04. R 데이터 시각화와 ggplot2 패키지

4주차

성현곤

충북대학교 도시공학과
Dept.of URBAN ENGINEERING

목차

- 시각화와 그래프 유형
  - 시각화의 정의와 역사
  - 그래프 형태와 선택
- R로 그래프 작성하기 기초
  - R 그래프의 par(), 색상 pallet 이해하기
  - R 시각화 내장함수 종류
- 내장함수로 그래프 작성하기
  - 박스 그래프: boxplot()
  - 점 그래프: dotchart()
  - 막대 그래프: barplot()
  - 히스토그램: hist()
  - 산포도: plot()
  - 선그래프: plot()
- ggplot2() 패키지와 데이터 시각화
  - ggplot2 개요와 기초
  - ggplot2로 그래프 작성하기: geom()
    - 한 변수, 두 변수, 세 변수, 연속 이변량 변수, 공간 자료 등 시각화
  - ggplot2로 그래프 작성하기: stat()
  - ggplot2로 그래프 꾸미기: scale_*()
  - ggplot2로 그래프 꾸미기: facet_*()
  - ggplot2로 그래프 꾸미기: position adj.
    - 그래프 표현 위치 등 조정
  - ggplot2로 그래프 꾸미기: Labels
  - ggplot2로 그래프 꾸미기: theme()
  - ggplot으로 그래프 그리기: 종합

시각화의 정의와 역사

- 시각화(visualization)
  - 데이터를 활용하여 차트 또는 그래프로 표출하는 것
    - 표(table)보다 더욱 빠른 이해와 직관을 주어 의사결정에 기여
- R 데이터 시각화
  - R 프로그램에서의 가장 중요한 또는 강력한 장점(특징)들 중 하나
  - 단지 한 단어 또는 한 두줄의 코딩으로 효율적인 그래프 생성
- R 데이터 시각화의 역사
  - 수작업 → 그림 → 엑셀/SPSS → Base R → ggplot2 → shiny/ggVis

출처: https://www.quora.com/How-good-is-R-for-data-visualization

그래프 형태와 선택

- 변수의 개수와 속성에 따른 그래프 선택

변수 개수	변수 형태	그래프
일변량 (변수 1개)	**연속형 데이터**	• 히스토그램 (Histogram) • 커널 밀도 곡선(Kernel Density Curve)
		• **박스 그래프(Box Plot)** • **바이올린 그래프(Violin Plot)**
	범주형 데이터 (명목형, 순서형)	• 막대 그림(Bar Chart) • 원 그림(Pie Chart)
다변량 (변수 2개 이상)	연속형 데이터	• 산점도 (행렬) • 선 그래프 • 시계열 그래프(x 시간*y Value)
	범주형 데이터	• 모자이크 그래프(Mosaic Plot)

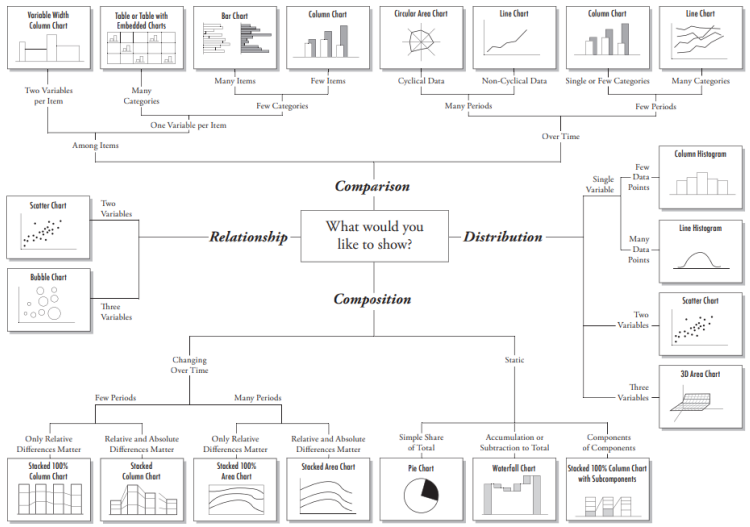
그래프 형태와 선택

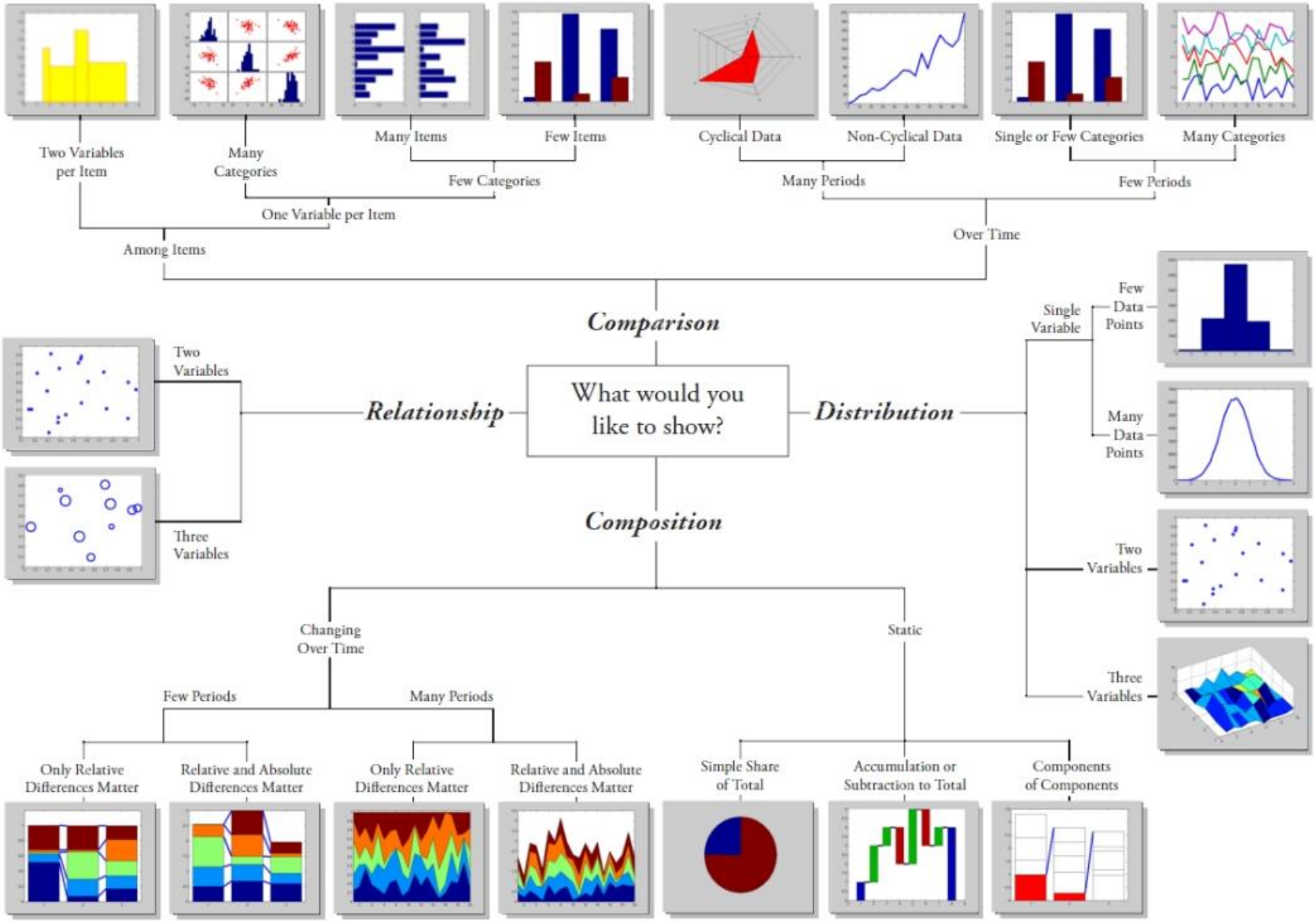
- 데이터 시각화의 4가지 유형
  - 시각화 표출 유형(presentation type)
    - 왜 하는가?, 어떻게 효과적으로 표출할까?
      - 비교(Comparison)
      - 구성(Composition)
      - 분포(Distribution)
      - 관계(Relationship)
  - 데이터 형태와 시각화 표출 유형
    - 표출유형과 데이터와의 관계를 고려한 시각화 표출 유형의 결정
      - 얼마나 많은 변수들이 하나의 그래프에서 표출되기를 원하는가?
      - 각 변수에 대하여 얼마나 많은 데이터 점들이 표현되어질 것인가?
      - 한 시점 또는 항목간 또는 집단간 값들을 비교하고자 하는가?
  - 유형과 데이터 형태에 따른 그래프 유형 선택

시각화와 그래프 유형

그래프 형태

- 데이터 형태와 그래프

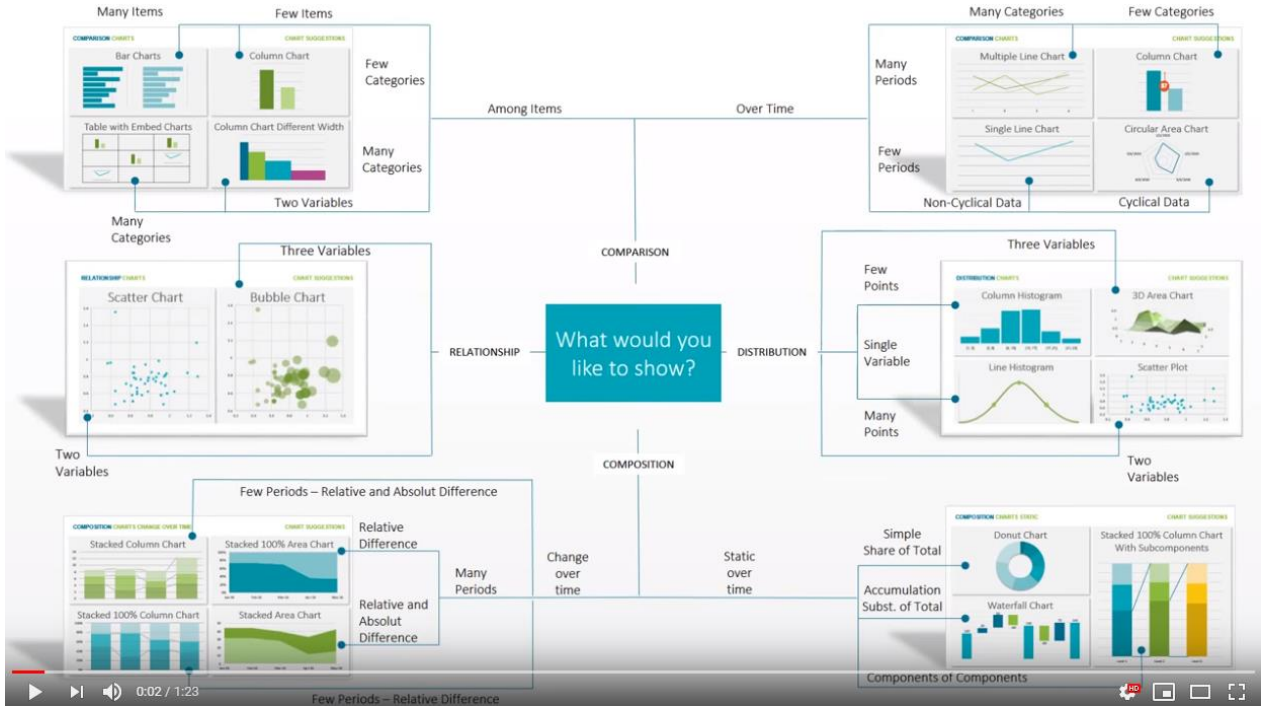




출처1: Original source: Andrew Abela(2009)[left]
출처2: https://i1.wp.com/www.tatvic.com/blog/wp-content/uploads/2016/12/Pic_2.png
출처3: https://apandre.files.wordpress.com/2011/02/chartchooserincolor.jpg [right]

그래프 형태와 선택

- YouTube 영상
  - Andrew Abela - Chart Chooser PowerPoint Template - By SlideModel
  - https://www.youtube.com/watch?v=00zjDdXUcy4



R 그래프의 par(), 색상 pallete 이해하기

- R 그래픽 패러미터 함수: par()
  - 그래프 속성의 정의(설정)
    - 설정 이후 그래픽 환경에 영향
      - 화면(창)의 분할, 여백, 점 크기 및 색 등
    - 단, 해당 plot() 함수 내 그래픽 속성 변경 가능
  - 화면(창)의 그래프 개수 설정
    - mfcol = c(nrow,ncol)
    - mfrow = c(nrow,ncol)
  - 그래프의 여백 설정
    - 내부: mar = c(bottom, left, top, right)
      - default: c(5.1, 4.1, 4.1, 2.1) lines
    - 외부: oma = c(bottom, left, top, right)
      - default: c(0, 0, 0, 0) lines

R 그래프의 par(), 색상 pallete 이해하기

- R 그래픽 패러미터 함수: par()
  - 인수 설명(표로 작성...)

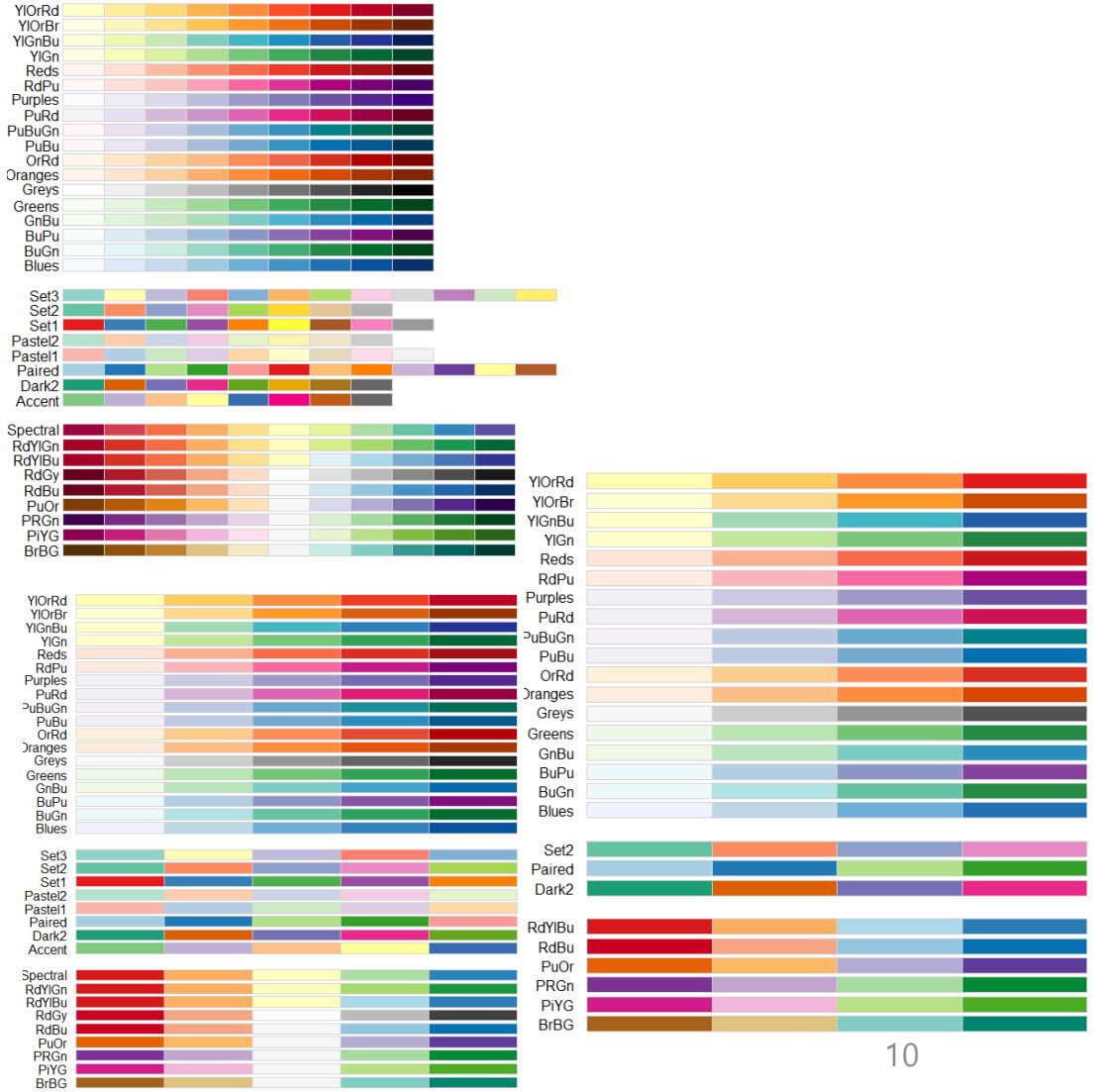
인수	내용설명
mfrow / mfcol	한 화면에 여러 그래프를 비교해볼 때 사용. figure region을 나누어 plot region을 만들고 배치 순서를 정하는 인수.
mfrow	그래프를 행 우선 배치. default =c(1,1)
mfcol	그래프를 열 우선 배치. default =c(1,1)
fig	분할된 영역의 크기 및 위치 지정. default =c(0,1,0,1) →복잡하게 분할할 경우 네 개의 인수를 일일이 계산해야 하므로 layout() 등 타 함수를 쓰는게 효율적
new	새로운 그래픽 함수 호출 시, figure regio을 초기화 할 것인가? 두개이상의 그래픽 함수를 한 figure region에 나타낼 때 주로 사용. default=FALSE → 현재 figure region은 초기화 시키고 새로운 그래프를 그림. default=TRUE → 화면 분할 없이 그래프 하나만 쓸 때, 이전 결과에 겹쳐서 새 결과를 출력함.
type	좌표 영역에 데이터를 출력하는 형태. "p" – points, 점 / "l" – line, 선 / "b" – both points and lines, 점과 선 / "c" – empty points joind by lines, 점이 빠진 선 "o" – overplotted p, and l, 겹친 점과 선/ "s"나 "S" – stair steps, 계단 모양 / "h" – histogram-like vertical lines, 수직선
mar	margin. 여백 지정.(plot margin) default=c(5,4,4,2)+0.1
oma	outer margin. 여백 지정(outer margin)
pty	plot type. plot region의 모양을 지정. "s"(square, x축과 y축 스케일(비율)이 동일하게), "m"(maximal, 최대 크기로)
bty	box type. plot region 둘레의 상자 모양을 지정. "o", "l", "7", "c", "u", "]" 글자와 닮은 형태. default="o"

인수	내용설명
pch	point character. 점의 모양. default=1
cex	character expansion. 문자나 점의 크기. default=1
lty	line type. 선의 모양. default="solid"(1)
col	color. 색상 지정. 1~8까지 정해진 색깔이 있음. "#RRGGBB"형식으로 색상 지정 가능. "begie", "aquamarine" 등 이름으로 색상 지정 가능
str	string rotation. 문자열 회전. default=0. (단위 degree, 음수도 사용가능) text() 함수에서 사용
family	"sans", "serif", "mono" 등 문자의 폰트 종류 지정
font	폰트 체 지정. "plain"=1, "bold"=2, "italic"=3, "bold italic"=4, "symbol"=5
fg	전경색. default="black"
bg	배경색. default="transparent"
tck	tick marks. 좌표 눈금선의 길이 지정. plot region의 크기를 기준으로 값이 계산됨
tcl	tick length. 좌표 눈금선의 길이 지정. cex=1일 때 문자의 길이를 tcl=1로 하여 값이 계산됨

출처: http://blog.naver.com/PostView.nhn?blogId=dic1224&logNo=80208529054

R 그래프의 par(), 색상 pallete 이해하기

- R 팔레트(Palettes)
  - install.packages("RColorBrewer")
  - library(RColorBrewer)
  - display.brewer.all() # 이산형 또는 범주형 변수들의 색상 지원
    - 순위형(Sequential), 발산형(Diverging), 범주형(Qualitative) 변수들의 색상 유형 제공
    - display.brewer.all(n=5) # 필요한 색의 수 = 5
    - display.brewer.all(colorblindFriendly = TRUE, n=4) # 색맹친화적인 색상 선택
  - 기타 색상 제공 함수
    - rainbow(), heat.colors(),
    - topo.colors(), cm.colors(),



R 시각화 내장함수 종류

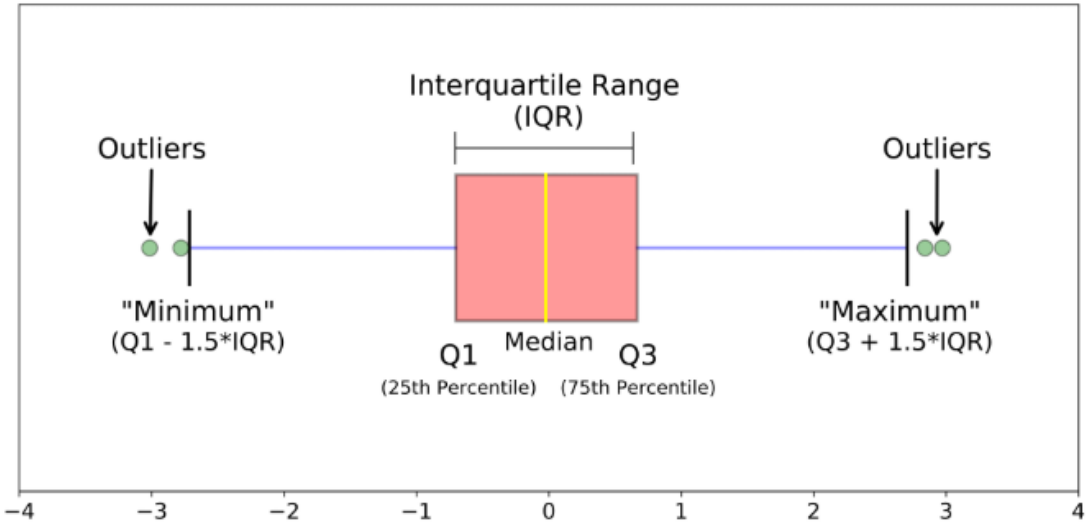
- R 시각화 함수
  - 내장함수(built-in functions):
    - 예: plot(), hist() 등
  - 외장함수: 패키지
    - ggplot2, leaflet, lattice 등

- R 그래프 내장함수
  - 기본 그래프
    - boxplot(x, ...) : 박스그래프
    - hist(x, ...) : 히스토그램
    - barplot(x, ...) : 막대그래프
    - dotchart(x, ...) : 점그래프
    - plot(x, ...) : 기본 그래프(선, 막대, 산점도 등 표현)
    - plot() 함수 실행 후 새 그래프 추가 가능
      - 추가할 새로운 함수 내에서 **"add=TRUE"**로 실행
        - 예: lines(x, add = TRUE)
    - 추가 가능 그래프(내용) 함수
      - 선그래프 : lines()
      - 점그래프 : points()
      - 축 : axis (side,...)
      - 축 라벨 : mtext(text,...)
      - 텍스트 : text(x, y, text,...)
      - 제목 : title(main,...)
      - 그리드: grid(.....)

내장함수로 그래프 작성하기

박스 그래프: boxplot()

- '상자 수염 그림'(Box-and-Whisker Plot) '상자 그림' 이라고도 함
  - 기술 통계학에서의 수치적 자료를 표현
  - 분포의 형태를 알 수 있음
  - 항목간 요약통계량과 분포 비교 가능
- 통계자료에서 얻은 다섯 수치 요약(five number summary) 통계량을 활용
  - 최솟값 : 제 1사분위에서 1.5 IQR을 뺀 위치
  - 제 1사분위(Q1) : 25%의 위치
  - 제 2사분위(Q2) : 50%의 위치로 중앙값(median)
  - 제 3사분위(Q3) : 75%의 위치
  - 최댓값 : 제 3사분위에서 1.5 IQR을 더한 위치
- 이상치(Outlier)
  - 최솟값과 최댓값을 넘어가는 위치에 있는 값



출처: https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51

내장함수로 그래프 작성하기

박스 그래프: boxplot()

- 실습
  - 데이터 불러오기
    - setwd(): 작업폴더 설정
    - read.csv(): 데이터 불러오기
    - str(): 데이터 구조 확인하기
  - 데이터 수정하기
    - 아파트 건축년대(1970s, 1980s, 1990s, 2000s, 2010s) 요인변수 생성과 확인
  - 데이터 접근하기: attach()

```
setwd("E:\\강의\\수치해석\\4주차_R_정형데이터_시각화와_ggplot패키지") # 작업 경로(폴더) 설정
apt <- read.csv('데이터_아파트매매가격.csv') # 실습데이터 불러오기(아파트 실거래 자료)
str(apt) # 데이터 구조 확인하기
apt$price_pyung = apt$apt_price / apt$area_m2 * 3.3 # 평당 아파트 거래가격(만원) 생성하기
str(apt$price_pyung) # 생성된 아파트 평당 가격 구조 확인
x <- ifelse(year_built <1980, "1970s", # 아파트 건축년대 변수생성 작업: 1970년대 건축된 아파트
            ifelse(year_built >=1980 & year_built < 1990, "1980s" , # 1980년대 건축된 아파트
                   ifelse(year_built >=1990 & year_built < 2000, "1990s" , # 1990년대 건축된 아파트
                          ifelse(year_built >=2000 & year_built < 2010, "2000s" ,  # 2000년대 건축된 아파트
                                 "2010s")))) # 2010년대 건축된 아파트
apt <- transform(apt, yr_built = x) # 아파트 건축년대 집단
table(apt$yr_built)

attach(apt)
```

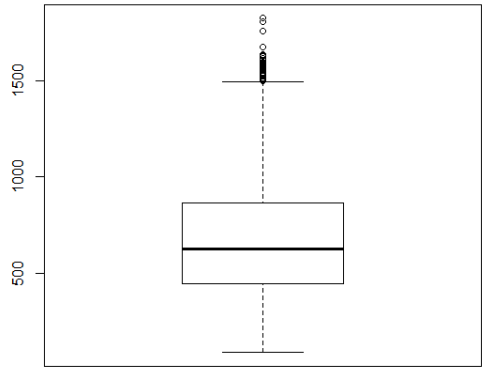
내장함수로 그래프 작성하기

박스 그래프: boxplot()

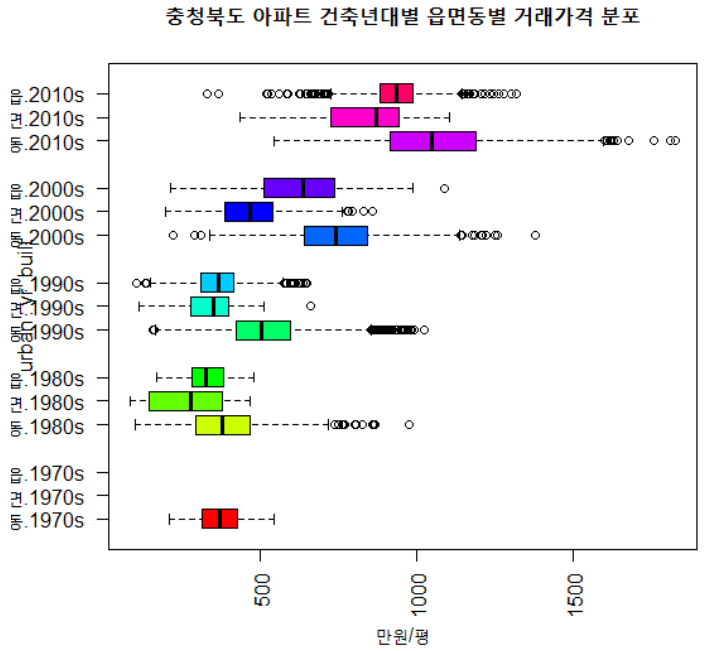
- 실습

```
?boxplot() # Box Plots
boxplot(price_pyung) # 평당 가격 박스 그래프
boxplot(price_pyung ~ yr_built) # 아파트 건축년대별 평당 가격 박스 그래프

boxplot(price_pyung ~ yr_built, col=(c("green", "blue", "yellow", "red", "grey"))
brewer.pal.info # 선택가능한 계열 색상 종류와 크기 확인
my_pal <- brewer.pal(5,"Oranges") # 오렌지계통의 5가지 색상 설정
boxplot(price_pyung ~ yr_built, col= my_pal) # 색상 변환

boxplot(price_pyung ~ urban + yr_built, # 읍면동별 아파트 건축년대별 박스 그래프
                horizontal = TRUE) # 수평으로 박스 그래프
boxplot(price_pyung ~ urban + yr_built,
        horizontal = TRUE, # 수평으로 박스 그래프
        las=2, # las=2, 축의 레이블을 세로로 나타내기
        at = c(1:3, 5:7, 9:11, 13:15, 17:19), # 박스그래프의 위치 설정(년대별 한칸 띄우기)
        col=rainbow(15),  # 무지개색 계통의 색상 15개 자동 설정
        main = "충청북도 아파트 건축년대별 읍면동별 거래가격 분포") # 제목 설정
?grid
grid(col="gray", lty=5, lwd=1.5) # 그리드 추가: 회색, 선형태, 두께

?rainbow # Color Palettes; heat.colors, terrain.colors 등 다수 사용 가능
show_line_types() # 선형태 확인하기
```

```
> boxplot(price_pyung) # 평당 가격 박스 그래프
```




```
> grid(col="gray", lty=5, lwd=1.5) # 그리드 추가: 회색, 선형태, 두께
> boxplot(price_pyung ~ urban + yr_built,
+         horizontal = TRUE, # 수평으로 박스 그래프
+         las=2, # las=2, 축의 레이블을 세로로 나타내기
+         at = c(1:3, 5:7, 9:11, 13:15, 17:19), # 박스그래프의 위치 설정(년대별 한칸 띄우기)
+         col=rainbow(15),  # 무지개색 계통의 색상 15개 자동 설정
+         main = "충청북도 아파트 건축년대별 읍면동별 거래가격 분포", # 제목 설정
+         xlab = "만원/평") # x축 제목 설정
```

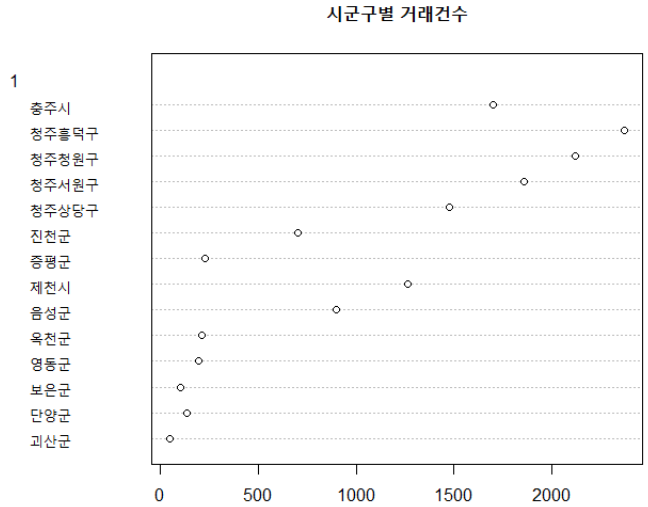


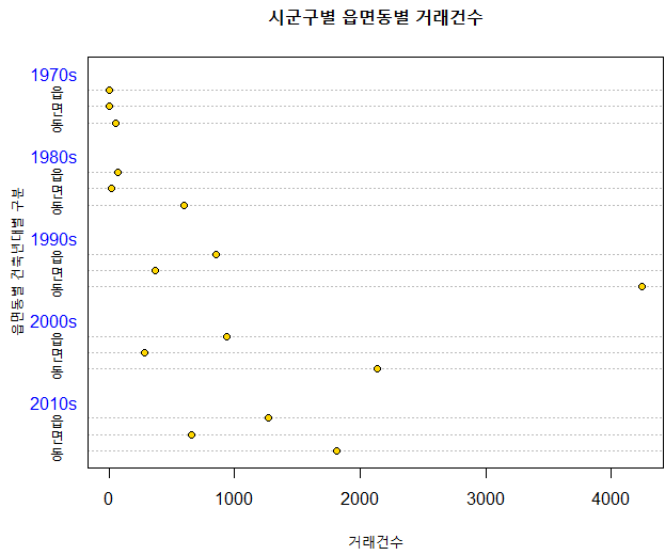
내장함수로 그래프 작성하기

점 그래프: dotchart()

- 정의
  - 점 도표(點圖表)라고도 함
  - 데이터들의 분포를 점을 통해 그래프로 나타냄
- 실습
  - ?dotchart() # Dot Plots: 벡터나 행렬에서만 유효한 사용 함수

```
> ### 점 그래프
> ?dotchart() # Dot Plots: 벡터나 행렬에서만 유효한 사용 함수
> x <- as.matrix(table(address_sigungu)) # 시군구별 일원 빈도분포표 행렬 할당
> dotchart(x, main = "시군구별 거래건수")
> table(address_sigungu) # 시군구별 빈도분포표와 비교
address_sigungu
    괴산군     단양군     보은군     영동군     옥천군     음성군     제천시     증평군
        48        135        102        198        213        897       1264
청주흥덕구     충주시
      2370       1698
> y <- as.matrix(table(urban, yr_built)) # 읍면동별 건축년대별 이원빈도분포표 행렬 할당
> dotchart(y,
+          xlab="거래건수",
+          ylab="읍면동별 건축년대별 구분",
+          bg="gold",
+          gcolor="blue",
+          main = "시군구별 읍면동별 거래건수")
```





막대 그래프: barplot()

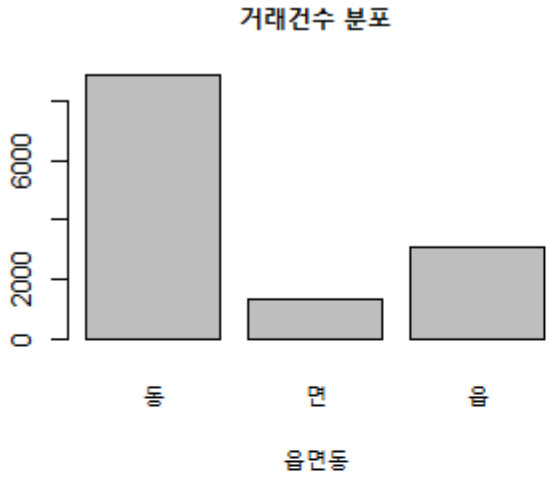
- **막대 그래프**(-graph)의 정의
  - **바 차트**(bar chart), **바 그래프**(bar graph)로 불림
  - 표현 값에 비례하여 높이와 길이를 지닌 직사각형 막대로 범주형 데이터를 표현하는 차트나 그래프
  - 막대는 수직으로나 수평으로 작성 가능
- 실습
  - ?barplot() : Creates a bar plot with vertical or horizontal bars.

