MapReduce Tasks

For solving this assignment, we have to first download dataset into our EMR instance and load it to HDFS. The steps for doing the same are mentioned below:

Step 1: Enter the EMR instance through the terminal (for Mac users) or putty (for Windows users) and enter the “hadoop” user login.

```
[hadoop@ip-172-31-19-104 ~]$ pwd
/home/hadoop
[hadoop@ip-172-31-19-104 ~]$
```

Step 2: Create the directory structure /home/hadoop/mapr_assignment/input_dataset and enter the "input_dataset" directory using the following sequence of commands:

```
[hadoop@ip-172-31-19-104 ~]$ mkdir mapr_assignment
[hadoop@ip-172-31-19-104 ~]$ cd mapr_assignment/
[hadoop@ip-172-31-19-104 mapr_assignment]$ mkdir input_dataset
[hadoop@ip-172-31-19-104 mapr_assignment]$ cd input_dataset/
[hadoop@ip-172-31-19-104 input_dataset]$
```

Step 3: For downloading the dataset we execute the following commands:

wget https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-01.csv

wget https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-02.csv

wget https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-03.csv

wget https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-04.csv

wget https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-05.csv

wget https://nyc-tlc-upgrad.s3.amazonaws.com/yellow_tripdata_2017-06.csv

Step 4: After executing all the commands we can execute ls -lrt and check if all the six files are downloaded or not.

Step 5: Create a directory "/user/hadoop/mapr_assignment/input" in HDFS. The command to create the directory in HDFS is:

```
hadoop fs -mkdir -p /user/hadoop/mapr_assignment/input
```

Step 6: Using the 'put' command add all the input data into the HDFS location created in the previous step.

```
hadoop fs -put yellow_tripdata_* /user/hadoop/mapr_assignment/input
```

Step 7: Execute the following command and check if the datasets are present in the HDFS location /user/hadoop/mapr_assignment/input/ or not:

```
[hadoop@ip-172-31-19-104 input_dataset]$ hadoop fs -ls /user/hadoop/mapr_assignment/input/
Found 6 items
-rw-r--r--   1 hadoop hadoop  914029540 2025-01-08 04:36 /user/hadoop/mapr_assignment/input/yellow_tripdata_2017-01.csv
-rw-r--r--   1 hadoop hadoop  863487050 2025-01-08 04:36 /user/hadoop/mapr_assignment/input/yellow_tripdata_2017-02.csv
-rw-r--r--   1 hadoop hadoop  969809025 2025-01-08 04:36 /user/hadoop/mapr_assignment/input/yellow_tripdata_2017-03.csv
-rw-r--r--   1 hadoop hadoop  946349441 2025-01-08 04:36 /user/hadoop/mapr_assignment/input/yellow_tripdata_2017-04.csv
-rw-r--r--   1 hadoop hadoop  951965526 2025-01-08 04:37 /user/hadoop/mapr_assignment/input/yellow_tripdata_2017-05.csv
-rw-r--r--   1 hadoop hadoop  910028408 2025-01-08 04:37 /user/hadoop/mapr_assignment/input/yellow_tripdata_2017-06.csv
[hadoop@ip-172-31-19-104 input_dataset]$
```

A: Which vendors have the most trips, and what is the total revenue generated by that vendor?

