

데이터 조작

학습 내용

1. plyr 패키지
2. 변수명 변경 및 변수 추가
3. 조건에 맞는 행만 추출
4. 일부 변수만 추출
5. 정렬 및 요약 통계
6. 데이터 병합 및 행 추가

dplyr 패키지

- 데이터프레임에서만 사용
- 간결한 코드 작성 가능
- 파이프 연산자 (%>%)
 - %>% 기호 이용하여 함수들을 나열하는 방식으로 코드 작성
 - 앞에서 실행된 결과를 다음 함수의 첫 번째 인수로 자동할당
 - 단축키 [Ctrl]+[Shift]+[M] 사용
 - 가독성있게 줄 바꾸기 : %>% 뒤에서 [Enter] 명령문의 어떤 위치에서나 실행 가능
- 변수명 앞에 데이터프레임명을 반복해 입력하지 않음

dplyr 패키지 설치 및 로드

파이프 연산자 (%>%)를 사용하거나 select(), filter(), mutate() 등의 함수를 사용하려면 먼저 dplyr 패키지를 설치하고 메모리로 로드한다.

```
install.packages("dplyr") #  
library(dplyr)            # dplyr
```

실습을 위해 mtcars 데이터셋을 사용한다. 먼저 앞쪽 자료 10개를 확인하고 자료의 구조정보를 확인한다.

```
head(mtcars, 10)          # mtcars
```

```
##           mpg cyl  disp  hp drat   wt  qsec vs am gear carb  
## Mazda RX4      21.0   6 160.0 110 3.90 2.620 16.46 0  1    4    4  
## Mazda RX4 Wag  21.0   6 160.0 110 3.90 2.875 17.02 0  1    4    4  
## Datsun 710     22.8   4 108.0  93 3.85 2.320 18.61 1  1    4    1  
## Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44 1  0    3    1
```

```
## Hornet Sportabout 18.7 8 360.0 175 3.15 3.440 17.02 0 0 3 2
## Valiant 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1
## Duster 360 14.3 8 360.0 245 3.21 3.570 15.84 0 0 3 4
## Merc 240D 24.4 4 146.7 62 3.69 3.190 20.00 1 0 4 2
## Merc 230 22.8 4 140.8 95 3.92 3.150 22.90 1 0 4 2
## Merc 280 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4
```

```
str(mtcars) # mtcars
```

```
## 'data.frame': 32 obs. of 11 variables:
## $ mpg : num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
## $ cyl : num 6 6 4 6 8 6 8 4 4 6 ...
## $ disp: num 160 160 108 258 360 ...
## $ hp : num 110 110 93 110 175 105 245 62 95 123 ...
## $ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
## $ wt : num 2.62 2.88 2.32 3.21 3.44 ...
## $ qsec: num 16.5 17 18.6 19.4 17 ...
## $ vs : num 0 0 1 1 0 1 0 1 1 1 ...
## $ am : num 1 1 1 0 0 0 0 0 0 0 ...
## $ gear: num 4 4 4 3 3 3 3 4 4 4 ...
## $ carb: num 4 4 1 1 2 1 4 2 2 4 ...
```

파이프(%>%) 연산자

다음과 같이 함수 안에 함수를 반복하여 사용하면 코드의 가독성이 저하된다.

```
round(mean(head(mtcars$mpg,7)), 1)
```

```
## [1] 19.6
```

위의 코드를 단계별로 풀어쓰면 아래와 같이 3단계로 작성된다.

```
hd7 <- head(mtcars$mpg, 7)
mhd7 <- mean(hd7)
round(mhd7, 1)
```

```
## [1] 19.6
```

파이프(%>%) 연산자를 사용하는 이유는 간결한 코드, 이해하기 쉬운 코드의 작성이 가능하기 때문이다.
%>% 연산자를 사용하면 의식의 흐름대로 **가독성이 좋은 코드**를 작성할 수 있다.

```
mtcars$mpg %>%
  head(7) %>%
  mean() %>%
  round(1)
```

```
## [1] 19.6
```

dplyr 패키지의 함수에서 %>% 사용 예제

```
mtcars %>% arrange(mpg)      # mpg
mtcars %>%
  arrange(cyl, desc(mpg))    # cyl , mpg

mtcars %>%                    # mpg
  summarise(sum_mpg=sum(mpg), mean_mpg=mean(mpg))

mtcars %>%                    # cyl mpg
  group_by(cyl) %>%
  summarise(count=n(), mean_mpg=mean(mpg))
```

