전제조건

> 대리점에서는 상품 A-Z 까지 26개의 상품을 취급. 매출이력과 고객 정보 데이터는 담당사원이 시스템에 직접 입력함.
> 집계 기간에 상품 단가의 변동은 없었고 매출 이력은 시스템에서 CSV 파일로 출력.
> 고객 정보는 대리점 관리자가 주별로 집계해서 엑셀로 관리함.

데이터 정보

> 1.uriage.csv : 매출이력 (기간: 2019-01 ~ 2019-07)
> 2.kokyaku_daicho.xlsx : 대리점에서 관리하는 고객 정보

데이터 읽어오기

In [1]:
```
import pandas as pd
uriage_data = pd.read_csv("uriage.csv")
uriage_data.head()
```

Out[1]:

	purchase_date	item_name	item_price	customer_name
0	2019-06-13 18:02	상품A	100.0	김가온
1	2019-07-13 13:05	상 품 S	NaN	김우찬
2	2019-05-11 19:42	상 품 a	NaN	김유찬
3	2019-02-12 23:40	상품Z	2600.0	김재현
4	2019-04-22 3:09	상품a	NaN	김강현

In [2]:
```
kokyaku_data = pd.read_excel("kokyaku_daicho.xlsx")
kokyaku_data.head()
```

Out[2]:

	고객이름	지역	등록일
0	김 현성	H시	2018-01-04 00:00:00
1	김 도윤	E시	42782
2	김 지한	A시	2018-01-07 00:00:00
3	김 하윤	F시	42872
4	김 시온	E시	43127

In [3]:
```
uriage_data['item_name'].head()
```

Out[3]:
```
0      상품A
1    상 품 S
2    상 품 a
3      상품Z
4      상품a
Name: item_name, dtype: object
```

In [5]:
```
uriage_data["purchase_date"]=pd.to_datetime(uriage_data["purchase_date"])
uriage_data["purchase_month"]=uriage_data["purchase_date"].dt.strftime("%Y%m")
res=uriage_data.pivot_table(index="purchase_month",columns="item_name",aggfunc="size",fill_value=0)
res
```

Out[5]:

item_name	상 품n	상 품 E	상 품 M	상 품 P	상 품 S	상품 W	상 품 X	상 품 W	상 품O	상 품Q	...	상 품 k	상 품 l	상 품 o	상 품 p	상 품r	상 품 s	상 품t	상 품 v	상 품x	상 품y
purchase_month																					
201901	1	0	0	0	0	0	0	0	0	0	...	1	1	1	0	0	0	0	0	0	0
201902	0	0	0	0	0	0	1	0	0	0	...	0	0	0	0	0	1	1	1	0	0
201903	0	1	1	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
201904	0	0	0	0	0	0	0	1	0	1	...	0	0	0	0	0	1	0	0	0	0
201905	0	0	0	0	1	0	0	0	0	0	...	0	1	0	0	0	0	0	0	0	1
201906	0	0	0	0	0	1	0	0	0	0	...	0	0	0	1	0	0	0	0	1	0
201907	0	0	0	0	0	0	0	0	1	0	...	0	0	1	0	2	0	0	0	0	0

7 rows × 99 columns

In [6]:
```
uriage_data['item_price'].head()
```

Out[6]:
```
0     100.0
1       NaN
2       NaN
3    2600.0
4       NaN
Name: item_price, dtype: float64
```

데이터 전처리

In [7]:
```
print(len(pd.unique(uriage_data.item_name)))
```

```
99
```

상품의 개수는 위에서 언급했듯이 26개여야 하는데 상품명의 정합성이 맞지 않아 99개로 증가함.

In [8]:
```
uriage_data["item_name"]=uriage_data["item_name"].str.upper()
uriage_data["item_name"]=uriage_data["item_name"].str.replace(" ","")
uriage_data["item_name"]=uriage_data["item_name"].str.replace("  ","")
uriage_data.sort_values(by=["item_name"],ascending=True)
```

