데이터베이스의 원리 및 응용
Homework #3

산업공학과
2014002542
황보성훈

목차

1. 파일 설명

(1) csv 파일

6 개의 csv 파일.

cars.csv: 자동차에 대한 정보

drivers.csv: 렌터카를 이용한 사람에 대한 정보

rent_comp.csv: 렌터카 업체에 관한 정보

rent.csv: 렌트에 관한 정보

repair_shop.csv: 정비소 업체에 관한 정보

repair.csv: 수리에 관한 정보

(2) sql 파일

hbsh_2014002542.sql: MySQL 에서 사용할 테이블 정의

(3) R 파일

rentacar.R: 필요한 함수 정의

rentacar_test.R: rentacar.R 에서 정의한 함수들이 제대로 동작하는지 확인하기 위해 테스트한 파일 **(혹시 함수의 실행 방법과 같은 구체적인 동작이 궁금하실 때 보시면 좋을 거 같습니다)**

2. 함수 설명

[1] 데이터 초기화 기능

```
# (1) 데이터 초기화 기능 ##################################################################################
# init_sql(mydata_name, mydata): sql에 mydata를 insert하는 함수
init_sql <- function (mydata_name, mydata){
  name <- names(mydata) # mydata의 열들의 이름을 저장
  name_length <- length(name) # 열 수 저장
  # "INSERT INTO rent (drid, cid, rentdate) VALUES(22, 106, '2018-02-11');"와 같은 query를 만드는 과정
  query1 <- paste('insert into', mydata_name, '(') # query1에서는 "INSERT INTO rent ("까지의 문자열

  for (i in 1:(name_length - 1)){
    query1 <- paste0(query1, name[i], sep = ',')
  }
  query1 <- paste0(query1, name[name_length], ') values(')

  for (j in 1:nrow(mydata)){
    query2 <- query1
    for (k in 1:(name_length - 1)){
      # 변수형이 숫자일 때는 그대로 입력하고 문자형일 때는 따옴표로 감싸야한다.
      if (is.numeric(mydata[,k])){
        query2 <- paste0(query2, mydata[j,k], sep = ',')
      }
      else{
        query2 <- paste0(query2, '\'',mydata[j,k],'\',')
      }
    }
    if (is.numeric(mydata[,name_length])){
      query2 <- paste0(query2, mydata[j,name_length], ');')
    }
    else{
      query2 <- paste0(query2, '\'',mydata[j,name_length],'\');')
    }

    query2 <- gsub('\'NA\'', 'NA' ,query2)
    query2 <- gsub('NA', 'null' ,query2)
    # print(query2)
    rsInsert <- dbSendQuery(con, query2) # query를 sql에 보낸다.
  }
}
##########################################################################################################
```

(1) 함수 이름 및 호출 형태: init_sql(mydata_name, mydata)

(2) 설명: mydata_name 에는 MySQL 에 정의한 릴레이션의 이름을 입력하고, mydata 에는 mydata_name 에 대응하는 (csv 파일을 읽어들인) 데이터프레임을 입력한다. mydata 의 값들을 query 를 통해서 MySQL 내부에 저장하도록 한다.

(3) SQL 문: mydata 의 각각의 값과 mydata_name 을 이용해 "INSERT INTO rent (drid, cid, rentdate) VALUES(22, 106, '2018-02-11');"와 같은 query 를 MySQL 에 보내면 데이터가 MySQL 내부에 저장된다.

[2] 데이터 출력 기능

```
# (2) 데이터 출력 기능 ##########################################################################
# print_sql(mydata_name, col_name): mydata_name에 해당하는 릴레이션을 col_name의 오름차순으로 출력
# [rent_comp], [drivers], [repair_shop]: 이름(rent_comp_name, dr_name, shop_name)의 오름차순
# [rent], [repair]: 날짜(rent_start_date, rep_date)의 오름차순
# [cars]: 등록 ID(car_id)의 오름차순
print_sql <- function(mydata_name, col_name){
  query <- paste('select * from', mydata_name, 'order by', col_name, ';',sep = ' ')
  result <-  data.frame(dbGetQuery(con, query))
  #print(query)
  print(result)
  #dbClearResult(dbListResults(con)[[1]])
}
################################################################################################
```

(1) 함수 이름 및 호출 형태: print_sql(mydata_name, col_name)

(2) 설명: mydata_name 에 해당하는 릴레이션을 col_name 에 해당하는 열의 오름차순 정렬을 통해 출력하는 함수

(3) SQL 문: select * from mydata_name order by col_name;