변수 변경 및 추가

외부 데이터 읽어오기

“exam.cav” 파일을 데이터프레임 df로 불러와 데이터와 구조를 확인한다.

```
str(df) #

## 'data.frame':   20 obs. of  7 variables:
## $ id      : int  20190001 20190002 20190003 20190004 20190005 20190006 20190007 20190008 20190009 20190010
## $ class   : Factor w/ 3 levels "A","B","C": 1 1 1 1 1 1 2 2 2 2 ...
## $ name    : Factor w/ 20 levels " ", " ", " ", ...: 2 13 19 7 11 8 1 16 15 10 ...
## $ gender  : Factor w/ 3 levels "F","M","X": 2 1 2 1 2 2 2 1 1 2 ...
## $       : int   50 60 45 84 25 100 80 90 20 50 ...
## $       : int   98 97 86 98 80 95 90 78 98 98 ...
## $       : int   50 60 78 58 65 98 85 25 15 45 ...
```

변수 변경

데이터 분석을 위해 변수명이나 변수의 자료형을 바꾸어야 하는 경우가 있다. ### 변수 이름 변경

변수명 변경을 위해서 dplyr 패키지에서 제공하는 **rename()** 함수를 사용한다.

데이터프레임 df의 ‘수학’, ‘영어’, ‘컴퓨터’ 변수명을 math, eng, com으로 변경해 보자.

```
df <- rename(df, math=" ", eng=" ", com=" ")
```

변수 자료형 변경

변수의 자료형을 변경하기 위해서 **as.xxxxx()** 함수를 사용한다.

데이터프레임 df에서 name 변수가 factor(요인)형으로 되어 있다. - **as.character()** 함수를 이용하여 문자형으로 변경한다. - name 변수에 변경 내용을 다시 저장해 주어야 자료형 변경이 완료된다.

```
df$name <- as.character(df$name)
```

변수 name의 자료형이 문자형으로 변경되었는지 확인한다.

```
str(df) #
```

```
## 'data.frame': 20 obs. of 7 variables:
## $ id : int 20190001 20190002 20190003 20190004 20190005 20190006 20190007 20190008 20190009 20190010
## $ class : Factor w/ 3 levels "A","B","C": 1 1 1 1 1 2 2 2 2 ...
## $ name : chr " " " " " " " " " " ...
## $ gender: Factor w/ 3 levels "F","M","X": 2 1 2 1 2 2 2 1 1 2 ...
## $ math : int 50 60 45 84 25 100 80 90 20 50 ...
## $ eng : int 98 97 86 98 80 95 90 78 98 98 ...
## $ com : int 50 60 78 58 65 98 85 25 15 45 ...
```

데이터프레임 df를 확인하자.

```
head(df)
```

```
##      id class  name gender math eng com
## 1 20190001    A      M   50  98  50
## 2 20190002    A      F   60  97  60
## 3 20190003    A      M   45  86  78
## 4 20190004    A      F   84  98  58
## 5 20190005    A      M   25  80  65
## 6 20190006    A      M  100  95  98
```

변수 추가하기

데이터를 분석을 위해 추가적인 변수가 필요한 경우 데이터프레임을 다시 만들지 않고 기존 데이터프레임에 변수를 추가할 수 있다. 새로운 변수를 생성하는 세 가지 방법을 소개한다.

- 기존 변수의 값을 조합하는 방법
- 조건문 함수를 활용하는 방법
- `mutate()` 함수 사용하는 방법

기존 변수 조합

기존 변수의 값을 조합하여 새로운 변수를 만들 수 있다.

먼저, 여러 방법을 사용할 것이므로 데이터프레임 df를 df_new로 복사한다.

기존 math, eng, com 변수의 값을 계산하여 새로운 변수 total과 avg를 만들 수 있다.

```
df_new$total <- df_new$math + df_new$eng + df_new$com
df_new$avg <- (df_new$total)/3
head(df_new)
```

```
##      id class  name gender math eng com total      avg
## 1 20190001    A      M   50  98  50  198 66.00000
## 2 20190002    A      F   60  97  60  217 72.33333
## 3 20190003    A      M   45  86  78  209 69.66667
## 4 20190004    A      F   84  98  58  240 80.00000
## 5 20190005    A      M   25  80  65  170 56.66667
## 6 20190006    A      M  100  95  98  293 97.66667
```

조건문 함수 활용

조건에 따라 특정 값이나 변수를 선택하는 경우 ifelse() 함수를 사용하여 새로운 변수를 생성할 수 있다.

