Load data from Kafka to Hadoop

1. Log into our cluster:

```
C:\Users\User>ssh -i "C:\Users\User\Downloads\emr-key.pem" hadoop@ec2-3-84-55-172.compute-1.amazonaws.com
Last login: Mon Jun 23 21:02:48 2025
```

2. Steps to load the data into Hadoop

Creating directory path in HDFS for storing the processed clickstream data and creating a **checkpoint directory** in HDFS

hdfs dfs -mkdir -p /user/poushali/clickstream
hdfs dfs -mkdir -p /user/poushali/checkpoints/clickstream

Spark – submit command:

```
spark-submit \
  --master yarn \
  --deploy-mode client \
  --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.3.0 \
  spark_kafka_to_local.py
```

Use of spark_kafka_to_local.py

- This PySpark script is designed to read batch data from a Kafka topic and store it in HDFS in JSON format.
- It begins by setting up the required environment variables for Python, Java, and Spark to ensure that the PySpark libraries and the Spark engine are properly accessible. The script then creates a Spark session named "KafkaToHDFSBatch" which is essential to run Spark operations.
- Using this Spark session, the script connects to a Kafka broker located at 18.211.252.152:9092 and subscribes to the topic de-capstone5. It reads all available Kafka messages starting from the earliest to the latest (batch mode, not streaming).
- From the Kafka records, it selects and casts the value field (which holds the actual message content) to a string, preparing it for storage. Finally, the script writes this extracted data to HDFS in JSON format under the directory /user/poushali/clickstream_json. The write mode is set to overwrite, meaning that if the directory already exists, its contents will be replaced.
- The script completes by stopping the Spark session to release resources.

```
EEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE:::::EEEEEEEEE:::E M::::::::M       M::::::::M R:::::RRRRRR:::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR:::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R::R      R::::R
  E:::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R:::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R::::::::::::RR
  E:::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R:::RRRRRR:::::R
  E::::E             M:::::M    M:::M    M:::::M   R::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R::R      R::::R
EE:::::EEEEEEEE::::E M:::::M             M:::::M   R::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR:::R      R::::R
EEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-81-255 ~]$ nano spark_kafka_to_local.py
[hadoop@ip-172-31-81-255 ~]$ spark-submit \
>   --master yarn \
>   --deploy-mode client \
>   --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.3.0 \
>   spark_kafka_to_local.py
:: loading settings :: url = jar:file:/usr/lib/spark/jars/ivy-2.5.1.jar!/org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /home/hadoop/.ivy2/cache
The jars for the packages stored in: /home/hadoop/.ivy2/jars
org.apache.spark#spark-sql-kafka-0-10_2.12 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-d64df4c2-b456-4b24-8671-b01412e53956;1.0
        confs: [default]
        found org.apache.spark#spark-sql-kafka-0-10_2.12;3.3.0 in central
        found org.apache.spark#spark-token-provider-kafka-0-10_2.12;3.3.0 in central
        found org.apache.kafka#kafka-clients;2.8.1 in central
        found org.lz4#lz4-java;1.8.0 in central
        found org.xerial.snappy#snappy-java;1.1.8.4 in central
        found org.slf4j#slf4j-api;1.7.32 in central
```

Our data is now successfully written to HDFS in JSON format

```
[hadoop@ip-10-0-9-85 ~]$ hdfs dfs -ls /user/poushali/clickstream_json
Found 2 items
-rw-r--r--   1 hadoop hdfsadmingroup          0 2025-06-24 18:55 /user/poushali/clickstream_json/_SUCCESS
-rw-r--r--   1 hadoop hdfsadmingroup 2051902653 2025-06-24 18:55 /user/poushali/clickstream_json/part-00000-30bb2b90-7d8e-4c59-b7f9-c3f541227826-c000.json
[hadoop@ip-10-0-9-85 ~]$
```

<u>Reading Json File:</u>

hdfs dfs -cat /user/poushali/clickstream_json/part-00000-68bcef36-7bc5-4ec1-86c2-f38954624e13-c000.json | head -n 5

```
[hadoop@ip-10-0-2-56 ~]$ hdfs dfs -ls /user/poushali/clickstream_json
Found 2 items
-rw-r--r--   1 hadoop hdfsadmingroup          0 2025-06-25 18:00 /user/poushali/clickstream_json/_SUCCESS
-rw-r--r--   1 hadoop hdfsadmingroup 1077675702 2025-06-25 18:00 /user/poushali/clickstream_json/part-00000-68bcef36-7bc5-4ec1-86c2-f38954624e13-c000.json
[hadoop@ip-10-0-2-56 ~]$ ^C
[hadoop@ip-10-0-2-56 ~]$ hdfs dfs -cat /user/poushali/clickstream_json/part-00000-68bcef36-7bc5-4ec1-86c2-f38954624e13-c000.json | head -n 5
{"value_str":"{\"customer_id\": \"63817546\", \"app_version\": \"4.3.5\", \"OS_version\": \"Android\", \"lat\": \"-64.847451\", \"lon\": \"-100.129008\", \"page_id\": \
"de54571l-3914-4450-8c1l-b17b8dabb5e1\", \"button_id\": \"a95dd57b-779f-49db-819d-b6960483e554\", \"is_button_click\": \"No\", \"is_page_view\": \"Yes\", \"is_scroll_up
\": \"No\", \"is_scroll_down\": \"No\", \"timestamp\\n\": \"2020-01-21 16:57:11\\n\"}"}
{"value_str":"{\"customer_id\": \"79407165\", \"app_version\": \"4.3.25\", \"OS_version\": \"Android\", \"lat\": \"85.4985045\", \"lon\": \"152.362461\", \"page_id\": \
"e7bc5fb2-1231-11eb-adc1-0242ac120002\", \"button_id\": \"ele99492-17ae-11eb-adc1-0242ac120002\", \"is_button_click\": \"Yes\", \"is_page_view\": \"Yes\", \"is_scroll_u
p\": \"Yes\", \"is_scroll_down\": \"Yes\", \"timestamp\\n\": \"2020-10-22 22:45:38\\n\"}"}
{"value_str":"{\"customer_id\": \"68513803\", \"app_version\": \"2.4.13\", \"OS_version\": \"iOS\", \"lat\": \"-66.981554\", \"lon\": \"113.405679\", \"page_id\": \"de5
45711-3914-4450-8c11-b17b8dabb5e1\", \"button_id\": \"ele99492-17ae-11eb-adc1-0242ac120002\", \"is_button_click\": \"No\", \"is_page_view\": \"No\", \"is_scroll_up\": \
"No\", \"is_scroll_down\": \"Yes\", \"timestamp\\n\": \"2020-11-01 14:51:33\\n\"}"}
{"value_str":"{\"customer_id\": \"12949619\", \"app_version\": \"1.4.34\", \"OS_version\": \"iOS\", \"lat\": \"-68.7261955\", \"lon\": \"-88.368629\", \"page_id\": \"b3
28829e-17ae-11eb-adc1-0242ac120002\", \"button_id\": \"fcba68aa-1231-11eb-adc1-0242ac120002\", \"is_button_click\": \"No\", \"is_page_view\": \"Yes\", \"is_scroll_up\":
 \"Yes\", \"is_scroll_down\": \"No\", \"timestamp\\n\": \"2020-03-23 20:13:52\\n\"}"}
{"value_str":"{\"customer_id\": \"73600498\", \"app_version\": \"4.4.30\", \"OS_version\": \"Android\", \"lat\": \"-69.4296995\", \"lon\": \"22.086996\", \"page_id\": \
"de54571l-3914-4450-8c1l-b17b8dabb5e1\", \"button_id\": \"a95dd57b-779f-49db-819d-b6960483e554\", \"is_button_click\": \"No\", \"is_page_view\": \"Yes\", \"is_scroll_up
\": \"Yes\", \"is_scroll_down\": \"Yes\", \"timestamp\\n\": \"2020-03-23 23:11:16\\n\"}"}
cat: Unable to write to output stream.
[hadoop@ip-10-0-2-56 ~]$
```

3. **Use of spark_local_flatten.py:**
    - The HDFS-stored Kafka JSON output is processed by this script. It starts by adding the appropriate PySpark and Py4J libraries to the system path and configuring the environment variables for Python, Java, and Spark.
    - After that, a Spark session is started to manage the data processing. In order to extract important fields like customer ID, app version, OS version, position coordinates, interaction flags, and timestamps, the script reads raw JSON files from the designated HDFS directory and uses the get_json_object function to flatten the hierarchical JSON structure.
    - Lastly, the result is consolidated into a single file for simpler analysis by writing the flattened data back to HDFS in CSV format with headers.

```
[hadoop@ip-172-31-81-255 ~]$ nano spark_lolcal_flatten.py
[hadoop@ip-172-31-81-255 ~]$ spark-submit \
>   --master yarn \
>   --deploy-mode client \
>   --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.3.0 \
>   spark_local_flatten.py
:: loading settings :: url = jar:file:/usr/lib/spark/jars/ivy-2.5.1.jar!/org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /home/hadoop/.ivy2/cache
The jars for the packages stored in: /home/hadoop/.ivy2/jars
org.apache.spark#spark-sql-kafka-0-10_2.12 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-6019ef38-b563-4cbb-8178-fe47e0bbac67;1.0
        confs: [default]
        found org.apache.spark#spark-sql-kafka-0-10_2.12;3.3.0 in central
        found org.apache.spark#spark-token-provider-kafka-0-10_2.12;3.3.0 in central
        found org.apache.kafka#kafka-clients;2.8.1 in central
        found org.lz4#lz4-java;1.8.0 in central
        found org.xerial.snappy#snappy-java;1.1.8.4 in central
        found org.slf4j#slf4j-api;1.7.32 in central
        found org.apache.hadoop#hadoop-client-runtime;3.3.2 in central
        found org.spark-project.spark#unused;1.0.0 in central
        found org.apache.hadoop#hadoop-client-api;3.3.2 in central
        found commons-logging#commons-logging;1.1.3 in central
        found com.google.code.findbugs#jsr305;3.0.0 in central
        found org.apache.commons#commons-pool2;2.11.1 in central
:: resolution report :: resolve 491ms :: artifacts dl 16ms
        :: modules in use:
        com.google.code.findbugs#jsr305;3.0.0 from central in [default]
        commons-logging#commons-logging;1.1.3 from central in [default]
        org.apache.commons#commons-pool2;2.11.1 from central in [default]
        org.apache.hadoop#hadoop-client-api;3.3.2 from central in [default]
        org.apache.hadoop#hadoop-client-runtime;3.3.2 from central in [default]
        org.apache.kafka#kafka-clients;2.8.1 from central in [default]
```

The flattened data is then written to a new HDFS location as **CSV format** with headers

```
[hadoop@ip-10-0-9-85 ~]$ hdfs dfs -ls /user/poushali/clickstream_flattened
Found 2 items
-rw-r--r--   1 hadoop hdfsadmingroup          0 2025-06-24 19:11 /user/poushali/clickstream_flattened/_SUCCESS
-rw-r--r--   1 hadoop hdfsadmingroup  643602981 2025-06-24 19:11 /user/poushali/clickstream_flattened/part-00000-20278922-d370-4b6d-a70e-df18a35c701d-c000.csv
[hadoop@ip-10-0-9-85 ~]$
```

hdfs dfs -cat /user/poushali/clickstream_flattened/part-00000-4c986a60-83e0-4cc0-8b89-221c944eac00-c000.csv| head -n 5

```
[hadoop@ip-10-0-9-85 ~]$ hdfs dfs -cat /user/poushali/clickstream_flattened/part-00000-20278922-d370-4b6d-a70e-df18a35c701d-c000.csv | head -n 5
customer_id,app_version,OS_version,lat,lon,page_id,button_id,is_button_click,is_page_view,is_scroll_up,is_scroll_down,timestamp
46118348,3.1.30,iOS,46.3480205,24.456918,de5457ll-3914-4450-8cll-b17b8dabb5el,ele99492-17ae-11eb-adc1-0242ac120002,Yes,Yes,Yes,Yes,
16805048,4.4.10,Android,87.757257,-125.933338,e7bc5fb2-1231-11eb-adc1-0242ac120002,a95dd57b-779f-49db-819d-b6960483e554,No,Yes,Yes,Yes,
71213090,3.3.5,Android,89.3223295,150.138307,b328829e-17ae-11eb-adc1-0242ac120002,fcba68aa-1231-11eb-adc1-0242ac120002,No,No,No,No,
55834829,2.1.32,iOS,40.0699955,103.879483,b328829e-17ae-11eb-adc1-0242ac120002,a95dd57b-779f-49db-819d-b6960483e554,No,Yes,Yes,Yes,
cat: Unable to write to output stream.
[hadoop@ip-10-0-9-85 ~]$
```