[3] 특정 렌터카 정보를 삭제하는 기능

```
# (3) 특정 렌터카 정보를 삭제하는 기능 ##########################################################
# delete_car(렌터카의 등록 ID, 렌터카 업체 이름)
delete_car <- function(car_id, rent_comp_name){
  query1 <- paste('select car_comp_id from cars c,
                  rent_comp rc where c.car_comp_id = rc.rent_comp_id and rc.rent_comp_name=\'',
                  rent_comp_name,'\';', sep = '')
  result <- unique(data.frame(dbGetQuery(con, query1)))
  comp_id <- result[1,1]
  #print(comp_id)
  #print(query1)

  query2 <- paste('delete from cars where car_id=', car_id, ' and car_comp_id=', comp_id, ';', sep='')
  # print(query2)
  dbSendQuery(con, query2)
}
################################################################################################
```

(1) 함수 이름 및 호출 형태: delete_car(**car_id**, **rent_comp_name**)

(2) 설명: 렌터카 업체 이름(rent_comp_name)을 통해서 렌터카 업체의 id(comp_id)를 찾고, 그 결과를 통해 cars 에서 해당 렌터카 정보를 삭제한다.

(3) SQL 문:

select car_comp_id from cars c, rent_comp rc where c.car_comp_id = rc.rent_comp_id and rc.rent_comp_name= **rent_comp_name**;

delete from cars where car_id= **car_id** and car_comp_id= **comp_id**;

[4] 특정 고객의 정보를 수정하는 기능

```
# (4) 특정 고객의 정보를 수정하는 기능 ##################################################################################
# modify_driver(운전면허번호, col_name, value): 운전면허번호에 해당하는 row의 col_name의 값을 value로 수정.
# drivers의 값(value)들은 모두 숫자형이 아니므로 수정할 값을 작은 따옴표로 감싸준다.
modify_driver <- function(driver_license, col_name, value) {
  query <- paste('update drivers set ', col_name, '=\'', value, '\' where driver_license=\'', driver_license, '\';', sep = '')
  #print(query)
  dbSendQuery(con, query)
}
#########################################################################################################################
```

(1) 함수 이름 및 호출 형태: modify_driver(**driver_license**, **col_name**, **value**)

(2) 설명: driver_license 에 해당하는 row 의 col_name 의 값을 value 로 수정

(3) SQL 문:

update drivers set **col_name** = '**value**' where driver_license= '**driver_license**';

[5] 특정 정비업소의 정보를 추가하는 기능

```
# (5) 특정 정비업소의 정보를 추가하는 기능  ##############################################################################
# add_repair_shop(정비소ID, 정비소이름, 정비소주소, 정비소전화번호, 담당자이름, 담당자이메일)
add_repair_shop <- function(shop_id, shop_name, shop_addr, shop_phone, shop_admin_name, shop_admin_email){
  query <- paste('insert into repair_shop values(', shop_id, ',\'', shop_name, '\',\'', shop_addr, '\',\'',
                 shop_phone, '\',\'', shop_admin_name, '\',\'', shop_admin_email,'\');', sep = '')
  #print(query)
  dbSendQuery(con, query)
}
#########################################################################################################################
```

(1) 함수 이름 및 호출 형태: add_repair_shop(shop_id, shop_name, shop_addr, shop_phone, shop_admin_name, shop_admin_email)

(2) 설명: repair_shop 에 인자들의 정보를 추가

(3) SQL 문:

insert into repair_shop values(shop_id, 'shop_name', 'shop_name', 'shop_addr', 'shop_phone', 'shop_admin_name', 'shop_admin_email');

[6] 대여 기간 연장 가능

```
# (6) 대여 기간 연장 가능 ###############################################################################
# lengthen_due(name): dr_name이 name인 고객 중 대여기간을 연장할 수 있는 rent_id에 대해서 대여기간을 5일 연장. 기타청구내역 'extended', 기타청구요금 50000.
lengthen_due <- function(name){
  # query: rent_days를 +5, r2.extra_bill을 'extended'로 바꾸고,
  # extra_pay는 null일 경우 50000으로, null이 아닐 경우 +50000 해준다.
  query1 <- 'select rent_id from rent r, drivers d where r.driver_license = d.driver_license and dr_name ='
  query1 <- paste(query1, '\'', name, '\' and date_add(r.rent_start_date, interval rent_days day) >= now();', sep='')
  able_rent_id <- dbGetQuery(con, query1) # able_rent_id는 대여 기간을 연장할 수 있는 rent_id. 즉, 반납일이 현재 날짜 이후인 rent_id.

  if(nrow(able_rent_id) == 0){
    print('There are no rows that meet the condition.')
  }else{
    # able_rent_id의 길이만큼 반복하면서 query를 날림.
    for(i in 1:nrow(able_rent_id)){
      query2 <- 'update rent set rent_days = rent_days + 5, extra_bill = \'extended\', extra_pay = if(extra_pay is null, 50000, extra_pay + 50000) where rent_id = '
      query2 <- paste(query2, able_rent_id[i,1], sep = '')
      dbSendQuery(con, query2)
    }
  }
}
#########################################################################################################################
```

