Raport pracy projektowej nr 2

Mariola Bartosik

Wstep

W raporcie przedstawie wyniki swoich badan prowadzonych na ramkach danych, dostepnych na stronie internetowej http://www.gagolewski.com/resources/data/.

Korzystałam z danych przedstawonych poniżej.

```
options(stringsAsFactors=FALSE)
Tags <- read.csv("~/R/travel_stackexchange_com/Tags.csv")
Badges <- read.csv("~/R/travel_stackexchange_com/Badges.csv")
Comments <- read.csv("~/R/travel_stackexchange_com/Comments.csv")
Posts <- read.csv("~/R/travel_stackexchange_com/Posts.csv")
Users <- read.csv("~/R/travel_stackexchange_com/Users.csv")
Votes <- read.csv("~/R/travel_stackexchange_com/Votes.csv")
PostLinks <- read.csv("~/R/travel_stackexchange_com/PostLinks.csv")
```

Kazde zadanie wykonane jest na 4 sposoby, za pomoca

- funkcji bazowych R,
- funkcji z biblioteki sqldf,
- funkcji z biblioteki dplyr,
- funkcji z biblioteki data.table.

Dodatkowo porównałam czasy wykonania napisanych przeze mnie funkcji przy użyciu jednego wywołania microbenchmark::microbenchmark().
W wiekszości zadań okazało sie, że funkcje korzystajace z biblioteki sqldf, osiagały najgorsze wyniki w badaniach, natomiast głównie to funkcje opierajace sie na funkcjach bazowych jezyka R były najszybsze.

W raporcie sprawdziłam także równoważność moich funkcji za pomoca

```
dplyr::all_equal(x, y)
```

```
## Error in equal_data_frame(target, current, ignore_col_order = ignore_col_order, :  nie znaleziono obiektu 'x'
```

Zadanie 1

Zadanie nr 1 bazowało na ramce danych Posts. Wybrałam z kolumny PostTypeId wiersze, z wartosciami rownymi 1, z kolumny FavoriteCount wiersze z wartosciami wiekszymi rownymi 25 oraz z kolumny ViewCount wiersze z wartosciami wiekszymi rownymi 10000. Na samym końcu wyodrebniłam kolumny o nazwach: "Title, Score, ViewCount, FavoriteCount ".

Wyniki w 4 przypadkach były identyczne. Uzyskałam ramke z czterema kolumnami i dziewietnastoma wierszami.

```
head(df_dplyr_1(), 2)
```

```
##                                                                                Title
## 1 When traveling to a country with a different currency, how should you take your money?
## 2                                            How can I do a "broad" search for flights?
##   Score ViewCount FavoriteCount
## 1   136     16838            35
## 2    95     33554            49
```

```
head(df_table_1(), 2)
```

```
##                                                                                 Title
## 1: When traveling to a country with a different currency, how should you take your money?
## 2:                                             How can I do a "broad" search for flights?
##    Score ViewCount FavoriteCount
## 1:   136     16838            35
## 2:    95     33554            49
```

Wyniki sprawdzenia równoważności:

```
dplyr::all_equal(df_sql_1(), df_table_1())
```

```
## [1] TRUE
```

```
dplyr::all_equal(df_sql_1(), df_base_1())
```

```
## [1] TRUE
```

Porównałam czasy wykonania napisanych przeze mnie funkcji w tym zadaniu przy użyciu wywolania microbenchmark::microbenchmark().

