

Tidyverse Practice

Juhyuk

2022-07-24

R Markdown

이것은 R Markdown 파일입니다. 위에 Knit 이라고 보이는 부분에서 “Knit to Word”를 선택해주세요. 혹시 에러가 뜬다면, 그냥 rmd 파일 그대로 제출해도 됩니다. 코드는 회색 부분에 작성하시면 됩니다. word 로 변환한 후, pdf 로 다시 저장하시고, 이 rmd 파일과 같이 github 로 업로드 한다음, github 주소를 보내주세요. 같이 드린 reserve.csv 는 저작권 문제 때문에 github 에 업로드하면 안됩니다.

회색 부분에 코드를 작성해주시면 됩니다. 빈칸을 채워야하는 경우는 ##### 처리를 해두었습니다.

목요일 자정까지 꼭 제출해주세요. 혹시 markdown 이나 github 관련하여 어려움이 있으시다면, 언제든지 연락주시면 됩니다.

필요한 Package 가져오기

이번 과제는 Tidyverse 연습을 하는 것입니다. Tidyverse 패키지를 불러와봅시다.

```
library(tidyverse)

## — Attaching packages ————— tidyverse 1.3.1 —

## ✔ ggplot2 3.3.6      ✔ purrr  0.3.4
## ✔ tibble  3.1.7      ✔ dplyr  1.0.9
## ✔ tidyr   1.2.0      ✔ stringr 1.4.0
## ✔ readr   2.1.2      ✔ forcats 0.5.1

## — Conflicts ————— tidyverse_conflicts() —
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()     masks stats::lag()
```

데이터 불러오기

같이 드린 reserve.csv 파일은 호텔 예약 정보를 모아둔 데이터셋입니다.

reserve.csv * reserve_id: 예약 ID * hotel_id: 호텔 ID * customer_id: 고객 ID *
reserve_datetime: 예약 일시 * checkin_date: 체크인 날짜 * checkin_time: 체크인 시각 *
checkout_date: 체크아웃 날짜 * people_num: 숙박 인원 * total_price: 합계 금액

같이 드린 reserve.csv 파일을 reserve 라는 이름의 tibble 로 불러와봅시다.

```
reserve <- as_tibble(read.csv("reserve.csv"))
```

Q1: reserve 데이터는 몇개의 row 가 있나요?

```
nrow(reserve)
## [1] 4030
```

가장 저렴한 호텔

알린이는 친구들과 갈 때 가장 저렴한 호텔을 찾고 있습니다. 1 일 기준 평균적인 예약가격이 가장 저렴한 호텔을 찾고 있습니다. 알린이는 해킹 능력이 매우 뛰어나 호텔 데이터베이스를 해킹해서 필요한 데이터를 가져왔지만, 데이터 분석은 해본적이 없어서, 가장 저렴한 호텔이 어딘지 계산하는거에 어려움을 겪고 있습니다. 우리가 도와줍시다.

Q2. select() 함수를 사용하여 분석에 필요한 열만을 추출하여 reserve_tb 라는 새로운 tibble 에 저장합시다. 1 일 기준 정보가 필요하니까 체크인 체크아웃 날짜도 필요하겠죠? 우선은 사람이 많아도 방 하나 가격은 같다고 가정하고, people_num 은 계산에서 제외합시다.

```
reserve_tb <- select(reserve, hotel_id, checkin_date, checkout_date, total_price)
reserve_tb

## # A tibble: 4,030 × 4
##   hotel_id checkin_date checkout_date total_price
##   <chr>      <chr>          <chr>          <int>
## 1 h_75      2016-03-26      2016-03-29      97200
## 2 h_219     2016-07-20      2016-07-21      20600
## 3 h_179     2016-10-19      2016-10-22      33600
## 4 h_214     2017-03-29      2017-03-30     194400
## 5 h_16      2017-09-22      2017-09-23      68100
```

```
## 6 h_241      2017-12-04    2017-12-06          36000
## 7 h_256      2018-01-25    2018-01-28         103500
## 8 h_241      2018-06-08    2018-06-09           6000
## 9 h_217      2016-03-25    2016-03-27          68400
## 10 h_240     2016-07-14    2016-07-17         320400
## # ... with 4,020 more rows
```

