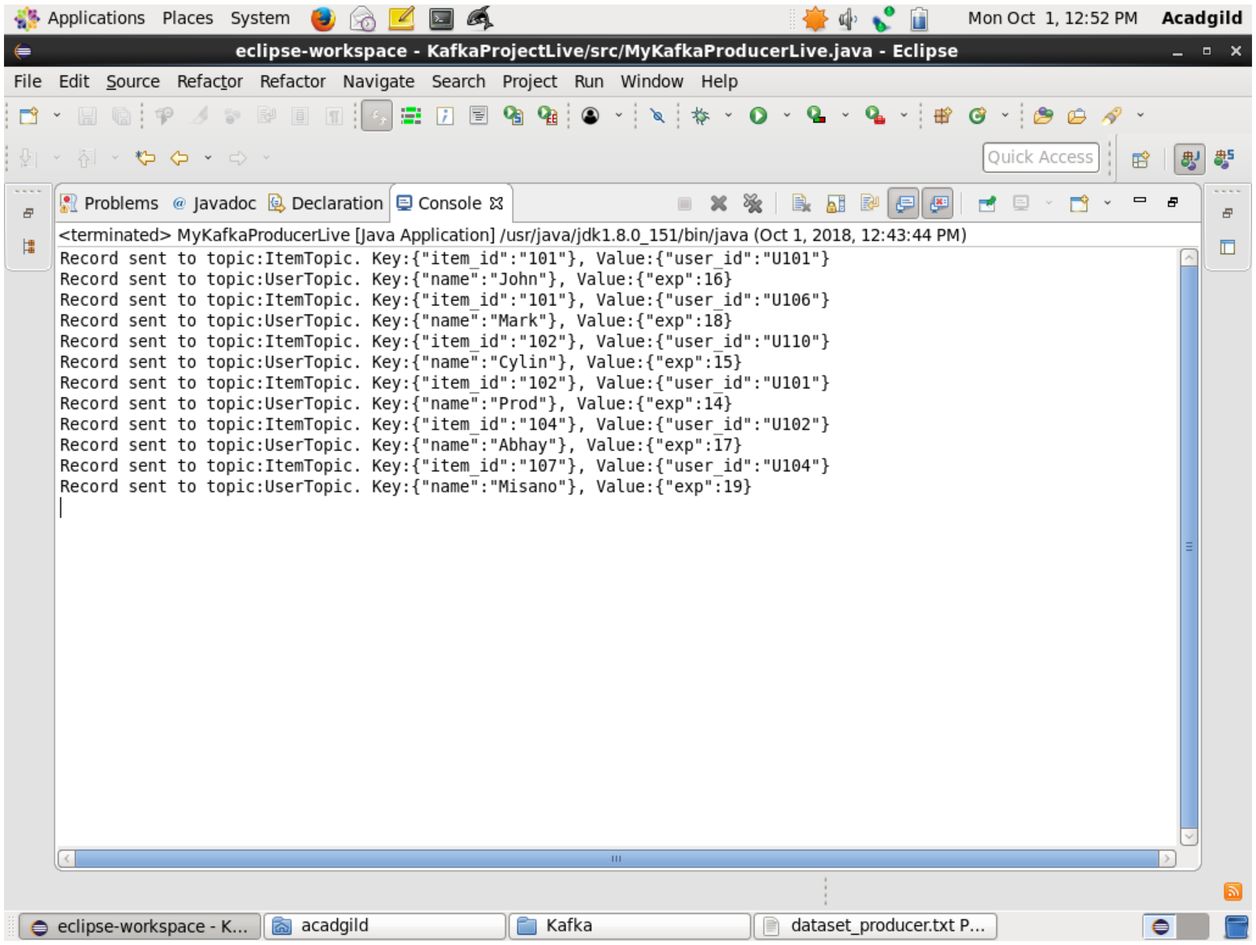
Name: MyKafkaProducerLive

Program arguments:
/home/acadgild/Kafka/dataset_producer.txt
-

VM arguments:

Working directory:
Default: ${workspace_loc:KafkaProjectLive}
Other:

```
String value = tempArray[2];
```