(1) 함수 이름 및 호출 형태: lengthen_due(**name**)

(2) 설명: dr_name 이 name 인 고객 중 대여기간을 연장할 수 있는

rent_id(반납일이 현재 날짜 이후인 rent_id)에 대해서 대여기간을 5 일 연장. 기타청구내역 'extended', 기타청구요금 50000.

(3) SQL 문:

select rent_id from rent r, drivers d where r.driver_license = d.driver_license and dr_name = '**name**' and date_add(r.rent_start_date, interval rent_days day) >= **now()**; # now()를 이용해서 현재 날짜와 비교해서 가능한 rent_id(able_rent_id)를 찾는다.

update rent set rent_days = rent_days + 5, extra_bill = 'extended', extra_pay = if(extra_pay is null, 50000, extra_pay + 50000) where rent_id = **able_rent_id**; # rent_days 를 +5, extra_bill 을 'extended'로 바꾸고, extra_pay 는 null 일 경우 50000 으로, null 이 아닐 경우 +50000 해준다.

[7] 렌터카 대여 회사 삭제 기능

```
# (7) 렌터카 대여 회사 삭제 기능 ##############################################################
# delete_rent_comp(comp_name): rent_comp 에서 rent_comp_name 이 comp_name 인 것의 정보 삭제
delete_rent_comp <- function(comp_name) {
  query <- 'delete from rent_comp where rent_comp_name = \''
  query <- paste(query, comp_name, '\';', sep='')
  dbSendQuery(con, query)
}
##############################################################################################
```

(1) 함수 이름 및 호출 형태: delete_rent_comp(**comp_name**)

(2) 설명: rent_comp 에서 rent_comp_name 이 comp_name 인 것의 정보 삭제

(3) SQL 문:

delete from rent_comp where rent_comp_name = **'comp_name'**;

[8] 특정 기간의 렌터카 내역 출력 가능

```
# (8) 특정 기간의 렌터카 내역 출력 가능 ########################################################
# print_drivers_november(): 2018년 11월 렌터카를 대여한 모든 고객의 이름과 주소,
# 전화번호 출력(여러 번 렌트 했더라도 한 번만 출력)
print_drivers_november <- function(){
  query <- 'select d.dr_name, d.dr_addr, d.dr_phone from rent r, drivers d
  where r.driver_license=d.driver_license and rent_start_date > \'2018-10-31\'
  and rent_start_date < \'2018-12-01\';'
  unique(dbGetQuery(con, query))
}
##############################################################################################
```

(1) 함수 이름 및 호출 형태: print_drivers_november()

(2) 설명: 2018 년 11 월 렌터카를 대여한 모든 고객의 이름과 주소, 전화번호 출력**(여러 번 렌트 했더라도 한 번만 출력)**

(3) SQL 문:

select d.dr_name, d.dr_addr, d.dr_phone from rent r, drivers d where r.driver_license=d.driver_license and rent_start_date > '2018-10-31' and rent_start_date < '2018-12-01';

[9] 주소에 따른 정비소 출력 기능

```
# (9) 주소에 따른 정비소 출력 기능 ##################################################################
# print_repair_shop(도시명): 해당 도시에 위치한 렌터카 정비소의 모든 정보 출력
print_repair_shop <- function(city){
  query <- paste('select * from repair_shop where shop_addr like \'%', city, '%\'', sep = '')
  #print(query)
  dbGetQuery(con, query)
}
#####################################################################################################
```

(1) 함수 이름 및 호출 형태: print_repair_shop(**city**)

(2) 설명: 해당 도시에 위치한 렌터카 정비소의 모든 정보 출력

(3) SQL 문:

select * from repair_shop where shop_addr like '%**city**%';

[10] 특정 렌터카 출력 기능

```
# (10) 특정 렌터카 출력 기능 ########################################################
# print_cars_2010_5people(): 렌터카 승차 인원이 5명 이상이고 렌터카 등록일자가
# 2010년식인 렌터카의 렌터카 번호, 이름, 대여 가격 출력
print_cars_2010_5people <- function(){
  query <- 'select car_id, car_name, car_rent_pay from cars where car_cap >= 5
  and car_reg_date >= \'2010-01-01\' and car_reg_date <= \'2010-12-31\';'
  #print(query)
  dbGetQuery(con, query)
}
#####################################################################################
```

(1) 함수 이름 및 호출 형태: print_cars_2010_5people()

(2) 설명: 렌터카 승차 인원이 5 명 이상이고 렌터카 등록일자가 2010 년식인 렌터카의 렌터카 번호, 이름, 대여 가격 출력

(3) SQL 문:

select car_id, car_name, car_rent_pay from cars where car_cap >= 5 and car_reg_date >= '2010-01-01' and car_reg_date <= '2010-12-31';

[11] 렌터카 대여 내역 통계 기능

```r
# (11) 렌터카 대여 내역 통계 기능 ##############################################################################
# november_top3(): 2018년 11월에 렌트를 가장 많이 한 고객 top3의 대여 횟수, 바차트(고객 이름 별 대여 횟수)
library(ggplot2)

november_top3 <- function(){
  query <- 'select d.dr_name as name, count(*) as count
from rent r, drivers d
where r.driver_license = d.driver_license
and r.rent_start_date > \'2018-10-31\'
and r.rent_start_date < \'2018-12-01\'
group by d.dr_name
order by count(*) desc;'

  result <- as.data.frame(dbGetQuery(con, query))[1:3,1:2]
  ggplot(data = result, aes(x = name, y = count, fill = name)) + geom_bar(stat = 'identity')
}
################################################################################################################
```

(1) 함수 이름 및 호출 형태: november_top3()

(2) 설명: 2018 년 11 월에 렌트를 가장 많이 한 고객 top3 의 대여 횟수, 바차트(고객 이름 별 대여 횟수)

(3) SQL 문:

```
select d.dr_name as name, count(*) as count
from rent r, drivers d
where r.driver_license = d.driver_license
and r.rent_start_date > ₩'2018-10-31₩'
and r.rent_start_date < ₩'2018-12-01₩'
group by d.dr_name
order by count(*) desc;
```

위의 SQL 문 결과 중 상위 3 개만 뽑아서 result 에 저장한 후, ggplot 을 이용해서 시각화

3. 실행 결과

[2] 데이터 출력 기능

```
> print_sql('rent_comp', 'rent_comp_name')
  rent_comp_id rent_comp_name                                rent_comp_addr rent_comp_phone rent_comp_admin  rent_comp_admin_email
1            2            AJ          117-7 Nonhyeon-dong Gangnam-gu Seoul     02-556-4258          Lee YJ         lyj34@gmail.com
2            6         GoGo       158-10 Haengdang-dong Seongdong-gu Seoul     02-2241-8278         Jang BJ         jbj93@daum.net
3            4        Kumho          107-1 Nonhyeon-dong Gangnam-gu Seoul     02-3443-8000          Son HM     yangbon92@gmail.com
4            3        Lotte 422 Teheran-ro Daechi 4-dong Gangnam-gu Seoul       1577-5100         Park JS        manu09@gmail.com
5            1        Oreum        Jeju Island Jeju-si Orasam-dong 2104-1    064-712-8570          Kim SS           osf@naver.com
6            5           SK              1 Junggok-dong Gwangjin-gu Seoul     02-498-0852          Lee DH lottejjang11@naver.com
```

