# DSAI510 Assignment 02

October 8, 2025

## 1 Assignment 2 - Deadline: Oct 12, 2025, Sun 11pm

**DSAI 510 Fall 2025** Complete the assignment below and upload both the .ipynb file and its pdf to https://moodle.boun.edu.tr by the deadline given above. The submission page on Moodle will close automatically after this date and time.

To make a pdf, this may work: Hit CMD+P or CTRL+P, and save it as PDF. You may also use other options from the File menu.

[2]:
```
# Run this cell first

import pandas as pd
import numpy as np

# Set the display option to show all rows scrolling with a slider
# pd.set_option('display.max_rows', None)
# To disable this, run the line below:
# pd.reset_option('display.max_rows')
```

### 1.1 Note:

In the problems below, if it asks, "show the number of records that are nonzero", the answer is a number; so you don't need to show the records themselves. But if it asks, "show the records with NaN", it wants you to print those records (rows) containing NAN and other entries, not asking how many such records there are. So be careful about what you're asked.

### 1.2 Problem 1 (10 pts)

a) Load **Electric_Vehicle_Population_Data-modified1.csv** and **Electric_Vehicle_Population_Data-modified2.csv** into pandas dataframes as df1 and df2.

b) Inspect the first and last five records with `head()` and `tail()` for both dataframes.

c) Use `len()` and `print()` [or `display()`] to show how many records each dataframe contains.

d) Use `info()` to get a summary of both dataframes.

e) Combine df1 and df2 into a new dataframe called df3 by using `concat()` and print the number of records in the new dataframe df3.

f) Find and print the number of duplicate records by using `duplicated()` and `sum()`.

g) Drop duplicates, save the new dataframe as dfALL and then print the number of records in dfALL.

[3]:
```
# part a)
df1 = pd.read_csv("Electric_Vehicle_Population_Data-modified1.csv")
df2 = pd.read_csv("Electric_Vehicle_Population_Data-modified2.csv")
```

[4]:
```
# part b)
print("=== df1 head ===")
print(df1.head())
print("\n")

print("=== df1 tail ===")
print(df1.tail())
print("\n")

print("=== df2 head ===")
print(df2.head())
print("\n")

print("=== df2 tail ===")
print(df2.tail())
print("\n")
```

```
=== df1 head ===
   VIN (1-10)    County      City State  Postal Code  Model Year     Make  \
0  KM8K33AGXL      King   Seattle    WA        98103        2020  HYUNDAI
1  1C4RJYB61N      King   Bothell    WA        98011        2022     JEEP
2  1C4RJYD61P    Yakima    Yakima    WA        98908        2023     JEEP
3  5YJ3E1EA7J      King  Kirkland    WA        98034        2018    TESLA
4  WBY7Z8C5XJ  Thurston   Olympia    WA        98501        2018      BMW

            Model                   Electric Vehicle Type  \
0            KONA          Battery Electric Vehicle (BEV)
1  GRAND CHEROKEE  Plug-in Hybrid Electric Vehicle (PHEV)
2  GRAND CHEROKEE  Plug-in Hybrid Electric Vehicle (PHEV)
3         MODEL 3          Battery Electric Vehicle (BEV)
4              I3  Plug-in Hybrid Electric Vehicle (PHEV)

  Clean Alternative Fuel Vehicle (CAFV) Eligibility  Electric Range  \
0           Clean Alternative Fuel Vehicle Eligible             258
1             Not eligible due to low battery range              25
2             Not eligible due to low battery range              25
3           Clean Alternative Fuel Vehicle Eligible             215
4           Clean Alternative Fuel Vehicle Eligible              97

   Base MSRP  Legislative District  DOL Vehicle ID  \
0          0                  43.0       249675142
```

```
1          0                  1.0       233928502
2          0                 14.0       229675939
3          0                 45.0       104714466
4          0                 22.0       185498386

                 Vehicle Location  \
0     POINT (-122.34301 47.659185)
1     POINT (-122.20578 47.762405)
2  POINT (-120.6027202 46.5965625)
3     POINT (-122.209285 47.71124)
4     POINT (-122.89692 47.043535)

                                Electric Utility  2020 Census Tract
0   CITY OF SEATTLE - (WA)|CITY OF TACOMA - (WA)        53033004800
1  PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)        53033021804
2                                    PACIFICORP        53077002900
3  PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)        53033021903
4                       PUGET SOUND ENERGY INC        53067010700


=== df1 tail ===
        VIN (1-10)     County        City State  Postal Code  Model Year  \
108441  WBY8P8C55K       King     Seattle    WA        98105        2019
108442  YV4H60CF3R     Pierce      Graham    WA        98338        2024
108443  1FADP5CU7F  Snohomish      Monroe    WA        98272        2015
108444  1G1FZ6S07L  Snohomish     Bothell    WA        98012        2020
108445  5YJ3E1EB1M      Grant  Moses Lake    WA        98837        2021

             Make    Model                   Electric Vehicle Type  \
108441        BMW       I3  Plug-in Hybrid Electric Vehicle (PHEV)
108442      VOLVO     XC90  Plug-in Hybrid Electric Vehicle (PHEV)
108443       FORD    C-MAX  Plug-in Hybrid Electric Vehicle (PHEV)
108444  CHEVROLET  BOLT EV           Battery Electric Vehicle (BEV)
108445      TESLA  MODEL 3           Battery Electric Vehicle (BEV)

        Clean Alternative Fuel Vehicle (CAFV) Eligibility  Electric Range  \
108441            Clean Alternative Fuel Vehicle Eligible             126
108442            Clean Alternative Fuel Vehicle Eligible              32
108443              Not eligible due to low battery range              19
108444            Clean Alternative Fuel Vehicle Eligible             259
108445  Eligibility unknown as battery range has not b…               0

        Base MSRP  Legislative District  DOL Vehicle ID  \
108441          0                 46.0       176391176
108442          0                  2.0       251387531
108443          0                 39.0       477108390
108444          0                  1.0       152533930
108445          0                 13.0       171366499
```

```
                      Vehicle Location  \
108441     POINT (-122.319115 47.66132)
108442  POINT (-122.2953401 47.0763961)
108443     POINT (-121.972215 47.85674)
108444   POINT (-122.1876761 47.820517)
108445  POINT (-119.2599876 47.1240154)

                                       Electric Utility  2020 Census Tract
108441       CITY OF SEATTLE - (WA)|CITY OF TACOMA - (WA)        53033004201
108442  BONNEVILLE POWER ADMINISTRATION||CITY OF TACOM…        53053073132
108443                              PUGET SOUND ENERGY INC        53061052113
108444                              PUGET SOUND ENERGY INC        53061052009
108445                            PUD NO 2 OF GRANT COUNTY        53025011001


=== df2 head ===
   VIN (1-10)     County         City State  Postal Code  Model Year  \
0  1FMCU0EZ1N     Chelan    Wenatchee    WA      98801.0        2022
1  5YJ3E1EB9K  Snohomish    Arlington    WA      98223.0        2019
2  5YJSA1E57N       King  Woodinville    WA      98072.0        2022
3  5YJ3E1EB4J  Snohomish    Snohomish    WA      98290.0        2018
4  KL8CK6S00F    Whatcom   Bellingham    WA      98225.0        2015

        Make    Model                    Electric Vehicle Type  \
0       FORD   ESCAPE  Plug-in Hybrid Electric Vehicle (PHEV)
1      TESLA  MODEL 3          Battery Electric Vehicle (BEV)
2      TESLA  MODEL S          Battery Electric Vehicle (BEV)
3      TESLA  MODEL 3          Battery Electric Vehicle (BEV)
4  CHEVROLET    SPARK          Battery Electric Vehicle (BEV)

   Clean Alternative Fuel Vehicle (CAFV) Eligibility  Electric Range  \
0           Clean Alternative Fuel Vehicle Eligible              38
1           Clean Alternative Fuel Vehicle Eligible             220
2  Eligibility unknown as battery range has not b…               0
3           Clean Alternative Fuel Vehicle Eligible             215
4           Clean Alternative Fuel Vehicle Eligible              82

   Base MSRP  Legislative District  DOL Vehicle ID  \
0          0                  12.0       226062931
1          0                  39.0       198860280
2          0                  45.0       220450240
3          0                  44.0       131652426
4          0                  42.0       177631044

                Vehicle Location  \
0    POINT (-120.32009 47.42255)
1    POINT (-122.12324 48.19485)
```

```
2   POINT (-122.151665 47.75855)
3  POINT (-122.091505 47.915555)
4  POINT (-122.486115 48.761615)

                                  Electric Utility  2020 Census Tract
0                         PUD NO 1 OF CHELAN COUNTY       5.300796e+10
1                            PUGET SOUND ENERGY INC       5.306105e+10
2      PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)       5.303303e+10
3                            PUGET SOUND ENERGY INC       5.306105e+10
4  PUGET SOUND ENERGY INC||PUD NO 1 OF WHATCOM CO…       5.307300e+10


=== df2 tail ===
       VIN (1-10)        County               City State  Postal Code  \
50479  WBY43AW05P  Grays Harbor          Montesano    WA      98563.0
50480  5YJ3E1EB7P          King            Seattle    WA      98104.0
50481  5YJYGDEEXM          King            Seattle    WA      98109.0
50482  5UXTA6C08P     Snohomish  Mountlake Terrace    WA      98043.0
50483  7SAYGDEF8N        Skagit       Mount Vernon    WA      98273.0

       Model Year   Make    Model                   Electric Vehicle Type  \
50479        2023    BMW       I4          Battery Electric Vehicle (BEV)
50480        2023  TESLA  MODEL 3          Battery Electric Vehicle (BEV)
50481        2021  TESLA  MODEL Y          Battery Electric Vehicle (BEV)
50482        2023    BMW       X5  Plug-in Hybrid Electric Vehicle (PHEV)
50483        2022  TESLA  MODEL Y          Battery Electric Vehicle (BEV)

       Clean Alternative Fuel Vehicle (CAFV) Eligibility  Electric Range  \
50479  Eligibility unknown as battery range has not b…               0
50480  Eligibility unknown as battery range has not b…               0
50481  Eligibility unknown as battery range has not b…               0
50482            Clean Alternative Fuel Vehicle Eligible              30
50483  Eligibility unknown as battery range has not b…               0

       Base MSRP  Legislative District  DOL Vehicle ID  \
50479          0                  19.0       251204075
50480          0                  43.0       241344414
50481          0                  43.0       180705626
50482          0                   1.0       240473950
50483          0                  40.0       207667589

                  Vehicle Location  \
50479  POINT (-123.60535 46.982215)
50480   POINT (-122.329075 47.6018)
50481  POINT (-122.34848 47.632405)
50482   POINT (-122.30842 47.78416)
50483  POINT (-122.338975 48.41333)
```

```
                                          Electric Utility  2020 Census Tract
50479  BONNEVILLE POWER ADMINISTRATION||PUD NO 1 OF G…       5.302700e+10
50480       CITY OF SEATTLE - (WA)|CITY OF TACOMA - (WA)       5.303301e+10
50481       CITY OF SEATTLE - (WA)|CITY OF TACOMA - (WA)       5.303301e+10
50482                           PUGET SOUND ENERGY INC       5.306105e+10
50483                           PUGET SOUND ENERGY INC       5.305795e+10
```

[5]:
```
# part c
print("df1:", len(df1))
print("df2:", len(df2))
```

```
df1: 108446
df2: 50484
```

[6]:
```
# part d
print("=== df1 info ===")
print(df1.info())

print("=== df2 info ===")
print(df2.info())
```

```
=== df1 info ===
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 108446 entries, 0 to 108445
Data columns (total 17 columns):
 #   Column                                             Non-Null Count   Dtype
---  ------                                             --------------   -----
 0   VIN (1-10)                                         108446 non-null  object
 1   County                                             108446 non-null  object
 2   City                                               108446 non-null  object
 3   State                                              108446 non-null  object
 4   Postal Code                                        108446 non-null  int64
 5   Model Year                                         108446 non-null  int64
 6   Make                                               108446 non-null  object
 7   Model                                              108446 non-null  object
 8   Electric Vehicle Type                              108446 non-null  object
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  108446 non-null  object
 10  Electric Range                                     108446 non-null  int64
 11  Base MSRP                                          108446 non-null  int64
 12  Legislative District                               108407 non-null  float64
 13  DOL Vehicle ID                                     108446 non-null  int64
 14  Vehicle Location                                   108445 non-null  object
 15  Electric Utility                                   108446 non-null  object
 16  2020 Census Tract                                  108446 non-null  int64
dtypes: float64(1), int64(6), object(10)
memory usage: 14.1+ MB
None
```

```
=== df2 info ===
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50484 entries, 0 to 50483
Data columns (total 17 columns):
 #   Column                                             Non-Null Count  Dtype
---  ------                                             --------------  -----
 0   VIN (1-10)                                         50484 non-null  object
 1   County                                             50481 non-null  object
 2   City                                               50481 non-null  object
 3   State                                              50484 non-null  object
 4   Postal Code                                        50481 non-null  float64
 5   Model Year                                         50484 non-null  int64
 6   Make                                               50484 non-null  object
 7   Model                                              50484 non-null  object
 8   Electric Vehicle Type                              50484 non-null  object
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  50484 non-null  object
 10  Electric Range                                     50484 non-null  int64
 11  Base MSRP                                          50484 non-null  int64
 12  Legislative District                               50170 non-null  float64
 13  DOL Vehicle ID                                     50484 non-null  int64
 14  Vehicle Location                                   50478 non-null  object
 15  Electric Utility                                   50481 non-null  object
 16  2020 Census Tract                                  50481 non-null  float64
dtypes: float64(3), int64(4), object(10)
memory usage: 6.5+ MB
None
```

[7]:
```
# part e
df3 = pd.concat([df1, df2])
print("df3:", len(df3))
```

```
df3: 158930
```

[8]:
```
# part f
print(df3.duplicated().sum())
```

```
8448
```

[9]:
```
# part g
dfALL = df3.drop_duplicates()
# dfALL = df3.drop_duplicates(keep='first')
# instead of dropping them all maybe we can keep the first occurences?
print(dfALL.duplicated().sum())
```

```
0
```

### 1.3 Problem 2 (10 pts)

a) Make a new dataframe df4, keep only the columns **Model Year**, **Make**, **Model**, **Electric Range**, **Vehicle Location** and drop all other columns from dfAll.

b) Change the column name **Model Year** into **Year**.

c) Show the record with index number 10.

d) As you see, the **Vehicle Location** shows the coordinates in the format "POINT (-122.20264 47.6785)". Here the first number (-122.20264) is the longitude and second number is the latitude. Make two new columns **Lattitude** and **Longitude**, carry the numbers to these columns by using `str` method from pandas. Finally change the type of **Lattitude** and **Longitude** into float if they're not already. Finally, drop the column **Vehicle Location**.

[10]:
```
# part a)
df4 = dfALL.loc[:, ['Model Year', 'Make', 'Model', 'Electric Range', 'Vehicle␣
 ↪Location']]
df4
```

[10]:

| | Model Year | Make | Model | Electric Range | \ |
|---|---|---|---|---|---|
| 0 | 2020 | HYUNDAI | KONA | 258 | |
| 1 | 2022 | JEEP | GRAND CHEROKEE | 25 | |
| 2 | 2023 | JEEP | GRAND CHEROKEE | 25 | |
| 3 | 2018 | TESLA | MODEL 3 | 215 | |
| 4 | 2018 | BMW | I3 | 97 | |
| … | … | … | … | … | |
| 50479 | 2023 | BMW | I4 | 0 | |
| 50480 | 2023 | TESLA | MODEL 3 | 0 | |
| 50481 | 2021 | TESLA | MODEL Y | 0 | |
| 50482 | 2023 | BMW | X5 | 30 | |
| 50483 | 2022 | TESLA | MODEL Y | 0 | |

| | Vehicle Location |
|---|---|
| 0 | POINT (-122.34301 47.659185) |
| 1 | POINT (-122.20578 47.762405) |
| 2 | POINT (-120.6027202 46.5965625) |
| 3 | POINT (-122.209285 47.71124) |
| 4 | POINT (-122.89692 47.043535) |
| … | … |
| 50479 | POINT (-123.60535 46.982215) |
| 50480 | POINT (-122.329075 47.6018) |
| 50481 | POINT (-122.34848 47.632405) |
| 50482 | POINT (-122.30842 47.78416) |
| 50483 | POINT (-122.338975 48.41333) |

[150482 rows x 5 columns]

[11]:
```
# part b)
df4 = df4.rename(columns={'Model Year': 'Year'})
df4
```

[11]:
```
        Year     Make           Model  Electric Range  \
0       2020  HYUNDAI            KONA             258
1       2022     JEEP  GRAND CHEROKEE              25
2       2023     JEEP  GRAND CHEROKEE              25
3       2018    TESLA         MODEL 3             215
4       2018      BMW              I3              97
…        …       …              …               …
50479   2023      BMW              I4               0
50480   2023    TESLA         MODEL 3               0
50481   2021    TESLA         MODEL Y               0
50482   2023      BMW              X5              30
50483   2022    TESLA         MODEL Y               0

                      Vehicle Location
0         POINT (-122.34301 47.659185)
1         POINT (-122.20578 47.762405)
2       POINT (-120.6027202 46.5965625)
3         POINT (-122.209285 47.71124)
4         POINT (-122.89692 47.043535)
…                                  …
50479     POINT (-123.60535 46.982215)
50480      POINT (-122.329075 47.6018)
50481     POINT (-122.34848 47.632405)
50482      POINT (-122.30842 47.78416)
50483     POINT (-122.338975 48.41333)

[150482 rows x 5 columns]
```

[12]:
```
# part c)
df4.iloc[10]
```

[12]:
```
Year                                      2018
Make                                     TESLA
Model                                  MODEL 3
Electric Range                             215
Vehicle Location    POINT (-122.20264 47.6785)
Name: 10, dtype: object
```

[13]:
```
# part d)
df4[['Longitude', 'Latitude']] = df4['Vehicle Location'].str.extract(r'POINT␣
 ↪\(([^ ]+) ([^ ]+)\)').astype(float)
df4 = df4.drop(columns=['Vehicle Location'])
df4
```

[13]:
```
        Year     Make           Model  Electric Range   Longitude   Latitude
0       2020  HYUNDAI            KONA             258 -122.343010  47.659185
1       2022     JEEP  GRAND CHEROKEE              25 -122.205780  47.762405
```

```
2      2023     JEEP  GRAND CHEROKEE              25 -120.602720  46.596562
3      2018    TESLA         MODEL 3             215 -122.209285  47.711240
4      2018      BMW              I3              97 -122.896920  47.043535
…       …       …               …               …          …          …
50479  2023      BMW              I4               0 -123.605350  46.982215
50480  2023    TESLA         MODEL 3               0 -122.329075  47.601800
50481  2021    TESLA         MODEL Y               0 -122.348480  47.632405
50482  2023      BMW              X5              30 -122.308420  47.784160
50483  2022    TESLA         MODEL Y               0 -122.338975  48.413330

[150482 rows x 6 columns]
```

### 1.4 Problem 3 (10 pts)

a) The file **EV_prices.csv** contains prices for various makes, models, and years of cars. Load this file into a dataframe. Rename the column **Model Year** to **Year**.

b) We want to add a new column **Price** to our dataframe from the previous problem (df4). This column will include the price of the car for the corresponding make, model and year if this information is available in the file **EV_prices.csv**. If not available, we'll still keep the record but the entry for price will be empty, NA, None or NaN. To achieve this, merge the dataframe from the previous problem (df4) with the dataframe containing the data from **EV_prices.csv**. Think carefully and decide if you need to merge with 'inner' or 'outer' method. At the end, we should have these columns in the merged dataframe: **Year**, **Make**, **Model**, **Electric Range**, **Latitude**, **Longitude** and **Price**. Again, the **Price** column will have numbers only for some records, but it will be empty or NaN for the rest.

c) Next, show the number of records which has price information in the **Price** column. Hint: You can use a one-liner containing `len()` and `dropna()` together.

[14]:
```
# part a)
car_df = pd.read_csv("EV_prices.csv")
car_df = car_df.rename(columns={'Model Year': 'Year'})
car_df
```

[14]:
```
     Year       Make           Model    Price
0    2020    HYUNDAI            KONA  22000.0
1    2022       JEEP  GRAND CHEROKEE      NaN
2    2023       JEEP  GRAND CHEROKEE      NaN
3    2018      TESLA         MODEL 3  44000.0
4    2018        BMW              I3      NaN
..    …         …               …       …
423  2024     NISSAN            LEAF      NaN
424  2021     JAGUAR          I-PACE      NaN
425  1999       FORD          RANGER      NaN
426  1997  CHEVROLET     S-10 PICKUP      NaN
427  2021    BENTLEY        BENTAYGA      NaN
```

```
[428 rows x 4 columns]
```

[15]:
```
# part b)
df5 = df4.merge(car_df, on=['Make', 'Year', 'Model'], how='left')
df5
```

[15]:

| | Year | Make | Model | Electric Range | Longitude | Latitude | \ |
|---|---|---|---|---|---|---|---|
| 0 | 2020 | HYUNDAI | KONA | 258 | -122.343010 | 47.659185 | |
| 1 | 2022 | JEEP | GRAND CHEROKEE | 25 | -122.205780 | 47.762405 | |
| 2 | 2023 | JEEP | GRAND CHEROKEE | 25 | -120.602720 | 46.596562 | |
| 3 | 2018 | TESLA | MODEL 3 | 215 | -122.209285 | 47.711240 | |
| 4 | 2018 | BMW | I3 | 97 | -122.896920 | 47.043535 | |
| … | … | … | … | … | … | … | |
| 150477 | 2023 | BMW | I4 | 0 | -123.605350 | 46.982215 | |
| 150478 | 2023 | TESLA | MODEL 3 | 0 | -122.329075 | 47.601800 | |
| 150479 | 2021 | TESLA | MODEL Y | 0 | -122.348480 | 47.632405 | |
| 150480 | 2023 | BMW | X5 | 30 | -122.308420 | 47.784160 | |
| 150481 | 2022 | TESLA | MODEL Y | 0 | -122.338975 | 48.413330 | |

| | Price |
|---|---|
| 0 | 22000.0 |
| 1 | NaN |
| 2 | NaN |
| 3 | 44000.0 |
| 4 | NaN |
| … | … |
| 150477 | NaN |
| 150478 | NaN |
| 150479 | NaN |
| 150480 | NaN |
| 150481 | NaN |

```
[150482 rows x 7 columns]
```

[ ]:
```
# part c)
print((~df5['Price'].isna()).sum())
```

```
19736
```

### 1.5 Problem 4 (10 pts)

a) Using the DataFrame from the previous problem, remove records where the **Year** column is for 2015 or earlier. Apply the format `dfmerged = dfmerged[condition]`, choosing the appropriate condition.

b) Generate the table, a screenshot of which is provided below, using the `pivot_table()` method and the aggregation function `size`. The entries in the table will indicate the number of cars with the specified make, model, and year in the dataset.

| | student_id | Subject_Quarter | Score |
|---|---|---|---|
| 0 | 1 | Math_Q1 | 85 |
| 1 | 2 | Math_Q1 | 90 |
| 2 | 3 | Math_Q1 | 82 |
| 3 | 1 | Math_Q2 | 88 |
| 4 | 2 | Math_Q2 | 85 |
| 5 | 3 | Math_Q2 | 80 |
| 6 | 1 | Math_Q3 | 87 |
| 7 | 2 | Math_Q3 | 83 |
| 8 | 3 | Math_Q3 | 84 |
| 9 | 1 | Science_Q1 | 78 |
| 10 | 2 | Science_Q1 | 88 |
| 11 | 3 | Science_Q1 | 80 |

c) Use the `groupby()` method to create a table similar to the one above but this time entries will indicate the average latitude of the car, instead of number of cars, with the specified make, model and year.

[17]:
```
# part a)
df5 = df5[df5['Year'] > 2015]
df5
```

[17]:

| | Year | Make | Model | Electric Range | Longitude | Latitude | \ |
|---|---|---|---|---|---|---|---|
| 0 | 2020 | HYUNDAI | KONA | 258 | -122.343010 | 47.659185 | |
| 1 | 2022 | JEEP | GRAND CHEROKEE | 25 | -122.205780 | 47.762405 | |
| 2 | 2023 | JEEP | GRAND CHEROKEE | 25 | -120.602720 | 46.596562 | |
| 3 | 2018 | TESLA | MODEL 3 | 215 | -122.209285 | 47.711240 | |
| 4 | 2018 | BMW | I3 | 97 | -122.896920 | 47.043535 | |
| … | … | … | … | … | … | … | |
| 150477 | 2023 | BMW | I4 | 0 | -123.605350 | 46.982215 | |
| 150478 | 2023 | TESLA | MODEL 3 | 0 | -122.329075 | 47.601800 | |
| 150479 | 2021 | TESLA | MODEL Y | 0 | -122.348480 | 47.632405 | |
| 150480 | 2023 | BMW | X5 | 30 | -122.308420 | 47.784160 | |
| 150481 | 2022 | TESLA | MODEL Y | 0 | -122.338975 | 48.413330 | |

| | Price |
|---|---|
| 0 | 22000.0 |
| 1 | NaN |
| 2 | NaN |
| 3 | 44000.0 |
| 4 | NaN |
| … | … |
| 150477 | NaN |
| 150478 | NaN |
| 150479 | NaN |
| 150480 | NaN |
| 150481 | NaN |

[134880 rows x 7 columns]

[28]:
```
# part b)
df6 = df5.pivot_table(aggfunc='size', index=['Make', 'Model'],␣
 ↪columns=['Year']).fillna(0)
df6
```

[28]:

| Year | | 2016 | 2017 | 2018 | 2019 | 2020 | 2021 | 2022 | 2023 | 2024 |
|---|---|---|---|---|---|---|---|---|---|---|
| Make | Model | | | | | | | | | |
| ALFA ROMEO | TONALE | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 12.0 |
| AUDI | A3 | 212.0 | 189.0 | 173.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | A7 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 12.0 | 0.0 | 0.0 | 0.0 |
| | A8 E | 0.0 | 0.0 | 0.0 | 0.0 | 3.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| | E-TRON | 0.0 | 0.0 | 0.0 | 443.0 | 0.0 | 183.0 | 228.0 | 125.0 | 0.0 |
| … | | … | … | … | … | … | … | … | … | … |
| VOLVO | S90 | 0.0 | 0.0 | 18.0 | 4.0 | 3.0 | 0.0 | 0.0 | 2.0 | 0.0 |
| | V60 | 0.0 | 0.0 | 0.0 | 0.0 | 4.0 | 6.0 | 4.0 | 9.0 | 5.0 |
| | XC40 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 238.0 | 252.0 | 350.0 | 0.0 |
| | XC60 | 0.0 | 0.0 | 131.0 | 105.0 | 88.0 | 147.0 | 239.0 | 220.0 | 19.0 |
| | XC90 | 102.0 | 111.0 | 83.0 | 73.0 | 83.0 | 216.0 | 332.0 | 217.0 | 16.0 |

```
[115 rows x 9 columns]
```

[ ]:

[27]:
```
# part c)
df7 = df5.groupby(['Make', 'Model', 'Year'])['Latitude'].mean()
df7
```

[27]:
```
Make        Model   Year
ALFA ROMEO  TONALE  2024    47.607409
AUDI        A3      2016    47.619987
                    2017    47.481332
                    2018    47.496019
            A7      2021    47.491215
                                ...
VOLVO       XC90    2020    47.605401
                    2021    47.498350
                    2022    47.501800
                    2023    47.462351
                    2024    47.460175
Name: Latitude, Length: 358, dtype: float64
```

### 1.6 Problem 5 (10 pts)

a) There is a 3-row, 7-columns table whose code is given below. Use `melt()` to convert that table into this form:

| | student_id | Subject_Quarter | Score |
|---|---|---|---|
| **0** | 1 | Math_Q1 | 85 |
| **1** | 2 | Math_Q1 | 90 |
| **2** | 3 | Math_Q1 | 82 |
| **3** | 1 | Math_Q2 | 88 |
| **4** | 2 | Math_Q2 | 85 |
| **5** | 3 | Math_Q2 | 80 |
| **6** | 1 | Math_Q3 | 87 |
| **7** | 2 | Math_Q3 | 83 |
| **8** | 3 | Math_Q3 | 84 |
| **9** | 1 | Science_Q1 | 78 |
| **10** | 2 | Science_Q1 | 88 |
| **11** | 3 | Science_Q1 | 80 |

[20]:
```python
# Sample dataset
data = {
    'student_id': [1, 2, 3],
    'Math_Q1': [85, 90, 82],
    'Math_Q2': [88, 85, 80],
    'Math_Q3': [87, 83, 84],
    'Science_Q1': [78, 88, 80],
    'Science_Q2': [82, 85, 78],
    'Science_Q3': [84, 87, 83]
```

```
}
scores_df = pd.DataFrame(data)
scores_df
```

[20]:

| | student_id | Math_Q1 | Math_Q2 | Math_Q3 | Science_Q1 | Science_Q2 | Science_Q3 |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 85 | 88 | 87 | 78 | 82 | 84 |
| 1 | 2 | 90 | 85 | 83 | 88 | 85 | 87 |
| 2 | 3 | 82 | 80 | 84 | 80 | 78 | 83 |

[21]:
```
# your solution goes here
melted_df = scores_df.melt(id_vars='student_id', var_name='Subject_Quarter',␣
 ↪value_name='Score')
melted_df
```

[21]:

| | student_id | Subject_Quarter | Score |
|---|---|---|---|
| 0 | 1 | Math_Q1 | 85 |
| 1 | 2 | Math_Q1 | 90 |
| 2 | 3 | Math_Q1 | 82 |
| 3 | 1 | Math_Q2 | 88 |
| 4 | 2 | Math_Q2 | 85 |
| 5 | 3 | Math_Q2 | 80 |
| 6 | 1 | Math_Q3 | 87 |
| 7 | 2 | Math_Q3 | 83 |
| 8 | 3 | Math_Q3 | 84 |
| 9 | 1 | Science_Q1 | 78 |
| 10 | 2 | Science_Q1 | 88 |
| 11 | 3 | Science_Q1 | 80 |
| 12 | 1 | Science_Q2 | 82 |
| 13 | 2 | Science_Q2 | 85 |
| 14 | 3 | Science_Q2 | 78 |
| 15 | 1 | Science_Q3 | 84 |
| 16 | 2 | Science_Q3 | 87 |
| 17 | 3 | Science_Q3 | 83 |

### 1.7 Problem 6 - Quality Control in a Manufacturing Plant (10 pts)

Imagine you work as a quality control analyst in a manufacturing plant that produces ball bearings. Each day, multiple batches of ball bearings are produced. To ensure the consistency and quality of the ball bearings, samples from each batch are measured to determine their diameters.

Over the course of a month, you've collected diameter data for these samples from various batches. The objective is to determine the batch consistency by measuring the standard deviation of the diameters. A lower standard deviation would indicate that the ball bearings in a batch are more consistent in size.

a) Load the **ball_bearings.csv** file into a dataframe.

b) Use `groupby()` to calculate the standard deviation for each batch.

c) Sort the results in descending order wrt standard deviation, showing the batch with highest standard deviation at the top.

[22]:
```
# part a)
ball_df = pd.read_csv("ball_bearings.csv")
ball_df
```

[22]:
```
     batch_id   diameter
0           1  50.248357
1           1  49.930868
2           1  50.323844
3           1  50.761515
4           1  49.882923
..        …        …
595        30  49.744992
596        30  49.865063
597        30  49.510618
598        30  49.777853
599        30  50.188650

[600 rows x 2 columns]
```

[23]:
```
# part b)
stddev_series = ball_df.groupby('batch_id')['diameter'].std()
stddev_series
```

[23]:
```
batch_id
1     0.480014
2     0.484019
3     0.410424
4     0.556044
5     0.345405
6     0.511339
7     0.534851
8     0.451574
9     0.502297
10    0.367445
11    0.538856
12    0.593042
13    0.516218
14    0.559482
15    0.474119
16    0.319080
17    0.431002
18    0.362575
19    0.441640
20    0.577886
21    0.446483
```

```
22    0.645821
23    0.386267
24    0.608465
25    0.508577
26    0.409027
27    0.425961
28    0.435184
29    0.610599
30    0.388254
Name: diameter, dtype: float64
```

[24]:
```
# part c)
stddev_series.sort_values(ascending=False)
```

[24]:
```
batch_id
22    0.645821
29    0.610599
24    0.608465
12    0.593042
20    0.577886
14    0.559482
4     0.556044
11    0.538856
7     0.534851
13    0.516218
6     0.511339
25    0.508577
9     0.502297
2     0.484019
1     0.480014
15    0.474119
8     0.451574
21    0.446483
19    0.441640
28    0.435184
17    0.431002
27    0.425961
3     0.410424
26    0.409027
30    0.388254
23    0.386267
10    0.367445
18    0.362575
5     0.345405
16    0.319080
Name: diameter, dtype: float64
```