Homework 5

Question 1

In [ ]:

```
import pandas as pd

mpg = pd.read_csv('mpg.csv')

mpg.describe()
```

Out[ ]:

	displ	year	cyl	cty	hwy
count	234.000000	234.000000	234.000000	234.000000	234.000000
mean	3.471795	2003.500000	5.888889	16.858974	23.440171
std	1.291959	4.509646	1.611534	4.255946	5.954643
min	1.600000	1999.000000	4.000000	9.000000	12.000000
25%	2.400000	1999.000000	4.000000	14.000000	18.000000
50%	3.300000	2003.500000	6.000000	17.000000	24.000000
75%	4.600000	2008.000000	8.000000	19.000000	27.000000
max	7.000000	2008.000000	8.000000	35.000000	44.000000

The equivalent of the describe() function from Python in R is the summary() function.

Question 2

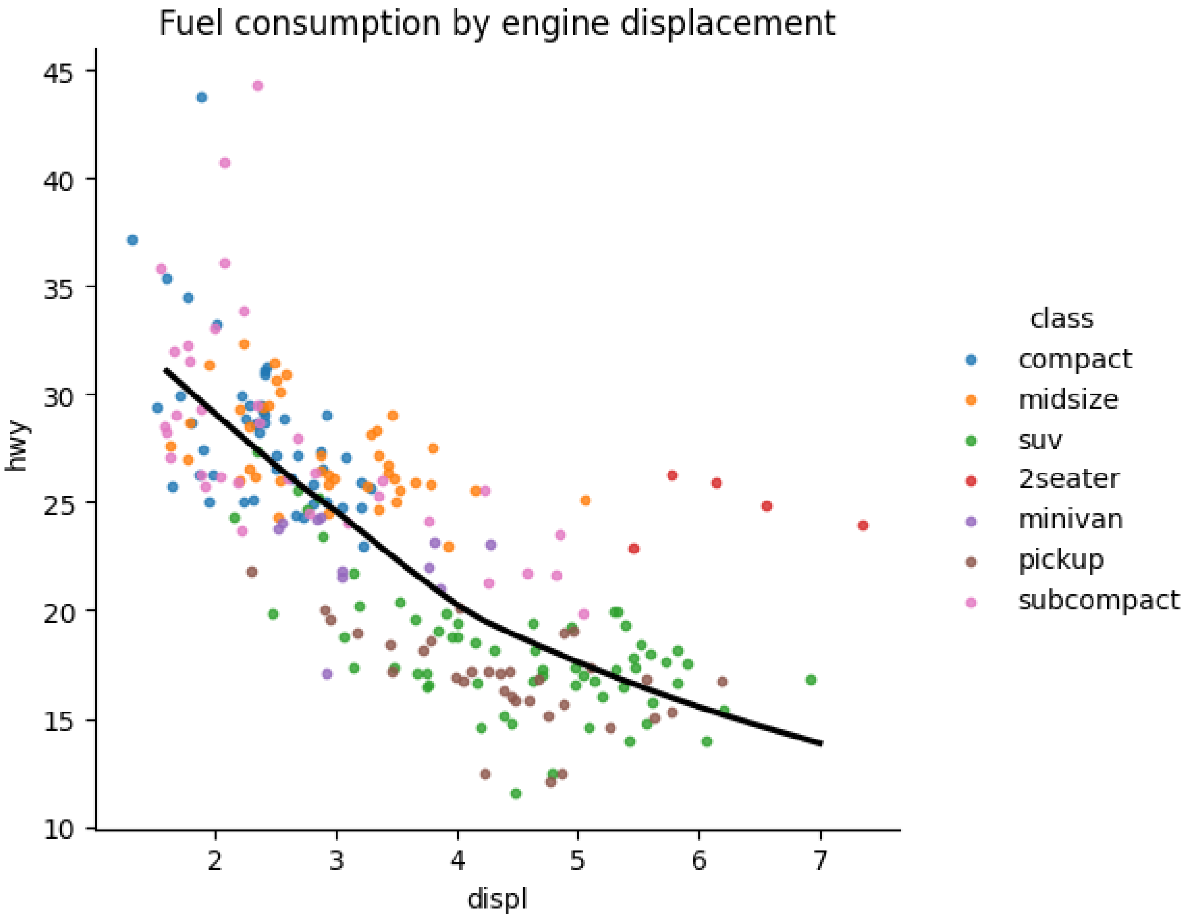
In [ ]:

```
import seaborn as sns
import pandas as pd

mpg = pd.read_csv('mpg.csv')

lm_plot = sns.lmplot(
    data = mpg,
    x = "displ",
    y = "hwy",
    hue = "class",
    fit_reg = False,
    scatter_kws = {"s": 10},
    ci = None,
    x_jitter = 0.5,
    y_jitter = 0.5
).set(
    title = "Fuel consumption by engine displacement",
    xlabel = "Engine displacement",
    ylabel = "Fuel consumption"
).tight_layout()

_ = sns.regplot(
    data = mpg,
    x = "displ",
    y = "hwy",
    scatter = False,
    ax = lm_plot.ax,
    line_kws = {"color": "black"},
    lowess = True
)
```



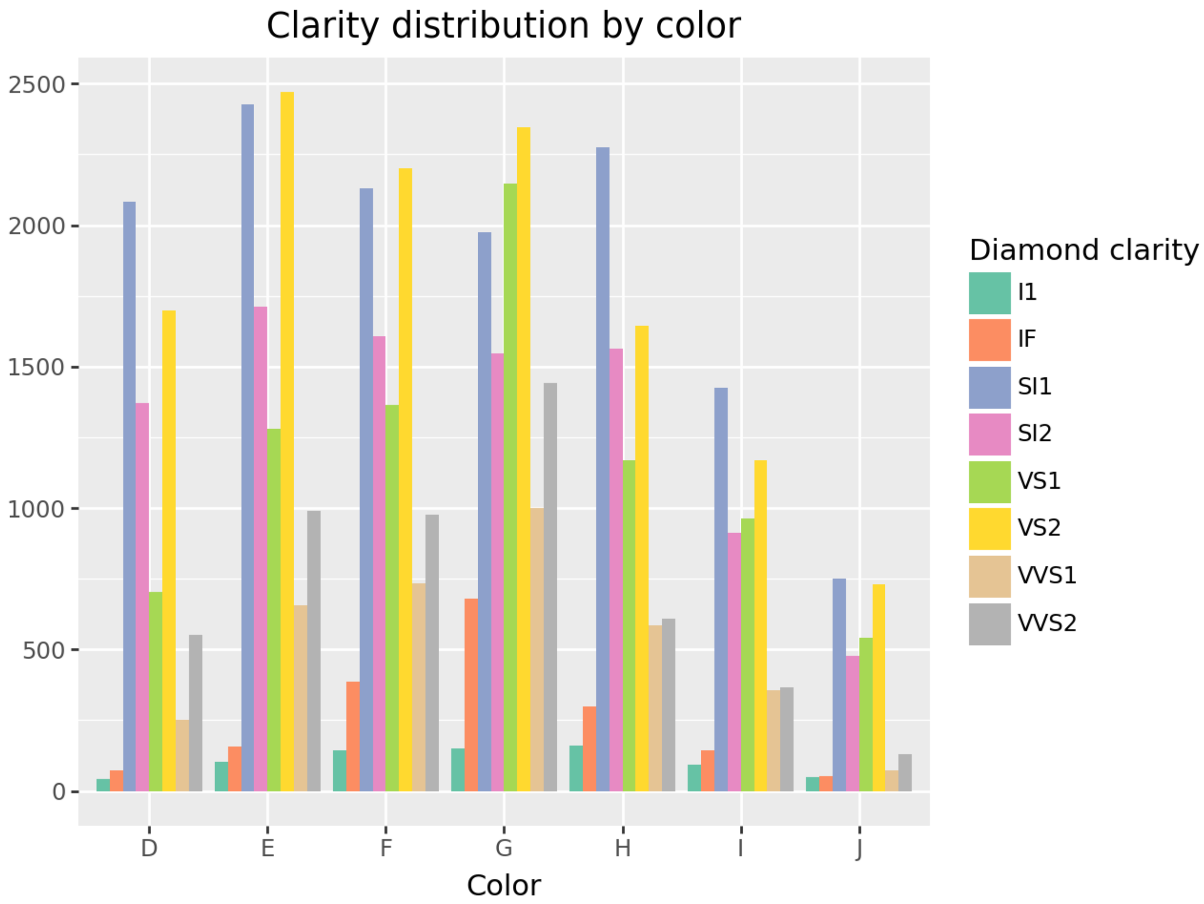
Question 3

In [ ]:

```
import pandas as pd
from plotnine import aes, geom_jitter, geom_smooth, geom_bar, ggplot, ggtitl

diamonds = pd.read_csv('diamonds.csv')

(
    ggplot(data = diamonds, mapping = aes(x = "color", fill = "pd.Categorica
     geom_bar(position = "dodge") +
     scale_fill_brewer(type = "qual", palette = "Set2", name = "Diamond clar
     xlab("Color") +
     ylab("") +
    ggtitle("Clarity distribution by color")
)
```



Question 4

HW2 Exercise 1

In [ ]:

```
import pandas as pd

flights = pd.read_csv('flights.csv')

(
    flights
    .query("month >= 7 & month <= 11")
    .assign(gain = flights["arr_delay"] - flights["dep_delay"])
    .groupby("carrier")
    .agg(average_gain = ("gain", "mean"))
    .rename(columns = {"average_gain": "average_gain"})
    .sort_values("average_gain", ascending = False)
)
```

Out[ ]:

	average_gain
carrier	
FL	1.285235
F9	0.545455
MQ	0.063590
HA	-0.909774
US	-2.837182
OO	-2.846154
YV	-3.017921
B6	-4.905070
EV	-5.787007
DL	-7.960461
WN	-8.571100
AA	-8.669899
UA	-9.621162
VX	-9.875375
9E	-10.288170
AS	-22.211409

HW2 Exercise 4

In [ ]:

```python
import pandas as pd

transactions = pd.read_csv('transactions.csv')
cols_to_keep = ["month"]

(
    transactions
    .melt(id_vars = ["person", "month"], value_vars = ["purchase", "sale"],
    .query("person == 'jenna' | person == 'john'")
    .drop_duplicates(subset = ['month'])
    [cols_to_keep]
    .reset_index(drop = True)
)
```

Out[ ]:

	month
0	1
1	3

Question 5

HW3 Exercise 1

In [ ]:

```
import pandas as pd

persons = pd.read_csv('persons.csv')
food = pd.read_csv('food.csv')
drinks = pd.read_csv('drinks.csv')
dinners = pd.read_csv('dinners.csv')
cols_to_keep = ["drink", "drink_price", "food", "food_price", "first_name",

dinners_explicit = (
    dinners
    .merge(drinks, left_on = 'drink_id', right_on = 'item_id', how = 'left')
    .rename(columns = {
        'price': 'drink_price',
        'item_name': 'drink'
    })
    .merge(food, left_on = 'food_id', right_on = 'food_id', how = 'left')
    .rename(columns = {
        'price': 'food_price',
        'name': 'food'
    })
    .merge(persons, left_on = 'person_id', right_on = 'id', how = 'left')
    [cols_to_keep]
)
dinners_explicit
```

Out[ ]:

	drink	drink_price	food	food_price	first_name	last_name	age
0	NaN	NaN	pasta	$8.50	Valter	Evangelista	34.0
1	NaN	NaN	ice cream	$4.50	Polly	Verity	61.0
2	water	$1.00	NaN	NaN	NaN	NaN	NaN
3	beer	$5.00	cake	$4.50	Aysha	Freitas	55.0
4	NaN	NaN	pizza	$12	Rayno	Van Kann	29.0
5	water	$1.00	fish	$15.00	Valter	Evangelista	34.0
6	NaN	NaN	pizza	$12	Rayno	Van Kann	29.0
7	sparkling water	$2.00	ice cream	$4.50	Ksenya	Dunai	31.0
8	soda	$2.50	pop corn	$1.50	Polly	Verity	61.0
9	water	$1.00	salad	$5.00	Aysha	Freitas	55.0
10	wine	$9.00	NaN	NaN	NaN	NaN	NaN
11	soda	$2.50	salad	$5.00	NaN	NaN	NaN
12	beer	$5.00	cake	$4.50	Aysha	Freitas	55.0
13	NaN	NaN	steak	$12.00	NaN	NaN	NaN
14	wine	$9.00	NaN	NaN	Valter	Evangelista	34.0
15	wine	$9.00	pop corn	$1.50	NaN	NaN	NaN
16	NaN	NaN	fish	$15.00	NaN	NaN	NaN
17	water	$1.00	fries	$3.00	Polly	Verity	61.0
18	sparkling water	$2.00	burger	$5.00	Polly	Verity	61.0
19	soda	$2.50	steak	$12.00	Polly	Verity	61.0
20	NaN	NaN	fries	$3.00	Valter	Evangelista	34.0
21	NaN	NaN	NaN	NaN	NaN	NaN	NaN
22	NaN	NaN	pasta	$8.50	NaN	NaN	NaN
23	soda	$2.50	pasta	$8.50	NaN	NaN	NaN
24	sparkling water	$2.00	salad	$5.00	NaN	NaN	NaN
25	sparkling water	$2.00	pizza	$12	Rayno	Van Kann	29.0
26	sparkling water	$2.00	ice cream	$4.50	NaN	NaN	NaN
27	sparkling water	$2.00	fish	$15.00	NaN	NaN	NaN

	drink	drink_price	food	food_price	first_name	last_name	age
28	NaN	NaN	fries	$3.00	Polly	Verity	61.0
29	beer	$5.00	pasta	$8.50	Valter	Evangelista	34.0
30	water	$1.00	burger	$5.00	NaN	NaN	NaN
31	NaN	NaN	pasta	$8.50	Rayno	Van Kann	29.0
32	NaN	NaN	salad	$5.00	Aysha	Freitas	55.0
33	NaN	NaN	fries	$3.00	NaN	NaN	NaN
34	soda	$2.50	cake	$4.50	NaN	NaN	NaN
35	sparkling water	$2.00	fish	$15.00	NaN	NaN	NaN
36	soda	$2.50	NaN	NaN	NaN	NaN	NaN
37	beer	$5.00	pop corn	$1.50	NaN	NaN	NaN
38	wine	$9.00	fish	$15.00	NaN	NaN	NaN
39	NaN	NaN	salad	$5.00	NaN	NaN	NaN
40	soda	$2.50	NaN	NaN	NaN	NaN	NaN
41	NaN	NaN	fries	$3.00	NaN	NaN	NaN
42	wine	$9.00	pizza	$12	Polly	Verity	61.0
43	soda	$2.50	NaN	NaN	Aysha	Freitas	55.0
44	water	$1.00	pizza	$12	NaN	NaN	NaN
45	water	$1.00	fries	$3.00	Polly	Verity	61.0
46	NaN	NaN	pop corn	$1.50	NaN	NaN	NaN
47	soda	$2.50	pizza	$12	Rayno	Van Kann	29.0
48	sparkling water	$2.00	pizza	$12	Aysha	Freitas	55.0
49	NaN	NaN	steak	$12.00	Polly	Verity	61.0

HW3 Exercise 2 - Top 2 Drinks

In [ ]:

```
import pandas as pd

(
    dinners_explicit
    .groupby('drink').size()
    .sort_values(ascending = False)
    .head(2)
)
```

```
Out[ ]: drink
        soda              9
        sparkling water   8
        dtype: int64
```

HW3 Exercise 2 - Top 2 Foods

```
In [ ]: import pandas as pd

        (
            dinners_explicit
            .groupby('food').size()
            .sort_values(ascending = False)
            .head(2)
        )
```

```
Out[ ]: food
        pizza    7
        fries    6
        dtype: int64
```