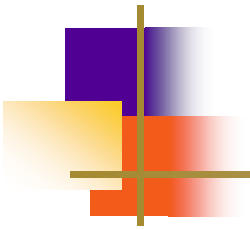
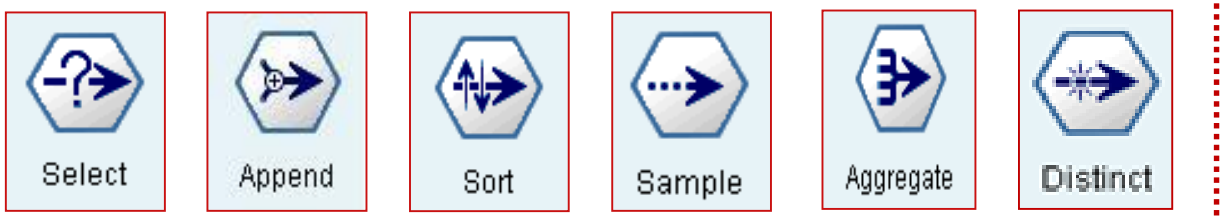


Data Munging with R (1)

Types of Data Munging

Record(Row) Operations



Field(Column) Operations

Data Munging with dplyr package

- # dplyr is a powerful R-package to transform and summarize tabular data with rows and columns.
- # The package contains a set of functions that perform common data manipulation operations such as filtering for rows, selecting specific columns, re-ordering rows, adding new columns and summarizing data.
- # In addition, dplyr contains a useful function to perform another common task which is the “split-apply-combine” concept.
- # For more details, refer to the next link: https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html

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

Selects a subset of records based on a specified condition

	A	B	C	D
1	ID	Exam1	Exam2	Quiz
2	1	3.3	2	3.7
3	2	4	4	4
4	3	2.3	0	3.3
5	4	2.2	1	3.3
6	5	3.1	1.2	3.9

exam1.csv

```
➢  filter(exam1, Exam2 >= 1 & Quiz < 3.9)
  ID Exam1 Exam2 Quiz
1  1   3.3     2  3.7
4  4   2.2     1  3.3

➢  filter(exam1, Exam2 >=1, Quiz < 3.9)
  ID Exam1 Exam2 Quiz
1  1   3.3     2  3.7
4  4   2.2     1  3.3
```

Exam1과 Exam2 둘 다 평균 이상인 학생은?

Appends records from multiple inputs

	A	B	C	D
1	ID	Exam1	Exam2	Quiz
2	1	3.3	2	3.7
3	2	4	4	4
4	3	2.3	0	3.3
5	4	2.2	1	3.3
6	5	3.1	1.2	3.9

- `app <- c(6,3.5,1.5,3.5)`
- `rbind(exam1,app)`

```
  ID Exam1 Exam2 Quiz
1  1   3.3   2.0  3.7
2  2   4.0   4.0  4.0
3  3   2.3   0.0  3.3
4  4   2.2   1.0  3.3
5  5   3.1   1.2  3.9
6  6   3.5   1.5  3.5
```

변수 **app**의 값이 **c(6,1)**이라면 결과는?

Reorders records according to the specified order criteria

	A	B	C	D
1	ID	Exam1	Exam2	Quiz
2	1	3.3	2	3.7
3	2	4	4	4
4	3	2.3	0	3.3
5	4	2.2	1	3.3
6	5	3.1	1.2	3.9

```
➢  arrange(exam1, Quiz)      # Use desc() in descending order
  ID Exam1 Exam2 Quiz
3  3   2.3   0.0  3.3
4  4   2.2   1.0  3.3
1  1   3.3   2.0  3.7
5  5   3.1   1.2  3.9
2  2   4.0   4.0  4.0
```

