R 프로그래밍 기초

한국고용정보원
민종열 대리

한국고용정보원
Korea Employment Information Service

PROFILE

민종열 대리

통계학과 졸업

한국고용정보원 빅데이터분석TF팀 재직 중

빅데이터 기획 및 분석 업무 담당

원내 빅데이터 분석 프로젝트 진행

CONTENTS

Chapter

I

빅데이터 분석 절차

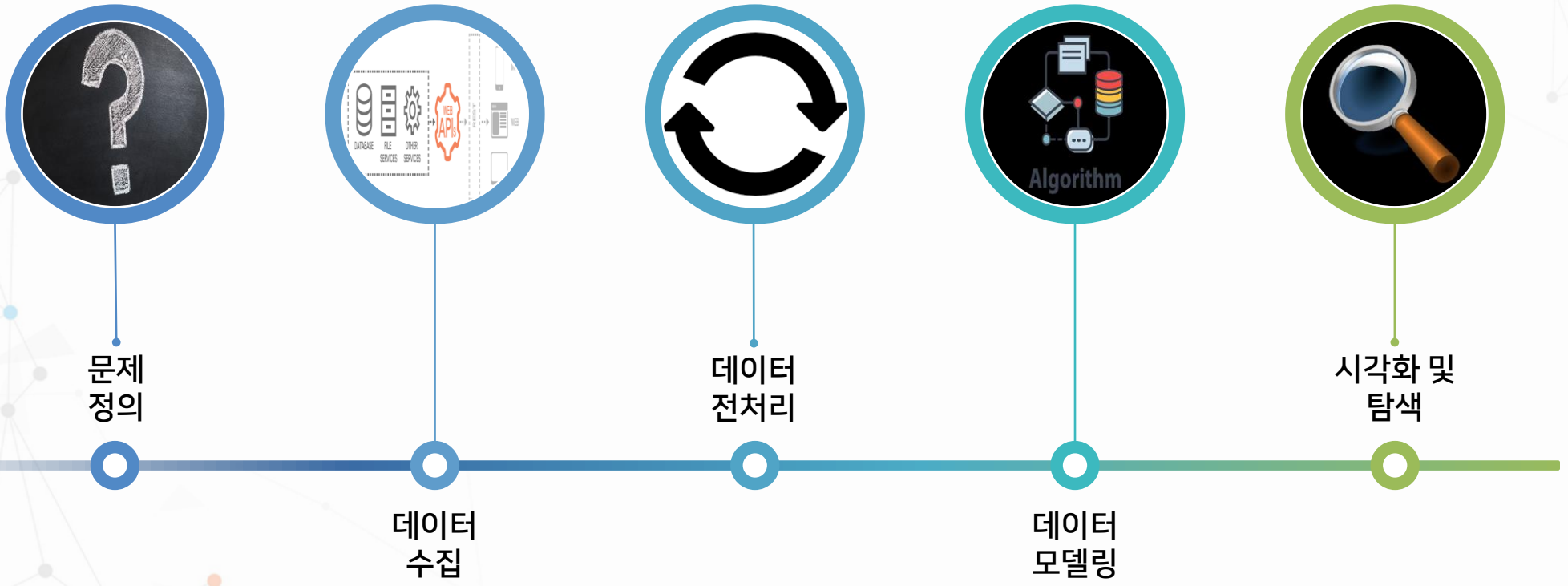
01 이론 : 확증적 데이터 분석(CDA)

1	2	3	4
가설 설정	**데이터 수집**	**통계 분석**	**가설 검증**
엄마가 키가 크면 딸이 키가 크다	엄마와 딸의 키 데이터	회귀분석, 상관관계	O / X
도로의 CCTV가 많으면 범죄예방 효과가 있다	도로 데이터, CCTV 데이터	상관관계, CCTV설치 전후 범죄발생율	O / X
퇴직금 화면을 많이 보는 사람이 퇴직할 확률이 높다	퇴직금 화면 조회 이력, 회사 임직원 정보	회귀분석, 상관관계	O / X
매장에 머무르는 시간이 높을 수록 매출이 높다.	매정 체류 시간, 매출 데이터	통계적 분석 기법	O / X

01 이론 : 탐색적 데이터 분석(EDA)

1	2	3	4
데이터 수집	**시각화/탐색**	**패턴 도출**	**인사이트 발견**
소비 데이터	소비 대시보드	장바구니 분석	남자들이 기저귀를 사러 와서 맥주를 사는구나
생존 데이터	생존 대시보드	생존 분석	어떤 음식을 먹는 사람이 암에 덜걸린다
매출 데이터	매출 대시보드	상권 분석	돈까스 집은 대학 주변에 서만 잘되는구나
인사 데이터	인사 대시보드	직무 분석	어떤 성격을 가진 사람이 해당 직무를 잘 수행한다

01 실제 빅데이터 분석 업무 절차



01 실제 : 빅데이터 분석 업무 절차

§ 빅데이터 분석 과정

1	문제 정의	1. 분석 대상의 이해 (고매출 매장) 2. 객관적이고 구체적으로 분석 대상 정의 (고매출 매장의 특징)
2	데이터 수집	1. 필요한 데이터 요건 정의 (사람, 매장의 매출 데이터가 필요) 2. 데이터 소재 파악 및 확보 (어떤 팀에서 데이터를 가지고 있나?)
3	데이터 전처리	1. 오류 사항 점검 및 조치 (데이터를 보니까 이상한게 있네?) 2. 데이터 구조 및 특성 변경 (모델링을 하기 위해서는 구조와 특성을 좀 바꿔야 겠다)
4	데이터 모델링	1. 다양한 알고리즘 구축 (다양한 알고리즘을 돌려보자) 2. 평가 및 선정 (그 중에 1번 알고리즘이 제일 좋네)
5	시각화 및 탐색	1. 결과 해석 (고매출 매장은 이런 손님이 많구나) 2. 시각화 (사장님 고매출 매장은 이런 손님이 많아요)

