PADR 2020/2021

Praca domowa nr 2 (max. = 30 p.)

Maksymalna ocena: 30 p.

Termin oddania pracy: 18.12.2020 r., godz. 10:00

Prace domowe należy przesłać za pośrednictwem platformy `Moodle` – **jedno archiwum `.zip`**[1] o nazwie typu `Nazwisko_Imie_NrAlbumu_Nick_pd2.zip`. W archiwum znajdować się powinien jeden katalog, `Nazwisko_Imie_NrAlbumu_Nick_pd2`, dopiero w którym umieszczone zostaną następujące pliki:

- `Nazwisko_Imie_NrAlbumu_Nick_pd2.Rmd` (rozwiązanie zadań)
- `Nazwisko_Imie_NrAlbumu_Nick_pd2.html` (skompilowana wersja powyższego).

  Nazwy plików nie powinny zawierać polskich liter diakrytyzowanych (przekształć $ą \rightarrow a$ itd.).

  W nazwach plików wynikowych, `Nazwisko_Imie_NrAlbumu_Nick_pd1.(Rmd pdf R)`, `Nick` oznacza wybrany przez Państwa pseudonim, którego będziemy używać do publikowania wyników (inny niż nazwa użytkownika na platformie Github). Nick powinien być dokładnie taki sam jak w przypadku pracy domowej nr 1.

1 Zbiory danych

Będziemy pracować na uproszczonym zrzucie zanonimizowanych danych z serwisu https://travel.stackexchange.com/ (na marginesie: pełen zbiór danych dostępny jest pod adresem https://archive.org/details/stackexchange), który składa się z następujących ramek danych:

- Badges.csv.gz
- Comments.csv.gz
- PostLinks.csv.gz
- Posts.csv.gz
- Tags.csv.gz
- Users.csv.gz
- Votes.csv.gz

Przykładowe wywołanie — ładowanie zbioru `Tags`:

```
options(stringsAsFactors=FALSE)
# ww. pliki pobralismy do katalogu travel_stackexchange_com/
Tags <- read.csv("travel_stackexchange_com/Tags.csv.gz")
head(Tags)
```

Przed przystąpieniem do rozwiązywania zadań zapoznaj się z ww. serwisem oraz znaczeniem poszczególnych kolumn we wspomnianych ramkach danych, zob. http://www.gagolewski.com/resources/data/travel_stackexchange_com/readme.txt.

[1] A więc nie: .rar, .7z itp.

2 Informacje ogólne

Rozwiąż poniższe zadania przy użyciu wywołań funkcji bazowych oraz tych, które udostępniają pakiety `dplyr` oraz `data.table` – nauczysz się ich samodzielnie; ich dokumentację znajdziesz łatwo w internecie. Każdemu z 5 poleceń SQL powinny odpowiadać cztery równoważne sposoby ich implementacji w R, kolejno:

1. `sqldf::sqldf()`;
2. tylko funkcje bazowe (1 p.);
3. `dplyr` (1 p.);
4. `data.table` (1 p.).

Upewnij się, że zwracane wyniki są ze sobą tożsame (ewentualnie z dokładnością do permutacji wierszy wynikowych ramek danych, zob. np. funkcję `dplyr::all_equal` lub `compare::compare`) (1 p.). W każdym przypadku należy podać słowną (opisową) interpretację każdego zapytania (1 p.).

Ponadto w każdym przypadku porównaj czasy wykonania napisanych przez Ciebie wyrażeń przy użyciu jednego wywołania `microbenchmark::microbenchmark()` (0.5 p.), np.:

```
microbenchmark::microbenchmark(
   sqldf=rozwiazanie_sqldf,
   base=rozwiazanie_bazowe,
   dplyr=rozwiazanie_dplyr,
   data.table=rozwiazanie_datatable
)
```

Wszystkie rozwiązania umieść w jednym (estetycznie sformatowanym) raporcie knitr/Markdown. Za bogate komentarze do kodu, dyskusję i ewentualne rozwiązania alternatywne można otrzymać max. 2.5 p.

3 Zadania do rozwiązania

```
--- 1)
SELECT Posts.Title, RelatedTab.NumLinks
FROM
    (SELECT RelatedPostId AS PostId, COUNT(*) AS NumLinks
     FROM PostLinks
     GROUP BY RelatedPostId) AS RelatedTab
JOIN Posts ON RelatedTab.PostId=Posts.Id
WHERE Posts.PostTypeId=1
ORDER BY NumLinks DESC
```

```
--- 2)
SELECT
    Users.DisplayName,
    Users.Age,
    Users.Location,
    SUM(Posts.FavoriteCount) AS FavoriteTotal,
    Posts.Title AS MostFavoriteQuestion,
    MAX(Posts.FavoriteCount) AS MostFavoriteQuestionLikes
FROM Posts
JOIN Users ON Users.Id=Posts.OwnerUserId
WHERE Posts.PostTypeId=1
GROUP BY OwnerUserId
ORDER BY FavoriteTotal DESC
LIMIT 10
```

```sql
--- 3)
SELECT
    Posts.Title,
    CmtTotScr.CommentsTotalScore
FROM (
        SELECT
            PostID,
            UserID,
            SUM(Score) AS CommentsTotalScore
        FROM Comments
        GROUP BY PostID, UserID
) AS CmtTotScr
JOIN Posts ON Posts.ID=CmtTotScr.PostID AND Posts.OwnerUserId=CmtTotScr.UserID
WHERE Posts.PostTypeId=1
ORDER BY CmtTotScr.CommentsTotalScore DESC
LIMIT 10
```

```sql
--- 4)
SELECT DISTINCT
    Users.Id,
    Users.DisplayName,
    Users.Reputation,
    Users.Age,
    Users.Location
FROM (
        SELECT
            Name, UserID
        FROM Badges
        WHERE Name IN (
            SELECT
                Name
            FROM Badges
            WHERE Class=1
            GROUP BY Name
            HAVING COUNT(*) BETWEEN 2 AND 10
        )
        AND Class=1
    ) AS ValuableBadges
JOIN Users ON ValuableBadges.UserId=Users.Id
```

```sql
--- 5)
SELECT
    Questions.Id,
    Questions.Title,
    BestAnswers.MaxScore,
    Posts.Score AS AcceptedScore,
    BestAnswers.MaxScore-Posts.Score AS Difference
FROM (
        SELECT Id, ParentId, MAX(Score) AS MaxScore
        FROM Posts
        WHERE PostTypeId==2
        GROUP BY ParentId
    ) AS BestAnswers
JOIN (
```

```
        SELECT * FROM Posts
        WHERE PostTypeId==1
    ) AS Questions
    ON Questions.Id=BestAnswers.ParentId
JOIN Posts ON Questions.AcceptedAnswerId=Posts.Id
WHERE Difference>50
ORDER BY Difference DESC
```