Assignment part - 1

January 10, 2022

[173]:
```
#Import all the required lib
import pyspark
from pyspark.sql import SparkSession
import pyspark.sql.functions as f
from pyspark.sql.types import StructType,StructField, StringType,
 IntegerType,␣↪DoubleType, DateType
from pyspark.sql.functions import concat_ws, split, lit,
 to_timestamp,␣↪unix_timestamp, acos, cos, sin, lit, toRadians, lag
from pyspark.sql import Window
```

[4]:
```
#Creating a spark session as this will be the entry part for the program
spark = SparkSession.builder.appName("Assignment 1").getOrCreate()
```

[79]:
```
#Here we create the schema for the dataframe which
we'll read schema = StructType([
    StructField("UserId",IntegerType(),True), \
    StructField("Latitude",DoubleType(),True), \
    StructField("Longitude",DoubleType(),True), \
    StructField("AllZero",IntegerType(),True), \
    StructField("Altitude",DoubleType(),True), \
    StructField("Timestamp",StringType(),True), \
    StructField("Date",StringType(),True), \
    StructField("Time",StringType(),True)
  ])
```

[80]:
```
#Read the file into the Dataframe
df = spark.read.option("multiline","true").option("header", "true").
 ↪schema(schema).csv("dataset.txt")
```

[81]:
```
#Show the content to Verify the records
df.show(5)
```

```
+------+-----------+------------+-------+---------------+---------------+---
-------+---------+
|UserId|   Latitude|   Longitude|AllZero|     Altitude|      Timestamp|
Date|   Time|
+------+-----------+------------+-------+---------------+---------------+---
-------+---------+
```

```
|   100|39.974408918|116.303522101|
0|480.287355643045|40753.5306944444|2011-07-29|12:44:12|
|   100|39.974397078|116.303526932|
0|480.121151574803|40753.5307060185|2011-07-29|12:44:13|
|   100|39.973982524|116.303621837|
0|478.499455380577|40753.5307291667|2011-07-29|12:44:15|
|   100|39.973943291|116.303632641|
0|479.176988188976|40753.5307407407|2011-07-29|12:44:16|
|   100|39.973937148|116.303639667|
0|479.129432414698|40753.5307523148|2011-07-29|12:44:17|
+------+-----------+------------+------+---------------+----------------+---
-------+---------+
only showing top 5 rows
```

[126]:
```
#Convert the Date and time column to a datetime column
dt_tm = df.select("UserId", "Latitude", "Longitude", "AllZero",
 "Altitude",␣,→"Timestamp", "Date", "Time", concat(df.Date, lit("
 "), df.Time).→alias("DateTime"))
df1 = dt_tm.withColumn("DateTime", =to_timestamp(dt_tm.DateTime))
```

[127]:
```
df1.show(5)
```

```
+------+-----------+------------+------+---------------+----------------+---
-------+---------+-------------------+
|UserId|   Latitude|   Longitude|AllZero|     Altitude|      Timestamp|
Date|   Time|           DateTime|
+------+-----------+------------+------+---------------+----------------+---
-------+---------+-------------------+
|   100|39.974408918|116.303522101|
0|480.287355643045|40753.5306944444|2011-07-29|12:44:12|2011-07-29 12:44:12|
|   100|39.974397078|116.303526932|
0|480.121151574803|40753.5307060185|2011-07-29|12:44:13|2011-07-29 12:44:13|
|   100|39.973982524|116.303621837|
0|478.499455380577|40753.5307291667|2011-07-29|12:44:15|2011-07-29 12:44:15|
|   100|39.973943291|116.303632641|
0|479.176988188976|40753.5307407407|2011-07-29|12:44:16|2011-07-29 12:44:16|
|   100|39.973937148|116.303639667|
0|479.129432414698|40753.5307523148|2011-07-29|12:44:17|2011-07-29 12:44:17|
+------+-----------+------------+------+---------------+----------------+---
-------+---------+-------------------+
only showing top 5 rows
```

[128]:
```
#Clean the data and change the Datetime from GMT to GMT+8
df2 = df1.withColumn('datetimeBj', f.from_utc_timestamp(df1.DateTime, 'GMT+8'))
```