데이터 분석 전과정

Chapter

II

R 자료 형태 및 객체

02 활용 프로그램

Colab

§ Google에서 제공하는 분석 가상 환경

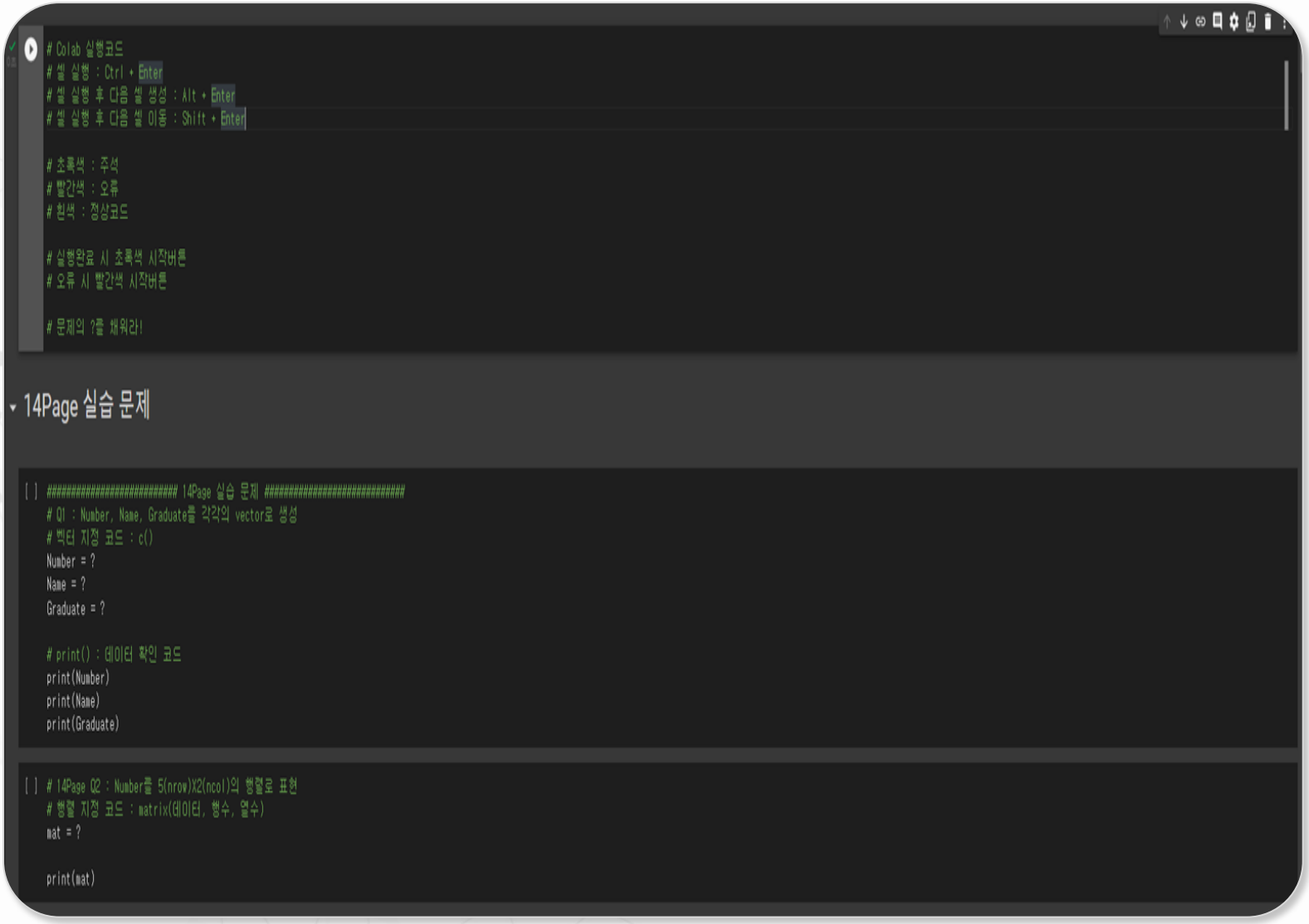
§ 현재 Python & R 사용 가능

※ R의 경우 (해당 URL 사용 필요)

https://colab.research.google.com/notebook#create=true&language=r

§ OneDrive와 동기화 가능

02 Colab 사용법



```
# Colab 실행코드
# 셀 실행 : Ctrl + Enter
# 셀 실행 후 다음 셀 생성 : Alt + Enter
# 셀 실행 후 다음 셀 이동 : Shift + Enter

# 초록색 : 주석
# 빨간색 : 오류
# 흰색 : 정상코드

# 실행완료 시 초록색 시작버튼
# 오류 시 빨간색 시작버튼

# 문제의 ?를 채워라!
```

14Page 실습 문제

```
[ ] ######################## 14Page 실습 문제 ########################
    # Q1 : Number, Name, Graduate를 각각의 vector로 생성
    # 벡터 지정 코드 : c()
    Number = ?
    Name = ?
    Graduate = ?

    # print() : 데이터 확인 코드
    print(Number)
    print(Name)
    print(Graduate)
```

```
[ ] # 14Page Q2 : Number를 5(nrow)X2(ncol)의 행렬로 표현
    # 행렬 지정 코드 : matrix(데이터, 행수, 열수)
    mat = ?

    print(mat)
```