- 조건이 한 개인 경우

```
df_new$test <- ifelse(df_new$total >= 200, "PASS", "FAIL")
```

- 조건이 두 개 이상이 경우

```
df_new$grade <- ifelse(df_new$avg >= 90, "A",  
                      ifelse(df_new$avg >= 80, "B",  
                              ifelse(df_new$avg >= 70, "C",  
                                      ifelse(df_new$avg >= 60, "D", "F") )))
```

데이터프레임을 확인해 보자.

```
head(df_new)
```

##	id	class	name	gender	math	eng	com	total	avg	test	grade
## 1	20190001	A		M	50	98	50	198	66.00000	FAIL	D
## 2	20190002	A		F	60	97	60	217	72.33333	PASS	C
## 3	20190003	A		M	45	86	78	209	69.66667	PASS	D
## 4	20190004	A		F	84	98	58	240	80.00000	PASS	B
## 5	20190005	A		M	25	80	65	170	56.66667	FAIL	F
## 6	20190006	A		M	100	95	98	293	97.66667	PASS	A

mutate() 함수 활용

mutate() 함수는 dplyr 패키지에서 제공하는 함수로 기존 변수를 사용하여 새로운 변수를 생성할 수 있다.

먼저, 데이터프레임 df를 df_new로 복사한다.

dplyr 패키지 함수이므로 데이터프레임명을 반복해서 입력하지 않아도 된다. math, eng, com 변수의 값을 더해 tot 변수를 추가 생성하고 다시 데이터프레임 df_new에 저장한다.

```
df_new <- df_new %>% mutate(total=math+eng+com)  
  
df_new %>%  
  mutate(avg=total/3,  
         test=ifelse(total >= 200, "PASS", "FAIL")) -> df_new  
  
df_new <- df_new %>%  
  mutate(grade = ifelse(avg >= 90, "A",  
                        ifelse(avg >= 80, "B",  
                                ifelse(avg >= 70, "C",  
                                        ifelse(avg >= 60, "D", "F") )))  
)
```

새로 생성한 변수가 데이터프레임에 추가되었는지 확인한다.

```
head(df_new)
```

```
##           id class  name gender math eng com total    avg test grade
## 1 20190001     A      M   50  98  50   198 66.00000 FAIL     D
## 2 20190002     A      F   60  97  60   217 72.33333 PASS     C
## 3 20190003     A      M   45  86  78   209 69.66667 PASS     D
## 4 20190004     A      F   84  98  58   240 80.00000 PASS     B
## 5 20190005     A      M   25  80  65   170 56.66667 FAIL     F
## 6 20190006     A      M  100  95  98   293 97.66667 PASS     A
```

[Tip] mutate()함수의 장점은 새로 생성한 변수를 또다른 신규 변수를 생성하는데 바로 사용할 수 있다는 점이다.

조건에 맞는 행만 추출

데이터 분석 시 전체 데이터를 분석하기도 하지만 일부 원하는 데이터만 추출해 분석하기도 한다. dplyr 패키지의 **filter()** 함수를 이용하여 원하는 데이터를 추출할 수 있다.

실습을 위해 데이터프레임 df를 exam로 복사한다.

filter() 안에 조건을 입력한다. 조건은 비교연산자나 논리연산으로 연산의 결과값이 TRUE 또는 FALSE이다. 입력된 조건에 부합되는 행만 추출한다.

