

**(Step1) Steps to Run python file to ingest data from Kafka and clean the data**

# Ingestion of data from Kafka

## Setup in Amazon

### I AM setup

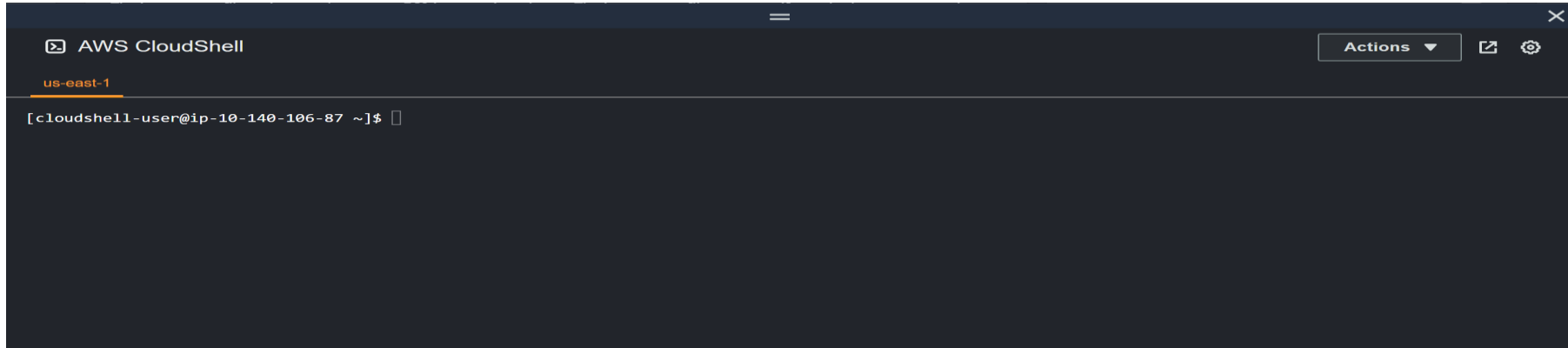
- Login in to the AWS account
- Navigate to I AM dashboard
- Create a user and download the key pair

### EMR Setup

- Navigate to EMR page
- Create a new cluster
- Application bundle for EMR should include the following ( Hue,Spark,Hadoop,Scoop,Hive)
- Cluster config, keep the default options or increase it based on requirement
- Cluster scaling and provisioning keep it as default or change it based on requirement
- Networking keep it as default or change it based on requirement
- Cluster termination set the idle time to automatically terminate the cluster
- Security configuration select the key pair created
- Select the IAM service role if created and make sure this role has permission to perform actions in the EMR so attach the necessary policy for this role such as (EMRFullAccess or Administrator Access) can modify latter based on requirement
- Select the EC2 instance profile for EMR (Will be equal to the user created in I AM setup)
- Click on Create cluster

## Ingestion of data from Kafka

- After creation of the cluster open AWS cloud shell on console



Click on actions dropdown as displayed in the image above and upload the key pair created (eg: dev.pem )

- Also Upload the `spark_kafka_to_local.py` and `spark_local_flatten.py` python files to the cloud shell

**\*Important** replace the below mentioned parameters in all commands as per your configurations

- 1) `i dev.pem`: This option specifies the private key file (eg dev.pem)
- 2) `ec2-user@ec2-54-162-87-251.compute-1.amazonaws.com`: This specifies the user and address of the remote machine
- 3) From Directory (if it's a copy)
- 4) To Directory (if it's a copy)

- Copy the `spark_kafka_to_local.py` and `spark_local_flatten.py` to EMR using the commands below

SCP (Secure Copy Protocol) command used to securely transfer files from your local machine to a remote machine

```
scp -i dev.pem spark_kafka_to_local.py ec2-user@ec2-54-162-87-251.compute-1.amazonaws.com:/home/ec2-user
```

```
scp -i dev.pem spark_kafka_to_local.py ec2-user@ec2-54-162-87-251.compute-1.amazonaws.com:/home/ec2-user
```

- Use the below SSH (Secure Shell) command used to log into EMR cluster

```
ssh -i dev.pem ec2-user@ec2-54-162-87-251.compute-1.amazonaws.com
```

\*Before Running the above commands it is required the key pair file permission is changed with the below mentioned command

```
chmod 400 dev.pem
```

