Praca domowa nr 1 - PD-R-Py

Igor Kołakowski

11 Kwietnia 2019

Wprowadzenie

Celem pracy domowej było przetłumaczenie 7 zapytań SQL na język R, czterema różnymi sposobami. Zadanie zostało rozwiązane przy użyciu:

1. sqldf::sqldf() <- funkcji przyjmującej zapytanie SQL w formie tekstu, wynik otrzymany za pomocą tej funkcji był rozwiązaniem referencyjnym do którego porównywane były wyniki otrzymane pozostałymi metodami.
2. Tylko funkcji bazowych języka R.
3. Pakietu dplyr.
4. Pakietu data.table.

Wszystkie zapytania zostały wykonane na uproszczonym zrzucie danych serwisu Travel Stack Exchange.

Wszystkie dane zostały załadowane do R-owych ramek danych. Rozwiązania sqldf, R i dplyr wykorzystają format data.frame.

```
Tags <- read.csv("C:\\ReposR\\R\\StackExchange\\Tags.csv.gz")
Badges <-read.csv("C:\\ReposR\\R\\StackExchange\\Badges.csv.gz")
Comments <-read.csv("C:\\ReposR\\R\\StackExchange\\Comments.csv.gz")
PostLinks <-read.csv("C:\\ReposR\\R\\StackExchange\\PostLinks.csv.gz")
Posts <-read.csv("C:\\ReposR\\R\\StackExchange\\Posts.csv.gz")
Users <-read.csv("C:\\ReposR\\R\\StackExchange\\Users.csv.gz")
Votes <-read.csv("C:\\ReposR\\R\\StackExchange\\Votes.csv.gz")
```

Natomiast rozwiązania pakietu data.table korzystają z formatu data.table.

```
library(data.table)
TagsDT <- data.table(Tags)
BadgesDT <-data.table(Badges)
CommentsDT <-data.table(Comments)
PostLinksDT <-data.table(PostLinks)
PostsDT <-data.table(Posts)
UsersDT <-data.table(Users)
VotesDT <-data.table(Votes)
```

Wszystkie wyniki, poza wzrokowym porównaniem przez osobę, która przygotowała ten raport, były sprawdzane wywołaniem funkcji:

```
cmpQueries(queryResult1,queryResult2)
```

Dla ramek danych (x,y) kod funkcji składa się z poniższych fragmentów:

1. Sprawdzenie czy kolumny mają identyczne nazwy i czy są ułożone w odpowiedniej kolejności

```
  if(!isTRUE(all.equal(colnames(x),colnames(y)))){
    all.equal(colnames(x),colnames(y))
    stop("Different colnames")
  }
```

2. Porównania liczby wierszy

```
if(nrow(x) != nrow(y)){
  stop("Different number of rows")
}
```

3. Uwzględnienia różnic które nie mają realnego wpływu na wynik zapytania, ale mogą mieć wpływ na wynik funkcji all.equal

```
x=as.data.frame(x)
y=as.data.frame(y)
x<-x[order(x),]
y<-y[order(y),]
rownames(x) <- NULL
rownames(y) <- NULL
```

4. Porównania zawartości wynikowych ramek danych zapytań

```
if(isTRUE(base::all.equal(x, y))){
  print("Query results equal")
}else{
  stop("Query results different")
}
```

Na koniec każdego rozdziału zostało przeprowadzone porównanie metod pod względem wydajności czasowej.

Zapytanie nr 1

Znajdź 10 użytkowników których posty były najczęściej dodawane do ulubionych, podaj ich podstawowe dane oraz pytanie które najczęściej trafiało do ulubionych.

```
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
```

sqldf

```
sqldf1 <- function(){
sqldf::sqldf("
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")
}
answer1 <- sqldf1()
answer1[,c("MostFavoriteQuestion")]
```

```
##  [1] "Tactics to avoid getting harassed by corrupt police?"
##  [2] "OK we're all adults here, so really, how on earth should I use a squat toilet?"
##  [3] "How to avoid drinking vodka?"
##  [4] "What is the highest viewing spot in London that is accessible free of charge?"
##  [5] "How do airlines determine ticket prices?"
##  [6] "Are there other places with gardens like those at Versailles?"
##  [7] "OK we're all nerds here, so really, how on earth should I use a Japanese toilet?"
##  [8] "Is there a good website to plan a trip via trains in Europe?"
##  [9] "What is the most comfortable way to sleep on a plane?"
## [10] "Should I submit bank statements when applying for a UK Visa? What do they say about me?"
```