tibble 은 자동으로 날짜 형식의 chr 을 date 타입의 변수로 바꿔줍니다. 밑에는 date 타입의 변수끼리의 차이를 계산하는 방법에 대한 예시입니다. as.Date 는 chr 을 date 타입으로 바꿔주는 함수이지만, reserve_tb 는 이미 자동으로 date 타입 변수로 만들어졌지요.

```
as.Date("2016-03-26") - as.Date("2016-03-23")
```

```
## Time difference of 3 days
```

mutate() 함수를 추가해서 num_date 라는, 호텔에 묵은 날짜 변수를 reserve_tb 에 추가하여 새로운 데이터 reserve_tb1 을 만들어봅시다. 날짜 데이터 다루는 법은 따로 다루지 않았기 때문에, 이번만은 정답을 보여드리겠습니다.

```
reserve_tb1 <- mutate(reserve_tb,
                      num_date = as.Date(checkout_date) - as.Date(checkin_date))
reserve_tb1
```

```
## # A tibble: 4,030 × 5
##   hotel_id checkin_date checkout_date total_price num_date
##   <chr>      <chr>          <chr>          <int> <drtn>
## 1 h_75      2016-03-26    2016-03-29          97200 3 days
## 2 h_219     2016-07-20    2016-07-21          20600 1 days
## 3 h_179     2016-10-19    2016-10-22          33600 3 days
## 4 h_214     2017-03-29    2017-03-30         194400 1 days
## 5 h_16      2017-09-22    2017-09-23          68100 1 days
## 6 h_241     2017-12-04    2017-12-06          36000 2 days
## 7 h_256     2018-01-25    2018-01-28         103500 3 days
## 8 h_241     2018-06-08    2018-06-09           6000 1 days
## 9 h_217     2016-03-25    2016-03-27          68400 2 days
## 10 h_240    2016-07-14    2016-07-17         320400 3 days
## # ... with 4,020 more rows
```

date 끼리의 계산은 difftime 으로 time 변수로 저장됩니다. 3 days, 5 days 이렇게 저장되지요. numeric 변수인 total_price 와 계산하기 위해서는 num_date 도 time 변수형이 아니라, numeric 변수형으로 저장 되어야합니다. 그래서 as.numeric 으로 변수형을 바꾸어주었습니다.

Q3. mutate() 함수를 사용하여 price_per_day 라는 1 일당 숙박 가격 변수를 추가한 새로운 데이터 reserve_tb2 를 만들어봅시다.

```
reserve_tb2 <- mutate(reserve_tb1,
                        price_per_day = total_price / as.numeric(num_date)
                        )
reserve_tb2
```

```
## # A tibble: 4,030 × 6
##   hotel_id checkin_date checkout_date total_price num_date price_per_day
##   <chr>      <chr>      <chr>      <int> <drtn>      <dbl>
## 1 h_75      2016-03-26    2016-03-29      97200 3 days      32400
## 2 h_219     2016-07-20    2016-07-21      20600 1 days      20600
## 3 h_179     2016-10-19    2016-10-22      33600 3 days      11200
## 4 h_214     2017-03-29    2017-03-30     194400 1 days     194400
## 5 h_16      2017-09-22    2017-09-23      68100 1 days      68100
## 6 h_241     2017-12-04    2017-12-06      36000 2 days      18000
## 7 h_256     2018-01-25    2018-01-28     103500 3 days      34500
## 8 h_241     2018-06-08    2018-06-09       6000 1 days        6000
## 9 h_217     2016-03-25    2016-03-27      68400 2 days      34200
## 10 h_240    2016-07-14    2016-07-17     320400 3 days     106800
## # ... with 4,020 more rows
```

Q4. group_by() 와 summarize() 함수를 이용해서 각 호텔별로 1 일당 평균 숙박 가격을 계산하여 reserve_tb3 에 저장해봅시다.

```
reserve_tb3 <- group_by(reserve_tb2, hotel_id) %>%
  summarize(mean_PPD = mean(price_per_day))
reserve_tb3
```

```
## # A tibble: 300 × 2
##   hotel_id mean_PPD
##   <chr>      <dbl>
## 1 h_1        67860
## 2 h_10       14933.
## 3 h_100      12960
## 4 h_101      33765.
## 5 h_102      16615.
## 6 h_103      46980
## 7 h_104      84400
## 8 h_105      25407.
## 9 h_106      66600
## 10 h_107     56400
## # ... with 290 more rows
```

Q5. arrange() 함수를 이용하여 1 일당 평균 숙박 가격을 오름차순으로 정렬해봅시다. 가장 저렴한 호텔은 어디인가요?

```
arrange(reserve_tb3, -desc(mean_PPD))
```

```
## # A tibble: 300 × 2
##   hotel_id mean_PPD
##   <chr>      <dbl>
## 1 h_235      8750
## 2 h_35       9406.
## 3 h_197     10133.
## 4 h_44      10574.
## 5 h_224     10667.
## 6 h_74      10909.
## 7 h_15      11108.
## 8 h_41      11345.
## 9 h_24      11500
## 10 h_50     11769.
## # ... with 290 more rows
```

Q6. 알린이는 형 알령이가 다른 호텔에서 혼자서 4500 원에 묵은 사실을 알고 데이터 분석 결과에 의문을 가졌습니다. 아차! people_num 에 따라 가격이 달라진다는 걸 고려하지 못했네요. 이번에는 1 인 기준으로 다시 summarize()를 해봅시다. 그런데 이번에는 여러 데이터프레임을 계속 저장하지 말고 한번에 계산을 하도록 파이프 (%>%)를 활용해서 한번에 계산을 해보아요!!

```
reserve %>%
  mutate(
    num_date = as.Date(checkout_date) - as.Date(checkin_date),
    realprice_per_day = total_price / as.numeric(num_date) / people_num
  ) %>%
  group_by(hotel_id) %>%
  summarize(mean_RPD = mean(realprice_per_day)) %>%
  arrange(-desc(mean_RPD))
```

```
## # A tibble: 300 × 2
##   hotel_id mean_RPD
##   <chr>      <dbl>
## 1 h_35       3500
## 2 h_53       3700
## 3 h_41       3900
## 4 h_197     4000
## 5 h_224     4000
## 6 h_235     4200
## 7 h_15     4300
## 8 h_24     4600
## 9 h_13     4700
## 10 h_100    4800
## # ... with 290 more rows
```

호텔 매출 비교

통린이는 호텔 그룹 CEO 입니다. 최근에 코로나로 디폴트 위기에 처하자, 매출이 낮은 호텔부터 처분하기로 하였습니다. 그런데 통린이는 주가가 하락할때부터 데이터분석 팀을 모두 해고해버렸기 때문에, 현재 가장 매출이 낮은 호텔이 어디인지 알수가 없네요. 통린이는 2017 년 6 월 이후 예약 기준으로 총 예약횟수 그리고 총 매출을 낮은순으로 나열한 데이터를 깔끔하게 보고 싶습니다. 통린이를 도와줍시다!! summarize 에서 count(아무 변수)를 사용하면, 데이터프레임 내에서 그 호텔별로 예약수를 구할 수 있습니다. date 비교는 다음과 같이 할 수 있습니다.

```
install.packages("lubridate")

## 'C:/Users/Ju Hyuk Lee/AppData/Local/R/win-library/4.2'의 위치에 패키지(들)을
## 설치합니다.
## (왜냐하면 'lib'가 지정되지 않았기 때문입니다)

## Error in contrib.url(repos, "source"): 미리 설정을 하지 않고서는 CRAN 을 사용
## 할 수 없습니다

library(lubridate)                                # 날짜와 시간 계산을 용이하게 해주는 패키
지/

##
## 다음의 패키지를 부착합니다: 'lubridate'

## The following objects are masked from 'package:base':
##
##   date, intersect, setdiff, union

print(today())                                    # today()는 오늘 날짜를 계산하고, print()
는 출력을 해줍니다.

## [1] "2022-07-25"

as.Date("2016-03-14") > today()                  # as.Date 는 문자열을 date 로 바꿔주고, 숫
자처럼, <와 >로 날짜를 비교할 수 있습니다.

## [1] FALSE
```

여기에 파이프를 사용해서 작성해주세요.

```

reserve %>%
  filter(reserve_datetime >= as.Date("2017-06-01")) %>%
  group_by(hotel_id) %>%
  summarize(total_num = n(),
             total_revenue = sum(total_price)) %>%
  arrange(-desc(total_revenue))

## # A tibble: 291 × 3
##   hotel_id total_num total_revenue
##   <chr>      <int>      <int>
## 1 h_24         1        4600
## 2 h_76         1        9200
## 3 h_229        1       14800
## 4 h_108        1       17200
## 5 h_102        1       18000
## 6 h_265        1       18600
## 7 h_279        2       18900
## 8 h_122        1       20100
## 9 h_208        1       26700
## 10 h_18        1       27800
## # ... with 281 more rows

```