- 비교연산자 : 같다(==), 같지 않다(!=), 크다(>), 크거나같다(>=), 작다(<), 작거나같다(<=)
 - 5 == 10 : 5와 10이 같은가? FALSE
 - 5 != 10 : 5와 10이 다른가? TRUE
- 논리연산자 : AND(&), OR(|)
 - 10 > 5 & 10 > 0 : 10이 5보다 크고 0보다 큰가? TRUE
 - 10 > 5 | 10 > 20 : 10이 5보다 크거나 20보다 큰가? TRUE

class가 'A'인 행 추출

```
exam %>% filter(class=='A')
```

```
##           id class  name gender math eng com
## 1 20190001     A      M   50  98  50
## 2 20190002     A      F   60  97  60
## 3 20190003     A      M   45  86  78
## 4 20190004     A      F   84  98  58
## 5 20190005     A      M   25  80  65
## 6 20190006     A      M  100  95  98
```

class가 'A'가 아닌 행 추출

```
exam %>% filter(class!='A')
```

##	id	class	name	gender	math	eng	com
## 1	20190007	B		M	80	90	85
## 2	20190008	B		F	90	78	25
## 3	20190009	B		F	20	98	15
## 4	20190010	B		M	50	98	45
## 5	20190011	B		M	65	65	65
## 6	20190012	B		F	45	85	32
## 7	20190013	B		M	46	98	65
## 8	20190014	C		M	48	87	12
## 9	20190015	C		M	75	56	78
## 10	20190016	C		M	88	98	65
## 11	20190017	C		X	65	68	98
## 12	20190018	C		M	80	78	56
## 13	20190019	C		F	89	68	87
## 14	20190020	C		F	100	83	58

math가 80 이상인 행 추출

```
exam %>% filter(math >= 80)
```

##	id	class	name	gender	math	eng	com
## 1	20190004	A		F	84	98	58
## 2	20190006	A		M	100	95	98
## 3	20190007	B		M	80	90	85
## 4	20190008	B		F	90	78	25
## 5	20190016	C		M	88	98	65
## 6	20190018	C		M	80	78	56
## 7	20190019	C		F	89	68	87
## 8	20190020	C		F	100	83	58

com이 40 미만인 행 추출

```
exam %>% filter(com < 40)
```

##	id	class	name	gender	math	eng	com
## 1	20190008	B		F	90	78	25
## 2	20190009	B		F	20	98	15
## 3	20190012	B		F	45	85	32
## 4	20190014	C		M	48	87	12

class가 'A'이고 math가 80 이상인 행 추출

```
exam %>% filter(class=="A" & math >= 80)
```

##	id	class	name	gender	math	eng	com
## 1	20190004	A		F	84	98	58
## 2	20190006	A		M	100	95	98

math, eng, com 모두 40 미만인 행 추출

```
exam %>% filter(math<40 & eng<40 & com<40)
```

```
## [1] id      class name  gender math  eng    com
## <0 rows> (or 0-length row.names)
```

math, eng, com 중 어느 하나라도 40 미만인 행 추출

```
exam %>% filter(math<40 | eng<40 | com<40)
```

```
##           id class  name gender math eng com
## 1 20190005     A      M   25  80  65
## 2 20190008     B      F   90  78  25
## 3 20190009     B      F   20  98  15
## 4 20190012     B      F   45  85  32
## 5 20190014     C      M   48  87  12
```

class가 'A'와 'C'인 행 추출

```
exam %>% filter(class %in% c('A', 'C'))
```

```
##           id class  name gender math eng com
## 1 20190001     A      M   50  98  50
## 2 20190002     A      F   60  97  60
## 3 20190003     A      M   45  86  78
## 4 20190004     A      F   84  98  58
## 5 20190005     A      M   25  80  65
## 6 20190006     A      M  100  95  98
## 7 20190014     C      M   48  87  12
## 8 20190015     C      M   75  56  78
## 9 20190016     C      M   88  98  65
## 10 20190017     C      X   65  68  98
## 11 20190018     C      M   80  78  56
## 12 20190019     C      F   89  68  87
## 13 20190020     C      F  100  83  58
```

조건에 맞는 행만 추출하여 새로운 데이터프레임 classA 생성

```
classA <- exam %>% filter(class=='A')
classA
```

```
##           id class  name gender math eng com
## 1 20190001     A      M   50  98  50
## 2 20190002     A      F   60  97  60
## 3 20190003     A      M   45  86  78
## 4 20190004     A      F   84  98  58
## 5 20190005     A      M   25  80  65
## 6 20190006     A      M  100  95  98
```


일부 변수만 추출

`select()` 안에 변수명을 입력하여 원하는 변수(열)만 추출한다.