[131]:
```
df2.show(5, False)
```

```
+------+-----------  +-------------+-------+---------------   +---------------+---
------+--------+-----------------+-------------------+
|UserId|Latitude    |Longitude   |AllZero|Altitude      |Timestamp
|Date      |Time    |DateTime           |datetimeBj         |
+------+-----------  +-------------+-------+---------------   +---------------+---
------+--------+-----------------+-------------------+
|100   |39.974408918|116.303522101|0
|480.287355643045|40753.5306944444|2011-07-29|12:44:12|2011-07-29
12:44:12|2011-07-29 20:44:12|
|100   |39.974397078|116.303526932|0
|480.121151574803|40753.5307060185|2011-07-29|12:44:13|2011-07-29
12:44:13|2011-07-29 20:44:13|
|100   |39.973982524|116.303621837|0
|478.499455380577|40753.5307291667|2011-07-29|12:44:15|2011-07-29
12:44:15|2011-07-29 20:44:15|
|100   |39.973943291|116.303632641|0
|479.176988188976|40753.5307407407|2011-07-29|12:44:16|2011-07-29
12:44:16|2011-07-29 20:44:16|
|100   |39.973937148|116.303639667|0
|479.129432414698|40753.5307523148|2011-07-29|12:44:17|2011-07-29
12:44:17|2011-07-29 20:44:17|
+------+-----------  +-------------+-------+---------------   +---------------+---
------+--------+-----------------+-------------------+
only showing top 5 rows
```

[132]:
```
#Number of times data has been recorded for each user
dfWithDay = df2.withColumn("day", f.dayofmonth(df2.datetimeBj))
question2 = dfWithDay.select("UserId", "day").distinct()
question2.groupBy("UserId").count().orderBy(f.col("count").desc(),f.
→col("UserId").asc()).show(5, False)
```

```
+------+-----  +
|UserId|count|
+------+-----  +
|104   |31   |
|112   |31   |
|119   |31   |
|126   |31   |
|128   |31   |
+------+-----  +
only showing top 5 rows
```

[133]:
```
#Number of times where the count is greater than 100
question3 = dfWithDay.select("UserId", "day").groupBy("UserId",
"day").count(). →filter(f.col('count') >= 100)
question3.show()
```

```
+------+---+-----+
|UserId|day|count|
+------+---+-----+
|   126| 26|16431|
|   128| 21|39158|
|   103| 19| 1381|
|   104|  5| 1974|
|   113| 27| 1445|
|   115| 19| 3618|
|   115|  5| 3966|
|   101| 30| 1532|
|   104| 29|  760|
|   125| 12| 1964|
|   128| 20|38632|
|   103|  6|  576|
|   112|  2| 1892|
|   114|  9|  579|
|   115| 16| 7342|
|   119| 15| 3707|
|   125|  2| 4859|
|   126| 25| 3126|
|   114|  8|  761|
|   104| 20| 1435|
+------+---+-----+
only showing top 20 rows
```

[137]:
```
  #The highest altitude for each person
w = Window.partitionBy('UserId')
question4 = dfWithDay.withColumn('maxB', f.max('Altitude').over(w))\
    .where(f.col('Altitude') == f.col('maxB'))\
    .drop('maxB')
question4.select("UserId", "Altitude", "Date").distinct().orderBy(f.
 ↪col("Altitude").desc()).show(5,False)
```

```
+------+---------------+----------+
|UserId|Altitude       |Date      |
+------+---------------+----------+
|128   |107503.3       |2009-11-02|
|106   |36581.3648293963|2007-10-09|
|103   |25259.2        |2008-09-12|
|101   |24806.4        |2008-03-28|
|126   |19432.4        |2008-06-22|
+------+---------------+----------+
only showing top 5 rows
```

[168]:
```
#Max timespan for each user
window = Window.partitionBy('UserId').orderBy('DateTime')
question5_a == dfWithDay.withColumn("days_passed",
 f.datediff(dfWithDay. ↪DateTime,
                                  f.lag(dfWithDay.DateTime, 1).over(window)))
question5_a.groupBy("UserId","DateTime").agg(f.max("days_passed")).orderBy(f.
 ↪col("max(days_passed)").desc()).show(5)
```

```
+------+-------------------        +----------------+
|UserId|           DateTime|max(days_passed)|
+------+-------------------        +----------------+
|  114|2010-05-10 13:24:00|             934|
|  111|2009-07-14 21:37:22|             675|
|  115|2008-04-09 10:27:03|             133|
|  128|2008-04-05 01:11:27|             129|
|  128|2007-11-28 12:30:35|             121|
+------+-------------------        +----------------+
only showing top 5 rows
```

[170]:
```
#Define a function to find the distance based in latitude and longitude
def dist(long_x, lat_x, long_y, lat_y):
    return acos(
        sin(toRadians(lat_x)) * sin(toRadians(lat_y)) +
        cos(toRadians(lat_x)) * cos(toRadians(lat_y)) *
            cos(toRadians(long_x) - toRadians(long_y))
    ) * lit(6371.0)
```

[181]:
```
#We define a window based on UsedId and order the data based on DateTime and␣
 ↪used lag function we find the distance travelled be each user
w_question6 = Window().partitionBy("UserId").orderBy("DateTime")
question6 = dfWithDay.withColumn("dist", dist(
    "Longitude", "Latitude",
    lag("Longitude", 1).over(w_question6), lag("Latitude", 1).over(w_question6)
).alias("dist"))
question6_a = question6.select("UserId", "dist", "Date").groupBy("UserId",␣
 ↪"Date").agg(f.sum("dist")).filter(f.col("sum(dist)") != "NaN")
question6_a.show()
```

```
+------+----------+------------------+
|UserId|      Date|        sum(dist)|
+------+----------+------------------+
|   108|2007-10-02|1.6587260860085606|
|   108|2007-10-03|43.631893458311964|
|   108|2007-10-04| 147.0055120203384|
|   108|2007-10-06|121.43545197781773|
|   108|2007-10-07| 7.560496310794932|
|   108|2007-10-08|3.5475681716161547|
```

```
|   108|2007-10-09| 1.526404310495542|
|   101|2007-11-30| 35.71357885259294|
|   101|2007-12-02| 26.28155622300305|
|   101|2007-12-03|13.946825605235945|
|   101|2007-12-07|21.582506892854884|
|   101|2007-12-11|1.2158358355356826|
|   101|2007-12-12| 5.240018538952616|
|   101|2007-12-13| 131.2705465948174|
|   101|2007-12-15| 134.2261667257604|
|   101|2007-12-19| 157.9404104628446|
|   101|2007-12-22| 222.8093068237573|
|   101|2007-12-23| 8.639073137599118|
|   101|2007-12-26|2.4209762057765114|
|   101|2007-12-27|3.9078701419516726|
+------+----------+------------------+
only showing top 20 rows
```

[186]:
```
#For each user output the (earliest) day they travelled the most
w_6_b = Window.partitionBy('UserId')
question6_b = question6_a.withColumn('maxB', f.max('sum(dist)').over(w_6_b))\
    .where(f.col('sum(dist)') == f.col('maxB'))\
    .drop('maxB')

question6_b.show()
```

```
+------+----------+------------------+
|UserId|      Date|        sum(dist)|
+------+----------+------------------+
|   108|2007-10-04| 147.0055120203384|
|   101|2008-01-25| 912.3501366350881|
|   115|2007-11-28| 2097.446018079143|
|   126|2008-05-01|372.51247632567714|
|   103|2008-09-19| 29.44931227567783|
|   128|2009-02-22|10090.016973407062|
|   122|2009-07-31|1967.2757652846492|
|   111|2007-09-05| 2462.021045854465|
|   117|2007-06-22| 26.30900937760673|
|   112|2008-02-02| 1078.383461221913|
|   127|2008-10-05|1028.5007633041885|
|   107|2007-10-07| 8.659731775734203|
|   114|2010-05-28| 46.56970415564099|
|   100|2011-07-29|10.965117553721749|
|   130|2009-07-12|103.34148374177562|
|   129|2008-05-02| 317.7130265707075|
|   102|2011-12-31|31.239379907177888|
|   113|2010-05-20|19.666718577249753|
|   121|2009-10-05|12.850327012071368|
```

```
|   125|2008-08-27|1597.3327329740112|
+------+----------+------------------+
only showing top 20 rows
```

[187]:
```
#Find the total distance by all the users
question6_c = question6_a.select(f.sum("sum(dist)").alias("total_sum"))
question6_c.show()
```

```
+------------------+
|         total_sum|
+------------------+
|124208.62254385433|
+------------------+
```