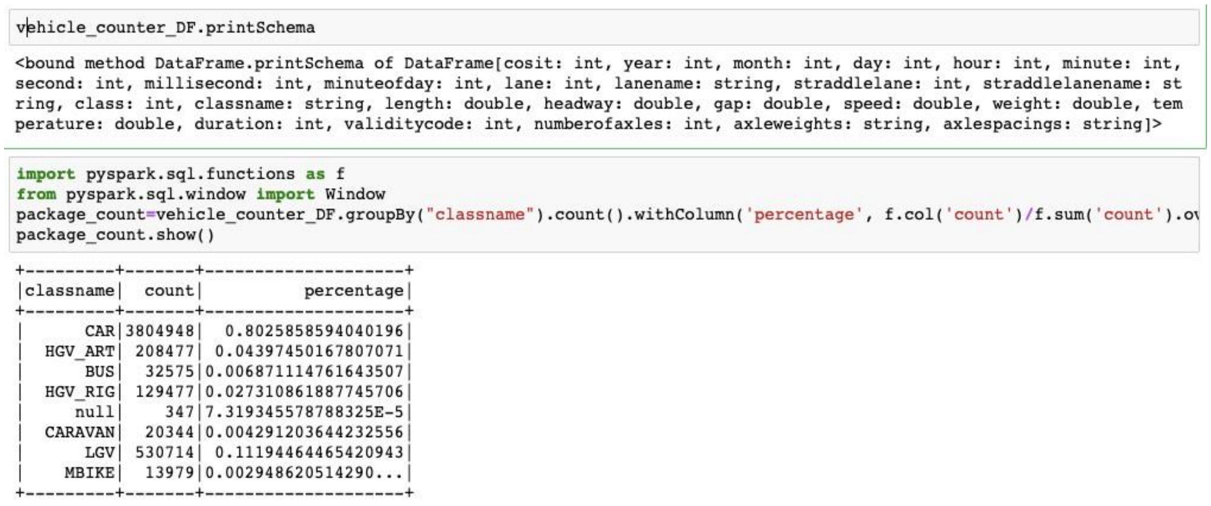
SUMMARY ANALYTICS

QUESTION 1



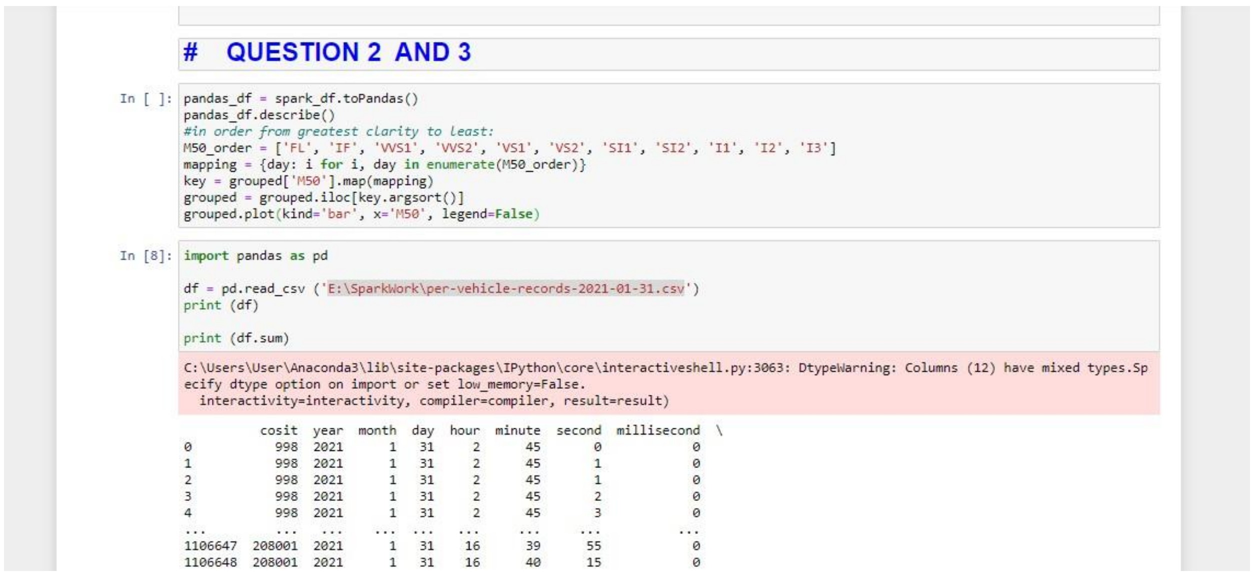
```
vehicle_counter_DF.printSchema
```

```
<bound method DataFrame.printSchema of DataFrame[cosit: int, year: int, month: int, day: int, hour: int, minute: int,
second: int, millisecond: int, minuteofday: int, lane: int, lanename: string, straddlelane: int, straddlelanename: st
ring, class: int, classname: string, length: double, headway: double, gap: double, speed: double, weight: double, tem
perature: double, duration: int, validitycode: int, numberofaxles: int, axleweights: string, axlespacings: string]>
```

```
import pyspark.sql.functions as f
from pyspark.sql.window import Window
package_count=vehicle_counter_DF.groupBy("classname").count().withColumn('percentage', f.col('count')/f.sum('count').ov
package_count.show()
```

```
+---------+-------+--------------------+
|classname|  count|          percentage|
+---------+-------+--------------------+
|      CAR|3804948|  0.8025858594040196|
|  HGV_ART| 208477| 0.04397450167807071|
|      BUS|  32575|0.006871114761643507|
|  HGV_RIG| 129477|0.027310861887745706|
|     null|    347|7.319345578788325E-5|
|  CARAVAN|  20344|0.004291203644232556|
|      LGV| 530714| 0.11194464465420943|
|    MBIKE|  13979|0.002948620514290...|
+---------+-------+--------------------+
```