Zadanie 2

Zadanie nr 2 polegało na wybraniu odpowiednich wartości z kolumny OwnerUserId (rożnych od -1) z ramki Tags. Korzystałam dwykrotnie z inner join.
Złaczenie tego typu zachowuje tylko wiersze wystepujace w obydwu złaczonych zbiorach.
Zlaczeniu uległy kolumny WikiPostId oraz Posts z ramek Tags i Posts, a także kolumny AccountId i OwnerUserId z ramek Tags i Users. Posortowałam malejaco kolumne powstala po zliczeniu oraz wybrałam kolumny TagName, Count, OwnerUserId, Age, Location, DisplayName.

```
head(df_base_2(), 2)

##     TagName Count OwnerUserId Age         Location DisplayName
## 711  canada   802         101  34    Mumbai, India       hitec
## 740  europe   681         583  35 Philadelphia, PA Adam Tuttle

head(df_dplyr_2(), 2)

##   TagName Count OwnerUserId Age         Location DisplayName
## 1  canada   802         101  34    Mumbai, India       hitec
## 2  europe   681         583  35 Philadelphia, PA Adam Tuttle

head(df_table_2(), 2)

##    TagName Count OwnerUserId Age         Location DisplayName
## 1:  canada   802         101  34    Mumbai, India       hitec
## 2:  europe   681         583  35 Philadelphia, PA Adam Tuttle
```

Wyniki sprawdzenia równoważności:

```
dplyr::all_equal(df_sql_2(), df_table_2())

## [1] TRUE

dplyr::all_equal(df_sql_2(), df_dplyr_2())

## [1] TRUE
```

Porównałam czasy wykonania napisanych przeze mnie funkcji w tym zadaniu przy użyciu wywolania microbenchmark::microbenchmark(). Oto wyniki:

Zadanie 3

Zadanie nr 3 polegało na pogrupowaniu wzgledem RelatedPostId. Zliczyłam wartosci w RelatedPostId i zapisałam kolumnw z wartościami zliczenia jako NumLinks. Poza tym należało zmiecić nazwe RelatedPostId na PostId, a także zapisac wynik w postaci ramki danych o nazwie RelatedTab. Kolejnym krokiem było złaczenie z ramek danych RelatedTab i Posts kolumn PostId i Id zachowujac wiersze wystepujace w obydwu kolumnach. Nastepnie wybrałam wartości w kolumnie PostTypeId równe 1. Posortowałam malejaco kolumne NumLinks oraz wybrałam poszczególne kolumny takie jak TagName, Count, OwnerUserId, Age, Location, DisplayName.

```
head(df_base_3(), 3)
```

```
##                                                                    Title
## 2262 Is there a way to find out if I need a transit visa for a layover in the UK?
## 2165               Do I need a visa to transit (or layover) in the Schengen area?
## 1168        Should my first trip be to the country which issued my Schengen Visa?
##      NumLinks
## 2262      594
## 2165      585
## 1168      331
```

```
head(df_sql_3(), 2)
```

```
##                                                                  Title
## 1 Is there a way to find out if I need a transit visa for a layover in the UK?
## 2               Do I need a visa to transit (or layover) in the Schengen area?
##   NumLinks
## 1      594
## 2      585
```

Sprawdziłam równoważność funkcji:

```
dplyr::all_equal(df_sql_3(), df_table_3())
```

```
## [1] TRUE
```

```
dplyr::all_equal(df_base_3(), df_dplyr_3())
```

```
## [1] TRUE
```

Porównałam czasy wykonania napisanych przeze mnie funkcji w tym zadaniu przy użyciu wywolania microbenchmark::microbenchmark().

Zadanie 4

Zadanie nr 4 na poczatku polegało na znalezieniu odpowiednich wartosci Name w ramce Badges. Takich, dla ktorych dane w kolumnie Class maja wartość równa 1. Pogrupowałam ramke wzgledem Name. Wybrałam wartości po zliczeniu wieksze od dwóch i mniejsze od 10. Potem wybrałam tylko te wartości Name z Users, które dostaliśmy po wyselekcjowaniu z Badges. Także, należało wybrać dane, dla których wartość z kolumny Class jest równa 1. Zapisałam to jako ValuableBadges. Złaczyłam tak jak wyżej z ramek Users i Badges odpowiednio kolumny UserId i Id. Na samym końcu wybrałam odpowiednio unikalne wartosci z kolumn Id, DisplayName, Reputation, Age, Location.

```
head(df_base_4(), 5)
```

```
##    Id        DisplayName Reputation Age                          Location
## 1  19              VMAtm      18556  33           Tampa, FL, United States
## 2 101          Mark Mayo     121667  37 Sydney, New South Wales, Australia
## 4 108     Ankur Banerjee      31273  27                         London, UK
## 6 466     iHaveacomputer       8360  NA                       Down underer
## 7 693 RoflcoptrException      33300  NA
```