4. **Using Hive to create tables for cleaned clickstream data**

```
CREATE EXTERNAL TABLE clickstream_cleaned (
   customer_id STRING,
   app_version STRING,
   OS_version STRING,
   lat DOUBLE,
   lon DOUBLE,
   page_id STRING,
   button_id STRING,
   is_button_click STRING,
   is_page_view STRING,
   is_scroll_up STRING,
   is_scroll_down STRING,
   `timestamp` STRING
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/poushali/clickstream_flattened'
TBLPROPERTIES ("skip.header.line.count"="1");
```

Data stored outside of Hive's internal storage system can be managed and queried using the clickstream_cleaned external table created by the supplied code. The EXTERNAL TABLE keyword guarantees that the actual data files in the specified HDFS location will not be deleted even if the table itself is dropped.

The table schema captures various aspects of user activity, including customer ID, application and operating system versions, geographic coordinates (latitude and longitude), identifiers for pages and buttons, user interaction flags (such as button clicks, page views, and scrolling actions), and a timestamp indicating the event's occurrence time.

The data is stored in CSV format (specified via ROW FORMAT DELIMITED and FIELDS TERMINATED BY ',') in the HDFS directory /user/poushali/clickstream_flattened, as declared in the LOCATION clause.

5. **Checking data in out HDFS**

We used Hive queries to confirm that the cleaned clickstream data was successfully ingested into Hadoop. The data stored in HDFS was queried through the external table clickstream_cleaned, and the output shows that all expected fields have been populated correctly.

```
hive> SHOW TABLES;
OK
clickstream_cleaned
Time taken: 0.402 seconds, Fetched: 1 row(s)
hive> DESCRIBE clickstream_cleaned;
OK
customer_id             string
app_version             string
os_version              string
lat                     double
lon                     double
page_id                 string
button_id               string
is_button_click         string
is_page_view            string
is_scroll_up            string
is_scroll_down          string
timestamp               string
Time taken: 0.087 seconds, Fetched: 12 row(s)
```

```
hive> SELECT * FROM clickstream_cleaned LIMIT 10;
OK
63817546        4.3.5   Android -64.847451       -100.129008     de545711-3914-4450-8c11-b17b8dabb5e1    a95dd57b-779f-49db-819d-b6960483e554    No      Yes     No      N
o
79407165        4.3.25  Android 85.4985045       152.362461      e7bc5fb2-1231-11eb-adc1-0242ac120002    e1e99492-17ae-11eb-adc1-0242ac120002    Yes     Yes     Yes     Y
es
68513803        2.4.13  iOS     -66.981554      113.405679      de545711-3914-4450-8c11-b17b8dabb5e1    e1e99492-17ae-11eb-adc1-0242ac120002    No      No      No      Y
es
12949619        1.4.34  iOS     -68.7261955     -88.368629      b328829e-17ae-11eb-adc1-0242ac120002    fcba68aa-1231-11eb-adc1-0242ac120002    No      Yes     Yes     N
o
73600498        4.4.30  Android -69.4296995      22.086996       de545711-3914-4450-8c11-b17b8dabb5e1    a95dd57b-779f-49db-819d-b6960483e554    No      Yes     Yes     Y
es
20001536        3.3.14  iOS     -32.799864      91.165241       de545711-3914-4450-8c11-b17b8dabb5e1    e1e99492-17ae-11eb-adc1-0242ac120002    No      Yes     No      N
o
45299375        1.4.22  iOS     24.030285       91.460414       e7bc5fb2-1231-11eb-adc1-0242ac120002    a95dd57b-779f-49db-819d-b6960483e554    Yes     Yes     Yes     Y
es
59008919        4.2.33  Android -72.712776       132.619151      e7bc5fb2-1231-11eb-adc1-0242ac120002    fcba68aa-1231-11eb-adc1-0242ac120002    Yes     Yes     Yes     Y
es
61809494        2.3.18  Android 82.4520345       -162.501195     de545711-3914-4450-8c11-b17b8dabb5e1    a95dd57b-779f-49db-819d-b6960483e554    Yes     No      No      N
o
49573886        3.4.4   iOS     31.839155       153.782118      b328829e-17ae-11eb-adc1-0242ac120002    fcba68aa-1231-11eb-adc1-0242ac120002    Yes     No      No      Y
es
Time taken: 2.484 seconds, Fetched: 10 row(s)
hive>
```