

# DPRPy 2020/2021

## Homework assignment no. 1 (max. = 30 p.)

Maximum grade: 30 p.

Deadline: ~~30.11.2020, 23:59~~ **24.12.2020, 23.59**

Homework should be sent via the Moodle platform - **one archive .zip**<sup>1</sup> named

`Last-name_First-name_Student-number_Nickname_assgmet_1.zip`

(one directory inside: `Last-name_First-name_Student-number_Nick_assgmet_1`), in which the following files will be placed:

- `Last-name_First-name_Student-number_Nickname_assgmet_1.Rmd` (report prepared with Markdown / knitr containing task solutions, comments, etc.),
- `Last-name_First-name_Student-number_Nickname_assgmet_1.html` (compiled to HTML version of the above).

The `Nickname` is an identifier of your choice that will appear in the sheet ratings and will provide you with adequate anonymity. Remember it for uploading next homework, you must use the same nickname.

## 1 Data description

We are working on a simplified dump of anonymised data from the website <https://travel.stackexchange.com/> (by the way: full data set is available at <https://archive.org/details/stackexchange>), which consists of the following data frames:

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

Before starting to solve the problems familiarize yourself with the said service and data sets structure (e.g. what information individual columns represent), see [http://www.gagolewski.com/resources/data/travel\\_stackexchange\\_com/readme.txt](http://www.gagolewski.com/resources/data/travel_stackexchange_com/readme.txt).

Example: loading the set `Tags`:

```
options(stringsAsFactors=FALSE)
# if files are saved at "pd1/travel_stackexchange_com/" directory
Tags <- read.csv("pd1/travel_stackexchange_com/Tags.csv.gz")
head(Tags)
```

---

<sup>1</sup>So not: .rar, .7z etc.

## 2 Tasks description

Solve the following tasks using base functions calls and those provided by the `dplyr` and `data.table` packages - you will learn them on your own; their documentation (and tutorials) is easy to find online. Each of the **5 SQL queries** should have four implementations in R:

1. `sqldf::sqldf()` – reference solution;
2. only base functions (1 p.);
3. `dplyr` (1 p.);
4. `data.table` (1 p.).

Make sure that the obtained results are equivalent (possibly with an accuracy of the row permutation of the result data frames) (up to 1.5 p. for each task). You can propose a function that implements relevant tests (e.g. based on `compare::compare()` or `dplyr::all_equal()`) - the results of such comparisons should be included in the final report. In addition, compare the execution times written by you in each case using one call to `microbenchmark::microbenchmark()` (0.5 p.), e.g.:

```
microbenchmark::microbenchmark(  
  sqldf=sqldf_solution,  
  base=base_functions_solution,  
  dplyr=dplyr_solutions,  
  data.table=datatable_solution  
)
```

In addition, in each case, it is necessary to provide “intuitive” interpretation of each query (0.5 p.).

Put all solutions in one (nicely formatted) `knitr` / `Markdown` report. For rich code comments, discussion and possible alternative solutions you can obtain max. 2.5 p.

## 3 SQL queries

```
--- 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

```

```

--- 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

```

```

--- 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

```

```

--- 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

```