안녕하세요, 에이블러 여러분!

AIVLE스쿨 '서울시 생활정보 기반 대중교통 수요 분석 ' 과정에 오신 여러분을 환영합니다.

- 본 과정에서는 실제 사례와 데이터를 기반으로 문제를 해결하는 전체 과정을 자기 주도형 실습으로 진행해볼 예정입니다.
- 앞선 교육과정을 정리하는 마음과 지금까지 배운 내용을 바탕으로 문제 해결을 해볼게요!
- 미니 프로젝트를 통한 문제 해결 과정 'A에서 Z까지', 지금부터 시작합니다!

데이터 분석부터 먼저 시작해보겠습니다.

"버스 정류장 데이터" 를 확인해 보도록 하겠습니다

In [1]:
```
# 필요 라이브러리부터 설치합니다.
%pip install pandas seaborn
```

```
Requirement already satisfied: pandas in c:\users\user\anaconda3\lib\site-packages (2.0.3)
Requirement already satisfied: seaborn in c:\users\user\anaconda3\lib\site-packages (0.13.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\user\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\user\anaconda3\lib\site-packages (from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\user\anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: numpy>=1.21.0 in c:\users\user\anaconda3\lib\site-packages (from pandas) (1.24.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in c:\users\user\anaconda3\lib\site-packages (from seaborn) (3.8.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\user\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.0.5)
Requirement already satisfied: cycler>=0.10 in c:\users\user\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\user\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.25.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\user\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\user\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (23.1)
Requirement already satisfied: pillow>=8 in c:\users\user\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (10.0.1)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\user\anaconda3\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.0.9)
Requirement already satisfied: six>=1.5 in c:\users\user\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

기본전제

- 제공되는 기본/추가 데이터는 '에이블러용' 폴더에 있습니다.

[기본 데이터]

- 1.1 bus_station_boarding_month_202401.csv
- 서울시 버스노선별 정류장별 승하차 인원 정보
- https://data.seoul.go.kr/dataList/OA-12912/S/1/datasetView.do

[추가 데이터]

- 1.1 bus_station_202401.xlsx
- 서울시 버스정류장 위치정보
- https://data.seoul.go.kr/dataList/OA-15067/S/1/datasetView.do

---

1.데이터 불러오기

모든 미니 프로젝트의 시작은 '데이터 불러오기' 부터입니다.

데이터 프레임을 불러오고 변수로 저장(CSV 기준으로 진행)

- csv : pd.read_csv("파일이름. csv")
- txt : pd.read_csv("파일이름. csv", sep="구분자")
- xlsx : pd.read_excel('파일이름.xlsx')
- pickle : pd.read_pickle("파일이름.pkl")
  [참고] pickle은 sklearn 라이브러리를 통해 모델을 학습시키고 저장할 때 많이 사용, 파이썬의 모든 객체를 파일로 저장할 수 있다.

[실습문제1] 데이터 로딩

- Pandas 라이브러리를 활용해서 '1.1 bus_station_boarding_month_202401.csv'파일을 'bus_station' 변수에 저장하세요.
  - 데이터 파일 로딩시 참고 사항
    - 구분자(sep)는 ',' 입니다
    - cp949 인코더를 사용해 주세요

In [2]:
```
# 아래 실습코드를 실행해주세요.
import sys
```

```
import numpy as np
import pandas as pd
```

In [3]:
```
# 아래에 실습코드를 작성하고 결과를 확인합니다.
bus_station = pd.read_csv('1.1 bus_station_boarding_month_202401.csv', sep=",", encoding = "cp
```

In [4]:
```
# 데이터 프레임의 Shape을 확인합니다.

bus_station.shape
```

Out[4]:
```
(1048575, 8)
```

2.기본 정보 확인 및 클렌징

- 데이터 클렌징 : 결측치, 이상치 등을 제거하여 데이터 분석 결과가 왜곡 되는 문제를 방지하기 위한 정제 과정

[실습문제2] 기본 정보 확인하기

- 'bus_station' 데이터의 정보를 확인해보세요.
- 'describe', 'info', 'head' 등 전부 활용해 보세요.

In [5]:
```
# head()
bus_station.head()
```

Out[5]:

	사용일자	노선번호	노선명	버스정류장ARS번호	역명	승차총승객수	하차총승객수	등록일자
0	20240101	101	101번(화계사~동대문)	6178	대광고등학교앞(00055)	39	76	20240104
1	20240101	9408	9408번(구미동차고지~고속터미널)	22337	헌인마을.서울농업기술센터(00042)	1	0	20240104
2	20240101	9408	9408번(구미동차고지~고속터미널)	22336	헌인마을.서울농업기술센터(00078)	2	0	20240104
3	20240101	9408	9408번(구미동차고지~고속터미널)	22334	헌인릉.강남서초과학화예비군훈련장(00043)	0	2	20240104
4	20240101	9408	9408번(구미동차고지~고속터미널)	22333	헌인릉.강남서초과학화예비군훈련장(00076)	2	1	20240104

In [6]:
```
# tail()
bus_station.tail()
```

Out[6]:

	사용일자	노선번호	노선명	버스정류장ARS번호	역명	승차총승객수	하차총승객수	등록일자
1048570	20240126	종로05	종로05(서대문역~배화여중고)	1837	스위스대사관(00025)	1	19	20240129
1048571	20240126	종로05	종로05(서대문역~배화여중고)	1831	월암공원(00024)	25	21	20240129
1048572	20240126	종로05	종로05(서대문역~배화여중고)	1514	적십자병원후문앞(00002)	9	6	20240129
1048573	20240126	성북07	성북07(정릉4동종점~길음역)	8454	정릉4동종점(00014)	0	181	20240129
1048574	20240126	종로05	종로05(서대문역~배화여중고)	1587	신학대학교총회본부(00011)	13	68	20240129

In [7]:
```
# 아래에 실습코드를 작성하고 결과를 확인합니다.
# info()
bus_station.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048575 entries, 0 to 1048574
Data columns (total 8 columns):
 #   Column      Non-Null Count    Dtype
---  ------      --------------    -----
 0   사용일자        1048575 non-null  int64
 1   노선번호        1048575 non-null  object
 2   노선명         1048575 non-null  object
 3   버스정류장ARS번호  1048575 non-null  object
 4   역명          1048575 non-null  object
 5   승차총승객수      1048575 non-null  int64
 6   하차총승객수      1048575 non-null  int64
 7   등록일자        1048575 non-null  int64
dtypes: int64(4), object(4)
memory usage: 64.0+ MB
```

In [8]:
```
# 아래에 실습코드를 작성하고 결과를 확인합니다.
# describe()
bus_station.describe().T
```

Out[8]:

	count	mean	std	min	25%	50%	75%	max
사용일자	1048575.0	2.024011e+07	7.455044	20240101.0	20240107.0	20240113.0	20240120.0	20240126.0
승차총승객수	1048575.0	1.030953e+02	148.680136	0.0	13.0	57.0	137.0	5027.0
하차총승객수	1048575.0	1.010178e+02	134.837425	0.0	16.0	62.0	137.0	4352.0
등록일자	1048575.0	2.024012e+07	7.455044	20240104.0	20240110.0	20240116.0	20240123.0	20240129.0

[실습문제3] 위 데이터에서 버스정류장 위치를 구 별로 구분 해보기

- 'bus_station' 데이터의 정보를 확인해보세요.
- 버스정류장 ARS번호의 앞 두자리가 구를 의미합니다.
- '자치구' column을 추가 -> 정류장이 위치한 구 이름을 등록

In [9]:
```
# "버스정류장ARS번호" 컬럼 확인하기
df= pd.DataFrame(bus_station['버스정류장ARS번호'])
df
```

Out[9]:

	버스정류장ARS번호
0	6178
1	22337
2	22336
3	22334
4	22333
...	...
1048570	1837
1048571	1831
1048572	1514
1048573	8454
1048574	1587

1048575 rows × 1 columns

In [10]:
```
# 버스정류장 ARS번호에서 시작하는 앞자리 2개로 '자치구'라는 새로운 컬럼 생성
# str() : 정수나 실수를 문자열 형태로 바꿔주는 함수, slice()