[3] 특정 렌터카 정보를 삭제하는 기능

```
> dbGetQuery(con, 'select * from cars;') # 삭제하기 전
   car_id car_comp_id  car_name car_number car_cap                                                                                                car_detail car_rent_pay car_reg_date
1       1          6    Avante    12a3231       5                                                                    Hyundai Motor's front-wheel drive semi-middle sedan       83000   2011-11-11
2       2          2   SantaFe   152d9822       7 A mid-size SUV with a monocoque body type produced by Hyundai Motor's front-wheelers modern car       149000   2015-12-23
3       3          1       G70    54b0639       5                                                                                The midsize sports sedan of Genesis       232000   2018-04-24
4       4          3  Grandeur    19a7777       5                                                                  Hyundai Motor's high-end front-wheel drive sedan       189000   2009-03-11
5       5          6       Ray    67e8016       6                                                                          Kia Motors' Box-typed Light Vehicle       109000   2014-11-30
6       6          4     Spark    52c3108       4                                                                                Chevrolet of light vehicles        63000   2017-07-07
7       7          1     Spark   152a3018       4                                                                                Chevrolet of light vehicles        62000   2018-03-22
8       8          2      Mini    39b2764       5                                                                           A front-wheel drive of the bmw       130000   2010-04-01
9       9          1       Qm3    52e3108       6                                                                           Renault Samsung Motors Small SUV       112000   2016-12-21
10     10          2       Sm3    91d1121       5                                                                      Renault Samsung Motor Co.'s semi-middle sedan        91000   2010-05-11
11     11          2    Sonata   132a1212       5                                                                           Hyundai Motor's mid-size front-wheel drive        99000   2010-10-01
12     12          3 CSL_class    23g9882       5                                                        Mercedes Benz's semi-large 4-door coupe sedan/station wagon       229000   2015-12-03
> delete_car(1, 'GoGo') # GoGo(rent_comp_id = 6) 소속의 차 중, car_id = 1인 차량 삭제
<MySQLResult:2,15,144>
> dbGetQuery(con, 'select * from cars;') # [cars]에서 car_id = 1인 차량이 삭제된 것을 확인할 수 있음
   car_id car_comp_id  car_name car_number car_cap                                                                                                car_detail car_rent_pay car_reg_date
1       2          2   SantaFe   152d9822       7 A mid-size SUV with a monocoque body type produced by Hyundai Motor's front-wheelers modern car       149000   2015-12-23
2       3          1       G70    54b0639       5                                                                                The midsize sports sedan of Genesis       232000   2018-04-24
3       4          3  Grandeur    19a7777       5                                                                  Hyundai Motor's high-end front-wheel drive sedan       189000   2009-03-11
4       5          6       Ray    67e8016       6                                                                          Kia Motors' Box-typed Light Vehicle       109000   2014-11-30
5       6          4     Spark    52c3108       4                                                                                Chevrolet of light vehicles        63000   2017-07-07
6       7          1     Spark   152a3018       4                                                                                Chevrolet of light vehicles        62000   2018-03-22
7       8          2      Mini    39b2764       5                                                                           A front-wheel drive of the bmw       130000   2010-04-01
8       9          1       Qm3    52e3108       6                                                                           Renault Samsung Motors Small SUV       112000   2016-12-21
9      10          2       Sm3    91d1121       5                                                                      Renault Samsung Motor Co.'s semi-middle sedan        91000   2010-05-11
10     11          2    Sonata   132a1212       5                                                                           Hyundai Motor's mid-size front-wheel drive        99000   2010-10-01
11     12          3 CSL_class    23g9882       5                                                        Mercedes Benz's semi-large 4-door coupe sedan/station wagon       229000   2015-12-03
```

[4] 특정 고객의 정보를 수정하는 기능

```
> dbGetQuery(con, 'select * from drivers where driver_license = \'12-10-239956-23\';') # 바꾸기 전
   driver_license dr_name                                dr_addr      dr_phone        dr_email dr_last_use_date dr_last_car
1 12-10-239956-23 Hong SC Gyonggi-do 3rd-dong Jugong 13th-dong 603 010-2435-2311 hsc01@gmail.com       2018-11-09         Qm3
> modify_driver('12-10-239956-23', 'dr_email', 'hong!!!@gmail.com')
<MySQLResult:207218448,15,147>
> dbGetQuery(con, 'select * from drivers where driver_license = \'12-10-239956-23\';') # 바뀐 후
   driver_license dr_name                                dr_addr      dr_phone         dr_email dr_last_use_date dr_last_car
1 12-10-239956-23 Hong SC Gyonggi-do 3rd-dong Jugong 13th-dong 603 010-2435-2311 hong!!!@gmail.com       2018-11-09         Qm3
```

[5] 특정 정비 업소의 정보를 추가하는 기능

```
> dbGetQuery(con, 'select * from repair_shop') # 수정하기 전
  shop_id   shop_name                                shop_addr   shop_phone shop_admin_name     shop_admin_email
1       1     Sungwoo       Seongsu-dong 1-ga Seongdong-gu Seoul  02-462-3554         Jang SW       jsm19@daum.net
2       2    KIA auto  Seocho-gil 42 Seocho-dong Seocho-gu Seoul  02-522-0427         Kang BR borissal12@gmail.com
3       3 Renault Sky            Hyoja-dong 65-2 Jongno-gu Seoul  02-739-6677           Do BS   tobong55@naver.com
4       4  Speed mate Seoksu-dong Manan-gu Anyang-si Gyeonggi-do 031-472-6051          Lee JG      ljg23@naver.com
5       5      Sarang       Haan-dong Gwangmyeong-si Gyeonggi-do  02-898-5959         Choi CM choichoi09@gmail.com
> add_repair_shop(8,'Koala', 'Song-am-ri Yangyang-eup Yangyang-gun Gangwon-do 223-1', '033-671-8275', 'Lee SK','kiki@naver.com')
<MySQLResult:164563464,15,150>
> dbGetQuery(con, 'select * from repair_shop') # 'Koala'라는 정비업소의 정보가 추가된 것을 확인할 수 있다
  shop_id   shop_name                                shop_addr   shop_phone shop_admin_name     shop_admin_email
1       1     Sungwoo       Seongsu-dong 1-ga Seongdong-gu Seoul  02-462-3554         Jang SW       jsm19@daum.net
2       2    KIA auto  Seocho-gil 42 Seocho-dong Seocho-gu Seoul  02-522-0427         Kang BR borissal12@gmail.com
3       3 Renault Sky            Hyoja-dong 65-2 Jongno-gu Seoul  02-739-6677           Do BS   tobong55@naver.com
4       4  Speed mate Seoksu-dong Manan-gu Anyang-si Gyeonggi-do 031-472-6051          Lee JG      ljg23@naver.com
5       5      Sarang       Haan-dong Gwangmyeong-si Gyeonggi-do  02-898-5959         Choi CM choichoi09@gmail.com
6       8       Koala Song-am-ri Yangyang-eup Yangyang-gun Gangwon-do 223-1 033-671-8275          Lee SK       kiki@naver.com
```

[6] 대여기간 연장 기능

```
> dbGetQuery(con, lengthen_due_result_query) # 수정하기 전
  rent_id car_id  driver_license rent_comp_id rent_start_date rent_days   pay_date rent_pay extra_bill extra_pay
1       1      2 11-18-123456-70            2      2018-10-11         1 2018-10-25   149000       <NA>        NA
2       5     NA 11-18-123456-70           NA      2018-10-19         3 2018-11-02   249000       <NA>        NA
3       9      2 11-18-123456-70            2      2018-10-24         1 2018-11-07   149000       <NA>        NA
4      20      7 11-18-123456-70            1      2018-11-05         1 2018-11-19    62000       <NA>        NA
5      34      3 11-18-123456-70            1      2018-11-22         1 2018-12-06   232000       <NA>        NA
6      36      6 11-18-123456-70            4      2018-11-23         3 2018-12-07   189000       <NA>        NA
7      44      9 11-18-123456-70            1      2018-11-30         1 2018-12-14   112000       <NA>        NA
8      46      5 11-18-123456-70            6      2018-12-28         1 2018-12-31   109000       <NA>        NA
> lengthen_due('Kang Juyoung') # 'Kang Juyoung' 고객의 대여기간 5일 연장.
> dbGetQuery(con, lengthen_due_result_query) # 수정 후 대여기간이 연장된 것을 확인할 수 있다.
  rent_id car_id  driver_license rent_comp_id rent_start_date rent_days   pay_date rent_pay extra_bill extra_pay
1       1      2 11-18-123456-70            2      2018-10-11         1 2018-10-25   149000       <NA>        NA
2       5     NA 11-18-123456-70           NA      2018-10-19         3 2018-11-02   249000       <NA>        NA
3       9      2 11-18-123456-70            2      2018-10-24         1 2018-11-07   149000       <NA>        NA
4      20      7 11-18-123456-70            1      2018-11-05         1 2018-11-19    62000       <NA>        NA
5      34      3 11-18-123456-70            1      2018-11-22         1 2018-12-06   232000       <NA>        NA
6      36      6 11-18-123456-70            4      2018-11-23         3 2018-12-07   189000       <NA>        NA
7      44      9 11-18-123456-70            1      2018-11-30         1 2018-12-14   112000       <NA>        NA
8      46      5 11-18-123456-70            6      2018-12-28         6 2018-12-31   109000   extended     50000
```

[7] 렌터카 대여 회사 삭제 기능

```
> dbGetQuery(con, 'select * from rent_comp;') # 'Oreum' 삭제 전
  rent_comp_id rent_comp_name                                rent_comp_addr rent_comp_phone rent_comp_admin  rent_comp_admin_email
1            1          Oreum        Jeju Island Jeju-si Orasam-dong 2104-1    064-712-8570          Kim SS          osf@naver.com
2            2             AJ          117-7 Nonhyeon-dong Gangnam-gu Seoul     02-556-4258          Lee YJ        lyj34@gmail.com
3            3          Lotte 422 Teheran-ro Daechi 4-dong Gangnam-gu Seoul       1577-5100         Park JS       manu09@gmail.com
4            4          Kumho          107-1 Nonhyeon-dong Gangnam-gu Seoul    02-3443-8000          Son HM    yangbon92@gmail.com
5            5             SK             1 Junggok-dong Gwangjin-gu Seoul     02-498-0852          Lee DH lottejjang11@naver.com
6            6           GoGo      158-10 Haengdang-dong Seongdong-gu Seoul    02-2241-8278         Jang BJ         jbj93@daum.net
> delete_rent_comp('Oreum')
<MySQLResult:1,15,157>
> dbGetQuery(con, 'select * from rent_comp;') # 'Oreum' 삭제 후 [rent_comp]에서 'Oreum' 정보 삭제
  rent_comp_id rent_comp_name                                rent_comp_addr rent_comp_phone rent_comp_admin  rent_comp_admin_email
1            2             AJ          117-7 Nonhyeon-dong Gangnam-gu Seoul     02-556-4258          Lee YJ        lyj34@gmail.com
2            3          Lotte 422 Teheran-ro Daechi 4-dong Gangnam-gu Seoul       1577-5100         Park JS       manu09@gmail.com
3            4          Kumho          107-1 Nonhyeon-dong Gangnam-gu Seoul    02-3443-8000          Son HM    yangbon92@gmail.com
4            5             SK             1 Junggok-dong Gwangjin-gu Seoul     02-498-0852          Lee DH lottejjang11@naver.com
5            6           GoGo      158-10 Haengdang-dong Seongdong-gu Seoul    02-2241-8278         Jang BJ         jbj93@daum.net
```

[8] 특정 기간의 렌터카 내역 출력 기능

```
> # print_drivers_november(): 2018년 11월 렌터카를 대여한 모든 고객의 이름과 주소, 전화번호 출력(여러 번 렌트 했더라도 한 번만 출력)
> print_drivers_november()
        dr_name                                                    dr_addr      dr_phone
1       Jeon KB                      Gwangmyeong-si 1-dong Jugong 5th Gyeonggi-do  02-3223-1232
2       Hong SC                          Gyonggi-do 3rd-dong Jugong 13th-dong 603 010-2435-2311
3        Kim HJ  202 109-dong Kumho Apartment Bundang-gu Seongnam-si Gyeonggi-do 010-1288-9384
4        Kim DH 1503 109-dong Hanshin Apartment Tangsan-ro Yeongdeungpo-gu Seoul 010-2445-3584
5        Kim JS              2101 110-dong Hillstate Apartment in Jongno-gu Seoul 010-0872-3218
6        Lim JH   801 103-dong Jangsu Apartments Onsu-dong Siheun-si Gyeonggi-do  02-9540-2495
7  Kang Juyoung         11-1 Hanyang University in Sageun-dong Seongdong-gu Seoul  02-2220-4458
13       Lee JS          3-dong Cheolsan Gwangmyeong-si Gyeonggi-do 108-dong 2102 010-2323-4345
```

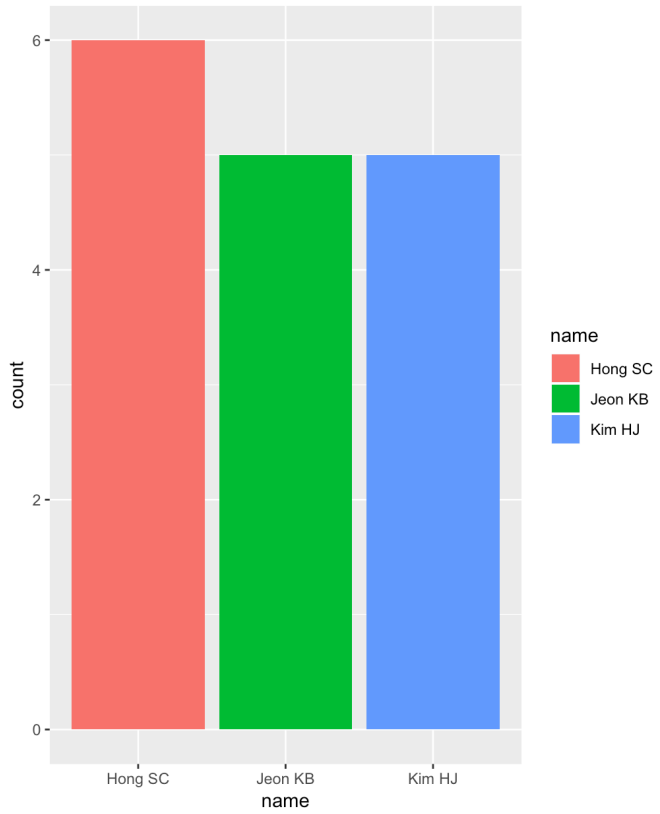
[9] 주소에 따른 정비소 출력 기능

```
> print_repair_shop('Seoul') # Seoul에 위치한 repair_shop
  shop_id   shop_name                                shop_addr  shop_phone shop_admin_name     shop_admin_email
1       1     Sungwoo      Seongsu-dong 1-ga Seongdong-gu Seoul 02-462-3554         Jang SW       jsm19@daum.net
2       2    KIA auto Seocho-gil 42 Seocho-dong Seocho-gu Seoul 02-522-0427         Kang BR borissal12@gmail.com
3       3 Renault Sky           Hyoja-dong 65-2 Jongno-gu Seoul 02-739-6677           Do BS   tobong55@naver.com
```

[10] 특정 렌터카 출력 기능

```
> print_cars_2010_5people()
  car_id car_name car_rent_pay
1      8     Mini       130000
2     10      Sm3        91000
3     11   Sonata        99000
```

[11] 렌터카 대여 내역 통계 기능



4. **과제를 하면서 느낀 점**

이번 과제를 하기 전까지는 DB 가 다른 app 들의 기반이 되는 시스템이라는 것을 잊고 있었습니다. 끊임없이 DB 그 자체에 대해서만 공부하다보니 가장 중요한 사실을 잊고 있었던 것 같습니다.

하지만 이번 과제를 통해 DB 는 데이터를 활용하는 프로그램이라면 반드시 필요한 시스템이라는 것을 MySQL 에 데이터를 저장하고, 저장한 데이터를 R 을 통해 목적에 맞게 가공하는 과정을 통해 느꼈습니다.

처음에는 어려움도 많았습니다. 소소하고 잡다한 문제가 문제를 해결할 때마다 나타났습니다. 예를 들어 데이터 타입 때문에 출력이 이상하게 나오기도 하고, 쿼리가 가지 않아서 애를 먹기도 했습니다. 쿼리가 가지 않은 이유가 단지 R 에서 작은 따옴표를 ₩'와 같은 형식으로 써야한다는 것 때문이라는 것을 한시간 씩 헤매고 나서야 알게 됐을 땐, 황당하기도 하고 그나마 해결해서 다행이라고 생각하기도 했습니다. 워낙에 많은 오류가 생겨서 나중에는 무덤덤해지기도 했습니다. 하지만 이 과정을 통해서 소소하고 잡다한 오류들을 해결하는 방법을 알게 돼서 기쁩니다.

만약 이번 과제를 하지 않았다면 나중에 DB 를 다른 app 과 연결시켜야 하는 상황이 생기더라도, 해보지 않았다는 것에 대한 막연한 두려움 때문에 잘 할 수 없었을 것입니다. 이렇게 R 과 MySQL 을 연결해서 사용해봄으로써 두려움이 사라졌고 어떠한 과정을 거쳐 DB 와 응용 app 이 데이터를 주고받는 것인지 알게 돼서 다행이라고 생각합니다.

이번에 실습한 데이터는 모두 크기가 작은 데이터들이었기 때문에 DB 없이 분석을 했어도 속도에는 큰 차이가 없을 것입니다. 하지만 만약 빅데이터라고 부를만한 데이터를 다룰 때는 DB 를 통해 app 을 다루는 것과 그렇지 않은 것의 차이가 클 것이기 때문에, 조만간에 제법 용량이 큰 데이터를 가지고 DB 를 사용하는 것과 사용하지 않는 것을 비교해보고자 하는 계획도 세웠습니다.

정말 이번 과제를 통해 제가 조금이라도 성장했다는 것을 느꼈습니다. 좋은 과제 내주셔서 감사합니다!