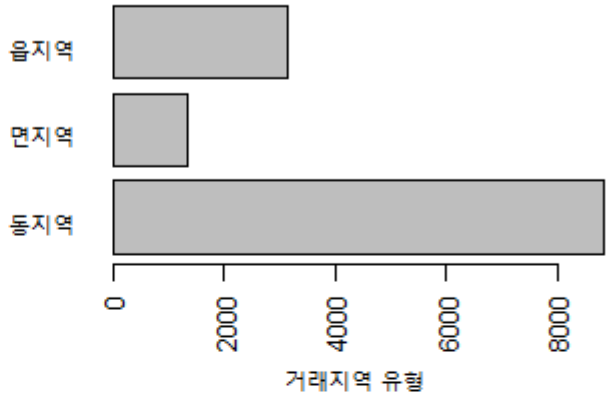
내장함수로 그래프 작성하기

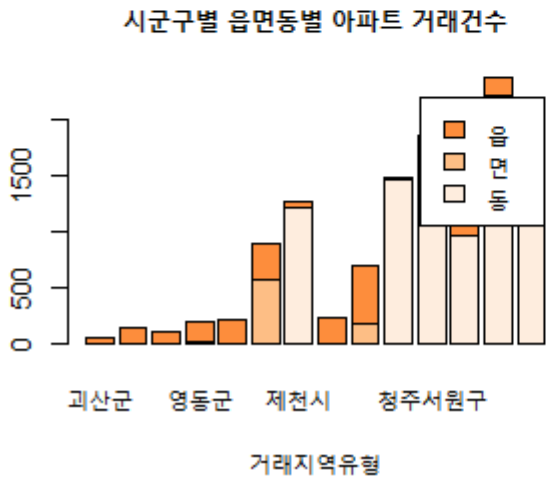
막대 그래프: barplot()

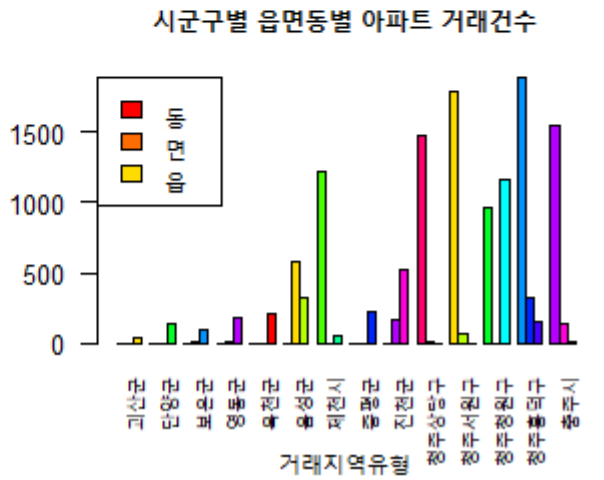
- 실습
  - ?barplot() :

```
> ### 막대그래프
> ?barplot # Bar Plots: 막대그래프: Creates a bar plot with vertical or hor
> par(mfrow=c(2,2)) # 그래프 패러미터 설정: 한 화면에 2*2행렬 창 설정
> counts <- table(urban)
> barplot(counts, main="거래건수 분포",
+         xlab="읍면등") # las=2, 축의 레이블을 세로로 나타내기
> barplot(counts, horiz=TRUE, xlab="거래지역 유형",
+         names.arg=c("동지역", "면지역", "읍지역"),
+         las= 2)
> # Stacked Bar Plot with Colors and Legend
> counts <- table(urban, address_sigungu)
> class(counts); ncol(counts)
[1] "table"
[1] 14
> barplot(counts, main="시군구별 읍면동별 아파트 거래건수",
+         xlab="거래지역유형", col=my_pal,
+         legend = rownames(counts))
> barplot(counts, main="시군구별 읍면동별 아파트 거래건수",
+         xlab="거래지역유형", # x축 제목
+         col=rainbow(ncol(counts)), # 색상과 색 갯수  설정
+         legend = rownames(counts),
+         beside=TRUE, # values in each column are juxtaposed rather than
+         args.legend = list(x="topleft"), # 범례 위치 변경
+         las =2 ) # las=2, 축의 레이블을 세로로 나타내기
```









히스토그램: hist()

- **히스토그램**(histogram)의 정의
  - 표로 되어 있는 도수 분포를 정보 그림으로 나타낸 것
    - 가로축이 계급, 세로축이 도수(값)
    - 반대로 작성 가능
    - 계급은 보통 변수의 구간이며, 서로 겹치지 않음
    - 계급(막대기)끼리는 서로 붙어 있어야 함
  - 막대 그래프와의 차이
    - 막대그래프는 계급 즉 가로를 고려하지 않고 세로의 높이(height)로만 표현
    - 히스토그램은 가로와 세로를 함께 표현

내장함수로 그래프 작성하기

히스토그램: hist()

- hist()의 주요 인수(argument)
  - X 축 계급(또는 구간) 설정방법
    - Breaks =

one of:

- a vector giving the breakpoints between histogram cells,
- a function to compute the vector of breakpoints,
- a single number giving the number of cells for the histogram,
- a character string naming an algorithm to compute the number of cells (see 'Details'),
- a function to compute the number of cells.

  - Y축 값 설정
    - freq = TRUE/FALSE

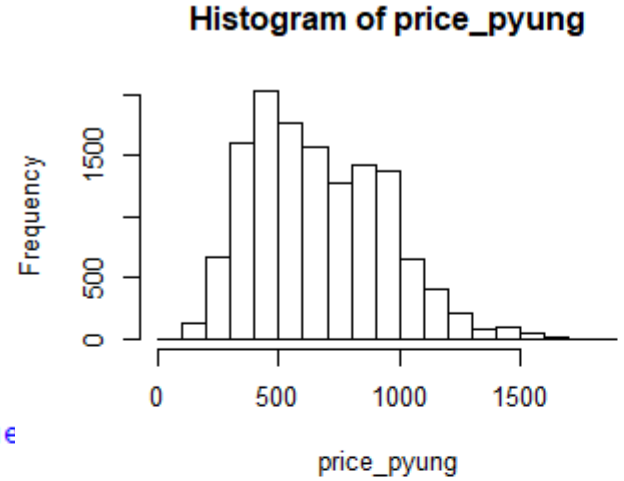
logical; if `TRUE`, the histogram graphic is a representation of frequencies, the `counts` component of the result; if `FALSE`, probability densities, component `density`, are plotted (so that the histogram has a total area of one). Defaults to `TRUE` *if and only if* `breaks` are equidistant (and `probability` is not specified).

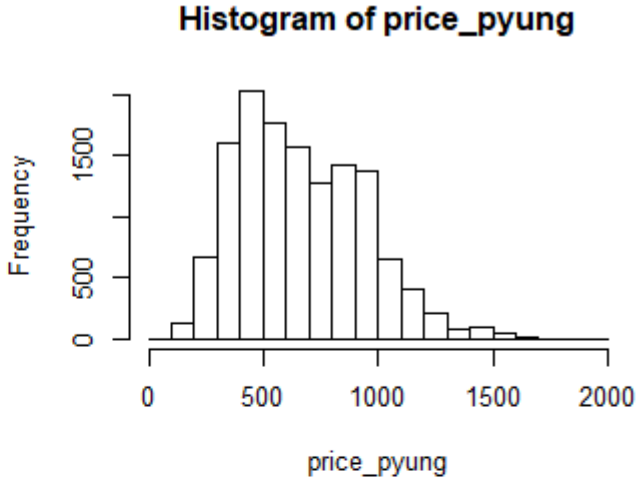
내장함수로 그래프 작성하기

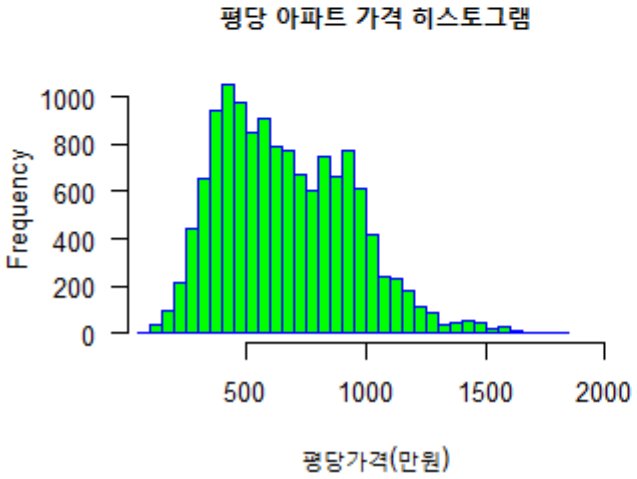
히스토그램: hist()

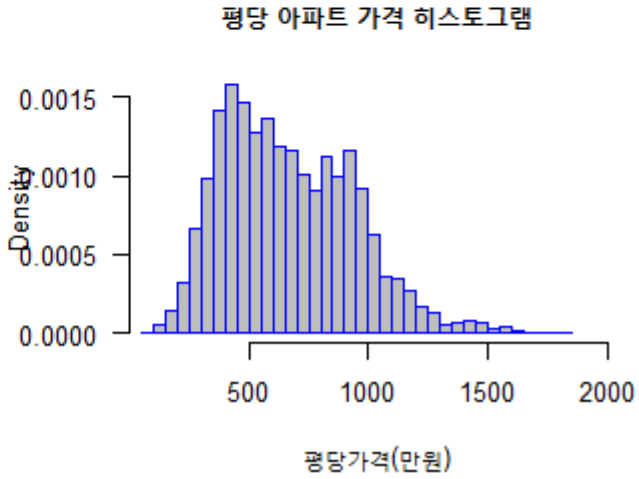
- 실습: hist()

```
> ### 히스토그램: hist()
> ?hist # Histograms
> hist(price_pyung) # 평당 가격에 대한 히스토 그램
> summary(price_pyung)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  85.48  444.44  627.35  666.15  866.05 1825.47
> hist(price_pyung, breaks=seq(0, 2000, 100)) # seque
> hist(price_pyung, # 평당 가격
+      main="평당 아파트 가격 히스토그램", # 제목
+      xlab="평당가격(만원)",  # x축 제목
+      border="blue", # 경계 색상
+      col="green", # 색상 설정
+      xlim=c(80,2000), # x축 최소 및 최대 값 설정
+      las=1, # 값 라벨 가로로
+      breaks=30) # 30개 구간으로 설정
> hist(price_pyung, # 평당 가격
+      main="평당 아파트 가격 히스토그램", # 제목
+      xlab="평당가격(만원)",  # x축 제목
+      freq=FALSE, # y축 값 표현을 확률밀도로
+      border="blue", # 경계 색상
+      col="gray", # 색상 설정
+      xlim=c(80,2000), # x축 최소 및 최대 값 설정
+      las=1, # 값 라벨 가로로
+      breaks=50) # 50개 구간으로 설정
```





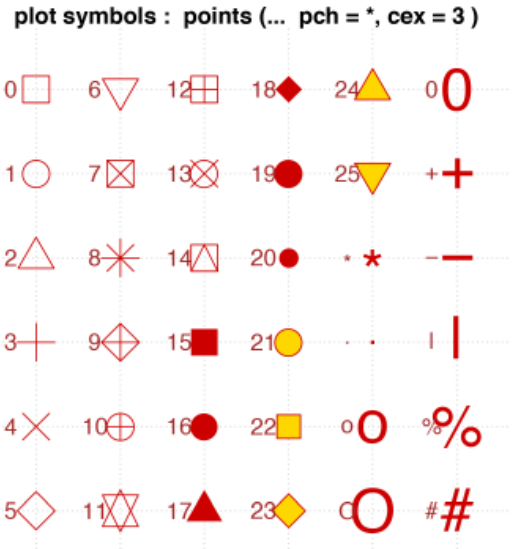




내장함수로 그래프 작성하기

산포도: plot()

- plot() 함수
  - 단일 변수 그래프: plot(x, ...)
    - 선그래프(뒤에 설명)
  - 두 변수 그래프: plot(x, y, ...)
    - 산포도
  - 인수(arguments) 활용 그래프 다듬기
    - 점의 유형(pch)과 불러오기 숫자
      - 예: pch=13



인수	설명
main="메인제목"	제목 설정
sub = "서브 제목"	서브 제목
xlab = "문자", ylab = "문자"	x, y축에 사용할 문자열을 지정합니다.
ann=F	x, y축 제목을 지정하지 않습니다.
tmag=2	제목 등에 사용되는 문자의 확대율 지정
axes=F	x, y축을 표시하지 않습니다.
axis	x, y축을 사용자의 지정값으로 표시합니다.
그래프 타입 선택	
type="p"	점 모양 그래프 (기본값)
type="l"	선 모양 그래프 (꺾은선 그래프)
type="b"	점과 선 모양 그래프
type="c"	"b"에서 점을 생략한 모양
type="o"	점과 선을 중첩해서 그린 그래프
type="h"	각 점에서 x축까지의 수직선 그래프
type="s"	왼쪽값을 기초로 계단모양으로 연결한 그래프
type="S"	오른쪽 값을 기초로 계단모양으로 연결한 그래프
type="n"	축만 그리고 그래프는 그리지 않습니다.
선의 모양 선택	
lty=0, lty="blank"	투명선
lty=1, lty="solid"	실선
lty=2, lty="dashed"	대쉬선
lty=3, lty="dotted"	점선
lty=4, lty="dotdash"	점선과 대쉬선
lty=5, lty="longdash"	긴 대쉬선
lty=6, lty="twodash"	2개의 대쉬선
색, 기호 등	
col=1, col="blue"	기호의 색지정, 1-검정, 2-빨강, 3-초록, 4-파랑, 5-연파랑, 6-보라, 7-노랑, 8-회색
pch=0, pch="문자"	점의 모양을 지정합니다.
bg="blue"	그래프의 배경색 지정
lwd="숫자"	선을 그릴 때 선의 굵기를 지정
cex="숫자"	점이나 문자를 그릴 때 점이나 문자의 굵기를 지정

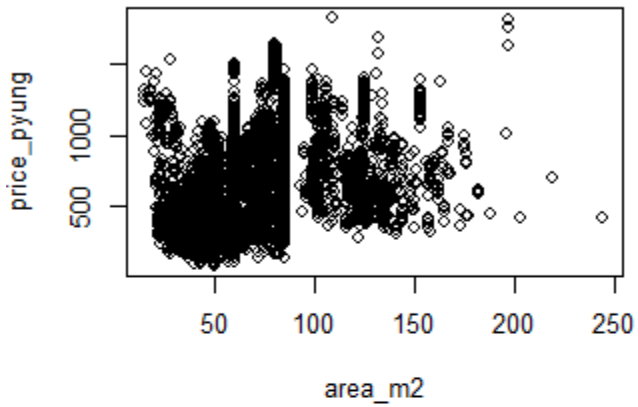
https://m.blog.naver.com/PostView.nhn?blogId=1stwook&logNo=220670083505&proxyReferer=https%3A%2F%2Fwww.google.com%2F

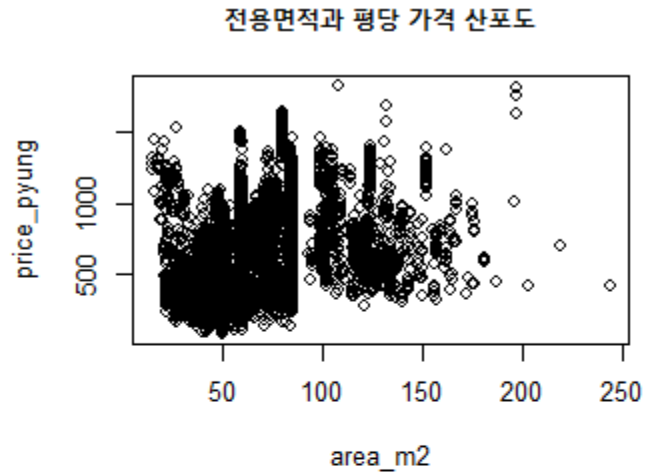
내장함수로 그래프 작성하기

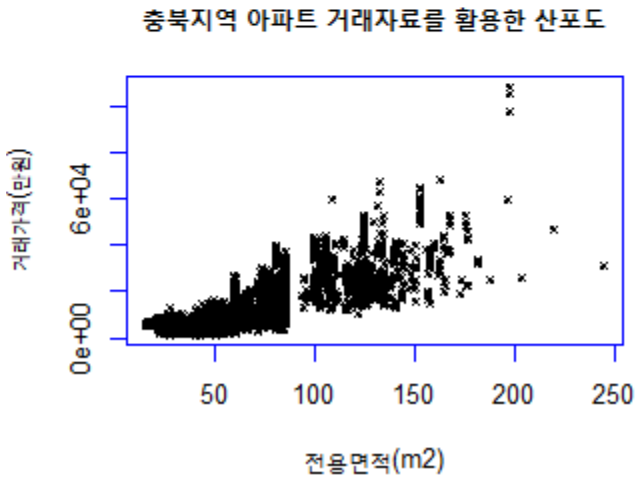
산포도: plot() 함수

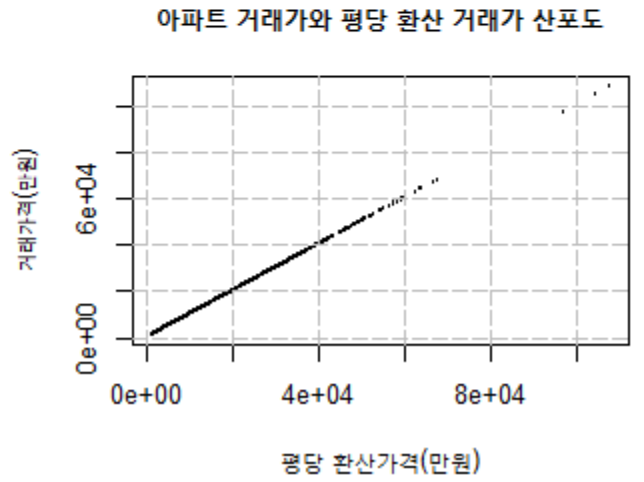
- 실습: plot() 함수

```
> plot(price_pyung) # 평당 가격
> plot(area_m2, price_pyung)
> plot(area_m2, price_pyung, # 전용면적과 평당 가격 산포도 작성
+      main = "전용면적과 평당 가격 산포도")
> plot(area_m2, apt_price,
+      cex = 0.5,  # 2배 작게
+      fg="blue", # 전경색 테두리 블루
+      pch=13, # 1 ~ 25개의 모양이 준비되어 있음
+      main = "충북지역 아파트 거래자료를 활용한 산포도",
+      xlab = "전용면적(m2)",
+      ylab = "거래가격(만원)")
> plot(apt_price, apt_price, # 실거래가와 평당 환산 거래가격 산포도
+      cex = 0.3, # 점크기 0.3배
+      main = "아파트 거래가와 평당 환산 거래가 산포도",
+      xlab = "평당 환산가격(만원)",
+      ylab = "거래가격(만원)")
> grid(col="gray", lty=5, lwd=1.5) # 그리드 추가: 회색, 선형태, 두께
> detach()
> par(mfrow=c(1,1)) # 그래픽 패러미터: 한 창에 한 개 그래프 설정
```









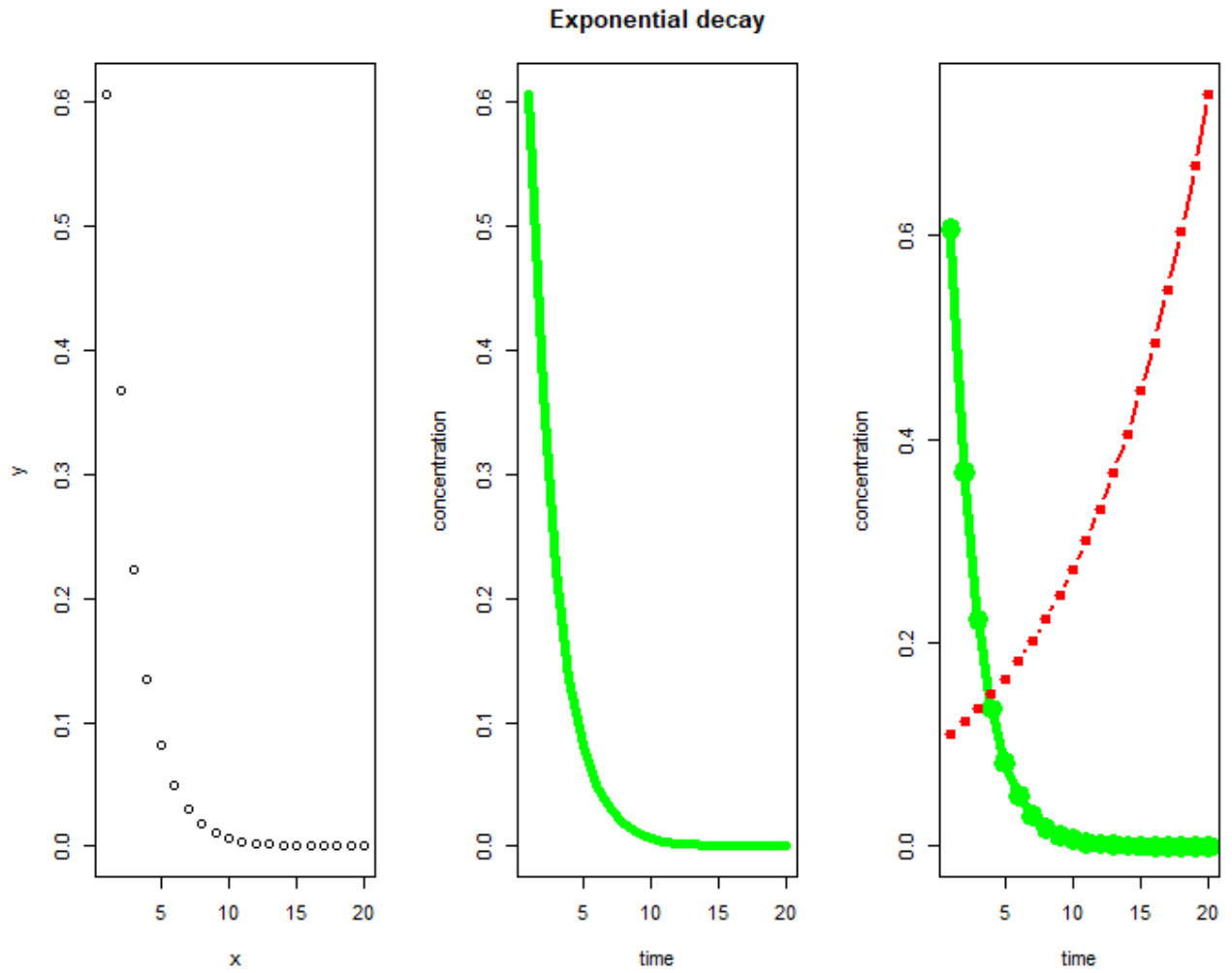
선그래프: plot()

- 선그래프
  - 꺾은 선 그래프라고도 함
  - 수량을 점으로 표시하고 그 점들을 선분으로 이어 그린 그래프
  - 시간에 따라 지속적으로 변화하는 것을 표현할 때 유용
    - 조사하지 않은 중간의 값도 대략 예측할 수 있음(외삽법)
  - 주요 활용 예시
    - 연속적 자료와 통시적 자료
    - 동일하거나 일련의 관찰대상의 추이 비교
    - 추세(trend) 관찰
- 선그래프 작성 plot() 함수
  - 실습

선그래프: plot() 함수

- 실습: plot() 함수
  - 선그래프 작성

```
> #### plot()함수: 선그래프 작성
> par(mfrow=c(1,3)) # 그래픽 패러미터: 한 창에 세 개 그래프 설정
> x <- 1:20 # 1부터 20까지
> y <-  exp(-x/2) # y 값 산정
> plot(x,y)
> plot(x,y,
+      type="l", # 선형태
+      col="green", # 선색
+      lwd=5, # 선 두께
+      xlab="time", # x축 제목
+      ylab="concentration", # y축 제목
+      main="Exponential decay") # 제목
> z = 0.1*exp(x/10) # z값 설정
> plot(x,y, type="b", col="green", lwd=5, pch=8,
+      xlab="time", ylab="concentration", ylim=range(y, z))
> lines(x, z, # 선그래프 추가
+       type="b", col="red", lwd=2, pch=19)
> par(mfrow=c(1,1)) # 그래픽 패러미터: 한 창에 한 개 그래프 설정
```



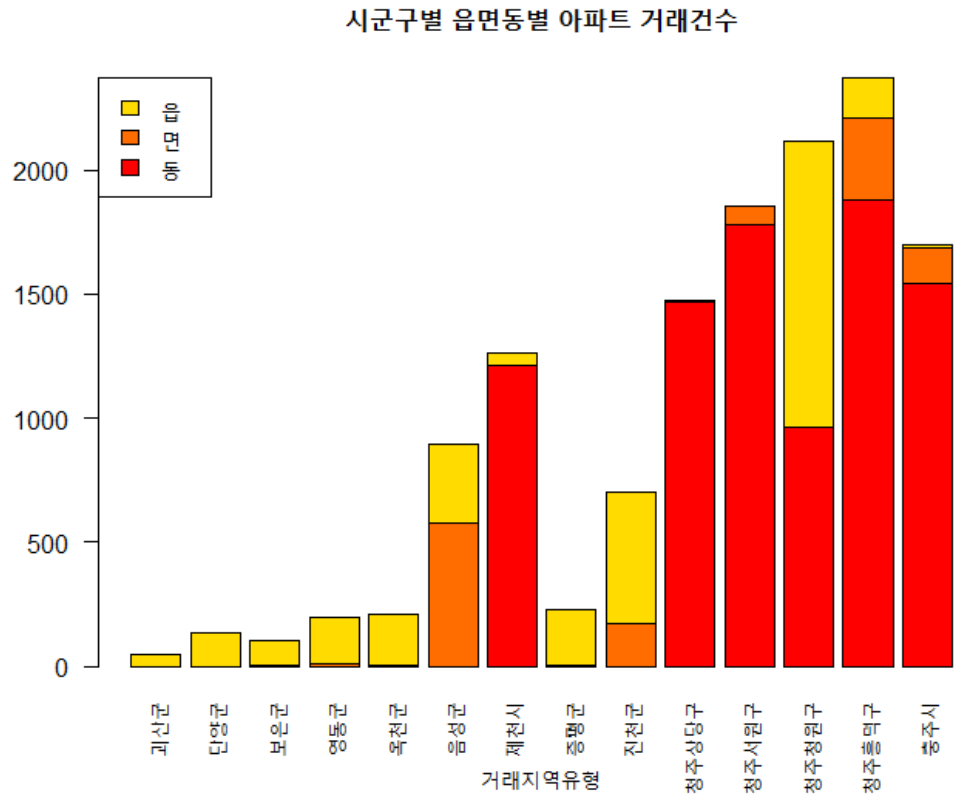
연습문제 01

- 다음 중 R에서 벡터나 행렬에서만 작성할 수 있는 그래프 함수를 모두 고르시오.

① 박스 그래프: boxplot()
② 점 그래프: dotchart()
③ 막대 그래프: barplot()
④ 히스토그램: hist()
⑤ 산포도: plot()
⑥ 선그래프: plot()

연습문제 02

- 시군구별 읍면동별 아파트 거래건수를 아래와 같이 그래프로 작성하고자 한다. 적합한 그래프 함수 명령문을 적으시오.