R

```
R1 <- function(){
Questions <- Posts[Posts$PostTypeId==1,]
UP <- merge(x=Questions,y=Users,by.x="OwnerUserId",by.y = "Id",all.x=TRUE)
splitUP <- split(UP,UP$OwnerUserId)
UPBestQuest <- do.call(rbind,
                       lapply(splitUP, function(x) {return(x[which.max(x$FavoriteCount),])}))
favCount <- aggregate(x = UP["FavoriteCount"],
                      by = UP[c("OwnerUserId")],
                      function(x){FavoriteTotal=sum(x,na.rm = TRUE)})
colnames(favCount) <- c("OwnerUserId","FavoriteTotal")
alldata <- merge(UPBestQuest,favCount,by="OwnerUserId")
alldata <- alldata[order(alldata["FavoriteTotal"],decreasing = TRUE),]
q1R <- head(alldata[,c("DisplayName","Age","Location","FavoriteTotal","Title","FavoriteCount")],n=10)
colnames(q1R)[5] <- "MostFavoriteQuestion"
colnames(q1R)[6] <- "MostFavoriteQuestionLikes"
q1R
}
q1R <- R1()
```

dplyr

```
dplyr1 <- function(){
  Posts %>%  filter(PostTypeId==1) %>% inner_join(Users,by=c("OwnerUserId" = "Id"))  %>%
  group_by(OwnerUserId) -> grouped

grouped %>% mutate(FavoriteTotal = sum(FavoriteCount,na.rm=TRUE))  %>%
 slice(which.max(FavoriteCount)) %>%
 arrange(desc(FavoriteTotal)) %>% head(10) %>% ungroup %>%
 select(DisplayName,Age,Location,FavoriteTotal,
        MostFavoriteQuestion=Title,
        MostFavoriteQuestionLikes=FavoriteCount) -> q1dplyr
  q1dplyr}
 q1dplyr <- dplyr1()
```

data.table

```
dt1 <- function(){
  PostsDT[PostTypeId==1] %>%
  merge(y=UsersDT,by.x="OwnerUserId",by.y="Id") -> joinDT

joinDT[,.(DisplayName,Age,Location,FavoriteTotal = sum(FavoriteCount,na.rm=TRUE),
          MostFavoriteQuestion = Title[which.max(FavoriteCount)],
          MostFavoriteQuestionLikes = FavoriteCount[which.max(FavoriteCount)]),
    keyby=OwnerUserId][,-c("OwnerUserId")][order(-FavoriteTotal)] %>% unique %>% head(10) -> q1dt
q1dt}
q1dt <- dt1()
```

Wyniki nr 1

```
microbenchmark::microbenchmark(
  sqldf = sqldf1(),
  base = R1(),
  dplyr = dplyr1(),
  data.table = dt1()
)
```

```
## Unit: milliseconds
##        expr       min        lq      mean    median        uq        max
##       sqldf  317.3908  341.2804  370.4188  356.7182  379.0039   636.1381
##        base 7372.2975 8625.0294 9284.3641 9265.0370 9839.6048 11595.8034
##       dplyr  277.2712  316.6770  353.3276  333.8553  354.7088   735.1972
##  data.table  186.6632  199.2008  225.9993  211.5863  239.8223   524.5638
##  neval
##    100
##    100
##    100
##    100
```

```
all(cmpQueries(answer1,q1dplyr),
cmpQueries(answer1,q1R),
cmpQueries(answer1,q1dt))
```

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

Zapytanie nr 2

Znajdź 10 pytań z największą liczbą punktowanych odpowiedzi.

```
SELECT
    Posts.ID,
    Posts.Title,
    Posts2.PositiveAnswerCount
FROM Posts
JOIN (
        SELECT
            Posts.ParentID,
            COUNT(*) AS PositiveAnswerCount
        FROM Posts
        WHERE Posts.PostTypeID=2 AND Posts.Score>0
        GROUP BY Posts.ParentID
    ) AS Posts2
    ON Posts.ID=Posts2.ParentID
ORDER BY Posts2.PositiveAnswerCount DESC
LIMIT 10
```

sqldf

```
sqldf2 <- function(){
answer2 <- sqldf::sqldf("SELECT
Posts.ID,
Posts.Title,
Posts2.PositiveAnswerCount
FROM Posts
JOIN (
SELECT
Posts.ParentID,
COUNT(*) AS PositiveAnswerCount
FROM Posts
WHERE Posts.PostTypeID=2 AND Posts.Score>0
GROUP BY Posts.ParentID
) AS Posts2
ON Posts.ID=Posts2.ParentID
ORDER BY Posts2.PositiveAnswerCount DESC
LIMIT 10
")
answer2}
answer2 <- sqldf2()
answer2[,c("Title")]
```

```
##  [1] "Which European cities have bike rental stations for tourists?"
##  [2] "When traveling to a country with a different currency, how should you take your money?"
##  [3] "How do you choose a restaurant when travelling?"
##  [4] "How can I deal with people asking to switch seats with me on a plane?"
##  [5] "Why would you wrap your luggage in plastic?"
##  [6] "Traveling in Europe Solo - 18 years old. Feasible?"
##  [7] "Long-life SIM cards in Europe"
##  [8] "Am I expected to tip wait staff in Europe?"
##  [9] "Is there a way to prevent \"looking like a tourist\" in order to not be harassed?"
## [10] "Is it rude to ask if the food contains pork or alcohol?"
```

R

```
R2 <- function(){
  filtered <- Posts[Posts$PostTypeId==2 & Posts$Score>0,]
  sel <- by(filtered, filtered$ParentId , function(x){
  c(
    ParentId = unique(x$ParentId),
    PositiveAnswerCount = nrow(x)
     )
},simplify = FALSE)
select1 <- data.frame(do.call(rbind,sel))
q2R <- merge(Posts,select1,by.x="Id",by.y="ParentId")
q2R <- q2R[,c("Id","Title","PositiveAnswerCount")]
q2R <- head(q2R[order(q2R["PositiveAnswerCount"],decreasing = TRUE),],10)
q2R}
q2R <- R2()
```

dplyr

```
dplyr2 <- function(){
  q2dplyr <- Posts %>% filter(PostTypeId==2 & Score>0) %>% group_by(ParentId) %>%
  summarise(PositiveAnswerCount=n()) %>% inner_join(x=Posts,by=c("Id"="ParentId")) %>%
  arrange(desc(PositiveAnswerCount)) %>% head(10) %>% select(Id,Title,PositiveAnswerCount)
  q2dplyr}
q2dplyr <- dplyr2()
```

data.table

```
dt2 <- function(){
  PostsAnsCountDT <- PostsDT[PostTypeId==2 & Score>0, .(PositiveAnswerCount = .N), keyby=ParentId]
Posts2 <- merge(PostsDT,PostsAnsCountDT,by.x = "Id", by.y="ParentId")
q2dt <- head(Posts2[order(-PositiveAnswerCount),.(Id,Title,PositiveAnswerCount)],10)
q2dt}
q2dt <- dt2()
```

Wyniki nr 2

```
microbenchmark::microbenchmark(
  sqldf = sqldf2(),
  base = R2(),
  dplyr = dplyr2(),
  data.table = dt2()
)
```

```
## Unit: milliseconds
##        expr        min         lq       mean     median         uq
##       sqldf  216.88765  233.28944  245.59232  239.46633  249.33611
##        base 3937.01953 4155.36429 4350.67115 4271.47923 4431.13664
##       dplyr   46.82446   51.51447   58.90985   54.66402   60.86114
##  data.table   26.56312   30.80046   32.54602   31.68690   32.87446
##         max neval
##   379.22861   100
##  5863.14572   100
##    97.02004   100
##    83.95693   100
```

```
all(cmpQueries(answer2,q2dplyr),
cmpQueries(answer2,q2R),
cmpQueries(answer2,q2dt))
```

