Praca projektowa nr 2
Rozwiązanie zadań
Maciej Chylak 305699 IAD

1 Wprowadzenie

Tytułem wstępu chciałbym napomnieć, iż ocena pomiaru niżej wymienionych rozwiązań zadań jest zależna od kilku czynników, z których główną rolę odgrywa implementacja poszczególnych funkcji. Warto także wziąć pod uwagę fakt, iż funkcję z rodziny df_table jako argument przyjmują ramki danych wczytane w formacie data.table, co może również znacząco wpłynąć na ostateczny rezultat.

Z uzyskaniem poprawnych rezultatów w każdym zadaniu najlepiej poradziły sobie funkcje z rodziny data_table. Niemniej jednak funkcje te mają jedną poważną wadę, mianowicie odchylenie standardowe, w porównaniu z innymi sposobami rozwiązania zadania, w każdej z tych funkcji było największe, zatem istnieję niezerowa szansa na to, że nasze rozwiązanie w pewnych przypadkach nie okaże się być tym najszybszym.

Z kolej jeżeli chodzi o funkcje z rodziny df_dplyr, wyniki również okazały się być zadowalające. Plusem dla tego typu implementacji jest klarowność opisu,

Funkcje z rodziny df_base były w podobnym stopniu skuteczne jak w przypadku df_dplyr. Wyjątek stanowiły df_base_5 oraz df_base_6, w których to pod kątem prędkości, być może z powodu niewłaściwej implementacji, wypadły najgorzej z całej czwórki.

Jeżeli chodzi o funkcje z rodziny df_sql, to pod względem prędkości wypadły one najgorzej. Jednak podobnie jak w przypadku funkcji z rodziny df_dplyr ich składnia jest bardzo przejrzysta.

2 Zadanie 01

2.1 Treść

Wybierz te wiersze z ramki Posts, w których PostTypeId = 1, FavoriteCount >= 25 i ViewCount >= 1000, a następnie wybierz kolumny Title, Score, ViewCount, FavouriteCount

2.2 Wynik

```
##                                                                                                  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?
## 3:                                    Tactics to avoid getting harassed by corrupt police?
##    Score ViewCount FavoriteCount
## 1:   136     16838            35
## 2:    95     33554            49
## 3:   156     13220            42
```

2.3 Sprawdzenie poprawności każdej funkcji

```
c(dplyr::all_equal(df_base_1(Posts), df_sql_1(Posts)) + dplyr::all_equal(df_sql_1(Posts), df

## [1] TRUE
```

2.4 Porównanie czasowe

```
## Unit: milliseconds
##        expr        min         lq       mean     median         uq       max
##       sqldf 166.678625 170.642474 181.426036 178.007900 191.655033 231.53607
##        base   2.089642   2.290913   3.626708   2.414404   2.718730  44.09640
##       dplyr   5.236137   5.601258   6.745126   5.988814   6.706767  41.28469
##  data.table   1.716773   1.833424   3.018755   1.949114   2.172851  37.03111
##  neval
##    100
##    100
##    100
##    100
```

3 Zadanie 02

3.1 Treść

Złącz wewnętrznie ramkę Tags z ramką Posts według atrybutów Id z ramki Posts oraz WikiPostId z ramki Tags. Następnie złącz wewnętrznie z ramką utworzoną z ramki Users, w której OwnerUserId jest różne od -1, według atrybutów AccountId z ramki Users oraz OwnerUserId z ramki Posts. Posortuj malejąco według Count. Wybierz kolumny TagName, Count, OwnerUserId, Age, Location, DisplayName.