 # Quiz 점수가 높은 사람부터 내림차순으로 정렬한다면?
 # Quiz와 Exam1 순서로 오름차순으로 정렬한다면?

Selects a random sample

	A	B	C	D
1	ID	Exam1	Exam2	Quiz
2	1	3.3	2	3.7
3	2	4	4	4
4	3	2.3	0	3.3
5	4	2.2	1	3.3
6	5	3.1	1.2	3.9

```
➢  sample_n(exam1, 3)          # Random sampling with a fixed number
  ID Exam1 Exam2 Quiz
5  5   3.1   1.2  3.9
1  1   3.3   2.0  3.7
4  4   2.2   1.0  3.3

➢  sample_frac(exam1, 0.4)   # Random sampling with  a fixed fraction

➢  exam1[as.logical((1:nrow(exam1))%%2),]        # 1-in-n sampling
  ID Exam1 Exam2 Quiz
1  1   3.3   2.0  3.7
3  3   2.3   0.0  3.3
5  5   3.1   1.2  3.9
```

Summarizes information on groups of records

	A	B	C	D	E
1	ID	Exam1	Exam2	Quiz	Gender
2	1	3.3	2	3.7	남
3	2	4	4	4	여
4	3	2.3	0	3.3	남
5	4	2.2	1	3.3	여
6	5	3.1	1.2	3.9	남

exam3.csv

```
➢  exam3 <- read.table(file="clipboard", sep="\t", header=T)

➢  by_gender = group_by(exam3, Gender)
➢  summarise(by_gender, exam1=mean(Exam1), quiz=median(Quiz))
  Gender exam1  quiz
1     남   2.9  3.70
2     여   3.1  3.65

➢  exam1 %>% group_by(Gender) %>%
 + summarise_each(funs(min,max), Exam1, Exam2, Quiz)
  Gender Exam1_min Exam2_min Quiz_min Exam1_max Exam2_max Quiz_max
1     남       2.3         0      3.3       3.3         2      3.9
2     여       2.2         1      3.3       4.0         4      4.0
```

Includes records with distinct values in specified fields

	A	B	C	D
1	ID	Exam1	Exam2	Quiz
2	1	3.3	2	3.7
3	2	4	4	4
4	3	2.3	0	3.3
5	4	2.2	1	3.3
6	5	3.1	1.2	3.9

duplicated

```
➢  filter(exam1,!duplicated(Quiz))
  ID Exam1 Exam2 Quiz
1  1   3.3   2.0  3.7
2  2   4.0   4.0  4.0
3  3   2.3   0.0  3.3
5  5   3.1   1.2  3.9
```

```
➢  distinct(exam1, Quiz)     # return unique values
  Quiz
1  3.7
2  4.0
3  3.3
4  3.9
```

Allows new fields to be generated based on existing fields

	A	B	C	D
1	ID	Exam1	Exam2	Quiz
2	1	3.3	2	3.7
3	2	4	4	4
4	3	2.3	0	3.3
5	4	2.2	1	3.3
6	5	3.1	1.2	3.9

```
> exam1 <- mutate(exam1, ExamSum=Exam1+Exam2, ExamMean=ExamSum/2)
> exam1
  ID Exam1 Exam2 Quiz ExamSum ExamMean
1  1   3.3   2.0  3.7     5.3     2.65
2  2   4.0   4.0  4.0     8.0     4.00
3  3   2.3   0.0  3.3     2.3     1.15
4  4   2.2   1.0  3.3     3.2     1.60
5  5   3.1   1.2  3.9     4.3     2.15
```

Allows fields to be renamed or removed

```
➢  select(exam1, ID:Exam2)
 ID Exam1 Exam2
1  1   3.3   2.0
2  2   4.0   4.0
3  3   2.3   0.0
4  4   2.2   1.0
5  5   3.1   1.2

➢  exam1 <- select(exam1, -ExamSum, -ExamMean); exam1
  ID Exam1 Exam2 Quiz
1  1   3.3   2.0  3.7
2  2   4.0   4.0  4.0
3  3   2.3   0.0  3.3
4  4   2.2   1.0  3.3
5  5   3.1   1.2  3.9

➢  rename(exam1, id=ID, quiz = Quiz, ex1=Exam1, ex2=Exam2)
  id ex1 ex2 quiz
1  1 3.3 2.0  3.7
2  2 4.0 4.0  4.0
3  3 2.3 0.0  3.3
4  4 2.2 1.0  3.3
5  5 3.1 1.2  3.9
```

