TASK 3

Bulk import data from two files in the dataset on your EMR cluster to your HBase Table using the relevant codes.

1) In this task, our objective is to import data from two CSV files, namely "yellow_tripdata_2017-03.csv" and "yellow_tripdata_2017-04.csv," into an HBase table.

2) We will initiate a new HBase table specifically for this task.

Note: Since the datasets do not contain a primary key, we will need to modify these datasets by adding a primary column ID. Given the large size of the dataset, we will split the files and work with them in smaller portions for further processing.

Create new HBase table –

```
hbase(main):001:0> create 'trip_data_batch', 'cf'
0 row(s) in 1.6310 seconds

=> Hbase::Table - trip_data_batch
hbase(main):002:0>
```

Batch Insert command and Execution –

```
[root@ip-172-31-44-105 hadoop]# python file/batch_insert.py
```

HBase Table Records After Import –

```
hbase(main):004:0> scan 'trip_data_batch'
ROW                                     COLUMN+CELL
 1                                      column=cf:DOLocationID, timestamp=1683444168419, value=42
 1                                      column=cf:PULocationID, timestamp=1683444168419, value=231
 1                                      column=cf:RatecodeID, timestamp=1683444168419, value=1
 1                                      column=cf:VendorID, timestamp=1683444168419, value=1
 1                                      column=cf:airport_fee, timestamp=1683444168419, value=
 1                                      column=cf:congestion_surcharge, timestamp=1683444168419, value=
 1                                      column=cf:extra, timestamp=1683444168419, value=0.5
 1                                      column=cf:fare_amount, timestamp=1683444168419, value=30.5
 1                                      column=cf:improvement_surcharge, timestamp=1683444168419, value=0.3
 1                                      column=cf:mta_tax, timestamp=1683444168419, value=0.5
 1                                      column=cf:passenger_count, timestamp=1683444168419, value=1
 1                                      column=cf:payment_type, timestamp=1683444168419, value=1
 1                                      column=cf:store_and_fwd_flag, timestamp=1683444168419, value=N
 1                                      column=cf:tip_amount, timestamp=1683444168419, value=6
 1                                      column=cf:tolls_amount, timestamp=1683444168419, value=0
 1                                      column=cf:total_amount, timestamp=1683444168419, value=37.8
 1                                      column=cf:tpep_dropoff_datetime, timestamp=1683444168419, value=01-03-2017 00:59
 1                                      column=cf:tpep_pickup_datetime, timestamp=1683444168419, value=01-03-2017 00:38
 1                                      column=cf:trip_distance, timestamp=1683444168419, value=10.5
 10                                     column=cf:DOLocationID, timestamp=1683444168469, value=82
 10                                     column=cf:PULocationID, timestamp=1683444168469, value=82
 10                                     column=cf:RatecodeID, timestamp=1683444168469, value=1
 10                                     column=cf:VendorID, timestamp=1683444168469, value=1
```

```
 98                                     column=cf:tpep_pickup_datetime, timestamp=1683444168752, value=01-03-2017 00:08
 98                                     column=cf:trip_distance, timestamp=1683444168752, value=7.2
 99                                     column=cf:DOLocationID, timestamp=1683444168757, value=170
 99                                     column=cf:PULocationID, timestamp=1683444168757, value=186
 99                                     column=cf:RatecodeID, timestamp=1683444168757, value=1
 99                                     column=cf:VendorID, timestamp=1683444168757, value=2
 99                                     column=cf:airport_fee, timestamp=1683444168757, value=
 99                                     column=cf:congestion_surcharge, timestamp=1683444168757, value=
 99                                     column=cf:extra, timestamp=1683444168757, value=0.5
 99                                     column=cf:fare_amount, timestamp=1683444168757, value=5.5
 99                                     column=cf:improvement_surcharge, timestamp=1683444168757, value=0.3
 99                                     column=cf:mta_tax, timestamp=1683444168757, value=0.5
 99                                     column=cf:passenger_count, timestamp=1683444168757, value=1
 99                                     column=cf:payment_type, timestamp=1683444168757, value=2
 99                                     column=cf:store_and_fwd_flag, timestamp=1683444168757, value=N
 99                                     column=cf:tip_amount, timestamp=1683444168757, value=0
 99                                     column=cf:tolls_amount, timestamp=1683444168757, value=0
 99                                     column=cf:total_amount, timestamp=1683444168757, value=6.8
 99                                     column=cf:tpep_dropoff_datetime, timestamp=1683444168757, value=01-03-2017 00:31
 99                                     column=cf:tpep_pickup_datetime, timestamp=1683444168757, value=01-03-2017 00:25
 99                                     column=cf:trip_distance, timestamp=1683444168757, value=0.86
198 row(s) in 1.3820 seconds
```

TASK 3

connecting to HBase server and opening the table
HBase Table Records After Import –
Code –

1) Create HBase Table

```
create 'trip_data_batch', 'cf'
```

2) Ingest Batch data code –

batch_ingest.py

```
import csv
import happybase
import glob
# Define the HBase table and column family
table_name = 'trip_data_batch'
column_family = 'cf'

# connecting to HBase server and opening the table
connection = happybase.Connection(host='ec2-3-80-189-243.compute-1.amazonaws.com')
table = connection.table(table_name)
csv_dir_path = '/home/hadoop/files/file'
csv_files = glob.glob(csv_dir_path + '/*.csv')

# open CSV file and read the data
for file in csv_files:
        with open(file, 'r') as csvfile:
                reader = csv.DictReader(csvfile)
                for row in reader:
# defining row key and column values for each row
                        row_key = row['ID']
                        column_values = {
                                f'{column_family}:VendorID': row['VendorID'],
                                f'{column_family}:tpep_pickup_datetime':
row['tpep_pickup_datetime'],
                                f'{column_family}:tpep_dropoff_datetime':
row['tpep_dropoff_datetime'],
                                f'{column_family}:passenger_count': row['passenger_count'],
                                f'{column_family}:trip_distance': row['trip_distance'],
                                f'{column_family}:RatecodeID': row['RatecodeID'],
                                f'{column_family}:store_and_fwd_flag': row['store_and_fwd_flag'],
                                f'{column_family}:PULocationID': row['PULocationID'],
                                f'{column_family}:DOLocationID': row['DOLocationID'],
                                f'{column_family}:payment_type': row['payment_type'],
                                f'{column_family}:fare_amount': row['fare_amount'],
                                f'{column_family}:extra': row['extra'],
                                f'{column_family}:mta_tax': row['mta_tax'],
```

```
                                f'{column_family}:tip_amount': row['tip_amount'],
                                f'{column_family}:tolls_amount': row['tolls_amount'],
                                f'{column_family}:improvement_surcharge':
row['improvement_surcharge'],
                                f'{column_family}:total_amount': row['total_amount'],
                                f'{column_family}:congestion_surcharge':
row['congestion_surcharge'],
                                f'{column_family}:airport_fee': row['airport_fee']
                                }
# inserting the row into the HBase table
                            table.put(row_key.encode('utf-8'), column_values)
# Close the connection
connection.close()
```