```python
In [7]:  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

         if __name__ == "__main__":
             filename = 'weather_underground.csv'
             output=num_rainy_days(filename)
             print(output)
```

```
   count(*)
0        10
```

```python
In [4]:  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

         if __name__ == "__main__":
             filename = 'weather_underground.csv'
             output=max_temp_aggregate_by_fog(filename)
             print(output)
```

```
   fog  max(cast (maxtempi as integer))
0    0                               86
1    1                               81
```

```
In [5]: 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
            """
            #Execute your SQL command against the pandas frame
            mean_temp_weekends = pandasql.sqldf(q.lower(), locals())
            return mean_temp_weekends

        if __name__ == "__main__":
            filename = 'weather_underground.csv'
            output=avg_weekend_temperature(filename)
            print(output)
```

```
   avg(cast (meantempi as integer))
0                         65.111111
```

```
In [6]: 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

        if __name__ == "__main__":
            filename = 'weather_underground.csv'
            output=avg_weekend_temperature(filename)
            print(output)
```

```
   avg(cast (meantempi as integer))
0                         65.111111
```