미니 프로젝트: 택시요금 데이터 다루기

데이터 불러오기, 데이터 확인

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

```
data = pd.read_csv('data/trip.csv')
```

```
data.head()
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distance	fa
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.34	
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.80	
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.00	
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.70	
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.37	

```
# Q. info() 메서드를 사용하여 데이터 컬럼명과 자료형을 확인합니다.

data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22701 entries, 0 to 22700
Data columns (total 9 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   passenger_name         22701 non-null  object
 1   tpep_pickup_datetime   22701 non-null  object
 2   tpep_dropoff_datetime  22701 non-null  object
 3   payment_method         22701 non-null  object
 4   passenger_count        22701 non-null  int64
 5   trip_distance          22701 non-null  float64
 6   fare_amount            22698 non-null  float64
 7   tip_amount             22701 non-null  float64
 8   tolls_amount           22701 non-null  float64
dtypes: float64(4), int64(1), object(4)
memory usage: 1.6+ MB
```

Q. describe() 메서드를 사용하여 데이터 컬럼별 통계량을 확인합니다.

data.describe()

	passenger_count	trip_distance	fare_amount	tip_amount	tolls_amount
count	22701.000000	22701.000000	22698.000000	22701.000000	22701.000000
mean	1.643584	2.913400	13.024009	1.835745	0.312514
std	1.304942	3.653023	13.240074	2.800537	1.399153
min	0.000000	0.000000	-120.000000	0.000000	0.000000
25%	1.000000	0.990000	6.500000	0.000000	0.000000
50%	1.000000	1.610000	9.500000	1.350000	0.000000
75%	2.000000	3.060000	14.500000	2.450000	0.000000
max	36.000000	33.960000	999.990000	200.000000	19.100000

중복 데이터 확인

Q. 중복 데이터를 확인합니다.

```
data[data.duplicated()]
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distance
17	Sarah Gross	08/15/2017 7:48:08 PM	08/15/2017 8:00:37 PM	Cash	1	3.6
204	Lisa Bullock	02/13/2017 4:25:41 PM	02/13/2017 4:55:35 PM	Cash	1	4.2

```
# Q. 중복 데이터를 확인합니다.
# 위에서 확인한 중복 데이터의 승객명을 [[PASSENGER_NAME]] 대신 넣어주세요.

data[data['passenger_name'] == 'Sarah Gross']
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distance	f
16	Sarah Gross	08/15/2017 7:48:08 PM	08/15/2017 8:00:37 PM	Cash	1	3.6	
17	Sarah Gross	08/15/2017 7:48:08 PM	08/15/2017 8:00:37 PM	Cash	1	3.6	

```
# Q. 중복 데이터를 제거합니다.

data = data.drop_duplicates()
```

```
data
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.3
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.8
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.0
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.7
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.3
...	...	...	...	...	...	.
22696	Austin Johnson	02/24/2017 5:37:23 PM	02/24/2017 5:40:39 PM	Cash	3	0.6
22697	Monique Williams	08/06/2017 4:43:59 PM	08/06/2017 5:24:47 PM	Cash	1	16.7
22698	Drew Graves	09/04/2017 2:54:14 PM	09/04/2017 2:58:22 PM	Debit Card	1	0.4
22699	Jonathan Copeland	07/15/2017 12:56:30 PM	07/15/2017 1:08:26 PM	Debit Card	1	2.3
22700	Benjamin Miller	03/02/2017 1:02:49 PM	03/02/2017 1:16:09 PM	Cash	1	2.1

22699 rows × 9 columns

결측치 확인

```
data.isna().sum()
```

```
passenger_name          0
tpep_pickup_datetime    0
tpep_dropoff_datetime   0
payment_method          0
passenger_count         0
trip_distance           0
fare_amount             3
tip_amount              0
tolls_amount            0
dtype: int64
```

Q. 전체 데이터 대비 결측치의 비율을 확인합니다.

```
data.isna().mean()
```

```
passenger_name          0.000000
tpep_pickup_datetime    0.000000
tpep_dropoff_datetime   0.000000
payment_method          0.000000
passenger_count         0.000000
trip_distance           0.000000
fare_amount             0.000132
tip_amount              0.000000
tolls_amount            0.000000
dtype: float64
```

Q. 결측치를 제거합니다.

```
data = data.dropna()
data
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.3
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.8
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.0
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.7
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.3
...	...	...	...	...	...	...
22696	Austin Johnson	02/24/2017 5:37:23 PM	02/24/2017 5:40:39 PM	Cash	3	0.6
22697	Monique Williams	08/06/2017 4:43:59 PM	08/06/2017 5:24:47 PM	Cash	1	16.7
22698	Drew Graves	09/04/2017 2:54:14 PM	09/04/2017 2:58:22 PM	Debit Card	1	0.4
22699	Jonathan Copeland	07/15/2017 12:56:30 PM	07/15/2017 1:08:26 PM	Debit Card	1	2.3
22700	Benjamin Miller	03/02/2017 1:02:49 PM	03/02/2017 1:16:09 PM	Cash	1	2.1

22696 rows × 9 columns

