1. R

R

- JAVA 설치: jdk se development 검색하여 해당 PC에 적합한 자바 프로그램 다운로드
- R 설치: CRAN 검색하여 R과 R studio 설치
- R studio의 구성:

콘솔 창: 명령어 실행

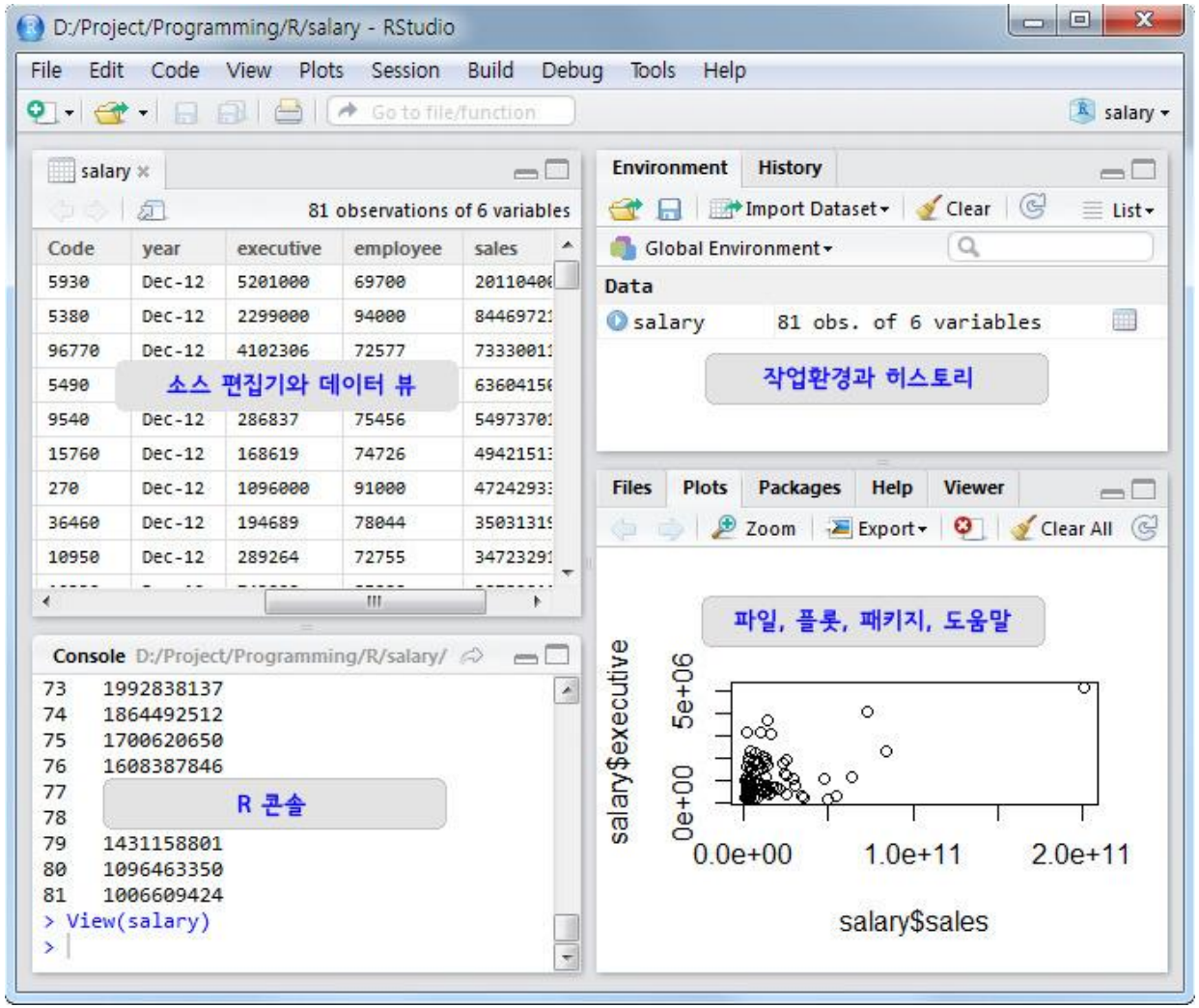
소스 창(script): 명령어 기록과 실행

- 소스창에서 명령 실행하려면, [control+enter] 누름
- 여러 명령문을 한꺼번에 실행 하려면, [shift]로 명령문 지정한 후 실행

