Final Assignment

Name : 황창현

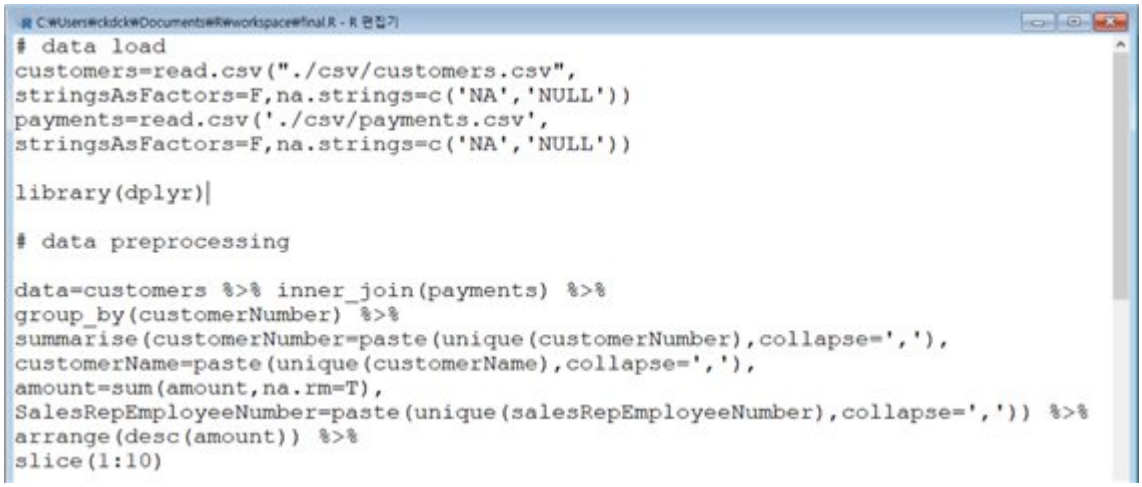
student id : 17010668

[Table1 Output] Top10 customers with high amount

Data: data

	customerNumber	customerName	amount	SalesRepEmployeeNumber
1	141	Euro+ Shopping Channel	715739.0	1370
2	124	Mini Gifts Distributors Ltd.	584188.2	1165
3	114	Australian Collectors, Co.	180585.1	1611
4	151	Muscle Machine Inc	177913.9	1286
5	148	Dragon Souveniers, Ltd.	156251.0	1621
6	323	Down Under Souveniers, Inc	154622.1	1612
7	187	AV Stores, Co.	148410.1	1501
8	276	Anna's Decorations, Ltd	137034.2	1611
9	321	Corporate Gift Ideas Co.	132340.8	1165
10	146	Saveley & Henriot, Co.	130305.4	1337

[R script]



```
# data load
customers=read.csv("./csv/customers.csv",
stringsAsFactors=F,na.strings=c('NA','NULL'))
payments=read.csv('./csv/payments.csv',
stringsAsFactors=F,na.strings=c('NA','NULL'))

library(dplyr)

# data preprocessing

data=customers %>% inner_join(payments) %>%
group_by(customerNumber) %>%
summarise(customerNumber=paste(unique(customerNumber),collapse=','),
customerName=paste(unique(customerName),collapse=','),
amount=sum(amount,na.rm=T),
SalesRepEmployeeNumber=paste(unique(salesRepEmployeeNumber),collapse=',')) %>%
arrange(desc(amount)) %>%
slice(1:10)
```

[Explanations]

먼저, data load를 했다.

주가 customers고, 서브가 payments 기 때문에

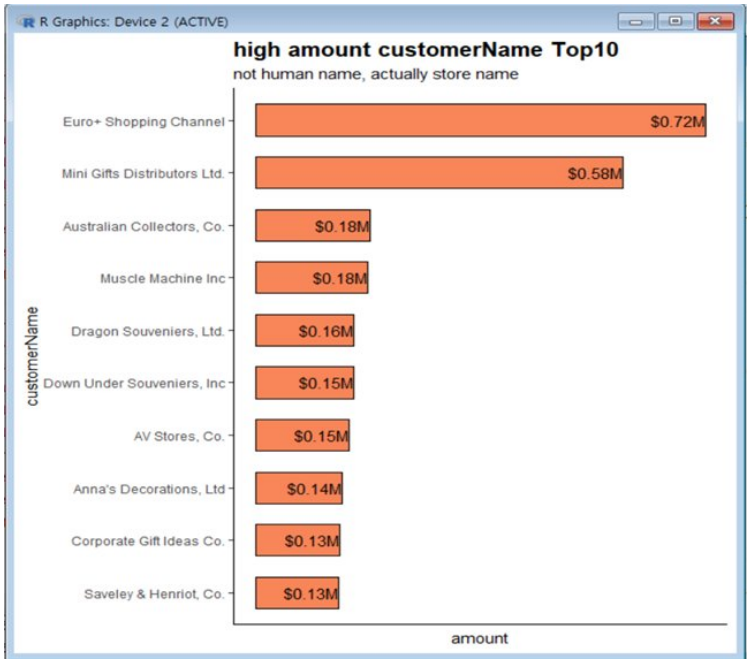
customers에 맞춰 inner_join을 했다.

후에 살려야 하는 column 들은 `paste(unique(...)...)` 로 살려놓았고,

총 매출합 내림차순으로

Top 10 개만 추렸다.

[Graph Output] Bar plot



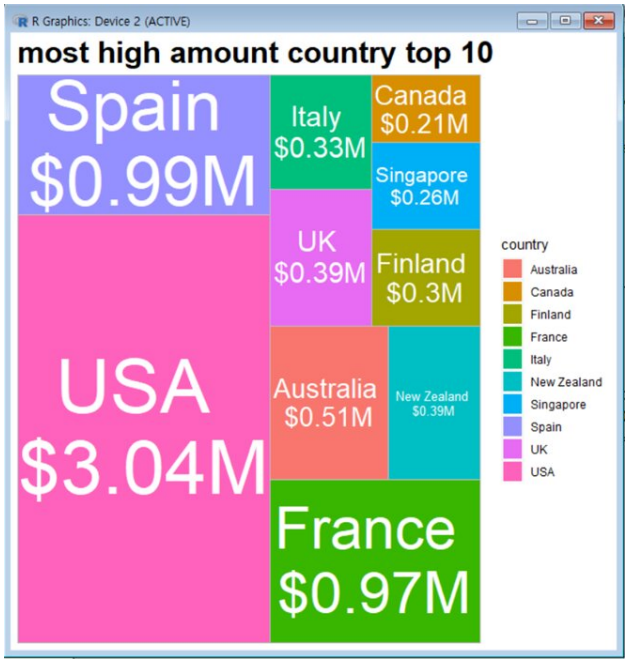
[R script]

```
# graph 1-1

library(ggplot2)
library(glue)

str="${round(amount/1000000,digit=2)}M"

ggplot(data,aes(y=reorder(customerName,amount),x=amount))+
geom_bar(stat='identity',colour='black',
fill='#f55211',alpha=0.7,width=0.6)+
geom_text(aes(label=glue(str)),color='black',hjust=1)+
labs(title='high amount customerName Top10',
subtitle='not human name, actually store name',
x='amount',y='customerName')+
theme_classic()+
theme(plot.title=element_text(size=15,face='bold'),
axis.ticks.x=element_blank(),
axis.text.x=element_blank())
```

[Explanations]

먼저, 데이터가 숫자가 거의 없다.
customerName. 숫자지만 고객을 분류해놓은 id 같은거다. 즉 연속적이고 의미있는 수가 아니다.
creditNumber 도 같은 느낌이다.
하지만, 반대로 보면 그만큼 기준이 많다는 것이다. 모두 분류형 숫자가 될 수 있기 때문이다.
그래서 수를 셀 수 있는 amount에 집중했고, 기준을 매장이름으로 잡았다.
가장 기본적인 plot을 시각화 했다.

[Graph Output] Treemap



[R script]

```
# graph 1-2

library(treemapify)

data1=customers %>% inner_join(payments) %>%
group_by(country) %>%
summarise(customerName=paste(unique(customerName),collapse=','),
country=paste(unique(country),collapse=','),
amount=sum(amount)) %>%
arrange(desc(amount)) %>%
slice(1:10)

str="{country} \n ${round(amount/1000000,digit=2)}M"

ggplot(data1,aes(area=amount,fill=country,
label=glue(str)))+
geom_treemap()+
geom_treemap_text(color='white',place='centre',grow=T)+
labs(title='most high amount country top 10')+
theme(plot.title=element_text(size=23,face='bold'))
```

[Explanations]

1-2 은 treemap graph 로 매출합 순 top10 나라를 알고 싶었다.

이번 그래프는 기준을 매장이름으로 잡았다.
저번 HW3 도 그렇고, 원하는 table을 만들고 나서 그 table을 위한 sub plot을 density plot이나 scatter plot으로 도와줬다.
이번엔 매장을 알았으니 각 기업은 매장을 글로벌진출을 할 것이다.
어느 나라에서 소비가 좋은 지 확인하고 그 나라의 매장에 대해서 더 지원할 수 있다.
(먼저 아이템을 푼다든지. 이벤트를 많이 열어 소비를 촉진시키는 등.)

[Key bindings]

먼저, 1-1 그래프에서 매출합 내림차순으로 정렬 했을 때,
Euro+ shopping channel 과 Mini gifts distributors Ltd. 매장이 가장 많은 매출을 올린 것을 확인 할 수 있다.
위 데이터는 어떤 아이템들에 대해서 가장 많은 수익을 올린 매장을 보는 데이터다.
그랬을 때, 단순히 어떤 매장이 매출이 잘 되는 매장인지 한눈에 볼 수 있는 장점이 있다.
내가 원하는 모양대로 나왔다.

또, 어떤 나라에서 가장 큰 매출이 발생하는지 확인하고 싶었다.

그래서 1-2 그래프로 어떤 나라에서 가장 큰 매출이 발생하는지 treemap으로 그려봤다.
미국이 1등 스페인이 2등 프랑스가 3등으로 다른 나라들에 비해 2배~6배로 차이가 났다.

실제로 찾아보니 미국에 Mini gifts distributors Ltd. 와 함께 다른 매출 많은 매장들 이름이 있었고,
Euro+ shopping channel 은 스페인에 있었다.
이 뜻은 매장이름이 나라마다 하나밖에 없다는 뜻이다.

원했던 내용은 어떤 나라에서 매출이 높고,
어떤 매장에서 매출이 높은지 확인하려고 했던 것이라 원하는 내용은 나왔지만,
위 그래프들로는 뭔가 아쉽다.

만약 scatter pie chart 같은 그래프로 세로를 나라 가로를 매장 으로 해놓고 amount에 따라
크기를 다르게 해놓았다면 더 보기 좋은 그래프가 나왔을 것 같다.

[Table2 Output] Top10 productCode with high margin

Data: data2

	productCode	productName	margin	gap
1	S18_3232	1992 Ferrari 360 Spider red	135996.78	33.87
2	S10_1949	1952 Alpine Renault 1300	95282.58	42.86
3	S12_1108	2001 Ferrari Enzo	93349.65	41.56
4	S10_4698	2003 Harley-Davidson Eagle Drag Bike	81031.30	38.73
5	S12_1099	1968 Ford Mustang	72579.26	38.91
6	S12_3891	1969 Ford Falcon	72399.77	32.87
7	S18_2795	1928 Mercedes-Benz SSK	68423.18	33.75
8	S12_2823	2002 Suzuki XREO	67641.47	30.12
9	S18_1662	1980s Black Hawk Helicopter	64599.11	31.54
10	S18_3685	1948 Porsche Type 356 Roadster	62725.78	26.84

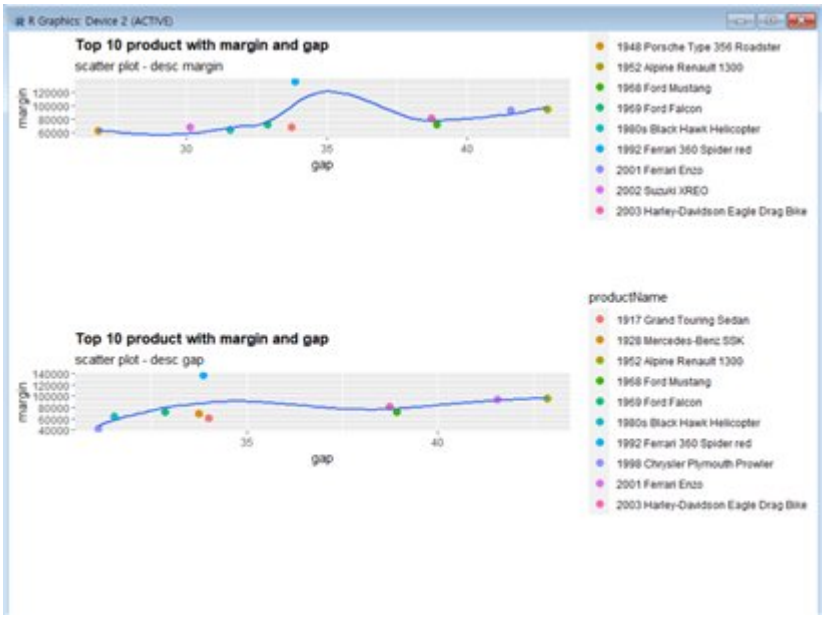
[R script]

```
# table 2
# data load
products=read.csv('./csv/products(1).csv',
stringsAsFactors=F,na.strings=c('NA','NULL'))
od=read.csv('./csv/orderdetails.csv',
stringsAsFactors=F,na.strings=c('NA','NULL'))

library(dplyr)

data2=od %>% inner_join(products) %>%
group_by(productCode) %>%
summarise(productCode=paste(unique(productCode),collapse=', '),
productName=paste(unique(productName),collapse=', '),
margin=sum((priceEach-buyPrice)*quantityOrdered),
gap=max(MSRP-priceEach)) %>%
arrange(desc(margin)) %>%
slice(1:10)
```

[Explanations]

저번 `midterm` 때 계산을 틀려서 잘못된 table이 나왔다.
저번과 다른 것이 있다면 margin을 order 별 priceEach에서 buyPrice를 빼고 그것의 Order수를 곱했다는 것이다.
또, gap을 조사하는 이유는 gap이 크면 보통 margin이 크지 않나? 하는 이유 때문이다.
왜냐하면 gap은 MSRP 라는 권장소매가격인데 보통 만들 때 이걸 비싸게 해놓고서 나중에 팔 때 싸게 팔면 이거 싸게 나왔구나 하는 생각을 가진다고 한다. (차 관련 돼서 많이 사용한다.)
그래서 확인해보니 실제로 gap이 클수록 margin이 컸다.
(위는 margin의 내림차순으로 정리했고, 마지막 arrange(desc(gap)) 하면 다른 productName은 두 개 밖에 없다.

[Graph Output] Scatter plot



[R script]

```
# graph 2-1

library(ggplot2)
library(cowplot)

g2<-ggplot(data2,aes(x=gap,y=margin))+
geom_point(stat='identity',aes(color=productName),size=3)+
geom_smooth(se=FALSE)+
labs(title="Top 10 product with margin and gap",
subtitle="scatter plot - desc margin",
x="gap",y="margin",color="productName")+
theme(plot.title=element_text(size=13,face='bold'))

g3<-ggplot(data3,aes(x=gap,y=margin))+
geom_point(stat='identity',aes(color=productName),size=3)+
geom_smooth(se=FALSE)+
labs(title="Top 10 product with margin and gap",
subtitle="scatter plot - desc gap",
x="gap",y="margin",color="productName")+
theme(plot.title=element_text(size=13,face='bold'))

plot_grid(g2,NULL,g3,label='AUTO',ncol=1)
```

[Explanations]

gap 이 클수록 margin이 크다는 그래프를 보았다.
data는 margin 으로 내림차순한 data2, gap으로 내림차순한 data3로 했다.
두 그래프를 한번에 쓰기 위해서 cowplot을 사용했고,
조정하는 법을 구글링하지 못해서 figure size가 이상하다.

smooth(se=False)로 추세선을 그렸고 실제로 우상향이다.

[Graph Output] Heatmap

	Classic Cars	Motorcycles	Planes	Ships	Trains	Trucks and Buses	Vintage Cars	Total
Autoart Studio Design	27	55	28	27		28	56	221
Carousel DieCast Legends	81			54	27	28	56	246
Classic Metal Creations	159		28		27	28	28	270
Exoto Designs	76	28				28	108	240
Gearbox Collectibles	132	27	28		27		28	242
Highway 66 Mini Classics	55	83	28			28	28	222
Min Lin Diecast	81	28	28	28			55	220
Motor City Art Classics	28		84			56	81	249
Red Start Diecast	25	28	28	27		28	53	189
Second Gear Diecast	109	27	56				28	220
Studio M Art Models	51	27		27		28	84	217
Unimax Art Galleries	78	28	28	55		28	27	244
Welly Diecast Productions	108	28		27		28	25	216
Total	1010	359	336	245	81	308	657	2996

