SQL for DataScience

Pankaj Shah

6/1/2019

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

SQL for DS

```
# Lets read the csv file from the source
econ_df <- readr::read_csv('https://raw.githubusercontent.com/shahnp/Teaching_course_materials/master/w
con <- DBI::dbConnect(RSQLite::SQLite(), ":memory:")
copy_to(con, econ_df)
```

Head

```
# Always better to look the data even before we do anything on it.
head(econ_df)
```

```
## # A tibble: 6 x 11
##      id referrer device bouncers n_visit n_pages duration country purchase
##   <dbl> <chr>    <chr>  <lgl>      <dbl>   <dbl>    <dbl> <chr>   <lgl>
## 1     1 google   laptop TRUE          10       1      693 Czech ~ FALSE
## 2     2 yahoo    tablet TRUE           9       1      459 Yemen   FALSE
## 3     3 direct   laptop TRUE           0       1      996 Brazil  FALSE
## 4     4 bing     tablet FALSE          3      18      468 China   TRUE
## 5     5 yahoo    mobile TRUE           9       1      955 Poland  FALSE
## 6     6 yahoo    laptop FALSE          5       5      135 South ~ FALSE
## # ... with 2 more variables: order_items <dbl>, order_value <dbl>
```

```
library(DataExplorer)
```

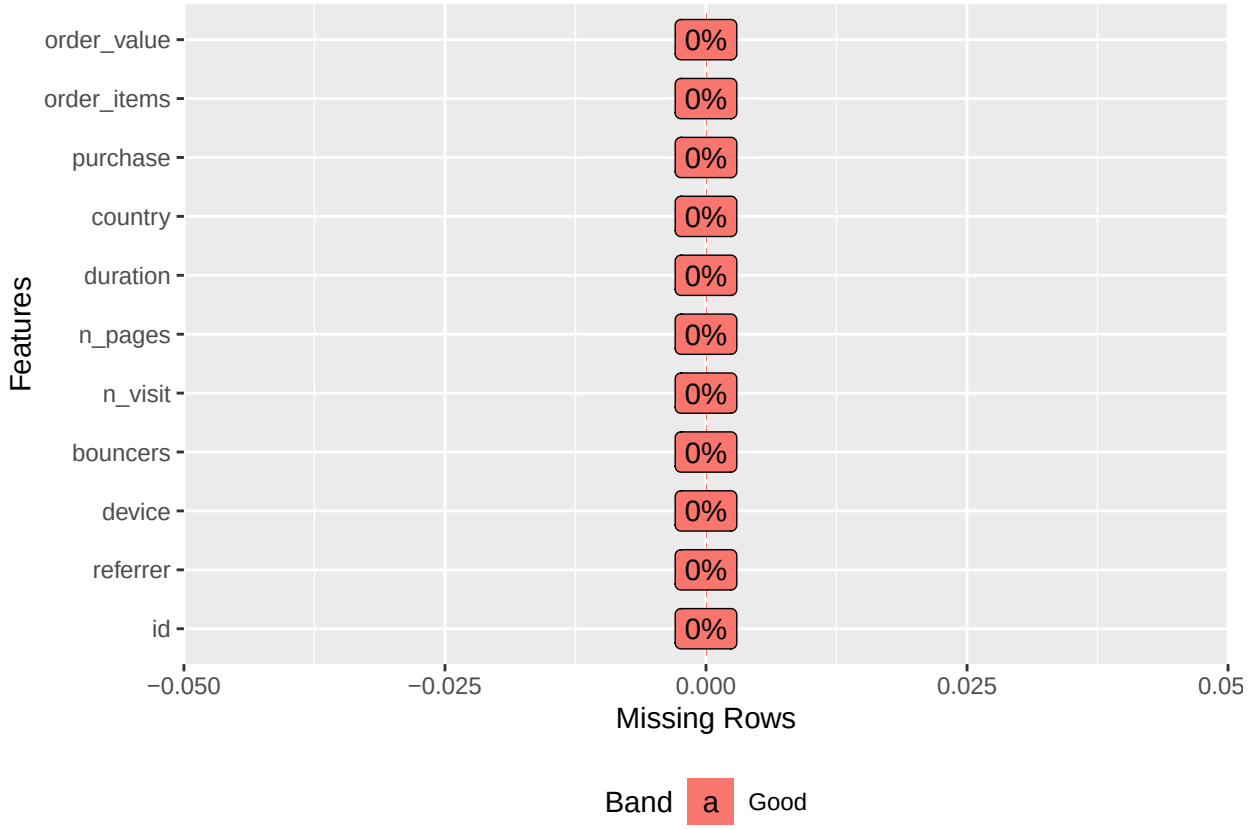
```
## Warning: package 'DataExplorer' was built under R version 3.5.2
```

```
df_info <- function(x) {
  data  <- as.character(substitute(x))  # data frame name
  size <- format(object.size(x), units="Mb")  # size (Mb)
  plot_missing(data.frame(x)) # Vizualization of Missing Data.

  ##column information
  column.info <- data.frame( column        = names(sapply(x, class)),
                             class         = sapply(x, class),
                             unique.values = sapply(x, function(y) length(unique(y))),
                             missing.count = colSums(is.na(x)),
                             missing.pct   = round(colSums(is.na(x)) / nrow(x) * 100, 2)) %>% arrange(d

  row.names(column.info) <- 1:nrow(column.info)
```

```
  list(data.frame     = data.frame(name=data, size=size),
       dimensions     = data.frame(rows=nrow(x), columns=ncol(x)),
       column.details = column.info)
}
df_info(econ_df) # Info about datasets.
```




```
## $data.frame
##      name   size
## 1 econ_df 0.1 Mb
##
## $dimensions
##   rows columns
## 1 1000      11
##
## $column.details
##          column     class unique.values missing.count missing.pct
## 1            id   numeric          1000             0           0
## 2      duration   numeric           507             0           0
## 3   order_value   numeric           239             0           0
## 4       country character           123             0           0
## 5       n_pages   numeric            20             0           0
## 6       n_visit   numeric            11             0           0
## 7   order_items   numeric            11             0           0
## 8      referrer character             5             0           0
## 9        device character             3             0           0
## 10     bouncers   logical             2             0           0
## 11     purchase   logical             2             0           0
```

```
library(kableExtra)
t(apply(econ_df, MARGIN = 2, function(x) range(x, na.rm=TRUE))) %>% kable()
```

id	1	1000
referrer	bing	yahoo
device	laptop	tablet
bouncers	TRUE	FALSE
n_visit	0	10
n_pages	1	20
duration	10	999
country	Afghanistan	Yemen
purchase	TRUE	FALSE
order_items	0	10
order_value	0	2992

SQL SELECT

```
# Lets get data about all the device used for purchase
library(DBI)
dbGetQuery(con, "SELECT device FROM econ_df") %>% table()
```

```
## .
## laptop mobile tablet
##    325    344    331
```

3 way table

```
# Lets group them looking at the purchase
dbGetQuery(con, "SELECT referrer, device, purchase  FROM econ_df") %>% table()
```

```
## , , purchase = 0
##
##         device
## referrer laptop mobile tablet
##   bing       54     61     62
##   direct     60     60     46
##   google     57     69     63
##   social     66     49     65
##   yahoo      57     69     59
##
## , , purchase = 1
##
##         device
## referrer laptop mobile tablet
##   bing        5      5      7
##   direct     10      4     11
##   google      6      9      4
##   social      4      8      8
##   yahoo       6     10      6
```

*

```
# Lets select everything from the datasets and se what other things we could look at.
dbGetQuery(con, "SELECT * FROM econ_df LIMIT 5")
```

```
##   id referrer device bouncers n_visit n_pages duration        country
## 1  1   google laptop        1      10       1      693 Czech Republic
## 2  2    yahoo tablet        1       9       1      459          Yemen
## 3  3   direct laptop        1       0       1      996         Brazil
## 4  4     bing tablet        0       3      18      468          China
## 5  5    yahoo mobile        1       9       1      955         Poland
##   purchase order_items order_value
## 1        0           0           0
## 2        0           0           0
## 3        0           0           0
## 4        1           6         434
## 5        0           0           0
```

* LIMIT

```
# If the page is too long we can limit that to 10.
dbGetQuery(con, "SELECT * FROM econ_df limit 10") # LIMIT and limit is same not case sensitive but rese
```

```
##    id referrer device bouncers n_visit n_pages duration        country
## 1   1   google laptop        1      10       1      693 Czech Republic
## 2   2    yahoo tablet        1       9       1      459          Yemen
## 3   3   direct laptop        1       0       1      996         Brazil
## 4   4     bing tablet        0       3      18      468          China
## 5   5    yahoo mobile        1       9       1      955         Poland
## 6   6    yahoo laptop        0       5       5      135   South Africa
## 7   7    yahoo mobile        1      10       1       75     Bangladesh
## 8   8   direct mobile        1      10       1      908      Indonesia
## 9   9     bing mobile        0       3      19      209    Netherlands
## 10 10   google mobile        1       6       1      208 Czech Republic
##    purchase order_items order_value
## 1         0           0           0
## 2         0           0           0
## 3         0           0           0
## 4         1           6         434
## 5         0           0           0
## 6         0           0           0
## 7         0           0           0
## 8         0           0           0
## 9         0           0           0
## 10        0           0           0
```

DISTINCT

```
# Lets see the referrer datasets as we have choosen distint it will all return value to 1.
dbGetQuery(con, "SELECT distinct referrer FROM econ_df") %>% table()
```

```
## .
##   bing direct google social  yahoo
##      1      1      1      1      1
```

```
# once we remove distinct we will get a table
dbGetQuery(con, "SELECT referrer FROM econ_df") %>% table()
```

```
## .
##   bing direct google social  yahoo
##    194    191    208    200    207
```

SLICE

> (Greater than)

```
# We can slice and dice based on Duration look at 468 apperance
dbGetQuery(con, "SELECT *
                 FROM econ_df
                 WHERE duration >= 468
                LIMIT 5")
```

```
##   id referrer device bouncers n_visit n_pages duration        country
## 1  1   google laptop        1      10       1      693 Czech Republic
## 2  3   direct laptop        1       0       1      996         Brazil
## 3  4     bing tablet        0       3      18      468          China
## 4  5    yahoo mobile        1       9       1      955         Poland
## 5  8   direct mobile        1      10       1      908      Indonesia
##   purchase order_items order_value
## 1        0           0           0
## 2        0           0           0
## 3        1           6         434
## 4        0           0           0
## 5        0           0           0
```

== (Equal to)

```
# Other way of doing slicing and dicing would be to used == sign

dbGetQuery(con, "SELECT *
                 FROM econ_df
                 WHERE device == 'mobile'
                 LIMIT 5")
```

```
##   id referrer device bouncers n_visit n_pages duration        country
## 1  5    yahoo mobile        1       9       1      955         Poland
## 2  7    yahoo mobile        1      10       1       75     Bangladesh
## 3  8   direct mobile        1      10       1      908      Indonesia
## 4  9     bing mobile        0       3      19      209    Netherlands
## 5 10   google mobile        1       6       1      208 Czech Republic
##   purchase order_items order_value
```

```
## 1        0           0           0
## 2        0           0           0
## 3        0           0           0
## 4        0           0           0
## 5        0           0           0
```

AND OR

```
#  OR | AND ==  >

dbGetQuery(con, "SELECT *
                 FROM econ_df
                 WHERE n_visit > 3 AND duration > 100 AND country == 'Ireland' OR country == 'France' LI
```

```
##   id referrer device bouncers n_visit n_pages duration country purchase
## 1 13   direct mobile        0       9      14      406 Ireland        1
## 2 15    yahoo mobile        0       7       1       19  France        0
## 3 49   social mobile        0       1       2       44  France        0
## 4 67    yahoo tablet        1       3       1      332  France        0
## 5 68   direct mobile        1       7       1      912  France        0
##   order_items order_value
## 1           3         651
## 2           7        2423
## 3          10        1515
## 4           0           0
## 5           0           0
```

BETWEEN OR

```
# Combination of AND & OR
dbGetQuery(con, "SELECT *
                 FROM econ_df
                 WHERE (n_visit == 5 OR n_visit == 3)
                 AND (device = 'Mobile' OR device = 'tablet') LIMIT 5") # case sensitive 'Mobile'
```

```
##   id referrer device bouncers n_visit n_pages duration     country
## 1  4     bing tablet        0       3      18      468       China
## 2 14    yahoo tablet        0       5       8       80 Philippines
## 3 17     bing tablet        0       5      16      368        Peru
## 4 50     bing tablet        1       5       1      831        Iran
## 5 53   social tablet        0       3      12      324       China
##   purchase order_items order_value
## 1        1           6         434
## 2        0           2         362
## 3        1           6        1049
## 4        0           0           0
## 5        0           0           0
```

```
dbGetQuery(con, "SELECT *
                 FROM econ_df
                 WHERE (n_visit == 5 OR n_visit == 3)
                 AND (device = 'mobile' OR device = 'tablet') LIMIT 5") # mobile -> double check with t
```

```
##   id referrer device bouncers n_visit n_pages duration     country
## 1  4     bing tablet        0       3      18      468       China
## 2  9     bing mobile        0       3      19      209 Netherlands
## 3 14    yahoo tablet        0       5       8       80 Philippines
## 4 17     bing tablet        0       5      16      368        Peru
## 5 22   google mobile        1       5       1      147      Brazil
##   purchase order_items order_value
## 1        1           6         434
## 2        0           0           0
## 3        0           2         362
## 4        1           6        1049
## 5        0           0           0
```

```
# Between Two Numbers
dbGetQuery(con, "SELECT *
                 FROM econ_df
                 WHERE n_visit BETWEEN 1 AND 3 AND device = 'mobile' LIMIT 5")
```

```
##   id referrer device bouncers n_visit n_pages duration     country
## 1  9     bing mobile        0       3      19      209 Netherlands
## 2 32   direct mobile        1       2       1      501 El Salvador
## 3 36     bing mobile        0       1       1       25     Ireland
## 4 38    yahoo mobile        1       3       1      700      Canada
## 5 42   direct mobile        0       1      13      234   Indonesia
##   purchase order_items order_value
## 1        0           0           0
## 2        0           0           0
## 3        0          10        1885
## 4        0           0           0
## 5        0           0           0
```

WHERE IN

```
# Choice of the number, character etc.

dbGetQuery(con, "SELECT *
                 FROM econ_df
                 WHERE n_visit IN (2, 4, 6, 8, 10) LIMIT 5")
```

```
##   id referrer device bouncers n_visit n_pages duration        country
## 1  1   google laptop        1      10       1      693 Czech Republic
## 2  7    yahoo mobile        1      10       1       75     Bangladesh
## 3  8   direct mobile        1      10       1      908      Indonesia
## 4 10   google mobile        1       6       1      208 Czech Republic
## 5 12   direct tablet        0       6      12      132        Estonia
##   purchase order_items order_value
## 1        0           0           0
## 2        0           0           0
## 3        0           0           0
## 4        0           0           0
## 5        0           0           0
```

```
dbGetQuery(con, "SELECT *
```

```
                FROM econ_df
                WHERE n_visit IN (2,4,6,8,10) AND duration > 300 AND
                      country IN ('China', 'Japan', 'Colombia') LIMIT 5")
```

```
##   id referrer device bouncers n_visit n_pages duration  country purchase
## 1 21   direct laptop        1       2       1      384    China        0
## 2 27   direct tablet        0       2      19      342    Japan        1
## 3 31   social tablet        1       2       1      795    Japan        0
## 4 33   direct laptop        1       8       1      658 Colombia        0
## 5 73   google tablet        1       4       1      565    China        0
##   order_items order_value
## 1           0           0
## 2           5         622
## 3           0           0
## 4           0           0
## 5           0           0
```

NULL

```
# No Null value  as we have seen in the beginning.
dbGetQuery(con, "SELECT *
                FROM econ_df
                WHERE device IS NULL") # Zero
```

```
##  [1] id          referrer    device      bouncers    n_visit
##  [6] n_pages     duration    country     purchase    order_items
## [11] order_value
## <0 rows> (or 0-length row.names)
```

LIKE (%)

```
# % represents rest part of the word.
dbGetQuery(con, "SELECT *
                FROM econ_df
                WHERE country LIKE 'P%' LIMIT 5") # Starting with P
```

```
##   id referrer device bouncers n_visit n_pages duration     country
## 1  5    yahoo mobile        1       9       1      955      Poland
## 2 14    yahoo tablet        0       5       8       80 Philippines
## 3 17     bing tablet        0       5      16      368        Peru
## 4 43     bing laptop        1       0       1      456    Portugal
## 5 59    yahoo tablet        1       9       1      706 Philippines
##   purchase order_items order_value
## 1        0           0           0
## 2        0           2         362
## 3        1           6        1049
## 4        0           0           0
## 5        0           0           0
```

```
dbGetQuery(con, "SELECT *
           FROM econ_df
           WHERE country LIKE '%A'  LIMIT 5") # Ending in A not case sensitive.
```

```
##   id referrer device bouncers n_visit n_pages duration      country
## 1  4     bing tablet        0       3      18      468        China
## 2  6    yahoo laptop        0       5       5      135 South Africa
## 3  8   direct mobile        1      10       1      908    Indonesia
## 4 11   direct laptop        1       9       1      738      Jamaica
## 5 12   direct tablet        0       6      12      132      Estonia
##   purchase order_items order_value
## 1        1           6         434
## 2        0           0           0
## 3        0           0           0
## 4        0           0           0
## 5        0           0           0
```

SUM

```
# 1.SUM
dbGetQuery(con, "SELECT SUM(n_visit) FROM econ_df")
```

```
##   SUM(n_visit)
## 1         4972
```

```
# 2.SUM WHERE ==
dbGetQuery(con, "SELECT SUM(n_visit)
                FROM econ_df
                WHERE referrer == 'direct'")
```

```
##   SUM(n_visit)
## 1          936
```

```
## 3.SUM  WHERE IN
dbGetQuery(con, "SELECT SUM(order_items)
                 FROM econ_df
                 WHERE device IN ('tablet', 'laptop')")
```

```
##   SUM(order_items)
## 1              880
```

```
## 4.SUM Count GROUP ORDER
dbGetQuery(con, "SELECT device,SUM(order_items),
                 Count(*) AS individual_device_group_by
                 FROM econ_df
                 GROUP BY device
                 ORDER by individual_device_group_by DESC")
```

```
##   device SUM(order_items) individual_device_group_by
## 1 mobile              501                        344
## 2 tablet              431                        331
## 3 laptop              449                        325
```

```
## 5. Count GROUP ORDER : No SUM
dbGetQuery(con, "SELECT device,
```

```
                count(*) AS visits_device_group_by
                FROM econ_df
                GROUP BY device
                ORDER by visits_device_group_by DESC")
```

```
##   device visits_device_group_by
## 1 mobile                    344
## 2 tablet                    331
## 3 laptop                    325
```

```
## 6. >
dbGetQuery(con, "SELECT SUM(n_visit)
                FROM econ_df
                WHERE n_visit > 5")
```

```
##   SUM(n_visit)
## 1         3574
```

AVERAGE

```
# AVERAGE
dbGetQuery(con, "SELECT AVG(n_visit) FROM econ_df")
```

```
##   AVG(n_visit)
## 1        4.972
```

```
# WHERE  LIKE
dbGetQuery(con, "SELECT AVG(n_visit)
                FROM econ_df
                WHERE country LIKE 'P%'")
```

```
##   AVG(n_visit)
## 1     5.079137
```

```
# WHERE ==
dbGetQuery(con, "SELECT AVG(n_visit) AS avg_of_all_the_mobile
                FROM econ_df
                WHERE device == 'mobile'")
```

```
##   avg_of_all_the_mobile
## 1              5.479651
```

MAX MIN

```
# MAX
dbGetQuery(con, "SELECT MAX(n_visit) FROM econ_df")
```

```
##   MAX(n_visit)
## 1           10
```

```
# MAX from single column
dbGetQuery(con, "SELECT MAX(n_visit)
```

```
                FROM econ_df
                WHERE device == 'tablet'")
```

```
##   MAX(n_visit)
## 1           10
```

```
# Define the column name: AS
dbGetQuery(con, "SELECT MAX(n_visit) AS max_visit
                FROM econ_df")
```

```
##   max_visit
## 1        10
```

```
# MAX GROUP_BY ORDER_BY
dbGetQuery(con, "SELECT device, MAX(duration) AS max_duration_of_all_device
                FROM econ_df
                GROUP BY device
                ORDER by max_duration_of_all_device DESC")
```

```
##   device max_duration_of_all_device
## 1 tablet                        999
## 2 laptop                        997
## 3 mobile                        994
```

```
# MIN
dbGetQuery(con, "SELECT MIN(n_visit) FROM econ_df")
```

```
##   MIN(n_visit)
## 1            0
```

```
# MIN WHERE
dbGetQuery(con, "SELECT MIN(n_visit)
                FROM econ_df
                WHERE duration BETWEEN 600 AND 900")
```

```
##   MIN(n_visit)
## 1            0
```

```
# MIN AS
dbGetQuery(con, "SELECT MIN(duration) AS min_duration_of_all_time
                FROM econ_df")
```

```
##   min_duration_of_all_time
## 1                       10
```

ORDER BY

```
# ORDER alphabetically countrywise
dbGetQuery(con, "SELECT *
                FROM econ_df
                ORDER BY country LIMIT 5")
```

```
##    id referrer device bouncers n_visit n_pages duration     country
## 1 232   social laptop        0       8       2       60 Afghanistan
## 2 299    yahoo laptop        0      10      18      180 Afghanistan
## 3 570   social laptop        1       2       1      274 Afghanistan
```

```
## 4 677   direct tablet        1      10       1      682 Afghanistan
## 5 682   direct tablet        0       5      18      414 Afghanistan
##   purchase order_items order_value
## 1        0           0           0
## 2        0           0           0
## 3        0           0           0
## 4        0           0           0
## 5        1           8        2006
```

```
# DESC ORDER by n_visit
dbGetQuery(con, "SELECT *
                 FROM econ_df
                 ORDER BY duration DESC LIMIT 5")
```

```
##    id referrer device bouncers n_visit n_pages duration       country
## 1 854     bing tablet        1       5       1      999        France
## 2 824    yahoo laptop        1       2       1      997       Somalia
## 3   3   direct laptop        1       0       1      996        Brazil
## 4  16     bing laptop        1       1       1      995 United States
## 5 267    yahoo laptop        1       5       1      994        Brazil
##   purchase order_items order_value
## 1        0           0           0
## 2        0           0           0
## 3        0           0           0
## 4        0           0           0
## 5        0           0           0
```

Facebook posts by Members of the U.S. Congress in 2017

SQLite database

```
# Library
library(DBI)
library("odbc")
```

```
# create database: this will create a file in our hard drive

db <- dbConnect(RSQLite::SQLite(), "facebook-db.sqlite")

# reading the first file
congress <- read.csv("https://raw.githubusercontent.com/shahnp/data/master/congress-facebook-2017.csv",s

# dbWriteTable : add dataframe to our database adding first table:
dbWriteTable(db, "congress", congress, overwrite = TRUE) # once it is written it is already there.

# testing that it works:
dbListFields(db, "congress")
```

```
## [1] "bioid"       "screen_name" "name"        "gender"      "type"
## [6] "party"
```

```
dbGetQuery(db, 'SELECT * FROM congress LIMIT 5')
```

```
##     bioid           screen_name               name gender type      party
## 1 A000055        RobertAderholt Robert B. Aderholt      M  rep Republican
```

```
## 2 A000360 senatorlamaralexander    Lamar Alexander      M  sen Republican
## 3 A000367        repjustinamash       Justin Amash      M  rep Republican
## 4 A000369         MarkAmodeiNV2     Mark E. Amodei      M  rep Republican
## 5 A000370    CongresswomanAdams      Alma S. Adams      F  rep   Democrat
```

