데이터 프레임

생성

- 행과 열로 구성된 2차원 구조를 가져, 행렬과 동일한 형태를 보임
- 각 열이 **서로 다른** 데이터 유형을 가질 수 있음
- 서로 다른 데이터 유형의 2차원 행렬을 표현 할 수 있음
- 동일한 길이의 벡터로 이루어진 리스트를 구성원소로 갖는 데이터 구조
- data.frame() 함수로 생성
    - product <- data.frame(v1,v2,v3)
    - 각 벡터는 데이터프레임의 열을 구성(변수), 열은 반드시 이름을 가져야 함 -> 벡터의 이름 사용
    - 벡터를 행의 이름으로 사용 할 수도 있음(row.names= 옵션 사용)
        - 벡터이름으로 하기싫으면 직접 인수로 지정
- 데이터 프레임 생성시 문자열 벡터는 팩터 유형으로 변환됨
    - 문자유형을 유지하고 싶으면 stringsAsFactors = FALSE
- 행렬이나 리스트 형식의 데이터로부터도 데이터 프레임 생성 가능
    - as.data.frame() 함수 이용 -> 이름을 자동으로 부여(열의 이름)
        - 없으면 v1, v2, v3와 같은 형태로 자동 생성
        - 변경하려면 colnames() 함수 이용
    - 리스트의 경우 원소의 이름을 사용, 없으면 자동 부여
        - 적절하지 못한 이름임
- 데이터 프레임에 포함된 행의 개수 : nrow() 함수로 get
    - 열의 개수는 ncol()(변수의 개수)
- length() 함수는 열의 개수를 반환함

인덱싱

- 열을 선택하는 방법이 매우 다양함
- 행렬과 리스트에서 사용하는 인덱싱 기법을 그대로 이용 할 수 있음
    - 열을 추가하거나 삭제하거나 변경하는 작업은 기존의 행렬 및 리스트의 인덱싱 기법을 이용함
- 두 개의 대괄호 / 한 개의 대괄호를 이용하여 열을 추출 할 수 있음
    - 두 개의 대괄호 : 팩터/벡터
    - 한 개의 대괄호 : 하나 또는 여러 개의 열을 추출 -> 선택된 열들로 구성된 데이터 프레임 반환
        - 하나의 열만 선택해도 데이터 프레임 형식
        - 두개 이상의 열을 선택 할 때는 c() 함수 이용
        - 행렬 처럼 행이나 열을 비워놓고 하나만 넣으면 해당 행/열만 반환 -> 벡터 타입 -> drop=FALSE 사용
- 이름을 이용한 열의 추출도 가능
    - us.state[["state.name"]]

    - us.state$state.name
    - us.state[,"state.name"]
- []에 열 이름을 인수로 주면, 선택된 열이 데이터프레임 형식으로 출력
    - us.state[c("state.name", "state.area")]
    - us.satae[,c("state.name", "state.area")]

