2023-04-25 화

1. fromJSON에서 값을 받아올 때 계속 에러를 일으키면 fromJSON(url())을 사용해라

2. data.frame (즉, table)에서 length를 하면 column수를 나타낸 것이고 nrow를 하면 행수를 나타내는 것이다. 입력되는 데이터의 개수를 알고 싶으면 nrow를 쓰는 것이 더 타당하다

3. nrow(데이터테이블) == 0 과 “NULL”을 다르다. 데이터 테이블이 NULL일 경우, nrow, length 등을 사용하여도 NULL을 반환한다.
   따라서 NULL를 걸러내기 위해서는 is.null(데이터테이블) == T를 사용하여야 된다.

4. try와 tryCatch의 차이점:
   * try : 데이터 값과 error 이유를 같이 출력해줌으로 변수를 필요로 한다.
   * tryCatch : 데이터의 에러가 발생시 어떤 것을 수행할 지만을 받으며 따로 값을 반환하지는 않는다.

   [try]

```
x <- "100"
y <- "abc"
z <- "200"

# Attempt to convert x, y, and z to numeric
result_x <- try(as.numeric(x))
result_y <- try(as.numeric(y))
result_z <- try(as.numeric(z))

# Print the results
print(result_x)

## [1] 100
## attr(,"class")


print(result_y)

## [1] "try-error"
## attr(,"condition")


print(result_z)

## <simpleError in eval(expr, envir, enclos): invalid 'type' (character)
of argument>
```

```
## [1] NA
```

[tryCatch 1.] —> ERROR 발생시 특별한 조치 없이 넘어감

```
tryCatch({
  # code to be executed
})
```

[tryCatch 2.] —> ERROR 발생시 다음에 어떤 행동을 수행하라는 명령

```
tryCatch({
  # code to be executed
}, error = function(e) {
  # code to handle the error
})
```

5. 2중 함수 구문에서 값을 반환하기 위해서는 그냥 전역변수를 사용해라
   —> 위의 방식은 확인할 필요가 있지만 현재까지의 최선으로 생각됨

6. for문에서 마지막 실행을 할 때 next를 만나면 오류를 일으킴.
   따라서 이 부분을 trycatch({실행할 구문})으로 처리해주어야함

2023-04-26 수

1. R에도 append function이 존재한다

   append(기존리스트, 추가할 값, after = 앞에 값을 몇 번째 뒤에 넣을 것인지)

```
append(x, values, after = length(x))

x <- c(1, 2, 3)
y <- c(4, 5, 6)

# Append y to x
z <- append(x, y)

# Append a single value to z
z <- append(z, 7)

# Append a single value after position 2 in z
z <- append(z, 8, after = 2)
```

```
# Print the final vector
z
## [1] 1 2 8 3 4 5 6 7
```

2. trycatch 구문 안에 있는 next 구문도 잘 작동한다고 함
   (일단 코드를 실행했을 때는 오류가 없었음)

```
tryCatch({
    if(is.null(weather_dttm)== T){
      print(paste("weather_dttm is NULL in ",stnid,"\n date
:",start_date," ~ ",end_date))
      next
    }

    if (nrow(weather_dttm) == 0){
      print(paste("there is no data in ",stnid,"\n date :",start_date,"
~ ",end_date))
      next
    }
```

3. do.call 함수
   `do.call()`은 인수 목록을 사용하여 함수를 호출할 수 있게 해주는 R의 강력한 함수입니다. 이는 함수에 전달하려는 인수 목록이 있지만 인수의 수를 알 수 없거나 변경될 수 있는 경우에 특히 유용할 수 있습니다.

```
# Define a function that takes two arguments and returns their sum
my_function <- function(a, b) {
  return(a + b)
}

# Create a list of arguments
my_args <- list(a = 2, b = 3)

# Call the function using the list of arguments
result <- do.call(my_function, my_args)

# Print the result
print(result) # Output: 5
```

`do.call()`을 사용하는 한 가지 이점은 함수에 전달하려는 인수 목록을 동적으로 생성할 수 있다는 것입니다. 예를 들어 사용자 입력 또는 파일의 데이터를 기반으로 인수 목록을 만들 수 있습니다.

2023-04-27 목

1. 날짜와 시간이 같이 있는 걸로 변환시킬 때는 as.POSIXct를 사용한다.
   [ex 1]

```
weather_dttm_temp <- weather_dttm_temp %>%
  mutate(dttm = as.POSIXct(tm, format = "%Y-%m-%d %H:%M"))
```

   [ex 2]

```
a <- seq(as.POSIXct('2023-03-01 00:00:00'),as.POSIXct('2023-03-31
23:00:00'), '1 hour')
```

2. 반복적인 컬럼을 만들어서 나중에 값을 비교하기 유용하게 만들려면 rep와 seq를 사용한다.
   아래에서 유의할 점은 리스트 형식으로 만들어지면 리스트를 한번 다 돌도 다음으로 돌기 때문에 rep(each = 몇번)을 이용하여 각각 여러번 값이 만들어지게 해야함

```
stnid <- c(
  90,93,95,98,99,100,101,102,104,105,106,108,
  112,114,115,119,121,127,129,130,131,133,
  135,136,137,138,140,143,146,152,155,156,
  159,162,165,168,169,170,172,174,177,184,
  185,188,189,192,201,202,203,211,212,216,
  217,221,226,232,235,236,238,239,243,244,
  245,247,248,251,252,253,254,255,257,258,
  259,260,261,262,263,264,266,268,271,272,
  273,276,277,278,279,281,283,284,285,288,
  289,294,295
)


df_dttm <- data.frame(dttm0 = rep(a, times = 95),
                      stnid0 = rep(stnid, each = 744))
```

3. 각각에 있는 컬럼값을 합친 새로운 컬럼을 만들려면 paste를 사용한다.

```
df_dttm$dt_stn <- paste(df_dttm$dttm0, df_dttm$stnid0,sep = "_")
```

4. 결측치가 있는 행을 반환하는 코드

```
# 결측치를 table로 보여주는 코드
view(df_dttm[is.na(df_dttm$tm),])

# 결측치의 개수를 세어주는 코드
```

```
nrow(df_dttm[is.na(df_dttm$tm),])
```

2023-04-28 금

1. 이름이 같은 행의 열끼리 값을 합치고 싶으면 group_by와 summarise를 사용하며 summarise에서 컬럼명을 사용한다.

   아래에서 값은 Y19, … , Y23 컬럼명이다.

```
all_wide <- all_wide %>% group_by(id) %>%
  summarise(Y19 = sum(Y19),
            Y20 = sum(Y20),
            Y21 = sum(Y21),
            Y22 = sum(Y22),
            Y23 = sum(Y23))
```

2. 특정 조건을 가진 행만 값을 특별처리 해주기 위해서
   데이터 프레임으로 특정조건을 가진 행만 가져오도록 한 후
   mutate_if로 값을 처리해준다.
   그리고 다시 그 특정조건의 행에 저장 해야한다.

```
sensor_wide[sensor_wide$frmhs_id == "김기택",]<- sensor_wide %>%
  filter(frmhs_id == "김기택") %>%
  mutate_if(is.numeric, function(x) x * 7)

sensor_wide[sensor_wide$frmhs_id%in%
c("김성진","이근재","이형재","전성욱","황국선") ,] <- sensor_wide %>%
  filter(frmhs_id %in% c("김성진","이근재","이형재","전성욱","황국선"))%>%
  mutate_if(is.numeric, function(x) x*6)
```

   *** mutate_if()
   mutate_if()은 지정해준 모든 변수에 대해 계산식을 적용시켜 줍니다. 예를 들어, 데이터의 타입을 변경하고 싶을 때, as.factor()같은 명령어를 하나씩 적용시키는 방법은 시간을 많이 잡아먹는 노가다 작업입니다. 이런 경우, mutate_if()를 통해 한번에 데이터 타입을 변경시켜줄 수 있습니다.

3. 특정 컬럼을 선택하지 않을 때는 '!'를 사용한다.
   '!'는 not의 의미를 내포하고 있다.
   그리고 가져오는 컬럼에 대한 내용은 ()로 감싼다.

```
sensor_wide[!(sensor_wide$frmhs_id %in%
c("김기택","김성진","이근재","이형재","전성욱","황국선")),] <-
  sensor_wide %>%
```

```
  filter(!(frmhs_id %in%
c("김기택","김성진","이근재","이형재","전성욱","황국선"))) %>%
  mutate_if(is.numeric, function(x) x *8)
```

2023-05-02 화

1. A테이블에 있던 정보를 B 테이블로 옮길 때 (단, 안의 열의 이름이 같아야 한다.)

```
insert into tb_weather_asos_dttm
select *
  from tb_weather_asos_dttm_temp;
```

2023-05-03 수

1. rbind는 열의 이름과 수가 같아야지 합칠 수 있다.