## Ingestion of data from Kafka

- Change permission of EC2 to write to HDFS

```
sudo -u hdfs hdfs dfs -chmod 777 /user
```

Since the ec2 user does not have write permissions in hdfs its required we run the spark jobs as a Hadoop admin so we

- Move the `spark_kafka_to_local.py` & `spark_local_flatten.py` files to hdfs from local

```
hdfs dfs -put spark_kafka_to_local.py /user
```

```
hdfs dfs -put spark_local_flatten.py /user
```

- To run as Hadoop admin use the below command

```
sudo su - hdfs
```

The below command is used in Hadoop environment to copy a file from the Hadoop Distributed File System (HDFS) to the local file system

```
hdfs dfs -get /user/spark_kafka_to_local.py /var/lib/hadoop-hdfs
```

```
hdfs dfs -get /user/spark_local_flatten.py /var/lib/hadoop-hdfs
```

- First run the `spark_kafka_to_local.py` spark job as Hadoop admin to get the streaming data from kafka by using the below mentioned command

```
spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.2.1 spark_kafka_to_local.py
```

- Only after the spark job for the file `spark_kafka_to_local.py` is completed and the csv file from that job is created, we can execute the `spark_local_flatten.py` as this file depends on the data from kafka which is ingested by running `spark_kafka_to_local.py`

- Run the spark job for `spark_local_flatten.py` to proceed with data transformation for the ingested data from kafka by using the below command

```
spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.2.1 spark_local_flatten.py
```

## Validate the data created after Running the spark jobs

- After running the spark jobs we validate the csv files are indeed present for us to use it further in hive table

**\*Important** Path to validate data created might defer based on the hdfs directory defined to write the data

Validate the data ingested from kafka and the transformed data

- Check whether the data from kafka is present in the directory defined during data ingestion

```
hdfs dfs -ls /user/ec2-user/Data/ClickStream
```

- View the csv file for the clickstream data ingested with the below mentioned command

```
hdfs dfs -cat /user/ec2-user/Data/ClickStream/part-00000-e2f64bd8-5e75-4135-b6df-3673dbc84881-c000.csv
```

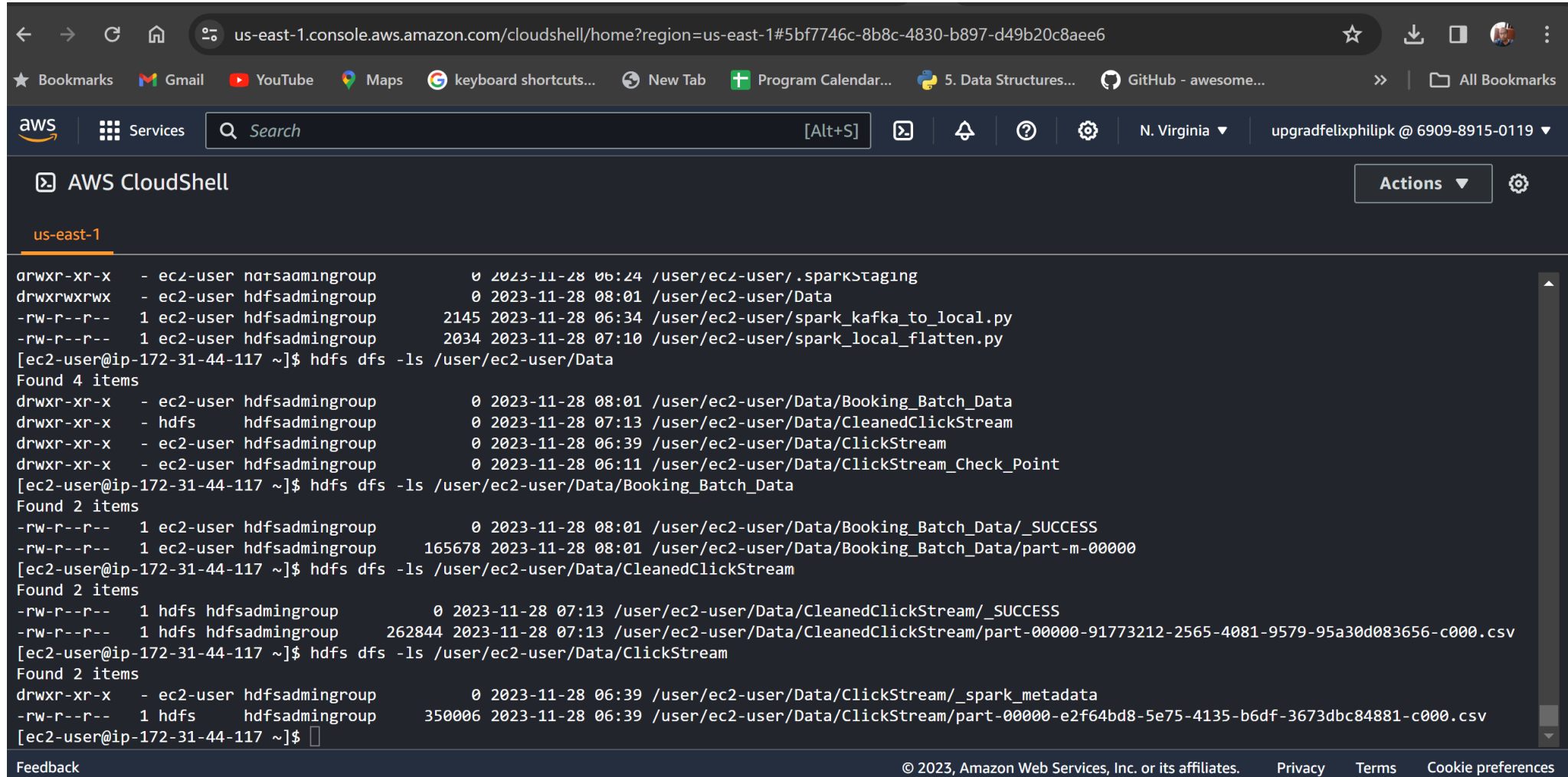
- Navigate to transformed streaming data by executing the below mentioned command

```
hdfs dfs -ls /user/ec2-user/Data/CleanedClickStream
```