bus_station['자치구'] = bus_station['버스정류장ARS번호'].astype(str).str.zfill(5).str.slice(0,
```

In [11]:
```
bus_station.head(10)
```

Out[11]:

	사용일자	노선번호	노선명	버스정류장ARS번호	역명	승차총승객수	하차총승객수	등록일자	자치구
0	20240101	101	101번(화계사~동대문)	6178	대광고등학교앞(00055)	39	76	20240104	06
1	20240101	9408	9408번(구미동차고지~고속터미널)	22337	헌인마을.서울농업기술센터(00042)	1	0	20240104	22
2	20240101	9408	9408번(구미동차고지~고속터미널)	22336	헌인마을.서울농업기술센터(00078)	2	0	20240104	22
3	20240101	9408	9408번(구미동차고지~고속터미널)	22334	헌인릉.강남서초과학화예비군훈련장(00043)	0	2	20240104	22
4	20240101	9408	9408번(구미동차고지~고속터미널)	22333	헌인릉.강남서초과학화예비군훈련장(00076)	2	1	20240104	22
5	20240101	9408	9408번(구미동차고지~고속터미널)	22297	매헌시민의숲.양재꽃시장(00048)	6	17	20240104	22
6	20240101	9408	9408번(구미동차고지~고속터미널)	22296	매헌시민의숲.양재꽃시장(00071)	16	5	20240104	22
7	20240101	9408	9408번(구미동차고지~고속터미널)	22190	논현역6번출구(00063)	4	1	20240104	22
8	20240101	9408	9408번(구미동차고지~고속터미널)	22183	논현역7번출구(00056)	1	5	20240104	22
9	20240101	9408	9408번(구미동차고지~고속터미널)	22023	구반포역.세화고등학교(00060)	0	3	20240104	22

In [12]:
```
bus_station.tail(10)
```

Out[12]:

	사용일자	노선번호	노선명	버스정류장ARS번호	역명	승차총승객수	하차총승객수	등록일자	자치구
1048565	20240126	종로05	종로05(서대문역~배화여중고)	1849	강북삼성병원(00027)	6	175	20240129	01
1048566	20240126	종로03	종로03(낙산공원~종로5가)	1825	창신초교(00021)	63	86	20240129	01
1048567	20240126	종로05	종로05(서대문역~배화여중고)	1796	배화여중고.매동초교(00017)	22	91	20240129	01
1048568	20240126	종로05	종로05(서대문역~배화여중고)	1820	교남동주민센터(00022)	7	25	20240129	01
1048569	20240126	종로05	종로05(서대문역~배화여중고)	1554	독립문역3번출구(00006)	343	191	20240129	01
1048570	20240126	종로05	종로05(서대문역~배화여중고)	1837	스위스대사관(00025)	1	19	20240129	01
1048571	20240126	종로05	종로05(서대문역~배화여중고)	1831	월암공원(00024)	25	21	20240129	01
1048572	20240126	종로05	종로05(서대문역~배화여중고)	1514	적십자병원후문앞(00002)	9	6	20240129	01
1048573	20240126	성북07	성북07(정릉4동종점~길음역)	8454	정릉4동종점(00014)	0	181	20240129	08
1048574	20240126	종로05	종로05(서대문역~배화여중고)	1587	신학대학교총회본부(00011)	13	68	20240129	01

- 버스정류장 ARS 번호 : 01~25까지 앞 숫자 두개가 위치한 구를 의미
  1. 종로구
  2. 중구
  3. 용산구
  4. 성동구
  5. 광진구
  6. 동대문구
  7. 중랑구
  8. 성북구
  9. 강북구
  10. 도봉구
  11. 노원구

12. 은평구
13. 서대문구
14. 마포구
15. 양천구
16. 강서구
17. 구로구
18. 금천구
19. 영등포구
20. 동작구
21. 관악구
22. 서초구
23. 강남구
24. 송파구
25. 강동구

In [13]:
```
# 구 코드를 구 이름으로 변환하기
# map() : 리스트, 튜플 등 반복 가능한 데이터 집합을 입력으로 받아 변환하는 함수