3.2 Wynik

```
##          TagName Count OwnerUserId Age         Location      DisplayName
## 1:        canada   802         101  34    Mumbai, India            hitec
## 2:        europe   681         583  35 Philadelphia, PA      Adam Tuttle
## 3: visa-refusals   554        1737  34     New York, NY Benjamin Pollack
```

3.3 Sprawdzenie poprawności każdej funkcji

```
c(dplyr::all_equal(df_base_2(Tags, Posts, Users), df_sql_2(Tags, Posts, Users)) + dplyr::al

## [1] TRUE
```

3.4 Porównanie czasowe

```
## Unit: milliseconds
##        expr        min         lq       mean     median         uq       max
##       sqldf 305.401793 326.947592 342.466708 339.928287 355.266732 427.06501
##        base  17.855599  20.076373  21.459971  21.051645  21.911977  58.36345
##       dplyr  37.348393  45.368230  51.942066  50.419800  54.721677  99.35744
##  data.table   5.615607   6.114613   9.573559   6.505729   7.591769 189.53614
##  neval
##    100
##    100
##    100
##    100
```

4 Zadanie 03

4.1 Treść

Zlicz jako NumLinks liczbę wystąpień w ramce PostLinks pogrupowanej według RelatedPostId każdego z RelatedPostId. Wybierz kolumnę RelatedPostId jako PostId oraz NumLinks. Powstałą ramki jako RelatedTab złącz wewnętrznie z ramką utworzoną z ramki Posts, w której PostTypeId = 1, według atrybutów PostId z ramki RelatedTab oraz Id z ramki Posts. Posortuj malejąco według NumLinks. Wybierz kolumny Title, NumLinks

4.2 Wynik

```
##                                                                        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?
## 3:        Should my first trip be to the country which issued my Schengen Visa?
##    NumLinks
## 1:      594
## 2:      585
## 3:      331
```

4.3 Sprawdzenie poprawności każdej funkcji

```
c(dplyr::all_equal(df_base_3(Posts, PostLinks), df_sql_3(Posts, PostLinks)) + dplyr::all_equ

## [1] TRUE
```

4.4 Porównanie czasowe

```
## Unit: milliseconds
##        expr        min         lq      mean    median        uq       max neval
##       sqldf 206.981964 213.686094 227.85343 229.00274 238.45789 265.17720   100
##        base  65.459344  69.074590  77.41733  74.24324  77.38945 118.53831   100
##       dplyr  32.601266  37.135529  40.47075  38.99532  41.23602  77.29096   100
##  data.table   9.030452   9.778537  12.12507  10.29657  11.59151  51.39161   100
```

5 Zadanie 04

5.1 Treść

Z ramki Badges wybierz te wiersze, dla których Class = 1. Pogrupuj według Name, a następnie wybierz te, dla których liczba wystąpień danego Name należy do przedziału od 2 do 10. Wybierz kolumne Name. Z ramki Badges wybierz te wiersze, których Name należy do wybranych wcześniej wartości z kolumny Name oraz Class = 1. Wybierz kolumny Name, UserId. Powstałą ramkę jako ValuableBadges złącz wewnętrznie z ramką Users weðług atrybutów UserdId z ramki ValuableBadges oraz Id z ramki Users. Wybierz kolumny Id, DisplayName, Reputation, Age, Location.

5.2 Wynik

```
##     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
## 3: 108 Ankur Banerjee      31273  27                         London, UK
```

5.3 Sprawdzenie poprawności każdej funkcji

```
c(dplyr::all_equal(df_base_4(Users, Badges), df_sql_4(Users, Badges)) + dplyr::all_equal(df_

## [1] TRUE
```

5.4 Porównanie czasowe

```
## Unit: milliseconds
##        expr        min         lq      mean    median        uq       max neval
##       sqldf 240.665145 250.71327 266.86133 267.31658 280.70418 310.42367   100
##        base   9.957277  11.22723  16.31617  11.92749  12.76059  51.09766   100
##       dplyr  15.318522  17.90277  20.73757  18.96483  20.62099  56.55788   100
##  data.table   9.235210  10.15075  14.54691  10.99801  12.15271  77.82625   100
```

6 Zadanie 05

6.1 Treść

Zlicz jako UpVotes liczbę wystąpień w ramce powstałej z ramki Votes, w której VoteTypeId = 2, każdego z PostId. Wybierz kolumny PostId oraz UpVotes jako UpVotesTab. Zlicz jako DownVotes liczbę wystąpień w ramce powstałej z ramki Votes, w której VoteTypeId = 3, każdego z PostId. Wybierz kolumny PostId oraz DownVotes jako DownVotesTab. Złącz lewostronnie ramke UpVotesTab z DownVotesTab według argumentu PostId. Wartości NULL w kolumnie DownVotes zamień na 0 jako. Wybierz kolumny PostId. UpVotes oraz DownVotes.