연습문제 03

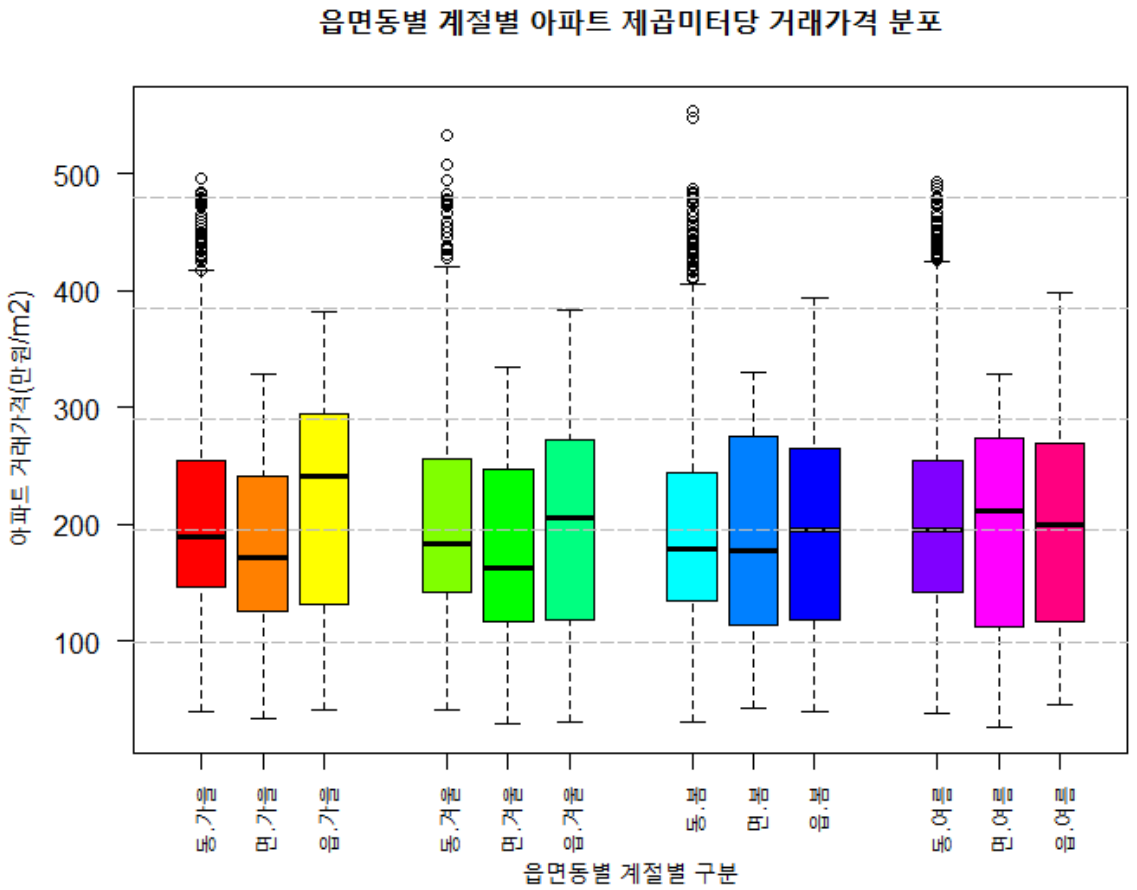
- 실습 데이터인 “데이터_아파트매매가격.csv“에 있는 아파트 거래년월(ym_sale)을 활용하여 계절(season) 변수를 봄, 여름, 가을, 겨울로 구분하여 해당 데이터에 요인변수를 생성하여 그래프를 작성하고자 한다.
- ifelse문을 활용하여 season변수를 생성하는 명령문을 작성하시오.
  - 이 때, 실습데이터를 apt로 객체에 할당한 후 추가 생성 변수(season)를 ifelse문으로 해당 데이터에 추가하시오.
  - 봄=201903:201905, 여름=201906:201908, 가을=201809:201811, 겨울=201812:201902로 구분하시오.
- 그리고 어느 계절에 가장 많은 아파트 거래 건수가 있으며, 거래건수가 얼마인지 적으시오.

연습문제 04

- 연습문제 3번과 같이 조건에 맞는 값으로 데이터를 구분하는 새로운 변수를 생성할 때, ifelse문 이외에 사용할 수 있는 방법은 무엇인가요?

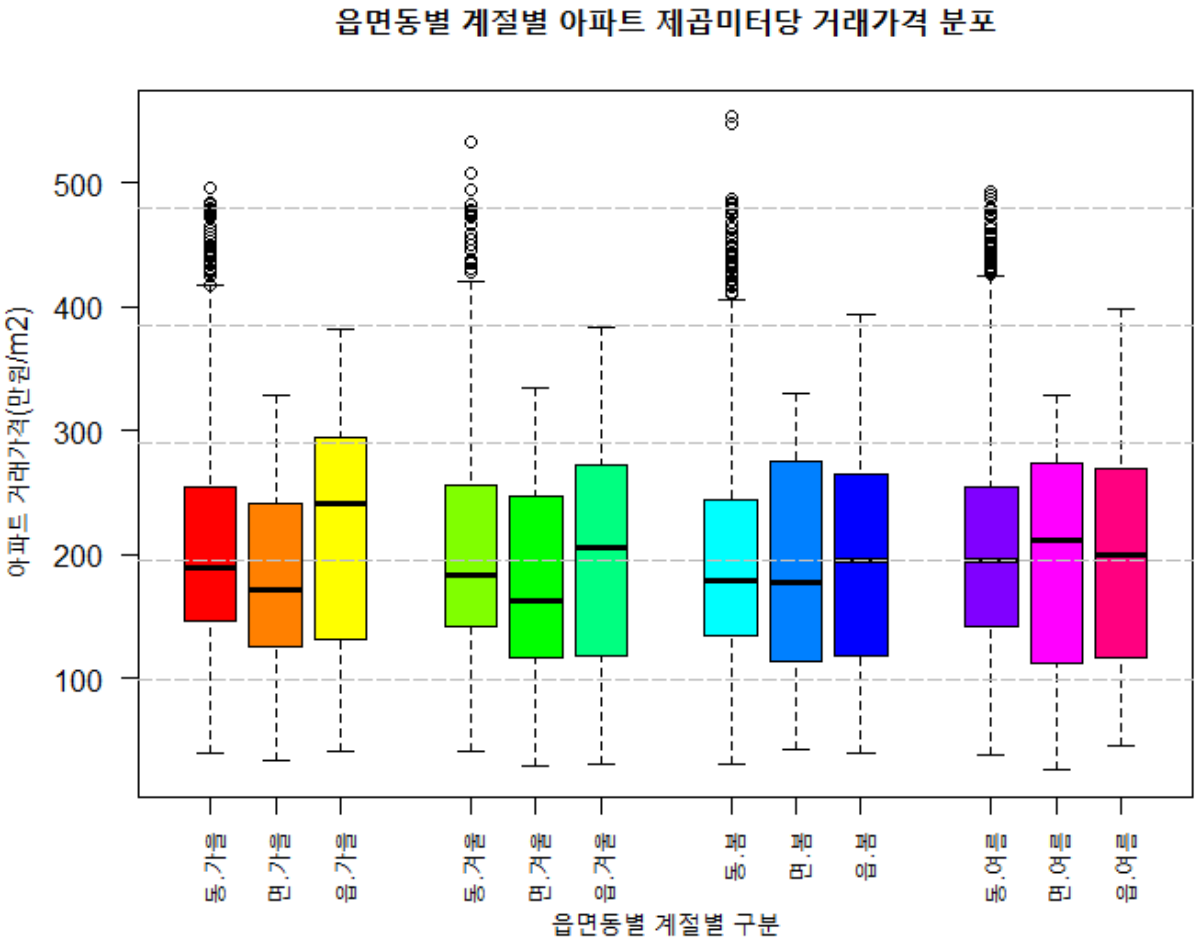
연습문제 05

- 제곱미터당 아파트 거래가격 변수를 생성(price_m2)하고, 읍면동별 계절별 박스그래프를 오른쪽과 같이 작성하고자 한다.
- 적합한 명령문들을 적으시오.
- 이 때, 제곱미터당 아파트 가격(price_m2)의 평균은 얼마인가요?



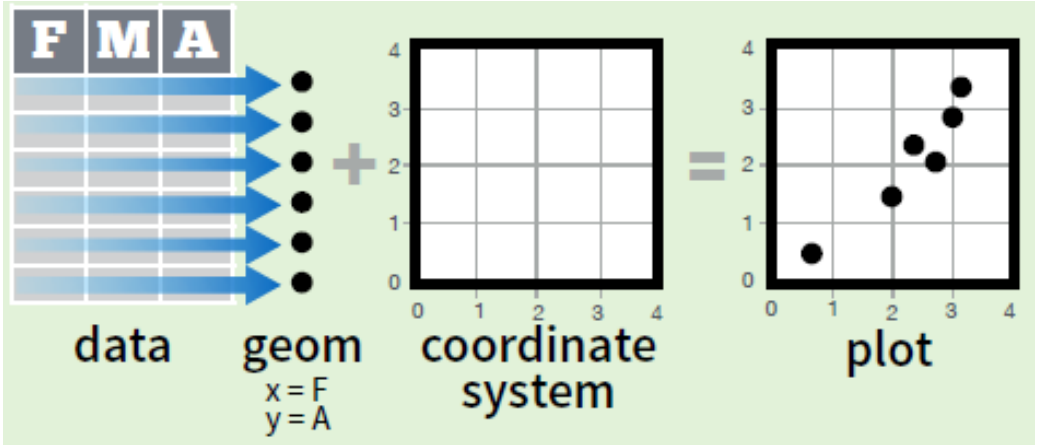
연습문제 06: 생각해보기

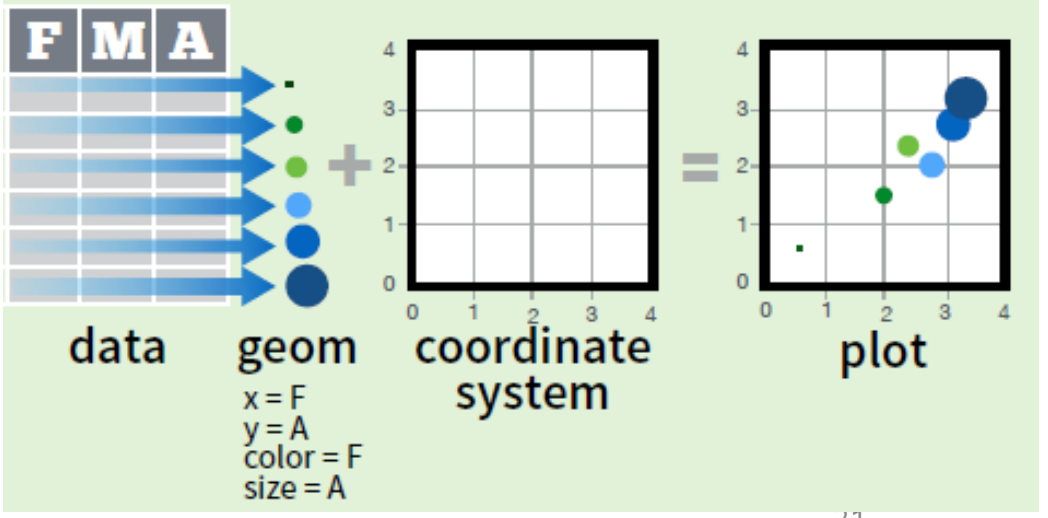
- 왜 가을에 읍지역에서 거래된 아파트의 제곱미터당 중위가격이 가장 높을까요?
  - dplyr패키지의 group_by()와 summarise() 함수를 이용하여 거래가격의 평균과 중위값, 그리고 거래건수를 확인하여 그 이유를 찾으시오.



ggplot2 개요와 기초

- ggplot2 개요
  - 그래픽 문법(grammar of graphics)에 기반한 그래프(plot)
    - 이의 아이디어는 데이터와 좌표체계 등 동일한 구성요소들로 부터 모든 그래프를 작성할 수 있다는 생각에 기반
  - geoms는 좌표체계와 데이터의 점들을 표현하는 시각적 부호(표시)
    - +geom(F, A)
  - 데이터의 값들을 표현하기 위하여 크기, 색상, 좌표(x, y)의 위치들을 geom의 속성들을 미학(aesthestics)적으로 매핑(mapping)함
    - geom 함수 = 데이터의 점들
    - geom의 미학적 속성들 = 데이터의 변수들



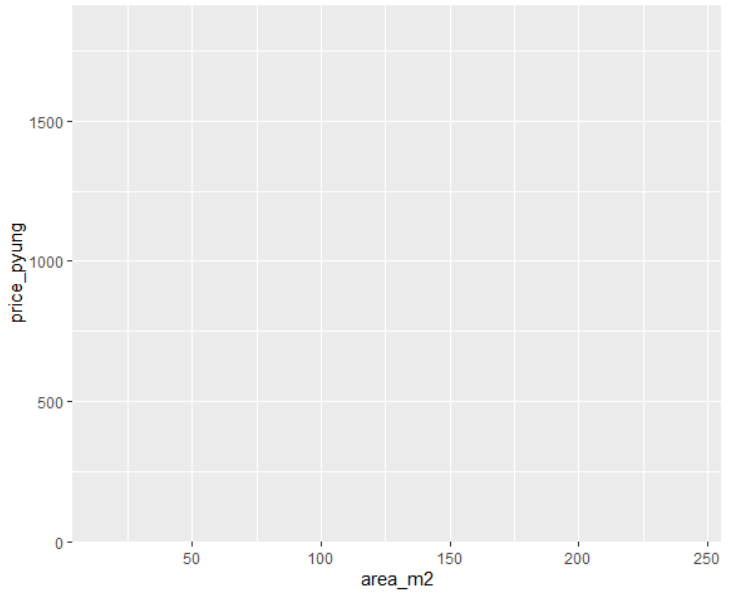


출처: https://www.rstudio.com/wp-content/uploads/2016/11/ggplot2-cheatsheet-2.1.pdf

ggplot2 패키지와 데이터 시각화

ggplot2 개요와 기초

- ggplot2로 그래프 작성 시작 두가지 기본함수
  - ggplot( )
    - Begins a plot that you finish by adding layers to.
    - Add one geom function per layer
  - qplot( )
    - Quick plot
    - Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.
    - a shortcut designed to be familiar if you're used to base plot().
    - It's a convenient wrapper for creating a number of different types of plots using a consistent calling scheme.
    - It's great for allowing you to produce plots quickly, but I **highly recommend learning** ggplot() as it makes it easier to create complex graphics.
- ggplot2 그래프 작성 후 주요 함수
  - last_plot()
    - Returns the last plot
  - ggsave("plot.png", width = 5, height = 5)
    - Saves last plot as 5′ x 5′ file named "plot.png" in working directory.
    - Matches file type to file extension.
    - 그래프 저장 확장자: "eps", "ps", "tex" (pictex), "pdf", "jpeg", "tiff", "png", "bmp", "svg" or "wmf"
- 실습



출처: https://www.rstudio.com/wp-content/uploads/2016/11/ggplot2-cheatsheet-2.1.pdf

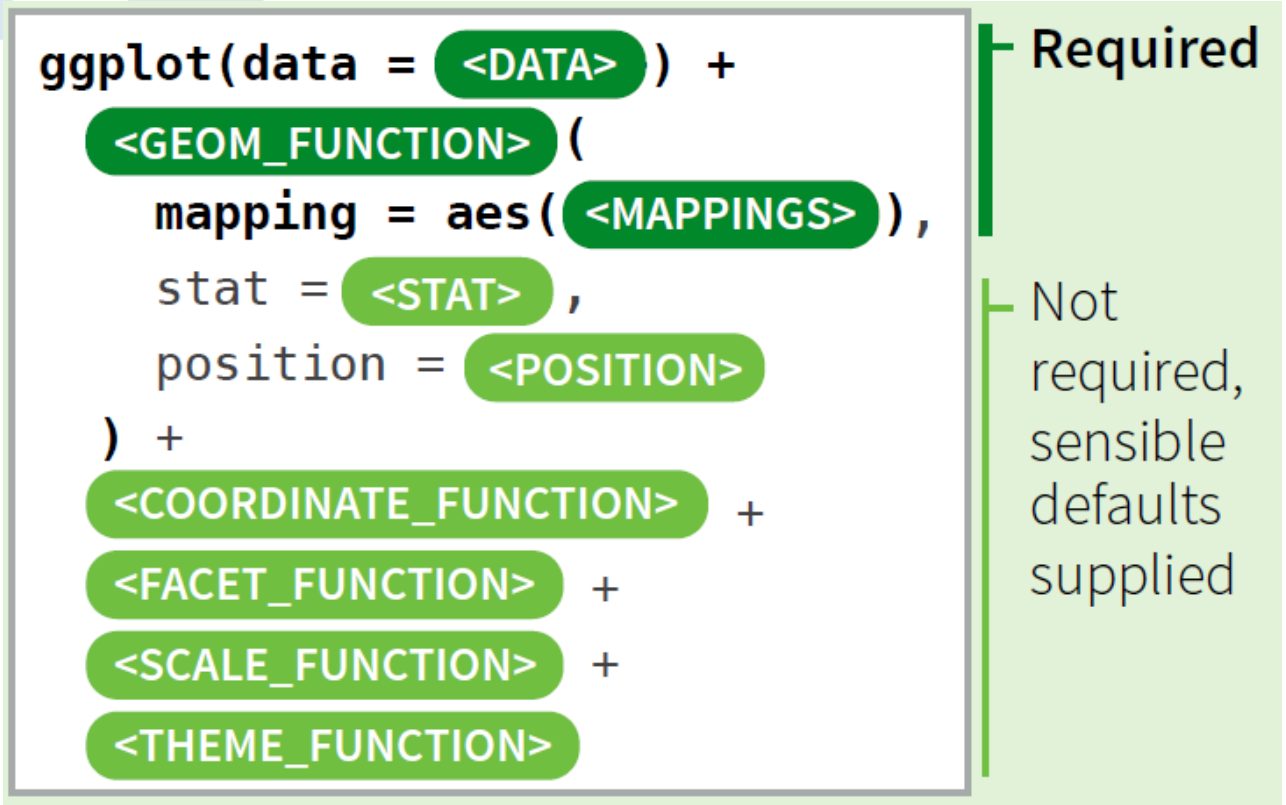
Ggplot2 개요와 기초

- ggplot2의 시각화의 3가지 필수요소
  - 데이터
    - 사용할 데이터의 이행
      - 변수의 척도는? 연속, 이산, 요인, 문자, 날짜
      - 데이터의 형태는? 횡단면, 시계열
  - 기하학적 함수
    - 그래프의 형태: geom() 또는 stat로 결정 및 수정
  - 시각적 매핑:
    - 미학(asesthetics) aes(……)
    - 매핑(mapping): x, y, z 변수와 시각적 특성(color, fill, shape, size, alpha 등)
      - 1차원: 한 변수
      - 2차원: 두 변수
      - 3차원: 세 변수
    - 스케일(scale): color, fill
      - scale_fill_*()
      - scale_*_munual()
      - scale_*_date()

출처: https://www.rstudio.com/wp-content/uploads/2016/11/ggplot2-cheatsheet-2.1.pdf

Ggplot2 개요와 기초

- ggplot2의 기초
  - 문법: ggplot() 함수의 구조



출처: https://www.rstudio.com/wp-content/uploads/2016/11/ggplot2-cheatsheet-2.1.pdf

Ggplot2 개요와 기초

- Ggplot2 패키지 설치와 불러오기
  - install.packages("ggplot2")
  - library(ggplot2)
  - ?ggplot2 # Create Elegant Data Visualisations Using the Grammar of Graphics
- 실습자료 불러오기와 데이터 생성
  - 아파트 실거래 가격 자료

```
> install.packages("ggplot2")
Error in install.packages : Updating loaded packages

Restarting R session...

Loading required package: sp
> library(ggplot2)

> setwd("E:/강의/수치해석/3주차_R_데이터마이닝_dplyr패키지") # 작업 경로(폴더) 설정
> apt <- read.csv('데이터_아파트매매가격.csv') # 실습데이터 불러오기(아파트 실거래 자료)
> str(apt) # 데이터 구조 확인하기
'data.frame':   13309 obs. of  14 variables:
 $ id             : int  1 2 3 4 5 6 7 8 9 10 ...
 $ apt_price      : int  4800 4500 4000 4000 4000 4000 4000 4000 4000 4800 ...
 $ area_m2        : num  39.8 39.8 39.8 39.8 39.8 39.8 39.8 39.8 39.8 39.8 ...
 $ floor_no       : int  2 6 7 9 3 7 5 9 8 5 ...
 $ year_built     : int  1998 1998 1998 1998 1998 1998 1998 1998 1998 1998 ...
 $ ym_sale        : int  201811 201811 201812 201812 201812 201812 201812 201812 201812
 $ day_sale       : int  6 13 10 10 10 10 10 10 10 10 ...
 $ urban          : Factor w/ 3 levels "동","면","읍": 3 3 3 3 3 3 3 3 3 3 ...
 $ apt_complex    : Factor w/ 802 levels "(1028-0)","(142-1)",..: 167 167 167 167 167 1
 $ address        : Factor w/ 181 levels "충청북도 괴산군 괴산읍 대사리",..: 1 1 1 1 1 1 1
 $ address_bunji  : Factor w/ 873 levels "1","1-1","1-16",..: 216 216 216 216 216 216 2
 $ address_sido   : Factor w/ 1 level "충청북도": 1 1 1 1 1 1 1 1 1 1 ...
 $ address_sigungu: Factor w/ 14 levels "괴산군","단양군",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ address_dong   : Factor w/ 107 levels "가경동","감곡면",..: 9 9 9 9 9 9 9 9 9 9 ...
> apt$price_pyung = apt$apt_price / apt$area_m2 * 3.3 # 평당 아파트 거래가격(만원) 생성하기
> str(apt$price_pyung) # 생성된 아파트 평당 가격 구조 확인
 num [1:13309] 398 373 332 332 332 ...
```

참고: 패키지 설치 및 로딩시의 다음의 오류 메시지 발생시 대처방법

Error in fetch(key) : lazy-load database '.....descopl.rdb' is corrupt

1. restarting your R session을 시도
2. remove.packages ()를 사용하여 제거 후 재설치
3. R 프로그램 제거 후 R 재설치

ggplot으로 그래프 작성하기: geom()

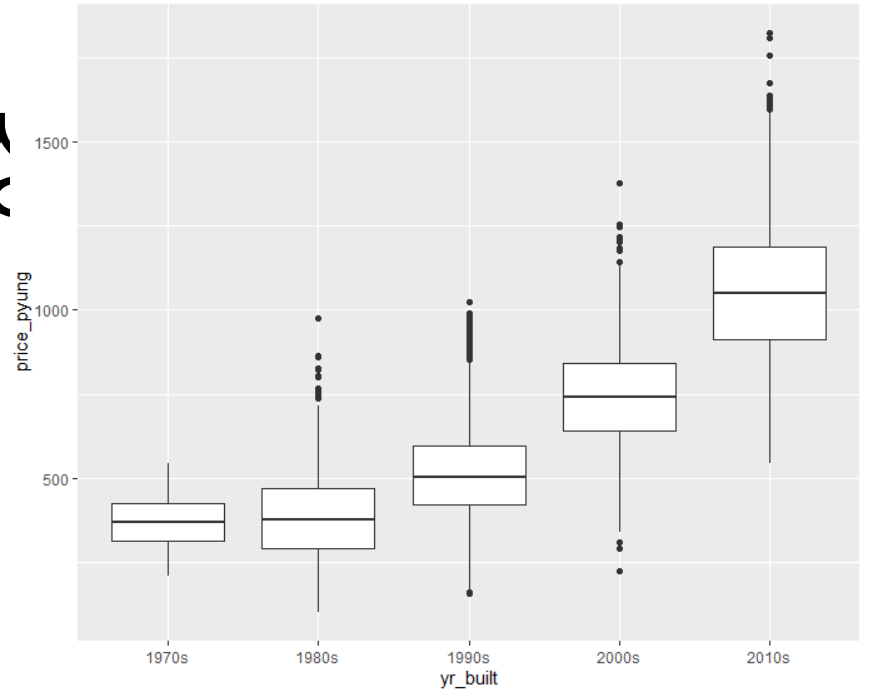
- geoms() 함수
  - 데이터 값(점)들을 표현하기 위하여 geom() 함수를 활용
    - geom_boxplot: 박스 그래프
    - geom_bar: 막대 그래프
    - geom_point: 산점도
    - geom_line: 라인 그래프
    - geom_count: 카운트 비교
    - geom_label: 그래프의 라벨을 텍스트로 표현
    - geom_errorbar: 신뢰구간(잘 사용하지 않음)
  - Position 매개변수를 통해 그래프를 조정
    - stack: 막대그래프의 디폴트 셋팅으로 아래 그래프에 쌓아서 그림
    - identity: 막대그래프를 그릴 때 count가 아니라 y값으로 그리고 싶을 때 사용
    - dodge: 막대그래프를 바로 옆에 붙여서 그림
    - fill: count가 아니라 백분율 기준
    - jitter: geom_point를 사용할 때 점을 좌우로 분리해 표현(같은 값들이 많을 경우 유용)
- 변수들을 표현하기 위하여 geom() 함수내에서 미학적 속성들로 표현
  - Geom_함수를 사용하기 전에 보통 aes함수를 이용하여 x축, y축, data를 매핑
  - geom함수 안에서도 매핑 가능
    - 2가지의 이상의 그래프를 한 화면에 그릴 때 유용
- 각 함수는 각각의 레이어로 작성됨
- dplyr 패키지 함수들과 함께 사용 가능
  - 실습

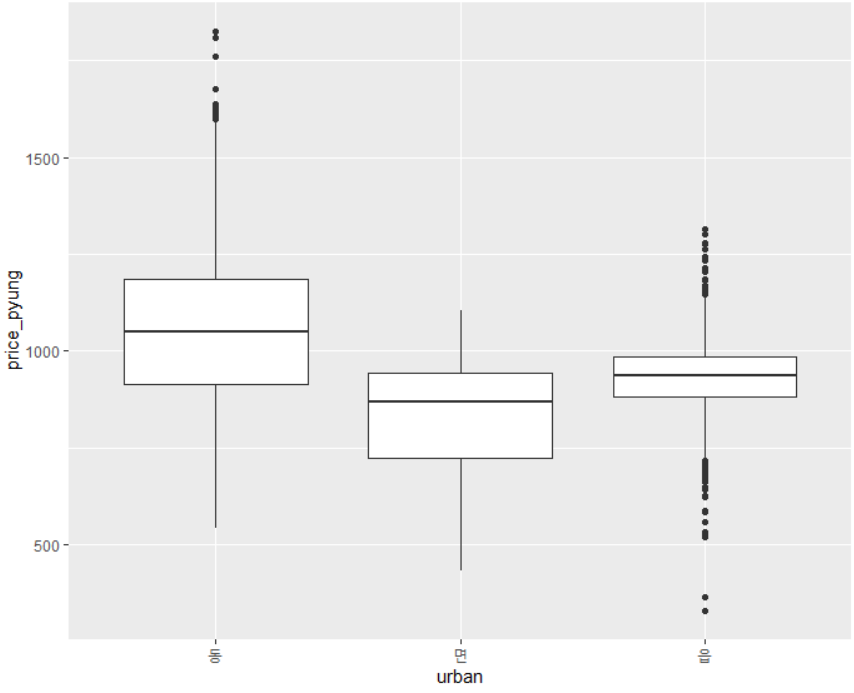
ggplot2 패키지와 데이터 시각화

Geom() 함수로 그래프 작성

- dplyr 패키지 함수들과 함께 사용 가능
  - 실습
    - 동지역에서의 건축년도별 평당 가격 박스그래프
    - 2010년대 건축된 아파트의 읍면동별 거래가격 박스그래프

```
> apt %>% # 사용할 데이터 셋
+   filter(urban=="동") %>% # 데이터 추출
+   ggplot(., aes(x=yr_built, y=price_pyung)) + # ggplot 그래프 셋팅
+      geom_boxplot() # 박스 그래프 생성
> apt %>% # 사용할 데이터 셋
+   filter(yr_built =="2010s") %>% # 데이터 추출
+   ggplot(., aes(x=urban, y=price_pyung)) + # ggplot 그래프 셋팅
+      geom_boxplot() # 박스 그래프 생성
```

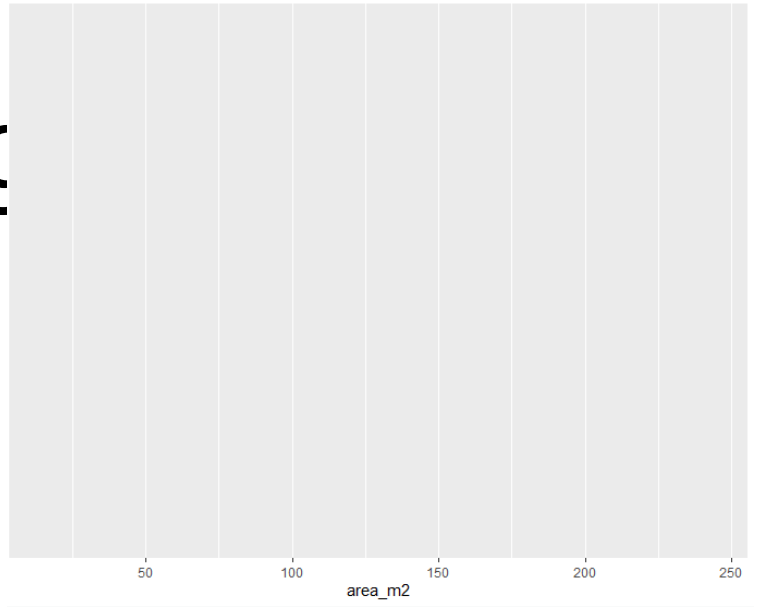


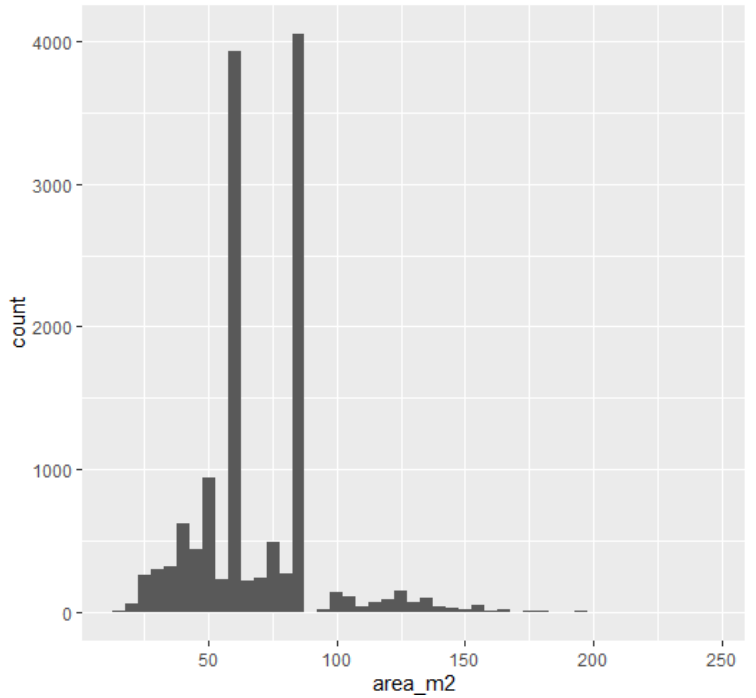


ggplot2 패키지와 데이터 시각화

ggplot으로 그래프 작성하기: g

- 한 개 변수(One Variable) 그래프 작성
  - x= 연속 변수(명목, 요인, 문자 등)
    - (a <- ggplot(data = apt, aes(x = area_m2))) # 전용면적 그래프 셋팅(x, y 좌표계)
    - ?geom_area # 리본 또는 면적 그래프(Ribbons and area plots)
      - a + geom_area(stat = "bin") # 면적 그래프
    - ?geom_density # 완만한 확률밀도 추정함수 그래프(Smoothed density estimates)
      - a + geom_density(kernel = "gaussian") # 커널밀도 함수 그래프
    - ?geom_dotplot # 점 그래프: a dot plot, the width of a dot corresponds to the bin width
      - a + geom_dotplot(binwidth = 5) # 5의 막대간격인 점 그래프
    - ?geom_freqpoly # 빈도폴리곤 그래프(frequency polygons)
      - a + geom_freqpoly(binwidth = 5) # 5의 간격인 빈도 폴리곤 그래프
    - ?geom_histogram # Histograms
      - a + geom_histogram(binwidth = 5) # 히스토 그램