QUESTION 2 AND 3



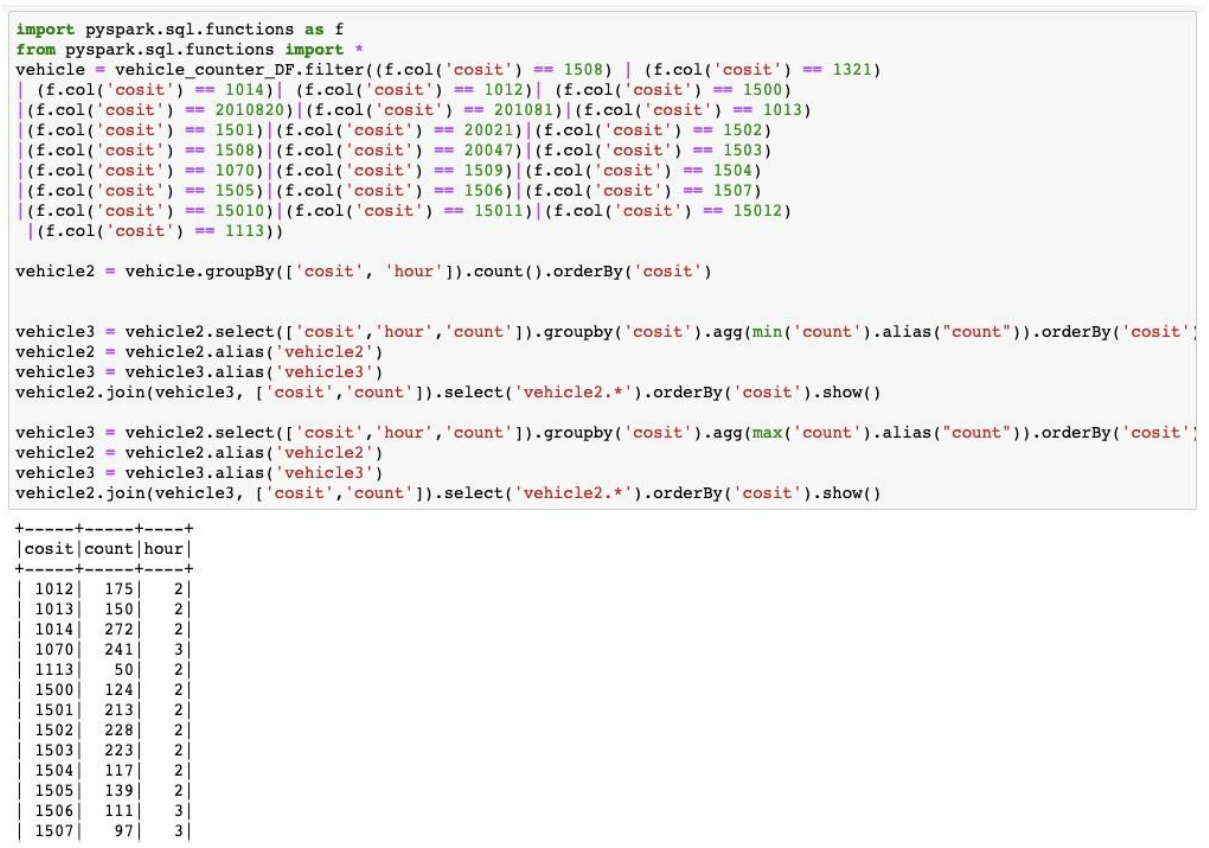
QUESTION 2 AND 3

```
In [ ]: pandas_df = spark_df.toPandas()
        pandas_df.describe()
        #in order from greatest clarity to least:
        M50_order = ['FL', 'IF', 'VVS1', 'VVS2', 'VS1', 'VS2', 'SI1', 'SI2', 'I1', 'I2', 'I3']
        mapping = {day: i for i, day in enumerate(M50_order)}
        key = grouped['M50'].map(mapping)
        grouped = grouped.iloc[key.argsort()]
        grouped.plot(kind='bar', x='M50', legend=False)
```

```
In [8]: import pandas as pd

        df = pd.read_csv ('E:\SparkWork\per-vehicle-records-2021-01-31.csv')
        print (df)

        print (df.sum)
```

```
C:\Users\User\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3063: DtypeWarning: Columns (12) have mixed types.Sp
ecify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

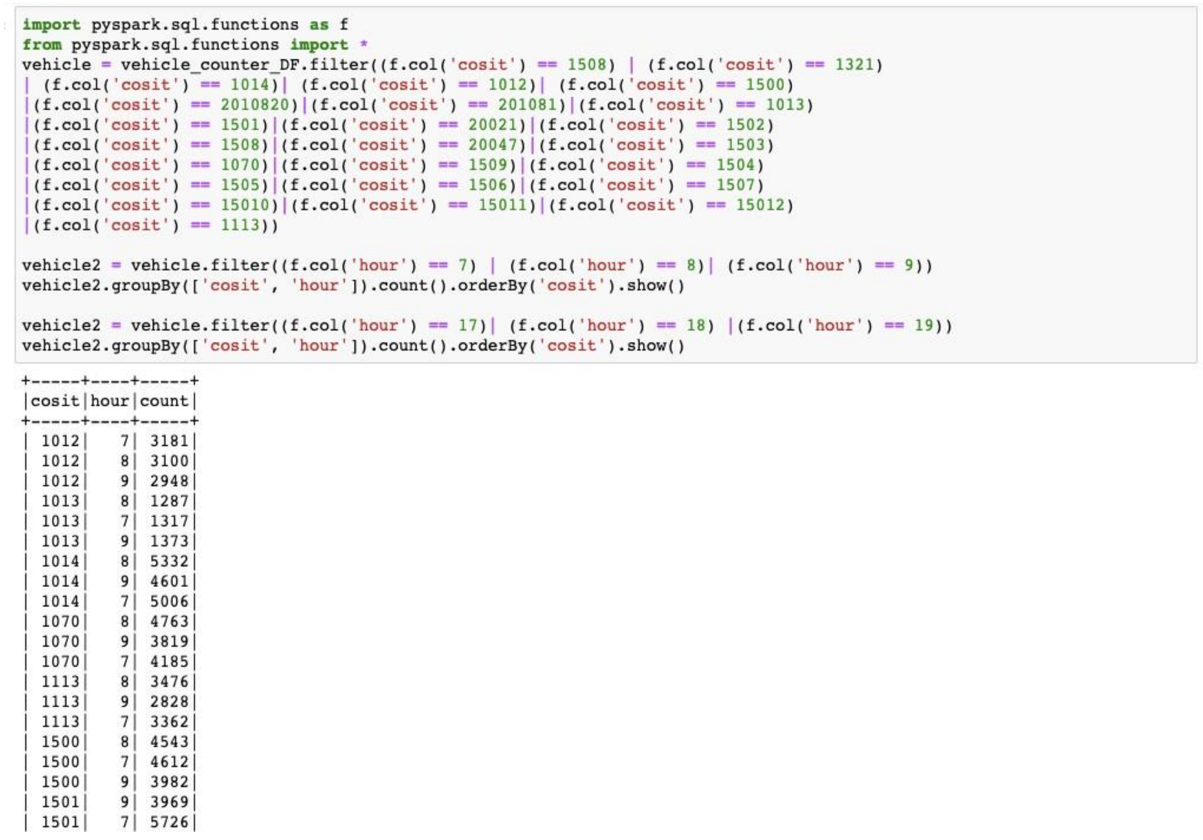
```
          cosit  year  month  day  hour  minute  second  millisecond  \
0           998  2021      1   31     2      45       0            0
1           998  2021      1   31     2      45       1            0
2           998  2021      1   31     2      45       1            0
3           998  2021      1   31     2      45       2            0
4           998  2021      1   31     2      45       3            0
...         ...   ...    ...  ...   ...     ...     ...          ...
1106647  208001  2021      1   31    16      39      55            0
1106648  208001  2021      1   31    16      40      15            0
```



