

Lab 1

```
# 1.1
# Import pandas

import pandas as pd

#Loading dataset

df = pd.read_csv("/content/Lab-04 diamonds.csv")

# 1.2
df.head()
```

	Unnamed: 0	carat	cut	color	clarity	depth	table	price	length	width	height
0	1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.61

```
# 1.3

print(df[:0])

Empty DataFrame
Columns: [Unnamed: 0, carat, cut, color, clarity, depth, table, price, length , width , height]
Index: []

# 1.4

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 53940 entries, 0 to 53939
Data columns (total 11 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Unnamed: 0   53940 non-null  int64
1   carat        53940 non-null  float64
2   cut          53940 non-null  object
3   color        53940 non-null  object
4   clarity      53940 non-null  object
5   depth        53940 non-null  float64
6   table        53940 non-null  float64
7   price        53940 non-null  int64
8   length       53940 non-null  float64
9   width        53940 non-null  float64
10  height       53940 non-null  float64
dtypes: float64(6), int64(2), object(3)
memory usage: 4.5+ MB

# 1.5

df.describe()
```

	Unnamed: 0	carat	depth	table	price	length
count	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.0000
mean	26970.500000	0.797940	61.749405	57.457184	3932.799722	5.7311
std	15571.281097	0.474011	1.432621	2.234491	3989.439738	1.1217
min	1.000000	0.200000	43.000000	43.000000	326.000000	0.0000
25%	13485.750000	0.400000	61.000000	56.000000	950.000000	4.7100
50%	26970.500000	0.700000	61.800000	57.000000	2401.000000	5.7000
75%	40455.250000	1.040000	62.500000	59.000000	5324.250000	6.5400
max	53940.000000	5.010000	79.000000	95.000000	18823.000000	10.7400

Lab 2- Data Cleaning Basics

```
print(df)
```

```

      Unnamed: 0  carat      cut color clarity depth table price \
0              1  0.23    Ideal     E    SI2   61.5   55.0   326
1              2  0.21  Premium     E    SI1   59.8   61.0   326
2              3  0.23    Good     E    VS1   56.9   65.0   327
3              4  0.29  Premium     I    VS2   62.4   58.0   334
4              5  0.31    Good     J    SI2   63.3   58.0   335
...          ...    ...      ...    ...    ...    ...    ...
53935         53936  0.72    Ideal     D    SI1   60.8   57.0  2757
53936         53937  0.72    Good     D    SI1   63.1   55.0  2757
53937         53938  0.70  Very Good     D    SI1   62.8   60.0  2757
53938         53939  0.86  Premium     H    SI2   61.0   58.0  2757
53939         53940  0.75    Ideal     D    SI2   62.2   55.0  2757

      length  width  height
0         3.95   3.98   2.43
1         3.89   3.84   2.31
2         4.05   4.07   2.31
3         4.20   4.23   2.63
4         4.34   4.35   2.75
...        ...    ...    ...
53935         5.75   5.76   3.50
53936         5.69   5.75   3.61
53937         5.66   5.68   3.56
53938         6.15   6.12   3.74
53939         5.83   5.87   3.64

```

```
[53940 rows x 11 columns]
```

```
# 2.1 Check for missing values
```

```
df.isnull().sum()
```

```
##Hence no missing values found
```

```

      Unnamed: 0    0
      carat      0
      cut      0
      color     0
      clarity   0
      depth     0
      table     0
      price     0
      length    0
      width     0
      height    0
      dtype: int64

```

```
# 2.2
```

```
print(df['color'].unique())
```

```

count_colors = len(df['color'].unique())
print('Count of unique colors:',count_colors)

```

```

['E' 'I' 'J' 'H' 'F' 'G' 'D']
Count of unique colors: 7

```

```
# 2.3
```

```

grouped_color = df.groupby('color')
print(grouped_color)

```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7c221902b700>
```

```
# 2.4
```

```
print(df[df['price']>5000])
```

```

      Unnamed: 0  carat      cut color clarity depth table price \
11416         11417  1.16    Ideal     E    SI2   62.7   56.0  5001
11417         11418  1.16    Ideal     E    SI2   59.9   57.0  5001
11418         11419  0.90    Good     G    VVS2   63.6   58.0  5001
11419         11420  0.90  Very Good     E    VS1   62.3   56.0  5001
11420         11421  0.90  Premium     D    VS2   62.6   59.0  5001
...          ...    ...      ...    ...    ...    ...    ...
27745         27746  2.00  Very Good     H    SI1   62.8   57.0  18803
27746         27747  2.07    Ideal     G    SI2   62.5   55.0  18804
27747         27748  1.51    Ideal     G    IF   61.7   55.0  18806
27748         27749  2.00  Very Good     G    SI1   63.5   56.0  18818

```

```
27749      27750    2.29    Premium    I    VS2    60.8    60.0    18823

      length    width    height
11416      6.69      6.73      4.21
11417      6.80      6.82      4.08
11418      6.10      6.11      3.88
11419      6.10      6.19      3.83
11420      6.14      6.17      3.85
...
27745      7.95      8.00      5.01
27746      8.20      8.13      5.11
27747      7.37      7.41      4.56
27748      7.90      7.97      5.04
27749      8.50      8.47      5.16

[14714 rows x 11 columns]
```

# 2.5

Lab 3

# 3.1

```
sort_df_price = df.sort_values('price')
print(sort_df_price)

      Unnamed: 0    carat      cut    color    clarity    depth    table    price  \
0              1    0.23      Ideal      E      SI2      61.5      55.0      326
1              2    0.21      Premium      E      SI1      59.8      61.0      326
2              3    0.23      Good      E      VS1      56.9      65.0      327
3              4    0.29      Premium      I      VS2      62.4      58.0      334
4              5    0.31      Good      J      SI2      63.3      58.0      335
...
27745      27746    2.00  Very Good      H      SI1      62.8      57.0    18803
27746      27747    2.07      Ideal      G      SI2      62.5      55.0    18804
27747      27748    1.51      Ideal      G      IF      61.7      55.0    18806
27748      27749    2.00  Very Good      G      SI1      63.5      56.0    18818
27749      27750    2.29      Premium      I      VS2      60.8      60.0    18823

      length    width    height
0          3.95      3.98      2.43
1          3.89      3.84      2.31
2          4.05      4.07      2.31
3          4.20      4.23      2.63
4          4.34      4.35      2.75
...
27745      7.95      8.00      5.01
27746      8.20      8.13      5.11
27747      7.37      7.41      4.56
27748      7.90      7.97      5.04
27749      8.50      8.47      5.16

[53940 rows x 11 columns]
```

# 3.2

```
# Add a new column named 'High_Price'
df['High_Price'] = ['High' if df.price > 5000 else 'Low' for x in df['price']]

# Print the DataFrame
print(df)
```

```
-----
ValueError                                Traceback (most recent call last)
<ipython-input-36-7ee33d420897> in <cell line: 4>()
      2
      3 # Add a new column named 'High_Price'
----> 4 df['High_Price'] = ['High' if df.price>5000 else 'Low' for x in df['price']]
      5
      6 # Print the DataFrame

----- 1 frames -----
/usr/local/lib/python3.10/dist-packages/pandas/core/generic.py in __nonzero__(self)
    1525     @final
    1526     def __nonzero__(self) -> NoReturn:
-> 1527         raise ValueError(
    1528             f"The truth value of a {type(self).__name__} is ambiguous. "
    1529             "Use a.empty, a.bool(), a.item(), a.any() or a.all()."

ValueError: The truth value of a Series is ambiguous. Use a.empty, a.bool(),
a.item(), a.any() or a.all().



SEARCH STACK OVERFLOW


```

# 3.3

```
price_mean = df.groupby('cut')['price'].mean()
print(price_mean)

cut
Fair      4358.757764
Good      3928.864452
Ideal     3457.541970
Premium   4584.257704
Very Good 3981.759891
Name: price, dtype: float64
```

# 3.4

```
df['High_price'] = df['price'].apply(lambda x: 'High' if x > 5000 else 'Low')
print(df)
```

	Unnamed: 0	carat	Cut	Shape	Color	Code	clarity	depth	table	price	\
0	1	0.23	Ideal		E	SI2	61.5	55.0	326		
1	2	0.21	Premium		E	SI1	59.8	61.0	326		
2	3	0.23	Good		E	VS1	56.9	65.0	327		
3	4	0.29	Premium		I	VS2	62.4	58.0	334		
4	5	0.31	Good		J	SI2	63.3	58.0	335		
...	...	...	...		...	...	...	...	...		
53935	53936	0.72	Ideal		D	SI1	60.8	57.0	2757		
53936	53937	0.72	Good		D	SI1	63.1	55.0	2757		
53937	53938	0.70	Very Good		D	SI1	62.8	60.0	2757		
53938	53939	0.86	Premium		H	SI2	61.0	58.0	2757		
53939	53940	0.75	Ideal		D	SI2	62.2	55.0	2757		

	length	width	height	price_scale	High_price
0	3.95	3.98	2.43	Low	Low
1	3.89	3.84	2.31	Low	Low
2	4.05	4.07	2.31	Low	Low
3	4.20	4.23	2.63	Low	Low
4	4.34	4.35	2.75	Low	Low
...	...	...	...	...	...
53935	5.75	5.76	3.50	Low	Low
53936	5.69	5.75	3.61	Low	Low
53937	5.66	5.68	3.56	Low	Low
53938	6.15	6.12	3.74	Low	Low
53939	5.83	5.87	3.64	Low	Low

[53940 rows x 13 columns]

Lab 4

# 4.1

```
subset= df[(df['price']>5000) & (df['Cut Shape']=='Premium')]
print(subset)
```

	Unnamed: 0	carat	Cut	Shape	Color	Code	clarity	depth	table	price	\
11420	11421	0.90	Premium		D	VS2	62.6	59.0	5001		
11428	11429	1.01	Premium		E	SI1	62.4	60.0	5002		
11429	11430	1.16	Premium		I	SI1	61.5	57.0	5002		
11431	11432	1.14	Premium		H	SI1	60.7	58.0	5003		
11432	11433	1.17	Premium		H	SI1	62.3	57.0	5004		
...	...	...	...		...	...	...	...	...		
27740	27741	1.71	Premium		F	VS2	62.3	59.0	18791		

```
27742      27743      2.04      Premium      H      SI1      58.1      60.0      18795
27743      27744      2.00      Premium      I      VS1      60.8      59.0      18795
27744      27745      2.29      Premium      I      SI1      61.8      59.0      18797
27749      27750      2.29      Premium      I      VS2      60.8      60.0      18823

length      width      height price_scale High_price
11420      6.14      6.17      3.85      High      High
11428      6.35      6.32      3.95      High      High
11429      6.79      6.73      4.16      High      High
11431      6.76      6.82      4.12      High      High
11432      6.75      6.71      4.19      High      High
...      ...      ...      ...      ...      ...
27740      7.57      7.53      4.70      High      High
27742      8.37      8.28      4.84      High      High
27743      8.13      8.02      4.91      High      High
27744      8.52      8.45      5.24      High      High
27749      8.50      8.47      5.16      High      High

[4717 rows x 13 columns]
```

# 4.2

```
df['Cut Shape'] = df['Cut Shape'].str.upper()
print(df)
```

```
Unnamed: 0      carat      Cut Shape      Color      Code      clarity      depth      table      price      \
0      1      0.23      IDEAL      E      SI2      61.5      55.0      326
1      2      0.21      PREMIUM      E      SI1      59.8      61.0      326
2      3      0.23      GOOD      E      VS1      56.9      65.0      327
3      4      0.29      PREMIUM      I      VS2      62.4      58.0      334
4      5      0.31      GOOD      J      SI2      63.3      58.0      335
...      ...      ...      ...      ...      ...      ...      ...      ...
53935      53936      0.72      IDEAL      D      SI1      60.8      57.0      2757
53936      53937      0.72      GOOD      D      SI1      63.1      55.0      2757
53937      53938      0.70      VERY GOOD      D      SI1      62.8      60.0      2757
53938      53939      0.86      PREMIUM      H      SI2      61.0      58.0      2757
53939      53940      0.75      IDEAL      D      SI2      62.2      55.0      2757

length      width      height price_scale High_price
0      3.95      3.98      2.43      Low      Low
1      3.89      3.84      2.31      Low      Low
2      4.05      4.07      2.31      Low      Low
3      4.20      4.23      2.63      Low      Low
4      4.34      4.35      2.75      Low      Low
...      ...      ...      ...      ...      ...
53935      5.75      5.76      3.50      Low      Low
53936      5.69      5.75      3.61      Low      Low
53937      5.66      5.68      3.56      Low      Low
53938      6.15      6.12      3.74      Low      Low
53939      5.83      5.87      3.64      Low      Low

[53940 rows x 13 columns]
```

```
# 4.3



s = df.duplicated()

print(s)
```

```
0      False
1      False
2      False
3      False
4      False
...
53935      False
53936      False
53937      False
53938      False
53939      False
Length: 53940, dtype: bool
```

# 5.1

```
df.describe()
```


	Unnamed: 0	carat	depth	table	price	length	width	height	
count	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	53940.000000	
mean	26970.500000	0.797940	61.749405	57.457184	3932.799722	5.731157	5.734526	3.538734	
std	15571.281097	0.474011	1.432621	2.234491	3989.439738	1.121761	1.142135	0.705699	
min	1.000000	0.200000	43.000000	43.000000	326.000000	0.000000	0.000000	0.000000	
25%	13485.750000	0.400000	61.000000	56.000000	950.000000	4.710000	4.720000	2.910000	
50%	26970.500000	0.700000	61.800000	57.000000	2401.000000	5.700000	5.710000	3.530000	
75%	40455.250000	1.040000	62.500000	59.000000	5324.250000	6.540000	6.540000	4.040000	
max	53940.000000	5.010000	79.000000	95.000000	18823.000000	10.740000	58.900000	31.800000	

# 5.2

```
# abc = df.groupby('cut').sum()
# print(abc, '\n')
```

5.3

```
print(df.carat.corr(df.price))
```

 0.9215913011934771