Out[8]:

	purchase_date	item_name	item_price	customer_name	purchase_month
0	2019-06-13 18:02:00	상품A	100.0	김가온	201906
1748	2019-05-19 20:22:00	상품A	100.0	김시훈	201905
223	2019-06-25 08:13:00	상품A	100.0	김유진	201906
1742	2019-06-13 16:03:00	상품A	100.0	김건희	201906
1738	2019-02-10 00:28:00	상품A	100.0	김하랑	201902
...	...	...	...	...	...
2880	2019-04-22 00:36:00	상품Y	NaN	김동욱	201904
2881	2019-04-30 14:21:00	상품Y	NaN	김하준	201904
1525	2019-01-24 10:27:00	상품Y	2500.0	김범준	201901
1361	2019-05-28 13:45:00	상품Y	2500.0	김수현	201905
3	2019-02-12 23:40:00	상품Z	2600.0	김재현	201902

2999 rows × 5 columns

In [9]:
```
print(pd.unique(uriage_data["item_name"]))
print(len(pd.unique(uriage_data["item_name"])))
```

```
['상품A' '상품S' '상품Z' '상품V' '상품O' '상품U' '상품L' '상품C' '상품I' '상품R' '상품X' '상품G'
 '상품P' '상품Q' '상품Y' '상품N' '상품W' '상품E' '상품K' '상품B' '상품F' '상품D' '상품M' '상품H'
 '상품T' '상품J']
26
```

금액의 결측치 수정

In [10]:
```
uriage_data.isnull().any(axis=0)
```

Out[10]:
```
purchase_date     False
item_name         False
item_price         True
customer_name     False
purchase_month    False
dtype: bool
```

In [11]:
```
#결측치가 있는 행
flg_is_null=uriage_data["item_price"].isnull()
#결측치가 있는 행의 상품명
for trg in list(uriage_data.loc[flg_is_null,"item_name"].unique()):
    #동일 상품명을 가지는 결측치가 없는 행의 판매가격 저장
    price=uriage_data.loc[(~flg_is_null)&(uriage_data["item_name"]==trg),"item_price"].max()
    #결측치가 없는 행에 위에서 가져온 판매가격을 저장하여 결측치 처리
    uriage_data["item_price"].loc[(flg_is_null)&(uriage_data["item_name"]==trg)]=price
uriage_data.head()
```

```
/var/folders/r9/0d4blp8n2wnf_hpl310kz6fr0000gn/T/ipykernel_46436/795925611.py:8: SettingWithCopyWarn
ing:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy
  uriage_data["item_price"].loc[(flg_is_null)&(uriage_data["item_name"]==trg)]=price
```

Out[11]:

	purchase_date	item_name	item_price	customer_name	purchase_month
0	2019-06-13 18:02:00	상품A	100.0	김가온	201906
1	2019-07-13 13:05:00	상품S	1900.0	김우찬	201907
2	2019-05-11 19:42:00	상품A	100.0	김유찬	201905
3	2019-02-12 23:40:00	상품Z	2600.0	김재현	201902
4	2019-04-22 03:09:00	상품A	100.0	김강현	201904

In [12]:
```
uriage_data.isnull().any(axis=0)
```

Out[12]:
```
purchase_date     False
item_name         False
item_price        False
customer_name     False
purchase_month    False
dtype: bool
```

In [13]:
```
for trg in list(uriage_data["item_name"].sort_values().unique()):
    print(trg+"의 최고가:"+str(uriage_data.loc[uriage_data["item_name"]==trg]["item_price"].max())
         +"/최저가:"+str(uriage_data.loc[uriage_data["item_name"]==trg]["item_price"].min(skipna=False
```

```
상품A의 최고가:100.0/최저가:100.0
상품B의 최고가:200.0/최저가:200.0
상품C의 최고가:300.0/최저가:300.0
상품D의 최고가:400.0/최저가:400.0
상품E의 최고가:500.0/최저가:500.0
상품F의 최고가:600.0/최저가:600.0
상품G의 최고가:700.0/최저가:700.0
상품H의 최고가:800.0/최저가:800.0
상품I의 최고가:900.0/최저가:900.0
상품J의 최고가:1000.0/최저가:1000.0
상품K의 최고가:1100.0/최저가:1100.0
상품L의 최고가:1200.0/최저가:1200.0
상품M의 최고가:1300.0/최저가:1300.0
상품N의 최고가:1400.0/최저가:1400.0
상품O의 최고가:1500.0/최저가:1500.0
상품P의 최고가:1600.0/최저가:1600.0
상품Q의 최고가:1700.0/최저가:1700.0
상품R의 최고가:1800.0/최저가:1800.0
상품S의 최고가:1900.0/최저가:1900.0
상품T의 최고가:2000.0/최저가:2000.0
상품U의 최고가:2100.0/최저가:2100.0
상품V의 최고가:2200.0/최저가:2200.0
상품W의 최고가:2300.0/최저가:2300.0
상품X의 최고가:2400.0/최저가:2400.0
상품Y의 최고가:2500.0/최저가:2500.0
상품Z의 최고가:2600.0/최저가:2600.0
```

각 상품에 대해 최대 금액과 최소 금액이 일치하는 것으로 보아 오류 수정이 잘 이루어짐.

고객 이름 오류 수정 : 성과 이름 사이의 공백 제거

In [14]:
```
kokyaku_data["고객이름"].head()
```

Out[14]:
```
0    김 현성
1    김 도윤
2    김 지한
3    김 하윤
4    김 시온
Name: 고객이름, dtype: object
```

In [15]:
```
uriage_data["customer_name"].head()
```

Out[15]:
```
0    김가온
1    김우찬
2    김유찬
3    김재현
4    김강현
Name: customer_name, dtype: object
```

In [16]:
```
kokyaku_data["고객이름"]=kokyaku_data["고객이름"].str.replace("  ","")
kokyaku_data["고객이름"]=kokyaku_data["고객이름"].str.replace(" ","")
kokyaku_data["고객이름"].head()
```

Out[16]:
```
0    김현성
1    김도윤
2    김지한
3    김하윤
4    김시온
Name: 고객이름, dtype: object
```

날짜 오류 수정 : 여러가지 형식의 날짜 수정

In [17]:
```
kokyaku_data["등록일"].dtypes
```

Out[17]:
```
dtype('O')
```

In [18]:
```
flg_is_serial = kokyaku_data["등록일"].astype("str").str.isdigit()
flg_is_serial.sum()
```

Out[18]:
```
22
```

In [19]:
```
fromSerial=pd.to_timedelta(kokyaku_data.loc[flg_is_serial,"등록일"].astype("float"),unit="D")+pd.to_da
fromSerial
```

Out[19]:
```
1     2017-02-18
3     2017-05-19
4     2018-01-29
21    2017-07-06
27    2017-06-17
47    2017-01-08
49    2017-07-15
53    2017-04-10
76    2018-03-31
80    2018-01-12
99    2017-06-01
114   2018-06-05
118   2018-01-31
122   2018-04-18
139   2017-05-27
143   2017-03-26
155   2017-01-21
172   2018-03-24
179   2017-01-10
183   2017-07-26
186   2018-07-15
192   2018-06-10
Name: 등록일, dtype: datetime64[ns]
```

In [20]:
```
fromString = pd.to_datetime(kokyaku_data.loc[~flg_is_serial,"등록일"])
fromString
```

Out[20]:
```
0     2018-01-04
2     2018-01-07
5     2017-06-20
6     2018-06-11
7     2017-05-19
         ...
195   2017-06-20
196   2018-06-20
197   2017-04-29
198   2019-04-19
199   2019-04-23
Name: 등록일, Length: 178, dtype: datetime64[ns]
```

In [21]:
```
kokyaku_data["등록일"]=pd.concat([fromSerial,fromString])
kokyaku_data
```

Out[21]:

	고객이름	지역	등록일
0	김현성	H시	2018-01-04
1	김도윤	E시	2017-02-18
2	김지한	A시	2018-01-07
3	김하윤	F시	2017-05-19
4	김시온	E시	2018-01-29
...	...	...	...
195	김재희	G시	2017-06-20
196	김도영	E시	2018-06-20
197	김이안	F시	2017-04-29
198	김시현	H시	2019-04-19
199	김서우	D시	2019-04-23

200 rows × 3 columns

In [22]:
```
kokyaku_data["등록연월"]=kokyaku_data["등록일"].dt.strftime("%Y%m")
rslt = kokyaku_data.groupby("등록연월").count()["고객이름"]
print(rslt)
print(len(kokyaku_data))
```

```
등록연월
201701    15
201702    11
201703    14
201704    15
201705    13
201706    14
201707    17
201801    13
201802    15
201803    17
201804     5
201805    19
201806    13
201807    17
201904     2
Name: 고객이름, dtype: int64
200
```

데이터 결합

고객이름을 키로 매출 이력과 고객 정보 데이터 결합

In [23]:
```
join_data = pd.merge(uriage_data,kokyaku_data,left_on="customer_name",right_on="고객이름",how="left")
join_data = join_data.drop("customer_name",axis=1)
join_data
```