Changes the sort order of fields

```
➢  select(exam1, ID, Quiz, Exam1:Exam2)
  ID Quiz Exam1 Exam2
1  1  3.7   3.3   2.0
2  2  4.0   4.0   4.0
3  3  3.3   2.3   0.0
4  4  3.3   2.2   1.0
5  5  3.9   3.1   1.2
```

Allows values in existing fields to be replaced by new values

```
> exam1$Extra <- c(1, 1, NA, NA, 2); exam1
  ID Exam1 Exam2 Quiz Extra
1  1   3.3   2.0  3.7     1
2  2   4.0   4.0  4.0     1
3  3   2.3   0.0  3.3    NA
4  4   2.2   1.0  3.3    NA
5  5   3.1   1.2  3.9     2

> exam1$Extra[is.na(exam1$Extra)] <- 0
  ID Exam1 Exam2 Quiz Extra
1  1   3.3   2.0  3.7     1
2  2   4.0   4.0  4.0     1
3  3   2.3   0.0  3.3     0
4  4   2.2   1.0  3.3     0
5  5   3.1   1.2  3.9     2
```

Merges records from multiple inputs

	A	B	C	D
1	ID	Exam1	Exam2	Quiz
2	1	3.3	2	3.7
3	2	4	4	4
4	3	2.3	0	3.3
5	4	2.2	1	3.3
6	5	3.1	1.2	3.9

exam1.csv

	A	B	C	D
1	CID	Exam3	Exam4	FinalExam
2	1	3.1	2.2	3.5
3	2	3.9	3.9	4
4	3	2.2	1.1	3.7
5	4	2.1	1	3.3
6	5	3	1.1	3.8

exam2.csv

```
➢ merge(exam1, exam2, by.x="ID", by.y="CID")
  ID Exam1 Exam2 Quiz Exam3 Exam4 FinalExam
1  1   3.3   2.0  3.7   3.1   2.2       3.5
2  2   4.0   4.0  4.0   3.9   3.9       4.0
3  3   2.3   0.0  3.3   2.2   1.1       3.7
4  4   2.2   1.0  3.3   2.1   1.0       3.3
5  5   3.1   1.2  3.9   3.0   1.1       3.8
```

Transposes records to fields and fields to records

	A	B	C	D
1	ID	Exam1	Exam2	Quiz
2	1	3.3	2	3.7
3	2	4	4	4
4	3	2.3	0	3.3
5	4	2.2	1	3.3
6	5	3.1	1.2	3.9

```
> t(exam1)
         [,1] [,2] [,3] [,4] [,5]
ID       1.00    2 3.00  4.0 5.00
Exam1    3.30    4 2.30  2.2 3.10
Exam2    2.00    4 0.00  1.0 1.20
Quiz     3.70    4 3.30  3.3 3.90
ExamSum  5.30    8 2.30  3.2 4.30
ExamMean 2.65    4 1.15  1.6 2.15
```

```
> t(t(exam1))      # use as.data.frame() to coerce it to a data frame
     ID Exam1 Exam2 Quiz ExamSum ExamMean
[1,]  1   3.3   2.0  3.7     5.3     2.65
[2,]  2   4.0   4.0  4.0     8.0     4.00
[3,]  3   2.3   0.0  3.3     2.3     1.15
[4,]  4   2.2   1.0  3.3     3.2     1.60
[5,]  5   3.1   1.2  3.9     4.3     2.15
```