Applications Places System
Mon Oct 1, 12:52 PM Acadgild
eclipse-workspace - KafkaProjectLive/src/MyKafkaProducerLive.java - Eclipse
File Edit Source Refactor Refactor Navigate Search Project Run Window Help
Quick Access
Problems Javadoc Declaration Console
eclipse-workspace - K... acadgild Kafka dataset_producer.txt P...

```
<terminated> MyKafkaProducerLive [Java Application] /usr/java/jdk1.8.0_151/bin/java (Oct 1, 2018, 12:43:44 PM)
Record sent to topic:ItemTopic. Key:{"item_id":"101"}, Value:{"user_id":"U101"}
Record sent to topic:UserTopic. Key:{"name":"John"}, Value:{"exp":16}
Record sent to topic:ItemTopic. Key:{"item_id":"101"}, Value:{"user_id":"U106"}
Record sent to topic:UserTopic. Key:{"name":"Mark"}, Value:{"exp":18}
Record sent to topic:ItemTopic. Key:{"item_id":"102"}, Value:{"user_id":"U110"}
Record sent to topic:UserTopic. Key:{"name":"Cylin"}, Value:{"exp":15}
Record sent to topic:ItemTopic. Key:{"item_id":"102"}, Value:{"user_id":"U101"}
Record sent to topic:UserTopic. Key:{"name":"Prod"}, Value:{"exp":14}
Record sent to topic:ItemTopic. Key:{"item_id":"104"}, Value:{"user_id":"U102"}
Record sent to topic:UserTopic. Key:{"name":"Abhay"}, Value:{"exp":17}
Record sent to topic:ItemTopic. Key:{"item_id":"107"}, Value:{"user_id":"U104"}
Record sent to topic:UserTopic. Key:{"name":"Misano"}, Value:{"exp":19}
```

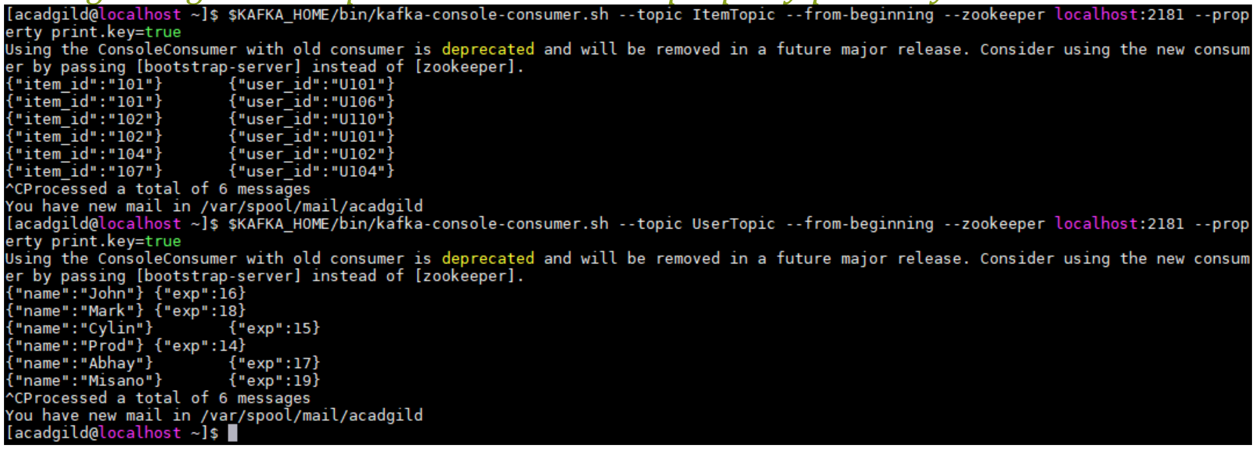
4. Next, we run the console consumer commands on terminal to view the output of the program, using the below command:

- To read contents of ItemTopic:

  $ $KAFKA_HOME/bin/kafka-console-consumer.sh --topic ItemTopic --from-beginning --zookeeper localhost:2181 --property print.key=true

- To read contents of UserTopic:

  $ $KAFKA_HOME./bin/kafka-console-consumer.sh --topic UserTopic --from-beginning --zookeeper localhost:2181 --property print.key=true



```
[acadgild@localhost ~]$ $KAFKA_HOME/bin/kafka-console-consumer.sh --topic ItemTopic --from-beginning --zookeeper localhost:2181 --prop
erty print.key=true
Using the ConsoleConsumer with old consumer is deprecated and will be removed in a future major release. Consider using the new consum
er by passing [bootstrap-server] instead of [zookeeper].
{"item_id":"101"}       {"user_id":"U101"}
{"item_id":"101"}       {"user_id":"U106"}
{"item_id":"102"}       {"user_id":"U110"}
{"item_id":"102"}       {"user_id":"U101"}
{"item_id":"104"}       {"user_id":"U102"}
{"item_id":"107"}       {"user_id":"U104"}
^CProcessed a total of 6 messages
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$ $KAFKA_HOME/bin/kafka-console-consumer.sh --topic UserTopic --from-beginning --zookeeper localhost:2181 --prop
erty print.key=true
Using the ConsoleConsumer with old consumer is deprecated and will be removed in a future major release. Consider using the new consum
er by passing [bootstrap-server] instead of [zookeeper].
{"name":"John"} {"exp":16}
{"name":"Mark"} {"exp":18}
{"name":"Cylin"}        {"exp":15}
{"name":"Prod"} {"exp":14}
{"name":"Abhay"}        {"exp":17}
{"name":"Misano"}       {"exp":19}
^CProcessed a total of 6 messages
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$
```

Task 2:

Modify the previous program MyKafkaProducer.java and create a new Java program KafkaProducerWithAck.java.

- This should perform the same task as of KafkaProducer.java with some modification.
    - When passing any data to a topic, it should wait for acknowledgement.
    - After acknowledgement is received from the broker, it should print the key and value which has been written to a specified topic.
    - The application should attempt for 3 retries before giving any exception.
- Pass dataset_producer.txt as the input file and -as delimiter.

Solution:

1. Program to perform the task:

- *Imports required for the program is given below:*

```
import org.apache.kafka.clients.producer.KafkaProducer;
import org.apache.kafka.clients.producer.ProducerRecord;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.Properties;
import java.util.concurrent.ExecutionException;
```

- *This is the class called "MyKafkaProducerWithAck" which takes two arguments (Input File name and delimiter) in the command line*

```
public class MyKafkaProducerWithAck
{
        public static void main(String[] args) throws IOException,
InterruptedException, ExecutionException
        {
                if (args.length != 2)
                {
                        System.out.println("Please provide appropriate
command line arguments");
                        System.exit(-1);
                }
```

- *We configure the properties for KafkaProducer:*
    - *We create a new instance of Properties called props*
    - *Using this instance we add properties to kafkaProducer like, bootstrapserver/meta-data-brokerlist, key and value serializers,acks and retries*
        - *Acks "all"- this means that the producer will receive a success response from the broker once all in-sync replicas received the message*
        - *Retries 3- When the producer receives an error message from the server, the error could be transient (e.g., a lack of leader for a partition). In this case, the value of the retries parameter will control how many times the producer will retry sending the message before giving up and notifying the client of an issue*

```
            Properties props = new Properties();
            props.put("bootstrap.servers", "localhost:9092");
            props.put("acks", "all");
            props.put("retries", 3);
            props.put("key.serializer",
"org.apache.kafka.common.serialization.StringSerializer");
            props.put("value.serializer",
"org.apache.kafka.common.serialization.StringSerializer");
```

- *We then instantiate the KafkaProducer class called producer, we have mentioned string in <> because both key and value are String*
- *We add the properties instance (props)to KafkaProducer instance*
- *We also instantiate ProducerRecord as producerRecord*

```
            KafkaProducer<String, String> producer = new
KafkaProducer<>(props);
            ProducerRecord<String, String> producerRecord = null;
```

- *Now we take the data provided in the command line i.e. file name and delimiter and save them in the array of string variables called filename and delimiter*

```
            String fileName = args[0];
            String delimiter = args[1];
```