Out[23]:

	purchase_date	item_name	item_price	purchase_month	고객이름	지역	등록일	등록연월
0	2019-06-13 18:02:00	상품A	100.0	201906	김가온	C시	2017-01-26	201701
1	2019-07-13 13:05:00	상품S	1900.0	201907	김우찬	C시	2018-04-07	201804
2	2019-05-11 19:42:00	상품A	100.0	201905	김유찬	A시	2018-06-19	201806
3	2019-02-12 23:40:00	상품Z	2600.0	201902	김재현	D시	2018-07-22	201807
4	2019-04-22 03:09:00	상품A	100.0	201904	김강현	D시	2017-06-07	201706
...	...	...	...	...	...	...	...	...
2994	2019-02-15 02:56:00	상품Y	2500.0	201902	김정민	B시	2017-07-01	201707
2995	2019-06-22 04:03:00	상품M	1300.0	201906	김재원	E시	2018-03-31	201803
2996	2019-03-29 11:14:00	상품Q	1700.0	201903	김지율	B시	2017-03-15	201703
2997	2019-07-14 12:56:00	상품H	800.0	201907	김승주	E시	2018-07-15	201807
2998	2019-07-21 00:31:00	상품D	400.0	201907	정준기	B시	2017-02-05	201702

2999 rows × 8 columns

In [24]:
```
dump_data = join_data[["purchase_date","purchase_month","item_name","item_price","고객이름","지역","등
dump_data
```

Out[24]:

	purchase_date	purchase_month	item_name	item_price	고객이름	지역	등록일
0	2019-06-13 18:02:00	201906	상품A	100.0	김가온	C시	2017-01-26
1	2019-07-13 13:05:00	201907	상품S	1900.0	김우찬	C시	2018-04-07
2	2019-05-11 19:42:00	201905	상품A	100.0	김유찬	A시	2018-06-19
3	2019-02-12 23:40:00	201902	상품Z	2600.0	김재현	D시	2018-07-22
4	2019-04-22 03:09:00	201904	상품A	100.0	김강현	D시	2017-06-07
...	...	...	...	...	...	...	...
2994	2019-02-15 02:56:00	201902	상품Y	2500.0	김정민	B시	2017-07-01
2995	2019-06-22 04:03:00	201906	상품M	1300.0	김재원	E시	2018-03-31
2996	2019-03-29 11:14:00	201903	상품Q	1700.0	김지율	B시	2017-03-15
2997	2019-07-14 12:56:00	201907	상품H	800.0	김승주	E시	2018-07-15
2998	2019-07-21 00:31:00	201907	상품D	400.0	정준기	B시	2017-02-05

2999 rows × 7 columns

In [26]:
```
dump_data.to_csv("dump_data.csv",index=False)
```

데이터 집계

In [27]:
```
import_data = pd.read_csv("dump_data.csv")
import_data
```

Out[27]:

	purchase_date	purchase_month	item_name	item_price	고객이름	지역	등록일
0	2019-06-13 18:02:00	201906	상품A	100.0	김가온	C시	2017-01-26
1	2019-07-13 13:05:00	201907	상품S	1900.0	김우찬	C시	2018-04-07
2	2019-05-11 19:42:00	201905	상품A	100.0	김유찬	A시	2018-06-19
3	2019-02-12 23:40:00	201902	상품Z	2600.0	김재현	D시	2018-07-22
4	2019-04-22 03:09:00	201904	상품A	100.0	김강현	D시	2017-06-07
...	...	...	...	...	...	...	...
2994	2019-02-15 02:56:00	201902	상품Y	2500.0	김정민	B시	2017-07-01
2995	2019-06-22 04:03:00	201906	상품M	1300.0	김재원	E시	2018-03-31
2996	2019-03-29 11:14:00	201903	상품Q	1700.0	김지율	B시	2017-03-15
2997	2019-07-14 12:56:00	201907	상품H	800.0	김승주	E시	2018-07-15
2998	2019-07-21 00:31:00	201907	상품D	400.0	정준기	B시	2017-02-05

2999 rows × 7 columns

In [28]:
```
byItem = import_data.pivot_table(index="purchase_month",columns="item_name",aggfunc="size",fill_value
byItem
```

