# 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.

A screenshot of a computer

Description automatically generated

ls command in Windows PowerShell:

A screenshot of a computer

Description automatically generated

### 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

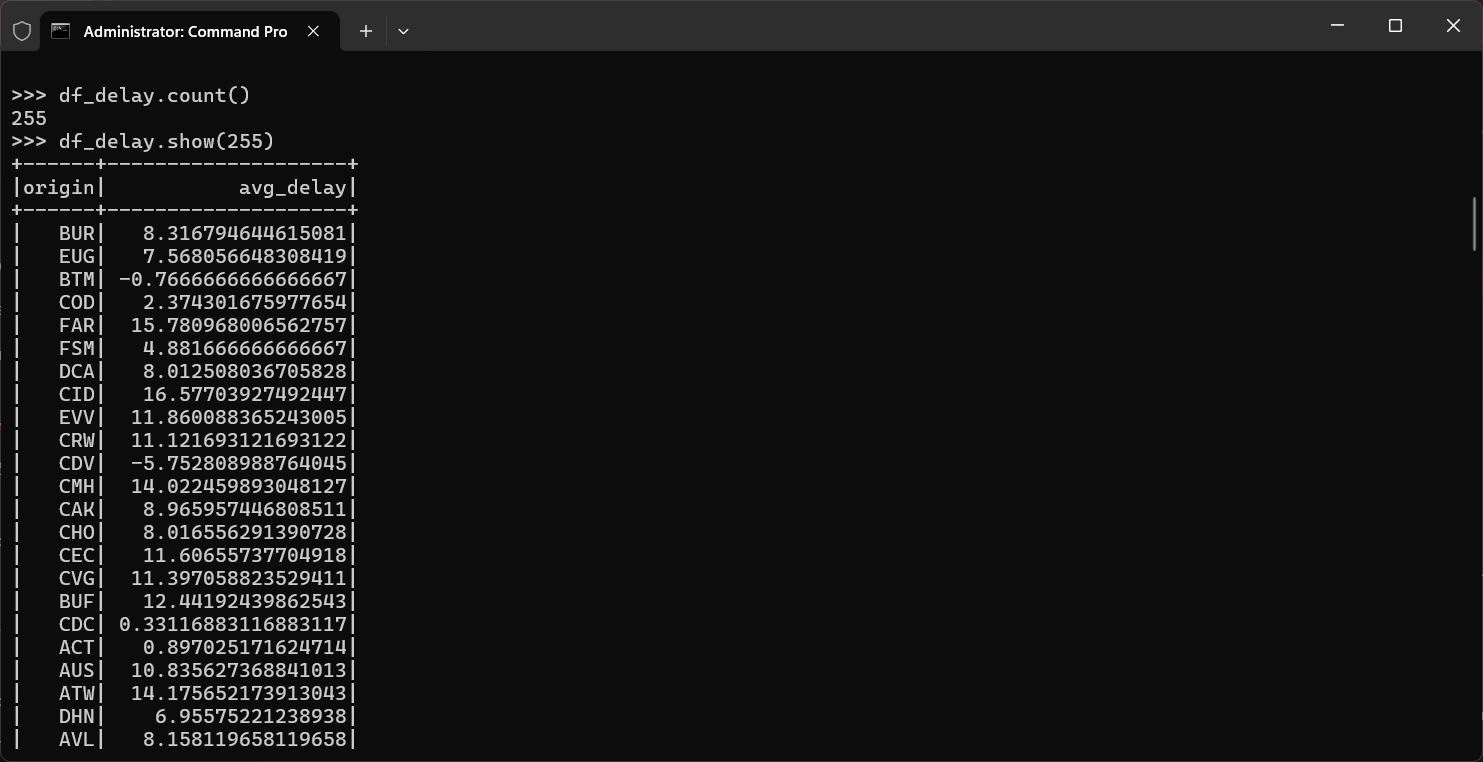
df\_delay=df.groupBy(‘origin’).agg(avg(‘delay’).alias(‘avg\_delay’))

### Show the entire average data

df\_delay.show()- shows only 20 records

df\_delay.count()

df\_delay.show(255)



### Save the average data in a parquet file

path = "C:\\spark\\output"  
df\_delay.write.format(“parquet”).save(path)

A screenshot of a computer

Description automatically generated

Powershell:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## **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")

A screenshot of a computer

Description automatically generated