- *We read the contents of the input file, and save their contents arrays in different variables:*
    - *We save the topic name i.e. first part of array(0th index elements) in String variable topic and similarly we save key and value variables too*

```
                try(BufferedReader br = new BufferedReader(new
FileReader(fileName)))
                {
                        for(String line; (line = br.readLine()) != null; )
                        {
                                String[] tempArray = line.split(delimiter);
                                String topic = tempArray[0];
                                String key = tempArray[1];
                                String value = tempArray[2];
```

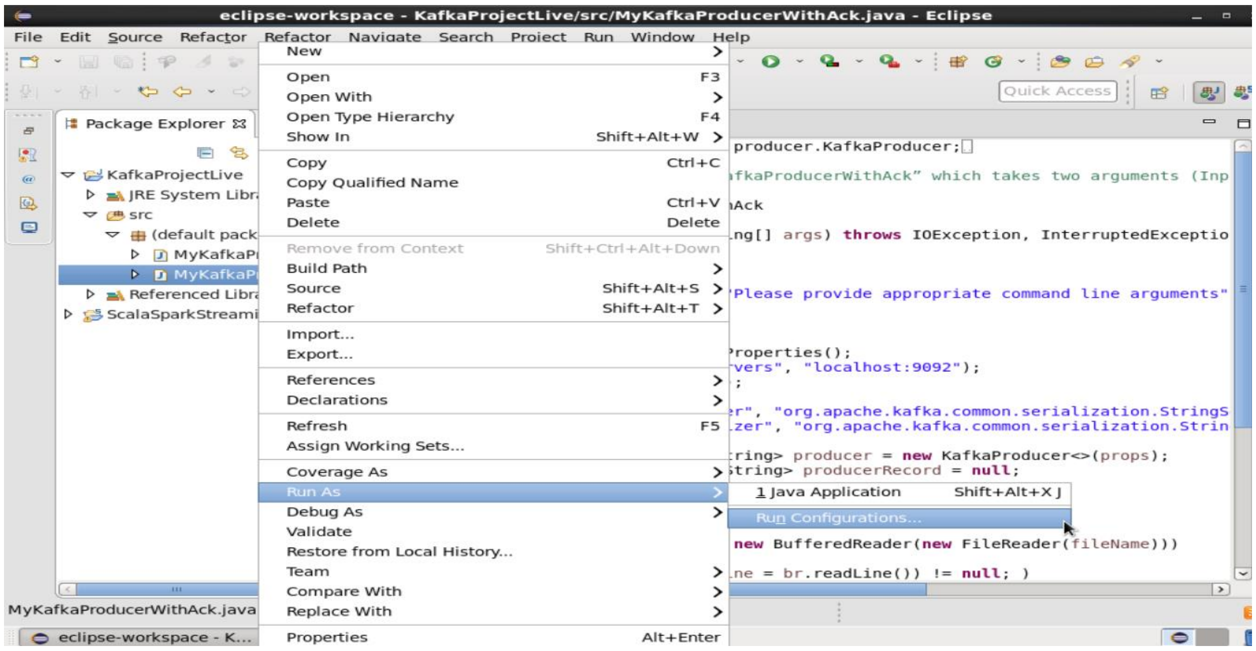
- *Now, we pass the variables topic, key and value to producer record.*
- *We also print appropriate message which shows the topics, key and value contents.*
- *We finally, close the producer*

```
                                producerRecord = new ProducerRecord<String,
String>(topic, key, value);
                                producer.send(producerRecord).get();
                                System.out.printf("Record sent to topic:%s and
acknowledged as well. Key:%s,Value:%s\n", topic, key, value);
                        }
                }

                producer.close();
        }
}
```