```python
import pyspark.sql.functions as f
from pyspark.sql.functions import *
vehicle = vehicle_counter_DF.filter((f.col('cosit') == 1508) | (f.col('cosit') == 1321)
| (f.col('cosit') == 1014)| (f.col('cosit') == 1012)| (f.col('cosit') == 1500)
|(f.col('cosit') == 2010820)|(f.col('cosit') == 201081)|(f.col('cosit') == 1013)
|(f.col('cosit') == 1501)|(f.col('cosit') == 20021)|(f.col('cosit') == 1502)
|(f.col('cosit') == 1508)|(f.col('cosit') == 20047)|(f.col('cosit') == 1503)
|(f.col('cosit') == 1070)|(f.col('cosit') == 1509)|(f.col('cosit') == 1504)
|(f.col('cosit') == 1505)|(f.col('cosit') == 1506)|(f.col('cosit') == 1507)
|(f.col('cosit') == 15010)|(f.col('cosit') == 15011)|(f.col('cosit') == 15012)
 |(f.col('cosit') == 1113))

vehicle2 = vehicle.groupBy(['cosit', 'hour']).count().orderBy('cosit')


vehicle3 = vehicle2.select(['cosit','hour','count']).groupby('cosit').agg(min('count').alias("count")).orderBy('cosit'
vehicle2 = vehicle2.alias('vehicle2')
vehicle3 = vehicle3.alias('vehicle3')
vehicle2.join(vehicle3, ['cosit','count']).select('vehicle2.*').orderBy('cosit').show()

vehicle3 = vehicle2.select(['cosit','hour','count']).groupby('cosit').agg(max('count').alias("count")).orderBy('cosit'
vehicle2 = vehicle2.alias('vehicle2')
vehicle3 = vehicle3.alias('vehicle3')
vehicle2.join(vehicle3, ['cosit','count']).select('vehicle2.*').orderBy('cosit').show()
```