2. ggplot에서 그래프를 만들때, ggplot(데이터(테이블), aes(x축값, y축값)) 이렇게 하며 group별로 색을 따로 주고 싶을 때는 aes(colour = 컬럼명), 이렇게 넣으면 된다.

```
ggplot(table2, aes(year, count))+
  geom_line(aes(group = country, colour = country))
```

2023-05-15 월

1. matrix(데이터, 행수, 열수) : matrix는 컬럼(열) 순으로 데이터를 채움
   데이터를 행 순으로 채우고 싶을 때는 byrow = 1 로 채우면 됨

2. rownames : 행이름 채우기
   colnames : 열이름 채우기
   `rownames <- NULL, colnames <- NULL` :행이름이나 열이름 비우기

3. R은 vector로 되어있기 때문에, 연산을 할 때 그냥 벡터(행렬)형태로 계산식을 적으면 됨(작은 단위로 쪼개서 계산할 때 오히려 시간이 더 오래 걸림)

```
 n1 <- 1:5
 n2 <- 6:10
 n2/n1
#[1] 6.000000 3.500000 2.666667 2.250000 2.000000
```

4. Data가 matrix가 아니라 vector여서 제대로 작동하지 않음
   따라서 dataframe에서 하나만 추출해서 그래프를 그릴려면 [1,,drop=F]를 이용해라

```
Data <- MinutesPlayed[1,,drop = F]
matplot(t(Data), type = "b", pch = 15:18, col = c(1:4,6))
legend("bottomleft", inset = 0.01, legend = Players[1], col = c(1:4,6),
pch = 15:18)
```

5. Error in plot.new() : figure margins too large
   위의 에러가 나타날 때는, matplot을 불러오기 전에
   par(mar = c(여백크기,여백크기,여백크기,여백크기))를 설정해줌

```
myplot <- function(data, rows=1:10){
  Data <- data[rows,,drop=F]
  par(mar=c(3, 3, 1, 1))
  matplot(t(Data), type="b", pch=15:18, col=c(1:4,6))
  legend("bottomleft", inset = 0.01, legend = Players[rows], col =
c(1:4,6), pch = 15:18, horiz = F)
}
```

6. gather() : wide_format을 long_format으로 변환하는데 사용
   spread() : long_format을 wide_format으로 변환하는데 사용

```
gather(
data = 데이터,
key =  "긴 형태로 나열하게 될 변수명",
value =  "key 변수에 대한 값 변수명",
-var1, -var2, ... # 고려하지 않는 변수
)

data %>%
gather(
key = "긴 형태로 나열하게 될 변수명",
value = "key 변수에 대한 값 변수명",
-var1, -var2
)
```

```
spread(
data = 데이터,
key = "넓은 형태로 나열하게 될 변수",
value = "key  변수에 대한 값"
```

```
)
data %>%
spread(
key = "넓은 형태로 나열하게 될 변수",
value = "key 변수에 대한 값"
)
```

2023-05-17 수

1. vector를 재사용하기 위해서는 vector가 전체 데이터 수의 약수이어야 한다.
   전체 데이터수가 vector의 개수의 배수이어야 한다.

2. column을 삭제하는 방법 : 데이터$컬럼명 <- NULL

```
stats$MyCalc <- NULL
stats$xyz <- NULL
```

2023-05-18 목

1. vector를 data.frame을 이용하여 합쳐서 colnames를 사용하여 컬럼명을 변환시킬 수 있음
   또는 data.frame(컬럼명1=vector1, 컬럼명2=vector2, …)를 사용하여 컬럼명을 변환시킬 수 있음

```
#1
mydf <- data.frame(countries_2012_dataset,
codes_2012_dataset,regions_2012_dataset)
columnnames(mydf) <- c("Country","Code", "Region")

#2
mydf <- data.frame(Country=countries_2012_dataset,
Code=codes_2012_dataset, Region=regions_2012_dataset)
```

2. 실수확률이 적은 통합된 데이터를 기준으로 데이터를 합쳐라
   merged해서 겹친 열(변수)가 있으면 NULL로 지워라

```
merged <- merge(stats, mydf, by.x = "Country.Code", by.y = "Code")

merged$Country <- NULL
```

3. qplot에서 그래프에 표시되는 색상을 변화시킬 때, I 파라미터를 사용한다.

```
#1. Shapes
```

```
qplot(data=merged, x = Internet.users, y = Birth.rate, colour = Region,
      size=I(5), shape = I(17))

#2. TransParency
qplot(data=merged, x = Internet.users, y = Birth.rate, colour = Region,
size = I(5),
      shape = I(19), alpha = I(0.6))

#3. Title
qplot(data=merged, x=Internet.users, y = Birth.rate, colour = Region,
      size = I(5), shape = I(19), alpha = I(0.6), main = "Birth Rate vs
Internet Users")
```

4. qplot에서 색상 전체를 같은 색을 변환하고 싶으면 colour = I("Blue")로 작성하고 특정 그룹별로 다른 색을 넣고 싶은면 I 인자를 사용하지 않고 작성한다.
   colour = 변수(=컬럼명) 을 작성한다.

```
qplot(data=merged, x = Internet.user, y = Birth.rate, colour = Region)
qplot(data=merged, x = Internet.user, y = Birth.rate, colour =
I("Blue"))
```

2023-05-19 목

1. 숫자로 되어 있는 값을 명목형 변수로 바꿀 때는 factor를 이용한다.
   아래 처럼 재할당 해주는 과정이 필요함

```
movies$Year <- factor(movies$Year)
```

2. aes() 안에는 x좌표, y좌표, colour, size 등을 넣을 수 있다.
   size를 할 때는 숫자변수로 하는 것이 좋다(등목변수, 비율변수)

```
ggplot(data=movies, aes(x=CriticRationg, y =AudienceRation,
colour=Genre, size = BudgetMillions))+
  geom_point()
```

3. ggplot에서 그래프를 그리기 위해서는 geom_point(), geom_line() 등의 항목이 필요하다.
   ggplot을 그래프 층을 쌓는 것이므로 마지막에 넣는 데이터가 잘 보인다.

```
p <- ggplot(data = movies, aes(x=CriticRation, y = AudienceRating,
colour=Genre, size = BudgetMillions))

#점
```

```
p + geom_point()

#선
p + geom_line()

# 여러 층
p + geom_point() + geom_lines() #선이 더 잘 보임
p + geom_line() + geom_point()  # 점이 더 잘 보임
```

4. 특정 그래프마다 aes를 따로 지정해 줄 수 있으며 새로 지정한 aes는 이전의 그래프에 영향을 미치지 않는다.
   원하는 점의 미관만 변화시킬 수 있다.

   aes는 미관(aesthetic)에 변수(variable)을 맵핑하는 것임
   -> 그래서 맨 마지막의 geom_line(size = 1)과 다름

```
q <- ggplot(data = moview, aes(x = CriticRating, y - AudienceRating,
                               colour = Genre, size = BudgetMillions))

q + geom_point()



# aes override
q + geom_point(aes(size = CriticRating))
q + geom_point(aes(colour = BudgetMillions))
q + geom_point()
q + geom_point(aes(x = BudgetMillions))

#reduece line size
q + geom_line(size = 1) + geom_point()
```

2023-05-20 토

1. facet_grid를 사용하여 한페이지에 여러 그래프로 나눌 때, scale의 범위를 다르게 설정하고 싶으면(y축 값이 통일 되어 있어 제대로 그래프 값의 확인이 어려운 경우), facet_grid(scales = “free”)를 사용하면 된다.

```
v + geom_histogram(binwidth = 10, aes(fill=Genre), colour = "Black") +
  facet_grid(Genre~., scales = "free")
```

2. ggplot에서 바로 ylim, xlim을 적용할 경우 특정 데이터를 삭제하는 것임
   scatter plot과 같은 그래프에서 유용, 연속해서 나타내는 histogram에는 좋지 않음

```
m <- ggplot(data=movies, aes(x = CriticRating, y = AudienceRating,
```

```
                              size = BudgetMillions,
                              colour = Genre))

m + geom_point()

m + geom_point() +
  xlim(50,100) +  # xlim, ylim을 사용하면 실제로 확대하는 것이 아니라
데이터를 제거하여
  ylim(50,100) # 데이터를 확대하는 것임 -> scatter plot에서는 유용하지만
히스토그램과 같은 그래프에는 유용하지 않음
```

3. histogram이나 facet_grid에서 데이터를 확대하고 싶으면 coord_cartesian()에 ylim과 xlim을 적용한다. 이때 ylim = c(0,50)처럼 벡터로 넣는 것을 잊지 않는다.

```
n <- ggplot(data = movies, aes(x = BudgetMillions))
n + geom_histogram(binwidth = 10, aes(fill = Genre), colour = "Black") +
  coord_cartesion(ylim = c(0,50))

# coord_cartesion()에 ylim을 사용하면 데이터를 확대할 수 있음

n + geom_point(aes(size = BudgetMillions)) +
  geom_smooth() +
  facet_grid(Genre~Year)+
  coord_cartesian(ylim=c(0,100))
# facet_grid와 coord_cartesian() ylim을 사용할 수 있음 그리하여 특정부분만
# 확대하여 나타냄
```

4. 레전드의 위치가 잘릴 수 있음에 유의!
   legend.position과 legend.justification은 벡터로 적음

```
# legend formatting
h +
  xlab("Money Axis") +
  ylab("Number of Movies") +
  theme(axis.title.x = element_text(colour = "DarkGreen", size = 30),
        axis.title.y = element_text(colour = "Pink", size = 30),
        axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20),

        legend.title = element_text(size = 30),
        legend.text = element_text(size = 20),
        legend.position = c(1,1), # 여기까지만 적으면 legend가 짤린다.
        legend.justification = c(1,1)) # 여기에서 끝이 어디까지 인지?
설정해 주어야 안 짤림
```

5. 전체 그래프의 제목을 넣을 때는 ggtitle을 사용하며, title의 스타일을 변경시킬 때는 theme의 plot.title = element_text(colour = “DarkBlue”, size = 40, family = “Courier”) 을 이용함

```
# title
h +
  xlab("Money Axis") +
  ylab("Number of Movies") +
  ggtitle("Movie Budget Distribution") +
  theme(axis.title.x = element_text(colour = "DarkGreen", size = 30),
        axis.title.y = element_text(colour = "Red", size = 30),
        axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20),

        legend.title = element_text(size = 30),
        legend.text = element_text(size = 20),
        legend.position = c(1,1),
        legend.justification = c(1,1),

        plot.title = element_text(colour = "DarkBlue",
                                  size = 40,
                                  family = "Courier" ))
```

2023-05-21 월

1. ggplot의 title을 설정할 때,
   ggtitle(“titlename”) 과 labs(title =”titlename”)을 사용할 수 있음
   마찬가지로 xlab(“x축 제목”),ylab(“y축 제목”)은 labs(x = “x축 제목”, y = “y축 제목”)으로 작성할 수 있음

```
# ver.1
ggplot(mpg, aes(displ, hwy)) + geom_point() + xlab("Displacement") +
ylab("Highway") + ggtitle("Plot title")

# ver.2
ggplot(mpg, aes(displ, hwy)) + geom_point() + labs(title = "Plot title",
x = "Displacement", y = "Highway")
```