- View the csv file for the clickstream transformed data with the below mentioned command

```
hdfs dfs -cat /user/ec2-user/Data/CleanedClickStream/part-00000-84fe7326-3c8d-44aa-aedc-4417b65f355b-c000.csv
```

# Validate data from kafka is present in the directory defined during data ingestion using -ls

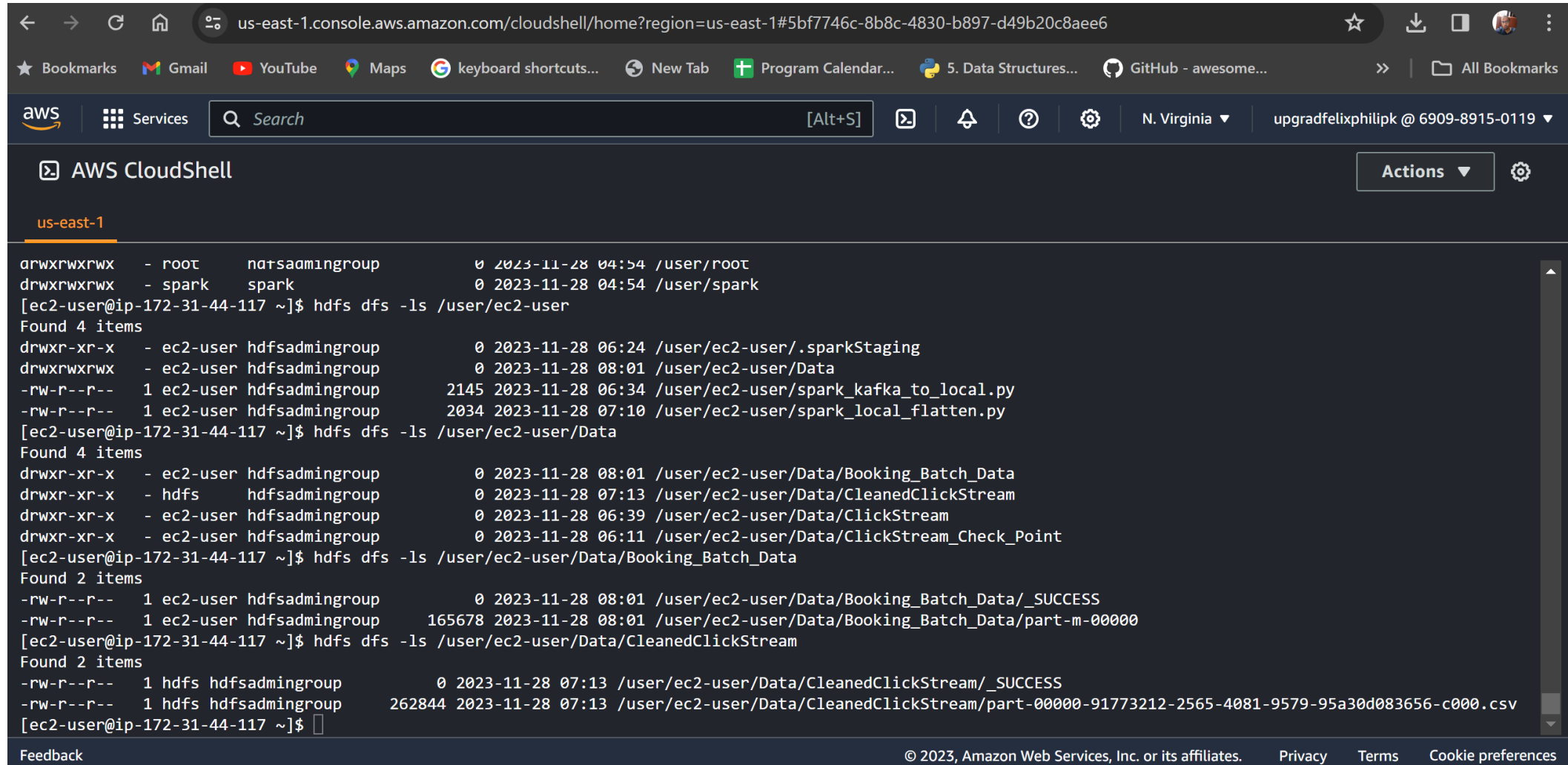


The screenshot shows the AWS CloudShell interface in a web browser. The terminal window displays the following commands and output:

```
us-east-1
AWS CloudShell
us-east-1
drwxr-xr-x - ec2-user natsaamingroup 0 2023-11-28 06:24 /user/ec2-user/.sparkstaging
drwxrwxrwx - ec2-user hdfsadmingroup 0 2023-11-28 08:01 /user/ec2-user/Data
-rw-r--r-- 1 ec2-user hdfsadmingroup 2145 2023-11-28 06:34 /user/ec2-user/spark_kafka_to_local.py
-rw-r--r-- 1 ec2-user hdfsadmingroup 2034 2023-11-28 07:10 /user/ec2-user/spark_local_flatten.py
[ec2-user@ip-172-31-44-117 ~]$ hdfs dfs -ls /user/ec2-user/Data
Found 4 items
drwxr-xr-x - ec2-user hdfsadmingroup 0 2023-11-28 08:01 /user/ec2-user/Data/Booking_Batch_Data
drwxr-xr-x - hdfs hdfsadmingroup 0 2023-11-28 07:13 /user/ec2-user/Data/CleanedClickStream