```
+-----+-----+----+
|cosit|count|hour|
+-----+-----+----+
| 1012|  175|   2|
| 1013|  150|   2|
| 1014|  272|   2|
| 1070|  241|   3|
| 1113|   50|   2|
| 1500|  124|   2|
| 1501|  213|   2|
| 1502|  228|   2|
| 1503|  223|   2|
| 1504|  117|   2|
| 1505|  139|   2|
| 1506|  111|   3|
| 1507|   97|   3|
```

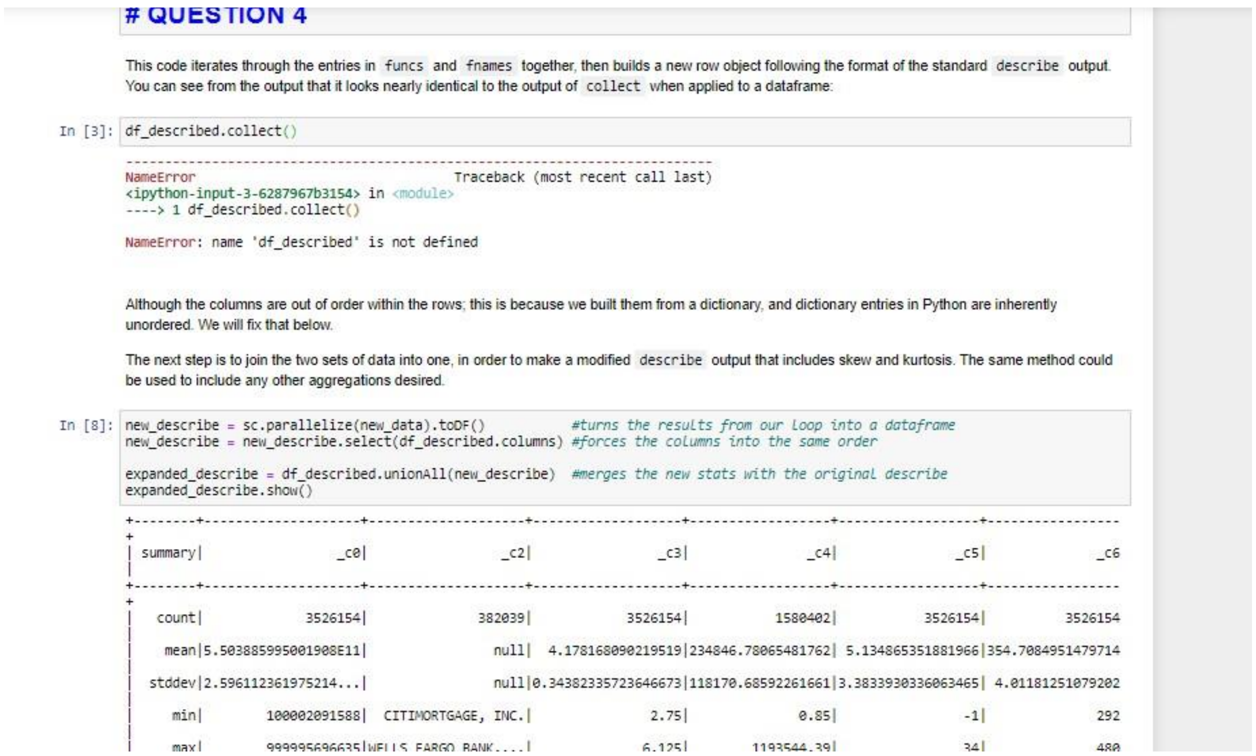
QUESTION 3