다루기

- rbind(), cbind() 함수를 이용해 데이터와 변수 추가 가능
    - rbind() : 새로운 데이터/행 추가, 각 원소는 데이트퍼레임의 각 변수에 대응되어야 함
        - product <- rbind(product, c("A004", "Monitor", 250000))
    - 두개 이상의 여러 행을 추가하려면 우선 추가하고자 하는 데이터를 데이터 프레임으로 만들고 rbind() 함수를 이용해 기존의 데이터프레임에 추가
        - 새로 추가되는 행들은 기존의 데이터프레임과 동일한 열 이름을 가져야 함
    - 새로운 열을 추가하려면 cbind() 함수 이용, 벡터의 이름이 새로운 변수 이름
        - $기호를 이용해 새로운 열 추가도 가능
    - 두개 이상의열을 추가 : 추가하고자 하는 열을 데이터 프레임으로 만들고, cbind()로 추가
        - cbind() 함수가 두개의 데이터프레임을 열의 방향으로 결합
        - cbind() 함수는 열 이름의 중복 여부는 체크하지 않음 -> 두 프레임이 동일한 변수 이름을 갖더라도 이름을 그대로 유지함 -> 주의
        - 행의 길이가 안맞는 경우 재사용 법칙으로 값을 채워넣음
    - 여러 개의 데이터프레임을 한꺼번에 결합 할 수 있음
- 데이터프레임들의 집합은 보통 하나의 리스트 형식으로 저장
    - rbind() 함수를 이용하면 리스트 형식으로 저장된 행 형태의 데이터 집합으로부터 데이터프레임 데이터셋을 쉽게 구축할 수 있음
    - **데이터프레임들의 리스트**인 경우
    - do.call() 함수와 rbind() 함수를 이용해 리스트 내의 각 원소를 행으로 결합하여 하나의 데이터프레임 생성
        - do.call() 함수는 첫번째 인자로 주어진 함수를 호출해 두번째 인수로 주어진 리스트의 **개별 원소에 차례대로 적용**
- 리스트의 리스트로 구성된 데이터 형식을 다뤄야 할 때
    - 동일한 방식으로 할려면, 리스트 형식으로 돼있는 행들을 데이터프레임으로 바꿔줘야함
    - 변환작업을 lapply() 함수를 이용하면 편함
        - 첫번째 인수로 주어진 리스트의 각 원소에 대해 두번째 인수로 주어진 함수를 반복하여 적용
        - do.call(rbind,lapply(lst,as.data.frame)) -> 리스트를 데이터프레임으로 바꾼뒤, 데이터프레임들을 통합
- merge()를 이용하는 방식 : 두 데이터프레임에 공통의 열이 존재 할 때 **공통의 열을 기준**으로 두 데이터프레임을 결합하는 방법
    - 서로 공통의 열의 내용이 일치하는 행만 추출함
    - SQL의 join 기능을 지원
        - all = TRUE로 지정시, 모든 행이 추출됨(합집합)
- subset() 함수 : 좀 더 편리하고 직관적인 방법으로 원하는 행과 열을 선택
    - select 인수 : 선택할 열의 이름 지정

    - subset 인수 : 행을 선택하는 논리식 지정(데이터 프레임의 열 이름 사용 가능)
        - subset(mtcars, select=mpg, subset = (mpg>30))
        - mtcars 데이터프레임에서mpg>30인 행들을 mpg열만 프린트
- 상관계수 행렬 : cor() 함수 사용
- 데이터프레임 열을 선택할 때 $기호를 쓰는게 너무 귀찮다
    - with(데이터프레임, 명령어)로 $기호 생략 가능
        - with(iris, Sepal.Length / Sepal.Width)
        - 여러 개의 명령어를 연속으로 입력해야 하면 중괄호를 이용해 감싸준다
        - summary() : 평균, 중위수ㅜ, 최소값 등 기본적인 기술통계 계산
        - 함수 내에서 이루어진 변수 할당을 밖에서는 사용 못함

          -> 변수 할당자를 <- 말고 <-를 써주면 글로벌 환경에 해당 변수 저장
    - within() 함수 : 데이터프레임의 이름과 $기호 없이 데이터 처리 결과를 함수 내에서 데이터프레임의 열에 직접 할당 가능
        - iris <- within(iris, Sepal.Ratio <- Sepal.Length / Sepal.Width)
- 데이터 프레임의 열에 좀 더 빈번히 반복 접근할 필요가 있다면 데이터프레임을 메모리에 적재하는 것이 편리
    - attach() 함수 : 데이터프레임을 메모리에 적재, 탐색경로상에 포함
        - 포함시키면 데이터프레임의 이름을 더이상 입력하지 않아도 됨
        - 복사본을 메모리에 적재함 -> 메모리에 적재된건 변경되지 않음
        - 값을 변경하는 것처럼 보여도, 로컬변수를 새로 생성하는 실수를 하는 것일 수도 있음 -> 원래의 데이터 프레임은 변하지 않음
        - 동일한 이름을 사용하는 객체간에 충돌이 발생하기도 함
    - detach() 함수 : 메모리에 적재한 데이터프레임을 해제
- sqldf() 함수 : SQL문을 이용한 데이터 처리 방법 제공-> SQL의 SELECT문 사용가능
    - sqldf("select * from mtcars where mpg > 30", row.names = TRUE)
        - mtcars에서 mpg>30인 모든 레코드 추출
    - sqldf("select avg(mpg) as avg_mpg, avg(wt) as avg_wt, gear from mtcars where cyl in (4, 6) group by gear")
        - avg(mpg)를 avg_mpg로, avg(wt)를 avg_wt로, gear를 cyl이 4,6인 걸 가져와서 기억 기준 정렬
        - 마침표가 중간에 들어간 열이름을 써야 하는 경우, 열이름을 큰따옴표나 대괄호로 감싸고 SQL문 전체를 작은 따옴표로 감싼다