Assignment 4- Aparna Bharathi Suresh

Question 1:

Install Spark 3.5.3 in your dev environment

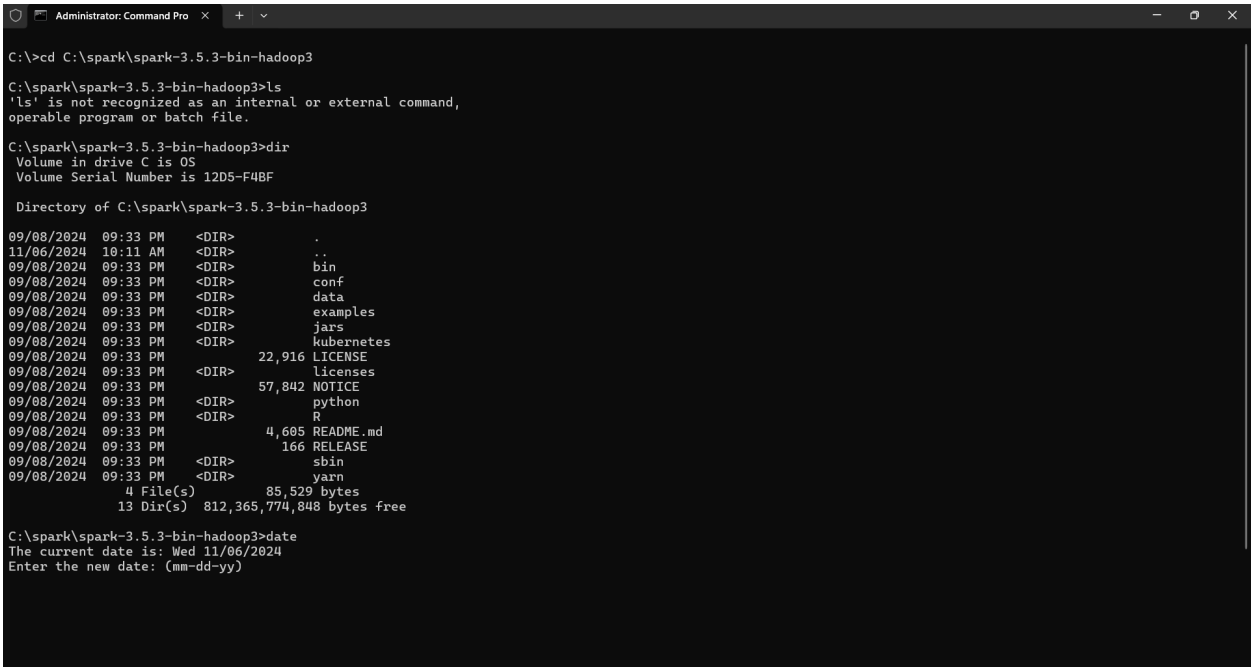
Once you have completed the installation, take a directory listing of the contents of the Spark installation directory:

- cd spark-3.5.3-bin-hadoop3
- ls
- date

Submit the screenshot of the output of those three commands.

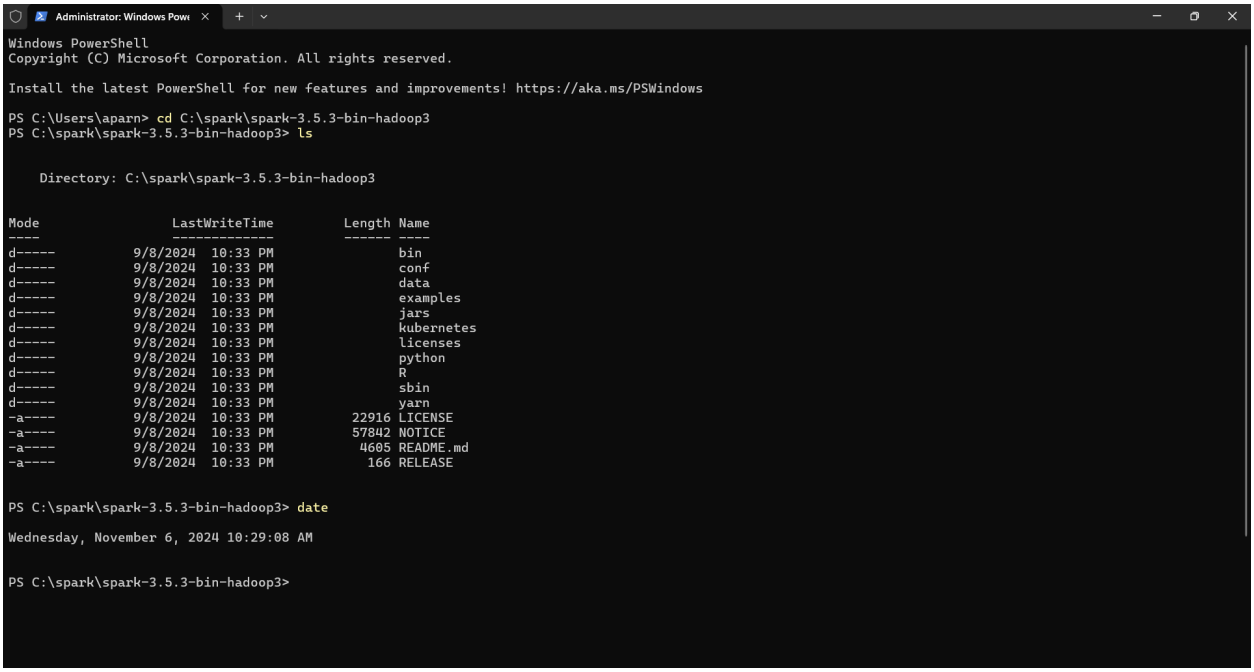
Answer-1:

ls command didn't work in Windows Command Prompt, so I have used dir command.



```
Administrator: Command Pro

C:\>cd C:\spark\spark-3.5.3-bin-hadoop3

C:\spark\spark-3.5.3-bin-hadoop3>ls
'ls' is not recognized as an internal or external command,
operable program or batch file.

C:\spark\spark-3.5.3-bin-hadoop3>dir
 Volume in drive C is OS
 Volume Serial Number is 12D5-F4BF

 Directory of C:\spark\spark-3.5.3-bin-hadoop3

09/08/2024  09:33 PM    <DIR>          .
11/06/2024  10:11 AM    <DIR>          ..
09/08/2024  09:33 PM    <DIR>          bin
09/08/2024  09:33 PM    <DIR>          conf
09/08/2024  09:33 PM    <DIR>          data
09/08/2024  09:33 PM    <DIR>          examples
09/08/2024  09:33 PM    <DIR>          jars
09/08/2024  09:33 PM    <DIR>          kubernetes
09/08/2024  09:33 PM            22,916 LICENSE
09/08/2024  09:33 PM    <DIR>          licenses
09/08/2024  09:33 PM            57,842 NOTICE
09/08/2024  09:33 PM    <DIR>          python
09/08/2024  09:33 PM    <DIR>          R
09/08/2024  09:33 PM             4,605 README.md
09/08/2024  09:33 PM               166 RELEASE
09/08/2024  09:33 PM    <DIR>          sbin
09/08/2024  09:33 PM    <DIR>          yarn
               4 File(s)         85,529 bytes
              13 Dir(s)  812,365,774,848 bytes free

C:\spark\spark-3.5.3-bin-hadoop3>date
The current date is: Wed 11/06/2024
Enter the new date: (mm-dd-yy)
```

ls command in Windows PowerShell:



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\aparn> cd C:\spark\spark-3.5.3-bin-hadoop3
PS C:\spark\spark-3.5.3-bin-hadoop3> ls


    Directory: C:\spark\spark-3.5.3-bin-hadoop3


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          9/8/2024  10:33 PM                bin
d-----          9/8/2024  10:33 PM                conf
d-----          9/8/2024  10:33 PM                data
d-----          9/8/2024  10:33 PM                examples
d-----          9/8/2024  10:33 PM                jars
d-----          9/8/2024  10:33 PM                kubernetes
d-----          9/8/2024  10:33 PM                licenses
d-----          9/8/2024  10:33 PM                python
d-----          9/8/2024  10:33 PM                R
d-----          9/8/2024  10:33 PM                sbin
d-----          9/8/2024  10:33 PM                yarn
-a----          9/8/2024  10:33 PM          22916 LICENSE
-a----          9/8/2024  10:33 PM          57842 NOTICE
-a----          9/8/2024  10:33 PM           4605 README.md
-a----          9/8/2024  10:33 PM            166 RELEASE


PS C:\spark\spark-3.5.3-bin-hadoop3> date

Wednesday, November 6, 2024 10:29:08 AM


PS C:\spark\spark-3.5.3-bin-hadoop3>
```

Question 2:

Show data

more departuredelays.csv

First, open a pyspark shell

pyspark

Define the right schema for departuredelays.csv programmatically and not using DDL

```
from pyspark.sql.types import *

sch = StructType([

  StructField("date", StringType(), False),

  StructField("delay", IntegerType(), False),

  StructField("distance", IntegerType(), False),

  StructField("origin", StringType(), False),

  StructField("destination", StringType(), False)
```

```
])
```

Read the departure delay data from the file using this schema

```
df = spark.read.csv('departuredelays.csv', header=True, schema=sch)
```

Show the first 10 rows of the data

```
df.show(10)
```

Print the schema using printSchema()

```
df.printSchema()
```

Create a new DataFrame where the destination is "SJC"

```
df_sj=df.filter(df["destination"]=="SJC")
```

Show the first 10 rows of that data

```
df_sj.show(10)
```

Calculate the departure delay averages grouped by the origin airports

```
from pyspark.sql.functions import avg

df_delay=df.groupBy('origin').agg(avg('delay').alias('avg_delay'))
```

Show the entire average data

df_delay.show()- shows only 20 records

```
df_delay.count()

df_delay.show(255)
```

```
Administrator: Command Pro

>>> df_delay.count()
255
>>> df_delay.show(255)
+------+--------------------+
|origin|           avg_delay|
+------+--------------------+
|   BUR|   8.3167946446150 81|
|   EUG|   7.568056648308419|
|   BTM| -0.7666666666666667|
|   COD|   2.374301675977654|
|   FAR|  15.780968006562757|
|   FSM|   4.881666666666667|
|   DCA|   8.012508036705828|
|   CID|   16.57703927492447|
|   EVV|  11.860088365243005|
|   CRW|  11.121693121693122|
|   CDV|  -5.752808988764045|
|   CMH|  14.022459893048127|
|   CAK|   8.965957446808511|
|   CHO|   8.016556291390728|
|   CEC|   11.60655737704918|
|   CVG|  11.397058823529411|
|   BUF|   12.44192439862543|
|   CDC| 0.33116883116883117|
|   ACT|   0.897025171624714|
|   AUS|  10.835627368841013|
|   ATW|  14.175652173913043|
|   DHN|    6.95575221238938|
|   AVL|   8.158119658119658|
```

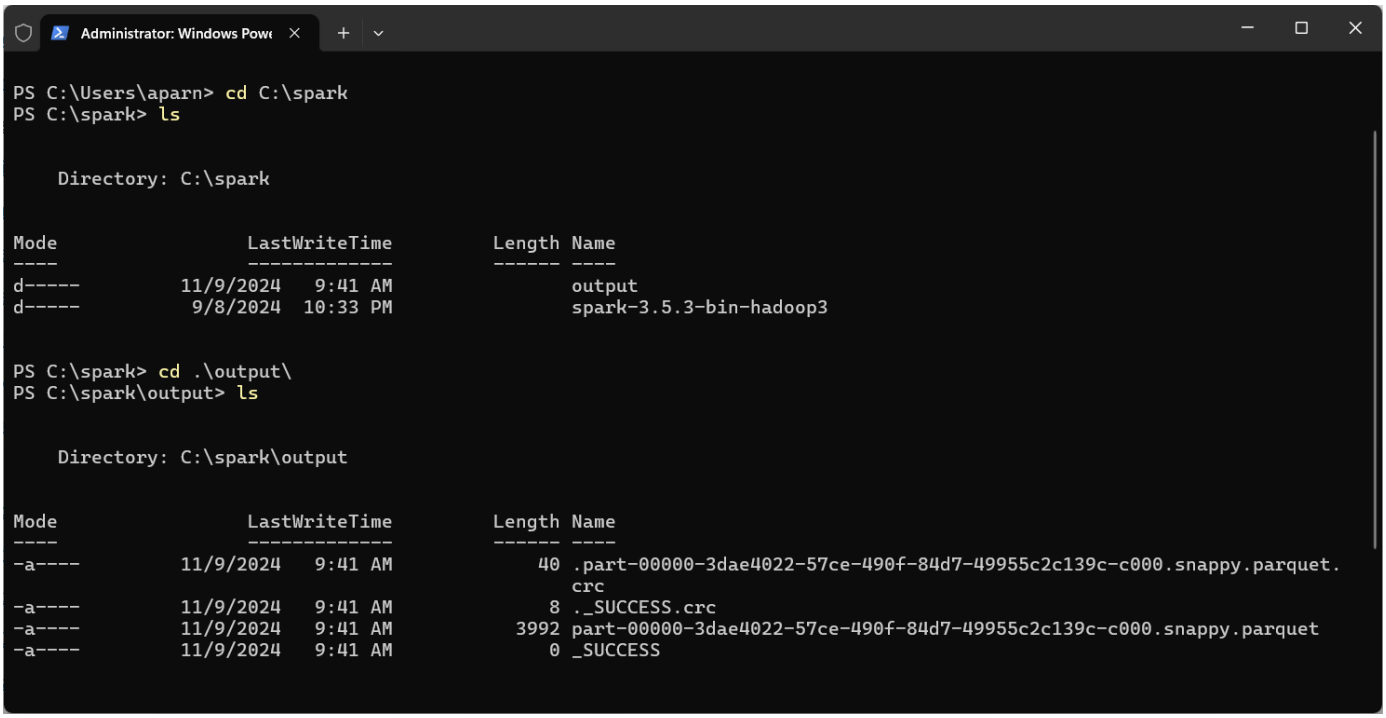
Save the average data in a parquet file

path = "C:\\spark\\output"

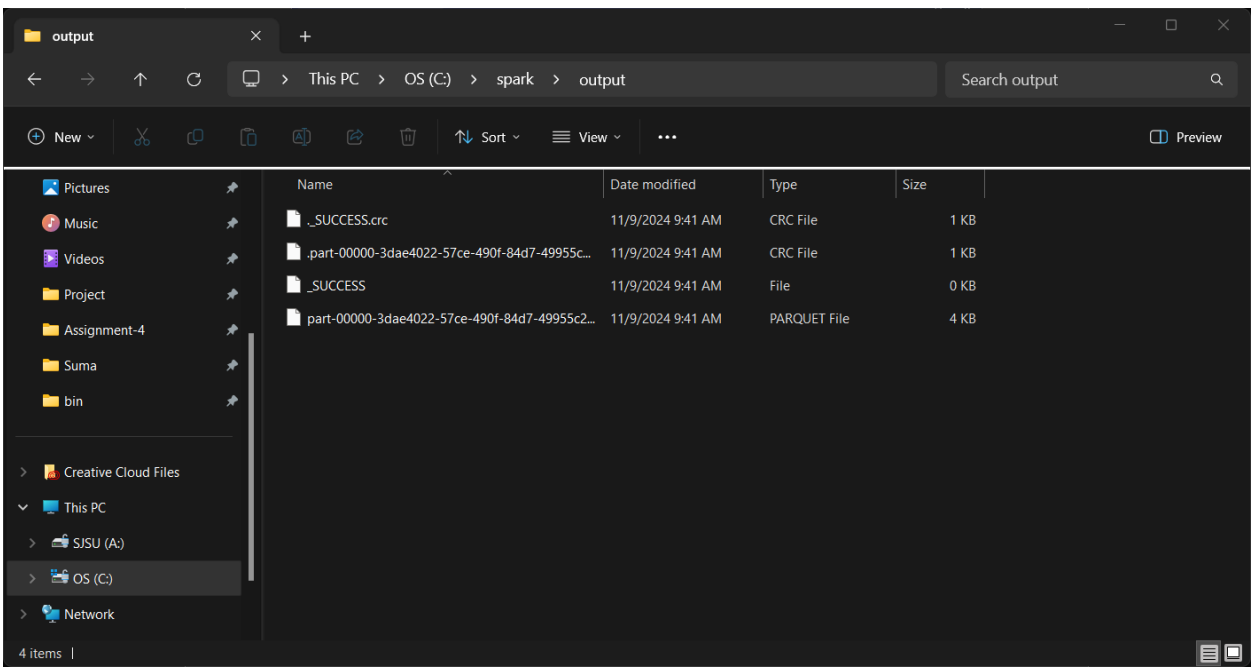
df_delay.write.format("parquet").save(path)

```
Administrator: Command Pro

C:\spark>dir
 Volume in drive C is OS
 Volume Serial Number is 12D5-F4BF

 Directory of C:\spark

11/09/2024  09:41 AM    <DIR>          .
11/09/2024  09:41 AM    <DIR>          output
09/08/2024  09:33 PM    <DIR>          spark-3.5.3-bin-hadoop3
               0 File(s)              0 bytes
               3 Dir(s)  809,998,536,704 bytes free

C:\spark>cd output

C:\spark\output>dir
 Volume in drive C is OS
 Volume Serial Number is 12D5-F4BF

 Directory of C:\spark\output

11/09/2024  09:41 AM    <DIR>          .
11/09/2024  09:41 AM    <DIR>          ..
11/09/2024  09:41 AM                40 .part-00000-3dae4022-57ce-490f-84d7-49955c2c139c-c000.snappy.parquet.crc
11/09/2024  09:41 AM                 8 ._SUCCESS.crc
11/09/2024  09:41 AM             3,992 part-00000-3dae4022-57ce-490f-84d7-49955c2c139c-c000.snappy.parquet
11/09/2024  09:41 AM                 0 _SUCCESS
               4 File(s)          4,040 bytes
               2 Dir(s)  809,998,602,240 bytes free

C:\spark\output>
```

Powershell:



```
PS C:\Users\aparn> cd C:\spark
PS C:\spark> ls


    Directory: C:\spark


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         11/9/2024   9:41 AM                output
d-----          9/8/2024  10:33 PM                spark-3.5.3-bin-hadoop3


PS C:\spark> cd .\output\
PS C:\spark\output> ls


    Directory: C:\spark\output


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         11/9/2024   9:41 AM             40 .part-00000-3dae4022-57ce-490f-84d7-49955c2c139c-c000.snappy.parquet.
                                                  crc
-a----         11/9/2024   9:41 AM              8 .._SUCCESS.crc
-a----         11/9/2024   9:41 AM           3992 part-00000-3dae4022-57ce-490f-84d7-49955c2c139c-c000.snappy.parquet
-a----         11/9/2024   9:41 AM              0 _SUCCESS
```



The best-performing origin airport and the worst-performing airport with their respective average values (as a text file)

best_performing = df_delay.orderBy('avg_delay').first()

worst_performing = df_delay.orderBy('avg_delay', ascending=False).first()

```
with open("best_and_worst_airports.txt", "w") as file:

    file.write(f"Best Performing Airport: {best_performing['origin']} - Avg Delay: {best_performing['avg_delay']}\n")

    file.write(f"Worst Performing Airport: {worst_performing['origin']} - Avg Delay: {worst_performing['avg_delay']}\n")
```