drwxr-xr-x - ec2-user hdfsadmingroup 0 2023-11-28 06:39 /user/ec2-user/Data/ClickStream
drwxr-xr-x - ec2-user hdfsadmingroup 0 2023-11-28 06:11 /user/ec2-user/Data/ClickStream_Check_Point
[ec2-user@ip-172-31-44-117 ~]$ hdfs dfs -ls /user/ec2-user/Data/Booking_Batch_Data
Found 2 items
-rw-r--r-- 1 ec2-user hdfsadmingroup 0 2023-11-28 08:01 /user/ec2-user/Data/Booking_Batch_Data/_SUCCESS
-rw-r--r-- 1 ec2-user hdfsadmingroup 165678 2023-11-28 08:01 /user/ec2-user/Data/Booking_Batch_Data/part-m-00000
[ec2-user@ip-172-31-44-117 ~]$ hdfs dfs -ls /user/ec2-user/Data/CleanedClickStream
Found 2 items
-rw-r--r-- 1 hdfs hdfsadmingroup 0 2023-11-28 07:13 /user/ec2-user/Data/CleanedClickStream/_SUCCESS
-rw-r--r-- 1 hdfs hdfsadmingroup 262844 2023-11-28 07:13 /user/ec2-user/Data/CleanedClickStream/part-00000-91773212-2565-4081-9579-95a30d083656-c000.csv
[ec2-user@ip-172-31-44-117 ~]$ hdfs dfs -ls /user/ec2-user/Data/ClickStream
Found 2 items
drwxr-xr-x - ec2-user hdfsadmingroup 0 2023-11-28 06:39 /user/ec2-user/Data/ClickStream/_spark_metadata
-rw-r--r-- 1 hdfs hdfsadmingroup 350006 2023-11-28 06:39 /user/ec2-user/Data/ClickStream/part-00000-e2f64bd8-5e75-4135-b6df-3673dbc84881-c000.csv
[ec2-user@ip-172-31-44-117 ~]$
```