Code:

```
[hadoop@ip-172-31-16-32 input_dataset]$ cat mrtask_a.py
from mrjob.job import MRJob
from mrjob.step import MRStep


class VendorTripsRevenue(MRJob):

    def steps(self):
        return [
            MRStep(
                mapper=self.mapper_get_trips_and_revenue,
                reducer=self.reducer_aggregate_trips_and_revenue
            ),
            MRStep(
                reducer=self.reducer_find_top_vendor
            )
        ]

    def mapper_get_trips_and_revenue(self, _, line):
        """Mapper: Extract VendorID and compute revenue."""
        try:
            # Parse the line into a list of fields
            fields = line.split(',')
            vendor_id = fields[0]
            fare_amount = float(fields[10])
            tip_amount = float(fields[13])

            # Calculate revenue
            revenue = fare_amount + tip_amount

            # Emit vendor_id, trips (1), and revenue
            yield vendor_id, (1, revenue)

        except (IndexError, ValueError):
            # Skip rows with invalid data
            pass

    def reducer_aggregate_trips_and_revenue(self, vendor_id, values):
        """Reducer: Aggregate trips and revenue for each VendorID."""
        total_trips = 0
        total_revenue = 0.0

        for trips, revenue in values:
            total_trips += trips
```

```
        for vendor_id, trips, revenue in vendor_data:
            if trips > max_trips:
                top_vendor = vendor_id
                max_trips = trips
                total_revenue = revenue

        # Emit the top vendor, their total trips, and total revenue
        yield top_vendor, (max_trips, total_revenue)


if __name__ == '__main__':
    VendorTripsRevenue.run()

[hadoop@ip-172-31-16-32 input_dataset]$
```

Following is the reference command to deploy it:

python mrtask_a.py -r hadoop hdfs:///user/hadoop/mapr_assignment/input/ \

--output-dir /user/hadoop/mapr_assignment/output/q1_all_files

To verify the job execution, we verify the output in the specified directory:

```
[hadoop@ip-172-31-16-32 input_dataset]$ hdfs dfs -ls  /user/hadoop/mapr_assignment/output/q1_all_files
Found 4 items
-rw-r--r--   1 hadoop hadoop          0 2025-01-08 10:15 /user/hadoop/mapr_assignment/output/q1_all_files/_SUCCESS
-rw-r--r--   1 hadoop hadoop         34 2025-01-08 10:15 /user/hadoop/mapr_assignment/output/q1_all_files/part-00000
-rw-r--r--   1 hadoop hadoop          0 2025-01-08 10:15 /user/hadoop/mapr_assignment/output/q1_all_files/part-00001
-rw-r--r--   1 hadoop hadoop          0 2025-01-08 10:15 /user/hadoop/mapr_assignment/output/q1_all_files/part-00002
[hadoop@ip-172-31-16-32 input_dataset]$
```

The result is:

```
[hadoop@ip-172-31-16-32 input_dataset]$ hdfs dfs -cat  /user/hadoop/mapr_assignment/output/q1_all_files/part-*
"2"     [32158202, 478032245.0149723]
[hadoop@ip-172-31-16-32 input_dataset]$
```

Hence, vendor 2 has most trips (32,158,202) and total revenue generated by the vendor is 478,032,245.01

B: Which pickup location generates the most revenue?

Code:

vi mrtask_b.py

hadoop@ip-172-31-16-32:~/mapr_assignment/input_dataset

```
from mrjob.job import MRJob
import csv

class MRRevenueByPickupLocation(MRJob):

    def mapper(self, _, line):
        """Mapper function to emit pickup location and total revenue."""
        try:
            row = dict(zip(
                ['VendorID', 'tpep_pickup_datetime', 'tpep_dropoff_datetime', 'passenger_count',
                 'trip_distance', 'RatecodeID', 'store_and_fwd_flag', 'PULocationID', 'DOLocationID',
                 'payment_type', 'fare_amount', 'extra', 'mta_tax', 'tip_amount', 'tolls_amount',
                 'improvement_surcharge', 'total_amount', 'congestion_surcharge', 'airport_fee'],
                next(csv.reader([line]))
            ))

            # Check if the row is valid
            if self.is_valid_row(row):
                # Emit pickup location and total revenue
                yield row['PULocationID'], float(row['total_amount'])
        except Exception as e:
            pass

    def reducer(self, location, revenues):
        """Reducer function to sum the total revenue for each pickup location."""
        total_revenue = sum(revenues)
        yield location, total_revenue

    def is_valid_row(self, row):
        """Validate if the row has required fields."""
        try:
            float(row['total_amount'])  # Ensure total_amount is a valid float
            int(row['PULocationID'])    # Ensure PULocationID is a valid integer
            return True
        except:
            return False

if __name__ == '__main__':
    MRRevenueByPickupLocation.run()
```