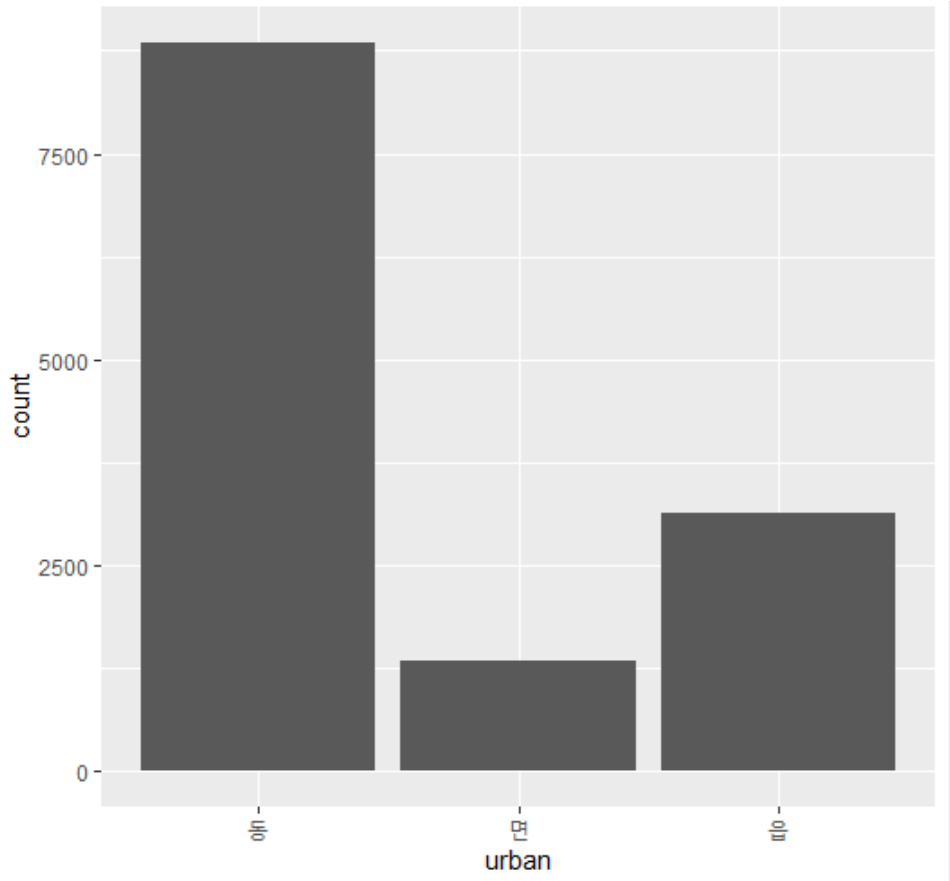




ggplot으로 그래프 작성하기: geom()

- 한 개 변수(One Variable) 그래프 작성
  - x= 이산 변수 (명목, 요인, 문자 등)

```
> ##### 이산변수
> b <- ggplot(apt, aes(urban)) # apt 자료의 urban 명목변수 그래프 셋팅
> ?geom_bar # Bar charts
> b + geom_bar() # 막대 그래프
```



ggplot으로 그래프 작성하기: geom()

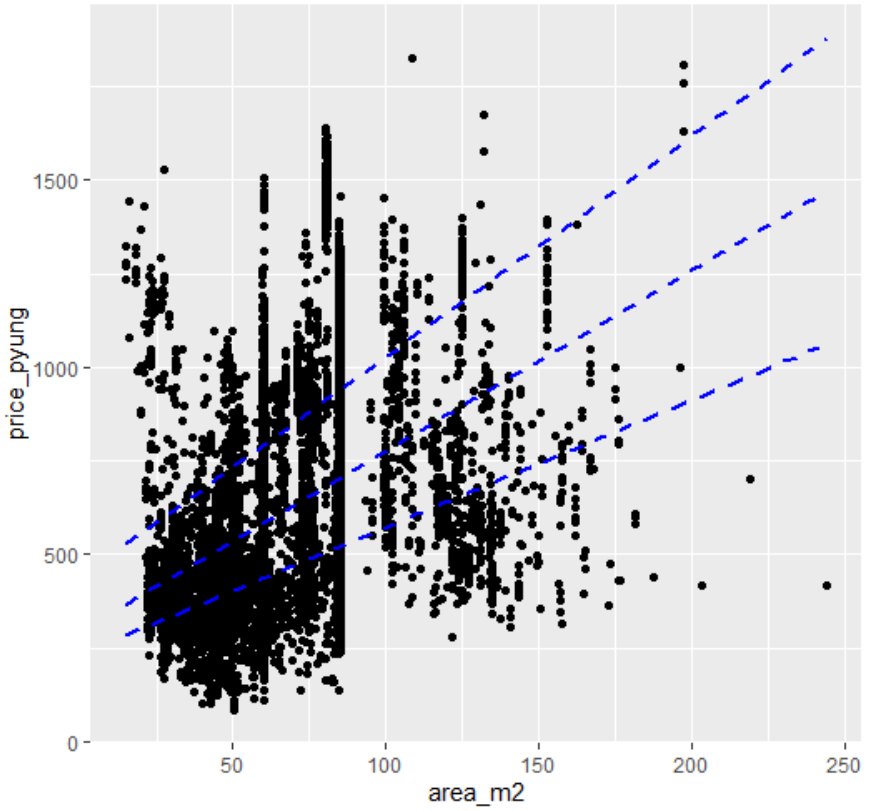
- 두 개 변수(Two Variables)
  - x= 연속변수, y=연속변수
    - ggplot( )
      - X, y 좌표계 그래프 셋팅
    - + geom_blank()
      - Draw nothing, a useful way of ensuring common scales between different plots
    - + geom_point()
      - Points: scatterplots
    - + geom_jitter
      - 중첩되지 않는 점 그래프(Jittered points)
      - a convenient shortcut for geom_point(position = "jitter").
      - adds a small amount of random variation to the location of each point,
      - and is a useful way of handling overplotting caused by discreteness in smaller datasets.
    - + geom_quantile()
      - Quantile regression 회귀직선
      - fits a quantile regression to the data and draws the fitted quantiles with lines.
      - a continuous analogue to geom_boxplot()
    - + geom_rug
      - Rug plots in the margins
      - a compact visualisation designed to supplement a 2d display with the two 1d marginal distributions.
      - Rug plots display individual cases so are best used with smaller datasets
    - + geom_smooth
      - Smoothed conditional means: Aids the eye in seeing patterns in the presence of overplotting
    - + geom_text
      - adds text directly to the plot.
    - + geom_label()
      - draws a rectangle behind the text, making it easier to read.

- +로 계속해서 추가 그래프 중첩하여 작성 가능

ggplot으로 그래프 작성하기: geom()

- 두 개 변수(Two Variables)
  - x= 연속변수, y=연속변수
    - 실습1

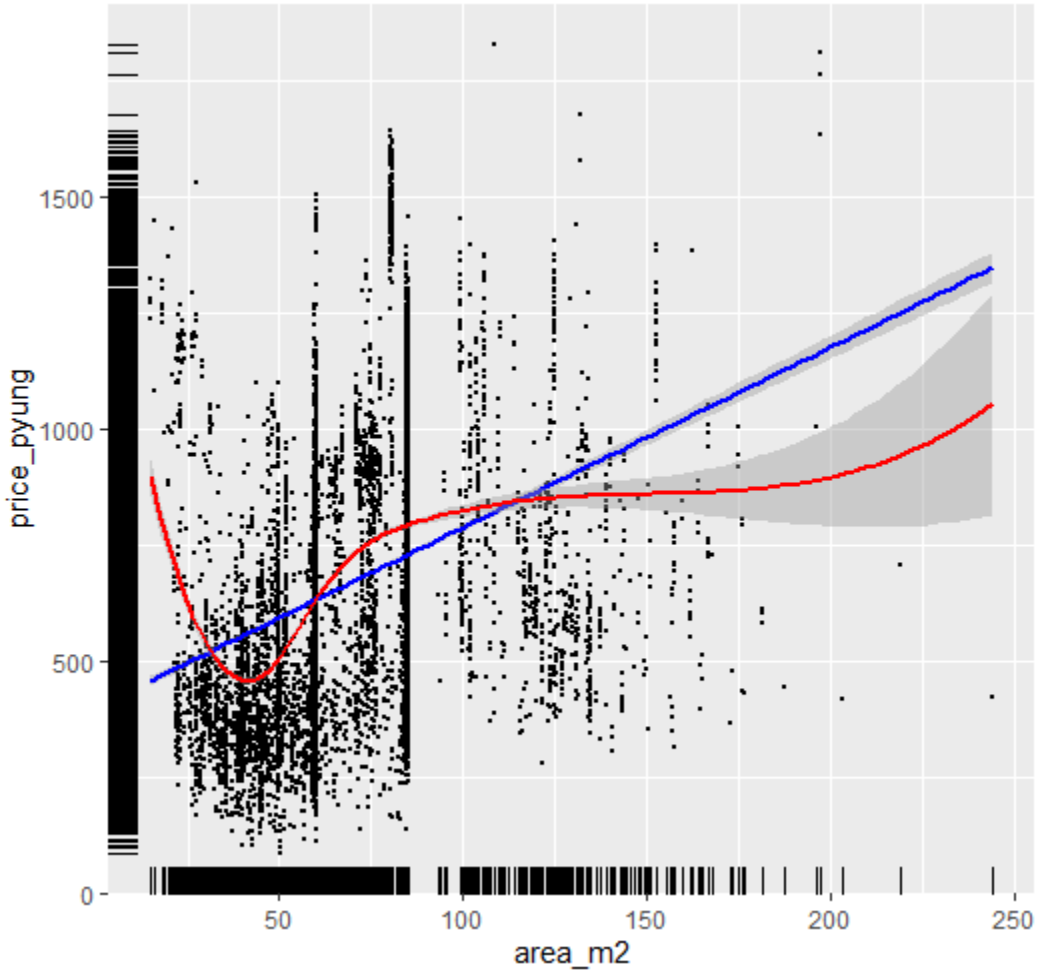
```
> c <- ggplot(apt, # 아파트 거래가격 데이터
+             aes(area_m2, price_pyung)) # 그래프 셋팅(x = 전용면적, y= 평당 아파트 가격)
> ?geom_blank() # Draw nothing, a useful way of ensuring common scales between different plots
> c + geom_blank() # 빈 그래프: x, y의 스케일 확인 방법
> c <- ggplot(apt, # 아파트 거래가격 데이터
+             aes(area_m2, price_pyung)) # 그래프 셋팅(x = 전용면적, y= 평당 아파트 가격)
> ?geom_blank() # Draw nothing, a useful way of ensuring common scales between different plots
> c + geom_blank() # 빈 그래프: x, y의 스케일 확인 방법
> ?geom_point() # Points: scatterplots
> c + geom_point() # 산점도 그래프
> ?geom_jitter # 중첩되지 않는 점 그래프(Jittered points)
> c + geom_jitter(cex=0.3) # 중첩되지 않는 점 그래프
> install.packages("quantreg")
Error in install.packages : Updating loaded packages
> library(`quantreg`)
> ?geom_quantile() # Quantile regression
> c + geom_quantile() # 분위별 회귀계수: quantiles = c(0.25, 0.5, 0.75)
Smoothing formula not specified. Using: y ~ x
> c + geom_quantile(quantiles = c(0.05, 0.5, 0.95)) # y ~ x
Smoothing formula not specified. Using: y ~ x
> c + geom_point() + geom_quantile(colour = "blue", size = 1, lty=2)
Smoothing formula not specified. Using: y ~ x
```



ggplot으로 그래프 작성하기: geom()

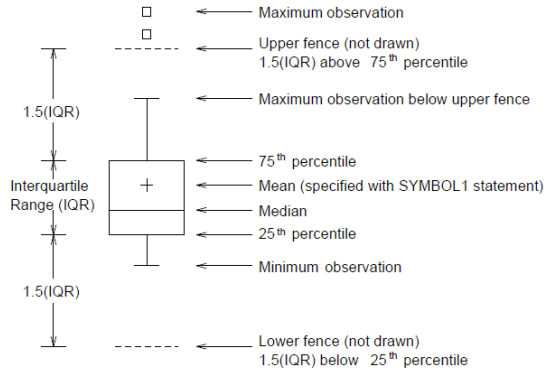
- 두 개 변수(Two Variables)
  - x= 연속변수, y=연속변수
    - 실습2

```
> c + geom_rug() + # 여백에 구간 분포 추가
+     geom_jitter(cex=0.3) + # 중첩되지 않는 점(jitters) 그래프 추가
+     geom_smooth(method = "lm", colour = "blue") + # 매끄러운 일반화 회귀 그래프 추가
+     geom_smooth(method = "loess", colour = "red") # loess 회귀 그래프 중첩
> ?geom_rug # Rug plots in the margins
> #*** a compact visualisation designed to supplement a 2d display with the two 1d ma
> #*** Rug plots display individual cases so are best used with smaller datasets
> c + geom_rug()  # 여백에 구획분포 그래프 생성
> c + geom_rug() +
+   geom_point(cex=0.2, colour = "gray") + # 구획 그래프와 점(산포도) 그래프 중첩
+   geom_quantile(quantiles = c(0.25, 0.5, 0.75)) # y ~ x 중첩
Smoothing formula not specified. Using: y ~ x
> ?geom_smooth # Smoothed conditional means: Aids the eye in seeing patterns in the p
> c + geom_smooth(method = "auto", colour = "green") # 자동선택: using method = 'lm',
`geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
> c + geom_smooth(method = "glm", colour = "yellow") # 매끄러운 선형회귀 그래프
> c + geom_point() +  # 산포도 & 매끄러운 회귀선 중첩
+   geom_smooth(method = "loess") # using method = 'loess'
> c + geom_rug() + # 여백에 구간 분포 추가
+     geom_jitter(cex=0.3) + # 중첩되지 않는 점(jitters) 그래프 추가
+     geom_smooth(method = "lm", colour = "blue") + # 매끄러운 일반화 회귀 그래프 추가
+     geom_smooth(method = "loess", colour = "red") # loess 회귀 그래프 중첩
> ?geom_text # adds text directly to the plot. geom_label() draws a rectangle behind
> c + geom_text(aes(label= urban)) # 값별 읍면동 라벨 부여
```



ggplot으로 그래프 작성하기: geom()

- 두 개 변수(Two Variables)
  - x= 이산변수, y=연속변수
    - ggplot()
    - + geom_bar
      - Bar charts
    - + geom_boxplot
      - box and whiskers plot
      - The boxplot compactly displays the distribution of a continuous variable.
      - It visualises five summary statistics (the median, two hinges and two whiskers), and all "outlying" points individually.

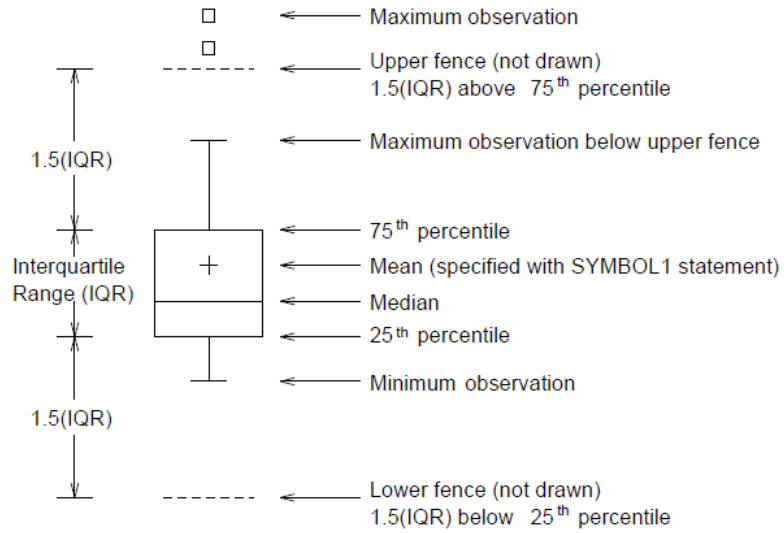


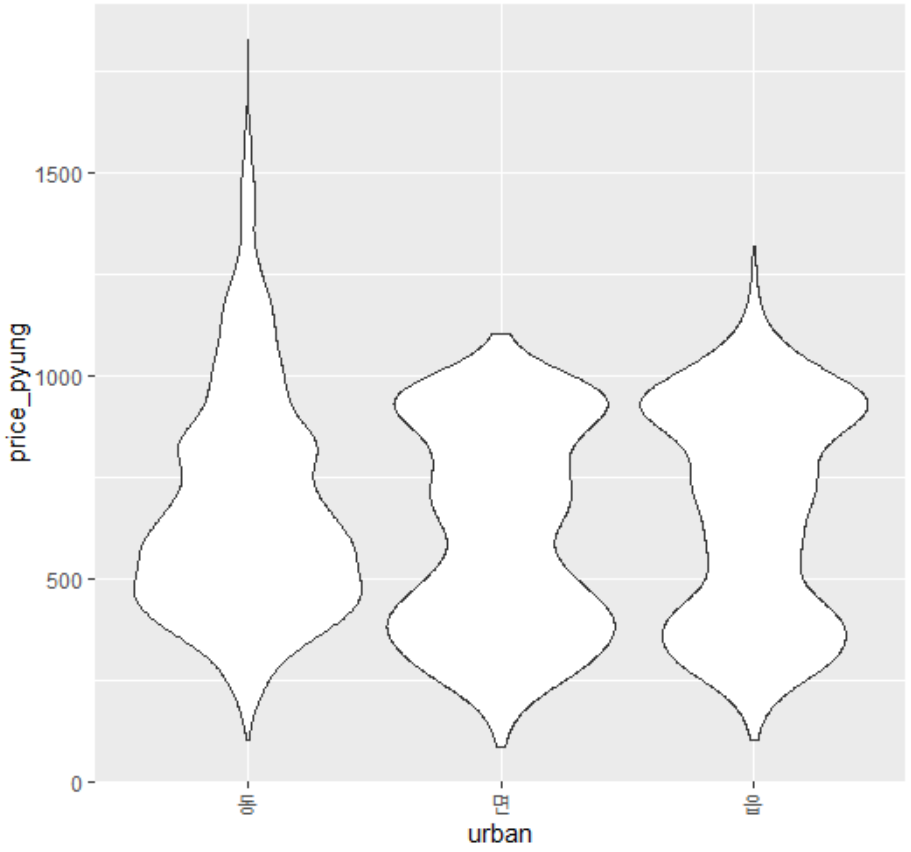
- + geom_dotplot()
  - Dot plot
    데이터가 클 경우에는 비추천
  - In a dot plot, the width of a dot corresponds to the bin width (or maximum width, depending on the binning algorithm)
  - and dots are stacked, with each dot representing one observation.
- + geom_violin
  - Violin plot, box plot과 유사하지만 분포의 형태 보다 시각적으로 표현

ggplot으로 그래프 작성하기

- 두 개 변수(Two Variables)
  - x= 이산변수, y=연속변수
    - 실습

```
> ##### x = 이산변수, y=연속변수
> d <- ggplot(apt, # 아파트 실거래 가격 자료
+             aes(urban, price_pyung)) # 미학 그래프 셋팅(x = 읍면동, y= 평당 아파트 가격)
> ?geom_bar # Bar charts
> d + geom_bar(stat="identity") # 막대 그리프(y=읍면동별 거래건수)
> ?geom_boxplot # box and whiskers plot
> #*** The boxplot compactly displays the distribution of a continuous variable.
> #*** It visualises five summary statistics (the median, two hinges and two whiskers)
> d + geom_boxplot()
> ?geom_dotplot() # Dot plot(데이터가 클 경우에는 비추천)
> d + geom_dotplot(binaxis = "y", # The axis to bin along, "x" (default) or "y"
+                  stackgroups = TRUE, # should dots be stacked across groups?
+                  method =  "dotdensity", #  "dotdensity" 또는 "histodot"
+                  binwidth = 4, # Defaults to 1/30 of the range of the data
+                  binpositions = "all") # "all" determines positions of the bins with
> ?geom_violin # Violin plot, bot plot과 유사하지만 분포의 형태 보다 시각적으로 표현
> d + geom_violin(scale="area") # (default), all violins have the same area (before tr
> d + geom_violin(scale="count") # areas are scaled proportionally to the number of obs
> d + geom_violin(scale="width") # all violins have the same maximum width.
```

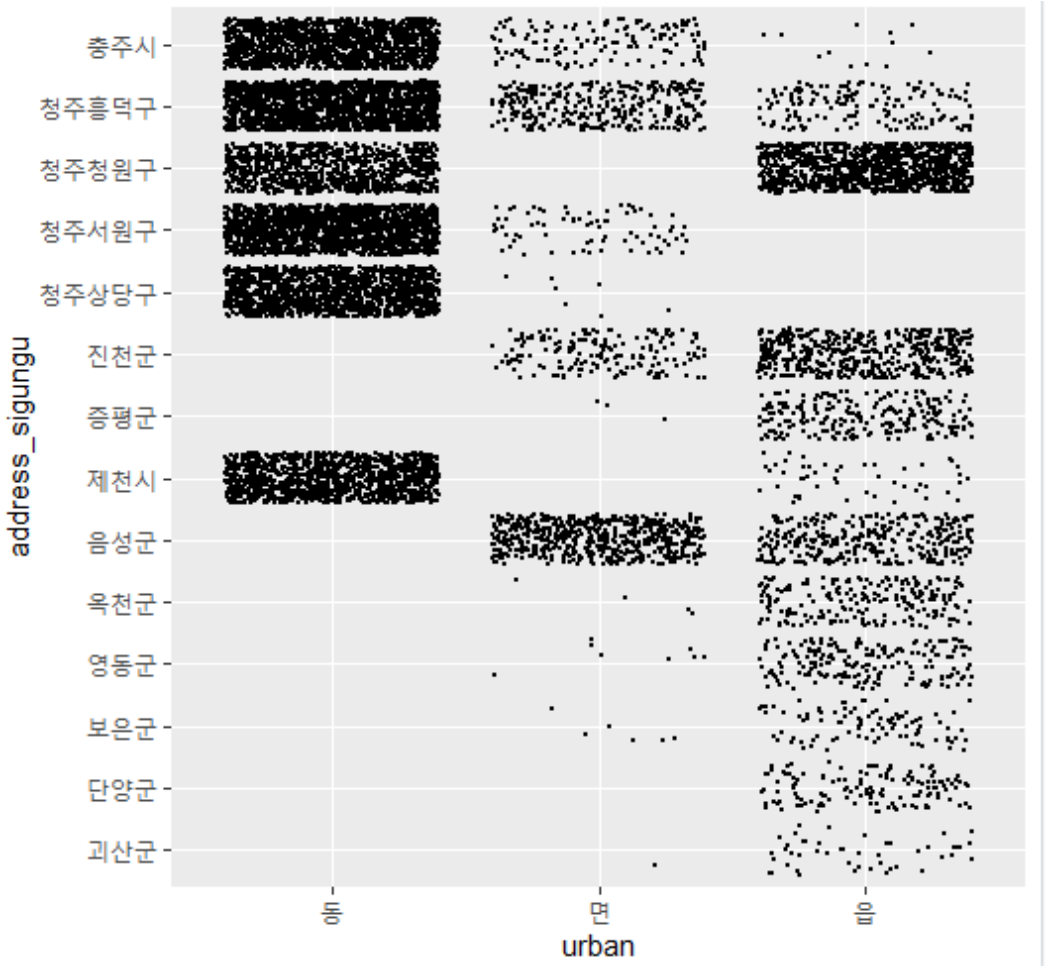




ggplot으로 그래프 작성하기: geom()

- 두 개 변수(Two Variables)
  - x= 이산변수, y=이산변수
    - 실습

```
> ##### 두 개의 이산변수들(x, y)
> ggplot(apt, # 아파트 거래 자료
+        aes(x=urban, # 읍면동
+            y=address_sigungu)) # 시군구
> e <- ggplot(apt, aes(x=urban, y=address_sigungu)) # e에 할당
> e + geom_jitter(cex=0.2) # 시군구별 읍면동별 jitter 그래프(점의 크기 = 0.2배)
```



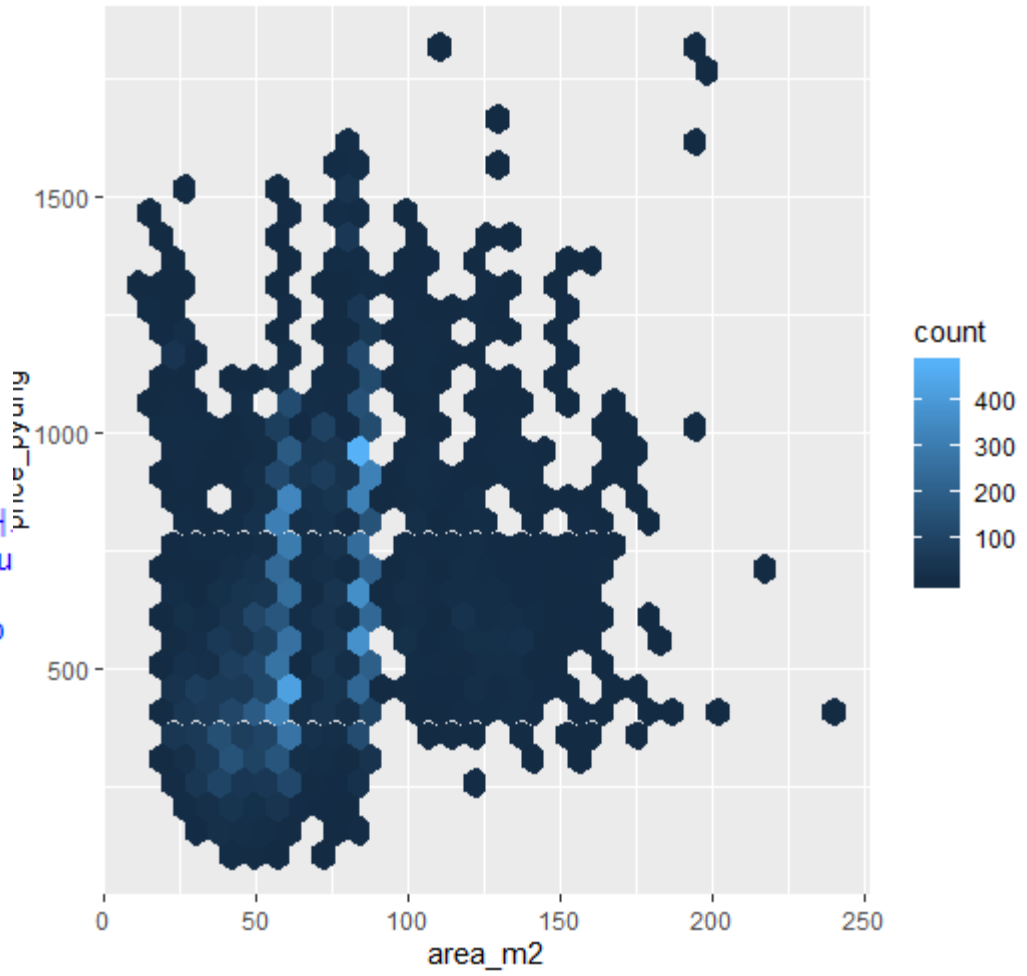
ggplot으로 그래프 작성하기: geom()

- 두 개 변수(Two Variables)
  - 지도(Maps)형태의 그래프
    - ggplot()
    - + geom_bin2d
      - 빈도를 추가한 사각형 셀 적외선 열지도(Heatmap of 2d bin counts)
    - + geom_density2d()
      - 커널밀도 함수 추정치의 등고선 그래프(맵)
      - Contours of a 2d density estimate: Perform a 2D kernel density estimation
    - + geom_hex
      - Hexagonal heatmap of 2d bin counts
      - 빈도를 추가한 육각형 셀 적외선 열지도(Heatmap of 2d bin counts)

ggplot으로 그래프 작성하기: geom()

- 두 개 변수(Two Variables)
  - 지도(Maps)형태의 그래프
    - 실습

```
> ##### 이변량 연속분포 그래프
> ggplot(apt, # 아파트 거래자료
+        aes(area_m2, # 전용면적
+            price_pyung)) # 평당 가격
> f <- ggplot(apt, aes(area_m2, price_pyung)) # 전용면적과 평당 가격 그래
> ?geom_bin2d # 빈도를 추가한 사각형 셀 적외선 열지도(Heatmap of 2d bin cou
> f + geom_bin2d()
> ?geom_density2d() # Contours of a 2d density estimate: Perform a 2D
> f + geom_density2d() # 커널밀도 함수 추정치의 등고선 그래프(맵)
> library(hexbin)
> ?geom_hex # exagonal heatmap of 2d bin counts: 빈도를 추가한 육각형 셀
> f + geom_hex()
```



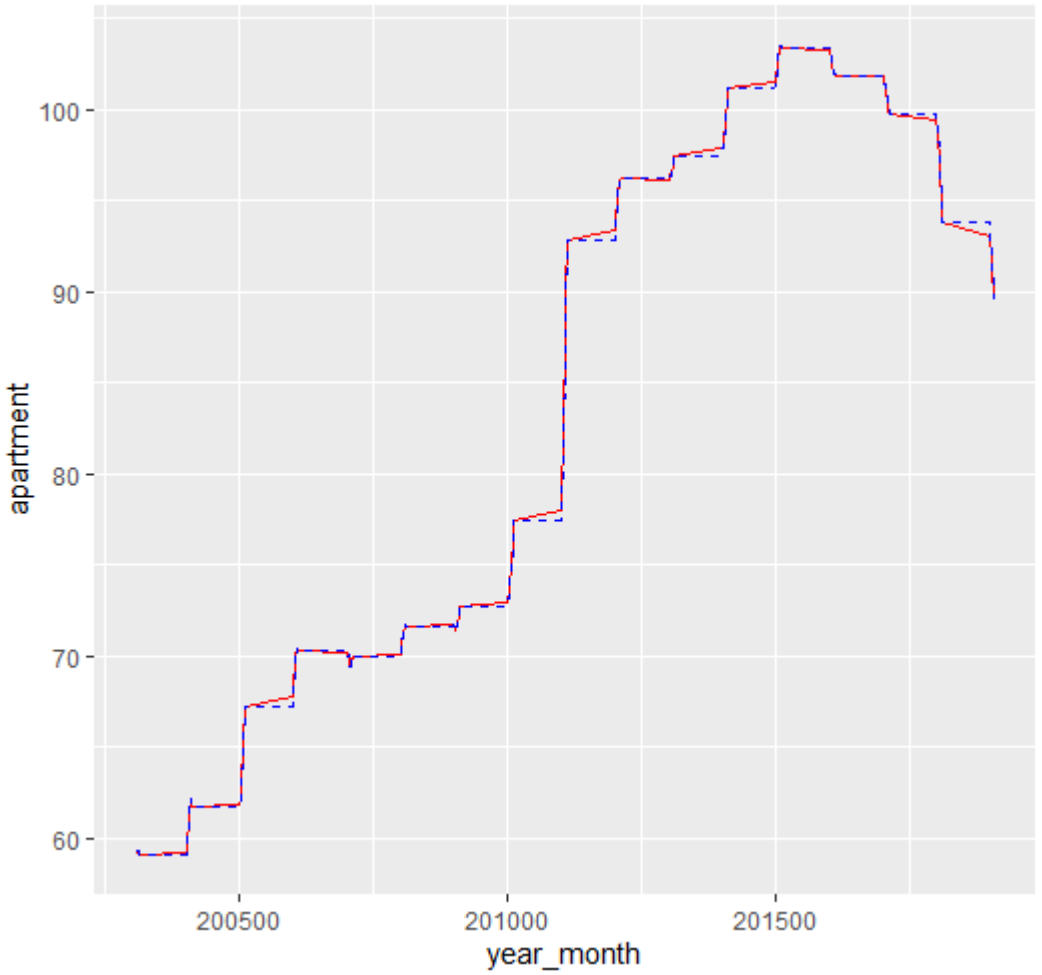
ggplot으로 그래프 작성하기: geom()

- 두 개 변수(Two Variables)
  - 연속 이변량 분포(Continuous Bivariate Distribution)
    - 시계열 그래프에 보다 적합
  - 실습 데이터 불러오기
    - 데이터 불러오기
      - 주택유형별_월별_충북지역_매매가격지수.csv
    - 월별 매매가격지수 활용
      - 2017년 11월 = 100(불변가격)
  - ggplot()
    - + geom_area
      - 면적
      - a y interval defined by ymin and yma
    - + geom_line()
      - Connect observations
    - + geom_step()
      - creates a stairstep plot, highlighting exactly when changes occur.

ggplot으로 그래프 작성하기: geom()

- 두 개 변수(Two Variables)
  - 실습

```
> # 기본 그래프 셋팅
> setwd("E:\\강의\\수치해석\\4주차_R_정형데이터_시각화와_ggplot패키지") # 작업 경로(
> m_housing_index <- read.csv('주택유형별_월별_충북지역_매매가격지수.csv') # 실습더
> str(m_housing_index) # 데이터 구조 확인하기: 2017년 11월 = 100(불변가격)
'data.frame':   189 obs. of  6 variables:
 $ year_month: int  200311 200312 200401 200402 200403 200404 200405 200406
 $ year      : int  2003 2003 2004 2004 2004 2004 2004 2004 2004 2004 ...
 $ month     : int  11 12 1 2 3 4 5 6 7 8 ...
 $ apartment : num  59.3 59.1 59.2 59.5 59.8 60.2 60.4 60.9 61.3 61.7 ...
 $ mult_f_h  : num  76.4 74.3 74.3 73 72.9 72.9 72.9 72.4 72.4 72.7 ...
 $ single_f_h: num  95.3 92.4 91.1 90.9 90.2 87.9 87.5 87.4 88.4 88.7 ...
> g <- ggplot(m_housing_index, aes(year_month, apartment))
> ?geom_area # 면적 a y interval defined by ymin and yma
> g + geom_area()
> ?geom_line() # Connect observations
> g + geom_line() # connects them in order of the variable on the x axis
> ?geom_step() # creates a stairstep plot, highlighting exactly when changes
> g + geom_step(colour="blue", lty=2) # creates a stairstep plot, highlighti
> g + geom_line(colour="red") + # 선그래프
+     geom_step(colour="blue", lty=2) # creates a stairstep plot, highlighti
```



ggplot으로 그래프 작성하

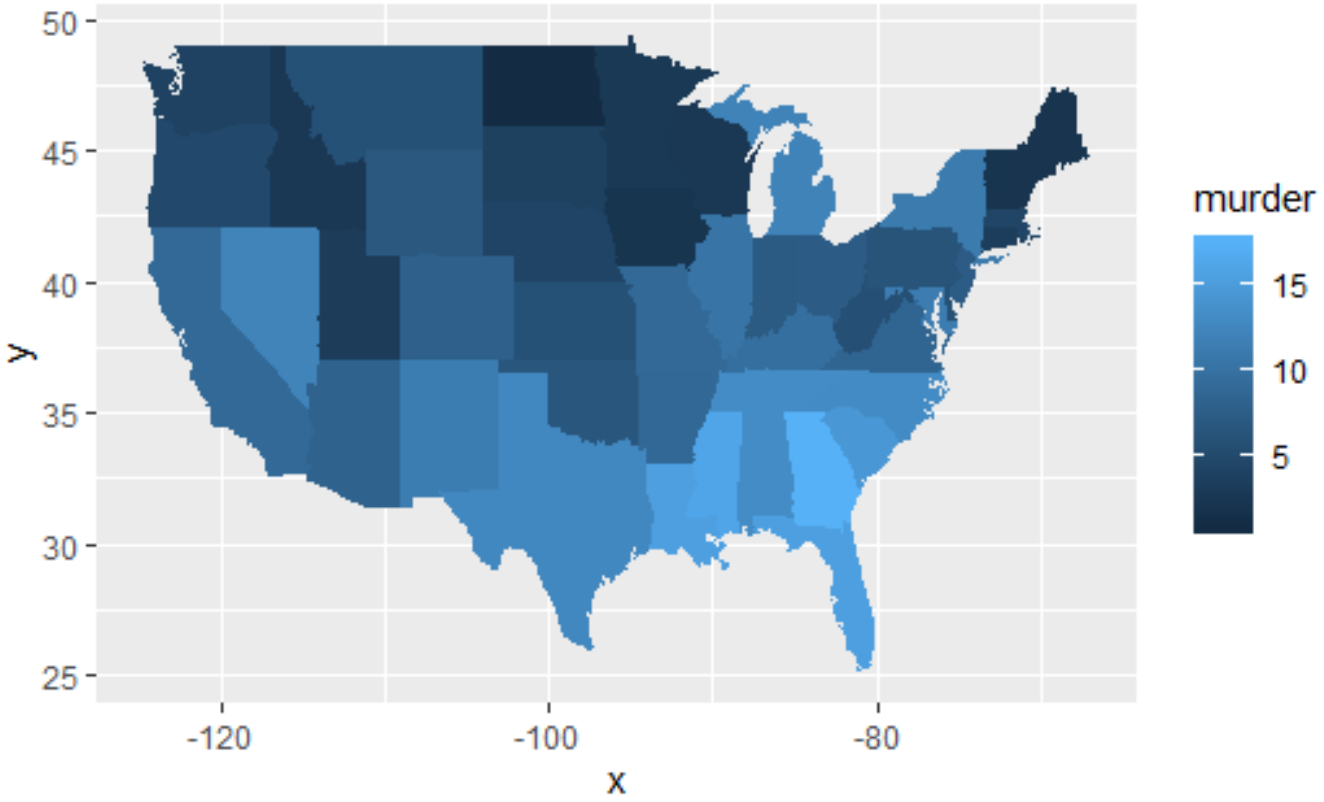
- 지도 그래프 작성
  - 공간자료와 데이터 프레임 활용 그래프 작성하기
    - 미국 범죄체포 자료와 미국 주별 경계 지도 활용
      - USArrests # Violent Crime Rates by US State
      - "state"맵
  - ggplot()
  - + geom_map
    - Polygons from a reference map
  - 실습
    - 데이터 불러와서 확인하기

```
> ##### 지도 그래프 작성
> ?USArrests # Violent Crime Rates by US State
> str(USArrests) # 데이터 구조 확인하기
'data.frame':   50 obs. of  4 variables:
 $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 
 $ Assault : int  236 263 294 190 276 204 110 238 335
 $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
 $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15
> View(USArrests)
> data <- data.frame(murder = USArrests$Murder, # 미국
+                    state = tolower(rownames(USArres
> head(data)
  murder       state
1   13.2     alabama
2   10.0      alaska
3    8.1     arizona
4    8.8    arkansas
5    9.0  california
6    7.9    colorado
> install.packages('maps') # maps 패키지 설치

> library(maps)
> head(map_data("state")) # xy 좌표와 주명 지도
      long      lat group order  region subregion
1 -87.46201 30.38968     1     1 alabama      <NA>
2 -87.48493 30.37249     1     2 alabama      <NA>
3 -87.52503 30.37249     1     3 alabama      <NA>
4 -87.53076 30.33239     1     4 alabama      <NA>
5 -87.57087 30.32665     1     5 alabama      <NA>
6 -87.58806 30.32665     1     6 alabama      <NA>
> map <- map_data("state")
> h <- ggplot(data, # 미국 주별 살인건수 데이터
+             aes(fill = murder)) # 미학요소로 살인건수 
> library(maps)
> head(map_data("state")) # xy 좌표와 주명 지도
      long      lat group order  region subregion
1 -87.46201 30.38968     1     1 alabama      <NA>
2 -87.48493 30.37249     1     2 alabama      <NA>
3 -87.52503 30.37249     1     3 alabama      <NA>
4 -87.53076 30.33239     1     4 alabama      <NA>
5 -87.57087 30.32665     1     5 alabama      <NA>
6 -87.58806 30.32665     1     6 alabama      <NA>
> map <- map_data("state")
```

ggplot으로 그래프 작성하기: geom()

- 실습
  - 공간지도 작성하기

```
> h <- ggplot(data, # 미국 주별 살인건수 데이터
+             aes(fill = murder)) # 미학요소로 살인건수 빈도로 색상 분포
> ?geom_map # Polygons from a reference map
> h + geom_map(aes(map_id = state), # 미학요소로 map_id는 필수, 여기서는 state
+              map = map) + # Data frame that contains the map coordinates.
+   expand_limits(x = map$long, y = map$lat) # x축과 y축의 한계값까지 경계 확장
```



ggplot으로 그래프 작성하기: geom()

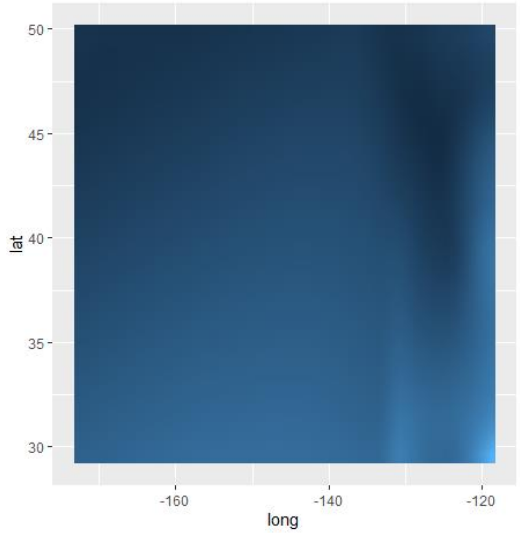
- 세가지 변수들: 공간(x, y)과 이동경로(방향, z)
  - 내장 데이터 불러오기
    - seals
      - Vector field of seal movements: the paths of moving animals
      - A data frame with 1155 rows and 4 variables
  - 데이터 가공하기
    - 이동거리 계산
  - ggplot()
    - + geom_contour()
      - 2d contours of a 3d surface
    - + geom_raster ()
      - Rectangles: a high performance special case for when all the tiles are the same size.
    - + geom_tile ()
      - uses the center of the tile and its size (x, y, width, height).
    - + geom_spoke ()
      - Line segments parameterised by location, direction and distance
      - It is useful when you have variables that describe direction and distance.
      - The angles start from east and increase counterclockwise.

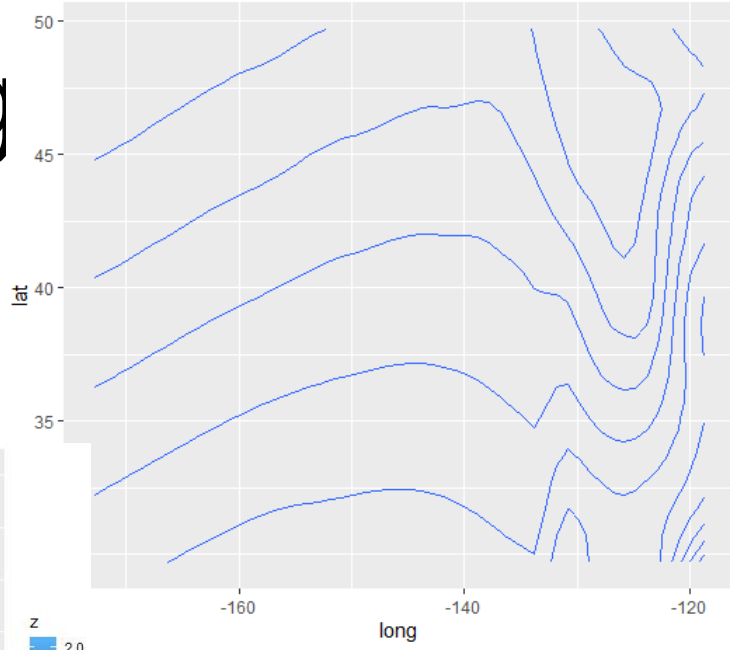
```
> ##### 세가지 변수늘 그래프
> ?seals # Vector field of seal movements: the paths of moving animals
> str(seals) # A data frame with 1155 rows and 4 variables
Classes 'tbl_df', 'tbl' and 'data.frame':       1155 obs. of  5 variables:
 $ lat       : num  29.7 30.7 31.7 32.7 33.7 34.7 35.7 36.7 37.7 38.7 ...
 $ long      : num  -173 -173 -173 -173 -173 ...
 $ delta_long: num  -0.915 -0.867 -0.819 -0.771 -0.723 ...
 $ delta_lat : num  0.1435 0.1284 0.1132 0.098 0.0828 ...
 $ z         : num  0.926 0.876 0.827 0.777 0.727 ...
> seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2)) # 동물 이동거리 계산
```

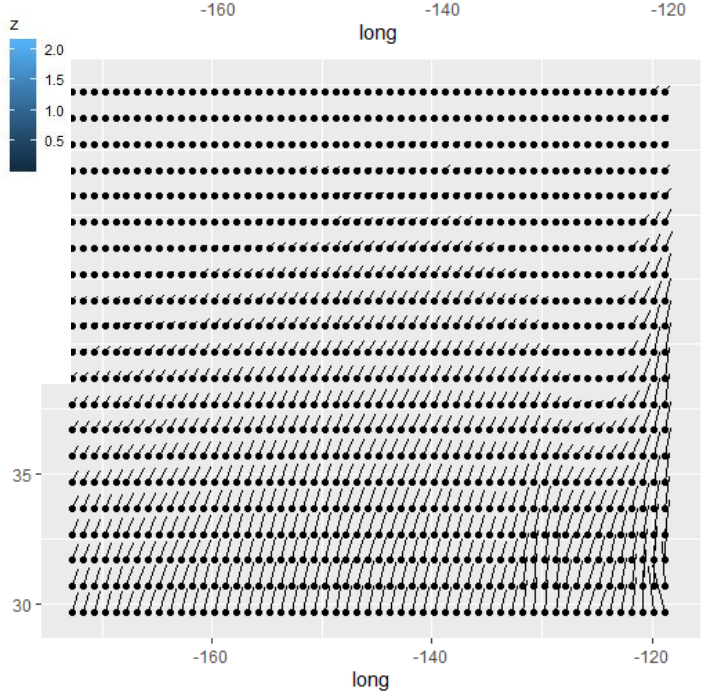
ggplot으로 그래프 작성하기: g

- 세가지 변수들: 공간(x, y)과 이동경로(방향, z)
  - 실습

```
> i <- ggplot(seals, # seals 데이터
+             aes(long, lat)) # x, y 좌표
> ?geom_contour # 2d contours of a 3d surface
> i + geom_contour(aes(z = z)) # z는 x, y 좌표별 한 개의 값일 경우 사용
> ?geom_raster # Rectangles: a high performance special case for whe
> i + geom_raster(aes(fill = z), hjust=0.5,
+                 vjust=0.5, interpolate=FALSE)
> i + geom_raster(aes(fill = z), hjust=0.5,
+                 vjust=0.5, interpolate=TRUE)
> ?geom_tile # uses the center of the tile and its size (x, y, width
> i + geom_tile(aes(fill = z))
> ?geom_spoke # Line segments parameterised by location, direction a
> #*** It is useful when you have variables that describe direction
> #*** The angles start from east and increase counterclockwise.
> i + geom_point() +
+     geom_spoke(angle= seals$z, radius=seals$z )
> i + geom_contour(aes(z = z)) # z는 x, y 좌표별 한 개의 값일 경우 사용
```







ggplot으로 그래프 작성하기: geom()

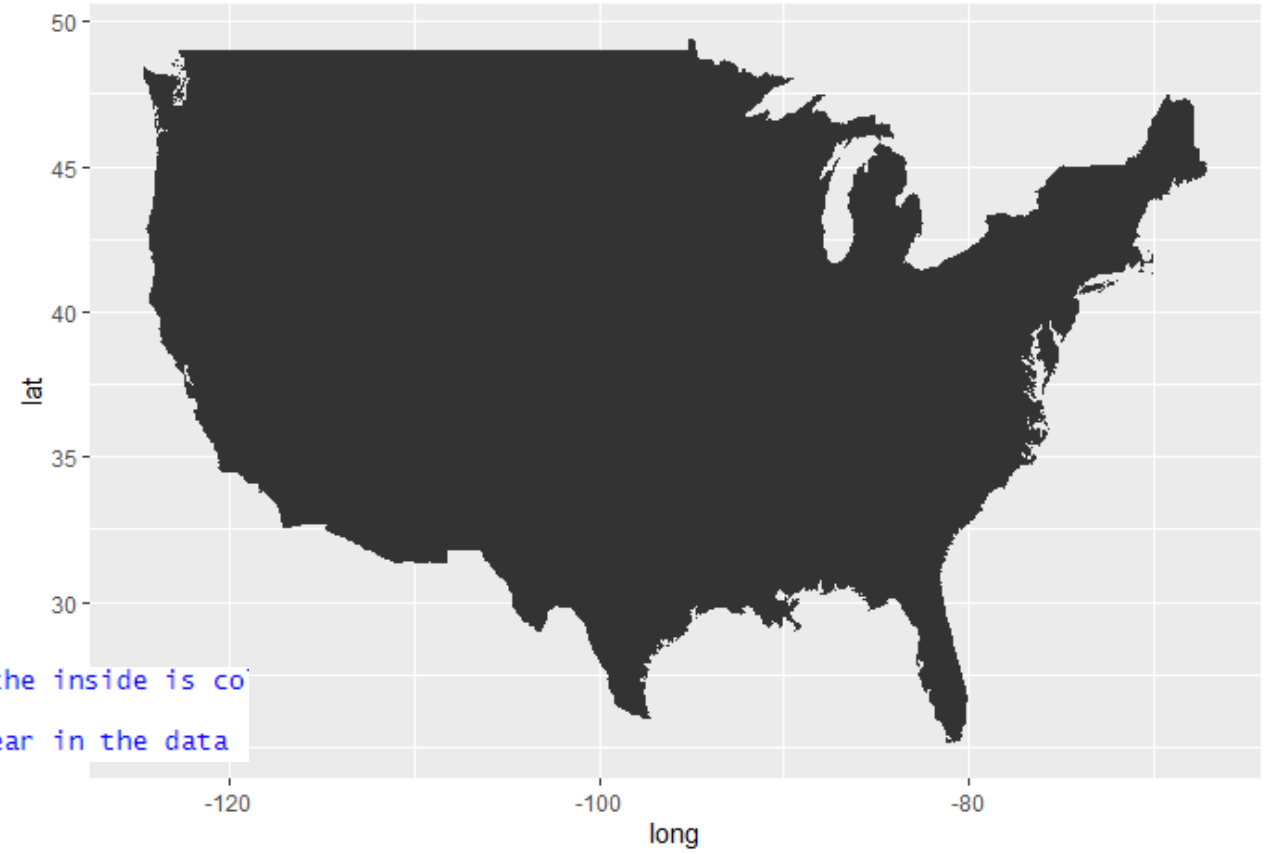
- 그래픽 요소들(Graphical Primitives)
  - ggplot()
  - + geom_polygon ()
    - A geom that draws a polygon
    - the start and end points are connected and the inside is coloured by fill
    - 인수들
      - group - (default: interaction of all categorical variables in the plot) how to group observations into polygons (each observation represents one point of a polygon)
      - **x** - (required) x-coordinate of the polygon's points
      - **y** - (required) y-coordinate of the polygon's points
      - size - (default: 0.5) line width of the polygon's outline
      - linetype - (default: 1=solid) line type of the polygon's outline
      - color - (default: NA=no outline) color of the polygon's outline
      - fill - (default: "grey20") fill color of the polygon
      - alpha - (default: 1=opaque) transparency of the polygon's fill
  - + geom_path ()
    - connects the observations in the order in which they appear in the data
  - + geom_ribbon ()
    - displays a y interval defined by ymin and ymax

ggplot으로 그래프 작성하기: geom()

- 그래픽 요소들(Graphical Primitives)
  - 실습1

```
> #### Graphical Primitives
> j <- ggplot(map, aes(long, lat))
> str(map)
'data.frame':   15537 obs. of  6 variables:
 $ long     : num  -87.5 -87.5 -87.5 -87.5 -87.6
 $ lat      : num  30.4 30.4 30.4 30.3 30.3 ...
 $ group    : num  1 1 1 1 1 1 1 1 1 1 ...
 $ order    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ region   : chr  "alabama" "alabama" "alabama"
 $ subregion: chr  NA NA NA NA ...
> summary(map)
      long             lat            group
 Min.   :-124.68   Min.   :25.13   Min.   : 1.00
 1st Qu.: -96.22   1st Qu.:33.91   1st Qu.:15.00
 Median : -87.61   Median :38.18   Median :26.00
 Mean   : -89.67   Mean   :38.18   Mean   :30.15
 3rd Qu.: -79.13   3rd Qu.:42.80   3rd Qu.:47.00
 Max.   : -67.01   Max.   :49.38   Max.   :63.00
```

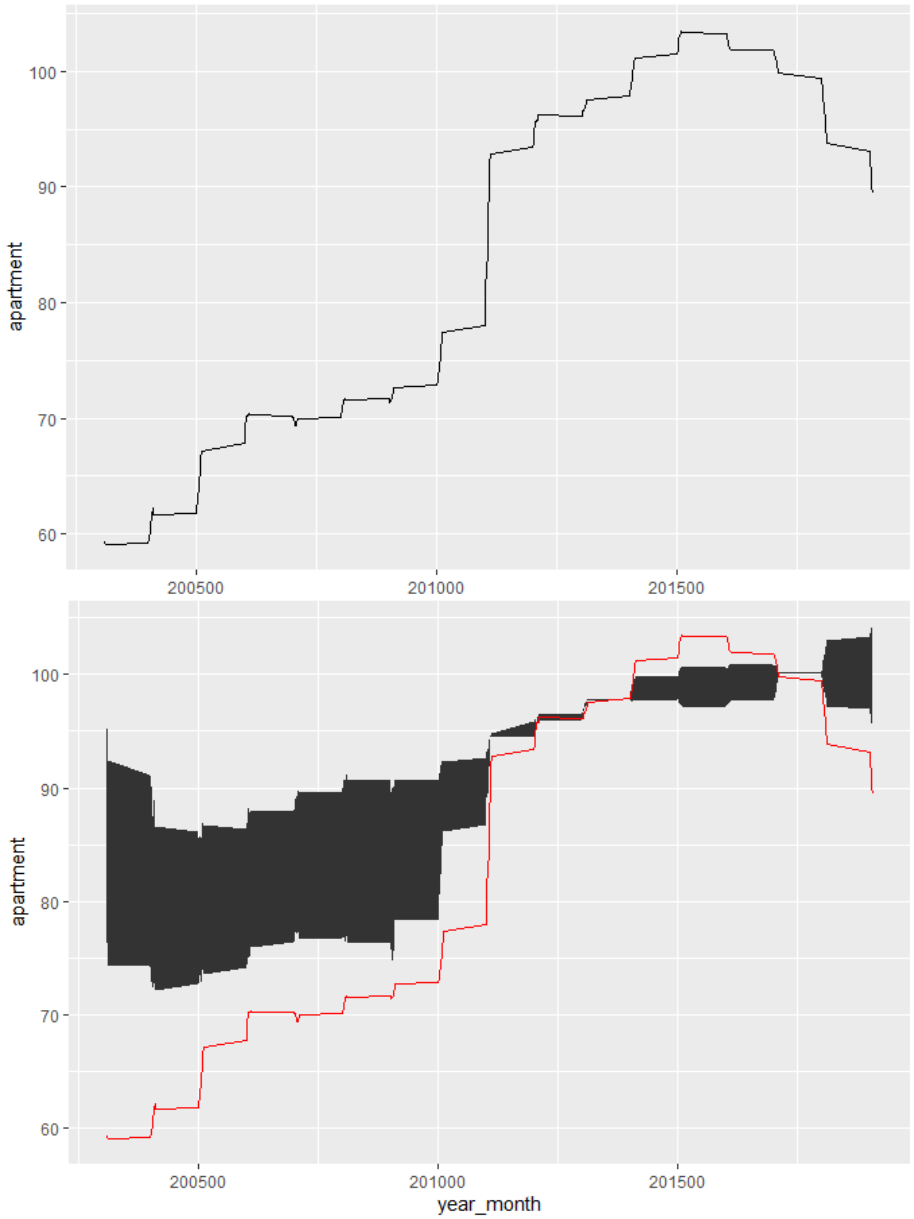
```
> ?geom_polygon # Polygons: the start and end points are connected and the inside is co
> j + geom_polygon(aes(group = group))
> ?geom_path # connects the observations in the order in which they appear in the data
```



ggplot으로 그래프 작성하기

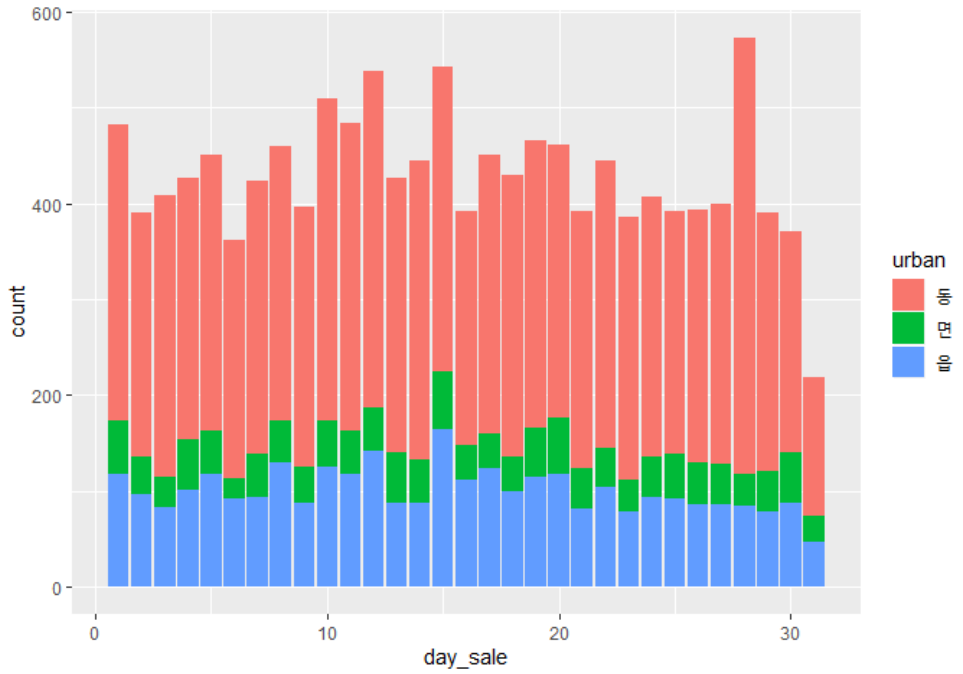
- 그래픽 요소들(Graphical Primitives)
  - 실습2

```
> str(m_housing_index)
'data.frame':   189 obs. of  6 variables:
 $ year_month: int  200311 200312 200401 200402 200403 200404 200405 200406 200407 2004
 $ year      : int  2003 2003 2004 2004 2004 2004 2004 2004 2004 2004 ...
 $ month     : int  11 12 1 2 3 4 5 6 7 8 ...
 $ apartment : num  59.3 59.1 59.2 59.5 59.8 60.2 60.4 60.9 61.3 61.7 ...
 $ mult_f_h  : num  76.4 74.3 74.3 73 72.9 72.9 72.9 72.4 72.4 72.7 ...
 $ single_f_h: num  95.3 92.4 91.1 90.9 90.2 87.9 87.5 87.4 88.4 88.7 ...
> k <- ggplot(m_housing_index, # 월별 매매 가격지수 데이터(객체)
+             aes(year_month, apartment)) # 년월과 아파트 매매거래지수 그래프 틀 작성
> ?geom_path # connects the observations in the order in which they appear in the data
> k + geom_path(lineend="butt", # Line end style (round, butt, square).
+               linejoin="round", # Line join style (round, mitre, bevel).
+               linemitre=1) # Line mitre limit (number greater than 1).: 1로 제한


> ?geom_ribbon # displays a y interval defined by ymin and ymax
> k + geom_ribbon(aes(ymin=apartment-5, ymax=apartment+5))
> k + geom_ribbon(aes(ymin=mult_f_h, ymax= single_f_h)) + # 리본과 선 그래프 겹쳐서 작성
+     geom_line(colour="red") # connects them in order of the variable on the x axis
```



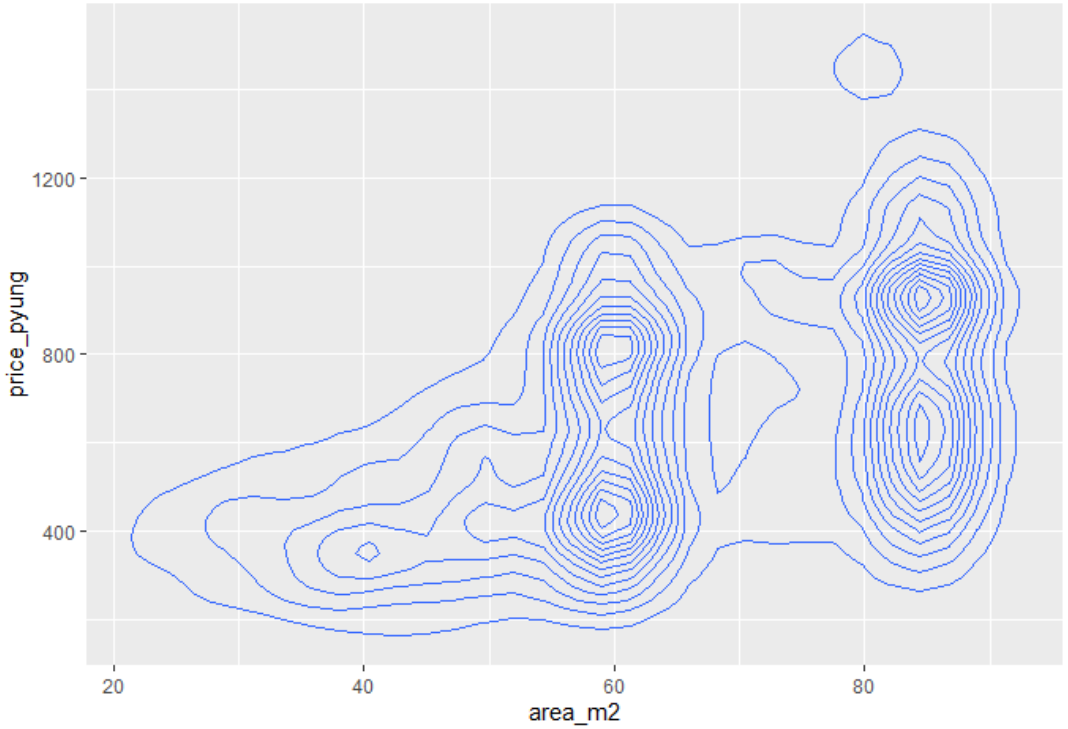
연습문제 07

- 아파트 실거래 자료(객체: apt) 중에서 거래일(day_sale)별 그래프 셋팅하고(틀을 만들고), 바그래프(geom_bar)를 만드는 데, 읍면동(urban) 거래건수를 쌓아서(stack), 아래와 같은 그래프를 작성하고자 한다.
- 적합한 명령문을 작성하시오



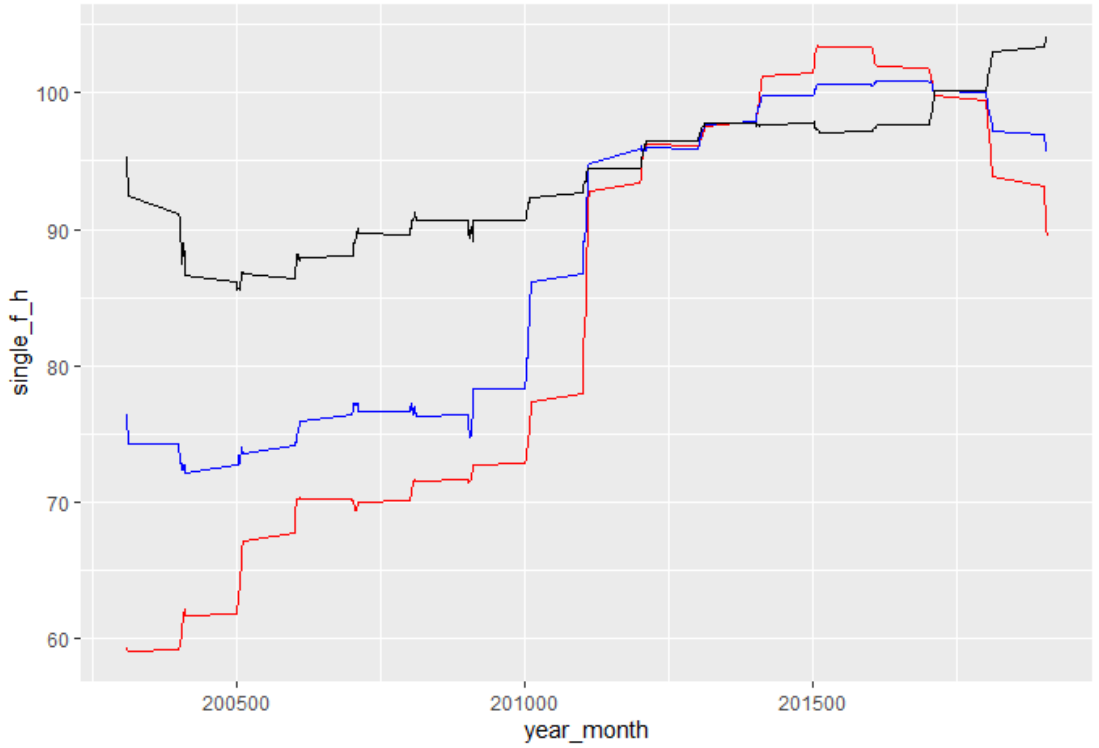
연습문제 08

- ggplot2 패키지 함수로 아파트 거래자료에서 전용면적과 평당 가격으로 그래프를 셋팅하고, 이들 두 값들의 분포에 대하여 커널밀도 추정을 하여 등고선 맵을 작성하라는 명령문은? 실행결과는 다음과 같다.



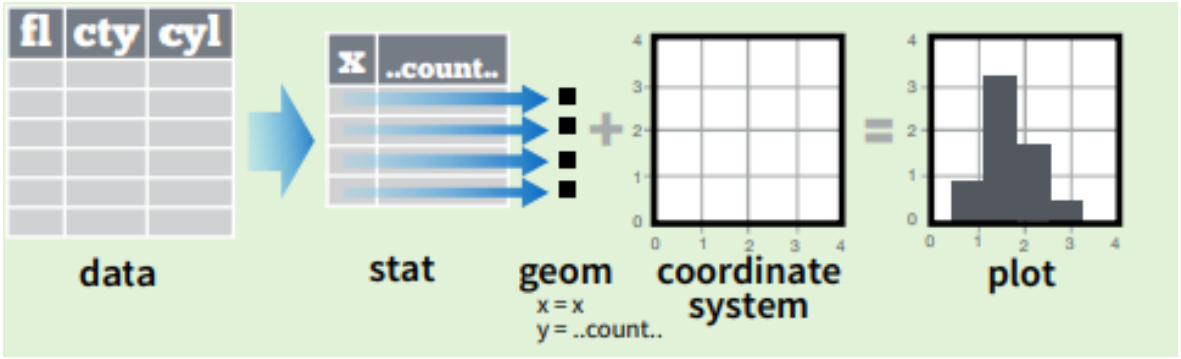
연습문제 09

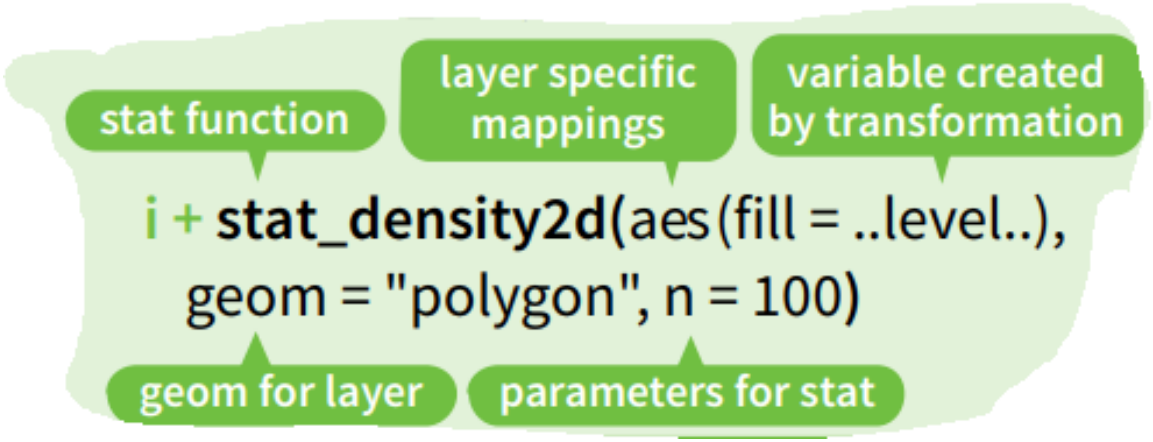
- ggplot2 패키지 의 geom_line() 함수로 주택유형(아파트, 연립/다세대, 단독)별 월별 매매가격지수의 변동 추세를 확인하기 위한 기본 그래프를 작성하시오.



ggplot으로 그래프 작성하기: stat()

- 인수로서의 stat
  - 원래 데이터 그래프 틀의 변형
    - 예: a + geom_bar(stat = "bin")
  - 각 stat는 미학적 속성을 매핑하는 부가적인 변수들을 생성
- 함수로서의 stat_*()
  - geom()함수와 함께 하나의 레이어를 만드는 결합하여 사용됨
    - stat_bin(geom="bar")와 geom_bar(stat="bin") 는 동일



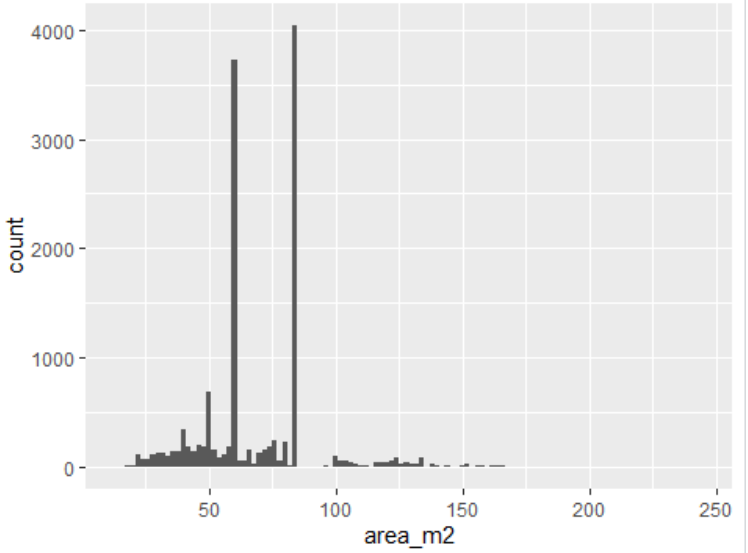


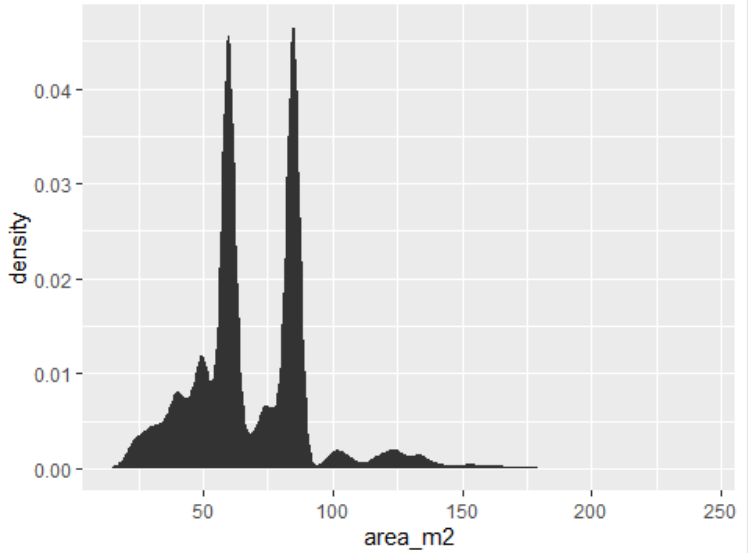
출처: https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf

ggplot으로 그래프 작성하기: stat()

- 1차원 분포(1D distributions)
  - ggplot()
  - + stat_bin()
    - geom_histogram()) 과 동일
    - Visualise the distribution of a single continuous variable by dividing the x axis into bins and counting the number of observations in each bin.
  - + stat_density()
    - geom_density()와 동일
    - Smoothed density estimates

```
> #### 1D distributions
> a <- ggplot(data = apt, # 데이터 셋
+             aes(x = area_m2)) # 전용면적 변수로 그래프 틀 작성
> ?stat_bin() #geom_histogram()) 과 동일, Visualise the distribution o
he number of observations in each bin.
> a + stat_bin(binwidth = 2) # Histograms (geom_histogram()) 과 동일
> ?stat_density() # Smoothed density estimates, geom_density()와 동일
> a + stat_density(adjust = 1, kernel = "gaussian") # geom_density()ㅇ
```

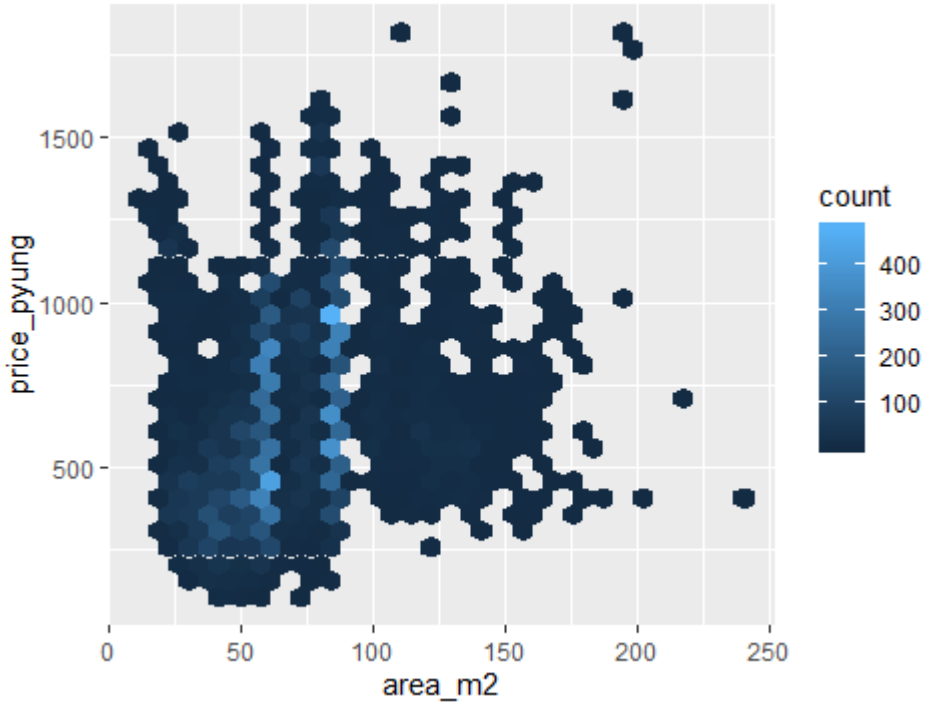




ggplot으로 그래프 작성하기: stat()

- 2차원 분포(2D distributions)
  - ggplot()
  - + stat_bin2d()
    - Heatmap of 2d bin counts
    - geom_bin2d()함수와 동일
  - + stat_binhex()
    - Hexagonal heatmap of 2d bin counts
    - geom_hex()와 동일
  - + stat_density2d()
    - Contours of a 2d density estimate
    - geom_density_2d()와 동일

```
> #### 2D distributions
> c <- ggplot(apt, # 아파트 거래가격 데이터
+             aes(area_m2, price_pyung)) # 그래프 셋팅(x =
> ?stat_bin2d # Heatmap of 2d bin counts: geom_bin2d()함
> c + stat_bin2d(bins = 30, drop = TRUE) # TRUE removes
> ?stat_binhex # Hexagonal heatmap of 2d bin counts: geo
> c  + stat_binhex(bins = 30)
> ?stat_density2d # Contours of a 2d density estimate: g
> c + stat_density2d(contour = TRUE, n = 100) # n
> c + stat_bin2d(bins = 30, drop = TRUE) # TRUE removes
> ?stat_binhex # Hexagonal heatmap of 2d bin counts: geo
> c  + stat_binhex(bins = 30)
> ?stat_density2d # Contours of a 2d density estimate: g
> c + stat_density2d(contour = TRUE, n = 100) # n
> ?stat_binhex # Hexagonal heatmap of 2d bin counts: geo
> c  + stat_binhex(bins = 30)
```



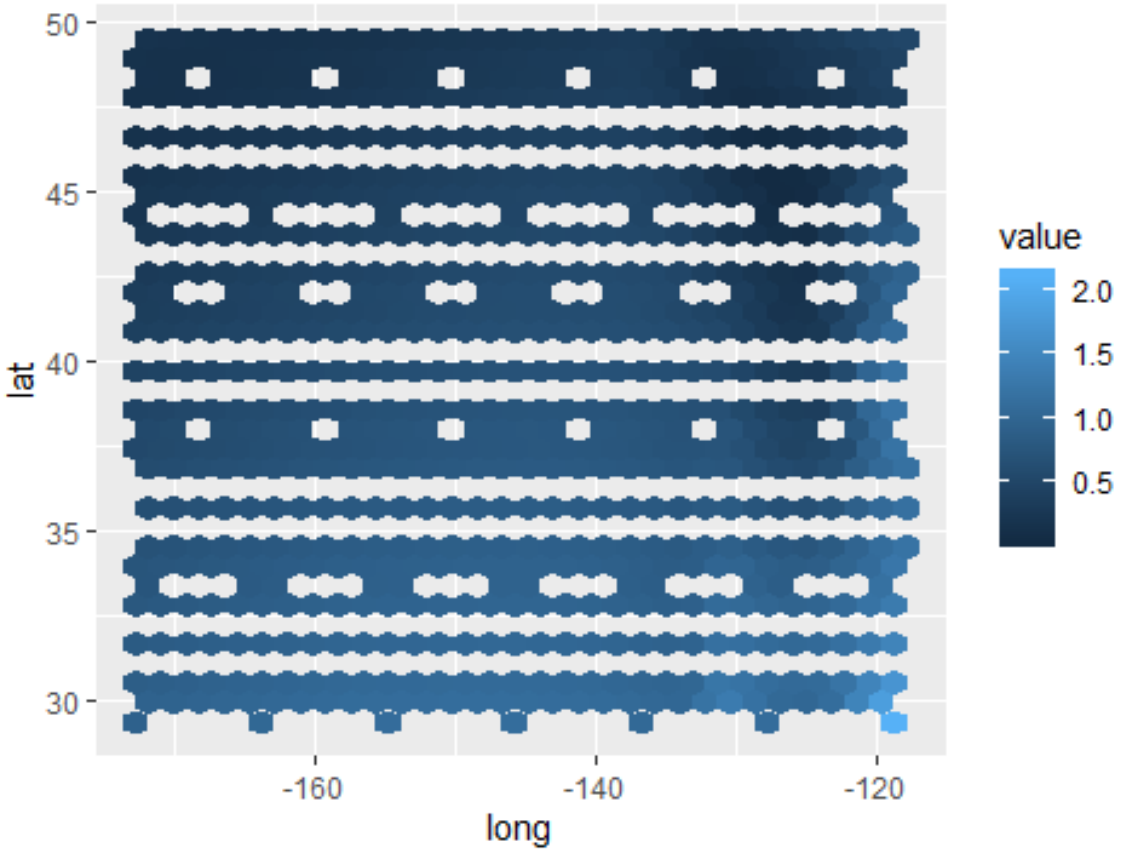
ggplot으로 그래프 작성하기: stat()

- 3차원 분포(3D distributions)
  - ggplot()
    - + stat_contour ()
      - 2d contours of a 3d surface: geom_contour()
    - + stat_spoke ()
      - Line segments parameterised by location, direction and distance
      - 더 이상 사용되지 않으므로, geom_spoke()로 사용 권장
    - + stat_summary_hex ()
      - Bin and summarise in 2d (rectangle & hexagons)
      - The data are divided into bins defined by x and y, and then the values of z in each cell is are summarised with fun.

ggplot으로 그래프 작성하기: stat()

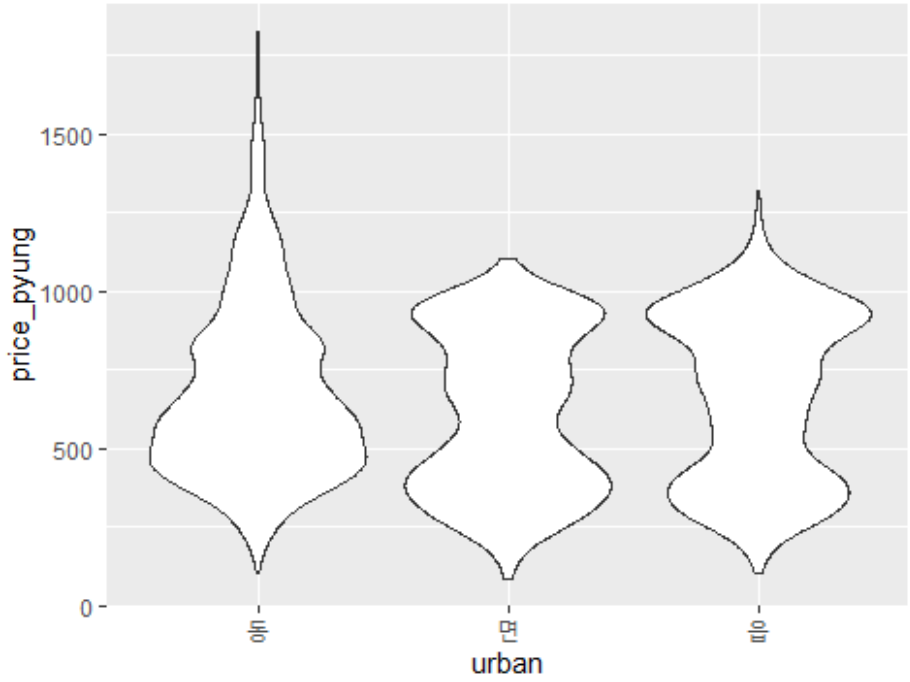
- 3차원 분포(3D distributions)
  - 실습: 동물 이동거리 자료 활용

```
> #### 3 Variables
> seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2)) # 동물 이동거리 계산
> i <- ggplot(seals, # seals 데이터
+             aes(long, lat)) # x, y 좌표
> ?stat_contour # 2d contours of a 3d surface: geom_contour()
> i + stat_contour(aes(z = z))
> ?stat_spoke # Line segments parameterised by location, direction and distan
> i + geom_point() +
+   stat_spoke(aes(radius= seals$z, angle =seals$ z)) # 더 이상 사용되지 않으므로
stat_spoke is deprecated, please use geom_spoke
> ?stat_summary_hex # Bin and summarise in 2d (rectangle & hexagons)
> #*** The data are divided into bins defined by x and y, and then the values
> i  + stat_summary_hex(aes(z = seals$z), bins = 30, fun = mean)
```



ggplot으로 그래프 작성하기: stat()

- Comparisons: x = 이산변수, y=연속변수
  - ggplot()
    - +stat_boxplot()
      - A box and whiskers plot
      - geom_boxplot() 동일
    - +stat_ydensity ()
      - Violin plot
      - geom_violin() 동일
  - 실습



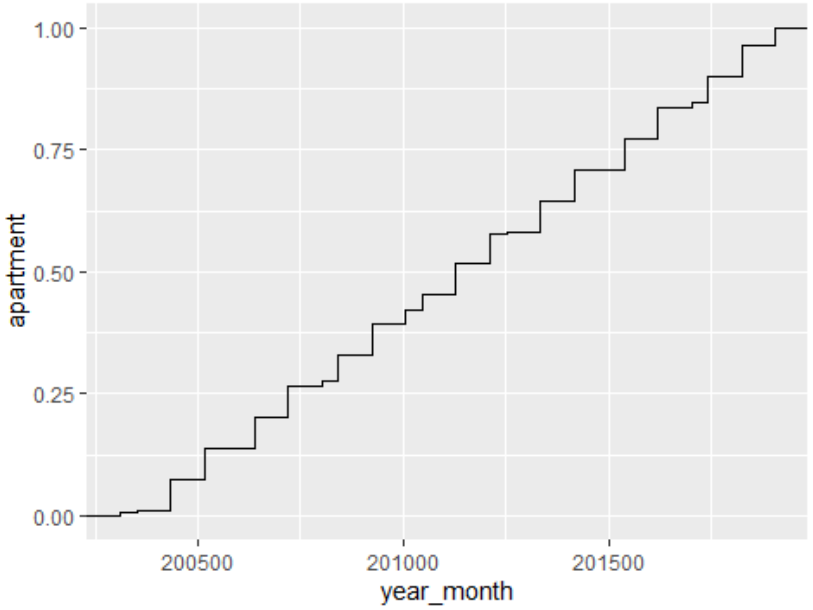

```
> #### Comparisons: x = 이산변수, y=연속변수
> d <- ggplot(apt, # 아파트 실거래 가격 자료
+             aes(urban, price_pyung)) # 미학 그래프 셋팅(x = 읍면동, y= 평당 아파트 가격)
> ?stat_boxplot # A box and whiskers plot: geom_boxplot()
> d + stat_boxplot(coef = 1.5) # coef   = Length of the whiskers as multiple of inter
> ?stat_ydensity # Violin plot: geom_violin()
> d + stat_ydensity(adjust = 1, kernel = "gaussian", scale = "area")
```

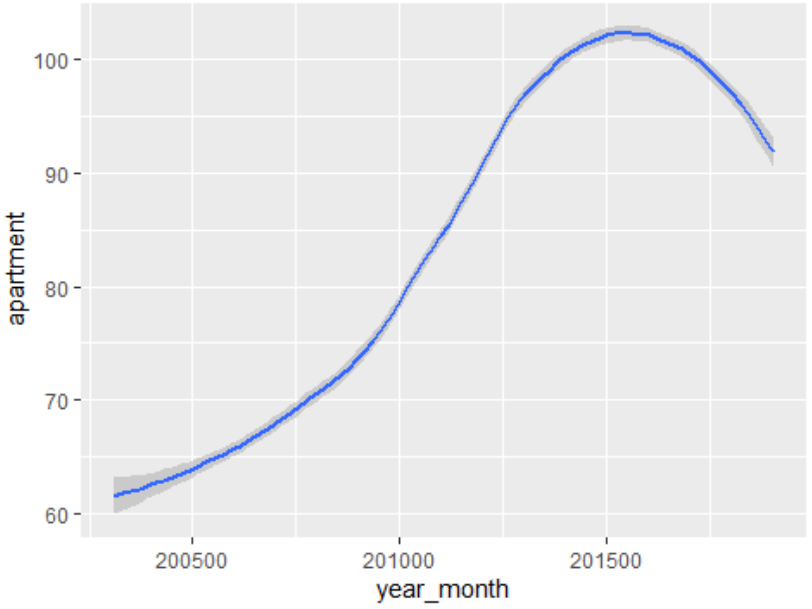
ggplot2 패키지와 데이터 시각화

ggplot으로 그래프 작성하

- Functions
  - ggplot()
    - + stat_ecdf()
      - Compute empirical cumulative distribution
    - + stat_quantile()
      - Quantile regression: geom_quantile()
    - + stat_smooth()
      - Smoothed conditional means: geom_smooth()

```
> #### Functions
> g <- ggplot(m_housing_index, # 월별 주택매매 가격지수 데이터
+             aes(year_month, apartment)) # 년월과 아파트 매매지수 그래프 틀 작성
> g + stat_ecdf(n = 40) # n: if NULL, do not interpolate. If not NULL, this is the number
> g + stat_quantile(quantiles = c(0.25, 0.5, 0.75), # 25%tile, 50%tile, 75%tile로 분위회귀
+                   formula = y ~ log(x), # y를 종속변수로 하고, 설명변수인 x는 로그화 하여 모형
+                       method = "rq") # 사용할 회귀식 함수:  "rq" (for quantreg::rq()) and
> ?stat_smooth() # Smoothed conditional means: geom_smooth()
> g + stat_smooth(method = "auto", # "auto" the smoothing method is chosen based on the s
+                 formula = y ~ x, # 선형회귀식
+                 se = TRUE, # Display confidence interval around smooth? (TRUE by defaul
+                 n = 80, # n   = Number of points at which to evaluate smoother.
+                 fullrange = FALSE, # fullrangeShould the fit span the full range of the
+                 level = 0.95) # Level of confidence interval to use (0.95 by default).
```

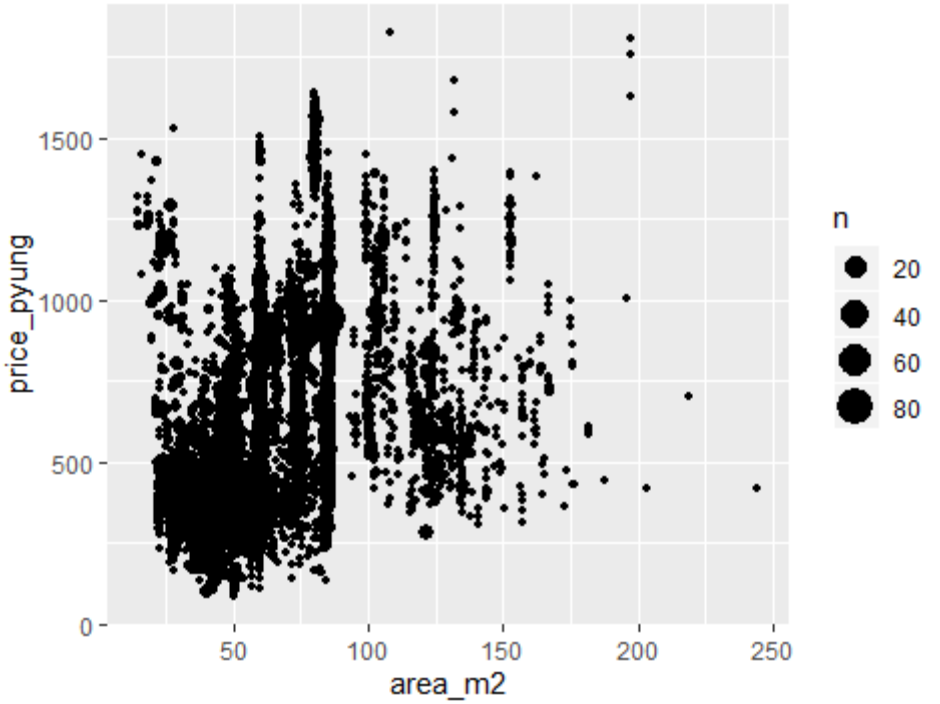




ggplot으로 그래프 작성하기: stat()

- General Purpose
  - stat_function
    - Compute function for each x value
  - stat_identity
    - Leave data as is
    - 데이터 그대로 표현
  - stat_qq
    - A quantile-quantile plot: geom_qq()와 동일
  - sample
    - Random Samples
  - stat_sum()
    - Count overlapping points: geom_count()
  - stat_unique
    - Remove duplicates

```
> ggplot() + # 그래프 셋팅
+   stat_function(aes(x = -3:3), # 데이터
+     fun = dnorm, # Function to use.: 정규확률밀도
+     n = 150, # Number of points to interpolate along
+     args = list(sd=0.5)) # List of additional arguments to pass to fun
> g + stat_identity() # The identity statistic leaves the data unchanged.
> c + stat_identity()
> ggplot() + stat_qq(aes(sample=1:100), # 1부터 100까지 숫자 벡터를 무작위 표본 추출
+                    distribution = qt, # Distribution function to use
+                    dparams = list(df=5)) # Additional parameters passed on to
> g  + stat_sum()
> c +stat_sum()
> g  + stat_unique()
> c  + stat_unique(cex=0.3)
```



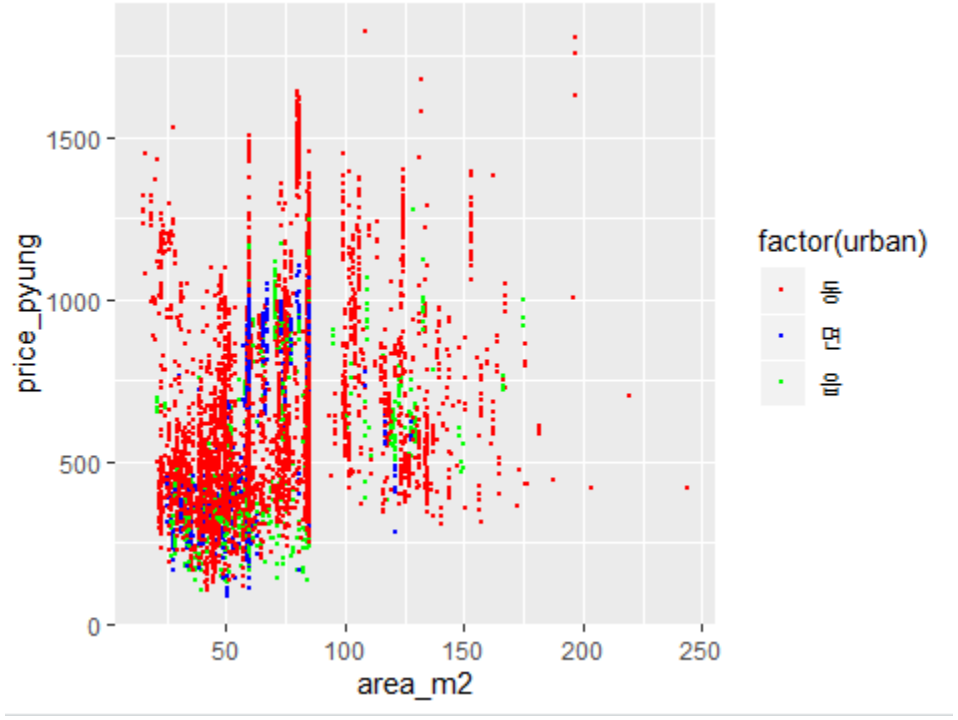
ggplot으로 그래프 꾸미기: scale_*()

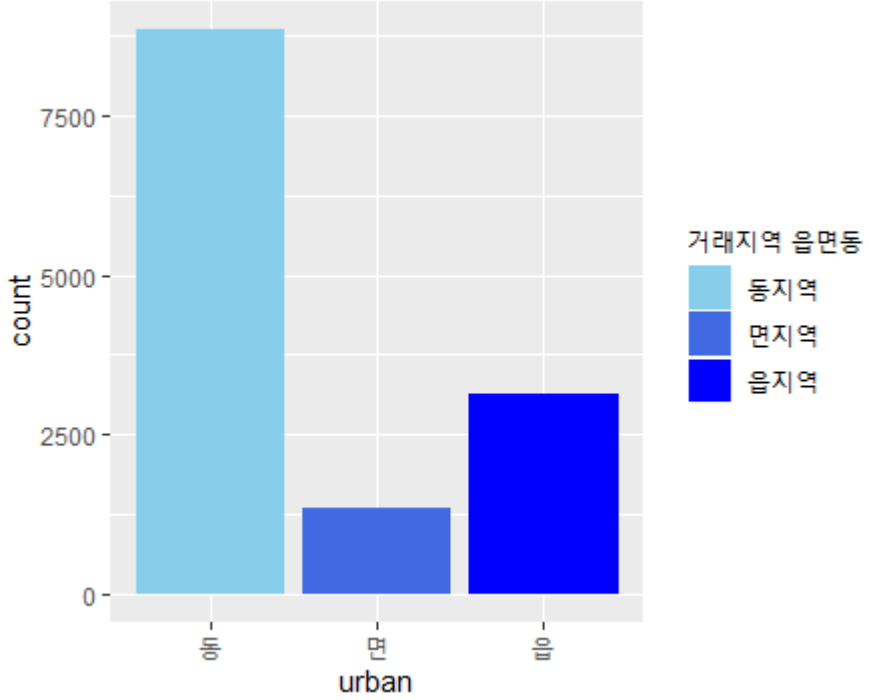
- Scales
  - 데이터 값들을 미학적 요소의 시각적 값들에 매핑하는 방식 관리
- scale_*_manual()
  - These functions allow you to specify your own set of mappings from levels in the data to aesthetic values
- scale_*_manual() 종류
  - 색: scale_colour_manual(…, values, aesthetics = "colour")
  - 분류(stack): scale_fill_manual(…, values, aesthetics = "fill")
  - 크기: scale_size_manual(…, values)
  - 형태: scale_shape_manual(…, values)
  - 선유형: scale_linetype_manual(…, values)
  - 알파-투명도: scale_alpha_manual(…, values)
  - 이산: scale_discrete_manual(aesthetics, …, values)
- X와 y의 축 스케일 조정
  - scale_x_log10()
    - x 또는 y 설정 가능
    - 변수에 상용로그(log10())을 취한 결과를 x 또는 y-위치로 한다.
  - scale_x_reverse()
    - x 또는 y-위치를 뒤집음
  - scale_x_sqrt()
    - 변수에 제곱근(sqrt())을 취한 결과를 x 또는 y-위치로 함

ggplot으로 그래프 꾸미기: scale_*()

- 실습1

```
> c <- ggplot(apt, # 아파트 거래가격 데이터
+             aes(area_m2, price_pyung)) # 그래프 셋팅(x = 전용면적, y= 평당 아파트 가격)
> cc <- c + geom_point(aes(colour = factor(urban)), cex = 0.5)
> cc + scale_colour_manual(values = c("red", "blue", "green"))
> b <- ggplot(apt, aes(urban)) # apt 자료의 urban 명목변수 그래프 셋팅
> (bb <- b + geom_bar(aes(fill=urban))) # 막대 그래프
> bb + scale_fill_manual(values = c("skyblue", "royalblue", "blue"), # 색상
+                        name = "거래지역 읍면동", # 범례 제목
+                        labels = c("동지역", "면지역", "읍지역")) # 라벨 재명명(순서대로)
```

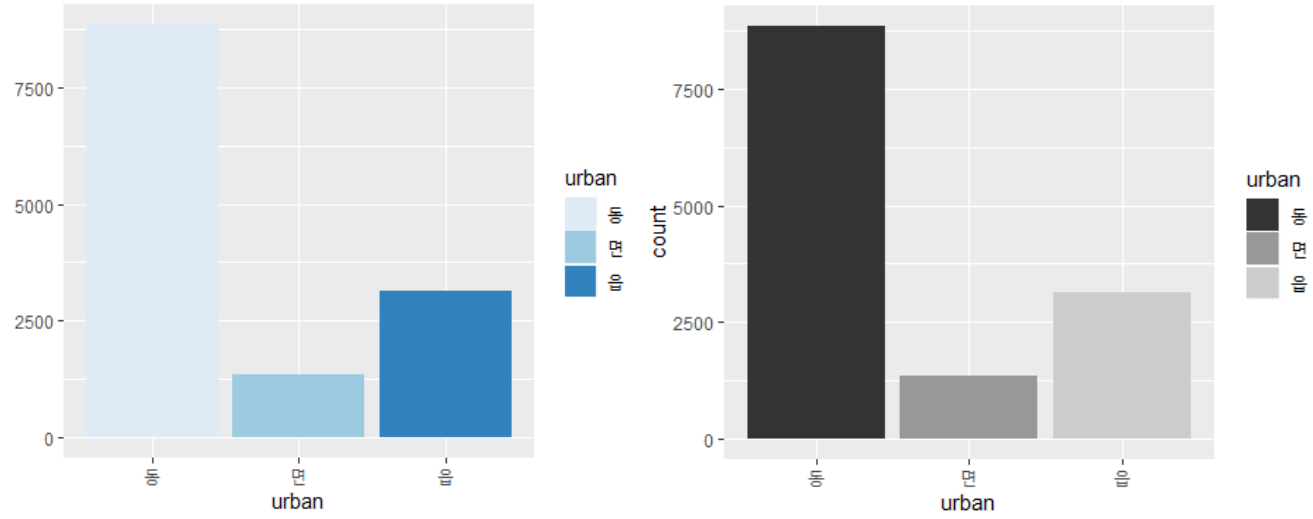




ggplot으로 그래프 꾸미기: scale_*()

- 실습2: 색상과 면색 스케일
  - 이산 변수
    - scale_fill_brewer # Sequential, diverging and qualitative colour scales from colorbrewer.org
    - scale_fill_grey # Sequential grey colour scales
  - 연속 변수: 작동하지 않음

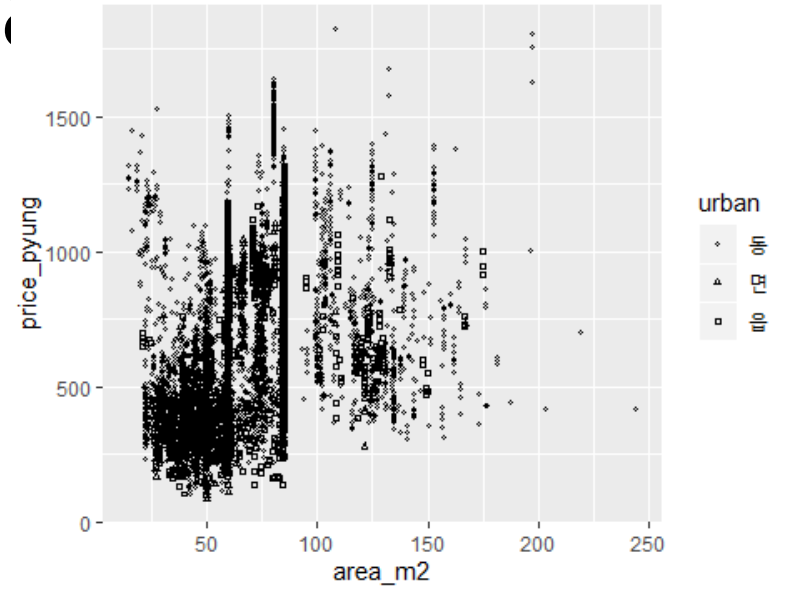
```
> ##### 이산 변수
> n <- b + geom_bar(
+   aes(fill = urban))
> n + scale_fill_brewer(
+   palette = "Blues") # 블루계통 연속 색상 자동 지정
> n + scale_fill_grey( # 회색계통의 연속 색상 설정
+   start = 0.2, # start        = grey value at low end of palette
+   end = 0.8, # end    = grey value at high end of palette
+   na.value = "red") # na.value = Colour to use for missing values
```

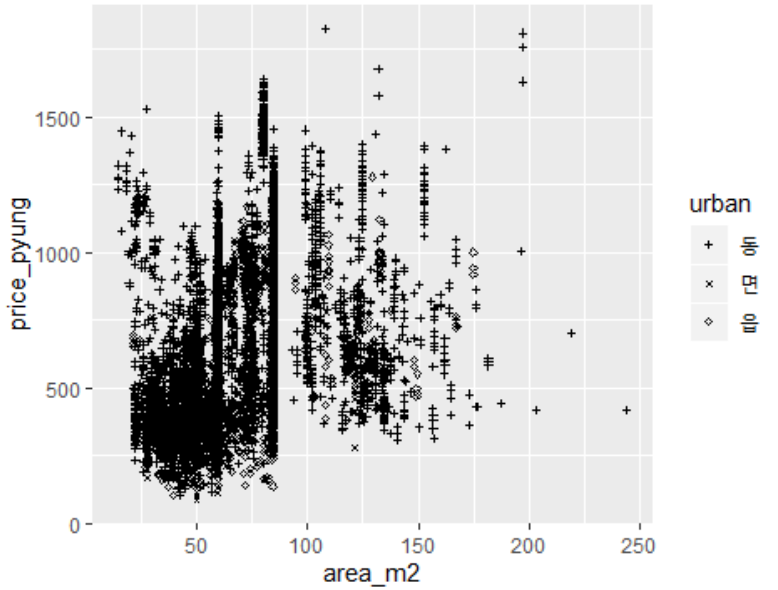


ggplot으로 그래프 꾸미기: scale_*()

- 실습3: 형태(모양) 스케일
  - scale_shape
    - Scales for shapes: maps discrete variables to six easily discernible shapes
  - scale_shape_manual
    - scale_shape_manual(..., values)

```
> c <- ggplot(apt, # 아파트 거래가격 데이터
+             aes(area_m2, price_pyung)) # 그래프 셋팅(x = 전용면적, y= 평당 아파트 가격)
> p <- c + geom_jitter(
+   aes(shape = urban), cex=0.7)
> p + scale_shape(
+   solid = FALSE) # Should the shapes be solid, TRUE, or hollow, FALSE?
> p + scale_shape_manual(
+     values = c(3:5)) # 점의 형태(유형) 지정
> ?scale_shape_manual
```

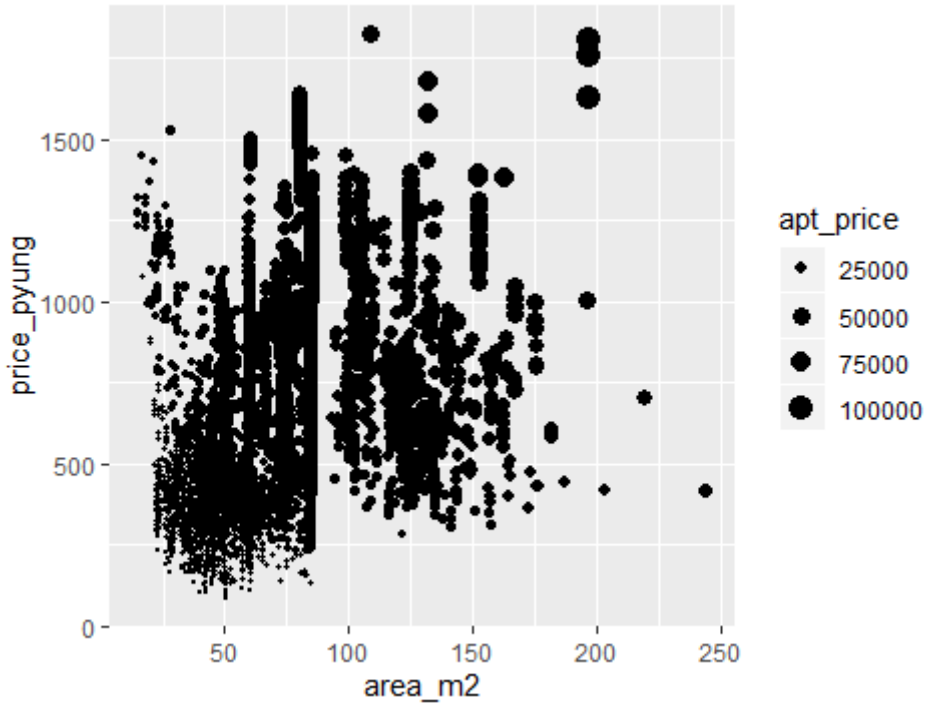




ggplot으로 그래프 꾸미기: scale_*()

- 실습4: 크기(size) 스케일
  - + scale_size_area()
    - Scales for area or radius

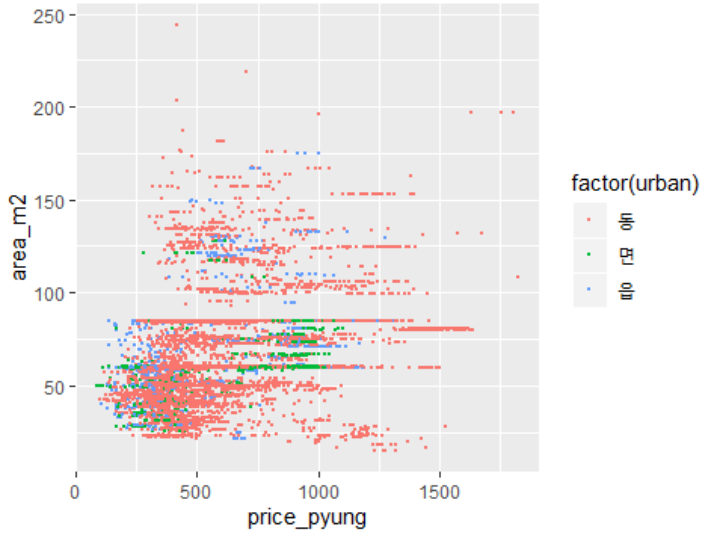
```
> #### Size scales
> q <- c + geom_point(
+   aes(size = apt_price))
> q
> q + scale_size_area(max_size = 4) # a value of 0 is mapped to a size of 0
```

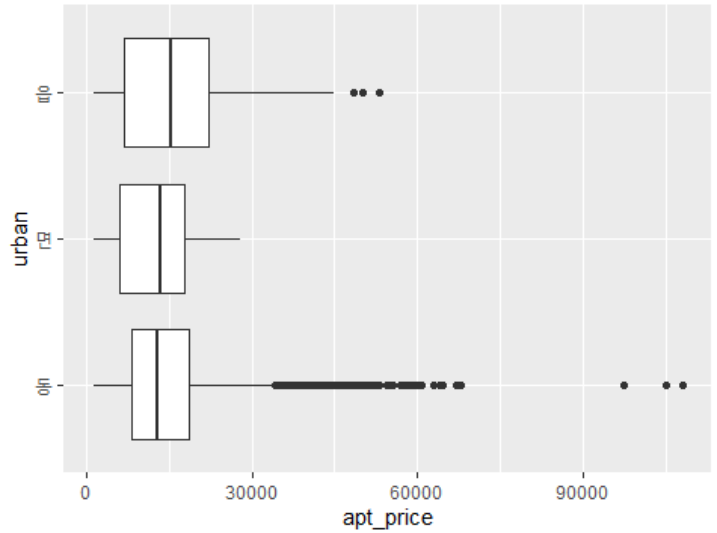


ggplot으로 그래프 꾸미기: scale_*()

- 실습4: 좌표계, 축 스케일
  - coord_cartesian
    - Cartesian coordinates(카테시안 좌표)
  - coord_fixed
    - Cartesian coordinates with fixed "aspect ratio"
  - coord_flip
    - Cartesian coordinates with x and y
    - flipped(뒤집다)

```
> ### Coordinate Systems
> b <- ggplot(apt, aes(urban)) # apt 자료의 urban 명목변수 그래프 셋팅
> (r <- b + geom_bar()) # 막대 그래프
> r + coord_cartesian(xlim = c(0, 5)) # Limits for the x and y axes.
> rr <- c + geom_point(aes(colour = factor(urban)), cex = 0.5)
> rr + coord_fixed(ratio = 1/2) # aspect ratio, expressed as y / x
> rr + coord_flip() # horizontal becomes vertical, and vertical, horizontal.
> str(apt)

> ggplot(apt, aes(urban, apt_price)) +
+   geom_boxplot() +
+   coord_flip()
> r + coord_polar(theta = "urban", direction=1 )
```

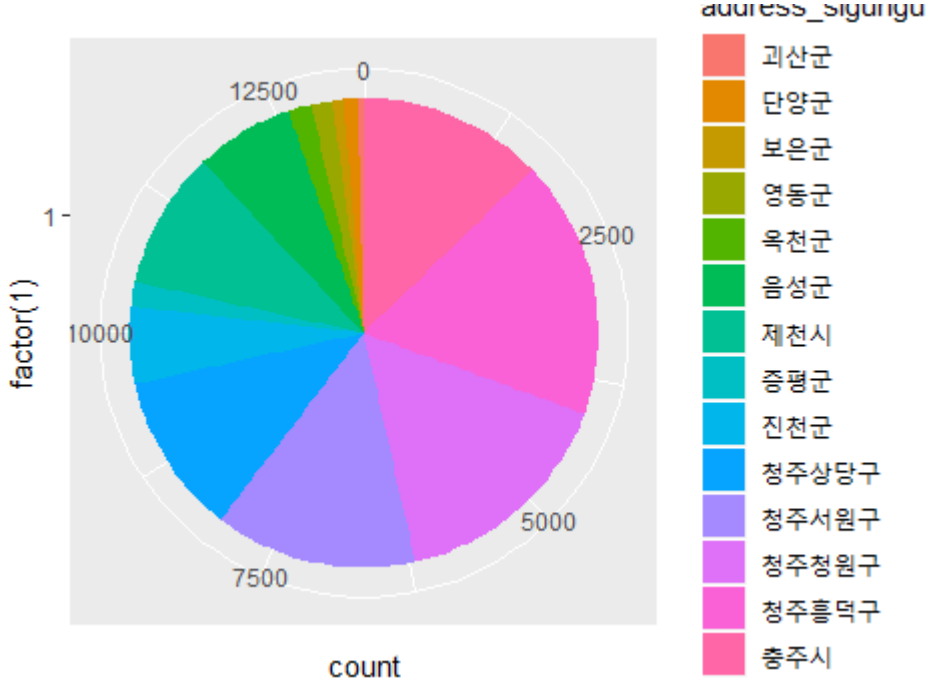




ggplot으로 그래프 꾸미기: scale_*()

- 실습5: 파이 차트 만들기
  - coord_polar
    - most commonly used for pie charts, which are a stacked bar chart in polar coordinates.
  - A pie chart = stacked bar chart + polar coordinates

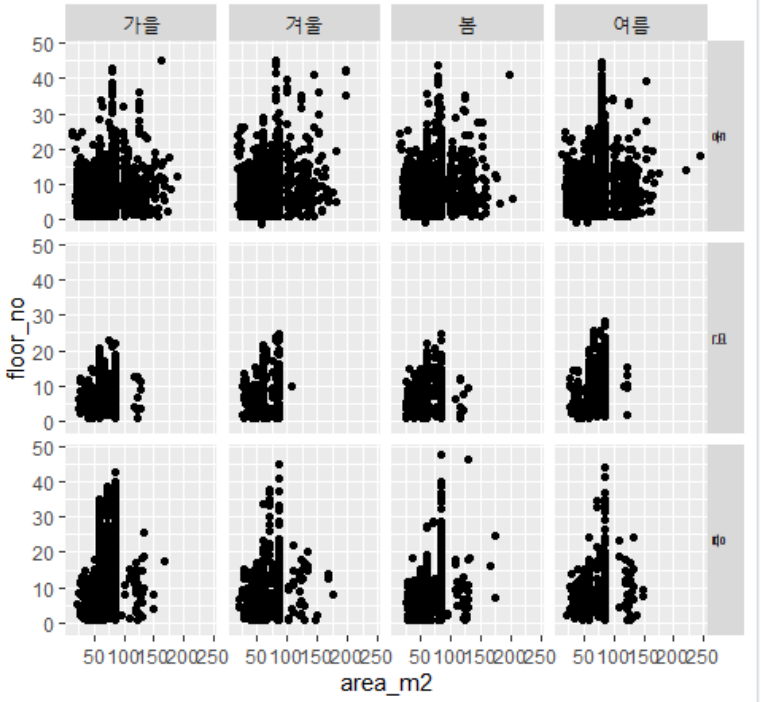
```
> # A pie chart = stacked bar chart + polar coordinates
> rrr <- ggplot(apt, aes(x = factor(1), fill = address_sigungu)) +
+   geom_bar(width = 1)
> rrr
> r + coord_polar(theta = "urban", direction=1 )
Error in match.arg(theta, c("x", "y")) :
  'arg' should be one of “x”, “y”
> # A pie chart = stacked bar chart + polar coordinates
> rrr <- ggplot(apt, aes(x = factor(1), fill = address_sigungu)) +
+   geom_bar(width = 1)
> rrr
> rrr + coord_polar(theta = "y")
```

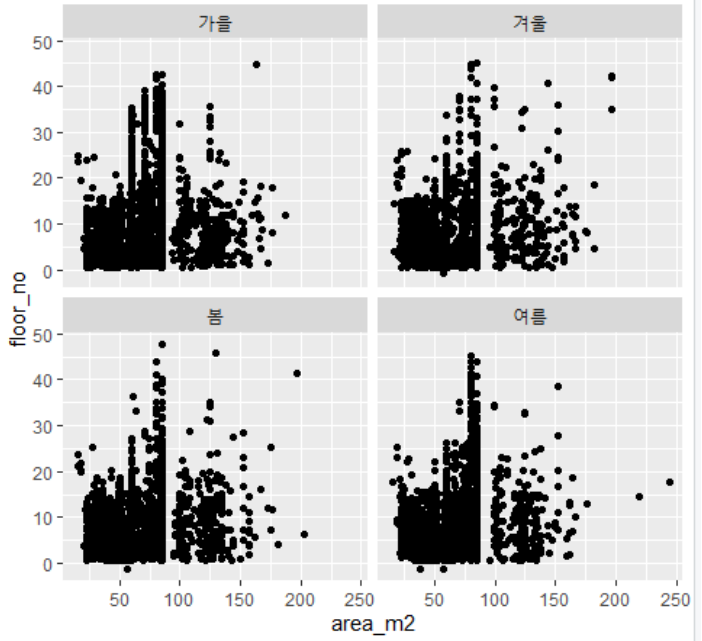


ggplot으로 그래프 꾸미기: fa

- Faceting
  - Facets divide a plot into subplots based on the values of one or more discrete variables
  - `facet_wrap()`, `facet_grid()` 함수
    - 요인 변수로 쪼개진 데이터에 대해 시각화해서 집합으로 묶어 그리기
  - facet_grid()
    - facet into columns based on "~ x", " y ~ ", or "y~ x"
  - facet_wrap()
    - wrap facets into a rectangular layout

```
> #### Faceting: Facets divide a plot into subplots based on the
> t <- ggplot(apt,
+             aes(area_m2, floor_no)) + # 면적과 층수
+           geom_jitter() # jitter 점분포 그래프
> t + facet_grid(. ~ season) # 계절별 그래프 세로로...
> t + facet_grid(urban ~ .) # 읍면동별 그래프 가로로...
> t + facet_grid(urban ~ season) # 계절별 세로, 읍면동별 세로로....
> t + facet_wrap(~ season) # 사각형 행렬 분할로 ...
```





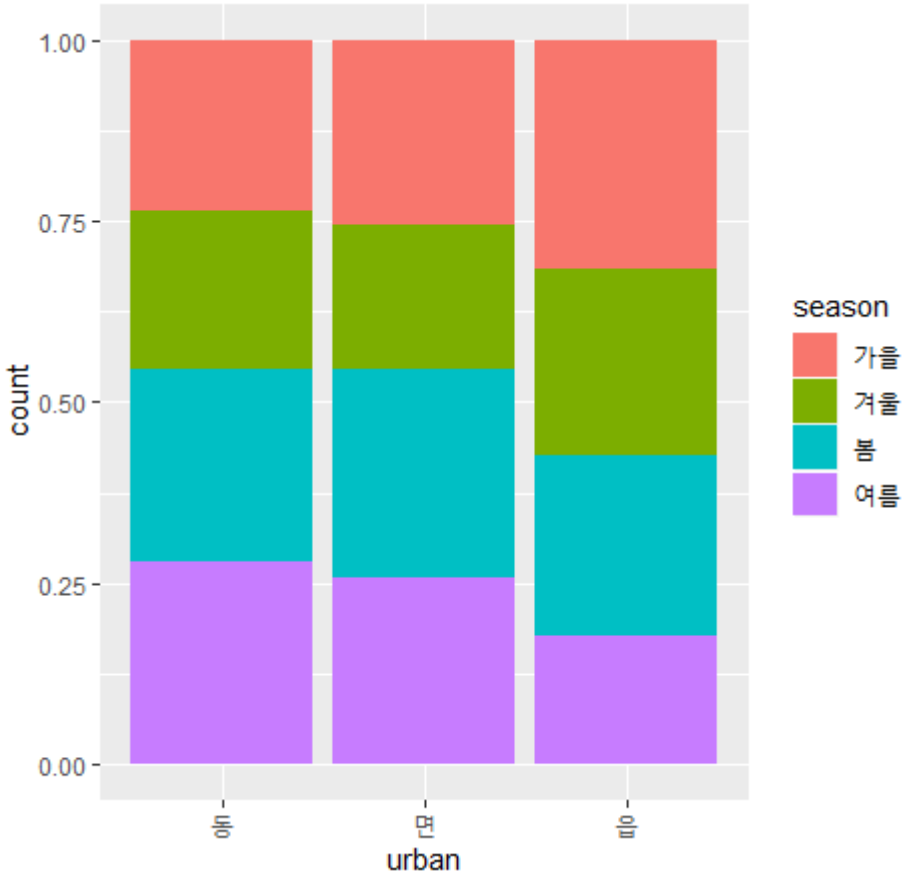
ggplot으로 그래프 꾸미기: position 조정

- Position adjustments
  - determine how to arrange geoms() that would otherwise occupy the same space
  - position = "dodge“
    - Arrange elements side by side
  - position = "fill"
    - Stack elements on top of one another, normalize height
  - position = "stack“
    - Stack elements on top of one another
  - position = "jitter“
    - Add random noise to X and Y position of each element to avoid overplotting

ggplot으로 그래프 꾸미기: position 조정

- Position adjustments
  - position = "dodge“
  - position = "fill"
  - position = "stack“
  - position = "jitter“

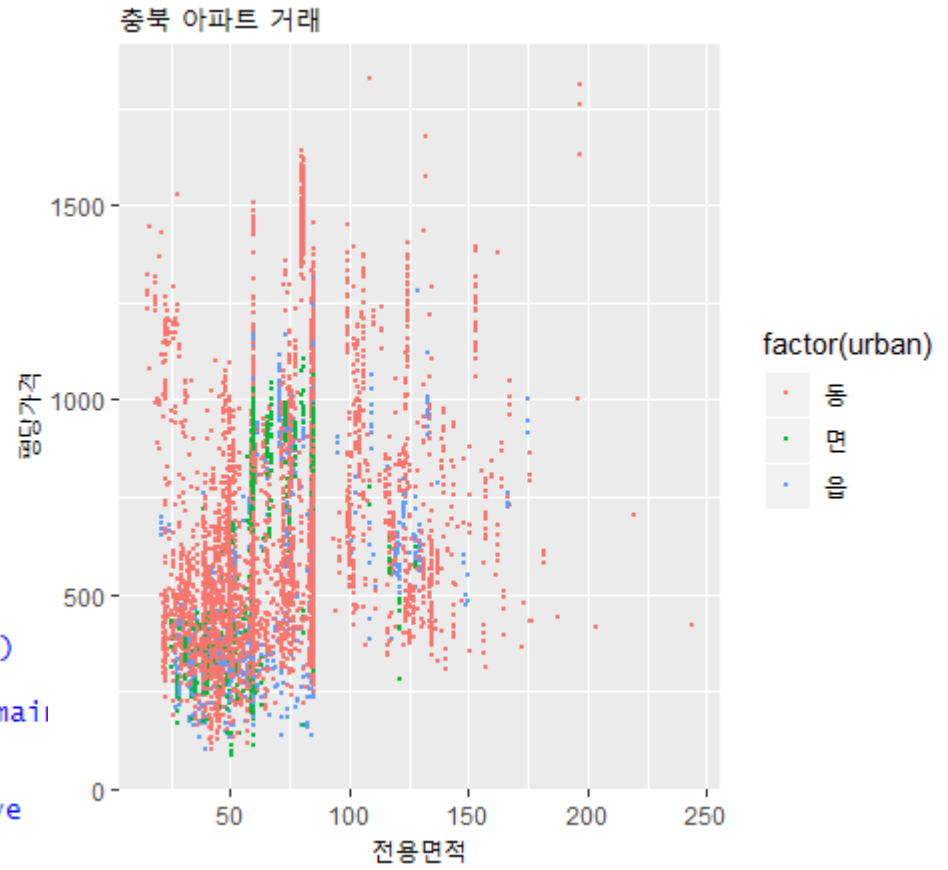
```
> #### Position Adjustments
> #****** Position adjustments determine how to arrange geoms that
> s <- ggplot(apt, aes(urban, fill = season))
> s + geom_bar(position = "dodge") # Arrange elements side by sid
> s + geom_bar(position = "fill") # Stack elements on top of one
> s + geom_bar(position = "stack") # Stack elements on top of one
> c + geom_point(position = "jitter") # Add random noise to X and
> s + geom_bar(position = "fill") # Stack elements on top of one
```



ggplot으로 그래프 꾸미기: Labels

- 라벨링
  - x축, y축, 제목 등 명명
    - + ggtitle()
    - + x lab()
    - + y lab()
    - + labs(title = …, x=…., y=…)

```
> #### Labels
> c <- ggplot(apt, # 아파트 거래가격 데이터
+             aes(area_m2, price_pyung)) # 그래프 셋팅(x = 전용면적, y= 평당 아파트 가격)
> cc <- c + geom_point(aes(colour = factor(urban)), cex = 0.5)
> cc + ggtitle("충북 아파트 실거래 자료: 읍면동별 전용면적과 평당 가격 산포도") +  # Add a mai
+    xlab("전용면적(m2)") +  # Change the label on the X axis
+    ylab("평당가격(단위: 만원)") # Change the label on the Y axis
> cc + labs(title ="충북 아파트 거래", x = "전용면적", y = "평당가격") # All of the above
```



ggplot으로 그래프 꾸미기: theme()

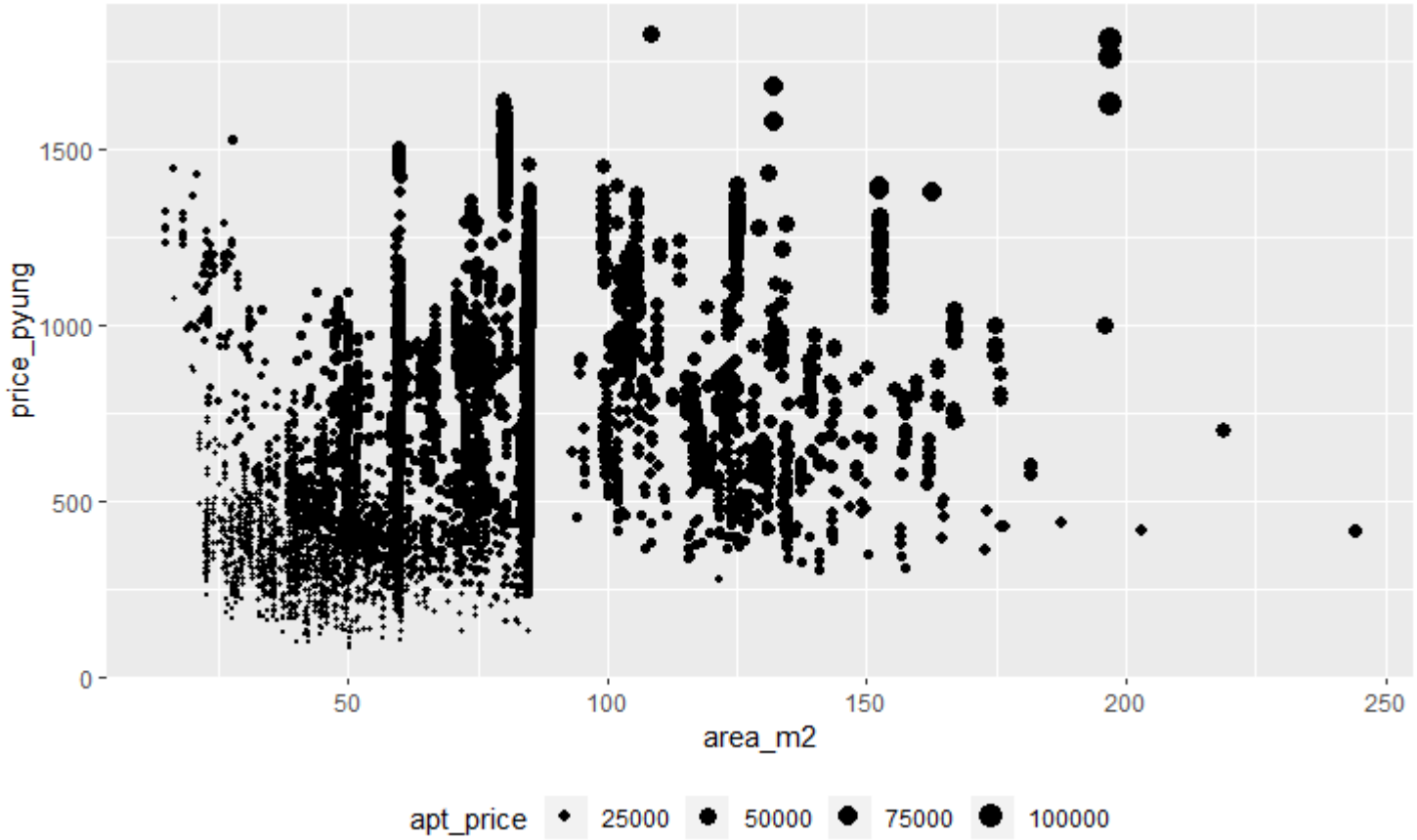
- Modify components of a theme
  - + theme()
  - theme()의 구성요소 확인
    - ?theme()

```
theme(line, rect, text, title, aspect.ratio, axis.title, axis.title.x,
  axis.title.x.top, axis.title.x.bottom, axis.title.y, axis.title.y.left,
  axis.title.y.right, axis.text, axis.text.x, axis.text.x.top,
  axis.text.x.bottom, axis.text.y, axis.text.y.left, axis.text.y.right,
  axis.ticks, axis.ticks.x, axis.ticks.x.top, axis.ticks.x.bottom,
  axis.ticks.y, axis.ticks.y.left, axis.ticks.y.right, axis.ticks.length,
  axis.ticks.length.x, axis.ticks.length.x.top, axis.ticks.length.x.botto
  axis.ticks.length.y, axis.ticks.length.y.left, axis.ticks.length.y.righ
  axis.line, axis.line.x, axis.line.x.top, axis.line.x.bottom, axis.line.
  axis.line.y.left, axis.line.y.right, legend.background, legend.margin,
  legend.spacing, legend.spacing.x, legend.spacing.y, legend.key,
  legend.key.size, legend.key.height, legend.key.width, legend.text,
  legend.text.align, legend.title, legend.title.align, legend.position,
  legend.direction, legend.justification, legend.box, legend.box.just,
  legend.box.margin, legend.box.background, legend.box.spacing,
  panel.background, panel.border, panel.spacing, panel.spacing.x,
  panel.spacing.y, panel.grid, panel.grid.major, panel.grid.minor,
  panel.grid.major.x, panel.grid.major.y, panel.grid.minor.x,
  panel.grid.minor.y, panel.ontop, plot.background, plot.title,
  plot.subtitle, plot.caption, plot.tag, plot.tag.position, plot.margin,
  strip.background, strip.background.x, strip.background.y,
  strip.placement, strip.text, strip.text.x, strip.text.y,
  strip.switch.pad.grid, strip.switch.pad.wrap, ..., complete = FALSE,
  validate = TRUE)
```

ggplot으로 그래프 꾸미기: theme()

- + theme()

```
> #### theme() 예: Legends 위치조정
> q <- c + geom_point(
+   aes(size = apt_price))
> qq <- q + scale_size_area(max_size = 4) # a value of 0 is mapped to a size of
> ?theme # Modify components of a theme
> qq + theme(legend.position = "bottom") # Place legend at "bottom", "top", "lef
```



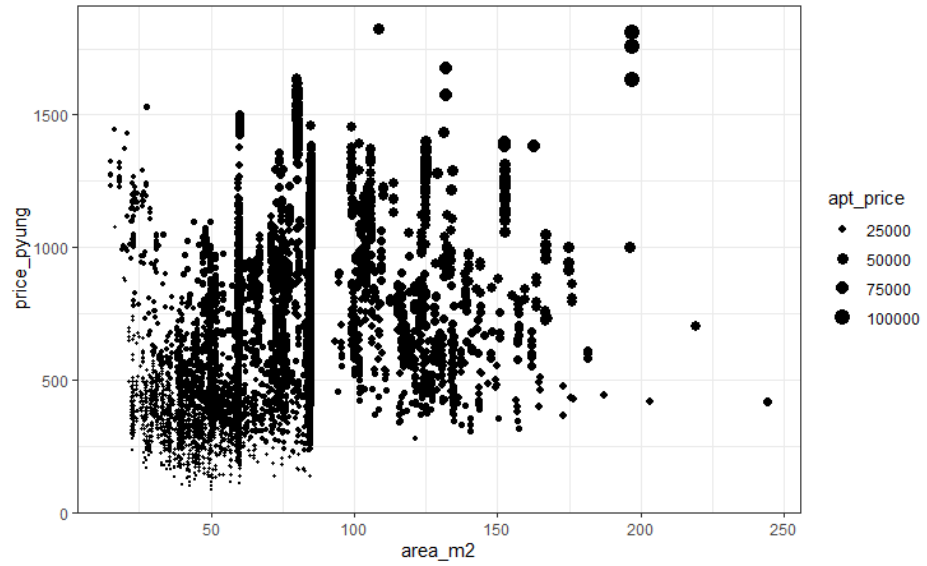
ggplot으로 그래프 꾸미기: theme()

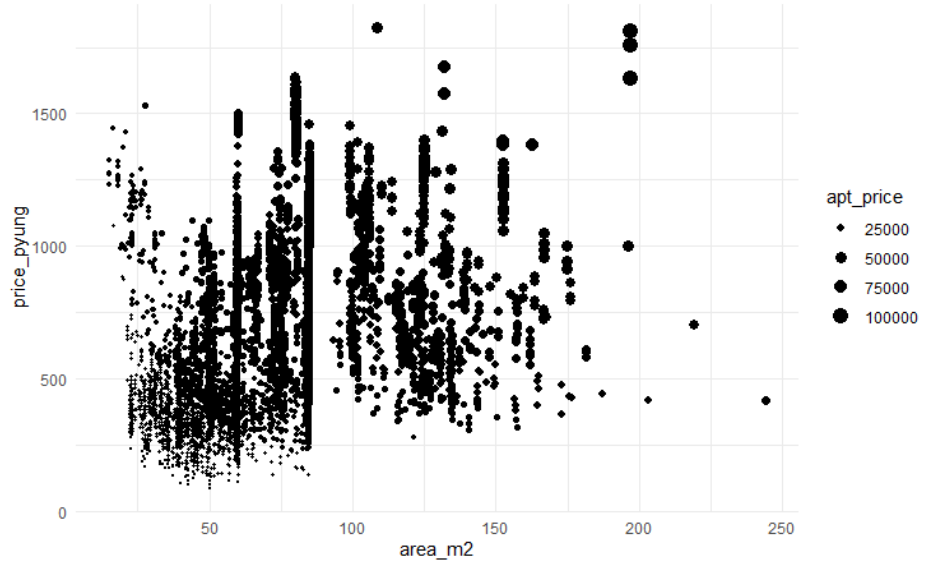
- ?theme_*
  - These are complete themes which control all non-data display
  - + theme_bw()
    - White background with grid lines
  - + theme_classic()
    - White background no gridlines
  - + theme_grey()
    - Grey background (default theme)
  - + theme_minimal()
    - Minimal theme
  - + theme_linedraw()
    - only black lines of various widths on white backgrounds,
  - + theme_light()
    - with light grey lines and axes, to direct more attention towards the data
  - + theme_dark()
    - a dark background.
  - + theme_void()
    - A completely empty theme.

ggplot으로 그래프 꾸미기: theme()

- + theme_*()

```
> qq + theme_bw() # White background with grid lines
> qq + theme_classic() #  White background no gridlines
> qq + theme_grey() # Grey background (default theme)
> qq + theme_minimal() # Minimal theme
> qq + theme_linedraw() # only black lines of various widths on white backgrounds,
> qq + theme_light() # with light grey lines and axes, to direct more attention towards the dat
> qq + theme_dark() # a dark background.
```

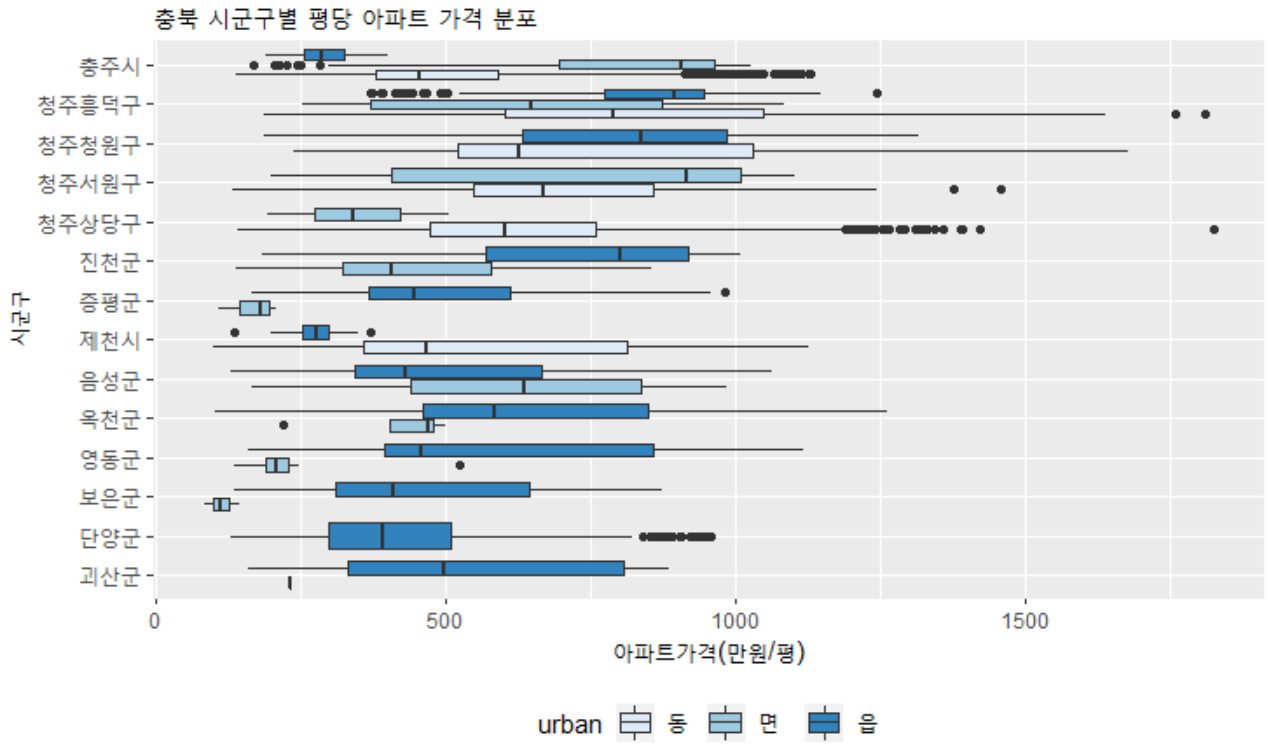




ggplot으로 그래프 그리기: 종합

- 종합 1:

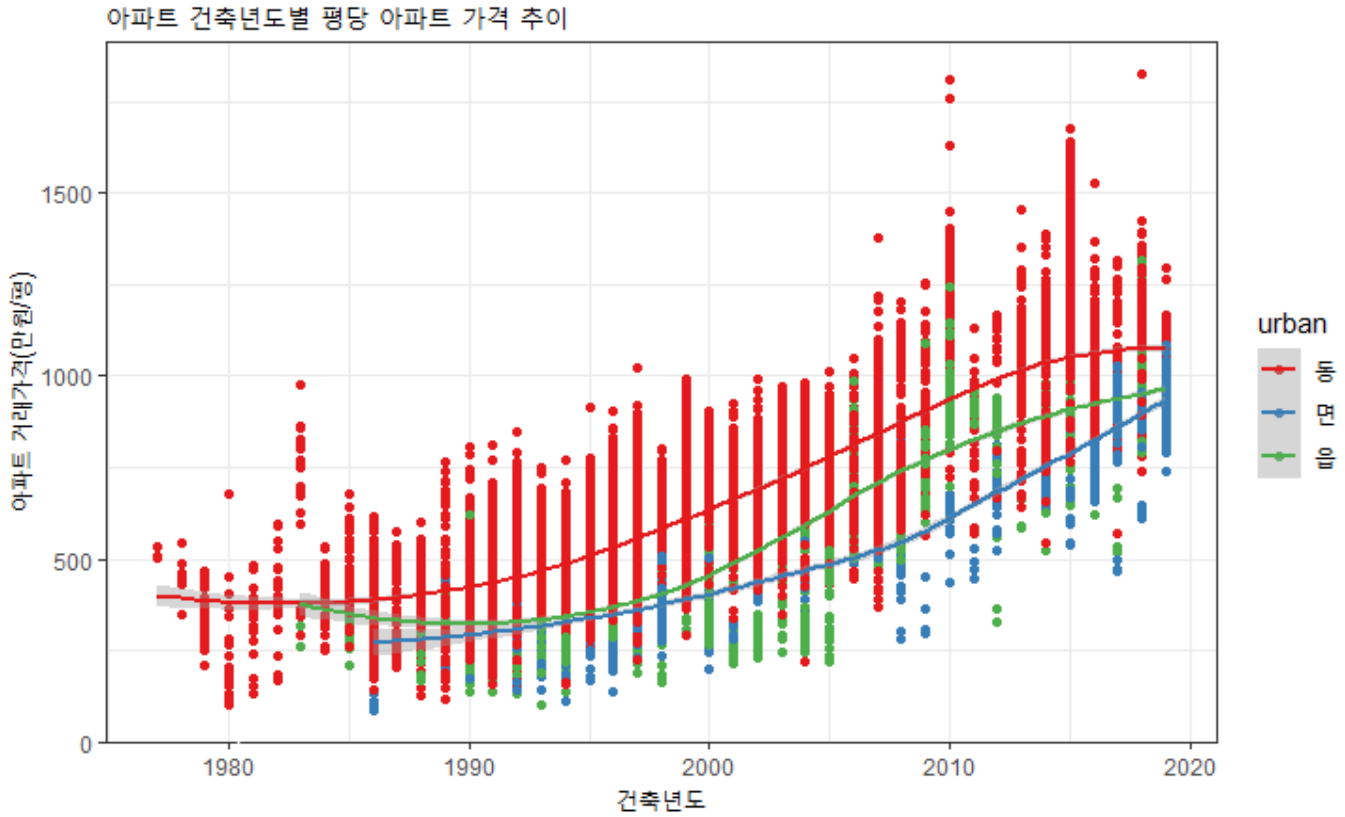
```
> # 종합 1:
> x <- ggplot()+
+   geom_boxplot(data=apt, # 데이터 입력
+                aes(x=address_sigungu,y=price_pyung, # 그래프 틀 작성
+                    fill=urban)) # 표현 데이터 입력
> x + coord_flip() + # 수직에서 수평으로 바꾸기
+   theme(legend.position = "bottom") +
+   labs(title ="충북 시군구별 평당 아파트 가격 분포",
+        x = "시군구",
+        y = "아파트가격(만원/평)" ) +
+   scale_fill_brewer(
+     palette = "Blues") # 블루계통 연속 색상 자동 지정
```



ggplot으로 그래프 그리기: 종합

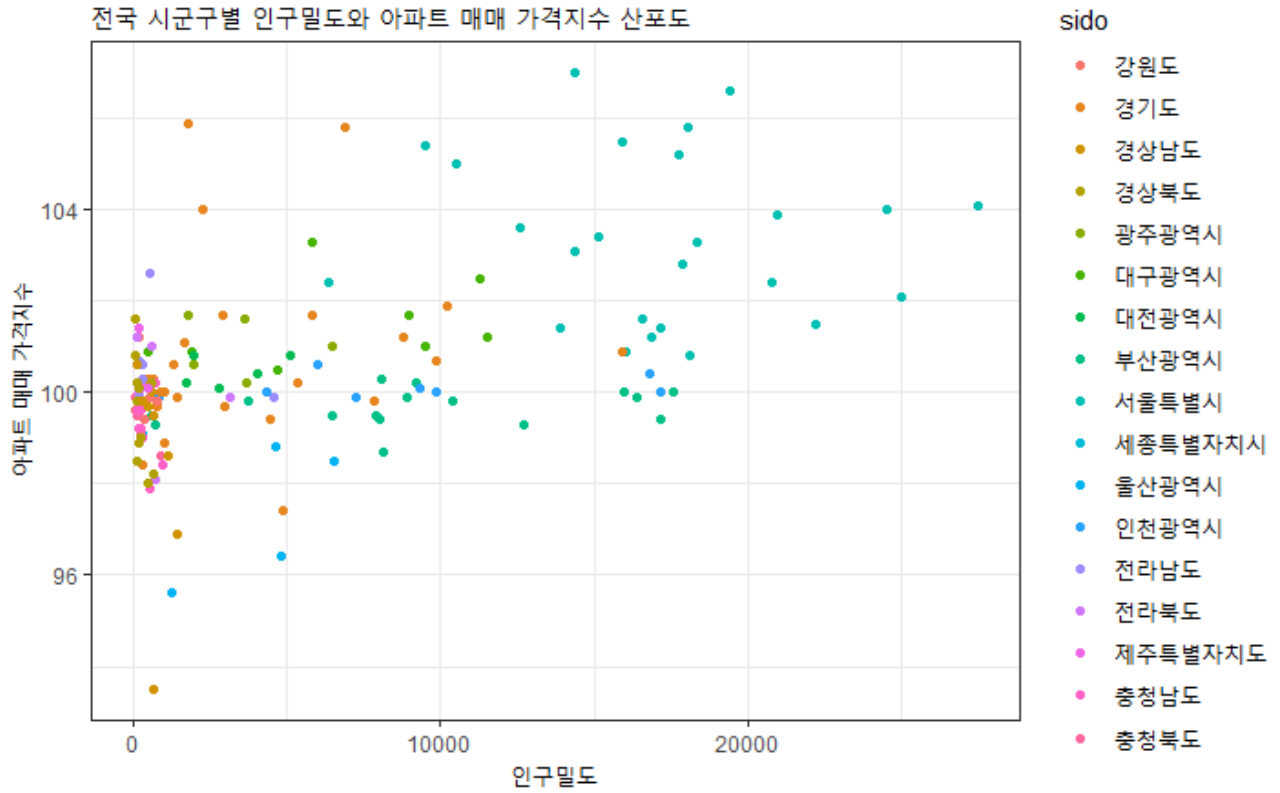
- 종합 2:

```
> yr_price <- ggplot(apt, aes(x = year_built, y = price_pyung, color = urban)) +
+   geom_point() + scale_color_brewer(palette = "Set1")
> yr_price + geom_smooth(method = 'loess') +
+   labs(title ="아파트 건축년도별 평당 아파트 가격 추이", x = "건축년도", y = "아파트 거래가격(만원/평")
> yr_price + geom_smooth(method = 'loess') +
+   labs(title ="아파트 건축년도별 평당 아파트 가격 추이",
+        x = "건축년도",
+        y = "아파트 거래가격(만원/평") +
+   theme_bw()
```



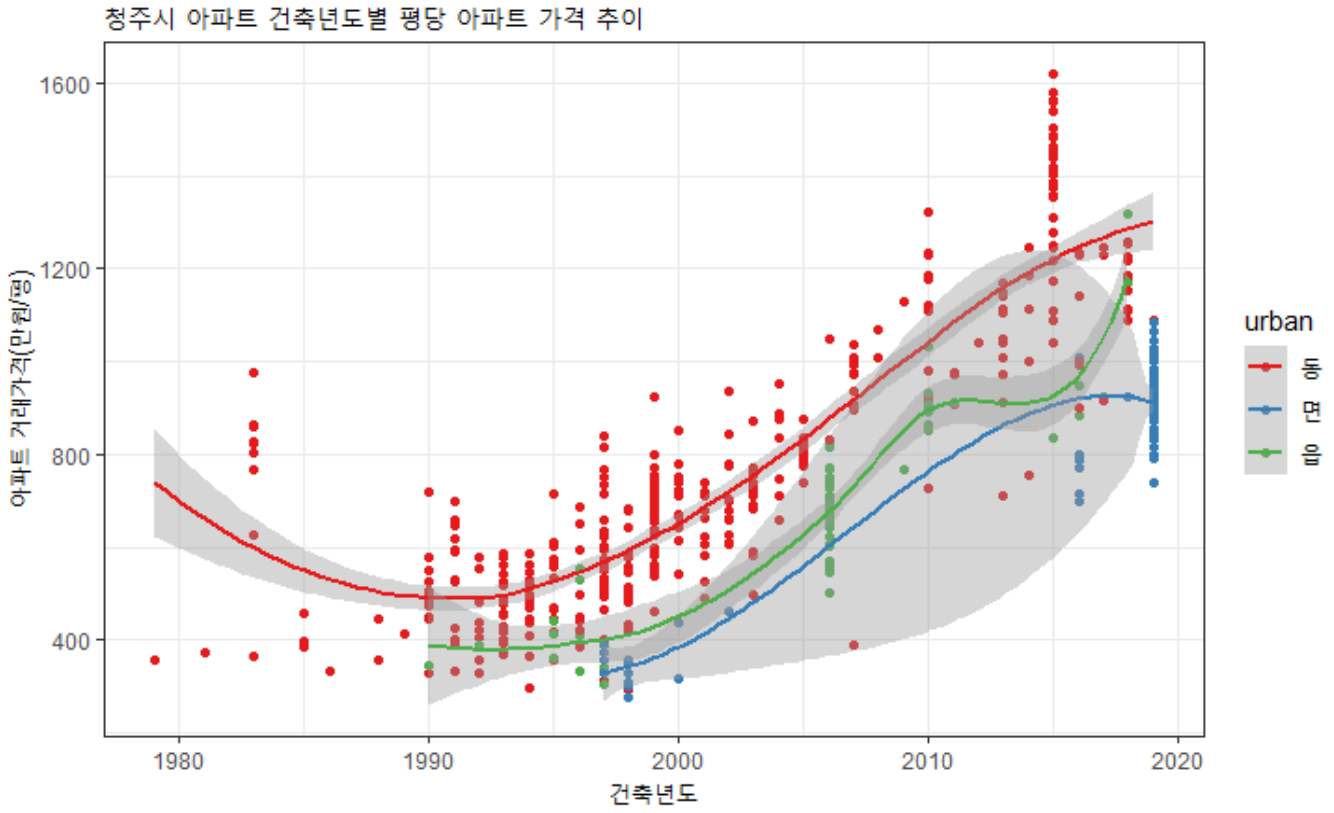
연습문제 10

- 실습자료인 "data_by_sigungu_2018.xlsx" 파일을 불러들인 후, 인구밀도(pop_density)와 아파트 매매가격지수(apt_sale_price)에 대하여 아래와 같은 그래프를 작성하고자 한다. 명령문을 작성하시오. 사용된 함수는 다음과 같다.
  - ggplot()
  - geom_point()
  - labs()
  - theme_bw()



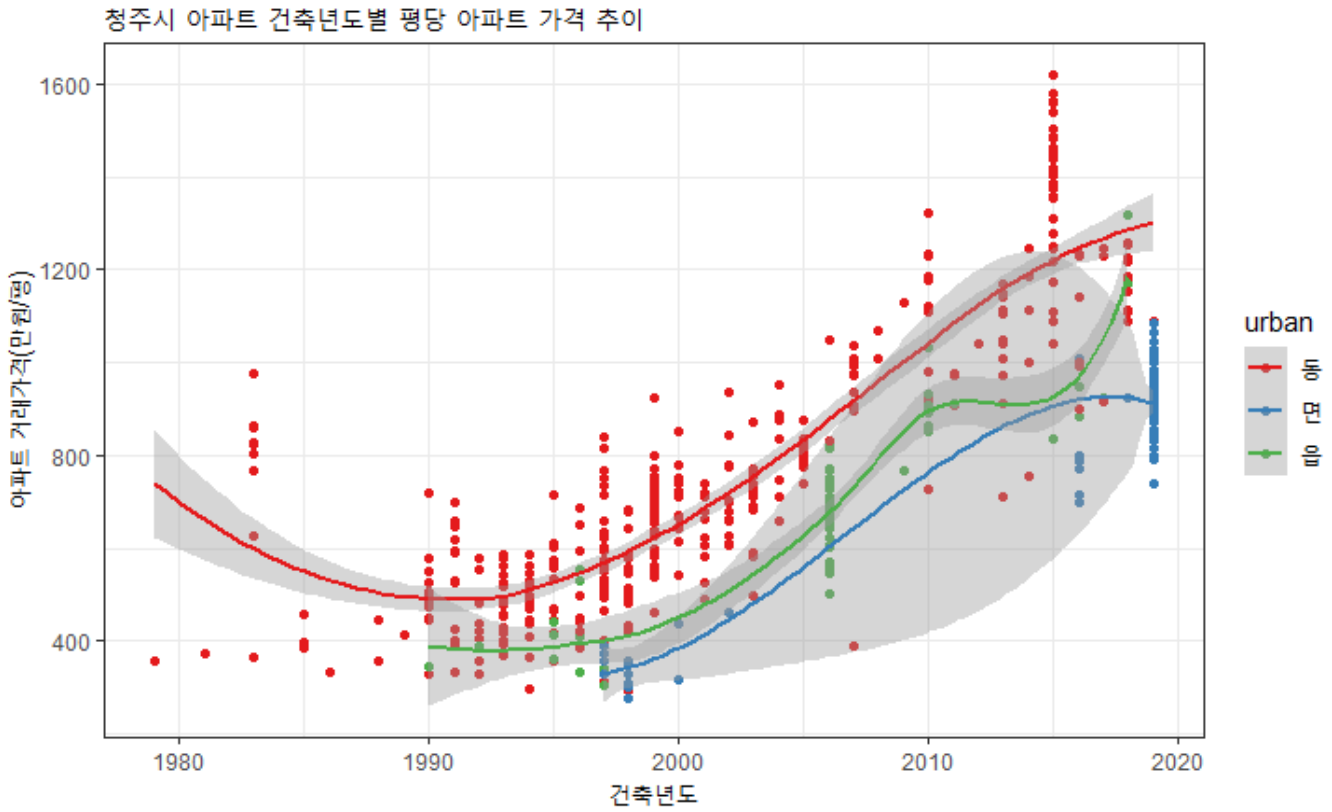
연습문제 11

- 종합 2의 그래프를 2019년 8월에 청주시에서만 거래된 데이터를 추출하여 동일한 그래프를 작성하는 명령문을 작성하시오. 이 때 dplyr 패키지 함수를 활용하시오.



연습문제 12

- 그래프를 보면 청주시 아파트 동지역에 거래된 아파트는 최근 년도로 올 수록 아파트 가격이 높지만 1990년 이전에 지어진 아파트들은 오히려 더 오래 될수록 가격이 높아지는 경향이 있다. 이유가 무엇일까?



요약

- 시각화와 그래프 유형
  - 시각화의 정의와 역사
  - 그래프 형태와 선택
- R로 그래프 작성하기 기초
  - R 그래프의 par(), 색상 pallet 이해하기
  - R 시각화 내장함수 종류
- 내장함수로 그래프 작성하기
  - 박스 그래프: boxplot()
  - 점 그래프: dotchart()
  - 막대 그래프: barplot()
  - 히스토그램: hist()
  - 산포도: plot()
  - 선그래프: plot()
- ggplot2() 패키지와 데이터 시각화
  - ggplot2 개요와 기초
  - ggplot2로 그래프 작성하기: geom()
    - 한 변수, 두 변수, 세 변수, 연속 이변량 변수, 공간 자료 등 시각화
  - ggplot2로 그래프 작성하기: stat()
  - ggplot2로 그래프 꾸미기: scale_*()
  - ggplot2로 그래프 꾸미기: facet_*()
  - ggplot2로 그래프 꾸미기: position adj.
    - 그래프 표현 위치 등 조정
  - ggplot2로 그래프 꾸미기: Labels
  - ggplot2로 그래프 꾸미기: theme()
  - ggplot으로 그래프 그리기: 종합

끝

- 질의와 토의(Question & Discussion)
  - 이번 강의 내용을 시청하고, 실행하면서 궁금한 점이나 어려운 점에 대하여 토의해봅시다.

- 다음 주 강의주제
  - **정형 데이터 분석(기초 통계와 상관 및 회귀분석)**