Colab 환경 설명

§ Colab은 각각의 셀을 실행하는 형태로 코드를 실행

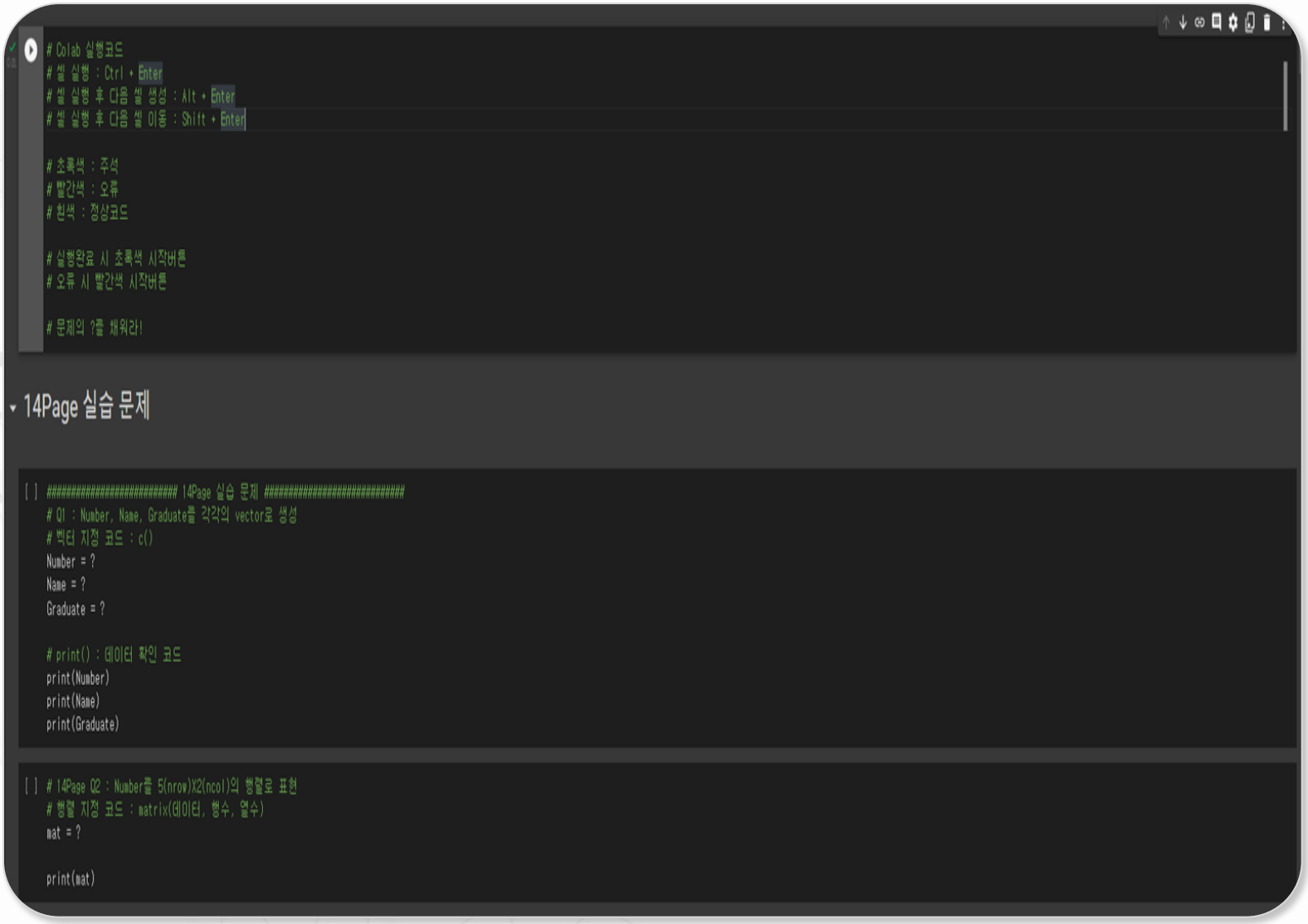
§ 실행 명령어

→ Ctrl + Enter : 해당셀 실행

→ Alt + Enter : 해당셀 실행 후 추가 셀 생성

→ Shift + Enter : 해당셀 실행 후 다음 셀로 커서 옮겨감

02 Colab 사용법



```
# Colab 실행코드
# 셀 실행 : Ctrl + Enter
# 셀 실행 후 다음 셀 생성 : Alt + Enter
# 셀 실행 후 다음 셀 이동 : Shift + Enter

# 초록색 : 주석
# 빨간색 : 오류
# 흰색 : 정상코드

# 실행완료 시 초록색 시작버튼
# 오류 시 빨간색 시작버튼

# 문제의 ?를 채워라!
```

14Page 실습 문제

```
[ ] ######################## 14Page 실습 문제 ########################
    # Q1 : Number, Name, Graduate를 각각의 vector로 생성
    # 벡터 지정 코드 : c()
    Number = ?
    Name = ?
    Graduate = ?

    # print() : 데이터 확인 코드
    print(Number)
    print(Name)
    print(Graduate)
```

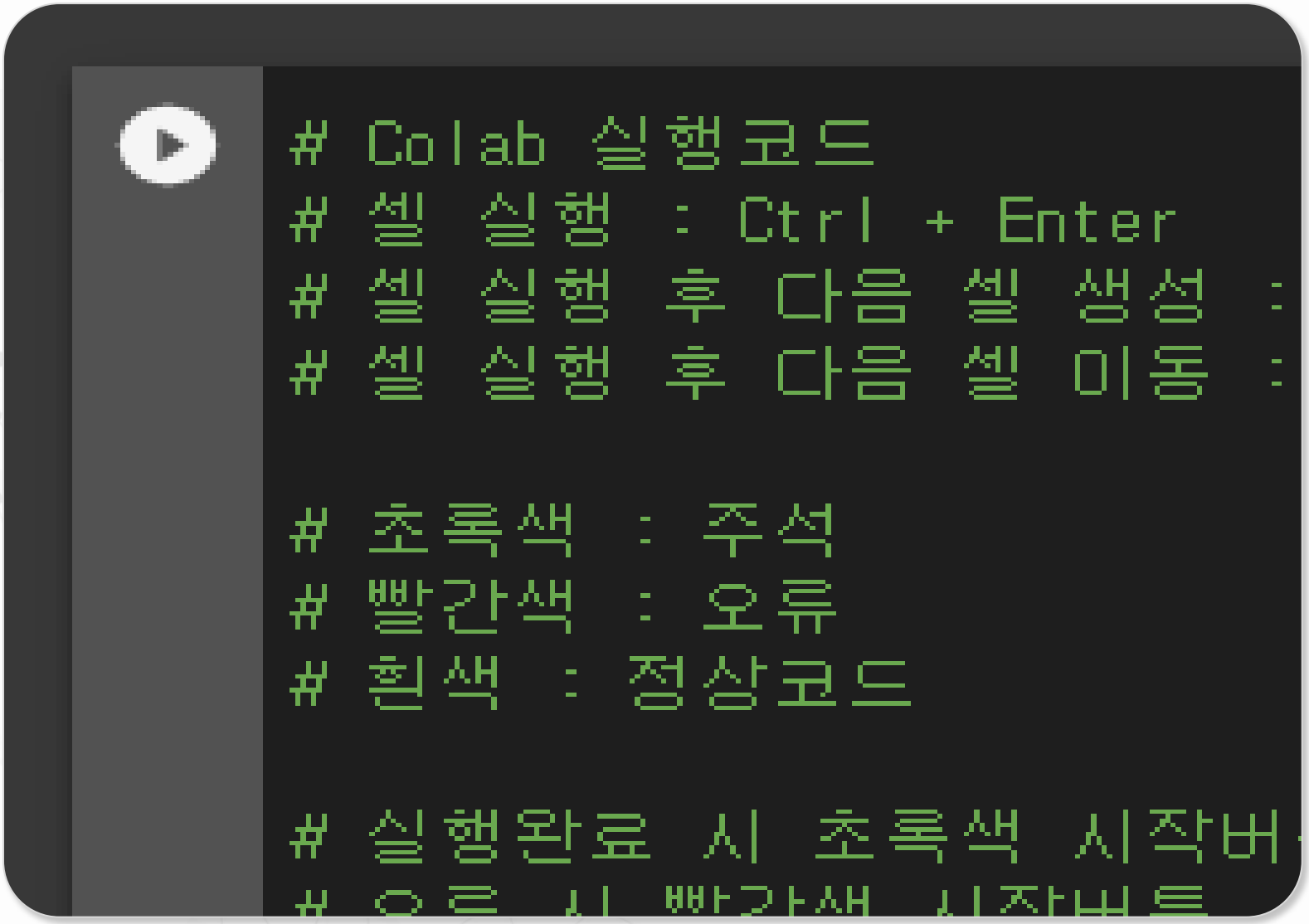
```
[ ] # 14Page Q2 : Number를 5(nrow)X2(ncol)의 행렬로 표현
    # 행렬 지정 코드 : matrix(데이터, 행수, 열수)
    mat = ?

    print(mat)
```

코드 색깔 의미

- 초록색 코드 : 주석 ※ 실행해도 아무것도 구동되지 않음
- 하얀색 코드 : 정상 코드
- 붉은색 코드 : 비정상 코드 ※ 작동시 오류
- 갈색 코드 : 문자형 코드 ex) 'A'

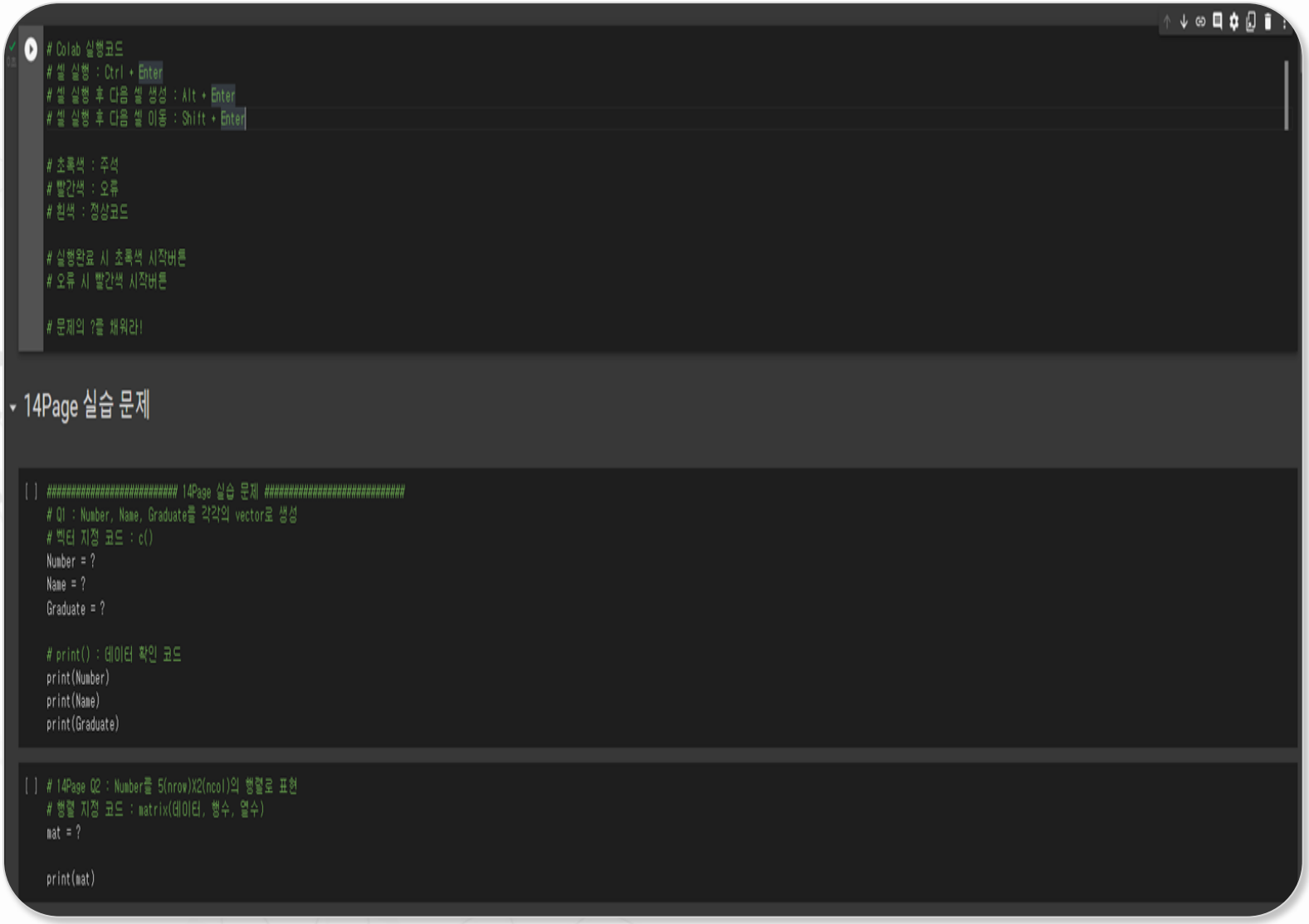
02 Colab 사용법



실행 버튼 색깔 의미

- 흰색 : 실행되지 않은 셀
- 초록색 : 정상적으로 실행된 셀
- 붉은색 : 오류가 발생된 셀

02 실습 진행 방식



실습 진행 방식

- ? 표시가 된 곳에 코드를 작성하고 셀 실행 (Ctrl + Enter)
- 해당 셀이 제대로 실행되는지(초록색 실행 버튼) 확인 후 다음 문제로 이동
- 어떤 코드를 사용해야 될 지 모를 때 hint 활용
- 정답은 밑에 코드로 제공

- print(객체) : 해당 객체를 확인하기 위한 코드
- head(객체) : 해당 객체의 5번까지의 데이터만 보여줌

코딩에 들어가기에 앞서!

1 겁먹지 말기

§ 코딩은 어려운 것이 아니다!

2 이론 수업을 듣고 실습

§ 실습용 문제를 준비했습니다.

3 엑셀로 대입해서 생각하기

§ 엑셀은 다들 잘 알잖아요

4 프로그램이랑 카톡하듯

§ 코드의 내용을 이해해야 해요

02 R 자료형태 및 객체

1	2	3	4
Numeric(숫자)	**Character(문자)**	**Logical(논리형)**	**Factor(요인)**
정수(1,2,3)(num)	문자(apple, banana)(chr)	True/False(logi)	명목형("F","M") (fac)
실수(1.3,1.4)(int)	문장(apple is good)(chr)	숫자 1,0	순서형("A","B","C") (fac)
R 코드) a = 1 a <- 1	R 코드) a = 'hello world' a <- 'hello world'	R 코드) a = True a = T	R코드) a = factor(1) a = factor('1')

Q) a = '1' 은 어떤 자료형태일까?

02 R 자료형태 및 객체

벡터(vector)

§ 벡터는 하나 이상의 원소로 이루어진 자료
§ 벡터를 구성하는 원소는 유형이 동일

§ R코드)
§ a = c(1,2,3,4) # 숫자형
§ a = c('1','2','3','4') # 문자형
§ a = c(factor(1), factor(2)) # factor형
§ a = c(T,F) # 논리형

행렬(matrix)

§ 벡터에 차원이 추가된 것
§ 2차원 데이터 구조
§ 수학 행렬과 동일

§ R코드)
§ a = matrix(c(1,2,3,4,5,6,7,8,9),nrow=3)

리스트(list)

§ 서로 다른 R 오브젝트들의 원소로 구성됨
§ lst = list(name='fred',wife='mary'

§ R코드)
§ lst = list(name='fred',wife='mary',childage = c(4,7,9)

데이터프레임(DataFrame)

§ 엑셀과 동일한 형태
§ 테이블 형태의 데이터 객체
§ 컬럼은 서로 다른 속성을 가질 수 있음
§ 변수(열) 길이는 모두 동일

§ R코드)
§ df = data.frame(name = 'mjy', age = 30, gender = factor('M'))

02 실습

<해당표를 데이터 프레임으로 만들어보자>

Number	Name	Graduate
1	'A'	True
2	'B'	False
3	'C'	True
4	'D'	False
5	'E'	True
6	'F'	False
7	'G'	True
8	'H'	False
9	'I'	True
10	'J'	False
Numeric	**내용**	**내용**

- 1) Number, Name, Graduate를 각각의 Vector로 생성
- 2) Number를 5(nrow)X2(ncol)의 행렬로 표현
- 3) 3가지 Vector를 list 형태로 생성
- 4) 3가지 Vector를 DataFrame 형태로 생성

02 정답

<해당표를 데이터 프레임으로 만들어보자>

Number	Name	Graduate
1	'A'	True
2	'B'	False
3	'C'	True
4	'D'	False
5	'E'	True
6	'F'	False
7	'G'	True
8	'H'	False
9	'I'	True
10	'J'	False
Numeric	**Character**	**Logical**

1) Number, Name, Graduate를 각각의 Vector로 생성

```
Number = c(1,2,3,4,5,6,7,8,9,10)
Name = c('A','B' ,'C' ,'D' ,'E' ,'F' ,'G' ,'H' ,'I' ,'J')
Graduate = c(T,F,T,F,T,F,T,F,T,F,T,F)
```

2) Number를 5(nrow)X2(ncol)의 행렬로 표현

```
Matrix(Number,nrow=5,ncol=2)
```

3) 3가지 Vector를 list 형태로 생성

```
lst = list(Number = Number, Name = Name, Graduate = Graduate)
```

4) 3가지 Vector를 DataFrame 형태로 생성

```
df = data.frame(Number = Number, Name = Name, Graduate = Graduate)
```

00 제목을 입력하세요

Chapter

III

R 조건문, 반복문

03 조건문

```
if ( 조건 ) {
    (조건이 True일 때 실행될) 문장 또는 명령어
} else {
    (조건이 False일 때 실행될) 문장 또는 명령어
}
```

```
> grade <- 75
> if (grade >= 70) {
+   print('합격')
+ } else {
+   print('불합격')
+ }
[1] "합격"
```

if

- 장점 : 문장 출력과 다른 명령어 수행을 할 수 있음
- 단점 : 벡터 연산이 불가능하고 한 건에 대해서만 검사가 가능

03 조건문

```
> vec1 <- c(10,20,30)
> if (vec1 == 10) {
+   print('인사부')
+ } else {
+   print('총무부')
+ }
[1] "인사부"                    # 결과값이 인사부인 이유는 첫 번째 요소인 10에 대해서만 조건 치환을 적용하기 때문
Warning message:
In if (vec1 == 10) { :
  the condition has length > 1 and only the first element will be used     # if문은 벡터 연산 불가
```

if문은 반복 연산이 불가하기 때문에

반복 연산이 필요하다면 for문(반복문)이나 if else문을 사용해야합니다.

조건문

- § 장점 : 문장 출력과 다른 명령어 수행을 할 수 있음
- § 단점 : 벡터 연산이 불가능하고 한 건에 대해서만 검사가 가능

03 조건문

```
if ( 조건 1 ) {
    ('조건1'일 때 실행될) 문장 또는 명령어
} else if ( 조건 2 ){
   ('조건1'이 아니고 '조건2'일 때 실행될) 문장 또는 명령어
} else {
    ('조건1'도, '조건2'도 아닐 때 실행될) 문장 또는 명령어
}
```

```
> grade <- 'A'
> if (grade == 'A') {
+   print('합격')
+ } else if (grade == 'B') {
+   print('보류')
+ } else {
+   print('불합격')
+ }
[1] "합격"
```

else if

§ 장점 : 여러 조건에 대해서 검사 가능

§ 단점 : 벡터 연산이 불가능하고 한 건에 대해서만 검사가 가능

03 조건문

3. ifelse 문

조건문에서 ifelse문의 장점은 if문의 한계를 해결하여 벡터 연산(각 요소별 조건 검사)이 가능합니다.

단점은 주어진 값에 따라 yes or no 를 반환해주고, 리턴값만 반환하기 때문에 오직 출력만 가능하고 조건별 명령어 수행은 불가합니다.

ifelse (조건, True일 때 리턴할 값, False일 때 리턴할 값)

```
> vec1 <- c(10,20,30)
> ifelse (vec1 == 10, '인사부', '총무부')      # in oracle : decode(vec1,10, '인사부', '총무부')
[1] "인사부" "총무부" "총무부"


# ifelse 는 중복사용도 가능합니다.
> ifelse (vec1 == 10, '인사부',
+         ifelse (vec1 == 20, '재무부', '총무부'))
[1] "인사부" "재무부" "총무부"
```

ifelse

- 장점 : 벡터 연산이 가능
- 단점 : 다양한 명령어 수행이 불가

03 반복문

```
for(반복변수 in 횟수) {
   반복할 식
}
```

```
> for (i in c(1:5)) {      # 숫자의 개수만큼 벡터의 요소를 i에 넣고 출력
+   print(i)
+ }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
> for (i in c('a','b','c')) {   # 문자의 개수만큼 벡터의 요소를 i에 넣고 출력
+   print(i)
+ }
[1] "a"
[1] "b"
[1] "c"
> for (i in c('a','b','c')) {    # 문자의 개수만큼 반복
+   print(10)
+ }
[1] 10
[1] 10
[1] 10
```

for

- 1) 1~5까지를 보여주는 반복문
- 2) ‘a’,’b’,’c’를 보여주는 반복문
- 3) vector 내의 수만큼 10을 보여주는 반복문

03 반복문

2. while 문

반복문에서 **while문**은 for문과 다르게 **시작 값의 정의와 증가시킬 값**(반복변수 증가 구문)**이 필요**합니다.

```
var <- 시작 값 정의
while(조건) {
   반복할 식
   증가 구문
}
```

```
> i <- 1
> while( i<=5 ) {   # 특정 조건이 맞을 때 까지 반복, 항상 True면 무한 루프에 빠지므로 주의
+   print(i)
+   i <- i + 1
+ }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

while

§ For문과 다르게 시작 값의 정의와 증가 시킬 값의 구문이 필요하다.

03 실습

<활용 데이터>

Number
1
2
3
4
5
6
7
8
9
10
Numeric

§ Q1) a가 10인 객체를 생성하고 5이상이면 '높음'이라는 조건문을 생성하라

§ Q2) Number를 사용하여 5이상은 '높음', 5미만은 '작음' 이라는 단어의 vector를 생성하라 (hint : ifelse() 함수)

§ Q3) Number를 사용하여 1~10까지를 보여주는 for문을 생성하라

§ Q4) 1~5이하의 값만 보여주는 while문을 생성하라

§ 보너스 문제 1) Number을 사용해서 순차적으로 5미만의 숫자는 '낮음', 5이상의 숫자는 '높음'을 for, if문을 활용해 생성하라

§ 보너스 문제 2) 5이하의 숫자 중에 3미만의 숫자는 '낮음', 3이상의 숫자는 '높음'을 while, if문을 활용해 생성하라

03 실습

<활용 데이터>

Number
1
2
3
4
5
6
7
8
9
10
Numeric

§ A1)

§ a = 10

§ if (a>=5) {print('높음')}

§ A2) ifelse(Number>=5,'높음','낮음')

§ A3) for (i in Number) {print(i)}

§ A4)

§ i = 1

§ while(i<=5) {

§ print(i)

§ i = i + 1}

03 실습

<활용 데이터>

Number
1
2
3
4
5
6
7
8
9
10
Numeric

§ 보너스 문제 A1)

§ for (i in Number) { if (i < 5) {print('낮음')} else {print('높음')}}

§ 보너스 문제 A2)

§ i = 1

§ while(i<=5) {

§ if (i >= 3) {print('높음')}

§ else {print('낮음')}

§ i = i + 1

§ }

00 제목을 입력하세요

Chapter

Ⅳ 데이터 가공 및 시각화

04 사용 데이터프레임

컬럼	컬럼명	데이터타입
Salary	임금	Numeric
Age	나이	Numeric
Edu	학력	Numeric
Startsal	시작임금	Numeric
Jobtime	근로월	Numeric
Prevexp	이전경력	Numeric
Minority	소수/다수	Factor
Gender	성별	Factor
jobcat	직업	Factor

Employee data

- 데이터 다운로드
- 코드)
- Install.packages('stima')
- library(stima)
- data(employee)
- str(employee) # 데이터 구조 확인
- summary(employee) # 데이터 분포 확인

04 원하는 row or column 선별

A data.frame: 6 × 9

	salary	age	edu	startsal	jobtime	prevexp	minority	gender	jobcat
	<int>	<int>	<int>	<int>	<int>	<int>	<fct>	<fct>	<fct>
1	57000	36	15	27000	98	144	no_min	m	manager
2	40200	30	16	18750	98	36	no_min	m	Clerical
3	21450	59	12	12000	98	381	no_min	f	Clerical
4	21900	41	8	13200	98	190	no_min	f	Clerical
5	45000	33	15	21000	98	138	no_min	m	Clerical
6	32100	30	15	13500	98	67	no_min	m	Clerical

Data.frame은 행열로 이루어져있다!

§ Data.frame indexing : df[행,열]

§ 행 : employee data의 한 사람

→ 목적 : 1번째부터 10번째 데이터

→ employee[1:10,]

§ 열 : 각각의 컬럼

→ 목적 : Salary 컬럼 추출

→ employee[,1]

→ employee['salary']

→ employee$salary

04 조건 비교

A data.frame: 6 × 9

	salary	age	edu	startsal	jobtime	prevexp	minority	gender	jobcat
	<int>	<int>	<int>	<int>	<int>	<int>	<fct>	<fct>	<fct>
1	57000	36	15	27000	98	144	no_min	m	manager
2	40200	30	16	18750	98	36	no_min	m	Clerical
3	21450	59	12	12000	98	381	no_min	f	Clerical
4	21900	41	8	13200	98	190	no_min	f	Clerical
5	45000	33	15	21000	98	138	no_min	m	Clerical
6	32100	30	15	13500	98	67	no_min	m	Clerical

데이터의 선별은 데이터 비교를 통해 이뤄진다.

- 1. 숫자 비교 함수 : <,>,=
- 2. 문자 비교 함수 : ==, !=
- 3. True는 선별하고 False 제외한다.

- 나이가 50이상 여부
- 코드 : employee$age >= 50

- 직업이 매니저인지 여부
- 코드 : employee$jobcat == 'manager'

04 조건 비교

```
employee$age >= 50
```

```
FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE TRUE FALSE FALSE FALSE FALSE TRUE FALSE TRUE FALSE FALSE FALSE FALSE
FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE FALSE TRUE FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
TRUE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE .... FALSE FALSE FALS
FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE TRUE F
TRUE TRUE FALSE FALSE FALSE FALSE FALSE TRUE TRUE FALSE FALSE FALSE FALSE FA
FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE
FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE
```

코드 실행 : 나이가 50이상이야?
답 : 맞아, 아니야

- 코드 : employee$age >= 50
- 답 : 첫번째 사람은 T, 두번째 사람은 F …
- True는 선택, False는 제외

- 해당 답을 통해 50살 이상인 직원을 추출하기 위해서는?
- 코드 : employee[employee$age >= 50,]

```
head(employee[employee$age >= 50,])
```

A data.frame: 6 × 9

	salary	age	edu	startsal	jobtime	prevexp	minority	gender	jobcat
	<int>	<int>	<int>	<int>	<int>	<int>	<fct>	<fct>	<fct>
3	21450	59	12	12000	98	381	no_min	f	Clerical
24	16950	55	12	9000	97	124	min	f	Clerical
40	19200	55	15	9000	96	23	min	f	Clerical
45	30750	50	12	13500	95	307	no_min	m	Custodial
47	30000	50	12	16500	95	228	no_min	f	Clerical
54	25050	57	12	13500	94	444	no_min	m	Clerical

04 실습

- Q1) employee 중 임금이 30000달러 이상의 사람들의 데이터 추출
- Q2) employee 중 성별이 '여성'인 사람들의 데이터 추출
- Q3) employee 중 임금이 30000달러 이상의 '여성' 데이터 추출 (hint : &(and)) 사용
- Q4) employee 중 임금이 30000달러 이상이거나 '남성'인 데이터 추출 (hint : |(or)) ※ | 기회는 shift + ₩

04 실습

§ Q1) employee 중 임금이 30000달러 이상의 사람들의 데이터 추출

§ A1) employee[employee$salary >= 30000,]

§ Q2) employee 중 성별이 '여성'인 사람들의 데이터 추출

§ A2) employee[employee$gender == 'f',]

§ Q3) employee 중 임금이 30000달러 이상의 '여성' 데이터 추출 (hint : &(and)) 사용

§ A3) employee[employee$salary >= 30000 & employee$gender == 'f',]

§ Q4) employee 중 임금이 30000달러 이상이거나 '남성'인 데이터 추출 (hint : |(or)) ※ | 기호는 shift + ₩

§ A4) employee[employee$salary >= 30000 | employee$gender == 'm',]

04 계산식

1

덧셈

1 + 4

2

뺄셈

4 - 1

3

곱셈

4 * 5

4

나눗셈

몫 : 45 / 7
나머지 : 45 %% 7

5

제곱

2^2

6

루트

sqrt()

04 계산식

1	2	3	4
합계	평균	최대값	최소값
sum()	mean()	max()	min()

5	6	7
중앙값	최빈값	개수
median()	mode()	count()

04 새로운 컬럼 만들기

A data.frame: 6 × 9

	salary	age	edu	startsal	jobtime	prevexp	minority	gender	jobcat
	<int>	<int>	<int>	<int>	<int>	<int>	<fct>	<fct>	<fct>
1	57000	36	15	27000	98	144	no_min	m	manager
2	40200	30	16	18750	98	36	no_min	m	Clerical
3	21450	59	12	12000	98	381	no_min	f	Clerical
4	21900	41	8	13200	98	190	no_min	f	Clerical
5	45000	33	15	21000	98	138	no_min	m	Clerical
6	32100	30	15	13500	98	67	no_min	m	Clerical

새로운 컬럼 만들기

- data.frame$새로운변수
- data.frame['새로운변수']

- Jobtime이 월로 되어있어서 보기가 어렵다
- Jobtime을 년도로 변경하자
- employee$jobtime/12

- 다른 컬럼을 생성하자
- employee$jobyear =
  employee$jobtime/12
- employee['jobyear'] =
  employee$jobtime/12

04 실습

- Q1) 시작임금에 비해 현재 임금이 얼마나 올랐는지 직원별로 계산해서 sal_inc_ratio라는 컬럼에 넣자
- Q2) sal_inc_ratio가 3배 이상 오른 직원들을 뽑자
- Q3) sal_inc_ratio가 3배 이상 오른 직원들의 평균 근로월을 구하자
- Q4) 평균 근로월이 월 기준으로 되어 있어서 보기가 힘들다. 평균 근로년수로 바꾸자

04 실습

§ Q1) 시작임금에 비해 현재 임금이 얼마나 올랐는지 직원별로 계산해서 sal_inc_ratio라는 컬럼에 넣자

§ A1) employee$sal_inc_ratio = employee$salary / employee$startsal

§ Q2) sal_inc_ratio가 3배 이상 오른 직원들을 뽑자

§ A2) employee[employee$sal_inc_ratio >= 3,]

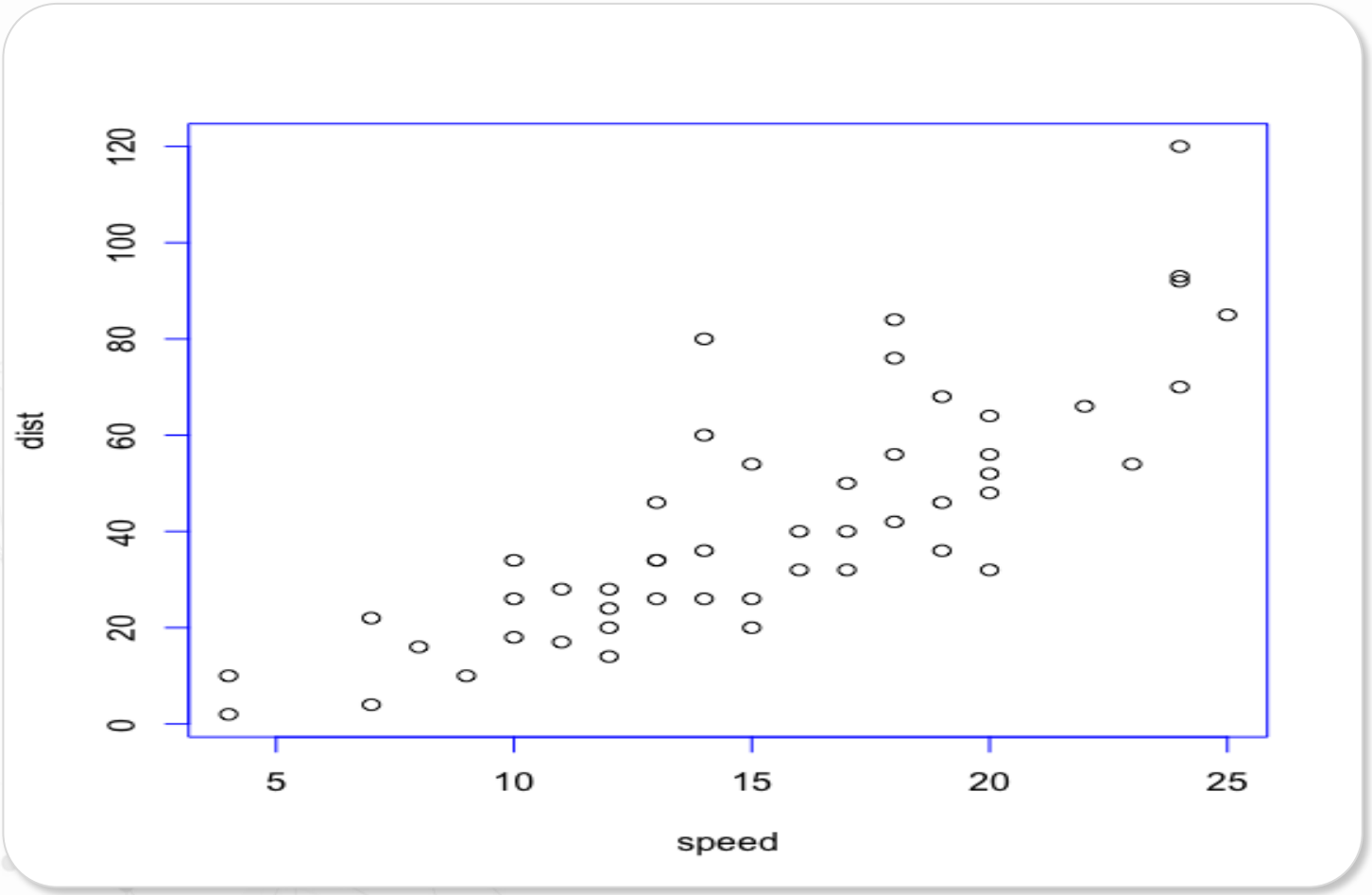
§ Q3) sal_inc_ratio가 3배 이상 오른 직원들의 평균 근로월을 구하자

§ A3) mean(employee$jobtime[employee$sal_inc_ratio >= 3])

§ Q4) 평균 근로월이 월 기준으로 되어 있어서 보기가 힘들다. 평균 근로년수로 바꾸자

§ mean(employee$jobtime[employee$sal_inc_ratio >= 3]) / 12

04 시각화



plot

§ 코드) plot(x,y)

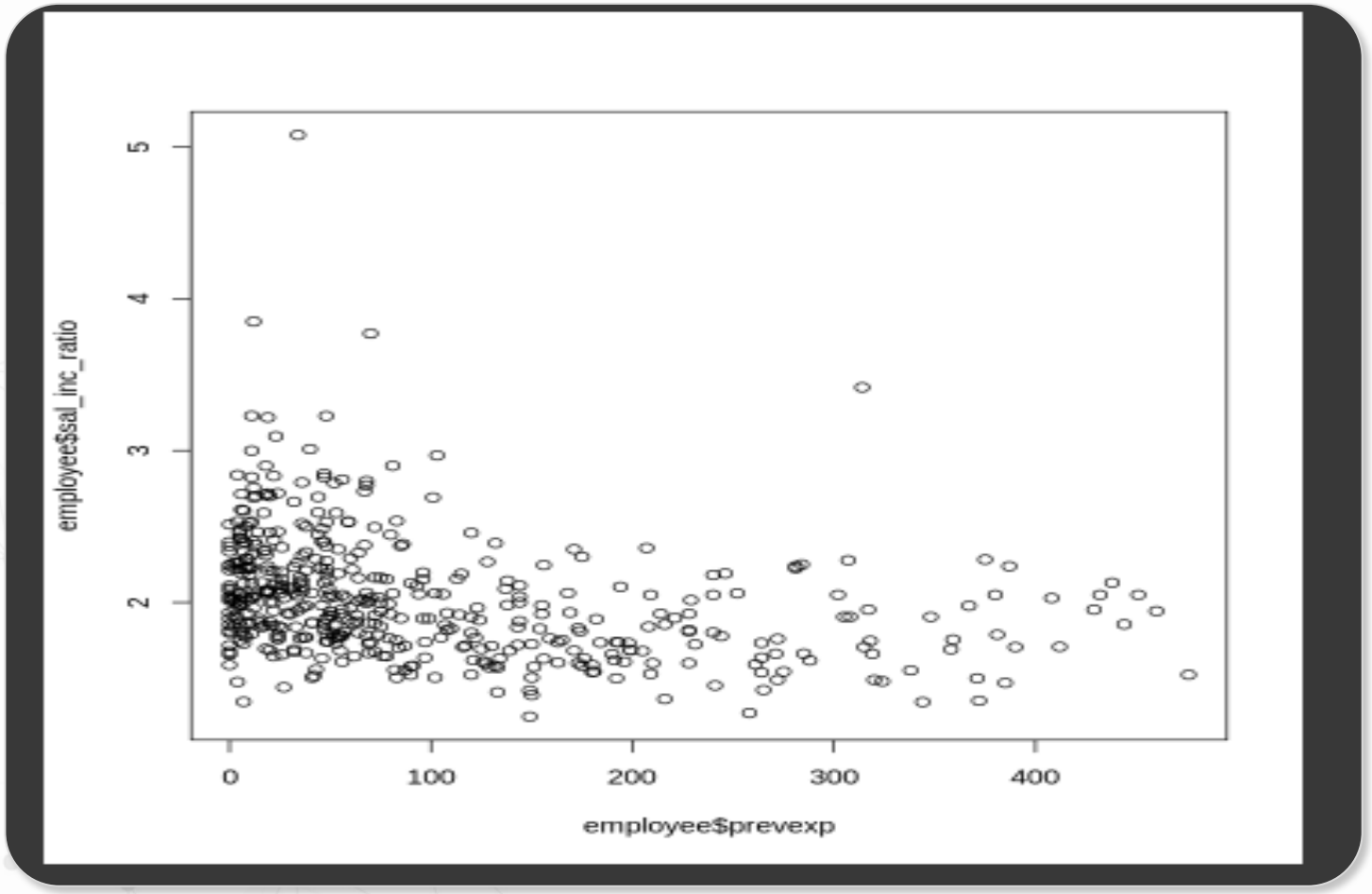
§ 예시) x : 자동차 속도, y : 거리

§ 해당 그림으로 알 수 있는 것?

§ 자동차 속도가 빠르면 빠를수록 거리는 멀어진다

§ Plot을 통해 두 변수의 관계를 알 수 있다!

04 임금상승률과 이전경력과의 plot



plot

§ 코드)

```
plot(employee$prevexp,employee$
sal_inc_ratio)
```

§ X축 : 이전경력, Y축 : 임금상승률

§ 해당 그림으로 알 수 있는 것?

§ 이전경력이 높고 낮음과 관계없이 대부분 동일한 임금상승률을 가지고 있다.

04 최종실습

- ※ 각 단계 단계 별로 새로운 data.frame을 사용해서 진행
- Q1) 1번부터 400번까지의 직원들만 선별
- Q2) 해당 직원들 중 이전 경력이 300이하의 직원들만 추출
- Q3) Q2의 직원들 중 'MANAGER' 직무는 제외
- Q4) Q3의 직원들 중 가장 높은 연봉을 계산
- Q5) Q4의 결과로 새로운 컬럼인 max_sal을 생성
- Q6) 직원들의 가장 높은 연봉 대비 몇 퍼센트의 비율을 받는지 max_sal_ratio 컬럼을 생성
- Q7) max_sal_ratio와 근무월 컬럼을 사용하여 plot을 생성
- Q8) 해당 plot을 보고 max_sal_ratio와 근무월 간의 관계를 해석

04 최종실습

- Q1) 1번부터 400번까지의 직원들만 선별
- A1) employee1 = employee[1:400,]

- Q2) 해당 직원들 중 이전 경력이 300이하의 직원들만 추출
- A2) employee2 = employee1[employee1$prevexp <= 300,]

- Q3) Q2의 직원들 중 'MANAGER' 직무는 제외
- A3) employee3 = employee2[employee2$jobcat != 'manager',]

- Q4) Q3의 직원들 중 가장 높은 연봉을 계산
- A4) max(employee3$salary)

04 최종실습

§ Q5) Q4의 결과로 새로운 컬럼인 max_sal을 생성

§ A5) employee3$max_sal = max(employee3$salary)

§ Q6) 직원들의 가장 높은 연봉 대비 몇 퍼센트의 비율을 받는지 max_sal_ratio 컬럼을 생성

§ Q6) employee3$max_sal_ratio = employee3$salary / employee3$max_sal

§ Q7) max_sal_ratio와 근무월 컬럼을 사용하여 plot을 생성

§ Q7) plot(employee3$max_sal_ratio, employee3$jobtime)

§ Q8) 해당 plot을 보고 max_sal_ratio와 근무월 간의 관계를 해석

§ A8) 크게 의미는 없지만 신기한 것이 근무월이 80개월인 직원이 가장 높은 임금을 받음

R을 마치며

궁금한게 있으시면

Name : 민종열 대리

Email : wpdntm3001@naver.com

Hp : 010-5439-5931

감사합니다

한국고용정보원
Korea Employment Information Service