

KONSTANTINOS PAGANOPOULOS 01769789

FUNDAMENTALS OF DATABASE TECHNOLOGIES

ASSIGNMENT 4

20/11/2019

INTRODUCTION

In the coursework assignment we are loading and analysing a CSV file. We first launch psql directly from the command line, connect to the database server with psql and make a new database to hold the CSV data. Through commands we will connected to the database we created, we will create a table to store the CSV data and load it to the database table. We will then run some queries on our dataset and finally export the result of the queries to a new exported CSV file. Various screenshots are taken through the whole procedure to help the reader follow easily (zoom in for more detail).

QUESTION 1

We launch psql directly from the command line and connect to the server as follows:

```
Command Prompt - psql -U postgres
C:\Users\fdt>psql
Password for user fdt:
psql: error: could not connect to server: FATAL: password authentication failed for user "fdt"

C:\Users\fdt>psql -U postgres
Password for user postgres:
psql (12.1)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

postgres=#
```

Screenshot 1: We connect to the postgres server.

QUESTION 2

We now download a CSV file from the Web for analysis. We choose to download a CSV file named "ILONDONL28_weather.csv". That file (39,106 lines and 17 columns) contains continuous data with no gaps (every 15 minutes) for the whole annual year of 2015, with weather observations of ILONDONL28 weather station. The station is located near the centre of London (Elev 16 ft, 51.49 °N, 0.07 °W). We edit that file and delete some columns of data ("number of observations per month", "Conditions", "Clouds", "SoftwareType", "DateUTC", "station", "WindDirectionDegrees", "WindSpeedKMH") that seem useless to us. We do not change any of the rest data, but we just change the order of the columns and their names to be in snake case form and more easily understandable. The final CSV file contains 39,106 lines and 10 columns ("Observation_id", "Time", "Temperature", "Humidity", "Dewpoint", "Pressure", "Wind_direction", "Wind_speed_gust", "Daily_rain", "Hourly_precip"). Screenshots of the two CSV files are attached below:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Observatio	Time	Temperatu	DewpointC	Pressureh	FWindDirec	WindDirec	WindSpec	WindSpec	Humidity	HourlyPrec	Conditions	Clouds	dailyrainM	SoftwareTy	DateUTC	station
2	1	1/1/2015 0:00	4.6	2.9	1031.7	West	273	0	1.6	89	0			0	WeatherCa	1/1/2015 0:00	ILONDONL28
3	2	1/1/2015 0:12	4.5	2.8	1031.4	WNW	291	0	1.6	89	0			0	WeatherCa	1/1/2015 0:12	ILONDONL28
4	3	1/1/2015 0:27	4.5	2.8	1031	SW	229	0	1.6	89	0			0	WeatherCa	1/1/2015 0:27	ILONDONL28
5	4	1/1/2015 0:42	4.8	3.2	1031.7	West	281	0	4.8	89	0			0	WeatherCa	1/1/2015 0:42	ILONDONL28
6	5	1/1/2015 0:57	5.2	3.5	1031.4	NW	309	0	1.6	89	0			0	WeatherCa	1/1/2015 0:57	ILONDONL28
7	6	1/1/2015 1:12	5.4	3.6	1031.4	NW	317	0	1.6	88	0			0	WeatherCa	1/1/2015 1:12	ILONDONL28
8	7	1/1/2015 1:27	5.6	3.9	1031	NE	36	0	0	89	0			0	WeatherCa	1/1/2015 1:27	ILONDONL28
9	8	1/1/2015 1:42	5.6	3.9	1030.7	WNW	293	0	0	89	0			0	WeatherCa	1/1/2015 1:42	ILONDONL28
10	9	1/1/2015 1:57	5.7	4	1030.7	South	181	0	1.6	89	0			0	WeatherCa	1/1/2015 1:57	ILONDONL28
11	10	1/1/2015 2:12	5.9	4.4	1030.7	East	95	0	4.8	90	0			0	WeatherCa	1/1/2015 2:12	ILONDONL28
12	11	1/1/2015 2:27	6	4.5	1030.7	ENE	61	0	1.6	90	0			0	WeatherCa	1/1/2015 2:27	ILONDONL28
13	12	1/1/2015 2:42	6.1	4.6	1031	South	188	0	1.6	90	0			0	WeatherCa	1/1/2015 2:42	ILONDONL28
14	13	1/1/2015 2:57	6.2	4.7	1031	ESE	118	0	0	90	0			0	WeatherCa	1/1/2015 2:57	ILONDONL28
15	14	1/1/2015 3:12	6.2	4.9	1030.7	East	91	0	0	91	0			0	WeatherCa	1/1/2015 3:12	ILONDONL28
16	15	1/1/2015 3:27	6.2	5	1030.7	ESE	116	0	8	92	0			0	WeatherCa	1/1/2015 3:27	ILONDONL28
17	16	1/1/2015 3:42	6.4	5	1030.4	SW	228	3.2	6.4	91	0			0	WeatherCa	1/1/2015 3:42	ILONDONL28
18	17	1/1/2015 3:57	6.6	5.4	1030.4	SSW	204	0	1.6	92	0			0	WeatherCa	1/1/2015 3:57	ILONDONL28
19	18	1/1/2015 4:12	6.8	5.4	1030.4	SSW	209	0	3.2	91	0			0	WeatherCa	1/1/2015 4:12	ILONDONL28
20	19	1/1/2015 4:27	6.9	5.6	1030	WSW	241	0	1.6	91	0			0	WeatherCa	1/1/2015 4:27	ILONDONL28
21	20	1/1/2015 4:42	7.2	5.8	1030	East	94	1.6	3.2	91	0			0	WeatherCa	1/1/2015 4:42	ILONDONL28
22	21	1/1/2015 4:57	7.3	5.9	1030	ESE	105	0	1.6	91	0			0	WeatherCa	1/1/2015 4:57	ILONDONL28
23	22	1/1/2015 5:12	7.4	6.2	1029.7	SSE	163	0	9.7	92	0			0	WeatherCa	1/1/2015 5:12	ILONDONL28
24	23	1/1/2015 5:27	7.5	6.3	1029.7	WSW	237	0	1.6	92	0			0	WeatherCa	1/1/2015 5:27	ILONDONL28
25	24	1/1/2015 5:42	7.5	6.1	1029.3	SSW	210	0	0	91	0			0	WeatherCa	1/1/2015 5:42	ILONDONL28
26	25	1/1/2015 5:57	7.5	6.3	1029.3	West	264	0	1.6	92	0			0	WeatherCa	1/1/2015 5:57	ILONDONL28

Screenshot 2: first 25 lines of “ILONDONL28_weather.csv” file.

	A	B	C	D	E	F	G	H	I	J
1	Observation_id	Time	Temperature	Humidity	Dewpoint	Pressure	Wind_direction	Wind_speed_gust	Daily_rain	Hourly_precip
2	1	1/1/2015 0:00	4.6	89	2.9	1031.7	West	1.6	0	0
3	2	1/1/2015 0:12	4.5	89	2.8	1031.4	WNW	1.6	0	0
4	3	1/1/2015 0:27	4.5	89	2.8	1031	SW	1.6	0	0
5	4	1/1/2015 0:42	4.8	89	3.2	1031.7	West	4.8	0	0
6	5	1/1/2015 0:57	5.2	89	3.5	1031.4	NW	1.6	0	0
7	6	1/1/2015 1:12	5.4	88	3.6	1031.4	NW	1.6	0	0
8	7	1/1/2015 1:27	5.6	89	3.9	1031	NE	0	0	0
9	8	1/1/2015 1:42	5.6	89	3.9	1030.7	WNW	0	0	0
10	9	1/1/2015 1:57	5.7	89	4	1030.7	South	1.6	0	0
11	10	1/1/2015 2:12	5.9	90	4.4	1030.7	East	4.8	0	0
12	11	1/1/2015 2:27	6	90	4.5	1030.7	ENE	1.6	0	0
13	12	1/1/2015 2:42	6.1	90	4.6	1031	South	1.6	0	0
14	13	1/1/2015 2:57	6.2	90	4.7	1031	ESE	0	0	0
15	14	1/1/2015 3:12	6.2	91	4.9	1030.7	East	0	0	0
16	15	1/1/2015 3:27	6.2	92	5	1030.7	ESE	8	0	0
17	16	1/1/2015 3:42	6.4	91	5	1030.4	SW	6.4	0	0
18	17	1/1/2015 3:57	6.6	92	5.4	1030.4	SSW	1.6	0	0
19	18	1/1/2015 4:12	6.8	91	5.4	1030.4	SSW	3.2	0	0
20	19	1/1/2015 4:27	6.9	91	5.6	1030	WSW	1.6	0	0
21	20	1/1/2015 4:42	7.2	91	5.8	1030	East	3.2	0	0
22	21	1/1/2015 4:57	7.3	91	5.9	1030	ESE	1.6	0	0
23	22	1/1/2015 5:12	7.4	92	6.2	1029.7	SSE	9.7	0	0
24	23	1/1/2015 5:27	7.5	92	6.3	1029.7	WSW	1.6	0	0
25	24	1/1/2015 5:42	7.5	91	6.1	1029.3	SSW	0	0	0
26	25	1/1/2015 5:57	7.5	92	6.3	1029.3	West	1.6	0	0

Screenshot 3: first 25 lines of “ILONDONL28_weather_edit.csv” file.