```
data.isna().mean()
```

```
passenger_name          0.0
tpep_pickup_datetime    0.0
tpep_dropoff_datetime   0.0
payment_method          0.0
passenger_count         0.0
trip_distance           0.0
fare_amount             0.0
tip_amount              0.0
tolls_amount            0.0
dtype: float64
```

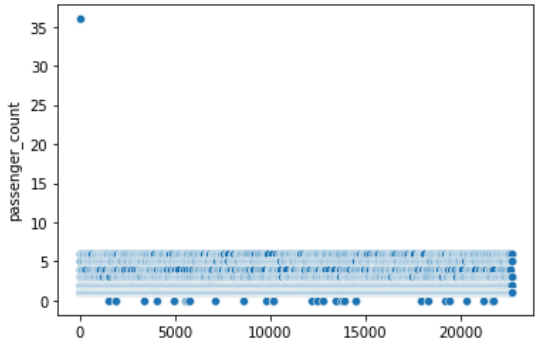
passenger_count 컬럼의 이상치 제거

```
# passenger_count 컬럼의 값을 기준으로 정렬합니다.

data['passenger_count'].sort_values()
```

```
12804     0
19458     0
5565      0
5670      0
13718     0
         ..
416       6
4322      6
14500     6
0         6
64       36
Name: passenger_count, Length: 22696, dtype: int64
```

```
# passenger_count 값의 scatter plot을 그립니다.

sns.scatterplot(x = data.index, y = data['passenger_count'])
```

```
<AxesSubplot:ylabel='passenger_count'>
```




```
# passenger_count 컬럼의 이상치를 제거합니다.
# (passenger_count가 6을 초과하는 경우)

data = data[data['passenger_count'] <= 6]
data
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.3
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.8
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.0
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.7
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.3
...	...	...	...	...	...	.
22696	Austin Johnson	02/24/2017 5:37:23 PM	02/24/2017 5:40:39 PM	Cash	3	0.6
22697	Monique Williams	08/06/2017 4:43:59 PM	08/06/2017 5:24:47 PM	Cash	1	16.7
22698	Drew Graves	09/04/2017 2:54:14 PM	09/04/2017 2:58:22 PM	Debit Card	1	0.4
22699	Jonathan Copeland	07/15/2017 12:56:30 PM	07/15/2017 1:08:26 PM	Debit Card	1	2.3
22700	Benjamin Miller	03/02/2017 1:02:49 PM	03/02/2017 1:16:09 PM	Cash	1	2.1

22695 rows × 9 columns

```
# passenger_count 컬럼의 이상치를 확인합니다.
# (passenger_count가 0인 경우)
```

```
len(data[data['passenger_count'] == 0])
```

```
33
```

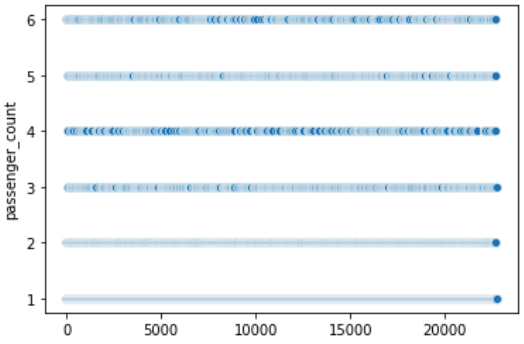
```
# passenger_count 컬럼의 이상치를 제거합니다.

data = data[data['passenger_count'] != 0]
data
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.3
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.8
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.0
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.7
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.3
...	...	...	...	...	...	.
22696	Austin Johnson	02/24/2017 5:37:23 PM	02/24/2017 5:40:39 PM	Cash	3	0.6
22697	Monique Williams	08/06/2017 4:43:59 PM	08/06/2017 5:24:47 PM	Cash	1	16.7
22698	Drew Graves	09/04/2017 2:54:14 PM	09/04/2017 2:58:22 PM	Debit Card	1	0.4
22699	Jonathan Copeland	07/15/2017 12:56:30 PM	07/15/2017 1:08:26 PM	Debit Card	1	2.3
22700	Benjamin Miller	03/02/2017 1:02:49 PM	03/02/2017 1:16:09 PM	Cash	1	2.1

22662 rows × 9 columns

```
# passenger_count의 scatter plot을 다시 그려봅니다.

sns.scatterplot(x = data.index, y = data['passenger_count'])
```

```
<AxesSubplot:ylabel='passenger_count'>
```



수치형 컬럼의 이상치 제거

```
# Q. trip_distance의 이상치를 확인합니다.