6.2 Wynik

```
##    PostId UpVotes DownVotes
## 1:      1      10         2
## 2:      2      32         0
## 3:      3      13         1
```

6.3 Sprawdzenie poprawności każdej funkcji

```
c(dplyr::all_equal(df_base_5(Votes), df_sql_5(Votes)) + dplyr::all_equal(df_sql_5(Votes), df

## [1] TRUE
```

6.4 Porównanie czasowe

```
## Unit: milliseconds
##        expr        min         lq       mean     median         uq       max
##       sqldf  773.66248  806.08797  836.86365  835.06373  868.70729  917.7681
##        base 1238.70133 1337.84380 1405.35314 1393.30328 1449.23537 1695.9507
##       dplyr  155.83285  170.16933  198.45092  196.10796  208.46925  355.1940
##  data.table   33.75347   36.99911   53.02782   39.98653   70.66193  216.6116
##  neval
##    100
##    100
##    100
##    100
```

7 Zadanie 06

7.1 Treść

Zlicz jako UpVotes liczbę wystąpień w ramce powstałej z ramki Votes, w której VoteTypeId = 2, każdego z PostId. Wybierz kolumny PostId oraz UpVotes jako UpVotesTab. Zlicz jako DownVotes liczbę wystąpień w ramce powstałej z ramki Votes, w której VoteTypeId = 3, każdego z PostId. Wybierz kolumny PostId oraz DownVotes jako DownVotesTab. Złącz lewostronnie ramke UpVotesTab z DownVotesTab według argumentu PostId. Wartości NULL w kolumnie DownVotes zamień na 0. Wybierz kolumny PostId. UpVotes oraz DownVotes. Zlicz jako DownVotes liczbę wystąpień w ramce powstałej z ramki Votes, w której VoteTypeId = 3, każdego z PostId. Wybierz kolumny PostId oraz DownVotes jako DownVotesTab. Zlicz jako UpVotes liczbę wystąpień w ramce powstałej z ramki Votes, w której VoteTypeId = 2, każdego z PostId. Wybierz kolumny PostId oraz UpVotes jako UpVotesTab. Złącz lewostronnie ramke DownVotesTab z UpVotesTab według argumentu PostId. Wartości NULL w kolumnie UpVotes zamień na 0. Wybierz kolumny PostId. UpVotes oraz DownVotes. Wyznacz unie wyżej obu ramek. Dodaj kolumne UpVotes - DownVotes jako Votes. Wybierz kolumny PostId oraz Votes.

7.2 Wynik

```
##    PostId Votes
## 1:      1     8
## 2:      2    32
## 3:      3    12
```

7.3 Sprawdzenie poprawności każdej funkcji

```
c(dplyr::all_equal(df_base_6(Votes), df_sql_6(Votes)) + dplyr::all_equal(df_sql_6(Votes), df

## [1] TRUE
```

7.4 Porównanie czasowe

```
## Unit: milliseconds
##        expr        min        lq      mean     median         uq       max
##       sqldf 1536.39052 1628.6840 1691.7907 1698.98354 1741.07997 1909.5070
##        base 1476.72735 1562.6910 1647.6407 1623.92772 1692.37475 2062.1517
##       dplyr  206.93016  226.3988  257.1470  254.21267  269.32761  451.3782
##  data.table   46.25895   51.0891   68.4659   55.22374   85.28021  222.8583
```

```
##  neval
##    100
##    100
##    100
##    100
```