Sprawdziłam równoważność funkcji:

```
dplyr::all_equal(df_sql_3(), df_table_3())
```

```
## [1] TRUE
```

```
dplyr::all_equal(df_base_3(), df_dplyr_3())
```

```
## [1] TRUE
```

Porównałam czasy wykonania napisanych przeze mnie funkcji w tym zadaniu przy użyciu wywolania microbenchmark::microbenchmark().

Zadanie 5

Zadanie nr 5 polegało na wybraniu z ramki Votes kolumny PostId, gdzie wartości w kolumnie VoteTypeId sa równe 2. Pogrupowałam PostId. Zliczyłam wartości i zapisałam jako UpVotes. Powstała nowa ramka, ktora zapisałam jako UpVotesTab. Analogicznie postepowałam przy tworzeniu DownVotesTab, gdzie zliczone wartosci zapisałam jako DownVotes, a wczesniej z kolumny VoteTypeId wybrałam wartości równe 3. Złaczyłam za pomoca left join kolumny PostId z obu ramek. Zlaczenie tego typu wybiera każdy wiersz z pierwszego zbioru, dołaczajac do niego wszystkie pasujace wiersze z drugiego zbioru wystepujace w lewym zbiorze. Wybrałam kolumny PostId, UpVotes, DownVotes, gdzie kolumnie DownVotes wszystkie wartości nieokreślone zostały zamienione zerem.

```
head(df_base_5(), 5)

##   PostId UpVotes DownVotes
## 1      1      10         2
## 2      2      32         0
## 3      3      13         1
## 4      4       9         1
## 5      5      14         0

head(df_table_5(), 5)

##    PostId UpVotes DownVotes
## 1:      1      10         2
## 2:      2      32         0
## 3:      3      13         1
## 4:      7       6         3
## 5:      6      82         0
```

Sprawdziłam równoważność funkcji:

```
dplyr::all_equal(df_sql_5(), df_table_5())

## [1] TRUE

dplyr::all_equal(df_sql_5(), df_dplyr_5())

## [1] TRUE
```

Porównałam czasy wykonania napisanych przeze mnie funkcji w tym zadaniu przy użyciu wywolania microbenchmark::microbenchmark().

Zadanie 6

Zadanie nr 6 było podobne do zadania 5, różnica, polegała na kolejności w złaczeniach. Przez co dostałam dwie różne ramki danych. Nastepnie nalezało połaczyć je i wybrać kolumny PostId i kolumne, kotra powstała po odjeciu UpVotes od DownVotes.

```
head(df_table_6(), 5)
```

```
##    PostId Votes
## 1:      1     8
## 2:      2    32
## 3:      3    12
## 4:      7     3
## 5:      6    82
```

```
head(df_sql_6(), 5)
```

```
##   PostId Votes
## 1      1     8
## 2      2    32
## 3      3    12
## 4      4     8
## 5      5    14
```

Sprawdziłam równoważność funkcji:

```
dplyr::all_equal(df_sql_6(), df_table_6())
```

```
## [1] TRUE
```

```
dplyr::all_equal(df_sql_6(), df_dplyr_6())
```

```
## [1] TRUE
```

Porównałam czasy wykonania napisanych przeze mnie funkcji w tym zadaniu przy użyciu wywolania microbenchmark::microbenchmark().