data[data['trip_distance'] == 0]
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
129	Linda Kaufman	06/22/2017 8:05:33 AM	06/22/2017 8:05:40 AM	Debit Card	1	0.
248	Erik Perez	09/18/2017 8:50:53 PM	09/18/2017 8:51:03 PM	Cash	1	0.
293	Deborah Sanford	10/04/2017 7:46:24 PM	10/04/2017 7:46:50 PM	Cash	1	0.
321	Ryan Hughes	02/22/2017 4:01:44 AM	02/22/2017 4:01:53 AM	Cash	1	0.
426	David Parker	01/14/2017 7:00:26 AM	01/14/2017 7:00:53 AM	Cash	1	0.
...	...	...	...	...	...	
22192	Angela French	10/16/2017 8:34:07 AM	10/16/2017 8:34:10 AM	Credit Card	1	0.
22327	Kelsey Rogers	07/21/2017 11:30:29 PM	07/21/2017 11:31:12 PM	Debit Card	1	0.
22385	Joseph Castillo	01/07/2017 4:48:42 AM	01/07/2017 4:51:03 AM	Cash	1	0.
22568	Christine Edwards	03/07/2017 2:24:47 AM	03/07/2017 2:24:50 AM	Credit Card	1	0.
22672	John Erickson	03/03/2017 11:09:16 PM	03/03/2017 11:09:35 PM	Debit Card	1	0.

147 rows × 9 columns

```
# Q. trip_distance의 이상치를 제거합니다.
```

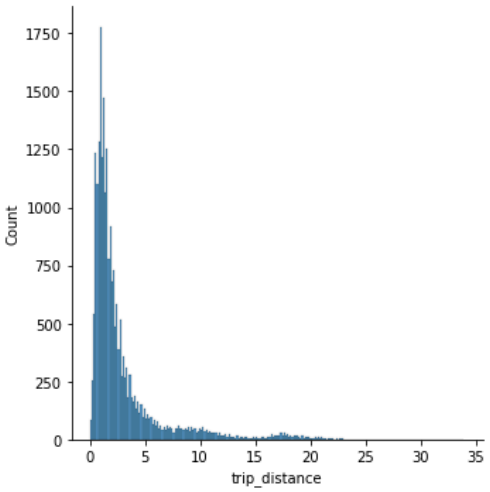
```
data = data[data['trip_distance'] != 0]
data
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.3
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.8
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.0
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.7
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.3
...	...	...	...	...	...	...
22696	Austin Johnson	02/24/2017 5:37:23 PM	02/24/2017 5:40:39 PM	Cash	3	0.6
22697	Monique Williams	08/06/2017 4:43:59 PM	08/06/2017 5:24:47 PM	Cash	1	16.7
22698	Drew Graves	09/04/2017 2:54:14 PM	09/04/2017 2:58:22 PM	Debit Card	1	0.4
22699	Jonathan Copeland	07/15/2017 12:56:30 PM	07/15/2017 1:08:26 PM	Debit Card	1	2.3
22700	Benjamin Miller	03/02/2017 1:02:49 PM	03/02/2017 1:16:09 PM	Cash	1	2.1

22515 rows × 9 columns

```
# Q. trip_distance의 히스토그램을 그립니다.

sns.displot(data['trip_distance'])
```

```
<seaborn.axisgrid.FacetGrid at 0x7ce9c43cbe20>
```



```
data.describe()
```

	passenger_count	trip_distance	fare_amount	tip_amount	tolls_amount
count	22515.000000	22515.000000	22515.000000	22515.000000	22515.000000
mean	1.645969	2.931924	12.958055	1.829513	0.309625
std	1.285783	3.657290	12.701799	2.767054	1.387300
min	1.000000	0.010000	-120.000000	0.000000	0.000000
25%	1.000000	1.000000	6.500000	0.000000	0.000000
50%	1.000000	1.630000	9.500000	1.360000	0.000000
75%	2.000000	3.090000	14.500000	2.450000	0.000000
max	6.000000	33.960000	999.990000	200.000000	19.100000

```
# Q. fare_amount의 이상치 데이터 개수를 확인합니다.
# (fare_amount가 0 이하인 경우)

len(data[data['fare_amount'] < 0])
```

```
13
```

```
# Q. fare_amount의 이상치를 제거합니다.

data = data[data['fare_amount'] > 0]
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.3
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.8
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.0
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.7
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.3
...	...	...	...	...	...	.
22696	Austin Johnson	02/24/2017 5:37:23 PM	02/24/2017 5:40:39 PM	Cash	3	0.6
22697	Monique Williams	08/06/2017 4:43:59 PM	08/06/2017 5:24:47 PM	Cash	1	16.7
22698	Drew Graves	09/04/2017 2:54:14 PM	09/04/2017 2:58:22 PM	Debit Card	1	0.4
22699	Jonathan Copeland	07/15/2017 12:56:30 PM	07/15/2017 1:08:26 PM	Debit Card	1	2.3
22700	Benjamin Miller	03/02/2017 1:02:49 PM	03/02/2017 1:16:09 PM	Cash	1	2.1

22499 rows × 9 columns

```
data.sort_values('fare_amount')
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
4063	Phillip Gonzalez	08/12/2017 8:49:29 PM	08/12/2017 9:18:50 PM	Cash	4	4.5(
14470	Leah Carrillo	09/09/2017 1:29:37 PM	09/09/2017 1:29:57 PM	Credit Card	3	0.0
2987	Christine Harper	11/24/2017 4:32:18 AM	11/24/2017 4:32:23 AM	Credit Card	1	0.0
16351	Nathan Salazar	05/13/2017 5:42:22 PM	05/13/2017 5:42:45 PM	Cash	1	0.0
6702	Yvonne Brooks	08/26/2017 7:33:22 AM	08/26/2017 7:34:18 AM	Debit Card	1	0.1(
...	...	...	...	...	...	..
16381	Erica Hernandez	11/30/2017 10:41:11 AM	11/30/2017 11:31:45 AM	Cash	1	25.5(
9282	Samantha Frederick	06/18/2017 11:33:25 PM	06/19/2017 12:12:38 AM	Cash	2	33.9(
3584	Matthew Chavez	01/01/2017 11:53:01 PM	01/01/2017 11:53:42 PM	Credit Card	1	7.3(
13863	William Yates	05/19/2017 8:20:21 AM	05/19/2017 9:20:30 AM	Credit Card	1	33.9
8478	Alexis Hanson	02/06/2017 5:50:10 AM	02/06/2017 5:51:08 AM	Credit Card	1	2.6(

22499 rows × 9 columns

Q. fare_amount의 이상치를 제거합니다.

```
data = data[data['fare_amount'] < 300]
data
```

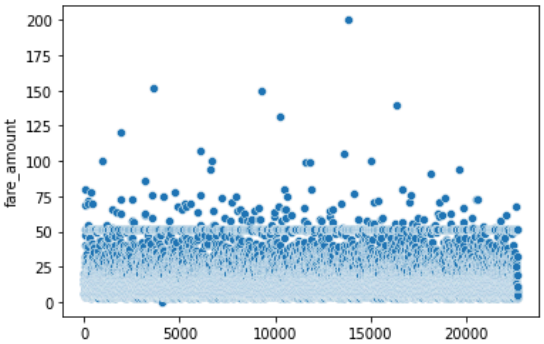
	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.3
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.8
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.0
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.7
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.3
...	...	...	...	...	...	.
22696	Austin Johnson	02/24/2017 5:37:23 PM	02/24/2017 5:40:39 PM	Cash	3	0.6
22697	Monique Williams	08/06/2017 4:43:59 PM	08/06/2017 5:24:47 PM	Cash	1	16.7
22698	Drew Graves	09/04/2017 2:54:14 PM	09/04/2017 2:58:22 PM	Debit Card	1	0.4
22699	Jonathan Copeland	07/15/2017 12:56:30 PM	07/15/2017 1:08:26 PM	Debit Card	1	2.3
22700	Benjamin Miller	03/02/2017 1:02:49 PM	03/02/2017 1:16:09 PM	Cash	1	2.1

22498 rows × 9 columns

Q. fare_amount의 scatter plot을 그립니다.

```
sns.scatterplot(x = data.index , y = data['fare_amount'])
```

```
<AxesSubplot:ylabel='fare_amount'>
```



```
# fare_amount가 150을 초과한다면 150으로 변환합니다.

def fare_func(x):
    if x > 150:
        return 150
    else:
        return x
```

```
data['fare_amount'].apply(fare_func)
```

```
0        13.0
1        16.0
2         6.5
3        20.5
4        16.5
         ...
22696     4.0
22697    52.0
22698     4.5
22699    10.5
22700    11.0
Name: fare_amount, Length: 22498, dtype: float64
```

```
data['fare_amount'] = data['fare_amount'].apply(lambda x: 150 if x > 150 el
se x)
data['fare_amount']
```

```
0        13.0
1        16.0
2         6.5
3        20.5
4        16.5
         ...
22696     4.0
22697    52.0
22698     4.5
22699    10.5
22700    11.0
Name: fare_amount, Length: 22498, dtype: float64
```

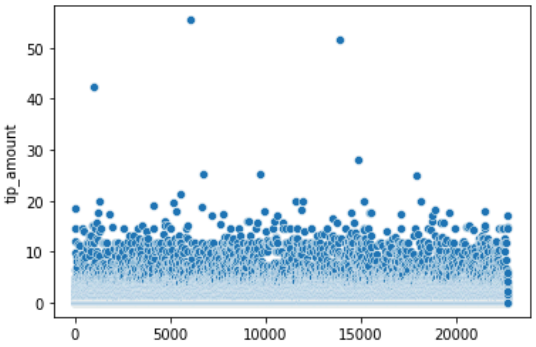
```
data.sort_values('fare_amount')
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
4063	Phillip Gonzalez	08/12/2017 8:49:29 PM	08/12/2017 9:18:50 PM	Cash	4	4.5(
16829	Jeffrey Jackson	05/02/2017 12:18:59 AM	05/02/2017 12:19:02 AM	Credit Card	1	0.0:
19371	Amanda Taylor	03/24/2017 8:59:58 PM	03/24/2017 9:00:06 PM	Cash	1	0.0:
15501	Julie Ferguson	12/29/2017 9:06:34 PM	12/29/2017 9:07:19 PM	Cash	1	4.2(
1077	Kyle Johnson	04/12/2017 8:51:58 PM	04/12/2017 8:52:07 PM	Cash	1	2.3(
...	...	...	...	...	...	..
10293	Emily Stevens	09/11/2017 11:41:04 AM	09/11/2017 12:18:58 PM	Cash	1	31.9
16381	Erica Hernandez	11/30/2017 10:41:11 AM	11/30/2017 11:31:45 AM	Cash	1	25.5(
13863	William Yates	05/19/2017 8:20:21 AM	05/19/2017 9:20:30 AM	Credit Card	1	33.9:
3584	Matthew Chavez	01/01/2017 11:53:01 PM	01/01/2017 11:53:42 PM	Credit Card	1	7.3(
9282	Samantha Frederick	06/18/2017 11:33:25 PM	06/19/2017 12:12:38 AM	Cash	2	33.9(

22498 rows × 9 columns

```
# Q. tip_amount의 scatter plot을 그립니다.

sns.scatterplot(x = data.index , y = data['tip_amount'])
```

```
<AxesSubplot:ylabel='tip_amount'>
```



Q. tip_amount의 이상치를 확인합니다.

```
data[data['tip_amount'] > 40]
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
986	Elaine Horton	08/23/2017 6:23:26 PM	08/23/2017 7:18:29 PM	Cash	1	16.7
6066	Tina Knight	06/13/2017 12:30:22 PM	06/13/2017 1:37:51 PM	Debit Card	1	32.7
13863	William Yates	05/19/2017 8:20:21 AM	05/19/2017 9:20:30 AM	Credit Card	1	33.9

Q. tip_amount의 이상치를 제거합니다.

```
data = data[data['tip_amount'] < 40]
data
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.3
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.8
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.0
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.7
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.3
...	...	...	...	...	...	.
22696	Austin Johnson	02/24/2017 5:37:23 PM	02/24/2017 5:40:39 PM	Cash	3	0.6
22697	Monique Williams	08/06/2017 4:43:59 PM	08/06/2017 5:24:47 PM	Cash	1	16.7
22698	Drew Graves	09/04/2017 2:54:14 PM	09/04/2017 2:58:22 PM	Debit Card	1	0.4
22699	Jonathan Copeland	07/15/2017 12:56:30 PM	07/15/2017 1:08:26 PM	Debit Card	1	2.3
22700	Benjamin Miller	03/02/2017 1:02:49 PM	03/02/2017 1:16:09 PM	Cash	1	2.1

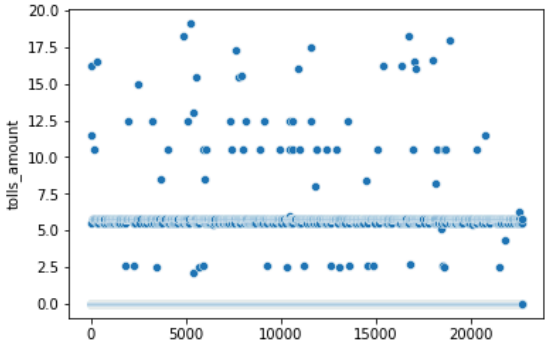
22495 rows × 9 columns

```
len(data)
```

```
22495
```

```
# Q. tolls_amount의 scatter plot을 그립니다.

sns.scatterplot(x = data.index , y = data['tolls_amount'])
```

```
<AxesSubplot:ylabel='tolls_amount'>
```



범주형 데이터 전처리

결제 방법: Debit Card와 Credit Card를 Card로 통합합니다.

```
data.head(10)
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distance	f
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.34	
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.80	
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.00	
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.70	
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.37	
5	Justin Smith	03/25/2017 8:34:11 PM	03/25/2017 8:42:11 PM	Debit Card	6	2.30	
6	Tonya Moreno	05/03/2017 7:04:09 PM	05/03/2017 8:03:47 PM	Cash	1	12.83	
7	Hannah Foley	08/15/2017 5:41:06 PM	08/15/2017 6:03:05 PM	Debit Card	1	2.98	
8	Katie Whitney	02/04/2017 4:17:07 PM	02/04/2017 4:29:14 PM	Cash	1	1.20	
9	Amanda Jones	11/10/2017 3:20:29 PM	11/10/2017 3:40:55 PM	Cash	1	1.60	
10	Cory Jensen	03/04/2017 11:58:00 AM	03/04/2017 12:13:12 PM	Cash	1	1.77	

```
# payment_method 컬럼에 어떤 값들이 있는지 살펴봅시다.

data['payment_method'].unique()
```

```
array(['Debit Card', 'Cash', 'Credit Card'], dtype=object)
```

```
data['payment_method'].nunique()
```

```
3
```

```
data['payment_method'].value_counts()
```

```
Cash           11094
Debit Card      5729
Credit Card     5672
Name: payment_method, dtype: int64
```

```
# Q. 'Debit Card'와 'Credit Card' 항목을 'Card'로 변환합니다.
# (힌트: replace() 메서드를 사용합니다.)

data['payment_method'].replace({'Debit Card': 'Card', 'Credit Card': 'Card'})
```

```
0        Card
1        Card
2        Card
3        Cash
4        Card
         ...
22696    Cash
22697    Cash
22698    Card
22699    Card
22700    Cash
Name: payment_method, Length: 22495, dtype: object
```

```
data['payment_method'].value_counts()
```

```
Cash           11094
Debit Card      5729
Credit Card     5672
Name: payment_method, dtype: int64
```

승객명: 성과 이름을 분리하여 성 부분만 저장해봅니다.

```
example = 'Susan Robinson'
```

```
example.split()
```

```
['Susan', 'Robinson']
```

```
# Q. passenger_name을 성과 이름으로 분리하여 성 부분만 passenger_first_name 컬럼으로 저장합니다.
```

```
data['passenger_first_name'] = data['passenger_name'].str.split(expand = True)[0]
```

```
0          Pamela
1        Michelle
2            Tina
3         Anthony
4         Brianna
           ...
22696      Austin
22697     Monique
22698        Drew
22699     Jonathan
22700     Benjamin
Name: passenger_first_name, Length: 22495, dtype: object
```

택시 탑승, 하차 시간을 활용해봅시다.

```
data.head()
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distance	fa
0	Pamela Duffy	03/25/2017 8:55:43 AM	03/25/2017 9:09:47 AM	Debit Card	6	3.34	
1	Michelle Foster	04/11/2017 2:53:28 PM	04/11/2017 3:19:58 PM	Debit Card	1	1.80	
2	Tina Combs	12/15/2017 7:26:56 AM	12/15/2017 7:34:08 AM	Debit Card	1	1.00	
3	Anthony Ray	05/07/2017 1:17:59 PM	05/07/2017 1:48:14 PM	Cash	1	3.70	
4	Brianna Johnson	04/15/2017 11:32:20 PM	04/15/2017 11:49:03 PM	Debit Card	1	4.37	

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22495 entries, 0 to 22700
Data columns (total 10 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   passenger_name        22495 non-null  object
 1   tpep_pickup_datetime  22495 non-null  object
 2   tpep_dropoff_datetime 22495 non-null  object
 3   payment_method        22495 non-null  object
 4   passenger_count       22495 non-null  int64
 5   trip_distance         22495 non-null  float64
 6   fare_amount           22495 non-null  float64
 7   tip_amount            22495 non-null  float64
 8   tolls_amount          22495 non-null  float64
 9   passenger_first_name  22495 non-null  object
dtypes: float64(4), int64(1), object(5)
memory usage: 1.9+ MB
```

Q. tpep_pickup_datetime 컬럼의 object 자료형을 datetime으로 변환합니다.

```
data['tpep_pickup_datetime'] = pd.to_datetime(data['tpep_pickup_datetime'])
data['tpep_pickup_datetime']
```

```
0       2017-03-25 08:55:43
1       2017-04-11 14:53:28
2       2017-12-15 07:26:56
3       2017-05-07 13:17:59
4       2017-04-15 23:32:20
               ...
22696   2017-02-24 17:37:23
22697   2017-08-06 16:43:59
22698   2017-09-04 14:54:14
22699   2017-07-15 12:56:30
22700   2017-03-02 13:02:49
Name: tpep_pickup_datetime, Length: 22495, dtype: datetime64[ns]
```

Q. tpep_dropoff_datetime 컬럼의 object 자료형을 datetime으로 변환합니다.

```
data['tpep_dropoff_datetime'] = pd.to_datetime(data['tpep_dropoff_datetime'])
data['tpep_dropoff_datetime']
```

```
0       2017-03-25 09:09:47
1       2017-04-11 15:19:58
2       2017-12-15 07:34:08
3       2017-05-07 13:48:14
4       2017-04-15 23:49:03
               ...
22696   2017-02-24 17:40:39
22697   2017-08-06 17:24:47
22698   2017-09-04 14:58:22
22699   2017-07-15 13:08:26
22700   2017-03-02 13:16:09
Name: tpep_dropoff_datetime, Length: 22495, dtype: datetime64[ns]
```

data.info()

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22495 entries, 0 to 22700
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   passenger_name         22495 non-null  object
 1   tpep_pickup_datetime   22495 non-null  datetime64[ns]
 2   tpep_dropoff_datetime  22495 non-null  datetime64[ns]
 3   payment_method         22495 non-null  object
 4   passenger_count        22495 non-null  int64
 5   trip_distance          22495 non-null  float64
 6   fare_amount            22495 non-null  float64
 7   tip_amount             22495 non-null  float64
 8   tolls_amount           22495 non-null  float64
 9   passenger_first_name   22495 non-null  object
dtypes: datetime64[ns](2), float64(4), int64(1), object(3)
memory usage: 1.9+ MB
```

Q. 하차 시각과 승차 시각의 차이를 travel_time 컬럼으로 저장합니다.

data['travel_time'] = data['tpep_dropoff_datetime'] - data['tpep_pickup_datetime']
data['travel_time']

```
0       0 days 00:14:04
1       0 days 00:26:30
2       0 days 00:07:12
3       0 days 00:30:15
4       0 days 00:16:43
              ...
22696   0 days 00:03:16
22697   0 days 00:40:48
22698   0 days 00:04:08
22699   0 days 00:11:56
22700   0 days 00:13:20
Name: travel_time, Length: 22495, dtype: timedelta64[ns]
```

data.head()

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distance	fa
0	Pamela Duffy	2017-03-25 08:55:43	2017-03-25 09:09:47	Debit Card	6	3.34	
1	Michelle Foster	2017-04-11 14:53:28	2017-04-11 15:19:58	Debit Card	1	1.80	
2	Tina Combs	2017-12-15 07:26:56	2017-12-15 07:34:08	Debit Card	1	1.00	
3	Anthony Ray	2017-05-07 13:17:59	2017-05-07 13:48:14	Cash	1	3.70	
4	Brianna Johnson	2017-04-15 23:32:20	2017-04-15 23:49:03	Debit Card	1	4.37	

data.info()

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22495 entries, 0 to 22700
Data columns (total 11 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   passenger_name        22495 non-null  object
 1   tpep_pickup_datetime  22495 non-null  datetime64[ns]
 2   tpep_dropoff_datetime 22495 non-null  datetime64[ns]
 3   payment_method        22495 non-null  object
 4   passenger_count       22495 non-null  int64
 5   trip_distance         22495 non-null  float64
 6   fare_amount           22495 non-null  float64
 7   tip_amount            22495 non-null  float64
 8   tolls_amount          22495 non-null  float64
 9   passenger_first_name  22495 non-null  object
 10  travel_time           22495 non-null  timedelta64[ns]
dtypes: datetime64[ns](2), float64(4), int64(1), object(3), timedelta64[ns](1)
memory usage: 2.1+ MB
```

```
# Q. travel_time 컬럼의 데이터를 초 단위로 변환합니다.

data['travel_time'] = data['travel_time'].dt.seconds
data['travel_time']
```

```
0         844
1        1590
2         432
3        1815
4        1003
         ...
22696     196
22697    2448
22698     248
22699     716
22700     800
Name: travel_time, Length: 22495, dtype: int64
```

보너스 (feature engineering 맛보기)

```
data.head()
```

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distance	fa
0	Pamela Duffy	2017-03-25 08:55:43	2017-03-25 09:09:47	Debit Card	6	3.34	
1	Michelle Foster	2017-04-11 14:53:28	2017-04-11 15:19:58	Debit Card	1	1.80	
2	Tina Combs	2017-12-15 07:26:56	2017-12-15 07:34:08	Debit Card	1	1.00	
3	Anthony Ray	2017-05-07 13:17:59	2017-05-07 13:48:14	Cash	1	3.70	
4	Brianna Johnson	2017-04-15 23:32:20	2017-04-15 23:49:03	Debit Card	1	4.37	

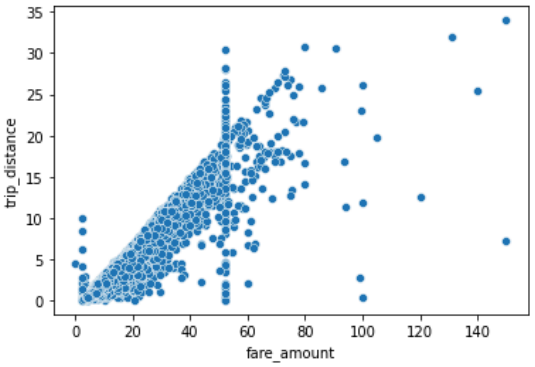
```
# Q. 승객이 지불한 총 요금을 total_amount 컬럼으로 저장합니다.

data['total_amount'] = data['fare_amount'] + data['tip_amount'] + data['tolls
_amount']
data['total_amount']
```

```
0        15.76
1        20.00
2         7.95
3        26.89
4        16.50
         ...
22696     4.00
22697    72.40
22698     4.50
22699    12.20
22700    13.35
Name: total_amount, Length: 22495, dtype: float64
```

Q. fare_amount와 trip_distance 사이의 관계를 scatter plot으로 표현합니다.

sns.scatterplot(x = data['fare_amount'], y = data['trip_distance'])

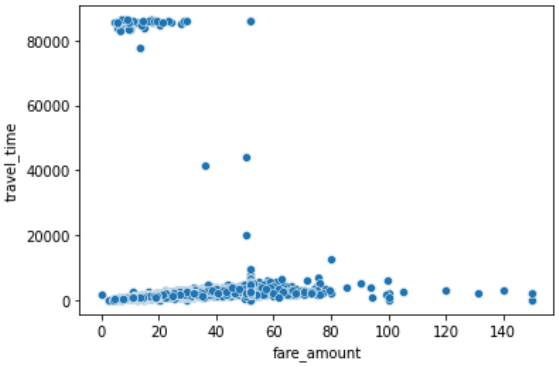
```
<AxesSubplot:xlabel='fare_amount', ylabel='trip_distance'>
```



Q. fare_amount와 travel_time 사이의 관계를 scatter plot으로 표현합니다.

sns.scatterplot(x = data['fare_amount'], y = data['travel_time'])

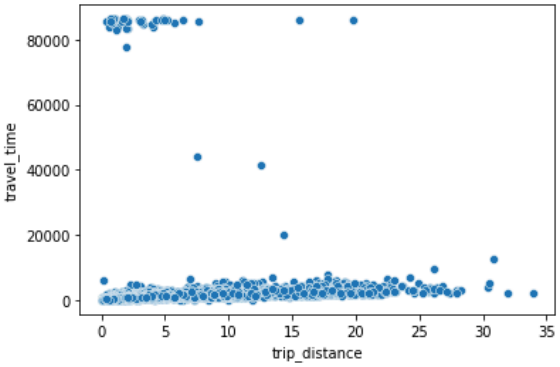
```
<AxesSubplot:xlabel='fare_amount', ylabel='travel_time'>
```



Q. trip_distance와 travel_time 사이의 관계를 scatter plot으로 표현합니다.

sns.scatterplot(x = data['trip_distance'], y = data['travel_time'])

```
<AxesSubplot:xlabel='trip_distance', ylabel='travel_time'>
```



Q. scatter plot으로 관찰된 travel_time의 이상치를 제거합니다.

data[data['travel_time'] > 60000]

	passenger_name	tpep_pickup_datetime	tpep_dropoff_datetime	payment_method	passenger_count	trip_distanc
699	Scott Garcia	2017-06-10 21:55:01	2017-06-11 21:45:51	Debit Card	1	1.3
926	Michael Perez	2017-02-09 23:24:58	2017-02-10 23:24:31	Cash	5	4.8
1012	James Anderson	2017-12-08 07:17:20	2017-12-09 07:07:22	Cash	1	0.3
1201	Carla Allen	2017-11-12 19:52:44	2017-11-13 19:37:35	Credit Card	1	4.1
1357	Jamie Collins	2017-04-17 21:26:49	2017-04-18 20:46:13	Cash	6	4.0
1760	Ronald Kidd	2017-12-28 23:58:24	2017-12-29 23:38:45	Cash	1	1.2
4602	Brandon Miller	2017-12-20 08:24:34	2017-12-21 07:39:27	Cash	4	1.2
5372	Catherine Ray	2017-12-13 19:40:05	2017-12-14 19:31:09	Cash	3	0.9
5480	Patricia Galvan	2017-09-19 13:16:13	2017-09-20 12:36:12	Credit Card	1	0.6
6495	Travis Tucker	2017-06-27 16:52:07	2017-06-28 16:49:57	Cash	1	15.6
6753	Justin Rosales	2017-06-14 11:51:18	2017-06-15 11:49:20	Credit Card	5	2.9
7014	Alex Cummings	2017-12-20 08:23:16	2017-12-21 08:19:56	Cash	1	19.7
7171	Michael Allen	2017-04-09 07:55:14	2017-04-10 07:02:02	Debit Card	1	1.1
7941	Benjamin Ortiz	2017-06-30 20:36:00	2017-07-01 20:34:28	Cash	1	1.0
8197	David Crane	2017-02-12 02:21:07	2017-02-13 00:00:00	Credit Card	1	1.9
8714	Rhonda Castillo	2017-06-18 09:21:07	2017-06-19 08:59:45	Debit Card	6	0.8
8871	Kathleen Welch	2017-07-12 21:55:00	2017-07-13 21:50:48	Debit Card	4	0.9
9210	Renee Bowman	2017-09-22 09:20:53	2017-09-23 09:04:02	Debit Card	1	1.8
9358	Donna Summers	2017-11-05 01:23:08	2017-11-05 01:06:09	Cash	1	5.7
10212	Dennis Goodwin	2017-06-30 22:39:13	2017-07-01 22:33:12	Cash	1	1.5
10931	Jesse Ward DVM	2017-04-02 17:28:22	2017-04-03 17:23:29	Cash	6	6.3
11674	Jesus Smith	2017-03-18 14:58:31	2017-03-19 14:31:35	Debit Card	3	3.3