Column “Conditions” was removed because can be replaced by “dailyrainMM” and “HourlyPrecipMM”. The “Clouds” column was removed since we are not particularly interested in cloud coverage. The column “SoftwareType” seems useless for our analysis, “DateUTC” can be replaced by “Time” as London time equals UTC +0, the column “station” is redundant since we only have data for one station, column “WindDirectionDegrees” can be replaced by “WindDirection” and finally column “WindSpeedKMH” is not so frequently used in weather as “WindSpeedGustKMH” especially in areas with not so high mean wind speed.

QUESTION 3

We are connected to the database server, so continuing to the same command prompt window, we first type the command “\l” and press “enter” to see the list of databases that we have. We then type the command “CREATE DATABASE ILONDONL28;” and press “enter”. To check that the new database named “ilondonl28” was created we type again the command “\l” and press “enter”. Now the list of databases also contains the database named “ilondonl28”, so our new database was successfully created.

```
Command Prompt - psql -U postgres

C:\Users\fdt>psql
Password for user fdt:
psql: error: could not connect to server: FATAL: password authentication failed for user "fdt"

C:\Users\fdt>psql -U postgres
Password for user postgres:
psql (12.1)
WARNING: Console code page (437) differs from Windows code page (1252)
        8-bit characters might not work correctly. See psql reference
        page "Notes for Windows users" for details.
Type "help" for help.

postgres=# \l

          List of databases
  Name | Owner | Encoding | Collate | Ctype | Access privileges
-----+-----+-----+-----+-----+-----
 postgres | postgres | UTF8 | English_United States.1252 | English_United States.1252 | 
 template0 | postgres | UTF8 | English_United States.1252 | English_United States.1252 | =c/postgres +
 template1 | postgres | UTF8 | English_United States.1252 | English_United States.1252 | =c/postgres +
 postgres=CTc/postgres
(3 rows)

postgres=# CREATE DATABASE ILONDONL28;
CREATE DATABASE
postgres=# \l

          List of databases
  Name | Owner | Encoding | Collate | Ctype | Access privileges
-----+-----+-----+-----+-----+-----
 ilondonl28 | postgres | UTF8 | English_United States.1252 | English_United States.1252 | 
 postgres | postgres | UTF8 | English_United States.1252 | English_United States.1252 | 
 template0 | postgres | UTF8 | English_United States.1252 | English_United States.1252 | =c/postgres +
 template1 | postgres | UTF8 | English_United States.1252 | English_United States.1252 | =c/postgres +
 postgres=CTc/postgres
(4 rows)

postgres=#
```

Screenshot 4: We make a new database to hold the CSV data.

QUESTION 4

We continue at the same command prompt window and now try to connect to the database (“ilondonl28”) we just created. We use the “\c” command inside psql. Hence, we type “\c ilondonl28” and press “enter”. We are now connected to our database.

```

Command Prompt - psql -U postgres

C:\Users\fdt>psql
Password for user fdt:
psql: error: could not connect to server: FATAL: password authentication failed for user "fdt"

C:\Users\fdt>psql -U postgres
Password for user postgres:
psql (12.1)
WARNING: Console code page (437) differs from Windows code page (1252)
        8-bit characters might not work correctly. See psql reference
        page "Notes for Windows users" for details.
Type "help" for help.

postgres=# \l

```

Name	Owner	Encoding	Collate	Ctype	Access privileges
postgres	postgres	UTF8	English_United States.1252	English_United States.1252	
template0	postgres	UTF8	English_United States.1252	English_United States.1252	=c/postgres + postgres=CTc/postgres
template1	postgres	UTF8	English_United States.1252	English_United States.1252	=c/postgres + postgres=CTc/postgres

```

(3 rows)

postgres=# CREATE DATABASE ILONDONL28;
CREATE DATABASE
postgres=# \l

```

Name	Owner	Encoding	Collate	Ctype	Access privileges
ilondonl28	postgres	UTF8	English_United States.1252	English_United States.1252	
postgres	postgres	UTF8	English_United States.1252	English_United States.1252	
template0	postgres	UTF8	English_United States.1252	English_United States.1252	=c/postgres + postgres=CTc/postgres
template1	postgres	UTF8	English_United States.1252	English_United States.1252	=c/postgres + postgres=CTc/postgres

```

(4 rows)

postgres=# \c ilondonl28
You are now connected to database "ilondonl28" as user "postgres".
ilondonl28=#

```

Screenshot 5: We connect to the database “ilondonl28”.

QUESTION 5

Now we want to create a table inside the database “ilondonl28”. We choose the following data types for the table columns:

Observation_id: INT
 Time: TIMESTAMP
 Temperature: REAL
 Humidity: INT
 Dewpoint: REAL
 Pressure: REAL
 Wind_direction: TEXT
 Wind_speed_gust: REAL
 Daily_rain: REAL
 Hourly_precip: REAL

The “CREATE TABLE” statement is the following:

```
CREATE TABLE Weather_data (  
    Observation_id INT PRIMARY KEY,  
    Time TIMESTAMP,  
    Temperature REAL,  
    Humidity INT,  
    Dewpoint REAL,  
    Pressure REAL,  
    Wind_direction TEXT,  
    Wind_speed_gust REAL,  
    Daily_rain REAL,  
    Hourly_precip REAL  
);
```

QUESTION 6

We then run the previous “CREATE TABLE” statement and have the following result:

```
ilondonl28=# CREATE TABLE Weather_data (  
ilondonl28=# Observation_id INT PRIMARY KEY,  
ilondonl28=# Time TIMESTAMP,  
ilondonl28=# Temperature REAL,  
ilondonl28=# Humidity INT,  
ilondonl28=# Dewpoint REAL,  
ilondonl28=# Pressure REAL,  
ilondonl28=# Wind_direction TEXT,  
ilondonl28=# Wind_speed_gust REAL,  
ilondonl28=# Daily_rain REAL,  
ilondonl28=# Hourly_precip REAL  
ilondonl28=# );  
CREATE TABLE  
ilondonl28=#
```

Screenshot 6: We create the table “Weather_data”.

We can see that the table “Weather_data” has been successfully created through the “\dt” command. We press “\dt” then “enter” and see the following result.

```
ilondonl28=# \dt  
          List of relations  
Schema |      Name      | Type | Owner  
-----+-----+-----+-----  
public | weather_data | table | postgres  
(1 row)  
  
ilondonl28=#
```

Screenshot 7: We see that the table “Weather_data” has been created.

QUESTION 7

We are now going to load the CSV file into the database table we created, using psql. The path of the CSV file is the following: 'C:\Windows\Temp\ILONDONL28_weather_edit.csv'

We now use the “COPY” command as follows:

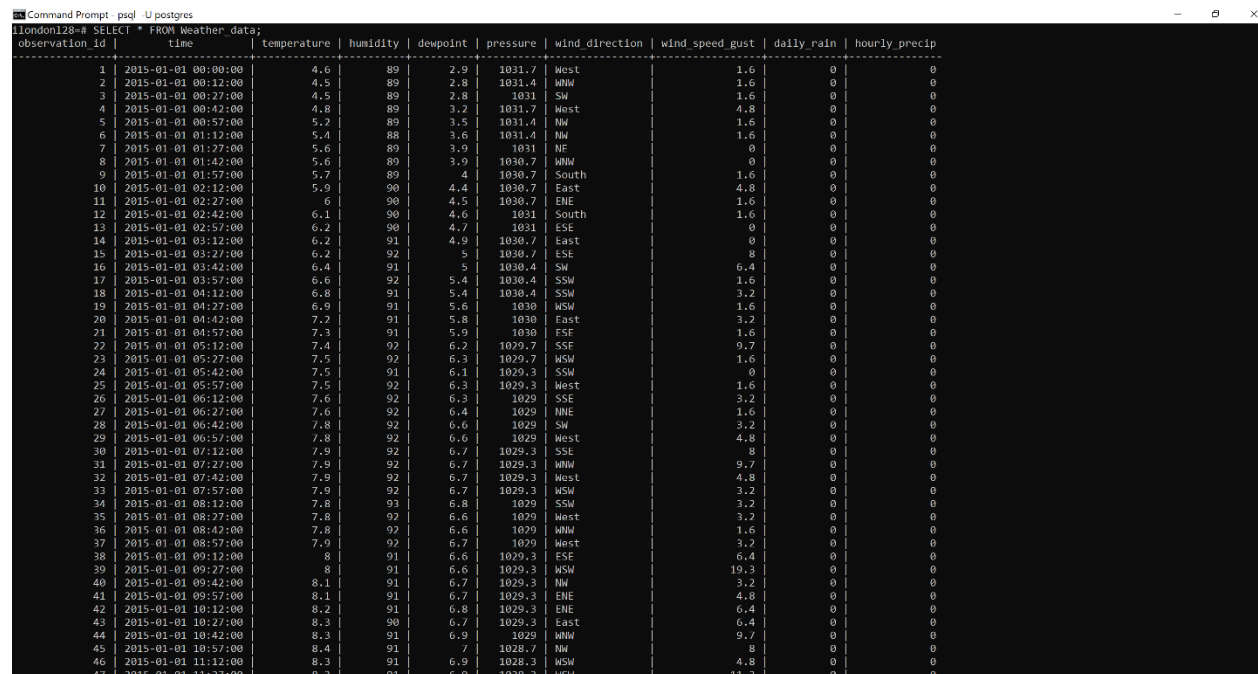
```
\copy Weather_data FROM 'C:\Windows\Temp\ILONDONL28_weather_edit.csv' DELIMITER ',' CSV HEADER;
```

```
ilondonl28=# \copy Weather_data FROM 'C:\Windows\Temp\ILONDONL28_weather_edit.csv' DELIMITER ',' CSV HEADER;
COPY 39106
ilondonl28=#
```

Screenshot 8: We see that all the 39,106 lines have been successfully copied.

QUESTION 8

We now select everything from the table “Weather_data” through the command “SELECT * FROM Weather_data;” to check that it has been imported as follows:



The screenshot shows a PostgreSQL command prompt window with the following content:

```
Command Prompt - psql - U postgres
ilondonl28=# SELECT * FROM Weather_data;
 observation_id | time           | temperature | humidity | dewpoint | pressure | wind_direction | wind_speed_gust | daily_rain | hourly_precip
-----
```

observation_id	time	temperature	humidity	dewpoint	pressure	wind_direction	wind_speed_gust	daily_rain	hourly_precip
1	2015-01-01 00:00:00	4.6	89	2.9	1031.7	West	1.6	0	0
2	2015-01-01 00:12:00	4.5	89	2.8	1031.4	WNW	1.6	0	0
3	2015-01-01 00:27:00	4.5	89	2.8	1031	SW	1.6	0	0
4	2015-01-01 00:42:00	4.8	89	3.2	1031.7	West	4.8	0	0
5	2015-01-01 00:57:00	5.2	89	3.5	1031.4	NW	1.6	0	0
6	2015-01-01 01:12:00	5.4	88	3.6	1031.4	NW	1.6	0	0
7	2015-01-01 01:27:00	5.6	89	3.9	1031	NF	0	0	0
8	2015-01-01 01:42:00	5.6	89	3.9	1030.7	WNW	0	0	0
9	2015-01-01 01:57:00	5.7	89	4	1030.7	South	1.6	0	0
10	2015-01-01 02:12:00	5.9	90	4.4	1030.7	East	4.8	0	0
11	2015-01-01 02:27:00	6	90	4.5	1030.7	ENE	1.6	0	0
12	2015-01-01 02:42:00	6.1	90	4.6	1031	South	1.6	0	0
13	2015-01-01 02:57:00	6.2	90	4.7	1031	ESE	0	0	0
14	2015-01-01 03:12:00	6.2	91	4.9	1030.7	East	0	0	0
15	2015-01-01 03:27:00	6.2	92	5	1030.7	ESE	8	0	0
16	2015-01-01 03:42:00	6.4	91	5	1030.4	SW	6.4	0	0
17	2015-01-01 03:57:00	6.6	92	5.4	1030.4	SSW	1.6	0	0
18	2015-01-01 04:12:00	6.8	91	5.4	1030.4	SSW	3.2	0	0
19	2015-01-01 04:27:00	6.9	91	5.6	1030	WSW	1.6	0	0
20	2015-01-01 04:42:00	7.2	91	5.8	1030	East	3.2	0	0
21	2015-01-01 04:57:00	7.3	91	5.9	1030	ESE	1.6	0	0
22	2015-01-01 05:12:00	7.4	92	6.2	1029.7	SSE	9.7	0	0
23	2015-01-01 05:27:00	7.5	92	6.3	1029.7	WSW	1.6	0	0
24	2015-01-01 05:42:00	7.5	91	6.1	1029.3	SSW	0	0	0
25	2015-01-01 05:57:00	7.5	92	6.3	1029.3	West	1.6	0	0
26	2015-01-01 06:12:00	7.6	92	6.3	1029	SSE	3.2	0	0
27	2015-01-01 06:27:00	7.6	92	6.4	1029	NNE	1.6	0	0
28	2015-01-01 06:42:00	7.8	92	6.6	1029	SW	3.2	0	0
29	2015-01-01 06:57:00	7.8	92	6.6	1029	West	4.8	0	0
30	2015-01-01 07:12:00	7.9	92	6.7	1029.3	SSE	8	0	0
31	2015-01-01 07:27:00	7.9	92	6.7	1029.3	WNW	9.7	0	0
32	2015-01-01 07:42:00	7.9	92	6.7	1029.3	West	4.8	0	0
33	2015-01-01 07:57:00	7.9	92	6.7	1029.3	WSW	3.2	0	0
34	2015-01-01 08:12:00	7.8	93	6.8	1029	SSW	3.2	0	0
35	2015-01-01 08:27:00	7.8	92	6.6	1029	West	3.2	0	0
36	2015-01-01 08:42:00	7.8	92	6.6	1029	WNW	1.6	0	0
37	2015-01-01 08:57:00	7.9	92	6.7	1029	West	3.2	0	0
38	2015-01-01 09:12:00	8	91	6.6	1029.3	ESE	6.4	0	0
39	2015-01-01 09:27:00	8	91	6.6	1029.3	WSW	19.3	0	0
40	2015-01-01 09:42:00	8.1	91	6.7	1029.3	NW	3.2	0	0
41	2015-01-01 09:57:00	8.1	91	6.7	1029.3	ENE	4.8	0	0
42	2015-01-01 10:12:00	8.2	91	6.8	1029.3	ENE	6.4	0	0
43	2015-01-01 10:27:00	8.3	90	6.7	1029.3	East	6.4	0	0
44	2015-01-01 10:42:00	8.3	91	6.9	1029	WNW	9.7	0	0
45	2015-01-01 10:57:00	8.4	91	7	1028.7	NW	8	0	0
46	2015-01-01 11:12:00	8.3	91	6.9	1028.3	WSW	4.8	0	0
47	2015-01-01 11:27:00	8.3	91	6.9	1028.3	WSW	11.3	0	0

Screenshot 9: We see that all the 39,106 lines of the table “Weather_data” have been successfully imported.

QUESTION 9

The first query we devise contains the keyword “WHERE” and selects the values from “Hourly_precip” column of “Weather_data” table, that have an hourly precipitation (rain rate) greater than 20, as well as their respective “Wind_direction” values. The reason why we chose that query, is to show a correlation between high rain rates and wind direction.

The query is the following:

```
SELECT Wind_direction, Hourly_precip FROM Weather_data
WHERE Hourly_precip >= 20
```

```
ilondon128=#
ilondon128=# SELECT Wind_direction, Hourly_precip FROM Weather_data
ilondon128=# WHERE Hourly_precip >= 20
ilondon128=# ;
 wind_direction | hourly_precip
-----
 SE              |          62
 SE              |         21.6
 NW              |         28.7
 NNW             |         95.2
 ESE             |          30
 SE              |         30.2
(6 rows)
```

Screenshot 10: We see the result of the first query

From the above results we can conclude that Southeast surface winds (and wind directions close to them) are highly correlated (4 out of 6 cases) with heavy rainfalls in central London (especially for year 2015). We can also see that Northwest winds (and wind directions close to them) are also correlated (2 out of 6 cases) with high rain rates.

The second query we devise contains the keyword “GROUP BY” and that of “WHERE” and selects the column “Wind_direction” from table “Weather_data”, as long as with the aggregate function “COUNT(*)”. We use that aggregate function to count the total number of observations where a rainfall occurs (“WHERE Hourly_precip” is not equal to zero) and group them by wind direction. We also sort the results by descending number of rainy days. The reason why we chose that query, is to show a correlation between rainfall and wind direction.

The query is the following:

```
SELECT Wind_direction, count(*) FROM Weather_data
WHERE Hourly_precip != 0
GROUP BY Wind_direction
ORDER BY count(*) DESC
```

```
ilondon128=# SELECT wind_direction, count(*) FROM weather_data
ilondon128=# WHERE hourly_precip != 0
ilondon128=# GROUP BY wind_direction
ilondon128=# ORDER BY count(*) DESC
ilondon128=# ;
 wind_direction | count
-----
 SE              |    135
 North           |    104
 NNE             |    102
 ESE             |     97
 East            |     88
 NNW             |     72
 SSE             |     64
 NW              |     57
 SW              |     46
 NE              |     43
 SSW             |     39
 WNW             |     39
 WSW             |     37
 ENE             |     34
 South           |     33
 West            |     30
(16 rows)
```

Screenshot 11: We see the result of the second query

From the above results we can conclude that Southeast surface winds (and wind directions close to them) are highly correlated with rainfalls (135 out of 1,020 days of precipitation) in central London (especially for year 2015). We can also see that North winds (and wind directions close to them) are also correlated with many rainy days. If we combine the first two queries, we can say that Southeast winds are stronger connected with heavy rainfalls, whereas north with light ones.

The third query we devise contains the keyword “OVER” and selects the column “Temperature” and that of “Wind_direction” from table “Weather_data”, as well as calculates the mean temperature according to wind direction through the aggregate function “AVG(Temperature)” partitioned by (“Wind_direction”). In other words, we use the command “AVG(Temperature) OVER (PARTITION BY Wind_direction) AS Mean_temp_per_wind_direction” so as to create a new column in table “Weather_data” called “Mean_temp_per_wind_direction” and store there the mean temperature according to wind direction.

The query is the following:

```
SELECT Temperature, Wind_direction,
AVG(Temperature) OVER (PARTITION BY Wind_direction) AS Mean_temp_per_wind_direction
FROM Weather_data
```

```
ilondon128=# SELECT Temperature, Wind_direction,
ilondon128=# AVG(Temperature) OVER (PARTITION BY Wind_direction) AS Mean_temp_per_wind_direction
ilondon128=# FROM Weather_data
ilondon128=# ;
```

temperature	wind_direction	mean_temp_per_wind_direction
18.4	East	12.249283033201717
17.8	East	12.249283033201717
15.2	East	12.249283033201717
16.2	East	12.249283033201717
7.3	East	12.249283033201717
17.4	East	12.249283033201717
15.8	East	12.249283033201717
12.1	East	12.249283033201717
6.1	East	12.249283033201717
15.9	East	12.249283033201717
11.1	East	12.249283033201717
18.3	East	12.249283033201717
12.6	East	12.249283033201717
5.2	East	12.249283033201717
5.4	East	12.249283033201717
3.6	East	12.249283033201717
20.8	East	12.249283033201717
2.4	East	12.249283033201717
16	East	12.249283033201717
2.1	East	12.249283033201717
1.9	East	12.249283033201717
10	East	12.249283033201717
0.7	East	12.249283033201717
19.3	East	12.249283033201717
9	East	12.249283033201717
9.2	East	12.249283033201717
3.4	East	12.249283033201717
9.2	East	12.249283033201717
9	East	12.249283033201717
13.3	East	12.249283033201717
8.3	East	12.249283033201717
14.6	East	12.249283033201717
6.3	East	12.249283033201717
16.9	East	12.249283033201717
14.1	East	12.249283033201717
4.6	East	12.249283033201717
17.9	East	12.249283033201717
17.6	East	12.249283033201717
8.1	East	12.249283033201717
15.1	East	12.249283033201717
8.7	East	12.249283033201717
3.9	East	12.249283033201717
6.4	East	12.249283033201717
7.8	East	12.249283033201717

Screenshot 12: We see the result of the third query.

We can now see despite all temperature values, the average temperature according to all wind directions. From the above results we can conclude that “West” winds are correlated with lower temperatures. A possible explanation for that phenomena is because west winds usually occur right after a “cold front”, which drops the temperature.

QUESTION 10

We devise a new query that returns a table as a result rather than a single value. The query contains the keyword “WHERE” and it selects all the rows of the table “Weather_data” that have a temperature greater than 30 degrees Celsius (column “Temperature” > 30). The new table has 79 rows.

The query is the following:

```
CREATE TABLE Heat_waves
AS (SELECT *
FROM Weather_data
WHERE Temperature > 30);
```

```
ilondonl28=# CREATE TABLE Heat_waves
ilondonl28=# AS (SELECT *
ilondonl28=# FROM Weather_data
ilondonl28=# WHERE Temperature > 30);
SELECT 79
ilondonl28=# SELECT * FROM Heat_waves;
```

observation_id	time	temperature	humidity	dewpoint	pressure	wind_direction	wind_speed_gust	daily_rain	hourly_precip
20617	2015-06-30 13:52:00	30.1	24	7.3	1017.2	SSE	0	0	0
20618	2015-06-30 14:07:00	30.1	26	8.4	1017.2	ESE	4.8	0	0
20619	2015-06-30 14:22:00	30.2	27	9.1	1016.8	SE	3.2	0	0
20620	2015-06-30 14:37:00	30.2	29	10.2	1016.8	NNW	3.2	0	0
20621	2015-06-30 14:52:00	30.4	30	10.9	1016.5	East	1.6	0	0
20622	2015-06-30 15:07:00	30.7	31	11.6	1016.5	SSW	4.8	0	0
20623	2015-06-30 15:22:00	30.9	31	11.8	1016.1	SE	3.2	0	0
20624	2015-06-30 15:37:00	31.2	31	12.1	1016.1	ESE	1.6	0	0
20625	2015-06-30 15:52:00	31.1	30	11.4	1016.1	ESE	4.8	0	0
20626	2015-06-30 16:07:00	31.1	28	10.4	1015.8	SE	4.8	0	0
20627	2015-06-30 16:22:00	31.1	31	11.9	1015.8	SE	3.2	0	0
20628	2015-06-30 16:37:00	31.3	31	12.1	1015.8	South	3.2	0	0
20629	2015-06-30 16:52:00	31.3	30	11.7	1015.5	WNW	6.4	0	0

-- More --

Screenshot 13: We see the result of the “CREATE TABLE AS” query.

QUESTION 11

We now export the table of Question 10 to a CSV file named “ILONDONL28_heat_waves.csv” through command prompt using the “\copy” command as follows:

```
\copy Heat_waves TO 'C:\Windows\Temp\ILONDONL28_heat_waves.csv' DELIMITER ',' CSV
HEADER;
```

```
ilondonl28=#
ilondonl28=#
ilondonl28=# \copy Heat_waves TO 'C:\Windows\Temp\ILONDONL28_heat_waves.csv' DELIMITER ',' CSV HEADER;
COPY 79
ilondonl28=#
```

Screenshot 14: We see that all the 79 lines have been successfully exported.

Then we open the CSV called file “ILONDONL28_heat_waves.csv” in notepad text editor and to check if the export was successful and we see the attached result. The new CSV file contains 79 rows and 10 columns. The new news CSV file has enough data for a further analysis. We can see that in column “Temperature” all the values are greater than 30.

The new CSV file is the following:

```

observation_id,time,temperature,humidity,dewpoint,pressure,wind_direction,wind_speed_gust,daily_rain,hourly_precip
20617,2015-06-30 13:52:00,30.1,24,7.3,1017.2,SSE,0,0,0
20618,2015-06-30 14:07:00,30.1,26,8.4,1017.2,ESE,4.8,0,0
20619,2015-06-30 14:22:00,30.2,27,9.1,1016.8,SE,3.2,0,0
20620,2015-06-30 14:37:00,30.2,29,10.2,1016.8,NNW,3.2,0,0
20621,2015-06-30 14:52:00,30.4,30,10.9,1016.5,East,1.6,0,0
20622,2015-06-30 15:07:00,30.7,31,11.6,1016.5,SSW,4.8,0,0
20623,2015-06-30 15:22:00,30.9,31,11.8,1016.1,SE,3.2,0,0
20624,2015-06-30 15:37:00,31.2,31,12.1,1016.1,ESE,1.6,0,0
20625,2015-06-30 15:52:00,31.1,30,11.4,1016.1,ESE,4.8,0,0
20626,2015-06-30 16:07:00,31.1,28,10.4,1015.8,SE,4.8,0,0
20627,2015-06-30 16:22:00,31.1,31,11.9,1015.8,SE,3.2,0,0
20628,2015-06-30 16:37:00,31.3,31,12.1,1015.8,South,3.2,0,0
20629,2015-06-30 16:52:00,31.3,30,11.7,1015.5,WNW,6.4,0,0
20630,2015-06-30 17:07:00,31.5,30,11.8,1015.5,NW,4.8,0,0
20631,2015-06-30 17:22:00,31.3,31,12.2,1015.5,NE,3.2,0,0
20632,2015-06-30 17:37:00,31.3,30,11.7,1015.1,SE,3.2,0,0
20633,2015-06-30 17:52:00,31.4,28,10.7,1015.1,East,1.6,0,0
20634,2015-06-30 18:07:00,31.2,33,12.9,1015.1,NE,4.8,0,0
20697,2015-07-01 10:07:00,30.3,44,16.7,1010.7,SE,0,0,0
20698,2015-07-01 10:22:00,31.2,42,16.7,1010.7,North,4.8,0,0
20699,2015-07-01 10:37:00,31.9,41,17,1010.7,SSW,0,0,0
20700,2015-07-01 10:52:00,33.2,38,16.9,1010.4,WSW,1.6,0,0
20701,2015-07-01 11:07:00,33.3,36,16.2,1010.4,SSE,4.8,0,0
20702,2015-07-01 11:22:00,32.7,39,16.9,1010.4,South,6.4,0,0
20703,2015-07-01 11:37:00,33.2,38,16.9,1010.4,South,4.8,0,0
20704,2015-07-01 11:52:00,33.2,38,16.9,1010.7,East,3.2,0,0
20705,2015-07-01 12:07:00,32.6,40,17.2,1010.4,SSE,1.6,0,0
20706,2015-07-01 12:22:00,31.9,40,16.6,1010.7,SE,3.2,0,0
20707,2015-07-01 12:37:00,31.3,42,16.8,1010,SSW,1.6,0,0
20708,2015-07-01 12:52:00,31.6,42,17.1,1010.4,NNW,3.2,0,0
20709,2015-07-01 13:07:00,32.1,43,17.9,1010.7,NNW,3.2,0,0
20710,2015-07-01 13:22:00,32.9,41,17.9,1010.4,SE,3.2,0,0
20711,2015-07-01 13:37:00,32.8,40,17.4,1010.4,NE,11.3,0,0
20712,2015-07-01 13:52:00,33.5,41,18.4,1010,East,1.6,0,0
20713,2015-07-01 14:07:00,34.4,37,17.6,1009.7,ESE,1.6,0,0
20714,2015-07-01 14:22:00,35.1,36,17.7,1009.7,WNW,4.8,0,0
20715,2015-07-01 14:37:00,35,36,17.7,1009.7,NE,4.8,0,0
20716,2015-07-01 14:52:00,34.4,36,17.2,1009.7,NNW,3.2,0,0
20717,2015-07-01 15:07:00,33.9,38,17.6,1009.7,SE,4.8,0,0
20718,2015-07-01 15:22:00,34.4,37,17.6,1009.7,NW,3.2,0,0
20719,2015-07-01 15:37:00,34.4,39,18.4,1009.7,SSE,1.6,0,0
20720,2015-07-01 15:52:00,35.6,35,17.8,1009.4,SW,4.8,0,0

```

Screenshot 15: The exported CSV file.

CONCLUSION

In this assignment we managed to load and analyse a CSV file. More specifically, we launched psql from the command line, connected to the database server, made a new database for the CSV data and connected to it, created a table for the CSV data and loaded them to it. Lastly, we run some queries on our dataset and finally export the result of one of the queries to a new CSV file. Various screenshots were taken through the whole procedure to help the reader understand the purpose of this assignment.