- 특정 변수를 제외하고 추출할 때 변수명 앞에 `-`를 기입
- `i` 번째 부터 `j` 번째 변수 추출할 때 `:`을 사용

```
head(exam, 10)      # id, class, name, gender, math, eng, com
```

```
##           id class  name gender math eng com
## 1  20190001     A      M   50  98  50
## 2  20190002     A      F   60  97  60
## 3  20190003     A      M   45  86  78
## 4  20190004     A      F   84  98  58
## 5  20190005     A      M   25  80  65
## 6  20190006     A      M  100  95  98
## 7  20190007     B      M   80  90  85
## 8  20190008     B      F   90  78  25
## 9  20190009     B      F   20  98  15
## 10 20190010     B      M   50  98  45
```

math 변수만 추출

```
exam %>% select(math)
```

```
##      math
## 1      50
## 2      60
## 3      45
## 4      84
## 5      25
## 6     100
## 7      80
## 8      90
## 9      20
## 10     50
## 11     65
## 12     45
## 13     46
## 14     48
## 15     75
## 16     88
## 17     65
## 18     80
## 19     89
## 20    100
```

id, gender, com 변수만 추출

```
exam %>% select(id, gender, com)
```

```
##           id gender com
## 1  20190001      M  50
## 2  20190002      F  60
## 3  20190003      M  78
## 4  20190004      F  58
## 5  20190005      M  65
## 6  20190006      M  98
## 7  20190007      M  85
## 8  20190008      F  25
## 9  20190009      F  15
## 10 20190010      M  45
## 11 20190011      M  65
## 12 20190012      F  32
## 13 20190013      M  65
## 14 20190014      M  12
## 15 20190015      M  78
## 16 20190016      M  65
## 17 20190017      X  98
## 18 20190018      M  56
## 19 20190019      F  87
## 20 20190020      F  58
```

math, eng, com 변수처럼 인접해 있는 변수 추출

```
exam %>% select(class:eng)
```

```
##   class  name gender math eng
## 1     A      M    50  98
## 2     A      F    60  97
## 3     A      M    45  86
## 4     A      F    84  98
## 5     A      M    25  80
## 6     A      M   100  95
## 7     B      M    80  90
## 8     B      F    90  78
## 9     B      F    20  98
## 10    B      M    50  98
## 11    B      M    65  65
## 12    B      F    45  85
## 13    B      M    46  98
## 14    C      M    48  87
## 15    C      M    75  56
## 16    C      M    88  98
## 17    C      X    65  68
## 18    C      M    80  78
## 19    C      F    89  68
## 20    C      F   100  83
```

```
exam %>% select(5:7) # 5 ~ 7
```

```
##      math eng com
## 1      50  98  50
## 2      60  97  60
## 3      45  86  78
## 4      84  98  58
## 5      25  80  65
## 6     100  95  98
## 7      80  90  85
## 8      90  78  25
## 9      20  98  15
## 10     50  98  45
## 11     65  65  65
## 12     45  85  32
## 13     46  98  65
## 14     48  87  12
## 15     75  56  78
## 16     88  98  65
## 17     65  68  98
## 18     80  78  56
## 19     89  68  87
## 20    100  83  58
```

math 변수만 제외하고 추출

```
exam %>% select(-math)
```

```
##      id class  name gender eng com
## 1 20190001    A      M   98  50
## 2 20190002    A      F   97  60
## 3 20190003    A      M   86  78
## 4 20190004    A      F   98  58
## 5 20190005    A      M   80  65
## 6 20190006    A      M   95  98
## 7 20190007    B      M   90  85
## 8 20190008    B      F   78  25
## 9 20190009    B      F   98  15
## 10 20190010   B      M   98  45
## 11 20190011   B      M   65  65
## 12 20190012   B      F   85  32
## 13 20190013   B      M   98  65
## 14 20190014   C      M   87  12
## 15 20190015   C      M   56  78
## 16 20190016   C      M   98  65
## 17 20190017   C      X   68  98
## 18 20190018   C      M   78  56
## 19 20190019   C      F   68  87
## 20 20190020   C      F   83  58
```

id, gender, class 변수 제외하고 추출

```
exam %>% select(-id, -gender, -class)
```

```
##      name math eng com
## 1      50  98  50
## 2      60  97  60
## 3      45  86  78
## 4      84  98  58
## 5      25  80  65
## 6     100  95  98
## 7      80  90  85
## 8      90  78  25
## 9      20  98  15
## 10     50  98  45
## 11     65  65  65
## 12     45  85  32
## 13     46  98  65
## 14     48  87  12
## 15     75  56  78
## 16     88  98  65
## 17     65  68  98
## 18     80  78  56
## 19     89  68  87
## 20    100  83  58
```

데이터프레임 정렬

특정 변수를 기준으로 정렬할 때 `arrange()` 함수를 이용한다.