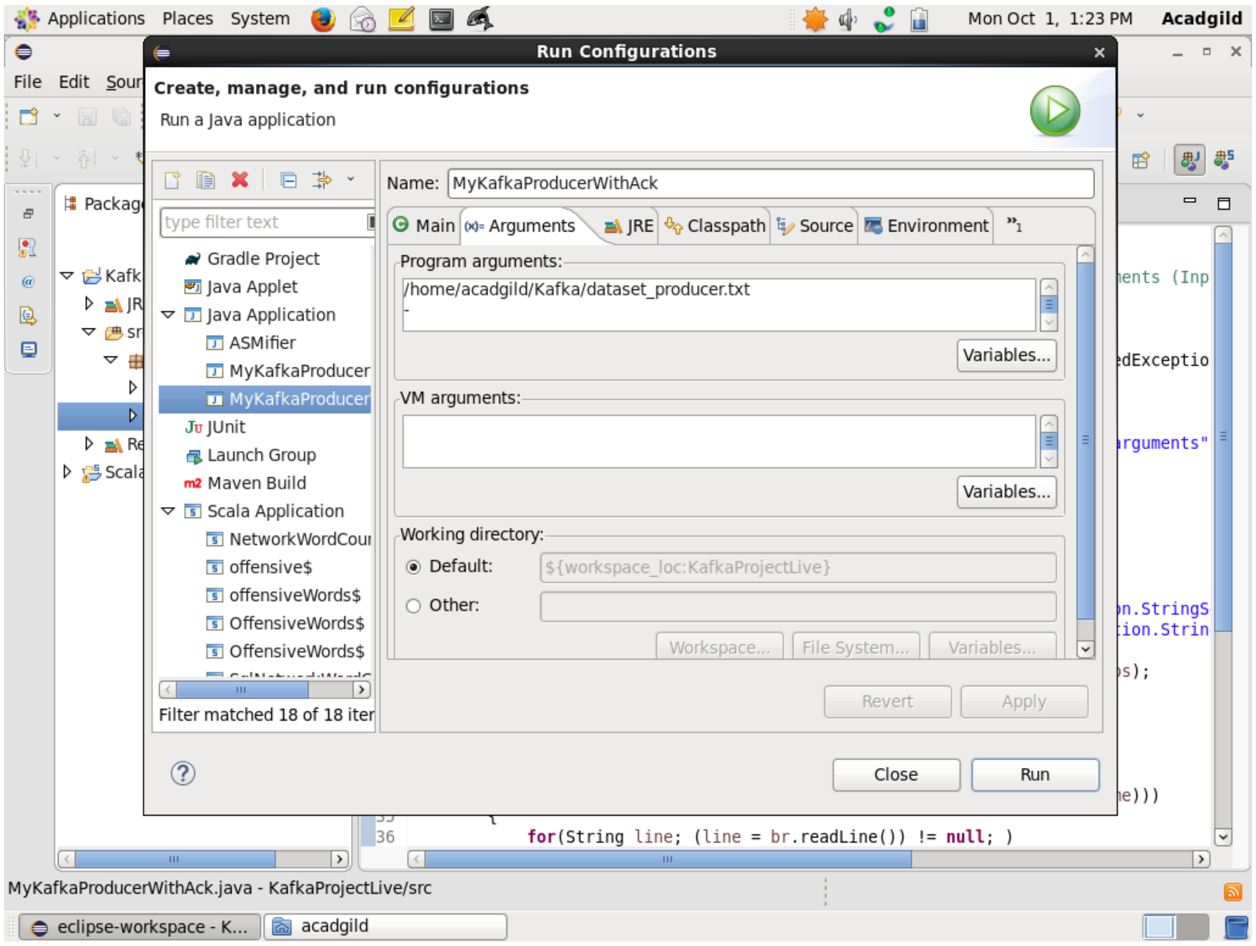
2. Next, we start Zookeeper and Kafka Server using the following commands:

- *$ $KAFKA_HOME/bin/zookeeper-server-start.sh $KAFKA_HOME/config/zookeeper.properties*
- *$ $KAFKA_HOME/bin/kafka-server-start.sh $KAFKA_HOME/config/server.properties*

3. Next, we pass the arguments in “Run Configurations”, and click on “Run” as shown below:



```
producer.KafkaProducer;
fkaProducerWithAck" which takes two arguments (Inp
Ack
ng[] args) throws IOException, InterruptedExceptio
Please provide appropriate command line arguments"
roperties();
vers", "localhost:9092");
;
r", "org.apache.kafka.common.serialization.StringS
zer", "org.apache.kafka.common.serialization.Strin
ring> producer = new KafkaProducer<>(props);
tring> producerRecord = null;
new BufferedReader(new FileReader(fileName)))
ne = br.readLine()) != null; )
```



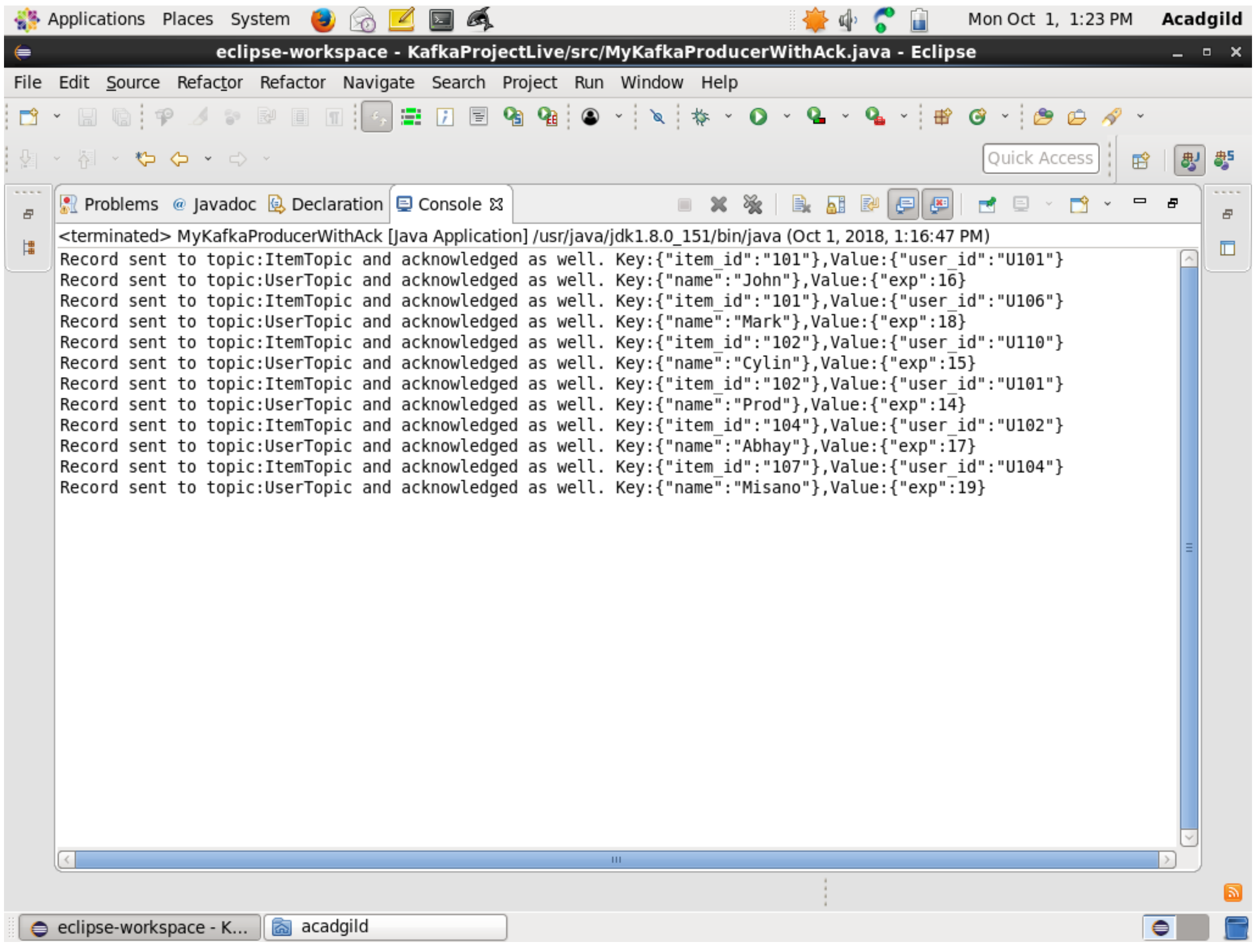
Name: MyKafkaProducerWithAck

Program arguments:
/home/acadgild/Kafka/dataset_producer.txt
-

VM arguments:

Working directory:
Default: ${workspace_loc:KafkaProjectLive}
Other:

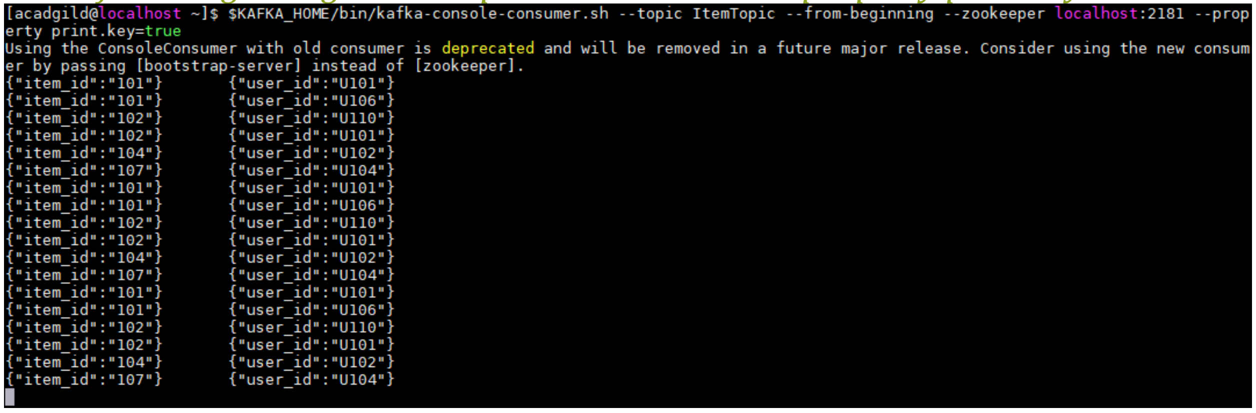
```
for(String line; (line = br.readLine()) != null; )
```


Applications Places System
Mon Oct 1, 1:23 PM Acadgild
eclipse-workspace - KafkaProjectLive/src/MyKafkaProducerWithAck.java - Eclipse
File Edit Source Refactor Refactor Navigate Search Project Run Window Help
Quick Access
Problems Javadoc Declaration Console
eclipse-workspace - K... acadgild

```
<terminated> MyKafkaProducerWithAck [Java Application] /usr/java/jdk1.8.0_151/bin/java (Oct 1, 2018, 1:16:47 PM)
Record sent to topic:ItemTopic and acknowledged as well. Key:{"item_id":"101"},Value:{"user_id":"U101"}
Record sent to topic:UserTopic and acknowledged as well. Key:{"name":"John"},Value:{"exp":16}
Record sent to topic:ItemTopic and acknowledged as well. Key:{"item_id":"101"},Value:{"user_id":"U106"}
Record sent to topic:UserTopic and acknowledged as well. Key:{"name":"Mark"},Value:{"exp":18}
Record sent to topic:ItemTopic and acknowledged as well. Key:{"item_id":"102"},Value:{"user_id":"U110"}
Record sent to topic:UserTopic and acknowledged as well. Key:{"name":"Cylin"},Value:{"exp":15}
Record sent to topic:ItemTopic and acknowledged as well. Key:{"item_id":"102"},Value:{"user_id":"U101"}
Record sent to topic:UserTopic and acknowledged as well. Key:{"name":"Prod"},Value:{"exp":14}
Record sent to topic:ItemTopic and acknowledged as well. Key:{"item_id":"104"},Value:{"user_id":"U102"}
Record sent to topic:UserTopic and acknowledged as well. Key:{"name":"Abhay"},Value:{"exp":17}
Record sent to topic:ItemTopic and acknowledged as well. Key:{"item_id":"107"},Value:{"user_id":"U104"}
Record sent to topic:UserTopic and acknowledged as well. Key:{"name":"Misano"},Value:{"exp":19}
```

4. Next, we run the console consumer commands on terminal to view the output of the program, using the below command:

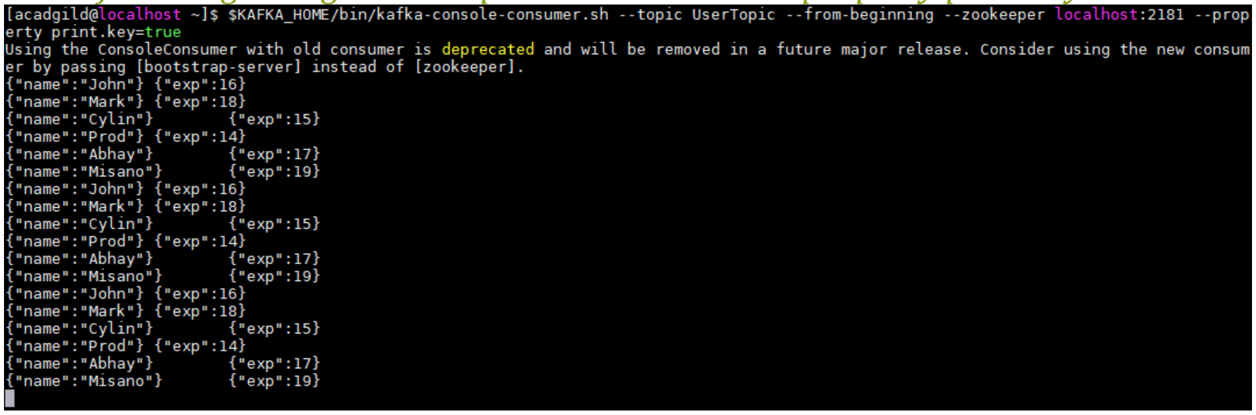
- To read contents of ItemTopic:
  $ $KAFKA_HOME/bin/kafka-console-consumer.sh --topic ItemTopic --from-beginning --zookeeper localhost:2181 --property print.key=true



```
[acadgild@localhost ~]$ $KAFKA_HOME/bin/kafka-console-consumer.sh --topic ItemTopic --from-beginning --zookeeper localhost:2181 --property print.key=true
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-server] instead of [zookeeper].
{"item_id":"101"}       {"user_id":"U101"}
{"item_id":"101"}       {"user_id":"U106"}
{"item_id":"102"}       {"user_id":"U110"}
{"item_id":"102"}       {"user_id":"U101"}
{"item_id":"104"}       {"user_id":"U102"}
{"item_id":"107"}       {"user_id":"U104"}
{"item_id":"101"}       {"user_id":"U101"}
{"item_id":"101"}       {"user_id":"U106"}
{"item_id":"102"}       {"user_id":"U110"}
{"item_id":"102"}       {"user_id":"U101"}
{"item_id":"104"}       {"user_id":"U102"}
{"item_id":"107"}       {"user_id":"U104"}
{"item_id":"101"}       {"user_id":"U101"}
{"item_id":"101"}       {"user_id":"U106"}
{"item_id":"102"}       {"user_id":"U110"}
{"item_id":"102"}       {"user_id":"U101"}
{"item_id":"104"}       {"user_id":"U102"}
{"item_id":"107"}       {"user_id":"U104"}
```

- To read contents of UserTopic:

$ $KAFKA_HOME/bin/kafka-console-consumer.sh --topic UserTopic --from-beginning --zookeeper localhost:2181 --property print.key=true



```
[acadgild@localhost ~]$ $KAFKA_HOME/bin/kafka-console-consumer.sh --topic UserTopic --from-beginning --zookeeper localhost:2181 --prop
erty print.key=true
Using the ConsoleConsumer with old consumer is deprecated and will be removed in a future major release. Consider using the new consum
er by passing [bootstrap-server] instead of [zookeeper].
{"name":"John"} {"exp":16}
{"name":"Mark"} {"exp":18}
{"name":"Cylin"}        {"exp":15}
{"name":"Prod"} {"exp":14}
{"name":"Abhay"}        {"exp":17}
{"name":"Misano"}       {"exp":19}
{"name":"John"} {"exp":16}
{"name":"Mark"} {"exp":18}
{"name":"Cylin"}        {"exp":15}
{"name":"Prod"} {"exp":14}
{"name":"Abhay"}        {"exp":17}
{"name":"Misano"}       {"exp":19}
{"name":"John"} {"exp":16}
{"name":"Mark"} {"exp":18}
{"name":"Cylin"}        {"exp":15}
{"name":"Prod"} {"exp":14}
{"name":"Abhay"}        {"exp":17}
{"name":"Misano"}       {"exp":19}
```