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

Zapytanie nr 3

Dla każdego roku przedstaw wpis który zebrał największą liczbę polubień w danym roku.

```
SELECT
    Posts.Title,
    UpVotesPerYear.Year,
    MAX(UpVotesPerYear.Count) AS Count
FROM (
        SELECT
            PostId,
            COUNT(*) AS Count,
            STRFTIME('%Y', Votes.CreationDate) AS Year
        FROM Votes
        WHERE VoteTypeId=2
        GROUP BY PostId, Year
      ) AS UpVotesPerYear
JOIN Posts ON Posts.Id=UpVotesPerYear.PostId
WHERE Posts.PostTypeId=1
GROUP BY Year
```

sqldf

```
sqldf3 <- function(){
answer3 <- sqldf::sqldf("SELECT
Posts.Title,
UpVotesPerYear.Year,
MAX(UpVotesPerYear.Count) AS Count
FROM (
SELECT
PostId,
COUNT(*) AS Count,
STRFTIME('%Y', Votes.CreationDate) AS Year
FROM Votes
WHERE VoteTypeId=2
GROUP BY PostId, Year
) AS UpVotesPerYear
JOIN Posts ON Posts.Id=UpVotesPerYear.PostId
WHERE Posts.PostTypeId=1
GROUP BY Year")
answer3}
answer3 <- sqldf3()
answer3[,c("Title")]
```

```
## [1] "OK we're all adults here, so really, how on earth should I use a squat toilet?"
## [2] "How to successfully haggle / bargain in markets"
## [3] "Why are airline passengers asked to lift up window shades during takeoff and landing?"
## [4] "How do you know if Americans genuinely/literally mean what they say?"
## [5] "Immigration officer that stopped me at the airport is texting me. What do I do?"
## [6] "I don't know my nationality. How can I visit Denmark?"
## [7] "Why prohibit engine braking?"
```

R

```
R3 <- function(){
UpVotes <- Votes[Votes$VoteTypeId==2,]
UpVotesYear <- cbind(UpVotes,Year=format(as.Date(UpVotes$CreationDate),"%Y"))
UpVotesPerYear <- by(UpVotesYear,list(UpVotesYear$PostId,UpVotesYear$Year),
                     function(x){
                       c(PostId = x[1,"PostId"],
                         Count= nrow(x),
                         Year = x[1,"Year"]
                         )},
                     simplify=FALSE)

UpVotesPerYearDF <- data.frame(do.call(rbind,UpVotesPerYear))
Questions <- Posts[Posts$PostTypeId==1,]
q3merge <- merge(Questions,UpVotesPerYearDF,by.x="Id",by.y="PostId")
q3split <- split(q3merge,q3merge$Year)
q3RAll <- do.call(rbind,lapply(q3split, function(x) {return(x[which.max(x$Count),])}))
q3R <- q3RAll[,c("Title","Year","Count")]
q3R$Count <- as.integer(q3R$Count)
q3R$Year <- as.character(q3R$Year)
q3R}
q3R <- R3()
```

dplyr

```
dplyr3 <- function(){
  q3dplyr <- Votes %>% filter(VoteTypeId==2) %>%
  mutate(Year = format(as.Date(.$CreationDate),"%Y")) %>% group_by(PostId,Year) %>%
           summarise(Count = n()) %>% inner_join(Posts,by=c("PostId"="Id")) %>%
           filter(PostTypeId==1) %>% group_by(Year) %>% slice(which.max(Count)) %>%
  select(Title,Year,Count)
  q3dplyr}
q3dplyr <- dplyr3()
```

Wyniki nr 3

```
microbenchmark::microbenchmark(
  sqldf = sqldf3(),
  base = R3(),
  dplyr = dplyr3()
)
```

```
## Unit: seconds
##   expr       min        lq      mean    median        uq       max neval
##  sqldf  1.182496  1.259198  1.338677  1.322160  1.398676  1.764082   100
##   base 17.298939 18.728597 19.986103 20.046666 20.996765 25.872868   100
##  dplyr  4.634595  4.914607  5.265094  5.186442  5.536618  6.745634   100
```

```
all(cmpQueries(answer3,q3dplyr),
cmpQueries(answer3,q3R))
```

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

Zapytanie nr 4

Wypisz pytania dla których różnica polubień między najlepszą odpowiedzią a tą zaakceptowaną była większa niż 50.

```
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
```

sqldf

```
sqldf4 <- function(){
answer4 <- sqldf::sqldf("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")
answer4}
answer4 <- sqldf4()
answer4[,c("Difference")]
```

```
## [1] 93 90 87 79 76 69 64 56
```

R

```
R4 <- function(){
  Answers <- Posts[Posts$PostTypeId==2,]
BestAnswersBy <- by(Answers,Answers$ParentId,function(x){
  c(Id=x$Id[which.max(x$Score)],
    ParentId = x$ParentId[which.max(x$Score)],
    MaxScore = max(x$Score)
  )})
BestAnswers <- data.frame(do.call(rbind,BestAnswersBy))
Questions<- Posts[Posts$PostTypeId==1,]
QandBestA<- merge(Questions,BestAnswers,by.x="Id",by.y="ParentId")
QBestAcc <- merge(QandBestA,Posts,by.x="AcceptedAnswerId",by.y="Id",suffixes = c("_Best", "_Acc"))
QBestAcc$Difference <- QBestAcc$MaxScore - QBestAcc$Score_Acc
q4R<-QBestAcc[QBestAcc$Difference>50,c("Id","Title_Best","MaxScore","Score_Acc","Difference")]
names(q4R)[names(q4R)=="Score_Acc"] <- "AcceptedScore"
names(q4R)[names(q4R)=="Title_Best"] <- "Title"
q4R}
q4R <- R4()
```

dplyr

```
dplyr4 <- function(){
Questions <- Posts %>% filter(PostTypeId==1) #Questions

q4dplyr <- Posts %>% filter(PostTypeId==2) %>%  group_by(ParentId) %>% slice(which.max(Score)) %>%
  ungroup %>%  rename(MaxScore=Score) %>%
  select(Id,ParentId,MaxScore) %>%  #BestAnswers
inner_join(x=Questions,by=c("Id"="ParentId")) %>% #BestAnswers and Questions
  inner_join(Posts,by=c("AcceptedAnswerId"="Id"),suffix = c("_Q", "_P")) %>% #Now with Posts
  mutate(Difference=MaxScore-Score_P) %>%
  select(Id,Title=Title_Q,MaxScore,AcceptedScore=Score_P,Difference) %>% filter(Difference>50) %>%
  arrange(desc(Difference))
  q4dplyr}

q4dplyr <- dplyr4()
```

Wyniki nr 4

```
microbenchmark::microbenchmark(
  sqldf = sqldf4(),
  base = R4(),
  dplyr = dplyr4()
)
```

```
## Unit: milliseconds
##   expr       min        lq      mean    median        uq       max neval
##  sqldf  322.8046  336.0910  353.8795  346.0442  365.9348  417.0975   100
##   base 4503.4460 4784.9926 4967.1495 4908.5271 5099.8960 6176.5570   100
##  dplyr  432.1735  468.9004  502.1448  490.8262  524.1702  714.9372   100
```

```
all(cmpQueries(answer4,q4dplyr),
cmpQueries(answer4,q4R))
```

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

Zapytanie nr 5

Jak duzo punktów można zdobyć na komentowaniu odpowiedzi pod swoim pytaniem?

```
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
```

sqldf

```
sqldf5 <- function(){
answer5 <- sqldf::sqldf("
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")
answer5}
answer5 <- sqldf5()

answer5[,c("CommentsTotalScore")]
```

```
##  [1] 75 32 26 25 25 25 24 23 20 20
```

dplyr

```
dplyr5 <- function(){
  q5dplyr <- Comments %>%  group_by(PostId,UserId) %>%
  summarise(CommentsTotalScore = sum(Score,na.rm=TRUE)) %>% ungroup %>% #CmtTotScr
  inner_join(x=(Posts %>% filter(PostTypeId==1)),by=c("Id"="PostId","OwnerUserId"="UserId")) %>%
  select(Title,CommentsTotalScore) %>% arrange(desc(CommentsTotalScore)) %>% head(10)
  q5dplyr}
q5dplyr <- dplyr5()
```

Wyniki nr 5

```
microbenchmark::microbenchmark(
  sqldf = sqldf5(),
  dplyr = dplyr5()
)
```

```
## Unit: milliseconds
##   expr      min       lq     mean   median       uq      max neval
##  sqldf 586.8292 599.3156 628.0881 611.4183 644.9414 826.8355   100
##  dplyr 235.0685 249.9699 287.4284 284.6935 303.1935 514.9078   100
```

```
all(cmpQueries(answer5,q5dplyr))
```

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

Zapytanie nr 6

Podaj dane użytkowników z rzadkimi odznakami.

```
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
```

sqldf

```
sqldf6 <- function(){
answer6 <- sqldf::sqldf("
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")
answer6}

answer6 <- sqldf6()
```

R

```
R6 <- function(){
BadgesClass <- Badges[Badges$Class==1,]
BadgesNameCounts <- aggregate(BadgesClass$Name,list(Name=BadgesClass$Name),length)
BadgesNames <- BadgesNameCounts[BadgesNameCounts$x>1 & BadgesNameCounts$x<11,"Name"]
ValuableBadges <- Badges[Badges$Class==1 & (Badges$Name %in% BadgesNames),c("Name","UserId")]
q6merge <- merge(Users,ValuableBadges,by.x="Id",by.y="UserId")
q6R <- unique(q6merge[,c("Id","DisplayName","Reputation","Age","Location")])
q6R}
q6R <- R6()
```

dplyr

```
dplyr6 <- function(){
  elite <- Badges %>% filter(Class==1) %>% group_by(Name) %>% summarise(count=n()) %>%
  filter(count>=2 & count<=10) %>%  select(Name)

ValuableBadges <- Badges %>% filter(Name %in%  elite$Name,Class==1) %>% select(Name,UserId)

q6dplyr <- ValuableBadges %>% inner_join(x=Users,c("Id"="UserId")) %>%
  select(Id,DisplayName,Reputation,Age,Location) %>% distinct()
q6dplyr}
q6dplyr <- dplyr6()
```

Wyniki nr 6

```
microbenchmark::microbenchmark(
  sqldf = sqldf6(),
  base = R6(),
  dplyr = dplyr6()
)
```

```
## Unit: milliseconds
##   expr       min         lq      mean    median        uq       max neval
##  sqldf 267.85568 274.047678 283.92784 279.83746 284.82924 399.04018   100
##   base   9.13956   9.697783  11.22311  10.17512  10.86067  50.22047   100
##  dplyr  12.41467  13.887118  16.68516  14.78712  16.19223  68.67381   100
```

```
all(cmpQueries(answer6,q6dplyr),
cmpQueries(answer6,q6R))
```

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

Zapytanie nr 7

Pokaż 10 najlepszych pytań sprzed 2016 roku.

```
"
SELECT
    Posts.Title,
    VotesByAge2.OldVotes
FROM Posts
JOIN (
      SELECT
          PostId,
          MAX(CASE WHEN VoteDate = 'new' THEN Total ELSE 0 END) NewVotes,
          MAX(CASE WHEN VoteDate = 'old' THEN Total ELSE 0 END) OldVotes,
          SUM(Total) AS Votes
      FROM (
          SELECT
              PostId,
              CASE STRFTIME('%Y', CreationDate)
                  WHEN '2017' THEN 'new'
                  WHEN '2016' THEN 'new'
                  ELSE 'old'
                  END VoteDate,
              COUNT(*) AS Total
          FROM Votes
          WHERE VoteTypeId=2
          GROUP BY PostId, VoteDate
      ) AS VotesByAge
      GROUP BY VotesByAge.PostId
      HAVING NewVotes=0
) AS VotesByAge2 ON VotesByAge2.PostId=Posts.Id
WHERE Posts.PostTypeId=1
ORDER BY VotesByAge2.OldVotes DESC
LIMIT 10"
```

sqldf

```
sqldf7 <- function(){
answer7 <- sqldf::sqldf("
SELECT
Posts.Title,
VotesByAge2.OldVotes
FROM Posts
JOIN (
SELECT
PostId,
MAX(CASE WHEN VoteDate = 'new' THEN Total ELSE 0 END) NewVotes,
MAX(CASE WHEN VoteDate = 'old' THEN Total ELSE 0 END) OldVotes,
SUM(Total) AS Votes
FROM (
SELECT
PostId,
CASE STRFTIME('%Y', CreationDate)
WHEN '2017' THEN 'new'
WHEN '2016' THEN 'new'
ELSE 'old'
END VoteDate,
COUNT(*) AS Total
FROM Votes
WHERE VoteTypeId=2
GROUP BY PostId, VoteDate
) AS VotesByAge
GROUP BY VotesByAge.PostId
HAVING NewVotes=0
) AS VotesByAge2 ON VotesByAge2.PostId=Posts.Id
WHERE Posts.PostTypeId=1
ORDER BY VotesByAge2.OldVotes DESC
LIMIT 10")
answer7}
answer7 <- sqldf7()
answer7[,c("Title")]
```

```
##  [1] "Which European cities have bike rental stations for tourists?"
##  [2] "Why do hostels require you to 'rent' bedding?"
##  [3] "What to do with your valuables on a low-cost holiday while swimming/diving in Central America?"
##  [4] "Can't check-in to a hotel because I am 18"
##  [5] "What are some good ways to find things to explore on-site in an unfamiliar place?"
##  [6] "Alarm Clock without Noise? To wake up in common sleeping rooms and airports without noise?"
##  [7] "What times of the year are best for visiting France?"
##  [8] "What is the business model of commercial free walking tours?"
##  [9] "Getting work on a cruise ship in order to travel"
## [10] "Carrying medicines internationally for a friend"
```

R

```
R7 <- function(){
UpVotes <- Votes[Votes$VoteTypeId==2,]
UpVotes$VoteYear <- format(as.Date(UpVotes$CreationDate),"%Y")
UpVotes$VoteDate <- ifelse(UpVotes$VoteYear=="2017" | UpVotes$VoteYear=="2016","new","old")
VotesByAge <- aggregate(list(Total=UpVotes$Id),list(PostId=UpVotes$PostId,
                                          VoteDate=UpVotes$VoteDate),length)

VotesByAgeBy <- by(VotesByAge,VotesByAge$PostId,function(x){
  c(PostId = x$PostId[1],
    NewVotes = max(ifelse(x$VoteDate=="new",x[x$VoteDate=="new","Total"],0)),
    OldVotes = max(ifelse(x$VoteDate=="old",x[x$VoteDate=="old","Total"],0)),
    Total = sum(x$Total,na.rm=TRUE))
})

VotesByAgeON <- data.frame(do.call(rbind,VotesByAgeBy))
VotesByAge2 <- VotesByAgeON[VotesByAgeON$NewVotes==0,]

Questions <- Posts[Posts$PostTypeId==1,]
q7merge <- merge(Questions,VotesByAge2,by.x="Id",by.y="PostId")
q7R <- head(q7merge[order(q7merge$OldVotes,decreasing = TRUE),c("Title","OldVotes")],10)
q7R}
q7R <- R7()
```

dplyr

```
dplyr7 <- function(){
  VotesByAge <- Votes %>% filter(VoteTypeId==2) %>%
  mutate(Year = as.numeric(format(as.Date(CreationDate),"%Y")),
         VoteDate= if_else(Year == 2017 | Year == 2016,"new","old")) %>%
  group_by(PostId,VoteDate) %>% summarise(Total=n())


VotesByAge2 <- VotesByAge  %>% group_by(PostId) %>%
  summarize(NewVotes = max(if_else(VoteDate=="new",Total,as.integer(0))),
            OldVotes = max(if_else(VoteDate=="old",Total,as.integer(0))),
            Votes = sum(Total,na.rm = TRUE)) %>%
  filter(NewVotes==0)

q7dplyr <- VotesByAge2 %>% inner_join(Posts,by=c("PostId"="Id")) %>% filter(PostTypeId==1) %>%
  arrange(desc(OldVotes)) %>%  select(Title,OldVotes) %>% head(10)
q7dplyr}
q7dplyr <- dplyr7()
```

Wyniki nr 7

```
microbenchmark::microbenchmark(
  sqldf = sqldf7(),
  base = R7(),
  dplyr = dplyr7()
)
```

```
## Unit: seconds
##   expr       min        lq      mean    median        uq       max neval
##  sqldf  1.140336  1.170589  1.302958  1.232485  1.383656  1.914357   100
##   base 13.874406 14.283487 16.145665 16.124923 17.521660 22.085221   100
##  dplyr  8.565294  8.886964 10.007626  9.904778 10.696204 13.715756   100
```

```
all(cmpQueries(answer7,q7dplyr),
cmpQueries(answer7,q7R))
```

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