Out[28]:

item_name	상품A	상품B	상품C	상품D	상품E	상품F	상품G	상품H	상품I	상품J	...	상품Q	상품R	상품S	상품T	상품U	상품V	상품W	상품X	상품Y	상품Z
purchase_month																					
201901	18	13	19	17	18	15	11	16	18	17	...	17	21	20	17	7	22	13	14	10	0
201902	19	14	26	21	16	14	14	17	12	14	...	22	22	22	23	19	22	24	16	11	1
201903	17	21	20	17	9	27	14	18	12	16	...	23	16	20	12	23	18	16	21	16	0
201904	17	19	24	20	18	17	14	11	18	13	...	20	20	16	16	11	15	14	16	20	0
201905	24	14	16	14	19	18	23	15	16	11	...	13	22	18	16	16	9	21	16	20	0
201906	24	12	11	19	13	18	15	13	19	22	...	15	16	21	12	18	20	17	15	13	0
201907	20	20	17	17	12	17	19	19	19	23	...	15	19	23	21	13	28	16	18	12	0

7 rows × 26 columns

In [29]:
```
byPrice = import_data.pivot_table(index="purchase_month",columns="item_name",values="item_price",aggf
byPrice
```

Out[29]:

item_name	상품A	상품B	상품C	상품D	상품E	상품F	상품G	상품H	상품I	상품J	...	상품Q	상품R	상품S	상품T	상품U
purchase_month																
201901	1800	2600	5700	6800	9000	9000	7700	12800	16200	17000	...	28900	37800	38000	34000	14700
201902	1900	2800	7800	8400	8000	8400	9800	13600	10800	14000	...	37400	39600	41800	46000	39900
201903	1700	4200	6000	6800	4500	16200	9800	14400	10800	16000	...	39100	28800	38000	24000	48300
201904	1700	3800	7200	8000	9000	10200	9800	8800	16200	13000	...	34000	36000	30400	32000	23100
201905	2400	2800	4800	5600	9500	10800	16100	12000	14400	11000	...	22100	39600	34200	32000	33600
201906	2400	2400	3300	7600	6500	10800	10500	10400	17100	22000	...	25500	28800	39900	24000	37800
201907	2000	4000	5100	6800	6000	10200	13300	15200	17100	23000	...	25500	34200	43700	42000	27300

7 rows × 26 columns

In [30]:
```
byCustomer = import_data.pivot_table(index="purchase_month",columns="고객이름",aggfunc="size",fill_val
byCustomer
```

Out[30]:

고객이름	김가온	김강민	김강현	김건우	김건희	김경민	김규민	김규현	김다온	김대현	...	김현수	김현승	김현우	김현준	김현진	김호준	정도형	정영훈	정우석	정준기
purchase_month																					
201901	1	2	1	2	5	3	1	1	1	5	...	3	2	4	2	1	1	4	4	4	3
201902	3	1	4	3	3	5	2	3	1	5	...	5	3	7	0	0	1	2	0	2	2
201903	4	0	3	1	1	2	2	1	1	5	...	3	2	1	2	1	2	1	2	3	4
201904	1	0	3	0	2	1	2	6	2	4	...	0	4	3	1	1	2	2	4	3	1
201905	0	0	2	2	1	4	6	3	3	1	...	2	2	1	4	3	0	0	0	1	2
201906	5	0	3	0	2	1	6	1	1	1	...	2	4	1	0	4	2	2	0	0	2
201907	3	1	3	2	2	1	4	1	4	3	...	2	1	0	4	0	2	6	1	2	3

7 rows × 199 columns

In [31]:
```
byRegion = import_data.pivot_table(index="purchase_month",columns="지역",aggfunc="size",fill_value=0)
byRegion
```

Out[31]:

지역	A시	B시	C시	D시	E시	F시	G시	H시
purchase_month								
201901	59	55	72	34	49	57	49	42
201902	71	46	65	48	61	52	43	63
201903	64	52	57	43	52	59	51	59
201904	64	48	54	45	48	58	40	52
201905	57	52	68	48	59	65	35	43
201906	53	47	61	30	51	51	58	58
201907	76	53	61	42	54	64	47	54

In [32]:
```
away_data = pd.merge(uriage_data,kokyaku_data,left_on="customer_name",right_on="고객이름",how="right")
away_data[away_data["purchase_date"].isnull()][["고객이름","등록일"]]
```

Out[32]:

	고객이름	등록일
2999	김서우	2019-04-23

집계 기간 내 이탈 고객

In [ ]: