Lesson 5 Solutions

Stefanie Molin

April 18, 2017

Let's do some practice problems to challenge your understanding.

1. Query Vertica for spend by day for the last 5 days for 3 clients of your choice. Reshape the long dataframe into a wide one with each client as a column.

```
library(reshape2)
```

```
# QueryVertica has been loaded along with username/password
query <- "
SELECT
    day
    , client_name
    , SUM(revenue) AS spend
FROM
    (SELECT
        client_name
        , client_id
    FROM
       [redacted]
    WHERE
        client_name IN ([redacted])
    GROUP BY
        client_name
        , client_id) cl
JOIN
  [redacted]
ON
    cl.client_id = stats.client_id
WHERE
    day >= CURRENT_DATE() - 5
GROUP BY
    day
    , client_name
"

df <- QueryVertica(username, query, password)

# inspect data
head(df)
```

```
##          day client_name    spend
## 1 2017-04-14  [redacted]
## 2 2017-04-14
## 3 2017-04-13
## 4 2017-04-16
## 5 2017-04-17
## 6 2017-04-16
```

```r
# reshape the data
(reshaped_df <- dcast(melt(df), day ~ client_name + variable))
```

```
## Using day, client_name as id variables

##          day [redacted]_spend [redacted]_spend [redacted]_spend
## 1 2017-04-13 [redacted]
## 2 2017-04-14
## 3 2017-04-15
## 4 2017-04-16
## 5 2017-04-17
```

2. Read in a 2M row excerpt of the [redacted] catalog from the provided textfile using the `fread()` function from `data.table` (this is faster than base R and automatically detects options). The file will be read into a `data.table`. (a) Drop the `sqlid` column. (b) Rename the `id` column `external_id`. (c) Make the `name` and `external_id` columns keys. (d) Select the `name` and `external_id` of the most expensive item and least expensive item. Limit the name of the selection to 35 characters.

```r
library(data.table)
library(stringr)
```

```r
# read in catalog using data.table's fread()
catalog <- fread("[redacted]_catalog.txt")
```

```
##
Read 0.0% of 2000000 rows
Read 4.5% of 2000000 rows
Read 10.0% of 2000000 rows
Read 12.0% of 2000000 rows
Read 15.5% of 2000000 rows
Read 19.5% of 2000000 rows
Read 22.0% of 2000000 rows
Read 24.0% of 2000000 rows
Read 29.0% of 2000000 rows
Read 36.0% of 2000000 rows
Read 43.0% of 2000000 rows
Read 47.0% of 2000000 rows
Read 61.0% of 2000000 rows
Read 73.0% of 2000000 rows
Read 81.5% of 2000000 rows
Read 84.0% of 2000000 rows
Read 2000000 rows and 5 (of 5) columns from 0.886 GB file in 00:00:28
```

```r
# drop column
catalog <- catalog[, sqlid := NULL]

# update id column name
setnames(catalog, "id", "external_id")

# make keys
setkey(catalog, name, external_id)

# select the most and least expensive items
solution <- catalog[c(which.max(price), which.min(price)), price,
                    by = .(name, external_id)]
solution[, c("max_or_min", "name") := .(c("Max", "Min"), str_sub(name, 1, 35))]
```

```
##                                        name    external_id    price max_or_min
## 1: [redacted]                                                99999.00        Max
## 2:                                                               0.02        Min
```

```
solution
```

```
##                                        name    external_id    price max_or_min
## 1: [redacted]                                                99999.00        Max
## 2:                                                               0.02        Min
```

3. Using the [redacted] catalog you obtained in `(2)`, (a) find number of products with extra data containing the word "promo". (b) Find the unique promo offers and display a few of them. You will need to use a regular expression to find the value in the `extra` field, then you will need to use `str_match()` to find that pattern, and use a function from the `apply` family to get the results of applying that function on all values of `extra`.

   Hint: If you are having trouble with the regex, you can take a few entries of the `extra` column in the data.table and work on adapting a regex here: http://regexr.com/. Be sure to look at how `str_match()` works and pick an appropriate `apply` family member; depending on how you do this, you may need to change the type of the object you give the function from the `apply` family.

```r
library(stringr)
```

```r
# check if each item has "promo" in extra field
on_promo <- lapply(catalog[, extra], str_detect, pattern = "promo")

# calculate number of promo products
sum(unlist(on_promo))
```

```
## [1] 536456
```

```r
# find all promo offers (this is different than just looking for the word promo)
promos <- apply(as.matrix(catalog[, extra]), 1, FUN = str_match,
                pattern = "(?:promo@V)([^@]+)")[2,]

# remove the NA's from promos
promos <- na.omit(promos)

# find all unique promos
unique_promos <- unique(promos)
length(unique_promos)
```

```
## [1] 238
```

```r
# display a few promos
head(unique_promos)
```

```
## [1] "FREE SHIPPING on qualified orders $59+"
## [2] "20% to 30% off select TVs"
## [3] "20% off air purifiers"
## [4] "40% off Avalon Bay Portable Ice Maker"
## [5] "Up to 25% OFF Kenmore Appliances"
## [6] "50% off or more on select Power Tools"
```