2. 저장할 때, ggsave를 사용함

```
ggsave( # 그냥 저장하면, 화면에서 보여지는 그래프 크기 그대로 저장함
  filename,
  plot = last_plot(), # 어떤 plot을 저장할 것인지 지정해줌, 기본으로 가장
```

```
마지막에 작업한 plot을 저장함
  device = NULL, # 어떤 형식의 데이터를 저장하고 싶은지 지정할 수 있음
  path = NULL, #저장위치, 따로 입력해주지 않으면 현재 디렉토리에 저장
  scale = 1, # 확대할 것인지, 축소할 것인지 설정할 수 있음
  width = NA,
  height = NA,
  units = c("in","cm","mm"),# 넓이와 크기의 단위를 무엇으로 할 것인지
설정할 수 있음
  dpi = 300, # 해상도
  limitsize = TRUE,
  ...
)
```

3. 특정 폰트를 설정해서 저장이 안될 때, ggsave(device = cairo_pdf)를 사용

```
q <- ggplot(mpg, aes(displ, hwy)) + geom_point() +
  labs(title = "Plot title",x = "Displacement", y = "Highway") +
theme(text = element_text(size = 15,
family = "Poppins"))
q

ggsave("plot2.pdf", plot = q,device = cairo_pdf) # 폰트 때문에 저장이
안되면 device = cairo_pdf를 사용해야함
```

4.

```
baby <- filter(babynames, name %in% c("Anna","Emma","Clara"), sex ==
"F")

library(viridis)
ggplot(baby, aes(x = year,y = n, group = name, color = name)) +
geom_line() + scale_color_viridis()
# name이 연속적인 값이 아니라 명목변수 이기 때문에,
scale_color_viridis(discrete = T) 로 설정해주어야 함.
# 원하는 값으로 그래프를 나누려면 group = 나눌 기준 컬럼명을 입력하면
컬럼값을 기준으로 그래프가 나누어지고
# 그래프를 나눈 후  color을 사용해야 그래프가 분리되고 색상도 분리되어 칠해짐

ggplot(baby, aes(year, n, group = name, color = name)) + geom_line() +
  scale_color_viridis(discrete = T)

baby <- filter(babynames, name %in% c("Jessica","Natalie","Saray"), sex
== "F")

ggplot(baby, aes(year, n, group = name, color = name)) +
geom_line(linetype = "")
```

5. 첫번째 그래프는 class별로 점과 선이 색이 다르게 생기고
   두번째 그래프는 점만 class 별로 색이 다르고 선도 하나밖에 없음

```
#1
ggplot(mpg, aes(displ, hwy,colour = class)) +
  geom_point() +
  geom_smooth(se = F)

#2
ggplot(mpg,aes(displ, hwy)) +
  geom_point(aes(colour = class)) +
  geom_smooth(se = F)
```

Geoms and stats

“geoms” control the way the plot **looks**
“stats” control the way data is **transformed**

Every geometry has a default stat….

geom_line : default stat is stat_identity
geom_point : default stat is stat_identity
geom_smooth : default stat is stat_smooth

…and each stat has a default geom

stat_smooth : default geom is geom_smooth
stat_count : default geom is geom_bar
stat_sum : default geom is geom_point

Interesting stats

stat_smooth (geom_smooth)
stat_unique (geom_point)
stat_summary (geom_pointrange)
stat_count (geom_bar)
stat_bin (geom_histogram)
stat_density(geom_density)
stat_boxplot (geom_boxplot)
stat_ydensity (geom_violin)

Computed aesthetics

when a stat performs a transformation, new variables are created:

e.g. geom_histogram(stat_bin)
count
    number of points in bin
density
    density of points in bin, scaled to integrate to 1

ncount
count, scaled to maximum of 1
ndensity
density, scaled to maximum of 1

To access them:
Before : **..name..**
Now : **stat(name)**

Displaying distributions

- Distribution of incode in a particular town

- Ways to look at distributions:

1. Scatter plots with some jitter(useful only for small data & we already know how to do this)

2. Histograms and frequency polygons

3. Density plots

4. Boxplots(to show the summary statistics of your distribution) ans violin plots

Histograms & freq.polygons

- Visualize the distribution of a single continuous variable by dividing the x axis into bins and counting the number of observations in each bin
- Histograms(geom_histogram()) display the counts with bars
- Frequency polygons(geom_freqploly()) display the counts with lines.
- Continuous x data.

geom_histogram

- Wrapper for geom_bar + stat_bin
- the importance of binning
  * binwidth* bins

- Map values to y to flip orientation

Boxplots

- A way to capacity display the summary statistics(not the whole distribution) of a continuous variable.

        - The median, tow hinges and two whiskers, and all “outlying” points individually.

- geom_boxplot(stat_boxplot)
    - Interesting params:
        - width & varwidth
        - show.legend
        - outlier.alpha
        - outlier.shape

Violin plots

A violin plot is a compact display of a continuous distribution. It is a blend of geom_boxplot() and geom_density() : a violin plot is a mirrored density plot displayed in the same say as a boxplot

---

2023-05-23 화

1. boxplot 안에 색 비우기
   ggplot에서 geom_boxplot()을 만들때, boxplot의 하얀 색상을 없애고 싶으면, fill = NA를 사용한다.

```
ggplot(mpg, aes(x= hwy, y = class)) +
  geom_violin(aes(fill = class),show.legend = F, color = NA, alpha = 0.4) +
  scale_fill_viridis(discrete = T) +
  geom_boxplot(fill = NA, width = 0.2)
```

2. boxplot, violinplot 안에 색 채우기(변수별로 다르게)
   geom_boxplot, geom_violin 을 사용할 때 안의 색상을 변수 별로 다르게 채우기 위해서는 aes(fill = calss)를 사용하여 색상을 채운다.
   여기서 viridis는 색상팔레트로 연속적인 색상을 나타낸다.
   불연속적인 값에서 viridis를 사용할 때는 discrete = T 를 이용하여 불연속적으로 색상을 채우게 한다.

```
ggplot(mpg, aes(x = hwy, y = class)) + geom_violin(aes(fill = class), show.legend = F, color = NA, alpha = 0.4) +
scale_fill_viridis(discrete = T)
```

3. boxplot, violinplot에서 outlier 색상 진하게 하기
   geom_boxplot, geom_violin 함수 안에서 alpha = 0.4를 하면 outlier도 연해짐
   이때, outlier.alpha = 1 을 사용하면 outlier의 색상이 진해짐
   <전>

```
ggplot(mpg, aes(x = hwy, y = class)) + geom_violin(aes(fill = class),
show.legend = F, color = NA, alpha = 0.4) +
scale_fill_viridis(discrete = T)
```

<후>

```
ggplot(mpg, aes(x = hwy, y = class)) + geom_violin(aes(fill = class),
show.legend = F, color = NA, alpha = 0.4, outlier.alpha = 1) +
scale_fill_viridis(discrete = T)
```

4. Position adjustment

position_dodge()

Dodging preserves the vertical position of a geom while adjusting the horizontal position

position_dodge2()

New version, adds padding, doesn't require a group variable

Parameters:

width (Default = 0.9)

preserve = "single"/"total" (Default = "total")

Only for dodge2:

reverse (Default = F) : bar 의 순서를 변화시킬 때 사용

padding (Default = 0.1) : 막대와 막대 사이에 여백을 줄 때 사용

5. position = "dodge"에서 geom_bar의 bar크기가 하나만 넘 클때
   position = position_dodge(preserve="single")를 사용

```
p + geom_bar(alpha = 0.6, position = position_dodge())
p + geom_bar(alpha = 0.6, position = position_dodge(preserve =
"single"))
```

6. By aesthetic > Position

DATE/DATETIME

To be used with aesthetics that map a variable of the class date/time.

class Date:

- scale_x_date / scale_y_date

datetimes(class POSIXct):

- scale_x_datetime / scale_y_datetime

times (class hms):

- scale_x_time / scale_y_time

7. 어제부터 그전 90일까지 날짜 추출

```
x <- Sys.Date() - 1:90
```

By aesthtic > Color

Continuous:
scale_color_continuous / scale_fill_contitnuous
scale_color_gradient / scale_fill_gradient
scale_color_gradient2 / scale_fill_gradient2
scale_color_gradeintn / scale_fill_gradientn

Binned:
scale_color_binned / scale_fill_binned
scale_color_steps / scale_fill_steps
scale_color_steps2 / scale_fill_steps2
scale_color_stepsn / scale_fill_stepsn

Discrete:
scale_color_discrete/ scale_color_hue / scale_fill_hue
scale_color_grey / scale_fill_grey

VIRIDIS FAMILY:requires package “viridis”
Continuous:
scale_colour_viridis_c / scale_fill_viridis_c

Binned:
scale_colour_viridis_b / scale_fill_viridis_b

Discrete:
scale_colour_viridis_d / scale_fill_viridis_d

USEFUL PARAMETERS

alpha

direction = (default = 1), if -1, reverse order

discrete = generate a discrete palette? (default = F)

option =

“A” = “magma”, “B” = “inferno”, “C” = “plasma”, “D” = “viridis” (the default), “E” = “cividis”

COLORBREWER FAMILY

Continuous:
scale_colour_distiller / scale_fill_distiller

Binned:
scale_colour_fermenter / scale_fill_fermenter

Discrete:
scale_colour_brewer / scale_fill_brewer

USEFUL PARAMETERS

type = “seq” (sequential, the default),
“div” (diverging),
“qual” (qualitative)

direction = (default = 1), if -1, reverse order

palette = Name of palette of index

Manual scales

SCALE_*_MANUAL

All of the previous scales can be used in manual mode.
Useful when you want to specify your own set of mappings from levels in the data to aesthetic values.
Parameter.”values”
Only available for discrete scales.

Examples
Choosing your set of colors sin discrete color scale
Specifying your own set of alpha’s
Specifying your own set of shapes
Specifying your own set of linetypes

Shortcuts

Simple manipulation of labels and limits

LABS

Shortcuts to modify axis, legend, and plot labels
xlab, ylab: modify x and y-axis label names
Most convenient way: use labs with arguments:
title
x
y
subtitle
caption
tag

LIMS

- Shortcuts to modify limits of the plot
- Either use xlim and ylim alone:
    - xlim(0,50) and ylim alone:
- Or use lims specifying vectors:
    - lims(y = c(0,50), y = c(NA,40))

Bar plots

A bar plot shows the relationship between a numeric and a categoric variable
Each bar is one categoric variable
The length of the bar encodes the numeric vlaeu

Bar plots vs histograms

geom_histogram()
stat_bin() : bins and counts
Requires continuous x data

geom_bar()
stat_count() : counts
Discrete and
continuous x data

geom_bar vs geom_col
geom_bar()
stat = “count”
Needs x

geom_col()
stat = “identity”
Needs x and y

Better : geom_point()

Bar plot

Two categories:
stacked or grouped or filled or small multiples?

If we have two categorical groups:
e.g. diamonds dataset:
group 1:cut
group 2: clarity

Options:
stacked - position_stack(the default for geom_bar)
Filled - position_fill
Grouped - position_dodge / dodge2
Small multiples - Use faceting

Bar plots

How to make bar plots look good?

Flip your axes
Order your bar by value
Change the color palette
Make bars thinner

Coordinate systems

cartesian coordinate system: (the default)
coord_cartesian()
Can be modified with:
coord_flip()
coord_fixed()
coord_amp()
coord_trans()

polar coordinate system:
coord_polar()

coord_cartesian()
The default coordinate system

How to zoom in a plot:

Setting the limits of the scale
It eliminates data outside of the plot(warning)

Setting the limits of the coordinate system
This is the proper way to zoom
It doesn't eliminate data outside of plot

Parameters:
xlim, ylim
expand
clip

coord_flip()

Flips cartesian coordinates so that horizontal becomes vertical, and vertical, horizontal.

Useful to draw plots in horizontal mode without habing to change the aesthetic mappings, which might affect other parts of the plot

coord_fixed()
Sets the aspect ratio of the plot.
Ensures that the plot will be kept in that proportion no matter the exporting size.

Set your wanted aspect ratio:

Ratio: aspect ratio, expressed as **y/x**
Default = 1(1 cm along the x axis represents the same range of data as 1 cm along the y axis)

Coordinate systems

coord_map() / coord_quickmap()
Coordinate systems for maps

Quick version:
coord_quickmap()
Best for smaller areas close to equator

Full version:
coord_map()
Computationally costly
Lets you choose the projection (requires package mapproj)

coord_polar()
Apply a polar coordinate system to your plot

Parameters
theta: variable to map angle to (x or y)(string)
direction: 1 (clockwise) or -1 anticlockwise
start: offset of starting point in radians

Don't tell anyone
You can draw pie charts if you apply a polar coordinate system to a bar plot

The Theme system

It controls the appearance of non-data elements of the plot.

It doesn't affect how the data is rendered by geoms, or how it is transformed by scales.

Themes give you control over items like:
The title appearance
The axis labels
The axis tick labels
The strips
The legends and legend key labels
The color of backgrounds

Included in ggplot2
theme_gray(): the default
theme_bw()

theme_linedraw()
theme_light()
theme_dark()
theme_minimal()
theme_classic()
theme_void()

Modifying theme components

To modify an individual theme component:

plot + theme(**element.name = element_function()**)
element name ->
plot elements,Axis elements, Legend element, Panel elements, facet elements

element_function() ->
Text: element_text(), Lines: element_line(), Rectangles: element_rect(), Nothing: element_blank(), Units: e.g. unit(1, “cm”), Margins: margin()

Examples:
plot + theme(axis.title = element_text(color = “red”))
plot + theme(panel.background = element_black())
plot + theme(legend.key = element_rect(fill = “blue”))

element_text()
(font) family
(font) face
(font) colour
(font) size (in points)
hjust [0..1] (0 = left, 1 = right)
vjust [0..1] (0 = bottom, 1 = top)
angle (in degrees)
lineheight (as ratio of fontcase)
margin

- margin(t,r,b,l) # remember trouble

Element functions

element_line()

colour
size
linetype
An integer(0:8), a name(blank, solid, dashed, dotted, datadash, longdash,

twodash)
lineend
(round, butt, square)
arrow
Arrow specification:
?arrow

---

facet_wrap()

Wraps a 1d sequence of panels into 2d.

Useful parameters

facets
vars(a,b) or classical form: a ~ b
First is rows, second is columns

nrow
Number of rows(ncol is automatically calculated)

ncol
Number of cols(nrow is automatically calculated)

dir
Direction of the sequence of panels:
“h” : horizontal
“v” : vertical

scales
Should all panels share the same scale?
yes : “fixed” (the default)
No :
“free_y” : y - scale is independent
“free_x” : x - scale is independent
“free” : x and y scales are independent

strip.position
“top”,”bottom”,”left”,”right”
If we want to rotate, use themes

facet_grid()

Similar to facet_wrap but here it creates a matrix of all possible combinations of variables

Useful parameters
cols & rows the variables we want on the facets
Either,
cols = vars(a), rows = vars(b)
Or the classical form : a ~ b

scales: we don't have the same freedom as before:
"free" doesn't set all panels free, all panels in a column must
have the same x scale, and all panels in a rwo must have the same y scale.

2023-05-28 일

Example

Faceted plot of the Iris dataset

Bonus trick
How to include all data in all panels but highlight only a group?

- 위와 같이 하려면, facet_wrap으로 나누는 기준을 없앤 dataset을 만들어서 그 데이터 셋으로 ggplot을 만든다.

```
ggplot(iris, aes(x = Sepal.Length, y = Petal.Length)) +
  geom_point(aes(color = Species)) +
  facet_wrap(vars(Species)) +
  scale_color_viridis_d() +
  theme(strip.background = element_rect(fill = "gray40", color = "NA"))

isis2 <- iris[-5]

ggplot(iris, aes(x = Sepal.Length, y = Petal.Length)) +
  geom_point(data = isis2, color="gray") +
  geom_point(aes(color = Species)) +
  facet_wrap(vars(Species)) +
  scale_color_viridis_d() +
  theme(strip.background = element_rect(fill = "gray40", color = "NA"),
        strip.text = element_text(color = "white"))
```

2. matrix를 만들 때, 행이름과 열이름을 설정해주고 싶으면,
matrix안에 dimnames를 이용하여 행이름과 열이름을 부여 한다.
dimnames = list(행이름리스트, 열이름리스트)를 matrix안에 넣는다.

```
> myrows <- c("A","B","C","D")
> mycols <- c("col1","col2","col3","col4")
```

```
> x <- matrix(1:16, nrow = 4, ncol = 4, dimnames = list(myrows, mycols))
> x
  col1 col2 col3 col4
A    1    5    9   13
B    2    6   10   14
C    3    7   11   15
D    4    8   12   16
```

3. 행이름이나 열이름 중 특정행이나 열이름을 변경하고 싶으면,
rownames(행렬)[위치]<-"변경행이름"
위와 같이 변화시킨다.

```
> colnames(x)
[1] "col1" "col2" "col3" "col4"
> rownames(x)
[1] "A" "B" "C" "D"
> rownames(x)[4] <- "zz"
> rownames(x)
[1] "A"  "B"  "C"  "zz"
```

4. cbind로 데이터를 합칠 때,
cbind로 새롭게 합치는 데이터의 컬럼이름을 부여하려면,
"새로운열이름" = c(데이터1, 데이터2,..,) 이런식으로 데이터를 합쳐라

```
> x
   col1 col2 col3 col4
A     1    5    9   13
B     2    6   10   14
C     3    7   11   15
zz    4    8   12   16
> cbind(x, c(11,22,33,44))
   col1 col2 col3 col4
A     1    5    9   13 11
B     2    6   10   14 22
C     3    7   11   15 33
zz    4    8   12   16 44
> cbind(x,"col5" = c(11,22,33,44))
   col1 col2 col3 col4 col5
A     1    5    9   13   11
B     2    6   10   14   22
C     3    7   11   15   33
zz    4    8   12   16   44
```

5. 행렬의 열과 행의 크기를 변화시킬 때,
dim이라는 함수와 벡터를 이용해라

```
> x
   col1 col2 col3 col4
A     1    5    9   13
B     2    6   10   14
```

```
C     3    7   11   15
zz    4    8   12   16
> dim(x)
[1] 4 4
> dim(x) <- c(2,8)
> x
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    1    3    5    7    9   11   13   15
[2,]    2    4    6    8   10   12   14   16
```

6. Arrays

These are multidimensional objects, They can store data in more than two dimensions

Creation:

array()

If we create an array of dimension (3,3,2) then it creates 2 rectangular matrices each with 3 rows and 3 columns.

Accessing:

[row, col, matrix]

7. array의 이름 붙이기, 이름 정하기

dimnames(행렬) <- list(행이름벡터, 열이름벡터, 차원이름벡터)

dinnames를 할 때, 이름을 붙여줄때는 벡터가 아니라 list가 와야함

```
> R <- c("r1","r2","r3")
> C <- c("c1","c2","c3")
> M <- c("m1","m2")

dimnames(a)
[[1]]
[1] "r1" "r2" "r3"

[[2]]
[1] "c1" "c2" "c3"

[[3]]
NULL

> dimnames(a) <- list(R,C,M)
> dimnames(a)
[[1]]
[1] "r1" "r2" "r3"

[[2]]
[1] "c1" "c2" "c3"

[[3]]
```

```
[1] "m1" "m2"

> a
, , m1

   c1 c2 c3
r1  1  4  7
r2  2  5  8
r3  3  6  9

, , m2

   c1 c2 c3
r1 10 13 16
r2 11 14 17
r3 12 15 18
```

8.

```
head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
> c <- cars
> head(c)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
> c[c$speed == 7,]
  speed dist
3     7    4
4     7   22
> c[c$speed==7,]$speed
[1] 7 7
> c[c$speed == 7, ]$speed <- 100
> head(c)
  speed dist
1     4    2
2     4   10
3   100    4
4   100   22
5     8   16
6     9   10
```

```
> c[1,1] <- 100
> c
   speed dist
1    100    2
2      4   10
3    100    4
4    100   22
5      8   16
6      9   10
```

2023-05-29 월

Reading data
read.table(): read file in table format and create a data.frame from it

- **file:** filename or file connection [required]
- **header :** boolean indicating if there's a headerline. Default = FALSE
- **sep :** string indicating separator. Default = " "(sep = ",", sep = "\t"(탭으로 나누어져 있을때 사용))
- **stringsAsFactors:** Whether your character variables should be encoded as Factors. Default = FALSE for v4.0.0 onwards. TRUE before that.
- **comment.char:** character string indicating the comment character. Default = "#"
- **nrows :** number of rows to be read from file. Default = Read all rows
- **skip :** number of lines to be skipped at the beginning. Default = 0

Transforming data

Easy data manipulation with dplyr

Select rows according to column values with **filter()**
Pick variables by their names with **select().**
Reorder the rows with **arrange().**
Create new variables with functions of existing variables with **mutate().**

All functions follow the same usage:
First argument: data.frame of interest
Following arguments: what todo with the data.frame
Result: a new data.frame

```
df2 <- filter(df1, what)
```

Transforming data > filter()

Subset rows using column values

- The ‘filter()’ function is used to subset a data frame, retaining all rows that satisfy your conditions.
- To be retained, the row must produce a value of ‘TRUE’ for all conditions.
- Note that when a condition evaluates to ‘NA’ the row will be dropped, unlike vase subsetting with ‘[‘.

Transforming data > select()

Pick variables by their names

Select (and optionally rename) variables in a data frame, using an easy language

Useful function
- start_with(“abc”)
- ends_with(“xyz”)
- contains(“ijk”)
- everything()

Related to select():
Rename the “name” of a column
rename(df, “new_name” = “old_name”)
Rename names according to function
rename_with(df, toupper)

Often used together with filter()

Transforming data > arrange()

Order the rows of a data frame by the values of selected columns

Order rows by values of a column or columns (low to high)

Use desc() to specify descending order

Transforming data > mutate()

Add new columns that are functions of existing columns

Add a new column as a result of operations
between columns

Keep option (default = ALL)

- .keep = “all”
- .keep = “none” # same as transmute()
- .keep = “used”
- .keep = “unused”

Utilities > Missing values

How to deal with missing values

Missing values

- **NA**: Not available
- **NaN**: Not a number
- A NaN value is also NA but the converse is not true.

Check if something is a missing value:

- is.na()
- is.nan()

Deal with missing values:

- complete.cases() : returns a logical vector
  indicating which cases are complete,
  i.e., have no missing values.
- na.omit():returns the object with the incomplete cases removed

More sophisticated treatment:

- function drop_na() from package “tidyr”

Utilities > seq

Easily generate regular sequences

seq(from, to, by, length.out, along.with)

from:starting value of sequence
(DEFAULT = 1)
to : Final value of sequence
(DEFAULT = 1)
by : increment
(DEFAULT = ((to - from)/(length.out - 1)))

length.out : desired length of the sequence
(DEFAULT = NULL)
along.with : take the length from the length of this argument
(DEFAULT = NULL)

SIster functions:
seq_along

seq_len

Utilites > Other ways of generating wequences

rep() : replicate values

```
rep(x, [times, length.out, each])
```

paste() : Concatenate vectors after converting to character.
Although this is not a sequence generator, it might be used to do so

```
paste(..., sep = " ", collapse = NULL, recycle0 = FALSE)
```

sample() : generate sequence of random values

```
> paste0("Col", 1:10)
 [1] "Col1"  "Col2"  "Col3"  "Col4"
 [5] "Col5"  "Col6"  "Col7"  "Col8"
 [9] "Col9"  "Col10"
> paste("Col", 1:10)
 [1] "Col 1"  "Col 2"  "Col 3"  "Col 4"
 [5] "Col 5"  "Col 6"  "Col 7"  "Col 8"
 [9] "Col 9"  "Col 10"
> paste("Col",1:10, sep = ".")
 [1] "Col.1"  "Col.2"  "Col.3"  "Col.4"
 [5] "Col.5"  "Col.6"  "Col.7"  "Col.8"
 [9] "Col.9"  "Col.10"
```

1. 컬럼순서를 변경하고 싶을 때(select 를 이용하여),

```
select(df, Active, everything())
# 위에처럼 하면,  가장 먼저 보고 싶은 컬럼, everything()을 이용하여 순서를 변경할 수 있음
```

2. 컬럼명을 변경하고 싶을 때,
   1) subset을 이용하여 데이터 컬럼명을 변경함.
   names(df)[3] <- “new_name”

   2) rename을 이용하여 데이터 컬럼명을 변경
   - rename(데이터프레임, “새로운이름” = “바꿀 이름”)
   - rename_with() : 함수를 이용하여 이름을 변경 -> rename_with(df, toupper) : 이 경우 컬럼명을 다 대문자로 변경
     rename_with() 는
     start_with(“abc”)
     ends_with(“xyz”)
     contains(“ijk”)
     everything() 와 같은 함수와 결합

```
rename_with(df, toupper, starts_with("A"))
```

```
# 1
select(filter(df, Active == TRUE), Age)

# 2
filter(df, Active == TRUE) %>%  select(Age)
# 위의 두개의 코드는 같은 내용임
# select는 종종 filter와 같이 결합함
```

```
complete.cases(airquality)
# complete.cases(airquality) 는 boolean type으로 값을 반환

# 1
airquality[complete.cases(airquality),]
# 위와 같이 하면 Na 값을 제외한 행을 반환해줌

# 2
na.omit(airquality)
# 1과 2의 코드는 같은 값을 반환해 줌
```

```
> sample(1:5, 1)
[1] 4
> sample(1:5, 1)
[1] 2
> sample(1:5, 5)
[1] 2 4 1 5 3
> sample(1:5, 5, replace = T)
[1] 5 4 3 5 2
> sample(1:5, 10, replace = T)
 [1] 1 4 1 1 2 1 4 3 2 5
> paste0("user_", sample(1:1000, 10))
 [1] "user_101" "user_881" "user_58"
 [4] "user_846" "user_167" "user_23"
 [7] "user_906" "user_858" "user_44"
[10] "user_445"
```

Utilities > Dates

Handling Dates and format conversion

Dates are stored using the Date class

- Internally, date are stored as the number of days since 1970-01-01 and times are stored in number

of seconds from 1970-01-01.
- Don't confuse it with date(), which returns today's date and time, but in character format

Converting a string to a Date:
- as.Date(" ") stores the date specified in the string in the date format.

strptime
Date-Time Conversion Functions To And From Character

- %d = day in number format
- %m = month in number format
- %y = year without century
- %Y = Year with century
- %a = Abbreviated weekday name
- %A = Full weekday name (in current locale)
- %b = Abbreviated month name
- %B = Full month name
- … there are many more

With Dates, you might want to:
- Change their printing format with **format()**
- Operate with them
- Create sequences out of them with **seq()**

More sophisticated treatment:
- package "lubridate"

```
> today <- as.Date("2021-01-18")
> today
[1] "2021-01-18"
> today_print <- format(today, "%A%B%Y")
> today_print
[1] "월요일1월2021"
> class(today_print)
[1] "character"
> today
[1] "2021-01-18"
```

```
> today <- as.Date("2021-01-18")
> today
[1] "2021-01-18"
> today + 1
[1] "2021-01-19"
> today - 1
[1] "2021-01-17"
> dborn <- as.Date("1980-01-01")
> today-dborn
```

```
Time difference of 14993 days
> as.numeric(today-dborn)
[1] 14993
> another_day <- as.Date("2020-12-01")
> another_day
[1] "2020-12-01"
> another_day - today
Time difference of -48 days
> seq(another_day, today, by = 1)
 [1] "2020-12-01" "2020-12-02"
 [3] "2020-12-03" "2020-12-04"
 [5] "2020-12-05" "2020-12-06"
 [7] "2020-12-07" "2020-12-08"
 [9] "2020-12-09" "2020-12-10"
[11] "2020-12-11" "2020-12-12"
[13] "2020-12-13" "2020-12-14"
[15] "2020-12-15" "2020-12-16"
[17] "2020-12-17" "2020-12-18"
[19] "2020-12-19" "2020-12-20"
[21] "2020-12-21" "2020-12-22"
[23] "2020-12-23" "2020-12-24"
[25] "2020-12-25" "2020-12-26"
[27] "2020-12-27" "2020-12-28"
[29] "2020-12-29" "2020-12-30"
[31] "2020-12-31" "2021-01-01"
[33] "2021-01-02" "2021-01-03"
[35] "2021-01-04" "2021-01-05"
[37] "2021-01-06" "2021-01-07"
[39] "2021-01-08" "2021-01-09"
[41] "2021-01-10" "2021-01-11"
[43] "2021-01-12" "2021-01-13"
[45] "2021-01-14" "2021-01-15"
[47] "2021-01-16" "2021-01-17"
[49] "2021-01-18"
> seq(another_day, today, by = 4)
 [1] "2020-12-01" "2020-12-05"
 [3] "2020-12-09" "2020-12-13"
 [5] "2020-12-17" "2020-12-21"
 [7] "2020-12-25" "2020-12-29"
 [9] "2021-01-02" "2021-01-06"
[11] "2021-01-10" "2021-01-14"
[13] "2021-01-18"
```

2023-05-30 화

1. c(“ ”, “ ”, “ ”, … , “ ”) 의 class와 typeof는 character 인 것을 명심.
   따라서 scale_x_discrete, scale_y_discrete 의 breaks = 의 값은 character 형식으로

들어가야함.

```
y_final <- seq(from =
min(as.numeric(final_03$ta)),to=max(as.numeric(final_03$ta)), by = 5)

ggplot() +
  geom_line(data = final_03, aes(x = m_d_time, y = ta, group = year_01, colour =
year_01))+
  # 일출
  geom_vline(xintercept = format(as.POSIXct("03-21 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-22 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-23 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-24 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-25 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-26 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-27 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-28 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-29 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-30 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  geom_vline(xintercept = format(as.POSIXct("03-31 07:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40", linetype = "dashed")+
  # 일몰
  geom_vline(xintercept = format(as.POSIXct("03-21 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  geom_vline(xintercept = format(as.POSIXct("03-22 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  geom_vline(xintercept = format(as.POSIXct("03-23 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  geom_vline(xintercept = format(as.POSIXct("03-24 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  geom_vline(xintercept = format(as.POSIXct("03-25 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  geom_vline(xintercept = format(as.POSIXct("03-26 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  geom_vline(xintercept = format(as.POSIXct("03-27 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  geom_vline(xintercept = format(as.POSIXct("03-28 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  geom_vline(xintercept = format(as.POSIXct("03-29 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  geom_vline(xintercept = format(as.POSIXct("03-30 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
```

```
  geom_vline(xintercept = format(as.POSIXct("03-31 19:00",format = "%m-%d
%H:%M"), "%m-%d %H:%M"), color = "gray40" )+
  theme_minimal()+
  scale_x_discrete(name = "DATE", breaks = c("03-21 12:00","03-22 12:00","03-23
12:00","03-24 12:00","03-25 12:00","03-26 12:00","03-27 12:00","03-28
12:00","03-29 12:00","03-30 12:00","03-31 12:00"))+
  scale_y_discrete(name = "TEMPERATURE", breaks = as.character(y_final))
```

2023-06-01 목

1. **10**개년 이상의 일자별 온도 데이터를 이용하여 평년 온도를 구할때**,** 굳이 이상치를 제거할 필요가 없음
   회귀선을 그릴 때, 데이터가 300~350개 이상이 넘어가면 일반화할 수 있다고 함
   2010년부터 2022년까지 철원 3월 온도 데이터를 가지고 회귀모델을 만들었으며, 이때 2011년의 온도가 낮았고, 2018년의 온도가 높았음.
   (각각 이상치의 비율이 오차값을 4로 기준으로 하였을 때, 35.48%, 32.25%으로 나왔음. 연평균 9.67%, 12.15%으로 큰 수치라고 판단)
   따라서 위의 두 년도의 데이터를 제거하고 다시 회귀모델을 이용하여, 데이터를 구하였음
   각각의 연평균 값이 상한 7.9%, 하한 10.6%으로 나왔으며, 기존 방식(전체 온도 데이터를 가지고 구했던 방식)으로 구했던 데이터와 값의 차가 각각 상한 1.8%, 하한 1.6%으로 나왔음

오차값	outrate_up2	outrate_down2	outrate_up2.5	outrate_down2.5	outrate_up3	outrate_down3.5	outrate_up3.5	outrate_up4	outrate_down4	outrate_up4.5	outrate_down4.5
이상치 비율	27.6	26.1	21.7	22	15.8	12.3	10.6	7.9	10.6	6.7	7

2023-06-05 월

1. **[API code]**
   이전에 제대로 돌아간 코드가 새로 실행하니까 오류를 일으키는 경우 업데이트를 해서 함수의 기능과 역할이 달라졌을 수 있음.
   print를 찍어서 일일이 내용을 확인해봄
   reqURL 03 DB에러의 경우, URL 내용을 꼼꼼히 살펴볼 필요가 있음
   (<코드 01>을 꼼꼼히 살펴볼 것)

2. **[DB table]**
   DB table (DB 테이블 **)**에 내용이 안 적어질 경우,
   1) 테이블이 있는지 확인 :
   dbWriteTable의 경우, append = T로 하면 데이터 테이블이 안 적어질 수 있음

   2) dbWriteTable을 실행하기 전,
   dbExecute(conn, 'set names "utf8"') 을 실행해라
   (<코드 01> 참고)

3. **[Error: external pointer is not valid]**
   DB 관련 에러 일 수 있음
   dbConnect를 정의한 변수와 불러오는 값들이 달라서 생김(dbExecute, dbWriteTable)

```
# asos_dttm_months----
library(XML)
library(jsonlite)
library(rjson)
library(tidyverse)
# library(clipr)
library(RMariaDB)
library(lubridate)



# parsing function() ----
parsing <- function(requrl){

  response <- xmlTreeParse(requrl, useInternalNodes = T, encoding =
"UTF-8")
  rn <- xmlRoot(response)

  weather <<- xmlToDataFrame(nodes = getNodeSet(rn,"//item"))

  return(weather)
}



# chulwon specific date load----
req_api_date <- function(start_date,start_hour,end_date,end_hour,stnid){

  start_date <- "20151111"
  start_hour <- "04"
  end_date <- "20151111"
  end_hour <- "23"
```

```
  stnid <- "95"


  if(as.Date(start_date, "%Y%m%d") > as.Date(end_date,"%Y%m%d")){
    stop("end_date에 오류 발생")
  }

  req <-
"http://apis.data.go.kr/1360000/AsosHourlyInfoService/getWthrDataList"
  mykey <-
"JDE0tEOvxpZeDfXMaz7OgzduY%2BMMG6kv24bobeSQ4E796aY0hU%2Bh%2BB5t9AT30aB0i
i%2B5favN7QB%2FIIWLA2PS2w%3D%3D"
  params <-
c("serviceKey","numofRows","pageNo","dataCd","dateCd","startDt","startHh
",
              "endDt","endHh","stnIds")

  # check time!!!
  values <-
c(mykey,10,1,"ASOS","HR",start_date,start_hour,end_date,end_hour,stnid)

  reqURL <- paste(req,
                  paste(params,values,sep="=",collapse = "&"),
                  sep = "?")
  print(reqURL)


  response <- xmlTreeParse(reqURL, useInternalNodes = T, encoding =
"UTF-8")
  print(response)

  rn <- xmlRoot(response)
  print(rn)


  totalRows <- xpathApply(rn,"//totalCount", xmlValue)
  print(totalRows)

  if(totalRows[[1]] != 0){
    weather_list <- list()

    for (j in 1:ceiling(as.numeric(totalRows)/10)){ # j <- 1

      # for (j in 1:10){

      values <-
```

```
c(mykey,10,j,"ASOS","HR",start_date,start_hour,end_date,end_hour,stnid)

      requrl <- paste(req,
                      paste(params, values, sep = "=", collapse = "&"),
                      sep = "?")

      # 에러처리 실패(1) ---> 성공! parsing의 weather 값을 전역변수로
설정해 주었음
      tryCatch(expr =
                 tryCatch(expr = {
                   message("try again")
                   message("Error!!"," date :",start_date, "~",
end_date,"\nstnid ;", stnid, "\npage :",j)
                   parsing(requrl)},

                   error = function(e1){
                     message("try again")
                     message("Error!!"," date :",start_date, "~",
end_date,"\nstnid ;", stnid, "\npage :",j)
                     parsing(requrl)
                     return(invisible())
                   },

                   warning = function(w1){
                     message("try again")
                     message("Warning!!"," date :",start_date, "~",
end_date,"\nstnid ;", stnid, "\npage :",j)
                     parsing(requrl)
                     return(invisible())
                   }),


               error = function(e){
                 tryCatch(expr = {
                   message("try again")
                   message("Error!!"," date :",start_date, "~",
end_date,"\nstnid ;", stnid, "\npage :",j)
                   parsing(requrl)},

                   error = function(e2){
                     message("try again")
                     message("Error!!"," date :",start_date, "~",
end_date,"\nstnid ;", stnid, "\npage :",j)
                     parsing(requrl)
                     return(invisible())
                   },
```

```
                  warning = function(w2){
                    message("try again")
                    message("Warning!!"," date :",start_date, "~",
end_date,"\nstnid ;", stnid, "\npage :",j)
                    parsing(requrl)
                    return(invisible())
                  })},


               warning = function(w){
                 conn0 <- dbConnect(dvr = MariaDB(),
                                    uwername = 'jinong',
                                    password = 'jinong#0801!',
                                    host = '211.237.5.46',
                                    port = 3306,
                                    dbname = 'jn_cmm')

                 missing_data <- data.frame(
                   stnid0 = stnid,
                   start_date0 = start_date,
                   start_time0 = start_hour,
                   end_date0 = end_date,
                   end_time0 = end_hour,
                   page_num0 = j,
                   URL = requrl)

                 dbExecute(conn0, 'set names "utf8"')

                 dbWriteTable(conn0, name =
"tb_weather_asos_dttm_missing",
                              value = missing_data, append = T,
                              overwrite = F)

                 dbDisconnect(conn0)

               })

      print(requrl)

      if (nrow(weather) == 0){
        print(paste("there is no data page", j,"stnid ;", stnid, "date
:",start_date, "~", end_date))
        next
      }
```

```
      weather_list[[j]] <- weather
    }

    all_weather <- do.call(rbind, weather_list)

    conn <- dbConnect(drv=MariaDB(),
                      username = 'jinong',
                      password = 'jinong#0801!',
                      host = '211.237.5.46',
                      port = 3306,
                      dbname = 'jn_cmm')


    # conn <- dbConnect(drv = MariaDB(),
    #                   username = 'root',
    #                   password = 'dusqhd1djr!',
    #                   host = 'localhost',
    #                   port = 3306,
    #                   dbname = 'chloe')

    dbExecute(conn, 'set names "utf8"')

    dbWriteTable(conn, name = "tb_weather_asos_dttm", value =
all_weather, append = T)

    dbDisconnect(conn)

    print("Done")
  } else{
    print(paste("Empty date:",start_date," ~ ",end_date, "stnid
:",stnid))



      empty_data <- data.frame(
      stnid0 = stnid,
      start_date0 = start_date,
      start_time0 = start_hour,
      end_date0 = end_date,
      end_time0 = end_hour,
      URL = reqURL)


    conn1 <- dbConnect(drv=MariaDB(),
                      username = 'jinong',
                      password = 'jinong#0801!',
```

```
                    host = '211.237.5.46',
                    port = 3306,
                    dbname = 'jn_cmm')

    dbExecute(conn1, 'set names "utf8"')

    dbWriteTable(conn1, name = "tb_weather_asos_dttm_empty",
                 value = empty_data, append = T,
                 overwrite = F)

    dbDisconnect(conn1)

  }

}
```

4. **[**컴퓨터 자동꺼짐짐**]**
   CPU 온도가 너무 높으면 자동으로 꺼질 수 있다**.**

   windows 긴급 보안 업데이트면 컴퓨터 업데이트를 다 꺼놓아도 자동으로 꺼질 수 있다**.**

5. **[SQL]**
   SQL 구문의 작성이 완료되면 JAVA와 마찬가지로 ;(세미콜론)을 작성하여 구문이 끝났음을 알려야 한다.

2023-06-06 화

1. **[R, ggplot] : ggplot**에서 두개의 **group**을 사용할 때 **interaction**을 사용함

```
ggplot(melted_data, aes(x = Year, y = value, group =
interaction(variable, Town), color = Town) +
geom_point() + geom_line()
```

2. **[R, ggplot] : ggplot**에서 특정타입에 따라 **linetype**을 다르게 하고 싶을 때

```
ggplot(melted_data, aes(x = Year, y = value, group =
interaction(variable, Town), color = Town, linetype = variable)) +
geom_line() + geom_point()
```

다른표현식

```
ggplot(data, aes(x = Year, y = Town)) +
geom_line(aes(y = Males)) + geom_line(aes(y = Females), linetype =
```

```
"dashed") +
geom_point(aes(y = Males)) + geom_point(aes(y = Females))
```

3. **[R, ggplot] : ggplot**에서 **mapping**을 할 때, **color** 속성을 **ggplot**에 적용하면, **geom_point()** 와 같은 다른 속성에 적용되지 않는다.

```
ggplot(data, aes(x = Year, y = value), color = "darkblue") +
geom_point(size = 3)
```

4. **[R, ggplot] : widedata**를 **longdata**로 변경하고 싶을 때,
   package **reshape2** 의 melt를 사용한다.
   (개인적으로 따로 무언가를 설치할 필요가 없는 spread나 gather를 추천_ 위와 같은 방식이 있다는 것을 알려주고자 얘기함)

2023-06-07 수

[R, ggplot2]

Scatterplots

```
geom_point(
mapping = NULL,
data = NULL,
stat = “identity”,
position = “identity”,
…,
na.rm = FALSE,
show.legend= NA,
inherit.aes = TRUE
)
```

Colors (Continuous Data)

- Define our own gradient
  scale_color_gradient /

```
ggplot(humans, aes(x = mass, y = height)) +
  geom_point(aes(color = BMI), size = 4) +
  scale_color_gradient(low = "green", high = "red")
```

scale_color_gradient2

```
ggplot(humans, aes(x = mass, y = height)) +
  geom_point(aes(color = BMI), size = 4) +
```

```
  scale_color_gradient2(low = "green",mid = 'yellow', midpoint = 22,
high = 'red')
```

- Use a predefined gradient scale
  scale_color_gradientn
  ex) rainbow, heat.colors, terrain.colors, topo.colors, cm.colors

```
ggplot(humans, aes(x = mass, y = height)) +
  geom_point(aes(color = BMI), size = 4) +
  scale_color_gradientn(colors = heat.colors(3))
```

<순서를 반대로 변경하고 싶을 때>

```
ggplot(humans, aes(x = mass, y = height)) +
  geom_point(aes(color = BMI), size = 4) +
  scale_color_gradientn(colors = rev(heat.colors(3)))
```

- Use our own palette

```
mypalette <-
c("royalblue","forestgreen","gold","darkorange",'firebrick')
ggplot(humans, aes(x = mass, y = height)) +
  geom_point(aes(color = BMI), size = 4) +
  scale_color_gradientn(colors = mypalette)
```

*For "fill", use: scale_fill_gradient**

Colors (Discrete Data)

- Map our color variable onto something discrete
  e.g. BMI ranges

- Use: scale_color_manual

```
humans$BMI_categories <- cut(humans$BMI, breaks =
c(0,18.5,25,30,35,Inf),
                             labels =
c("Underweight","Normal","Overweight","Obese","Extremely Obese"),
                             right = FALSE)

mypalette <-
c("royalblue","forestgreen","gold","darkorange",'firebrick')
```

```
ggplot(humans, aes(x = mass, y = height)) +
  geom_poin(aes(color = BMI_categories), size = 4) +
  scale_color_manual(values = mypalette)
```

```
ggplot(humans, aes(x = mass, y = height)) +
  geom_point(aes(color = BMI_categories), size = 10, alpha = 0.4) +
  scale_color_manual(values = mypalette, name = "BMI") +
geom_text(aes(label = name))
```

```
# geom_text 가 예쁘지 않을 때, ggrepel을 사용한다.
library(ggrepel)
ggplot(humans, aes(x = mass, y = height)) +
  geom_point(aes(color = BMI_categories), size = 10, alpha = 0.4) +
  scale_color_manual(values = mypalette, name = "BMI") +
  geom_text_repel(aes(label = name))
```

1. **[R] :** 연속적인 데이터를 불연속적인 데이터로 만들기
   cut 함수를 이용.
   cut(data, breaks = c(자르는 기준), label = c(자른 데이터 이름))
   right 역할을 뭔지 모르겠음…

```
humans$BMI_categories <- cut(humans$BMI, breaks =
c(0,18.5,25,30,35,Inf),
                             labels =
c("Underweight","Normal","Overweight","Obese","Extremely Obese"),
                             right = FALSE)
```

2023-06-07 수

1. **[R, ggplot] : ggplot**이 안 그려지는 경우
   빈 플롯이 나타날 경우, 데이터가 없거나 부적절한 데이터 값이 원인
   x축을 lubridate의 package로 사용하여 class가 period인 경우, 빈 플롯이 그려짐

   ggplot에서 x축을 날짜 데이터를 사용할 때는 as.numeric()이나 as.POSIXct()를 사용하여 x 축의 데이터 형식을 바꿔줌

2023-06-08 목

1. geom_density는 선을 그리고 나머지 영역을 채우는 함수이다.
   geom_area와 비슷해 보이나 전혀 그렇지 않다.

   geom_density는 y 값이 없어도 알아서 그래프를 그림
   geom_area는 y 값이 있어야만 그래프를 그림

```
ggplot(diamonds, aes(x = depth, color = cut, fill = cut)) +
geom_density(alpha = 0.3)

ggplot(diamonds, aes(x = depth)) + geom_density()

ggplot(diamonds, aes(x = depth)) + geom_line(stat = "density")
```

2023-06-09 금

1. 문제가 생겼을 때, 한번에 여러개 고치지 말고, 하나씩 실행해보면서 결과 남기기 시키는 일만 열심히 하기....

2. data를 읽어오는 유형에 따라서 데이터 유형?이 바뀐다.
   ex) read.csv -> data.frame 형태
   read_excel -> tibble 형태

   뿐만 아니라 data를 읽어오는 유형에 따라서 컬럼명이 변화할 수 있다.
   csv 같은 경우 () 괄호를 인식하지 못하여, .(온점)으로 표시
   위와 같은 경우 do.call(rbind, list)를 할 때 제대로 인식하지 못 하는 경우가 발생할 수 있다.
   위의 같은 경우를 사용하기 위해서는 do.call 컬럼명이 같아야 함.

3. excel에서 확장자를 변경할 때, 데이터의 오류를 일으킬 수 있음.
   .xlsx -> .csv -> .xlsx로 변경하면 데이터 값(내용)에 변화가 생김(즉, 오류 발생!!)

2023-06-10 토

1. **[R] :** 화면에 출력할 때,
   print() : 어떤 문장만 출력함
   cat() : 변수와 문장을 같이 출력할 때 사용

2. **[R] :** 변수 할당시 **L**을 입력하는 이유
   integer라고 바로 알려주기 위해서 이다

```
a <- 1L
typeof(a)
>#  integer
```

3. **[R] : c() -> concat** 의 약자이며 이렇게 했을 때,
   typeof 는 벡터이다.
   벡터는 무조건 다 같은 형식으로 데이터를 만든다.

```
> v <- c(1,"hi")
> v
[1] "1"  "hi"
```

4. **[R] :** 빈 벡터를 만드는 법,

```
> x <- vector("numeric", length = 5)
> x
[1] 0 0 0 0 0
```

5. **[R] :** 특정 벡터에 접근하는 방법,

```
> v <- c("A","B","C")
> v[1]
[1] "A"
> v[1:2]
[1] "A" "B"
> v[c(T,T,F)]
[1] "A" "B"
```

   Boolean 타입으로 데이터를 특정 벡터를 출력할 수 있음

```
> v1 <- 1:3
> v1
[1] 1 2 3
> v2 <- c(1,2,3)
> class(v2)
[1] "numeric"
> identical(v1,v2)
[1] FALSE
> v2 <- c(1L,2L,3L)
> identical(v1,v2)
[1] TRUE
```

6. [R] : vector들을 concat 할 때(붙일 때),
   c()를 이용하여 붙임

```
> a <- 1:3
> b <- 4:6
> z <- c(a,b)
> z
[1] 1 2 3 4 5 6
```

2023-06-11 일

1. list는 여러가지 데이터 형식을 저장
   list안에 list를 넣을 수 있음

2. list안에 구성요소에 접근할 때,
   변수명으로 따로 값을 저장하여 접근하려면 [[]]밖에 안된다

```
xx <- "Retired"
l[[xx]] # 가능
l$xx # 불가능능
```

3. list의 특정 위치의 이름을 바꾼다고 하면,
   names(list)[6] 이런 식으로 접근할 수 있음

4. list를 append 할 때,
   list$새컬럼이름 <- 내용
   이런 식으로 접근할 수 있음

5. matrix
   matrix는 vector다. dimension만 제외하면 vector와 똑같음
   matirx 생성은 matrix() 이렇게 하고 생성할 수 있음
   dimnames = 와 list 를 이용하여 row 이름과 col 이름을 만들 수 있음

```
> x <- matrix(1:12, nrow = 4, ncol = 3)
> x
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
>
> x <- matrix("Hello", nrow = 4, ncol = 4)
> x
     [,1]    [,2]    [,3]    [,4]
[1,] "Hello" "Hello" "Hello" "Hello"
[2,] "Hello" "Hello" "Hello" "Hello"
[3,] "Hello" "Hello" "Hello" "Hello"
[4,] "Hello" "Hello" "Hello" "Hello"
>
> myrows <- c("A","B","C","D")
> mycols <- c("col1","col2","col3","col4")
> x <- matrix(1:16, nrow = 4, ncol = 4, dimnames = list(myrows, mycols))
> x
  col1 col2 col3 col4
A    1    5    9   13
B    2    6   10   14
C    3    7   11   15
D    4    8   12   16
```

6. matrix의 row이름과 col이름에 접근하는 방법,

rownames(x), colnames(x)에 접근함
이것들을 수정하고 싶을 때는
rownames(x)[4] 이런 식으로 특정위치를 하여 접근
colnames(x)[3] <- "새로운 이름" 이런 식으로 할 수 있음

7. matrix의 행과 열 변환시키는 방법
행과 열을 변환시키면 기존에 가지고 있던 colnames와 rownames의 값은 잃는다.

```
> dim(x)
[1] 4 4
> dim(x) <- c(2,8)
> x
```

8. array : multidimensional objects.

```
> a <- array(1:18, dim=c(3,3,2))
> a
, , 1

     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

, , 2

     [,1] [,2] [,3]
[1,]   10   13   16
[2,]   11   14   17
[3,]   12   15   18
```

```
> a[1,1,1]
[1] 1
> a[,,1]
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```
> a[,1,]
     [,1] [,2]
[1,]    1   10
[2,]    2   11
[3,]    3   12
```

10. rownames와 colnames는 있지만, matrixnames는 없으므로
dimnames 라는 이름으로 list(행이름list, 열이름list, 매트릭스이름 list) 이런식으로 함

11. matirx와 data.frame 모두 데이터를 합칠 때,

cbind(), rbind() 로 합쳐서 하면 됨

12. data.frame에서 특정조건의 특정 열의 값을 바꿀 때,

```
> c <- cars
> head(c)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
> c[c$speed == 7,]$speed <- 100
> head(c)
  speed dist
1     4    2
2     4   10
3   100    4
4   100   22
5     8   16
6     9   10
```

13. ggplot은 오직 data.frame만 input 값으로 받는다!!(명심)

14. data.frame에서 factor에 따라 재정렬하는 방법
    - ggplot의 순서는 종종 factor의 순서에 따라 정렬하기 때문
    - 과정:
      1) 먼저 순서를 정의한다.
      2) 정의한 순서를 가지고 data.frame의 level of the factor를 재정렬한다.

```
> opinions <- c("Horrible","Bad","Neutral","Good","Fantastic")
> opinions <- as.factor(opinions)
> opinions
[1] Horrible  Bad       Neutral   Good      Fantastic
Levels: Bad Fantastic Good Horrible Neutral
> count <- c(45,35,22,60,15)
> df <- data.frame(opinions, count)
> df
   opinions count
1  Horrible    45
2       Bad    35
3   Neutral    22
4      Good    60
5 Fantastic    15
> opinions
[1] Horrible  Bad       Neutral   Good      Fantastic
Levels: Bad Fantastic Good Horrible Neutral
> newOrder <- c("Horrible","Bad","Neutral","Good","Fantastic")
> dff <- transform(df, opinions=factor(opinions, levels = newOrder))
```

```
> dff
   opinions count
1  Horrible    45
2       Bad    35
3   Neutral    22
4      Good    60
5 Fantastic    15
```

15. 설치되어 있는 packages 확인방법
installed.packages() 이렇게 하면 확인할 수 있음

16. data.frame에서 특정 컬럼명 변경 방법:

```
rename(df, "Opinion" = "Result")
rename(df, "new_name" = "old_name")
rename_with(df, toupper) # 모든 컬럼이름을 대문자로 변경
rename_with(df, toupper, start_with("a")) # a로 시작하는 모든 컬럼이름을 대문자로 변경함
```

17. [R] : select와 filter를 같이 사용할 수 있음

```
select(filter(df, Active==TRUE), Age)
```

18. mutate() 의 기능 중 특정컬럼 날리기

keep option (default = ALL)

- .keep = "all"
- .keep = "none" # Same as transmute() # 계산 결과의 컬럼만 표시 나머지 삭제
- .keep = "used" # 계산에 사용된 컬럼과 계산 결과의 컬럼 표시
- .keep = "unused" # 계산에 사용되지 않은 컬럼과 계산 결과의 컬럼 표시

```
mutate(df, BMI = Weights / Heights ^ 2, .keep = "all")
mutate(df, BMI = Weights / Heights ^ 2, .keep = "none") # 계산 결과의 컬럼 빼고 모든 컬럼 제외
```

19. [R] : NA 와 NaN
NaN은 NaN와 NA 둘 다 될 수 있지만
NA는 NA만 된다.
NaN -> is.na(), is.nan() 둘 다 TRUE 반환
NA - > is.na() 만 가능함

20. [R] : 결측치 제외하는 방법
complete.cases() 나 na.omit()을 사용
complete.cases() : 값을 반환할 때, logical type(TRUE,FALSE)로 값을 반환
na.omit() : 결측치가 있는 값을 제외하고 data.frame을 반환

```
complete.cases(airquality)
airquality[complete.cases(airquality),] # 1
na.omit(airquality) #2

# 1과 2의 결과가 같음!
```

21. [R] : seq()
along.with : take the length from the length of this argument

```
> v <- c(11,22,33,44,55)
> seq(1,10, along.with = v)
[1]  1.00  3.25  5.50  7.75 10.00
```

22. 편하게 colname을 만드는 방법:
paste와 paste0를 이용하라
R이 알아서 반복한다는 성질을 이용하라

```
> paste0("col",1:10)
 [1] "col1"  "col2"  "col3"  "col4"  "col5"  "col6"  "col7"
 [8] "col8"  "col9"  "col10"
> paste("col",1:10)
 [1] "col 1"  "col 2"  "col 3"  "col 4"  "col 5"  "col 6"
 [7] "col 7"  "col 8"  "col 9"  "col 10"
> paste("col",1:10,sep = ".")
 [1] "col.1"  "col.2"  "col.3"  "col.4"  "col.5"  "col.6"
 [7] "col.7"  "col.8"  "col.9"  "col.10"
```

23. [R] : format으로 print 모양 변경할 수 있음

```
> today <- as.Date("2021-01-18")
> today
[1] "2021-01-18"
> today_print <- format(today, "%A %B %Y")
> today_print
[1] "월요일 1월 2021"
> class(today_print)
[1] "character"
```

24. [R] : merge를 이용해서 데이터를 합칠 때,
같은 컬럼명을 기준으로 데이터를 합치면 by 만 사용하며 합치면 된다

```
merge(x, y, by = "공통의 컬럼명")
```

2023-06-12 월

1. 지상(ASOS) 시간자료 조회서비스에서 tm의 시간부분이 00:00이면
   엑셀에서는 날짜만 나타남 시간이 제외됨
   위 점에 유의함!
   결측치 가져올 때 날짜만 나와있다고 제외하지 말고 다 포함해서 가져올 것
   + 다음부터는 결측치를 꼭 Rstudio 환경에서 view를 사용하여 확인할 것!

2. [R, ggplot] : geom_path()와 geom_polygon()은 밀접한 관련이 있음

3. [R, ggplot] : geom_line(), geom_path()는 비슷하지만
   geom_line() : x축을 기준으로 먼저 연결
   geom_path() : data 순서의 영향을 받음

4. [R, ggplot] : geom_area()는 영역을 채우기 때문에 y축이 0부터 시작함
   but geom_line()은 그렇지 않음

5. [R, ggplot] : colours() 를 이용하면 다양한 색상을 확인할 수 있음

6. [R, ggplot] : 그룹별 다른 색상 설정
   그룹별로 다른 컬러를 사용하려면 해당 geometry에 aes(group = 해당그룹, color = 해당그룹) 을 설정한 후, scale_color_viridis(discrete = T) 라고 설정함.

```
ggplot(data, aes(x = year, y = n))+
  geom_point(aes(group = mygroup, color = mygroup))+
  scale_color_viridis(discrete = T)
```

2023-06-23 금

1. [R, ggsave] : ggsave를 했을 때 확장자를 타이틀에 안 써주었을 때 생기는 에러

```
Error in `ggsave()`:
! `filename` has no file extension and `device` is "NULL".
```

   해결 : filename = “파일이름.확장자” 로 저장

```
  ggsave(filename = paste0(input_year,"년 ",input_month,"월의 주간 평균
회귀값과 최고 온도의 차.jpg"),
         plot = g,
         width = 20, height = 12, units = "cm", dpi = 320 )
```

2023-06-26 월

1. **[R, ggplot, barplot]**
   ggplot 에서 group을 사용해서 나누는 그래프가 아니면 **width =** 를 설정해서 일정한 크기로 만들어 준다.

```
  g <- ggplot(rg_day_2011) +
    geom_bar(data = rg_day_2011 %>% filter(diff_max_rg > 0), aes(day, diff_max_rg),
             stat = "identity", fill = "firebrick", position = position_dodge("single")) +
    geom_bar(data = rg_day_2011 %>% filter(diff_max_rg < 0), aes(day, diff_max_rg),
             stat = "identity", fill = "darkblue", position = position_dodge("single"))+
    geom_text(data = rg_day_2011 %>% filter(diff_max_rg > 0),
              aes(x = day, y = diff_max_rg + 0.5, label = max_ta), size = 2) +
    geom_text(data = rg_day_2011 %>% filter(diff_max_rg < 0),
              aes(x = day, y = diff_max_rg - 0.5, label = max_ta), size = 2) +
    geom_text(aes(x = day, y = 0.1, label = round(tm_regress_day,2)), size = 2, fontface = "bold") +
    labs(title = paste0(input_year,"년 ",input_month,"월의 주간 평균 회귀값과 최고 온도의 차"),
         x = "일",
         y = "온도")
```

```
  g <- ggplot(rg_day_2011) +
    geom_bar(data = rg_day_2011 %>% filter(diff_max_rg > 0), aes(day, diff_max_rg),
             stat = "identity", fill = "firebrick", width = 0.9)+
    geom_bar(data = rg_day_2011 %>% filter(diff_max_rg < 0), aes(day, diff_max_rg),
             stat = "identity", fill = "darkblue", width = 0.9)+
    geom_text(data = rg_day_2011 %>% filter(diff_max_rg > 0),
              aes(x = day, y = diff_max_rg + 0.5, label = max_ta), size = 2) +
    geom_text(data = rg_day_2011 %>% filter(diff_max_rg < 0),
              aes(x = day, y = diff_max_rg - 0.5, label = max_ta), size = 2) +
    geom_text(aes(x = day, y = 0.1, label = round(tm_regress_day,2)), size = 2, fontface = "bold") +
    labs(title = paste0(input_year,"년 ",input_month,"월의 주간 평균 회귀값과 최고 온도의 차"),
```

```
        x = "일",
        y = "온도")
```

2.