[R script]

```
library(pivottabler)

data=od %>% inner_join(products)

pt=PivotTable$new()
pt$addData(data)
pt$addColumnDataGroups("productLine")
pt$addRowDataGroups("productVendor")
pt$defineCalculation(calculationName="productName", summariseExpression="n()")
pt$theme="standardtable"
pt$setStyling(rowNumbers=3,columnNumbers=1,
declarations=list("background-color"="pink"))
pt$setStyling(rowNumbers=6,columnNumbers=2,
declarations=list("background-color"="pink"))
pt$setStyling(rowNumbers=8,columnNumbers=3,
declarations=list("background-color"="pink"))
pt$setStyling(rowNumbers=12,columnNumbers=4,
declarations=list("background-color"="pink"))
pt$setStyling(rowNumbers=c(2,3,5),columnNumbers=5,
declarations=list("background-color"="lightblue"))
pt$setStyling(rowNumbers=8,columnNumbers=6,
declarations=list("background-color"="pink"))
pt$setStyling(rowNumbers=4,columnNumbers=7,
declarations=list("background-color"="pink"))
pt$renderPivot()
```

[Explanations]

이번 그래프는 어떤 공급업체가 product를 많이 냈는지 heatmap 으로 그렸다.
사실 heatmap하면 전체가 색깔로 알 수 있는 걸 보통 부르는데,
이번엔 그냥 중요한 부분만 styling 했다.
먼저 `공급업체 별 total` 은 평균 210을 기준으로 `차이가 많이 없지` 만,
각 공급업체마다 맡고있는 product들이 다르다.
Exoto design은 plane,ship,train 같은 덩치큰 것은 과감히 포기하고 운송에 투자한다면,
Classic Metal Creation은 주로 차에 많이 투자하고, 다른곳에 적게 발을 담그고 있다.

[Key bindings]

저번 중간시험 때 틀린 수식으로 잘못된 정보를 분석했다.
MSRP는 권장소비가격이고 보통 차나 값비싼 product를 팔 때 업체에서 붙인다고 한다.
그래서 MSRP가 높다가 팔 때 싸게 올려놓는다고 많이 한다.
이번 table2의 gap column이 MSRP 와 Price간의 가격 차이를 본 것이다.
또, product 별 margin을 조사했다. 이는 어떤 product 가 인기가 많고, 많이 팔렸는지 알기 위해서 이고, gap 과 어떤 관계가 있는지 알기 위해서 구했다.

실제로 gap과 margin은 서로 상관관계가 있었다.

더 나아가, 공급업체 별 그래프를 통해서 어떤 공급업체가 어떤 product를 담당하는지 조사해봤다.
왜냐하면 table에서 알 수 있기도 하고 일단 차가 거의 판매의 대부분이다.
그래서 공급업체 간 차이점을 보려고 heatmap을 준비했고, 업체들은 보통 차에 많이 투자하지만, 몇몇 업체들은 차가 아닌 바이크나 비행기에 많이 투자하는 것을 알았다.

아쉬운 점은 업체 간 margin 이나 매출도 조사해보면 더 좋을 것 같다.

[Bonus Question]

데이터 조작에서 비슷한 역할을 하는 R, SQL의 개인적으로 느낀 프로그래밍 유사점은

- 특정 목적에 즉 데이터 분석 편리하게 프로그래밍 됐다는 것이다.
  - 특히 SQL 같은 경우는 MySQL을 배우면서 되게 쉽게 배웠다는 것이다.
  - 평소에 Python을 공부했었기 때문에 데이터분석 라이브러리나 명령어들이랑 match 가 쉬웠다.

```
## R
1. read.csv(path)
2. df %>%
    groupby(column) %>%
    summerise(name=mean(column))
## SQL
1. select column from df
2.  select mean(column)
    from df
    group by(column)
```

반대로 R, SQL의 개인적으로 느낀 차이점은

- 문법이 다르다.
  - 이게 무슨뜻인가 하면 SQL은 select (선택해라) from (어디서) group by(묶고) ... 이런식이다.
  - 영어식이라는 뜻이다. 명령어 인자. 명령어 인자.
  - 그래서 나중에 sql은 많이 까먹어도 다시 찾아볼 수 있다. 영어 생각하면 편리하기 때문에.
  - 하지만 R은 다르다. R은 그럼에도 지켜야하는 뭔가 순서가 있다. 모듈 + 인자 = 명령어 랄까 약간 헷갈린다.