Following is the reference command to deploy it:

python mrtask_b.py -r hadoop hdfs:///user/hadoop/mapr_assignment/input/ \

--output-dir /user/hadoop/mapr_assignment/output/q2_all_files

To verify the job execution, we verify the output in the specified directory:

```
[hadoop@ip-172-31-16-32 input_dataset]$ hdfs dfs -ls  /user/hadoop/mapr_assignment/output/q2_all_files
Found 4 items
-rw-r--r--   1 hadoop hadoop          0 2025-01-08 10:36 /user/hadoop/mapr_assignment/output/q2_all_files/_SUCCESS
-rw-r--r--   1 hadoop hadoop       2080 2025-01-08 10:33 /user/hadoop/mapr_assignment/output/q2_all_files/part-00000
-rw-r--r--   1 hadoop hadoop       2101 2025-01-08 10:35 /user/hadoop/mapr_assignment/output/q2_all_files/part-00001
-rw-r--r--   1 hadoop hadoop       2074 2025-01-08 10:36 /user/hadoop/mapr_assignment/output/q2_all_files/part-00002
[hadoop@ip-172-31-16-32 input_dataset]$
```

The result is:

```
[hadoop@ip-172-31-16-32 input_dataset]$ hdfs dfs -cat  /user/hadoop/mapr_assignment/output/q2_all_files/part-* | sort -k2 -nr | head -5
"132"   77196812.23975265
"138"   64480311.15005931
"161"   32910784.810575873
"230"   31638136.22054061
"186"   29804472.580555458
[hadoop@ip-172-31-16-32 input_dataset]$
```

PULocationID 132 is the highest revenue-generating pickup location with over 77 million in revenue.

C: What are the different payment types used by customers and their count? The final results should be in a sorted format.

Code:

```
[hadoop@ip-172-31-16-32 input_dataset]$ cat mrtask_c.py
from mrjob.job import MRJob
from mrjob.step import MRStep


class PaymentTypeCount(MRJob):

    def steps(self):
        return [
            MRStep(
                mapper=self.mapper_get_payment_types,
                reducer=self.reducer_count_payment_types
            ),
            MRStep(
                reducer=self.reducer_sort_payment_types
            )
        ]

    def mapper_get_payment_types(self, _, line):
        """Mapper: Extract Payment Type from each line."""
        try:
            # Split the line into fields
            fields = line.split(',')
            # Skip the header row by checking if it contains non-numeric payment_type
            if fields[9].isdigit():
                payment_type = fields[9]

                # Emit payment_type with a count of 1
                yield payment_type, 1

        except (IndexError, ValueError):
            # Skip lines with invalid data
            pass

    def reducer_count_payment_types(self, payment_type, counts):
        """Reducer: Aggregate counts for each payment type."""
        # Sum up all counts for the payment type
        total_count = sum(counts)
        # Emit payment_type and its total count
        yield None, (payment_type, total_count)

    def reducer_sort_payment_types(self, _, payment_data):
        """Reducer: Sort payment types by count."""
        # Sort by count in descending order
        sorted_payment_data = sorted(payment_data, key=lambda x: x[1], reverse=True)
```

```
        # Sort by count in descending order
        sorted_payment_data = sorted(payment_data, key=lambda x: x[1], reverse=True)

        # Emit sorted payment types and their counts
        for payment_type, count in sorted_payment_data:
            yield payment_type, count


if __name__ == '__main__':
    PaymentTypeCount.run()
```