bus_station['자치구'] = bus_station['자치구'].map({
    '01': '종로구',
    '02': '중구',
    '03': '용산구',
    '04': '성동구',
    '05': '광진구',
    '06': '동대문구',
    '07': '중랑구',
    '08': '성북구',
    '09': '강북구',
    '10': '도봉구',
    '11': '노원구',
    '12': '은평구',
    '13': '서대문구',
    '14': '마포구',
    '15': '양천구',
    '16': '강서구',
    '17': '구로구',
    '18': '금천구',
    '19': '영등포구',
    '20': '동작구',
    '21': '관악구',
    '22': '서초구',
    '23': '강남구',
    '24': '송파구',
    '25': '강동구'})
```

In [14]:
```
# 아래에 실습코드를 작성하고 결과를 확인합니다.
# tail(10)
bus_station.tail(10)
```

Out[14]:

	사용일자	노선번호	노선명	버스정류장ARS번호	역명	승차총승객수	하차총승객수	등록일자	자치구
1048565	20240126	종로05	종로05(서대문역~배화여중고)	1849	강북삼성병원(00027)	6	175	20240129	종로구
1048566	20240126	종로03	종로03(낙산공원~종로5가)	1825	창신초교(00021)	63	86	20240129	종로구
1048567	20240126	종로05	종로05(서대문역~배화여중고)	1796	배화여중고.매동초교(00017)	22	91	20240129	종로구
1048568	20240126	종로05	종로05(서대문역~배화여중고)	1820	교남동주민센터(00022)	7	25	20240129	종로구
1048569	20240126	종로05	종로05(서대문역~배화여중고)	1554	독립문역3번출구(00006)	343	191	20240129	종로구
1048570	20240126	종로05	종로05(서대문역~배화여중고)	1837	스위스대사관(00025)	1	19	20240129	종로구
1048571	20240126	종로05	종로05(서대문역~배화여중고)	1831	월암공원(00024)	25	21	20240129	종로구
1048572	20240126	종로05	종로05(서대문역~배화여중고)	1514	적십자병원후문앞(00002)	9	6	20240129	종로구
1048573	20240126	성북07	성북07(정릉4동종점~길음역)	8454	정릉4동종점(00014)	0	181	20240129	성북구
1048574	20240126	종로05	종로05(서대문역~배화여중고)	1587	신학대학교총회본부(00011)	13	68	20240129	종로구

[실습문제4] 결측치 처리하기

In [15]:
```
# (가상), (기점가상) 정류장은 ARS번호가 '~'로 나옵니다.
# (가상) 정류장 : 버스정보시스템 상에서 위치 정보를 표시하기 위해 임의로 가상의 정류장을 설정,
# '버스정류장ARS번호'가 '~'인 곳을 확인해주세요.

#bus_station.loc[len(bus_station)] = ['20240526', '가상05', '가상의 노선', '~', '(가상정류장F)
bus_station.loc[bus_station['버스정류장ARS번호'] == '~']
```

Out[15]:

	사용일자	노선번호	노선명	버스정류장ARS번호	역명	승차총승객수	하차총승객수	등록일자	자치구
699	20240101	441	441번(월암공영차고지~신사사거리)	~	월암차고지(종점가상)(00126)	0	1	20240104	NaN
1261	20240101	7021	7021번(은평공영차고지~롯데백화점)	~	은평공영차고지(가상)(00076)	0	10	20240104	NaN
1337	20240101	7017	7017번(은평공영차고지~롯데백화점)	~	은평공영차고지(가상)(00090)	0	10	20240104	NaN
1427	20240101	7016	7016번(은평차고지~상명대)	~	은평공영차고지(가상)(00107)	0	16	20240104	NaN
3332	20240101	750B	750B번(은평차고지~서울대)	~	은평공영차고지(가상)(00076)	0	15	20240104	NaN
...	...	...	...	...	...	...	...	...	...
1040945	20240126	271	271번(용마문화복지센터~월드컵파크7단지)	~	경성여객(종점가상)(00125)	0	35	20240129	NaN
1040953	20240126	271	271번(용마문화복지센터~월드컵파크7단지)	~	경성여객(기점가상)(00001)	61	12	20240129	NaN
1041836	20240126	N64	N64번(염곡공영차고지~강서공영차고지)	~	대흥교통(기점가상)(00001)	0	1	20240129	NaN
1042754	20240126	N72	N72중랑 (중랑공영차고지~은평공영차고지)	~	중랑공영차고지(기점가상)(00001)	3	0	20240129	NaN
1043252	20240126	01A	01A번(예장주차장~예장주차장)	~	남산예장버스환승주차장(종점가상)(00025)	2	165	20240129	NaN

4358 rows × 9 columns

In [16]:
```
bus_station = bus_station.loc[bus_station['버스정류장ARS번호'] != '~']
```

In [17]:
```
bus_station.tail(10)
```

Out[17]:

	사용일자	노선번호	노선명	버스정류장ARS번호	역명	승차총승객수	하차총승객수	등록일자	자치구
1048565	20240126	종로05	종로05(서대문역~배화여중고)	1849	강북삼성병원(00027)	6	175	20240129	종로구
1048566	20240126	종로03	종로03(낙산공원~종로5가)	1825	창신초교(00021)	63	86	20240129	종로구
1048567	20240126	종로05	종로05(서대문역~배화여중고)	1796	배화여중고.매동초교(00017)	22	91	20240129	종로구
1048568	20240126	종로05	종로05(서대문역~배화여중고)	1820	교남동주민센터(00022)	7	25	20240129	종로구
1048569	20240126	종로05	종로05(서대문역~배화여중고)	1554	독립문역3번출구(00006)	343	191	20240129	종로구
1048570	20240126	종로05	종로05(서대문역~배화여중고)	1837	스위스대사관(00025)	1	19	20240129	종로구
1048571	20240126	종로05	종로05(서대문역~배화여중고)	1831	월암공원(00024)	25	21	20240129	종로구
1048572	20240126	종로05	종로05(서대문역~배화여중고)	1514	적십자병원후문앞(00002)	9	6	20240129	종로구
1048573	20240126	성북07	성북07(정릉4동종점~길음역)	8454	정릉4동종점(00014)	0	181	20240129	성북구
1048574	20240126	종로05	종로05(서대문역~배화여중고)	1587	신학대학교총회본부(00011)	13	68	20240129	종로구

In [18]:
```
# 컬럼별 NaN 값이 있는지 확인, info()

bus_station.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1044217 entries, 0 to 1048574
Data columns (total 9 columns):
 #   Column      Non-Null Count    Dtype
---  ------      --------------    -----
 0   사용일자        1044217 non-null  int64
 1   노선번호        1044217 non-null  object
 2   노선명         1044217 non-null  object
 3   버스정류장ARS번호  1044217 non-null  object
 4   역명          1044217 non-null  object
 5   승차총승객수      1044217 non-null  int64
 6   하차총승객수      1044217 non-null  int64
 7   등록일자        1044217 non-null  int64
 8   자치구         960811 non-null   object
dtypes: int64(4), object(5)
memory usage: 79.7+ MB
```

In [19]:
```
# NaN 값을 제거 해주세요.
# dropna()
bus_station['자치구'].isna().sum()
```

Out[19]:
```
83406
```

In [20]:
```
# 컬럼별 NaN 값이 있는지 확인, info()
bus_station.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1044217 entries, 0 to 1048574
Data columns (total 9 columns):
 #   Column      Non-Null Count    Dtype
---  ------      --------------    -----
 0   사용일자        1044217 non-null  int64
 1   노선번호        1044217 non-null  object
 2   노선명         1044217 non-null  object
 3   버스정류장ARS번호  1044217 non-null  object
 4   역명          1044217 non-null  object
 5   승차총승객수      1044217 non-null  int64
 6   하차총승객수      1044217 non-null  int64
 7   등록일자        1044217 non-null  int64
 8   자치구         960811 non-null   object
dtypes: int64(4), object(5)
memory usage: 79.7+ MB
```

In [21]:
```
# "버스정류장ARS번호" dtype을 정수형(int)으로 변경
bus_station['버스정류장ARS번호'] = bus_station['버스정류장ARS번호'].astype('int64')
```

In [22]:
```
bus_station.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1044217 entries, 0 to 1048574
Data columns (total 9 columns):
 #   Column      Non-Null Count    Dtype
---  ------      --------------    -----
 0   사용일자        1044217 non-null  int64
 1   노선번호        1044217 non-null  object
 2   노선명         1044217 non-null  object
 3   버스정류장ARS번호  1044217 non-null  int64
 4   역명          1044217 non-null  object
 5   승차총승객수      1044217 non-null  int64
 6   하차총승객수      1044217 non-null  int64
 7   등록일자        1044217 non-null  int64
 8   자치구         960811 non-null   object
dtypes: int64(5), object(4)
memory usage: 79.7+ MB
```

[실습문제5] 구별로 버스정류장의 개수 확인하기 (서울시)

In [23]:
```
bus_station.head(2)
```

Out[23]:

	사용일자	노선번호	노선명	버스정류장ARS번호	역명	승차총승객수	하차총승객수	등록일자	자치구
0	20240101	101	101번(화계사~동대문)	6178	대광고등학교앞(00055)	39	76	20240104	동대문구
1	20240101	9408	9408번(구미동차고지~고속터미널)	22337	헌인마을.서울농업기술센터(00042)	1	0	20240104	서초구

In [24]:
```
# 자치구별 버스정류장 고유값들의 갯수를 출력하여 'bus_station_count' 변수로 저장
# nunique(), groupby()

bus_station.groupby(by='자치구', as_index=False)['버스정류장ARS번호'].nunique()
```

Out[24]:

	자치구	버스정류장ARS번호
0	강남구	501
1	강동구	367
2	강북구	410
3	강서구	567
4	관악구	468
5	광진구	280
6	구로구	486
7	금천구	347
8	노원구	496
9	도봉구	368
10	동대문구	308
11	동작구	441
12	마포구	565
13	서대문구	460
14	서초구	601
15	성동구	433
16	성북구	599
17	송파구	470
18	양천구	323
19	영등포구	467
20	용산구	326
21	은평구	497
22	종로구	357
23	중구	174
24	중랑구	428

In [25]:
```
# 중랑구에 428개의 버스정류장이 있다는데, 실제 ARS번호를 확인해봅시다.
# 자치구별 버스정류장 고유값들을 출력하여 'bus_staiton_unique' 변수로 저장
# 중랑구 버스정류장ARS번호 전체 출력

bus_staiton_unique = bus_station.loc[bus_station['자치구'] == '중랑구']['버스정류장ARS번호'].u
```

In [26]:
```
bus_staiton_unique
```

Out[26]:
```
<bound method Series.unique of 182        7142
183        7136
184        7138
185        7465
186        7440
           ...
1042698    7009
1042699    7418
1042700    7010
1042702    7007
1042703    7554
Name: 버스정류장ARS번호, Length: 41629, dtype: int64>
```

In [27]:
```
# 서울 지역 외 '버스정류장ARS번호' 샘플 확인
bus_station.loc[~bus_station['자치구'].isin(['종로구',
'중구',
'용산구',
'성동구',
'광진구',
'동대문구',
'중랑구',
'성북구',
'강북구',
'도봉구',
'노원구',
'은평구',
'서대문구',
'마포구',
'양천구',
'강서구',
'구로구',
'금천구',
'영등포구',
'동작구',
'관악구',
'서초구',
'강남구',
'송파구',
'강동구'])]
# 없는 듯
```

Out[27]:

	사용일자	노선번호	노선명	버스정류장ARS번호	역명	승차총승객수	하차총승객수	등록일자	자치구
29	20240101	9701	9701번(가좌동~서울역)	36735	고양예고입구(00007)	10	0	20240104	NaN
30	20240101	9701	9701번(가좌동~서울역)	36734	고양예고입구(00122)	0	25	20240104	NaN
31	20240101	9701	9701번(가좌동~서울역)	36349	가좌동종점(00001)	2	0	20240104	NaN
32	20240101	9701	9701번(가좌동~서울역)	36701	알미공원(중)(00020)	18	2	20240104	NaN
33	20240101	9701	9701번(가좌동~서울역)	36330	송산3통(00008)	3	0	20240104	NaN
...	...	...	...	...	...	...	...	...	...
1045311	20240126	도봉09	도봉09(창동역~도봉산역)	61500	수락리버시티2단지정문(00023)	222	80	20240129	NaN
1045327	20240126	도봉09	도봉09(창동역~도봉산역)	61501	수락리버시티1단지후문(00032)	32	24	20240129	NaN
1045335	20240126	도봉09	도봉09(창동역~도봉산역)	61501	수락리버시티1단지후문(00024)	103	15	20240129	NaN
1047179	20240126	서초20	서초20(서초더샵포레아파트~양재역)	37501	잔디마을(00026)	5	76	20240129	NaN
1047337	20240126	서초08	서초08(서울추모공원~양재역)	37501	잔디마을(00025)	1	375	20240129	NaN

83406 rows × 9 columns

In [28]:
```
# 중랑구의 버스정류장ARS번호 7674 인 정류장 확인해보기
bus_station.loc[(bus_station['자치구'] == '중랑구') & (bus_station['버스정류장ARS번호'] == 767
```

Out[28]:

	역명
43608	이우중고(00005)
84449	이우중고(00005)
206641	이우중고(00005)
287124	이우중고(00005)
327856	이우중고(00005)
450342	이우중고(00005)
612051	이우중고(00005)
652849	이우중고(00005)
693641	이우중고(00005)
814818	이우중고(00005)
855341	이우중고(00005)
896148	이우중고(00005)
977959	이우중고(00005)
1018890	이우중고(00005)

In [29]:
```
# 서울의 버스 정류장 데이터만 포함하고 있는 excel 파일 열기 ('1.1 bus_station_202401.xlsx')
# https://data.seoul.go.kr/dataList/OA-15067/S/1/datasetView.do (출처:서울열린데이터광장)
# 'only_seoul' 변수로 저장

only_seoul = pd.read_excel('1.1 bus_station_202401.xlsx')
```

In [30]:
```
# 데이터 구조 확인, info()

only_seoul.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11263 entries, 0 to 11262
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   ARS_ID  11263 non-null  int64 
 1   정류소명    11263 non-null  object
dtypes: int64(1), object(1)
memory usage: 176.1+ KB
```

In [31]:
```
# 데이터 확인

only_seoul.head()
```

Out[31]:

	ARS_ID	정류소명
0	1001	종로2가사거리
1	1002	창경궁.서울대학교병원
2	1003	명륜3가.성대입구
3	1004	종로2가.삼일교
4	1005	혜화동로터리.여운형활동터

In [32]:
```
# 서울 지역 외 '버스정류장ARS번호' 샘플 확인

only_seoul['자치구'] = only_seoul['ARS_ID'].astype(str).str.zfill(5).str.slice(0, 2)
only_seoul['자치구'] = only_seoul['자치구'].astype('int64')
only_seoul.loc[only_seoul['자치구'] > 25] # 없음
```

Out[32]:

	ARS_ID	정류소명	자치구

In [33]:
```
only_seoul.drop('자치구', axis=1, inplace=True)
```

In [34]:
```
only_seoul.head()
```

Out[34]:

	ARS_ID	정류소명
0	1001	종로2가사거리
1	1002	창경궁.서울대학교병원
2	1003	명륜3가.성대입구
3	1004	종로2가.삼일교
4	1005	혜화동로터리.여운형활동터

In [35]:
```
# 'ARS-ID'열 이름을 '버스정류장ARS번호'로 바꾸기


only_seoul.columns = ['버스정류장ARS번호', '정류소명']
```

In [36]:
```
only_seoul.head()
```

Out[36]:

	버스정류장ARS번호	정류소명
0	1001	종로2가사거리
1	1002	창경궁.서울대학교병원
2	1003	명륜3가.성대입구
3	1004	종로2가.삼일교
4	1005	혜화동로터리.여운형활동터

In [37]:
```
# 'only_seoul'과 'bus_station' 데이터 병합

seoul_bus_station =  pd.merge(only_seoul, bus_station, on='버스정류장ARS번호', how='inner')
```

In [38]:
```
seoul_bus_station.head()
```

Out[38]:

	버스정류장ARS번호	정류소명	사용일자	노선번호	노선명	역명	승차총승객수	하차총승객수	등록일자	자치구
0	1001	종로2가사거리	20240101	N37	N37번(송파공영차고지~진관공영차고지)	종로2가사거리(00032)	3	5	20240104	종로구
1	1001	종로2가사거리	20240101	N37	N37번(진관공영차고지~송파공영차고지)	종로2가사거리(00089)	21	11	20240104	종로구
2	1001	종로2가사거리	20240101	470	470번(상암차고지~안골마을)	종로2가사거리(00066)	98	117	20240104	종로구
3	1001	종로2가사거리	20240101	741	741번(진관차고지~헌인릉입구)	종로2가사거리(00075)	92	116	20240104	종로구
4	1001	종로2가사거리	20240102	N37	N37번(송파공영차고지~진관공영차고지)	종로2가사거리(00032)	2	11	20240105	종로구

In [39]:
```
seoul_bus_station.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 957609 entries, 0 to 957608
Data columns (total 10 columns):
 #   Column      Non-Null Count   Dtype 
---  ------      --------------   ----- 
 0   버스정류장ARS번호  957609 non-null  int64 
 1   정류소명        957609 non-null  object
 2   사용일자        957609 non-null  int64 
 3   노선번호        957609 non-null  object
 4   노선명         957609 non-null  object
 5   역명          957609 non-null  object
 6   승차총승객수      957609 non-null  int64 
 7   하차총승객수      957609 non-null  int64 
 8   등록일자        957609 non-null  int64 
 9   자치구         957609 non-null  object
dtypes: int64(5), object(5)
memory usage: 73.1+ MB
```

In [40]:
```
# 구 별로 버스 정류장의 개수 확인하기, 'seoul_bus_station_ARS' 변수로 저장

seoul_bus_station_ARS = seoul_bus_station.groupby(by='자치구', as_index=False)['버스정류장ARS번
```

In [41]:
```
seoul_bus_station_ARS
```

Out[41]:

	자치구	버스정류장ARS번호
0	강남구	501
1	강동구	367
2	강북구	410
3	강서구	567
4	관악구	466
5	광진구	274
6	구로구	486
7	금천구	346
8	노원구	495
9	도봉구	366
10	동대문구	307
11	동작구	435
12	마포구	558
13	서대문구	456
14	서초구	601
15	성동구	432
16	성북구	595
17	송파구	470
18	양천구	319
19	영등포구	465
20	용산구	326
21	은평구	497
22	종로구	356
23	중구	173
24	중랑구	384

In [42]:
```
# 서울시에 있는 버스정류장 개수 구하기

seoul_bus_station_ARS['버스정류장ARS번호'].sum()
```

Out[42]:
```
10652
```

[실습문제6] 구 별로 버스 노선의 개수 확인하기

In [43]:
```
seoul_bus_station.head()
```

Out[43]:

	버스정류장ARS번호	정류소명	사용일자	노선번호	노선명	역명	승차총승객수	하차총승객수	등록일자	자치구
0	1001	종로2가사거리	20240101	N37	N37번(송파공영차고지~진관공영차고지)	종로2가사거리(00032)	3	5	20240104	종로구
1	1001	종로2가사거리	20240101	N37	N37번(진관공영차고지~송파공영차고지)	종로2가사거리(00089)	21	11	20240104	종로구
2	1001	종로2가사거리	20240101	470	470번(상암차고지~안골마을)	종로2가사거리(00066)	98	117	20240104	종로구
3	1001	종로2가사거리	20240101	741	741번(진관차고지~헌인릉입구)	종로2가사거리(00075)	92	116	20240104	종로구
4	1001	종로2가사거리	20240102	N37	N37번(송파공영차고지~진관공영차고지)	종로2가사거리(00032)	2	11	20240105	종로구

In [44]:
```
# 자치구별 노선번호의 개수 구하기, 'seoul_bus_staion_line' 변수로 저장

seoul_bus_station_line = seoul_bus_station.groupby(by='자치구', as_index=False)['노선번호'].nu
```

In [45]:
```
seoul_bus_station_line
```

Out[45]:

	자치구	노선번호
0	강남구	98
1	강동구	22
2	강북구	71
3	강서구	51
4	관악구	81
5	광진구	43
6	구로구	80
7	금천구	56
8	노원구	58
9	도봉구	54
10	동대문구	74
11	동작구	93
12	마포구	100
13	서대문구	109
14	서초구	99
15	성동구	58
16	성북구	98
17	송파구	60
18	양천구	57
19	영등포구	100
20	용산구	69
21	은평구	74
22	종로구	105
23	중구	102
24	중랑구	50

[실습문제7] 각 구별로 승차 총 승객수, 하차 총 승객수 구하기

In [46]:
```
seoul_bus_station.head()
```

Out[46]:

	버스정류장ARS번호	정류소명	사용일자	노선번호	노선명	역명	승차총승객수	하차총승객수	등록일자	자치구
0	1001	종로2가사거리	20240101	N37	N37번(송파공영차고지~진관공영차고지)	종로2가사거리(00032)	3	5	20240104	종로구
1	1001	종로2가사거리	20240101	N37	N37번(진관공영차고지~송파공영차고지)	종로2가사거리(00089)	21	11	20240104	종로구
2	1001	종로2가사거리	20240101	470	470번(상암차고지~안골마을)	종로2가사거리(00066)	98	117	20240104	종로구
3	1001	종로2가사거리	20240101	741	741번(진관차고지~헌인릉입구)	종로2가사거리(00075)	92	116	20240104	종로구
4	1001	종로2가사거리	20240102	N37	N37번(송파공영차고지~진관공영차고지)	종로2가사거리(00032)	2	11	20240105	종로구

In [47]:
```
# 자치구별 "승차총승객수","하차총승객수"의 합 구하기, 'seoul_bus_station_sum' 변수로 저장
# groupby()

seoul_bus_station_sum = seoul_bus_station.groupby(by='자치구', as_index=False)[['승차총승객수'
```

In [48]:
```
seoul_bus_station_sum
```

Out[48]:

	자치구	승차총승객수	하차총승객수
0	강남구	6960336	6597087
1	강동구	2515582	2460905
2	강북구	3998077	3858057
3	강서구	3981173	3963283
4	관악구	5862490	5996915
5	광진구	2365355	2382437
6	구로구	4204229	4023135
7	금천구	3133461	3014364
8	노원구	3553485	3497450
9	도봉구	2771076	2695540
10	동대문구	4530373	4426822
11	동작구	4125466	3979142
12	마포구	4517711	4395035
13	서대문구	5069197	4978622
14	서초구	5919552	5755058
15	성동구	2371151	2331836
16	성북구	5051648	5060251
17	송파구	4746623	4744766
18	양천구	3478159	3344254
19	영등포구	4987187	5041026
20	용산구	3495720	3422432
21	은평구	4296944	4223666
22	종로구	4500874	4348356
23	중구	3138112	3050645
24	중랑구	3200879	3155827

[실습문제8] 각 구별 승차 평균 승객수, 하차 평균 승객수 구하기

In [49]:
```
# 자치구별 "승차총승객수","하차총승객수"의 평균 구하기, 'seoul_bus_station_mean' 변수로 저장
# '승차총승객수' -> '승차평균승객수', '하차총승객수' -> '하차평균승객수'로 열이름 변경

seoul_bus_station_mean = seoul_bus_station.groupby(by='자치구', as_index=False)[['승차총승객수
seoul_bus_station_mean.columns = ['자치구', '승차평균승객수', '하차평균승객수']
```

In [50]:
```
seoul_bus_station_mean
```

Out[50]:

	자치구	승차평균승객수	하차평균승객수
0	강남구	123.257234	116.824633
1	강동구	91.578944	89.588445
2	강북구	126.465395	122.036345
3	강서구	86.932767	86.542122
4	관악구	134.946712	138.040996
5	광진구	107.491706	108.267985
6	구로구	114.700415	109.759781
7	금천구	123.088384	118.410025
8	노원구	83.356439	82.041989
9	도봉구	101.649829	98.878985
10	동대문구	127.573018	124.657074
11	동작구	108.467845	104.620655
12	마포구	95.578543	92.983159
13	서대문구	104.720330	102.849216
14	서초구	113.621221	110.463886
15	성동구	90.495039	88.994581
16	성북구	112.288788	112.480017
17	송파구	99.003483	98.964751
18	양천구	96.452095	92.738804
19	영등포구	116.711217	117.971168
20	용산구	119.962938	117.447907
21	은평구	93.725603	92.127252
22	종로구	123.298104	119.119987
23	중구	113.699710	110.530616
24	중랑구	79.082866	77.969784

[실습문제9] 데이터 프레임 합치기

In [51]:
```
# 네 개 파일을 합쳐주세요.

# seoul_bus_station_ARS
# seoul_bus_station_line
# seoul_bus_station_sum
# seoul_bus_station_mean

seoul_bus_station_info = pd.merge(seoul_bus_station_ARS, seoul_bus_station_line, on='자치구',
seoul_bus_station_info = pd.merge(seoul_bus_station_info, seoul_bus_station_sum, on='자치구',
seoul_bus_station_info = pd.merge(seoul_bus_station_info, seoul_bus_station_mean, on='자치구',
```

In [52]:
```
seoul_bus_station_info
```

Out[52]:

	자치구	버스정류장ARS번호	노선번호	승차총승객수	하차총승객수	승차평균승객수	하차평균승객수
0	강남구	501	98	6960336	6597087	123.257234	116.824633
1	강동구	367	22	2515582	2460905	91.578944	89.588445
2	강북구	410	71	3998077	3858057	126.465395	122.036345
3	강서구	567	51	3981173	3963283	86.932767	86.542122
4	관악구	466	81	5862490	5996915	134.946712	138.040996
5	광진구	274	43	2365355	2382437	107.491706	108.267985
6	구로구	486	80	4204229	4023135	114.700415	109.759781
7	금천구	346	56	3133461	3014364	123.088384	118.410025
8	노원구	495	58	3553485	3497450	83.356439	82.041989
9	도봉구	366	54	2771076	2695540	101.649829	98.878985
10	동대문구	307	74	4530373	4426822	127.573018	124.657074
11	동작구	435	93	4125466	3979142	108.467845	104.620655
12	마포구	558	100	4517711	4395035	95.578543	92.983159
13	서대문구	456	109	5069197	4978622	104.720330	102.849216
14	서초구	601	99	5919552	5755058	113.621221	110.463886
15	성동구	432	58	2371151	2331836	90.495039	88.994581
16	성북구	595	98	5051648	5060251	112.288788	112.480017
17	송파구	470	60	4746623	4744766	99.003483	98.964751
18	양천구	319	57	3478159	3344254	96.452095	92.738804
19	영등포구	465	100	4987187	5041026	116.711217	117.971168
20	용산구	326	69	3495720	3422432	119.962938	117.447907
21	은평구	497	74	4296944	4223666	93.725603	92.127252
22	종로구	356	105	4500874	4348356	123.298104	119.119987
23	중구	173	102	3138112	3050645	113.699710	110.530616
24	중랑구	384	50	3200879	3155827	79.082866	77.969784

In [53]:
```
# '버스정류장ARS번호' -> '정류장수', '노선번호' -> '노선수'로 열이름 변경

seoul_bus_station_info.columns = ['자치구', '정류장수', '노선수', '승차총승객수', '하차총승객수
```

In [54]:
```
# 데이터를 합친 결과를 확인합니다.

seoul_bus_station_info
```

Out[54]:

	자치구	정류장수	노선수	승차총승객수	하차총승객수	승차평균승객수	하차평균승객수
0	강남구	501	98	6960336	6597087	123.257234	116.824633
1	강동구	367	22	2515582	2460905	91.578944	89.588445
2	강북구	410	71	3998077	3858057	126.465395	122.036345
3	강서구	567	51	3981173	3963283	86.932767	86.542122
4	관악구	466	81	5862490	5996915	134.946712	138.040996
5	광진구	274	43	2365355	2382437	107.491706	108.267985
6	구로구	486	80	4204229	4023135	114.700415	109.759781
7	금천구	346	56	3133461	3014364	123.088384	118.410025
8	노원구	495	58	3553485	3497450	83.356439	82.041989
9	도봉구	366	54	2771076	2695540	101.649829	98.878985
10	동대문구	307	74	4530373	4426822	127.573018	124.657074
11	동작구	435	93	4125466	3979142	108.467845	104.620655
12	마포구	558	100	4517711	4395035	95.578543	92.983159
13	서대문구	456	109	5069197	4978622	104.720330	102.849216
14	서초구	601	99	5919552	5755058	113.621221	110.463886
15	성동구	432	58	2371151	2331836	90.495039	88.994581
16	성북구	595	98	5051648	5060251	112.288788	112.480017
17	송파구	470	60	4746623	4744766	99.003483	98.964751
18	양천구	319	57	3478159	3344254	96.452095	92.738804
19	영등포구	465	100	4987187	5041026	116.711217	117.971168
20	용산구	326	69	3495720	3422432	119.962938	117.447907
21	은평구	497	74	4296944	4223666	93.725603	92.127252
22	종로구	356	105	4500874	4348356	123.298104	119.119987
23	중구	173	102	3138112	3050645	113.699710	110.530616
24	중랑구	384	50	3200879	3155827	79.082866	77.969784

In [55]:
```
# 해당 데이터프레임을 csv 파일로 저장하세요.

seoul_bus_station_info.to_csv('seoul_bus_station_info.csv', index=False)
```

3.데이터 분석하기

- KeyPoint : 데이터의 형태를 살펴보고 다양한 분석기법을 통해 모델링에 적합하도록 정제요소를 선별할 수 있다.

- 데이터들의 패턴 탐색
- 변수들간의 관계 파악

In [56]:
```
# 시각화 한글폰트 설정
import seaborn as sns
import matplotlib.pyplot as plt

plt.rc('font', family='Malgun')
sns.set(font="Malgun Gothic",
        rc={"axes.unicode_minus":False}, # 마이너스 부호 깨짐 현상 해결
        style='darkgrid')
```

[실습문제10] 데이터 분포 알아보기

- 다양한 변수를 기준으로 그래프를 그려보고 인사이트를 도출해보세요.

In [57]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [58]:
```
# 자치구별 정류장 수를 볼 수 있는 그래프를 출력해주세요.
bus =  pd.read_csv('seoul_bus_station_info.csv')
```

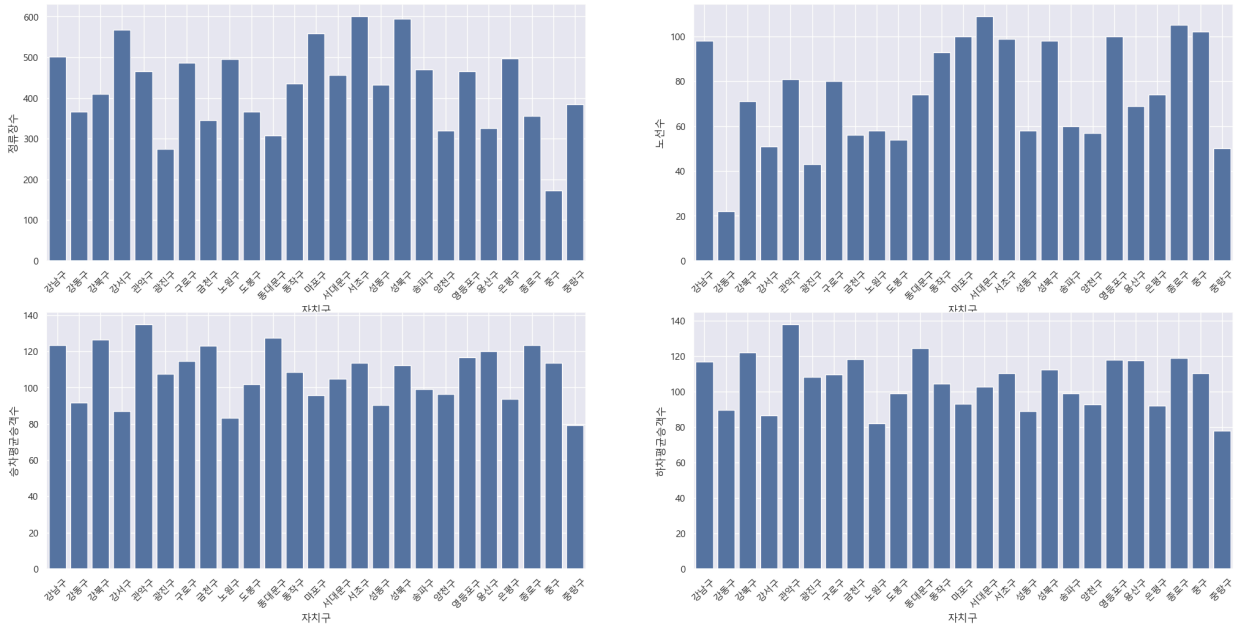
In [61]:
```
bus.head()
```

Out[61]:

	자치구	정류장수	노선수	승차총승객수	하차총승객수	승차평균승객수	하차평균승객수
0	강남구	501	98	6960336	6597087	123.257234	116.824633
1	강동구	367	22	2515582	2460905	91.578944	89.588445
2	강북구	410	71	3998077	3858057	126.465395	122.036345
3	강서구	567	51	3981173	3963283	86.932767	86.542122
4	관악구	466	81	5862490	5996915	134.946712	138.040996

In [65]:
```
plt.figure(figsize=(25, 12))

plt.subplot(2, 2, 1)
sns.barplot(x='자치구', y='정류장수', data=bus)
plt.xticks(rotation=45)
plt.grid(axis='x')

plt.subplot(2, 2, 2)
sns.barplot(x='자치구', y='노선수', data=bus)
plt.xticks(rotation=45)
plt.grid(axis='x')

plt.subplot(2, 2, 3)
sns.barplot(x='자치구', y='승차평균승객수', data=bus)
plt.xticks(rotation=45)
plt.grid(axis='x')

plt.subplot(2, 2, 4)
sns.barplot(x='자치구', y='하차평균승객수', data=bus)
plt.xticks(rotation=45)
plt.grid(axis='x')
```

```
plt.show()
```



In [60]:
```
# 위 차트를 통해 알게된 사실을 개인별로 정리해봅시다.
# 1. 서초구가 가장 정류장 수가 많다.
# 2. 중구가 가장 정류장 수가 적다.
# 3. 면적과 유동량의 따라 정류장수가 관계가 있는 것같다.
# 4. 승차, 하차 비슷 출,퇴근이 대부분인 것같음
```