환경창: 실행된 명령어 보여줌

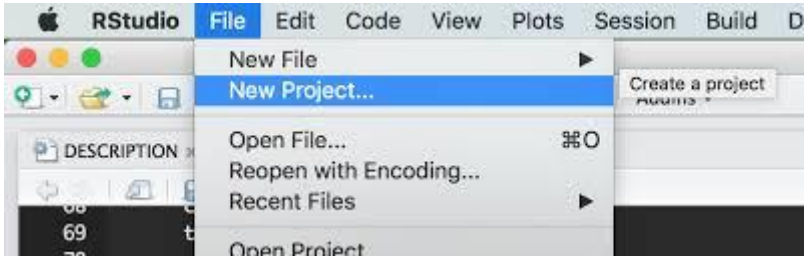
파일창: 폴더에 있는 파일을 보여줌

- working directory의 내용을 보여줌



R

- R studio 프로젝트 만들기



New project → New directory → empty project

→ 영어로 directory name작성하고 저장 위치 설정

→ create project 클릭 → 파일 창에 프로젝트 폴더 생성

: 생성된 폴더가 워킹 디렉토리가 됨

(참고) working directory: 분석결과를 저장하거나 외부에서 파일을 불러올 때 사용하는 폴더

- 스크립트 저장

소스 창에 명령어 입력 후 [control+s] 또는 [file → save] 클릭하면, 생성된 프로젝트 폴더에 새로운 파일이름으로 저장 가능

- 새로운 스크립트 생성

[file → new file → R script]

- 인코딩 방식 설정: 한글이 깨져 나올 때 처리 방법

[tools → project options → code editing] text encoding이 UTF-8로 되어 있는지 확인

R

- 기초연산

```
> a=1 # 변수 a에 1을 할당
> a
[1] 1
> b=2
> a+b
[1] 3
> v1=c(1, 3, 5, 6) # 여러개의 숫자로 된 변수 생성
> v1
[1] 1 3 5 6
> v2=seq(1,10,by=2)
> v2
[1] 1 3 5 7 9

> ch1="데이터마이닝"
> ch1="데이터마이닝"   # 문자로 된 변수 생성
> ch1
[1] "데이터마이닝"
> ch2=c("빅데이터", "데이터마이닝")   #여러 개의 문자로 된 변수 생성
> ch2
[1] "빅데이터"     "데이터마이닝"

> paste(ch2, collapse=",") #쉼표를 구분자로 ch2의 단어들을 하나로 합치기
[1] "빅데이터,데이터마이닝"
> paste(ch2, collapse=" ") #빈칸을 구분자로 ch2의 단어들을 하나로 합치기
[1] "빅데이터 데이터마이닝"
```

R

- 패키지: 함수의 묶음 (패키지 설치→패키지 로드→패키지 안의 특정 함수사용)

```
> install.packages("moments")   # 패키지(moments) 설치
> library(moments)              # 패키지(moments) 로드
> help(skewness)                # 함수(skewness) 설명서 불러오기
> ?skewness                     # 함수(skewness) 설명서 불러오기

> getwd()          # 현재의 working directory 확인 (패키지 로드는 현재 사용하는 working directory에서)
> setwd("c:/")     # working directory 지정
```

R

- 변수 타입

```
> v1=c(1,2,3,1,1)                    # numeric 변수
> v2=factor(c(1,2,3,1,1))            # factor 변수
> v1+2                               #연산이 가능함
> v2+2                               # 연산이 불가능함

> class(v2)                          # 변수 타입 확인
> levels(v2)
  # factor 변수의 구성 범주 확인

> v3=c("a", "b", "c")                  # character 변수
> v4=factor(c("a", "b", "c"))          # 문자로 된 factor 변수

> nv2=as.numeric(v2)                 # numeric 타입으로 변환
```

변수 타입	의미	비고
Numeric	실수	연산 가능
Integer	정수	연산 가능
Complex	복소수	연산 가능
Character	문자	연산 불가능
Logical	논리	연산 불가능
Factor	범주	연산 불가능
Date	날짜	연산 불가능

R

- 데이터 구조

```
# vector
> a=1
>  b=c("hello", "data")

# data frame
> x1=data.frame(a1=c(1, 5), b1=c("big", "data"))
> class(x1)              # 데이터 구조 확인

# matrix
> x2=matrix(c(1:12), ncol=4)

# array
> x3=array(1:20, dim=c(2,5,2))

# list
> x4=list(f1=a, f2=x1, f3=x2, f4=x3)
> x4$f3[ ,3][1]
> x4$f4[ , ,1][ ,2][2]
```

데이터 구조	차원	비고
Vector	1	한 가지 변수 타입
Data frame	2	다양한 변수 타입
Matrix	2	한 가지 변수 타입
Array	다	2차원 이상 matrix
List	다	서로 다른 데이터 구조 포함

R

- 데이터 프레임

  1) R의 많은 함수는 데이터를 읽을 때 데이터 프레임 양식으로 읽음

  2) 데이터를 data frame 양식으로 변형할 필요가 있음: 데이터 안의 변수 이용 시 $ 형식 이용

```
> english=c(90,80,70,75)
> math=c(100, 75, 80,90)
> mid_score=data.frame(english, math)

> mid_score
  english math
1      90  100
2      80   75
3      70   80
4      75   90

> mean(mid_score$english)    # data_name$variable_name
[1] 78.75

> class(english)                  # 변수의 타입 확인
[1] "numeric"
> class(mid_score)                # 변수의 타입 확인
[1] "data.frame"
```

