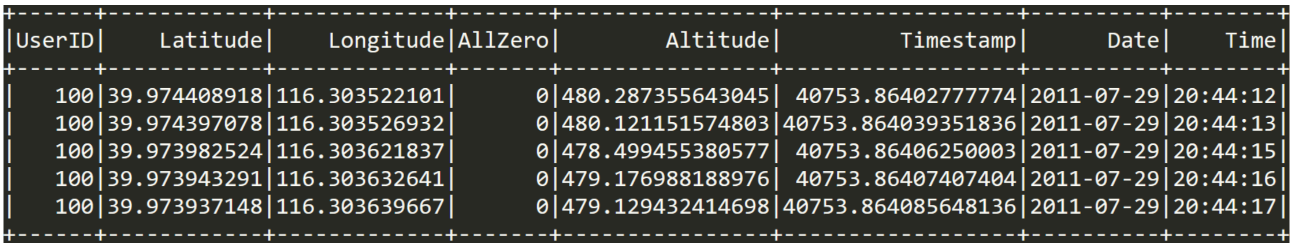
COMP336 Assignment2 Part1 Report

1. **Convert all the time to Beijing Time.**
   Code:

```
## Convert to Beijing Time
def ConvertTime(Date, Time):
        ## combine Date and Time to form n DateTime
        DateTime = '{} {}'.format(Date, Time)
        ## turn it into timestamp format
        DateTime = datetime.strptime(DateTime, "%Y-%m-%d %H:%M:%S")
        ## add 8 hours to DateTime
        DateTime = DateTime + timedelta(hours=8)
        return DateTime.strftime('%Y-%m-%d %H:%M:%S')

ConvertTimeUDF = udf(ConvertTime)

## turn the new column DateTime into timestamp format
temp = df.withColumn('DateTime', ConvertTimeUDF(col('Date'), col('Time')))

## update Date and Time based on the new DateTime
temp = temp.withColumn("Date", date_format('DateTime', 'yyyy-MM-dd'))
temp = temp.withColumn('Time', date_format('DateTime', 'HH:mm:ss'))

## add 8 hors to Timestamp
temp = temp.withColumn('Timestamp', col('Timestamp')+8/24)
```

Output result:

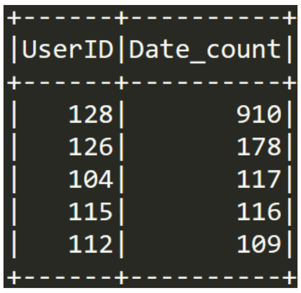


```
+------+------------+-------------+-------+---------------+------------------+----------+--------+
|UserID|    Latitude|    Longitude|AllZero|       Altitude|         Timestamp|      Date|    Time|
+------+------------+-------------+-------+---------------+------------------+----------+--------+
|   100|39.974408918|116.303522101|      0|480.287355643045| 40753.86402777774|2011-07-29|20:44:12|
|   100|39.974397078|116.303526932|      0|480.121151574803|40753.864039351836|2011-07-29|20:44:13|
|   100|39.973982524|116.303621837|      0|478.499455380577| 40753.86406250003|2011-07-29|20:44:15|
|   100|39.973943291|116.303632641|      0|479.176988188976| 40753.86407407404|2011-07-29|20:44:16|
|   100|39.973937148|116.303639667|      0|479.129432414698|40753.864085648136|2011-07-29|20:44:17|
+------+------------+-------------+-------+---------------+------------------+----------+--------+
```

2. **The Number of days with data recorded of each user**
   Code:

```
## group by UserID, count the number of the different dates
df2 = df1.groupBy("UserID").agg(countDistinct("Date").alias('Date_count'))

## first sort by Date_count descendingly, and then sort by UserID ascendingly
df2 = df2.orderBy(col('Date_count').desc(),col("UserID").asc())
```

Output result:



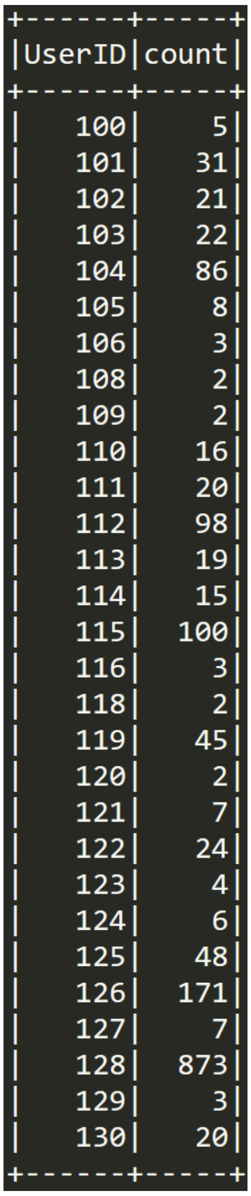
```
+------+----------+
|UserID|Date_count|
+------+----------+
|   128|       910|
|   126|       178|
|   104|       117|
|   115|       116|
|   112|       109|
+------+----------+
```

3. **The number of days with at least 100 data points of each user**

Code:

```
## group by UserID and Date, and then count the data points
df3 = df1.groupBy("UserID","Date").count()
## filter the ones with at least 100 data points, and then count then number of the
remaining dates for each user
df3 = df3.where(col("count") >= 100).groupBy("UserID").count()
```

Output result:

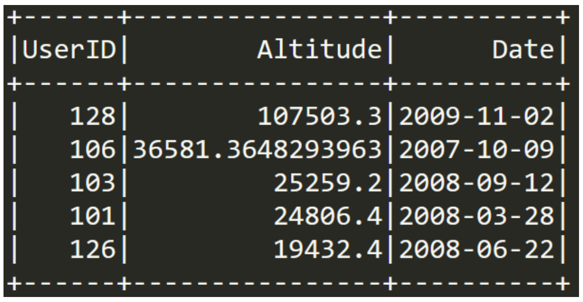


```
+------+-----+
|UserID|count|
+------+-----+
|   100|    5|
|   101|   31|
|   102|   21|
|   103|   22|
|   104|   86|
|   105|    8|
|   106|    3|
|   108|    2|
|   109|    2|
|   110|   16|
|   111|   20|
|   112|   98|
|   113|   19|
|   114|   15|
|   115|  100|
|   116|    3|
|   118|    2|
|   119|   45|
|   120|    2|
|   121|    7|
|   122|   24|
|   123|    4|
|   124|    6|
|   125|   48|
|   126|  171|
|   127|    7|
|   128|  873|
|   129|    3|
|   130|   20|
+------+-----+
```

4. **The highest altitude and the reaching date of each user**
   Code:

```
## find the highest Altitude for each person
df4 = df1.groupBy("UserID").agg(max('Altitude').alias('Altitude'))
## find the corresponding date(s) at which a person reach the Altitude
df4 = df4.join(df1, on=['UserID','Altitude'], how='left')
## choose the smallest date
df4 = df4.groupBy("UserID", 'Altitude').agg(min('Date').alias('Date'))
## sort by Altitude (and UserID)
df4 = df4.orderBy(col('Altitude').desc(),col("UserID").asc())
```

Output result:

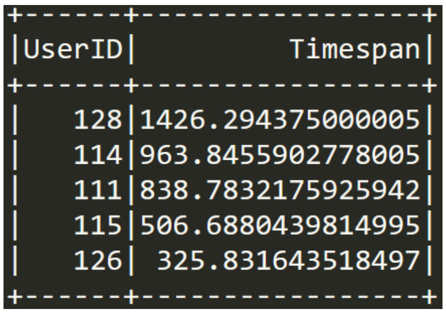


```
+------+----------------+----------+
|UserID|        Altitude|      Date|
+------+----------------+----------+
|   128|        107503.3|2009-11-02|
|   106|36581.3648293963|2007-10-09|
|   103|         25259.2|2008-09-12|
|   101|         24806.4|2008-03-28|
|   126|         19432.4|2008-06-22|
+------+----------------+----------+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

5. **The timespan of each user**
   Code:

```
## find the max and min Timestamp for each person
df5 = df1.groupBy("UserID") \
            .agg(max('Timestamp').alias('max_T'), \
                  min('Timestamp').alias('min_T'))
## calculate the Difference between max_T and min_T as Timespan
df5 = df5.withColumn('Timespan', col('max_T')-col('min_T')) \
          .select(col("UserID"), col('Timespan'))
## sort by Timespan (and UserID)
df5 = df5.orderBy(col('Timespan').desc(),col("UserID").asc())
```

Output result:



```
+------+-----------------+
|UserID|         Timespan|
+------+-----------------+
|   128|1426.294375000005|
|   114|963.8455902778005|
|   111|838.7832175925942|
|   115|506.6880439814995|
|   126| 325.831643518497|
+------+-----------------+
```

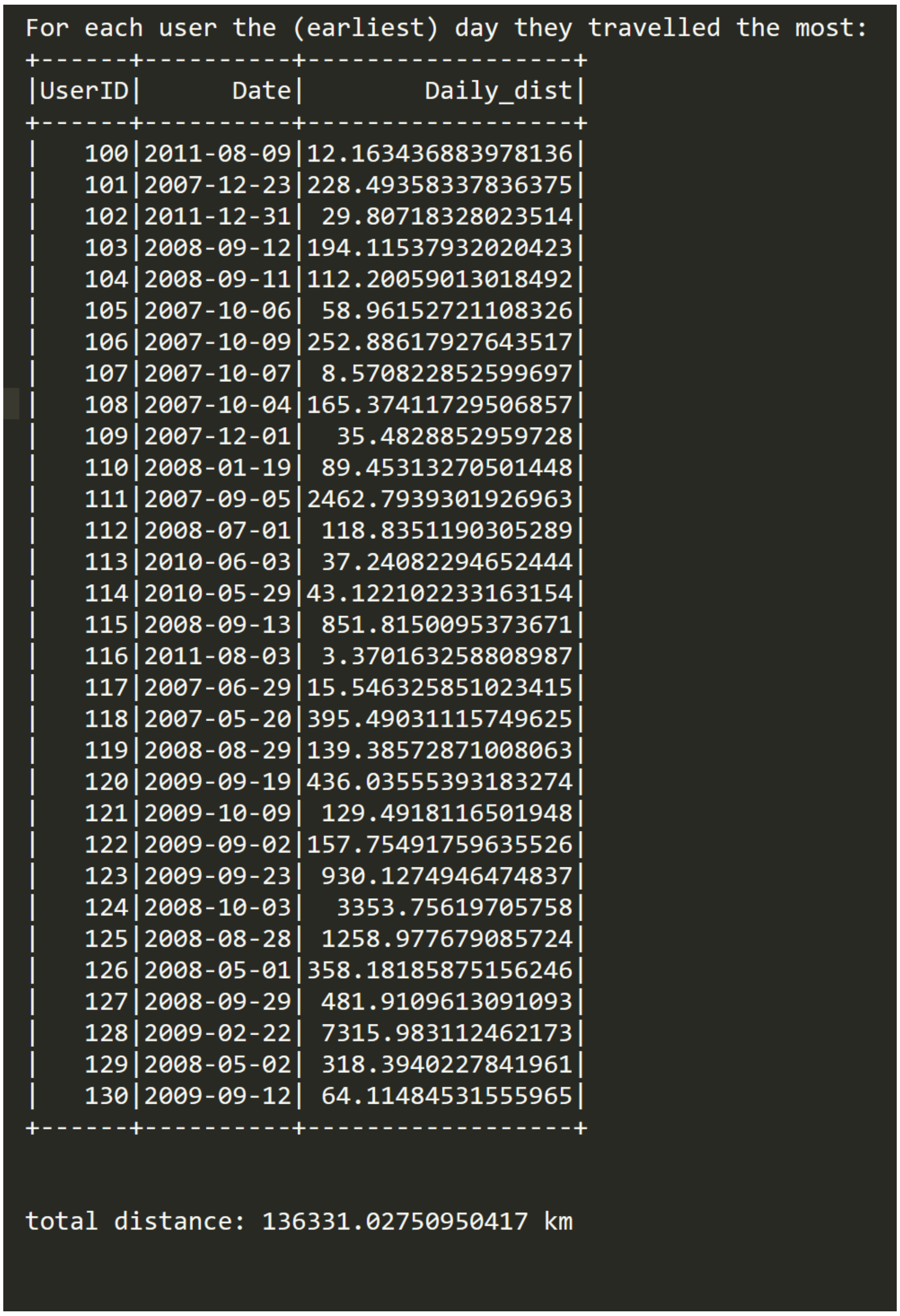
6. The earliest days on which each user travels most and the total distance

Code:

```
## calculate distance between two points
## Reference: https://stackoverflow.com/questions/19412462/getting-distance-between-two-points-
based-on-latitude-longitude
def distance(lat1,lon1,lat2,lon2):
        # approximate radius of earth in km
        R = 6373.0

        lat1 = math.radians(lat1)
        lon1 = math.radians(lon1)
        lat2 = math.radians(lat2)
        lon2 = math.radians(lon2)

        dlon = lon2 - lon1
        dlat = lat2 - lat1

        a = math.sin(dlat / 2)**2 + math.cos(lat1) * math.cos(lat2) * math.sin(dlon / 2)**2
        c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))

        d = R * c
        return d

DistanceUDF=udf(distance, DoubleType())

## partition by UserID, Date and sort by Timestamp
windowSpec = Window.partitionBy("UserID", 'Date').orderBy('Timestamp')
windowSpecAgg = Window.partitionBy("UserID", 'Date')

## selected needed columns and add row number
df6 = df1.select(col('UserID'), col('Date'), col('Timestamp'), \
                  col('Latitude').alias("lat2"), col('Longitude').alias("lon2"))
df6 = df6.withColumn("row",row_number().over(windowSpec))

## lag the positoin
df6 = df6.withColumn("lat1",lag("lat2",1).over(windowSpec))
df6 = df6.withColumn("lon1",lag("lon2",1).over(windowSpec))
## delete the null values
df6 = df6.na.drop()

## calculate distance and sum the daily distance within each partition
df6 = df6.withColumn('Distance', DistanceUDF(col('lat1'),col('lon1'),col('lat2'),col('lon2')))
df6 = df6.withColumn('Daily_dist', sum(col('Distance')).over(windowSpecAgg)) \
                .where(col("row")==2).select('UserID','Date','Daily_dist')

## find the longest daily distance for each user
```

```
windowSpec = Window.partitionBy("UserID").orderBy(col('Daily_dist').desc(),col("Date").asc())
df6_1 = df6.withColumn("row",row_number().over(windowSpec)).where(col("row")==1).drop('row')

## sum all the daily distance
n= df6.agg({'Daily_dist': 'sum'}).collect()[0][0]
```

Output result:



```
For each user the (earliest) day they travelled the most:
+------+----------+------------------+
|UserID|      Date|        Daily_dist|
+------+----------+------------------+
|   100|2011-08-09|12.163436883978136|
|   101|2007-12-23|228.49358337836375|
|   102|2011-12-31| 29.80718328023514|
|   103|2008-09-12|194.11537932020423|
|   104|2008-09-11|112.20059013018492|
|   105|2007-10-06| 58.96152721108326|
|   106|2007-10-09|252.88617927643517|
|   107|2007-10-07| 8.570822852599697|
|   108|2007-10-04|165.37411729506857|
|   109|2007-12-01|  35.4828852959728|
|   110|2008-01-19| 89.45313270501448|
|   111|2007-09-05|2462.7939301926963|
|   112|2008-07-01| 118.8351190305289|
|   113|2010-06-03| 37.24082294652444|
|   114|2010-05-29|43.122102233163154|
|   115|2008-09-13| 851.8150095373671|
|   116|2011-08-03| 3.370163258808987|
|   117|2007-06-29|15.546325851023415|
|   118|2007-05-20|395.49031115749625|
|   119|2008-08-29|139.38572871008063|
|   120|2009-09-19|436.03555393183274|
|   121|2009-10-09| 129.4918116501948|
|   122|2009-09-02|157.75491759635526|
|   123|2009-09-23| 930.1274946474837|
|   124|2008-10-03|  3353.75619705758|
|   125|2008-08-28| 1258.977679085724|
|   126|2008-05-01|358.18185875156246|
|   127|2008-09-29| 481.9109613091093|
|   128|2009-02-22| 7315.983112462173|
|   129|2008-05-02| 318.3940227841961|
|   130|2009-09-12| 64.11484531555965|
+------+----------+------------------+


total distance: 136331.02750950417 km
```