여러 개의 기준에 의해 정렬하려면 기준이 되는 변수들을 순서대로 나열한다.

`arrange()` 함수는 기본적으로 오름차순으로 정렬된다. 내림차순으로 정렬하려면 `desc()`을 사용한다.

math 변수값에 따라 오름차순 정렬

```
exam %>% arrange(math)
```

```
##      id class  name gender math eng com
## 1 20190009    B      F   20  98  15
## 2 20190005    A      M   25  80  65
## 3 20190003    A      M   45  86  78
## 4 20190012    B      F   45  85  32
## 5 20190013    B      M   46  98  65
## 6 20190014    C      M   48  87  12
## 7 20190001    A      M   50  98  50
## 8 20190010    B      M   50  98  45
## 9 20190002    A      F   60  97  60
## 10 20190011   B      M   65  65  65
## 11 20190017   C      X   65  68  98
## 12 20190015   C      M   75  56  78
## 13 20190007   B      M   80  90  85
```

```
## 14 20190018      C      M   80  78  56
## 15 20190004      A      F   84  98  58
## 16 20190016      C      M   88  98  65
## 17 20190019      C      F   89  68  87
## 18 20190008      B      F   90  78  25
## 19 20190006      A      M  100  95  98
## 20 20190020      C      F  100  83  58
```

math 변수값에 따라 내림차순 정렬

```
exam %>% arrange(desc(math))
```

```
##           id class  name gender math eng com
## 1  20190006      A      M   100  95  98
## 2  20190020      C      F   100  83  58
## 3  20190008      B      F    90  78  25
## 4  20190019      C      F    89  68  87
## 5  20190016      C      M   88  98  65
## 6  20190004      A      F   84  98  58
## 7  20190007      B      M   80  90  85
## 8  20190018      C      M   80  78  56
## 9  20190015      C      M   75  56  78
## 10 20190011      B      M   65  65  65
## 11 20190017      C      X   65  68  98
## 12 20190002      A      F   60  97  60
## 13 20190001      A      M   50  98  50
## 14 20190010      B      M   50  98  45
## 15 20190014      C      M   48  87  12
## 16 20190013      B      M   46  98  65
## 17 20190003      A      M   45  86  78
## 18 20190012      B      F   45  85  32
## 19 20190005      A      M   25  80  65
## 20 20190009      B      F    20  98  15
```

class 순, math 순으로 오름차순 정렬

```
exam %>% arrange(class, math)
```

```
##           id class  name gender math eng com
## 1  20190005      A      M    25  80  65
## 2  20190003      A      M   45  86  78
## 3  20190001      A      M   50  98  50
## 4  20190002      A      F   60  97  60
## 5  20190004      A      F   84  98  58
## 6  20190006      A      M  100  95  98
## 7  20190009      B      F    20  98  15
## 8  20190012      B      F   45  85  32
## 9  20190013      B      M   46  98  65
## 10 20190010      B      M   50  98  45
## 11 20190011      B      M   65  65  65
## 12 20190007      B      M   80  90  85
```

## 13	20190008	B	F	90	78	25
## 14	20190014	C	M	48	87	12
## 15	20190017	C	X	65	68	98
## 16	20190015	C	M	75	56	78
## 17	20190018	C	M	80	78	56
## 18	20190016	C	M	88	98	65
## 19	20190019	C	F	89	68	87
## 20	20190020	C	F	100	83	58

유일한 값 추출

`distinct()` 함수를 이용하여 중복되지 않는 유일한 값을 추출한다.
두 개 이상의 변수를 기준으로 중복되지 않는 값을 검색할 수도 있다.

exam의 변수 class에서 중복되지 않는 값 추출

```
exam %>% distinct(class)
```

```
##   class
## 1     A
## 2     B
## 3     C
```

exam에서 class, gender 두 변수를 기준으로 중복되지 않는 값 추출

```
exam %>% distinct(class, gender)
```

```
##   class gender
## 1     A      M
## 2     A      F
## 3     B      M
## 4     B      F
## 5     C      M
## 6     C      X
## 7     C      F
```