Following is the reference command to deploy it:

python mrtask_c.py -r hadoop hdfs:///user/hadoop/mapr_assignment/input/ \

--output-dir /user/hadoop/mapr_assignment/output/q3_all

To verify the job execution, we verify the output in the specified directory:

```
[hadoop@ip-172-31-16-32 input_dataset]$ hdfs dfs -ls  /user/hadoop/mapr_assignment/output/q3_all
Found 4 items
-rw-r--r--   1 hadoop hadoop         0 2025-01-08 11:17 /user/hadoop/mapr_assignment/output/q3_all/_SUCCESS
-rw-r--r--   1 hadoop hadoop        53 2025-01-08 11:16 /user/hadoop/mapr_assignment/output/q3_all/part-00000
-rw-r--r--   1 hadoop hadoop         0 2025-01-08 11:17 /user/hadoop/mapr_assignment/output/q3_all/part-00001
-rw-r--r--   1 hadoop hadoop         0 2025-01-08 11:17 /user/hadoop/mapr_assignment/output/q3_all/part-00002
```

The result is:

```
[hadoop@ip-172-31-16-32 input_dataset]$ hdfs dfs -cat  /user/hadoop/mapr_assignment/output/q3_all/part-*
"1"     39754212
"2"     18832370
"3"     306912
"4"     88794
"5"     3
[hadoop@ip-172-31-16-32 input_dataset]$
```

The payment_type 1 which is credit card is used by maximum customers.

D: What is the average trip time for different pickup locations?

Code:

```
[hadoop@ip-172-31-28-99 input_dataset]$ cat mrtask_d.py
from mrjob.job import MRJob
from mrjob.step import MRStep
from datetime import datetime


class AverageTripTimeByPickup(MRJob):

    def steps(self):
        return [
            MRStep(
                mapper=self.mapper_get_trip_times,
                reducer=self.reducer_sum_trip_times
            ),
            MRStep(
                reducer=self.reducer_calculate_average
            )
        ]

    def mapper_get_trip_times(self, _, line):
        """Mapper: Extract pickup location ID and calculate trip duration."""
        try:
            # Split the line into fields
            fields = line.split(',')

            # Skip header row by checking if pickup_location_id is numeric
            if fields[7].isdigit():
                pickup_location_id = fields[7]
                pickup_time = fields[1]
                dropoff_time = fields[2]

                # Parse timestamps to calculate trip time in minutes
                pickup_dt = datetime.strptime(pickup_time, '%Y-%m-%d %H:%M:%S')
                dropoff_dt = datetime.strptime(dropoff_time, '%Y-%m-%d %H:%M:%S')
                trip_duration = (dropoff_dt - pickup_dt).total_seconds() / 60  # Convert to minutes

                # Emit pickup location ID and trip duration
                yield pickup_location_id, trip_duration

        except (IndexError, ValueError):
            # Skip lines with invalid data
            pass

    def reducer_sum_trip_times(self, pickup_location_id, durations):
        """Reducer: Calculate total trip time and count for each pickup location."""
```

```
            pass

    def reducer_sum_trip_times(self, pickup_location_id, durations):
        """Reducer: Calculate total trip time and count for each pickup location."""
        total_time = 0
        trip_count = 0

        for duration in durations:
            total_time += duration
            trip_count += 1

        # Emit pickup location ID, total time, and trip count
        yield pickup_location_id, (total_time, trip_count)

    def reducer_calculate_average(self, pickup_location_id, values):
        """Final Reducer: Calculate average trip time."""
        total_time = 0
        trip_count = 0

        for value in values:
            total_time += value[0]
            trip_count += value[1]

        # Calculate average trip time
        if trip_count > 0:
            average_time = total_time / trip_count
            yield pickup_location_id, average_time


if __name__ == '__main__':
    AverageTripTimeByPickup.run()

[hadoop@ip-172-31-28-99 input_dataset]$
```

Following is the reference command to deploy it:

python mrtask_d.py -r hadoop hdfs:///user/hadoop/mapr_assignment/input/ \

--output-dir /user/hadoop/mapr_assignment/output/q4_all_files

To verify the job execution, we verify the output in the specified directory:

```
[hadoop@ip-172-31-28-99 input_dataset]$ hdfs dfs -ls  /user/hadoop/mapr_assignment/output/q4_all_files
Found 4 items
-rw-r--r--   1 hadoop hadoop          0 2025-01-11 04:17 /user/hadoop/mapr_assignment/output/q4_all_files/_SUCCESS
-rw-r--r--   1 hadoop hadoop       2133 2025-01-11 04:17 /user/hadoop/mapr_assignment/output/q4_all_files/part-00000
-rw-r--r--   1 hadoop hadoop       2132 2025-01-11 04:17 /user/hadoop/mapr_assignment/output/q4_all_files/part-00001
-rw-r--r--   1 hadoop hadoop       2132 2025-01-11 04:17 /user/hadoop/mapr_assignment/output/q4_all_files/part-00002
[hadoop@ip-172-31-28-99 input_dataset]$
```

The result is:

hadoop@ip-172-31-28-99:~/mapr_assignment/input_dataset

```
[hadoop@ip-172-31-28-99 input_dataset]$ hdfs dfs -cat  /user/hadoop/mapr_assignment/output/q4_all_files/part-*
"102"   22.112369791666666
"105"   19.977777777777778
"108"   14.208414239482197
"111"   11.634657836644594
"114"   15.92129646297791
"117"   19.31763565891473
"12"    24.35871344023625
"120"   13.808120868744103
"123"   15.514665127020757
"126"   18.56177873152122
"129"   14.25869664200331
"132"   43.77028871964125
"135"   18.103793774319097
"138"   37.32248703573864
"141"   12.262026243285721
"144"   16.803778739970095
"147"   13.154071246819335
"15"    14.541896024464833
"150"   18.49230769230768
"153"   13.562989801395604
"156"   19.404368932038835
"159"   14.20484398216938
"162"   15.103685552024887
"165"   18.458268590455052
"168"   12.855600663523674
"171"   12.873132183908044
"174"   13.067625899280577
"177"   19.16066056245915
"18"    14.32647482014389
"180"   30.037775246772938
"183"   12.080208333333337
"186"   16.811887227994635
"189"   15.166321270502547
"192"   18.24926590538336
"195"   20.830252339098372
"198"   13.249704098239397
"201"   9.88239436619718
"204"   3.551190476190476
"207"   8.466067108446378
"21"    20.49817708333332
"210"   17.948028169014083
"213"   16.387551867219926
"216"   28.41855829576593
"219"   45.4526845258279
"222"   29.712301587301596
"225"   14.926513473606509
"228"   15.833887297039206
```

The output lists average trip time for different pickup locations.

E: Calculate the average tips to revenue ratio of the drivers for different pickup locations in sorted format.

Code:

vi mrtask_e.py

hadoop@ip-172-31-28-99:~/mapr_assignment/input_dataset

```
from mrjob.job import MRJob
from mrjob.step import MRStep

class TipsToRevenueRatio(MRJob):

    def steps(self):
        return [
            MRStep(mapper=self.mapper_get_ratios,
                   reducer=self.reducer_calculate_average),
            MRStep(reducer=self.reducer_sort_by_ratio)
        ]

    def mapper_get_ratios(self, _, line):
        # Skip the header
        if line.startswith("VendorID"):
            return
        fields = line.split(",")
        try:
            PULocationID = fields[7]  # Pickup location ID
            tip_amount = float(fields[13])
            total_amount = float(fields[16])

            # Avoid division by zero
            if total_amount > 0:
                yield PULocationID, (tip_amount / total_amount)
        except (ValueError, IndexError):
            pass

    def reducer_calculate_average(self, PULocationID, ratios):
        # Calculate the average tips-to-revenue ratio for each location
        ratios_list = list(ratios)
        avg_ratio = sum(ratios_list) / len(ratios_list)
        yield None, (avg_ratio, PULocationID)

    def reducer_sort_by_ratio(self, _, location_ratios):
        # Sort by ratio in descending order
        for avg_ratio, PULocationID in sorted(location_ratios, reverse=True):
            yield PULocationID, avg_ratio

if __name__ == "__main__":
    TipsToRevenueRatio.run()
```

Following is the reference command to deploy it:

python mrtask_e.py -r hadoop hdfs:///user/hadoop/mapr_assignment/input/ \

--output-dir /user/hadoop/mapr_assignment/output/q5_all

To verify the job execution, we verify the output in the specified directory:

```
[hadoop@ip-172-31-27-36 input_dataset]$ hdfs dfs -ls  /user/hadoop/mapr_assignment/output/q5_all
Found 4 items
-rw-r--r--   1 hadoop hadoop          0 2025-01-12 04:14 /user/hadoop/mapr_assignment/output/q5_all/_SUCCESS
-rw-r--r--   1 hadoop hadoop       6734 2025-01-12 04:14 /user/hadoop/mapr_assignment/output/q5_all/part-00000
-rw-r--r--   1 hadoop hadoop          0 2025-01-12 04:14 /user/hadoop/mapr_assignment/output/q5_all/part-00001
-rw-r--r--   1 hadoop hadoop          0 2025-01-12 04:14 /user/hadoop/mapr_assignment/output/q5_all/part-00002
```

The result is:

hadoop@ip-172-31-27-36:~/mapr_assignment/input_dataset

```
[hadoop@ip-172-31-27-36 input_dataset]$ hdfs dfs -cat  /user/hadoop/mapr_assignment/output/q5_all/part-*

"104"   0.2000665778961385
"30"    0.15834022865441705
"176"   0.13873652132712305
"172"   0.12636039409236893
"109"   0.12527506733080965
"138"   0.12367633428689961
"199"   0.1216192061849817
"13"    0.11720991870020904
"125"   0.1163443845653772
"87"    0.11581969453118245
"52"    0.11511551760555802
"234"   0.11430589214213145
"249"   0.11302496141033676
"117"   0.11297366786253282
"107"   0.11221707026481366
"231"   0.11199090023306242
"158"   0.11180661389605968
"246"   0.11146601334824383
"90"    0.11133141659728978
"114"   0.11069018493840198
"162"   0.11068168738171777
"113"   0.11068166988794204
"79"    0.11059416421849462
"170"   0.1099113189579127
"33"    0.10892803655847132
"66"    0.10865769171725821
"194"   0.10843759375638361
"255"   0.10830625418346451
"148"   0.10764788279336289
"211"   0.10739991766232336
"137"   0.10720968167419584
"233"   0.10700916117170012
"144"   0.10692203763100984
"54"    0.10687510207502059
"68"    0.1062404501305579
"40"    0.10618875562165203
"224"   0.10607831728927934
"239"   0.10605284458463737
"161"   0.10578506990012851
"143"   0.10555707232397082
"88"    0.1054631535409606
"209"   0.10449487178971409
"142"   0.10439274256314794
"263"   0.10425889887618153
"1"     0.10421893358472592
"264"   0.10396980029455873
```

The output lists the average tips-to-revenue ratio for different pickup locations, sorted in descending order. Each line represents a unique pickup location ID and its corresponding ratio.

F: How does revenue vary over time? Calculate the average trip revenue per month - analyzing it by hour of the day (day vs night) and the day of the week (weekday vs weekend).

We classify the hour into "day" (6 AM to 6 PM) and "night" (6 PM to 6 AM) and the day into "weekday" (Mon-Fri) and "weekend" (Sat-Sun).

Code:

hadoop@ip-172-31-27-36:~/mapr_assignment/input_dataset

```
[hadoop@ip-172-31-27-36 input_dataset]$ cat mrtask_f.py
from mrjob.job import MRJob
from mrjob.step import MRStep
from datetime import datetime

class RevenueAnalysis(MRJob):

    def mapper(self, _, line):
        # Skip header line if present
        if "tpep_pickup_datetime" in line:
            return

        # Split the line into columns
        fields = line.split(',')

        try:
            # Extract the necessary fields
            pickup_datetime = fields[1]  # tpep_pickup_datetime
            fare_amount = float(fields[7])  # fare_amount

            # Convert pickup_datetime to datetime object
            pickup_datetime = datetime.strptime(pickup_datetime, '%Y-%m-%d %H:%M:%S')

            # Extract month, hour, and day of the week
            month = pickup_datetime.month
            hour = pickup_datetime.hour
            weekday = pickup_datetime.weekday()  # Monday is 0, Sunday is 6

            # Classify hour into 'day' or 'night'
            time_period = 'day' if 6 <= hour < 18 else 'night'

            # Classify day into 'weekday' or 'weekend'
            day_type = 'weekend' if weekday in [5, 6] else 'weekday'

            # Emit key as (month, time_period, day_type) and value as the fare amount
            yield (month, time_period, day_type), fare_amount

        except Exception as e:
            # Skip any invalid records
            pass

    def reducer(self, key, values):
        # Calculate the average fare amount for each key (month, time_period, day_type)
        total_fare = 0
        count = 0
        for fare in values:
            total_fare += fare
            count += 1
```

```
    def reducer(self, key, values):
        # Calculate the average fare amount for each key (month, time_period, day_type)
        total_fare = 0
        count = 0
        for fare in values:
            total_fare += fare
            count += 1

        # Emit the key (month, time_period, day_type) and the average revenue
        if count > 0:
            avg_fare = total_fare / count
            yield key, avg_fare

if __name__ == '__main__':
    RevenueAnalysis.run()

[hadoop@ip-172-31-27-36 input_dataset]$
```

Following is the reference command to deploy it:

python mrtask_f.py -r hadoop hdfs:///user/hadoop/mapr_assignment/input/ \

--output-dir /user/hadoop/mapr_assignment/output/q6_all

To verify the job execution, we verify the output in the specified directory:

```
[hadoop@ip-172-31-27-36 input_dataset]$ hdfs dfs -ls  /user/hadoop/mapr_assignment/output/q6_all
Found 4 items
-rw-r--r--   1 hadoop hadoop          0 2025-01-12 04:53 /user/hadoop/mapr_assignment/output/q6_all/_SUCCESS
-rw-r--r--   1 hadoop hadoop        330 2025-01-12 04:50 /user/hadoop/mapr_assignment/output/q6_all/part-00000
-rw-r--r--   1 hadoop hadoop        291 2025-01-12 04:52 /user/hadoop/mapr_assignment/output/q6_all/part-00001
-rw-r--r--   1 hadoop hadoop        376 2025-01-12 04:53 /user/hadoop/mapr_assignment/output/q6_all/part-00002
```

The result is:

```
[hadoop@ip-172-31-27-36 input_dataset]$ hdfs dfs -cat  /user/hadoop/mapr_assignment/output/q6_all/part-*
[1, "night", "weekday"] 162.66949766901578
[2, "day", "weekday"]   167.07750739355671
[2, "night", "weekend"] 155.839403482894
[3, "day", "weekend"]   164.68551861356397
[4, "night", "weekday"] 161.28524256986532
[5, "day", "weekday"]   166.48943855807
[5, "night", "weekend"] 155.96165614108895
[6, "day", "weekday"]   166.2721714834558
[1, "day", "weekend"]   165.238292430468
[2, "night", "weekday"] 161.95570717608274
[3, "day", "weekday"]   165.86213973407382
[3, "night", "weekend"] 156.15588418464506
[4, "day", "weekend"]   163.1632534538379
[5, "night", "weekday"] 161.25816450580552
[6, "night", "weekend"] 154.5754968059548
[1, "day", "weekday"]   167.44587441207224
[1, "night", "weekend"] 156.34597756349498
[2, "day", "weekend"]   164.34590449470863
[3, "night", "weekday"] 161.13070093683532
[4, "day", "weekday"]   166.30639130883034
[4, "night", "weekend"] 155.75345690230233
[5, "day", "weekend"]   163.0289596623916
[6, "day", "weekend"]   162.26646468705528
[6, "night", "weekday"] 161.05295011180957
[hadoop@ip-172-31-27-36 input_dataset]$
```

The output is the average trip revenue for each month, segmented by time period (day vs night) and day type (weekday vs weekend).