The footer of the CloudShell interface includes a Feedback link, a copyright notice for Amazon Web Services, Inc. or its affiliates, and links to Privacy, Terms, and Cookie preferences.

# Validate csv file is present for the cleaned kafka data with -ls



The screenshot shows the AWS CloudShell interface in a web browser. The terminal window displays the following commands and output:

```
us-east-1
us-east-1.console.aws.amazon.com/cloudshell/home?region=us-east-1#5bf7746c-8b8c-4830-b897-d49b20c8aee6

AWS CloudShell
us-east-1

drwxrwxrwx - root natsaamgroup 0 2023-11-28 04:54 /user/root
drwxrwxrwx - spark spark 0 2023-11-28 04:54 /user/spark
[ec2-user@ip-172-31-44-117 ~]$ hdfs dfs -ls /user/ec2-user
Found 4 items
drwxr-xr-x - ec2-user hdfsadmingroup 0 2023-11-28 06:24 /user/ec2-user/.sparkStaging
drwxrwxrwx - ec2-user hdfsadmingroup 0 2023-11-28 08:01 /user/ec2-user/Data
-rw-r--r-- 1 ec2-user hdfsadmingroup 2145 2023-11-28 06:34 /user/ec2-user/spark_kafka_to_local.py
-rw-r--r-- 1 ec2-user hdfsadmingroup 2034 2023-11-28 07:10 /user/ec2-user/spark_local_flatten.py
[ec2-user@ip-172-31-44-117 ~]$ hdfs dfs -ls /user/ec2-user/Data
Found 4 items
drwxr-xr-x - ec2-user hdfsadmingroup 0 2023-11-28 08:01 /user/ec2-user/Data/Booking_Batch_Data
drwxr-xr-x - hdfs hdfsadmingroup 0 2023-11-28 07:13 /user/ec2-user/Data/CleanedClickStream
drwxr-xr-x - ec2-user hdfsadmingroup 0 2023-11-28 06:39 /user/ec2-user/Data/ClickStream
drwxr-xr-x - ec2-user hdfsadmingroup 0 2023-11-28 06:11 /user/ec2-user/Data/ClickStream_Check_Point
[ec2-user@ip-172-31-44-117 ~]$ hdfs dfs -ls /user/ec2-user/Data/Booking_Batch_Data
Found 2 items
-rw-r--r-- 1 ec2-user hdfsadmingroup 0 2023-11-28 08:01 /user/ec2-user/Data/Booking_Batch_Data/_SUCCESS
-rw-r--r-- 1 ec2-user hdfsadmingroup 165678 2023-11-28 08:01 /user/ec2-user/Data/Booking_Batch_Data/part-m-00000
[ec2-user@ip-172-31-44-117 ~]$ hdfs dfs -ls /user/ec2-user/Data/CleanedClickStream
Found 2 items
-rw-r--r-- 1 hdfs hdfsadmingroup 0 2023-11-28 07:13 /user/ec2-user/Data/CleanedClickStream/_SUCCESS
-rw-r--r-- 1 hdfs hdfsadmingroup 262844 2023-11-28 07:13 /user/ec2-user/Data/CleanedClickStream/part-00000-91773212-2565-4081-9579-95a30d083656-c000.csv
[ec2-user@ip-172-31-44-117 ~]$
```

Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences