Step1. 'wine.csv' 파일로부터 데이터 불러오기

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:

```
file_path = "../data/wine.csv"
wines = pd.read_csv(file_path)
```

Step2. info() 함수로 기본 정보 불러오기

In [3]:

```
wines.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6497 entries, 0 to 6496
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   type                  6497 non-null   object 
 1   fixed acidity         6497 non-null   float64
 2   volatile acidity      6497 non-null   float64
 3   citric acid           6497 non-null   float64
 4   residual sugar        6497 non-null   float64
 5   chlorides             6497 non-null   float64
 6   free sulfur dioxide   6497 non-null   float64
 7   total sulfur dioxide  6497 non-null   float64
 8   density               6497 non-null   float64
 9   pH                    6497 non-null   float64
 10  sulphates             6497 non-null   float64
 11  alcohol               6497 non-null   float64
 12  quality               6497 non-null   int64  
dtypes: float64(11), int64(1), object(1)
memory usage: 660.0+ KB
```

Step3. 함수를 사용해 수치 통계(8개) 구하기

In [4]:

```
wines.describe()
```

Out[4]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total su dio
count	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.000000	6497.00
mean	7.215307	0.339666	0.318633	5.443235	0.056034	30.525319	115.744
std	1.296434	0.164636	0.145318	4.757804	0.035034	17.749400	56.521
min	3.800000	0.080000	0.000000	0.600000	0.009000	1.000000	6.00
25%	6.400000	0.230000	0.250000	1.800000	0.038000	17.000000	77.00
50%	7.000000	0.290000	0.310000	3.000000	0.047000	29.000000	118.00
75%	7.700000	0.400000	0.390000	8.100000	0.065000	41.000000	156.00
max	15.900000	1.580000	1.660000	65.800000	0.611000	289.000000	440.00

Step4. 함수를 사용해 수치 통계(mean, min, max) 구하기

In [5]:

```
wines.agg(['mean','min','max'])
```

Out[5]:

	type	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	de
mean	NaN	7.215307	0.339666	0.318633	5.443235	0.056034	30.525319	115.744574	0.99
min	red	3.800000	0.080000	0.000000	0.600000	0.009000	1.000000	6.000000	0.98
max	white	15.900000	1.580000	1.660000	65.800000	0.611000	289.000000	440.000000	1.03

Step5. 와인의 품질 등급(quality) 단계 알아보기

In [20]:

```
wines['quality'].unique().tolist()
```

Out[20]:

```
[5, 6, 7, 4, 8, 3, 9]
```

Step6. 와인의 품질 등급(quality)별 수(빈도수) 구하기

In [7]:

```
wines['quality'].value_counts()
```

Out[7]:

```
6    2836
5    2138
7    1079
4     216
8     193
3      30
9       5
Name: quality, dtype: int64
```

Step7. 가장 빈도수가 많은 품질 등급(quality) 구하기

In [8]:

```
wines['quality'].value_counts().idxmax()
```

Out[8]:

```
6
```

Step8. 유형(type)별로 품질 등급(quality)별 수치 통계 구하기

In [9]:

```
wines.groupby('type')['quality'].describe()
```

Out[9]:

	count	mean	std	min	25%	50%	75%	max
type								
red	1599.0	5.636023	0.807569	3.0	5.0	6.0	6.0	8.0
white	4898.0	5.877909	0.885639	3.0	5.0	6.0	6.0	9.0

Step9. 유형(type)에 따른 품질 등급(quality) 시각화 하기

In [10]:

```
result = wines.groupby('type')['quality'].agg(['mean','min','max'])
n_groups=len(result.index)
```

In [17]:

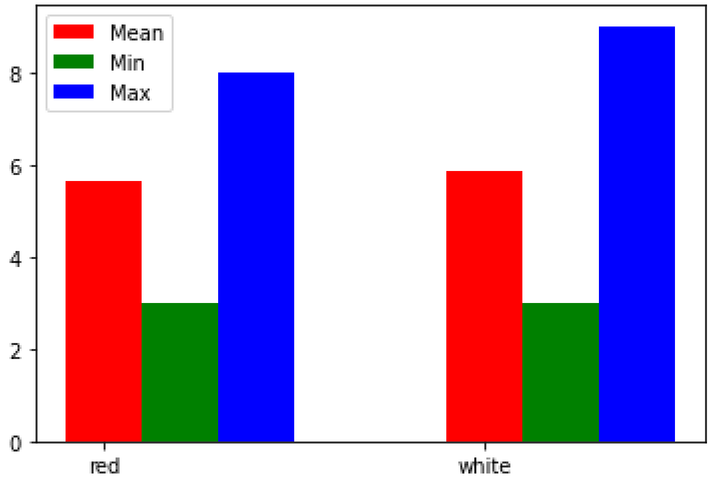
```
n_groups = len(result.index)
n_groups

means = result['mean'].tolist()
mins = result['min'].tolist()
maxs = result['max'].tolist()

index = np.arange(n_groups)
bar_width = 0.2

#평균값에 대한 바 생성
rects1 = plt.bar(index, means, bar_width, color='r', label = 'Mean')
rects2 = plt.bar(index + bar_width, mins, bar_width, color='g', label = 'Min')
rects3 = plt.bar(index + bar_width *2, maxs, bar_width, color='b', label = 'Max')


plt.xticks(index, result.index.tolist())
plt.legend() #범례
plt.show()
```



In [22]:

```
!pip install seaborn
```

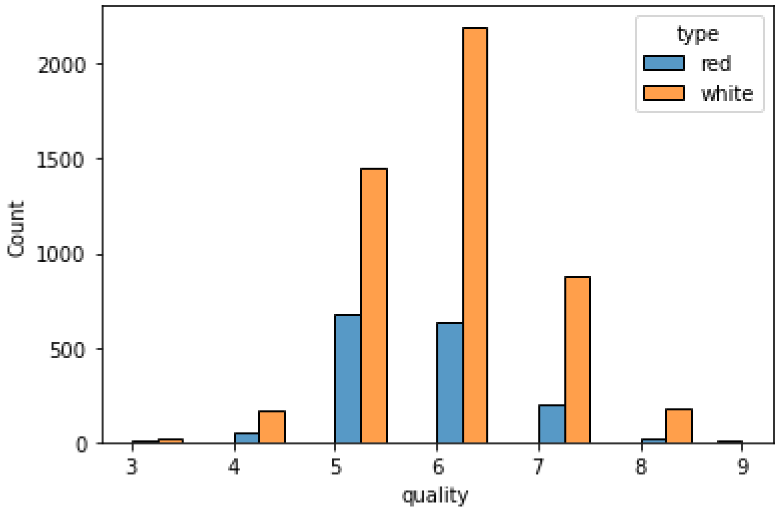
```
Requirement already satisfied: seaborn in c:₩users₩yj₩anaconda3₩lib₩site-pa
ckages (0.11.2)
Requirement already satisfied: matplotlib>=2.2 in c:₩users₩yj₩anaconda3₩lib
₩site-packages (from seaborn) (3.4.3)
Requirement already satisfied: scipy>=1.0 in c:₩users₩yj₩anaconda3₩lib₩site-
packages (from seaborn) (1.7.1)
Requirement already satisfied: pandas>=0.23 in c:₩users₩yj₩anaconda3₩lib₩s
ite-packages (from seaborn) (1.3.4)
Requirement already satisfied: numpy>=1.15 in c:₩users₩yj₩anaconda3₩lib₩s
ite-packages (from seaborn) (1.20.3)
Requirement already satisfied: pyparsing>=2.2.1 in c:₩users₩yj₩anaconda3₩li
b₩site-packages (from matplotlib>=2.2->seaborn) (3.0.4)
Requirement already satisfied: cycler>=0.10 in c:₩users₩yj₩anaconda3₩lib₩sit
e-packages (from matplotlib>=2.2->seaborn) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:₩users₩yj₩anaconda3₩li
b₩site-packages (from matplotlib>=2.2->seaborn) (1.3.1)
Requirement already satisfied: python-dateutil>=2.7 in c:₩users₩yj₩anaconda
3₩lib₩site-packages (from matplotlib>=2.2->seaborn) (2.8.2)
Requirement already satisfied: pillow>=6.2.0 in c:₩users₩yj₩anaconda3₩lib₩si
te-packages (from matplotlib>=2.2->seaborn) (8.4.0)
Requirement already satisfied: six in c:₩users₩yj₩anaconda3₩lib₩site-package
s (from cycler>=0.10->matplotlib>=2.2->seaborn) (1.16.0)
Requirement already satisfied: pytz>=2017.3 in c:₩users₩yj₩anaconda3₩lib₩si
te-packages (from pandas>=0.23->seaborn) (2021.3)
```

In [23]:

```
import seaborn as sns
sns.histplot(data = wines, x ="quality", binwidth=0.5, hue="type", multiple= "dodge")
```

Out[23]:

```
<AxesSubplot:xlabel='quality', ylabel='Count'>
```


type
red
white
Count
quality