R

- 데이터 프레임

  3) csv나 txt 자료를 data frame으로 변형하여 사용함

```
# csv 파일을 읽어서 data frame으로 변형
data_cv=read.csv("bank.csv");                    # header(변수 명)=T가 default

# txt 파일 불러와서 data frame으로 변환
data_tx=read.table("bank.txt", header=TRUE );    # header=F가 default


#csv나 txt 파일은 문자를 factor로 자동 변환하니 이를 방지하기 위한 방법
data_cv1=read.csv("bank_name.csv", stringsAsFactors=F);
data_tx1=read.table("bank_name.txt", header=TRUE, stringsAsFactors=F);

# 빈칸이 아닌 다른 문자(방식)으로 필드(원소 값) 간 구분이 되어 있는 경우
data_tx2=read.table("bank.txt", sep="," , header=TRUE); # , 로 필드 구분되어 있는 경우

# 결측치가 NA가 아닌 다른 문자로 표시된 경우 처리방법
data_tx3=read.table("bank_missing.txt", header=TRUE, na.strings="."); # 결측치가 . 으로 표기된 경우
data_tx3=read.delim("bank_emp.txt", header=TRUE); # 결측치가 빈칸으로 있는 자료의 경우
```

R

- 데이터 프레임

4) xlsx 자료를 data frame으로 변형하여 사용함

```
# openxlsx 패키지 사용
install.packages("openxlsx"); library(openxlsx);
data_xl=read.xlsx(xlsxFile="bank.xlsx", sheet=1);


# readxl 패키지 사용
install.packages("readxl"); library(readxl);
data_xm=read_excel("bank.xlsx");          # 엑셀 문서(bank.xlsx) 불러서 data frame으로 변환
                                            # 첫 째 행에 변수 명이 포함된 것으로 인식

data_xm2=read_excel("bank.xlsx", col_names=F)  # 첫 째 행에 변수 명이 없는 경우의 처리방법
data_xm3=read_excel("bank.xlsx", sheet=2)       # 엑셀 파일에 시트가 여럿 있는 경우, 해당 시트 지정
data_xm5=read_excel("d:/document/bank.xlsx")  # working directory가 아닌 다른 폴더에서 불러올 때
```

R

- 데이터 프레임 저장

  5) 데이터 프레임 자료를 외부형식으로 저장하기

```
> write.csv(data_xm, file="newdata.csv")        # 데이터 프레임 파일을 csv 형식으로 저장
> save(data_xm, file="newdata.rda")             # 데이터 프레임 파일을 RData 형식으로 저장
> da1=load("newdata.rda")                            # rda 자료 불러오기
> da1
[1] "data_xm"

> library(MASS)                                          # write.matrix 함수 사용 위한 패키지 로드
> write.matrix(my_data, "c:/new_data.txt")       # 텍스트 파일 new_data.txt 이름으로 저장
```

R

- 기본 함수

```
> head(data_xm)                # 데이터 앞부분 6행 출력
> tail(data_xm)                # 데이터 뒷부분 6행 출력
> tail(data_xm,10)             # 데이터 뒷부분 10행 출력
> View(data_xm)                # 뷰어 창에서 데이터 확인
> dim(data_xm)                 # 데이터 차원 출력
> str(data_xm)                 # 데이터 속성 출력
> summary(data_xm)             # 요약 통계량 출력

> install.packages("dplyr")          # R3.4.1 이상 필요
> library(dplyr)
> va1=c(90,85,65)
> va2=c(90,80,75)
> d1=data.frame(va1,va2)

> d1=rename(d1, v2=va2)        # 변수 명 바꾸기: va2를 v2로 수정
> d1$avg=(d1$va1+d1$v2)/2      # 파생변수 만들기: va1과 v2의 평균 변수 avg를 생성
> d1$result=ifelse(d1$avg>=80, "pass", "fail") # 평균이 85점 이상이면 pass, 아니면 fail
> head(d1)
> table(d1$result)       # 합격 빈도표 생성

> library(ggplot2)
> qplot(d1$result)       # 합격 빈도 막대 그래프 생성
> d1$grade=ifelse(d1$avg>=90, "A",ifelse(d1$avg>=80, "B", "C")) # 중첩 조건문 생성
```

R

- 표본추출

```
> k=100
> data1=seq(1,k)
> n=20
> samps =sample(data1, n, replace=F)   # data1에서 20개 랜덤추출. F: 비복원 추출, T: 복원 추출
> samps =sample(data1, n, replace=T)
> samps =sample(data1, n)                    # default는?

> sort(samps)    # 오름차순 정렬
> sort(samps, decreasing=T)  #내림차순 정렬
```