Creates new fields from one or more categorical fields - Averaging values

	A	B	C	D	E
1	ID	Exam1	Exam2	Quiz	Gender
2	1	3.3	2	3.7	남
3	2	4	4	4	여
4	3	2.3	0	3.3	남
5	4	2.2	1	3.3	여
6	5	3.1	1.2	3.9	남

```
➢ tapply(exam3$Quiz, exam3$Gender, sum)
  남   여
10.9  7.3

➢ tapply(exam3[,2], exam3$Gender, mean)
 남  여
2.9 3.1
```

Creates new fields from one or more categorical fields - Melting & Casting (1/3)

```
tr <- read.table(text="
        id      site    pageview  dwelltime
1       1       a       1         7
2       1       b       2         6
3       1       c       3         5
4       1       a       4         4
5       2       a       5         3
6       2       b       6         2
7       2       b       7         1")

library(reshape)
tr.melt <- melt(tr, id.vars=c("id","site"),
measure.vars=c("pageview","dwelltime"))
```

id.var을 기준으로 데이터를 아래로 펼침

Creates new fields from one or more categorical fields - Melting & Casting (2/3)

```
> tr.melt
   id site  variable value
1   1    a  pageview     1
2   1    b  pageview     2
3   1    c  pageview     3
4   1    a  pageview     4
5   2    a  pageview     5
6   2    b  pageview     6
7   2    b  pageview     7
8   1    a dwelltime     7
9   1    b dwelltime     6
10  1    c dwelltime     5
11  1    a dwelltime     4
12  2    a dwelltime     3
13  2    b dwelltime     2
14  2    b dwelltime     1
```

```
> cast(tr.melt, id ~ site, sum, subset=variable=="pageview")
  id a  b c
1  1 5  2 3
2  2 5 13 0
```

formular=var1~var2:
var1의 level을 행으로 var2의 level을 열 방향으로 설정해 value의 값을 function으로 집계

Creates new fields from one or more categorical fields - Melting & Casting (3/3)

```
➢  cast(tr.melt, id+site~variable, length)
  id site pageview dwelltime
1  1    a        2         2
2  1    b        1         1
3  1    c        1         1
4  2    a        1         1
5  2    b        2         2

➢  cast(tr.melt, id ~ variable, mean, subset=variable=="pageview")
  id pageview
1  1      2.5
2  2      6.0
```

 # summarize()함수를 사용하여 위 코드와 동일한 결과를 얻을 수 있을까?

```
➢  cast(tr.melt, id ~ variable, range)
  id pageview_X1 pageview_X2 dwelltime_X1 dwelltime_X2
1  1           1           4           4           7
2  2           5           7           1           3
```

Converts numeric fields into discrete pieces

	A	B	C	D
1	ID	Exam1	Exam2	Quiz
2	1	3.3	2	3.7
3	2	4	4	4
4	3	2.3	0	3.3
5	4	2.2	1	3.3
6	5	3.1	1.2	3.9

```
> exam1 <- mutate(exam1, ExamSum=Exam1+Exam2)

> exam1$Level <- cut(exam1$ExamSum,breaks=3,labels=F); exam1
  ID Exam1 Exam2 Quiz ExamSum Level
1  1   3.3   2.0  3.7     5.3     2
2  2   4.0   4.0  4.0     8.0     3
3  3   2.3   0.0  3.3     2.3     1
4  4   2.2   1.0  3.3     3.2     1
5  5   3.1   1.2  3.9     4.3     2

> exam1$Level <- cut(exam1$ExamSum,c(0,2,4,6,8),labels=F); exam1
  ID Exam1 Exam2 Quiz ExamSum Level
1  1   3.3   2.0  3.7     5.3     3
2  2   4.0   4.0  4.0     8.0     4
3  3   2.3   0.0  3.3     2.3     2
4  4   2.2   1.0  3.3     3.2     2
5  5   3.1   1.2  3.9     4.3     3

> class(exam1$Level)
[1] "factor"
```