```
import pyspark.sql.functions as f
from pyspark.sql.functions import *
vehicle = vehicle_counter_DF.filter((f.col('cosit') == 1508) | (f.col('cosit') == 1321)
| (f.col('cosit') == 1014)| (f.col('cosit') == 1012)| (f.col('cosit') == 1500)
|(f.col('cosit') == 2010820)|(f.col('cosit') == 201081)|(f.col('cosit') == 1013)
|(f.col('cosit') == 1501)|(f.col('cosit') == 20021)|(f.col('cosit') == 1502)
|(f.col('cosit') == 1508)|(f.col('cosit') == 20047)|(f.col('cosit') == 1503)
|(f.col('cosit') == 1070)|(f.col('cosit') == 1509)|(f.col('cosit') == 1504)
|(f.col('cosit') == 1505)|(f.col('cosit') == 1506)|(f.col('cosit') == 1507)
|(f.col('cosit') == 15010)|(f.col('cosit') == 15011)|(f.col('cosit') == 15012)
|(f.col('cosit') == 1113))

vehicle2 = vehicle.filter((f.col('hour') == 7) | (f.col('hour') == 8)| (f.col('hour') == 9))
vehicle2.groupBy(['cosit', 'hour']).count().orderBy('cosit').show()

vehicle2 = vehicle.filter((f.col('hour') == 17)| (f.col('hour') == 18) |(f.col('hour') == 19))
vehicle2.groupBy(['cosit', 'hour']).count().orderBy('cosit').show()
```

```
+-----+----+-----+
|cosit|hour|count|
+-----+----+-----+
| 1012|   7| 3181|
| 1012|   8| 3100|
| 1012|   9| 2948|
| 1013|   8| 1287|
| 1013|   7| 1317|
| 1013|   9| 1373|
| 1014|   8| 5332|
| 1014|   9| 4601|
| 1014|   7| 5006|
| 1070|   8| 4763|
| 1070|   9| 3819|
| 1070|   7| 4185|
| 1113|   8| 3476|
| 1113|   9| 2828|
| 1113|   7| 3362|
| 1500|   8| 4543|
| 1500|   7| 4612|
| 1500|   9| 3982|
| 1501|   9| 3969|
| 1501|   7| 5726|
```

QUESTION 4 AND 5



QUESTION 4

This code iterates through the entries in `funcs` and `fnames` together, then builds a new row object following the format of the standard `describe` output. You can see from the output that it looks nearly identical to the output of `collect` when applied to a dataframe:

```
In [3]: df_described.collect()
```

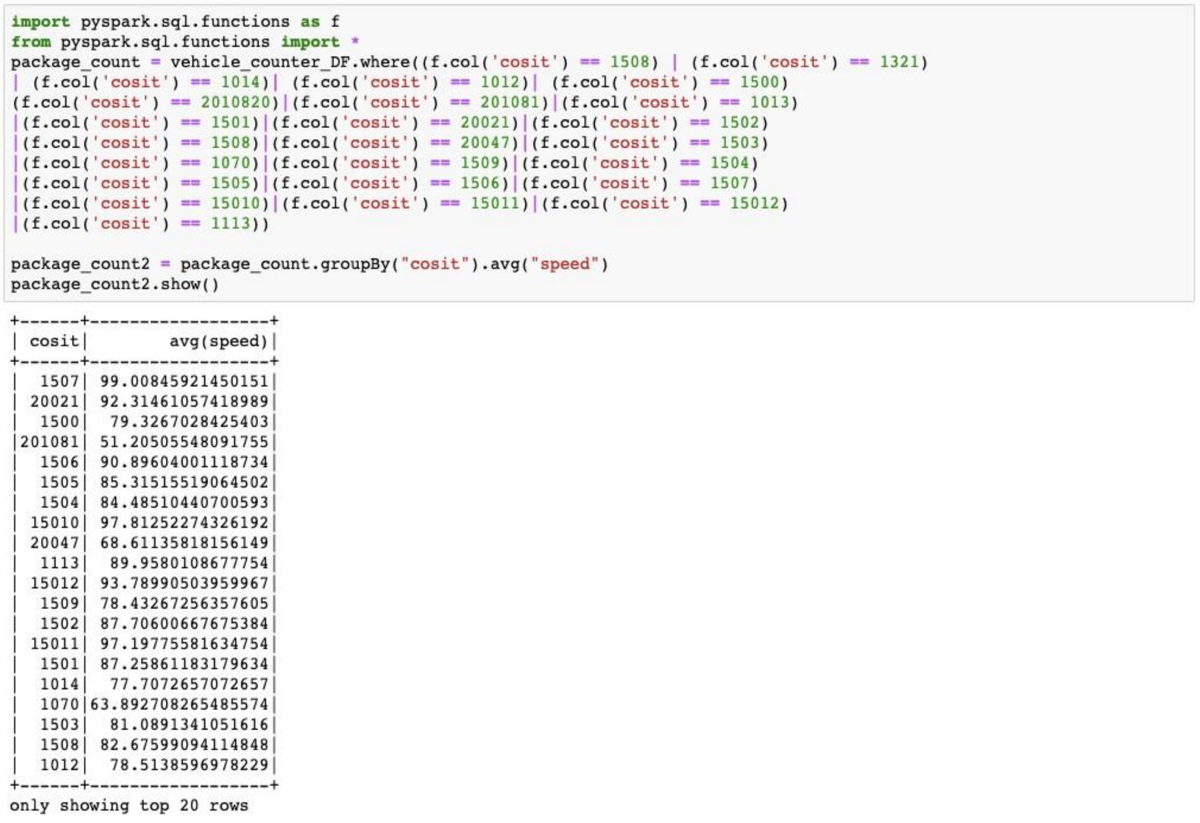
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-6287967b3154> in <module>
----> 1 df_described.collect()

NameError: name 'df_described' is not defined
```

Although the columns are out of order within the rows; this is because we built them from a dictionary, and dictionary entries in Python are inherently unordered. We will fix that below.

The next step is to join the two sets of data into one, in order to make a modified `describe` output that includes skew and kurtosis. The same method could be used to include any other aggregations desired.

```
In [8]: new_describe = sc.parallelize(new_data).toDF()           #turns the results from our loop into a dataframe
        new_describe = new_describe.select(df_described.columns) #forces the columns into the same order

        expanded_describe = df_described.unionAll(new_describe)  #merges the new stats with the original describe
        expanded_describe.show()
```

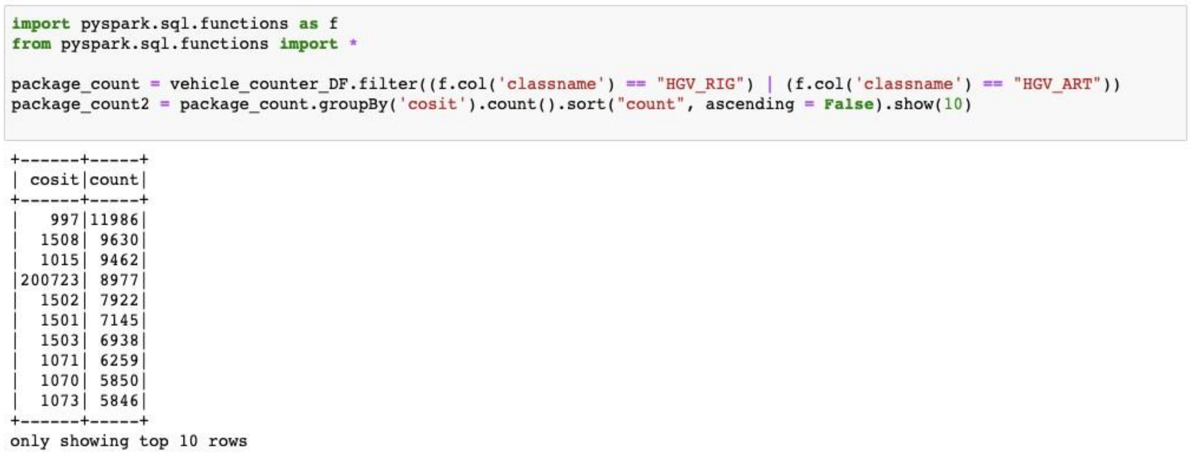
```
+--------+-------------------+-------------------+------------------+------------------+------------------+-----------------
+
| summary|                _c0|                _c2|               _c3|               _c4|               _c5|              _c6
|
+--------+-------------------+-------------------+------------------+------------------+------------------+-----------------
+
|   count|            3526154|             382039|           3526154|           1580402|           3526154|          3526154
|
|    mean|5.503885995001908E11|               null|  4.178168090219519|234846.78065481762| 5.134865351881966|354.7084951479714
|
|  stddev|2.596112361975214...|               null|0.34382335723646673|118170.68592261661|3.3833930336063465| 4.01181251079202
|
|     min|        100002091588|  CITIMORTGAGE, INC.|              2.75|              0.85|                -1|              292
|
|     max|        999995696635|WELLS FARGO BANK....|             6.125|         1193544.39|                34|              480
```



```
import pyspark.sql.functions as f
from pyspark.sql.functions import *
package_count = vehicle_counter_DF.where((f.col('cosit') == 1508) | (f.col('cosit') == 1321)
| (f.col('cosit') == 1014)| (f.col('cosit') == 1012)| (f.col('cosit') == 1500)
(f.col('cosit') == 2010820)|(f.col('cosit') == 201081)|(f.col('cosit') == 1013)
|(f.col('cosit') == 1501)|(f.col('cosit') == 20021)|(f.col('cosit') == 1502)
|(f.col('cosit') == 1508)|(f.col('cosit') == 20047)|(f.col('cosit') == 1503)
|(f.col('cosit') == 1070)|(f.col('cosit') == 1509)|(f.col('cosit') == 1504)
|(f.col('cosit') == 1505)|(f.col('cosit') == 1506)|(f.col('cosit') == 1507)
|(f.col('cosit') == 15010)|(f.col('cosit') == 15011)|(f.col('cosit') == 15012)
|(f.col('cosit') == 1113))

package_count2 = package_count.groupBy("cosit").avg("speed")
package_count2.show()
```

```
+------+------------------+
| cosit|        avg(speed)|
+------+------------------+
|  1507| 99.00845921450151|
| 20021| 92.31461057418989|
|  1500|  79.3267028425403|
|201081| 51.20505548091755|
|  1506| 90.89604001118734|
|  1505| 85.31515519064502|
|  1504| 84.48510440700593|
| 15010| 97.81252274326192|
| 20047| 68.61135818156149|
|  1113|  89.9580108677754|
| 15012| 93.78990503959967|
|  1509| 78.43267256357605|
|  1502| 87.70600667675384|
| 15011| 97.19775581634754|
|  1501| 87.25861183179634|
|  1014|  77.7072657072657|
|  1070|63.892708265485574|
|  1503|  81.0891341051616|
|  1508| 82.67599094114848|
|  1012|  78.5138596978229|
+------+------------------+
only showing top 20 rows
```

QUESTION 6



```
import pyspark.sql.functions as f
from pyspark.sql.functions import *

package_count = vehicle_counter_DF.filter((f.col('classname') == "HGV_RIG") | (f.col('classname') == "HGV_ART"))
package_count2 = package_count.groupBy('cosit').count().sort("count", ascending = False).show(10)
```

```
+------+-----+
| cosit|count|
+------+-----+
|   997|11986|
|  1508| 9630|
|  1015| 9462|
|200723| 8977|
|  1502| 7922|
|  1501| 7145|
|  1503| 6938|
|  1071| 6259|
|  1070| 5850|
|  1073| 5846|
+------+-----+
only showing top 10 rows
```