Data Science – Learning Management Questions

**Week Two – Question #1 (Data Wrangling Subway Data attached )**

import pandas

import pandasql

def num\_rainy\_days(filename):

'''

This function should run a SQL query on a dataframe of

weather data. The SQL query should return one column and

one row - a count of the number of days in the dataframewhere

the rain column is equal to 1 (i.e., the number of days it

rained). The dataframe will be titled 'weather\_data'. You'll

need to provide the SQL query. You might find SQL's count function

useful for this exercise. **You can read more about it here:**

https://dev.mysql.com/doc/refman/5.1/en/counting-rows.html

You might also find that interpreting numbers as integers or floats may not

work initially. In order to get around this issue, it may be useful to cast

these numbers as integers. This can be done by writing cast(column as integer).

So for example, if we wanted to cast the maxtempi column as an integer, we would actually

write something like where cast(maxtempi as integer) = 76, as opposed to simply

where maxtempi = 76.

You can see the weather data that we are passing in below:

**https://s3.amazonaws.com/content.udacity-data.com/courses/ud359/weather\_underground.csv**

'''

weather\_data = pandas.read\_csv(filename)

q="""

select count(\*) from weather\_data where rain = 1;

"""

#Execute your SQL command against the pandas frame

rainy\_days = pandasql.sqldf(q.lower(), locals())

return rainy\_days

**Week Two – Question #2 (weather Underground data attached)**

import pandas

import pandasql

def max\_temp\_aggregate\_by\_fog(filename):

'''

This function should run a SQL query on a dataframe of

weather data. The SQL query should return two columns and

two rows - whether it was foggy or not (0 or 1) and the max

maxtempi for that fog value (i.e., the maximum max temperature

for both foggy and non-foggy days). The dataframe will be

titled 'weather\_data'. You'll need to provide the SQL query.

You might also find that interpreting numbers as integers or floats may not

work initially. In order to get around this issue, it may be useful to cast

these numbers as integers. This can be done by writing cast(column as integer).

So for example, if we wanted to cast the maxtempi column as an integer, we would actually

write something like where cast(maxtempi as integer) = 76, as opposed to simply

where maxtempi = 76.

You can see the weather data that we are passing in below:

https://s3.amazonaws.com/content.udacity-data.com/courses/ud359/weather\_underground.csv

'''

weather\_data = pandas.read\_csv(filename)

q = """

select fog, max(cast (maxtempi as integer)) from weather\_data group by fog;

"""

#Execute your SQL command against the pandas frame

foggy\_days = pandasql.sqldf(q.lower(), locals())

return foggy\_days

**Week Two – Question #3 (Weather Data Set attached)**

import pandas

import pandasql

def avg\_weekend\_temperature(filename):

'''

This function should run a SQL query on a dataframe of

weather data. The SQL query should return one column and

one row - the average meantempi on days that are a Saturday

or Sunday (i.e., the the average mean temperature on weekends).

The dataframe will be titled 'weather\_data' and you can access

the date in the dataframe via the 'date' column.

You'll need to provide the SQL query.

You might also find that interpreting numbers as integers or floats may not

work initially. In order to get around this issue, it may be useful to cast

these numbers as integers. This can be done by writing cast(column as integer).

So for example, if we wanted to cast the maxtempi column as an integer, we would actually

write something like where cast(maxtempi as integer) = 76, as opposed to simply

where maxtempi = 76.

Also, you can convert dates to days of the week via the 'strftime' keyword in SQL.

For example, cast (strftime('%w', date) as integer) will return 0 if the date

is a Sunday or 6 if the date is a Saturday.

You can see the weather data that we are passing in below:

https://s3.amazonaws.com/content.udacity-data.com/courses/ud359/weather\_underground.csv

'''

weather\_data = pandas.read\_csv(filename)

q = """

|  |  |
| --- | --- |
| SELECT avg(cast (meantempi as integer))FROM weather\_data WHERE cast (strftime('%w', date) as integer)=0 or cast (strftime('%w', date) as integer)=6 | |
|  |  |
|  |  |

"""

#Execute your SQL command against the pandas frame

mean\_temp\_weekends = pandasql.sqldf(q.lower(), locals())

return mean\_temp\_weekends

**Week Two – Question #4 (Weather Data Set attached)**

import pandas

import pandasql

def avg\_min\_temperature(filename):

'''

This function should run a SQL query on a dataframe of

weather data. More specifically you want to find the average

minimum temperature (mintempi column of the weather dataframe) on

rainy days where the minimum temperature is greater than 55 degrees.

You might also find that interpreting numbers as integers or floats may not

work initially. In order to get around this issue, it may be useful to cast

these numbers as integers. This can be done by writing cast(column as integer).

So for example, if we wanted to cast the maxtempi column as an integer, we would actually

write something like where cast(maxtempi as integer) = 76, as opposed to simply

where maxtempi = 76.

You can see the weather data that we are passing in below:

https://s3.amazonaws.com/content.udacity-data.com/courses/ud359/weather\_underground.csv

'''

weather\_data = pandas.read\_csv(filename)

q = """

select avg(cast (mintempi as integer)) from weather\_data where mintempi > 55 and rain = 1

"""

#Execute your SQL command against the pandas frame

avg\_min\_temp\_rainy = pandasql.sqldf(q.lower(), locals())

return avg\_min\_temp\_rainy