However, the files are too big to open them all in memory. Instead, we will open them one by one, and then append them to the table.

```
fls <- list.files("~/Desktop/SQL-workshop-master/data/posts", full.names=TRUE)

for (f in fls){

  message(f)

  # read file into memory
  fb <- read.csv(f, stringsAsFactors=F)

  # adding to table in SQL database
  dbWriteTable(db, "posts", fb, append=TRUE)

}

# testing that it works
dbListFields(db, "posts")
```

```
##  [1] "screen_name"    "id"             "from_name"      "date"
##  [5] "datetime"       "message"        "type"           "link"
##  [9] "domain"         "likes_count"    "comments_count" "shares_count"
## [13] "love_count"     "haha_count"     "wow_count"      "sad_count"
## [17] "angry_count"
```

```
dbGetQuery(db, 'SELECT * FROM posts LIMIT 5')
```

```
##    screen_name                                id from_name       date
## 1 1.066316e+14 106631626049851_1273804329332569   Ted Poe 2017-01-03
## 2 1.066316e+14 106631626049851_1274017475977921   Ted Poe 2017-01-03
## 3 1.066316e+14 106631626049851_1275444719168530   Ted Poe 2017-01-05
## 4 1.066316e+14 106631626049851_1275484255831243   Ted Poe 2017-01-05
## 5 1.066316e+14 106631626049851_1276290242417311   Ted Poe 2017-01-06
##              datetime
## 1 2017-01-03 17:16:40
## 2 2017-01-03 23:11:15
## 3 2017-01-05 14:36:24
## 4 2017-01-05 15:38:22
## 5 2017-01-06 14:31:19
##
## 1
## 2
## 3
## 4
## 5 TERRORISM UPDATE:  ·ISIS fighters attacked an Iraqi army outpost and a police station near the city
##    type
## 1  link
## 2  link
## 3 photo
## 4  link
```

```
## 5  link
##
## 1                    https://www.c-span.org/video/?420804-1%2Fus-house-meets-elect-speaker-swear-mer
## 2                                  http://poe.house.gov/2017/1/congressman-poe-introduces-the-smart-
## 3 https://www.facebook.com/JudgeTedPoe/photos/a.137207746325572.19816.106631626049851/12754447191685
## 4                                  http://poe.house.gov/press-releases?ID=443ADE64-03E2-461A-BA56-FF
## 5                                                                  http://poe.house.gov/index.cfm?p=terror
##          domain likes_count comments_count shares_count love_count
## 1    c-span.org         121             15            6          5
## 2 poe.house.gov         182             16           31          8
## 3          <NA>         137             13           25          9
## 4 poe.house.gov        1125            166          170        244
## 5 poe.house.gov          24              8           12          1
##   haha_count wow_count sad_count angry_count
## 1          0         0         0           0
## 2          0         0         1           2
## 3          0         0         0           0
## 4          0        10         0           0
## 5          0         5         1           3
```

```
# what if we make a mistake and want to remove the table?
# dbRemoveTable(db, "posts")

# and we close the connection for now
dbDisconnect(db)
```

Querying an SQL database:

```
db <- dbConnect(RSQLite::SQLite(), "facebook-db.sqlite")
test <- dbGetQuery(db, 'SELECT * FROM congress LIMIT 5')
glimpse(test)
```

```
## Observations: 5
## Variables: 6
## $ bioid       <chr> "A000055", "A000360", "A000367", "A000369", "A000370"
## $ screen_name <chr> "RobertAderholt", "senatorlamaralexander", "repjus...
## $ name        <chr> "Robert B. Aderholt", "Lamar Alexander", "Justin A...
## $ gender      <chr> "M", "M", "M", "M", "F"
## $ type        <chr> "rep", "sen", "rep", "rep", "rep"
## $ party       <chr> "Republican", "Republican", "Republican", "Republi...
```

```
# test if we can extract any data.
dbGetQuery(db, "SELECT * FROM congress LIMIT 5")
```

```
##     bioid           screen_name               name gender type      party
## 1 A000055        RobertAderholt Robert B. Aderholt      M  rep Republican
## 2 A000360 senatorlamaralexander    Lamar Alexander      M  sen Republican
## 3 A000367        repjustinamash       Justin Amash      M  rep Republican
## 4 A000369         MarkAmodeiNV2     Mark E. Amodei      M  rep Republican
## 5 A000370    CongresswomanAdams      Alma S. Adams      F  rep   Democrat
```

```
# Querying :one column
dbGetQuery(db, "SELECT name FROM congress LIMIT 5") # select certain column from the database.
```

```
##                 name
## 1 Robert B. Aderholt
## 2    Lamar Alexander
## 3       Justin Amash
## 4     Mark E. Amodei
## 5      Alma S. Adams
```

```
# Select multiple columns
dbGetQuery(db, "SELECT name, party FROM congress LIMIT 5")
```

```
##                 name      party
## 1 Robert B. Aderholt Republican
## 2    Lamar Alexander Republican
## 3       Justin Amash Republican
## 4     Mark E. Amodei Republican
## 5      Alma S. Adams   Democrat
```

Lets look at the post

```
dbGetQuery(db, "SELECT * FROM posts LIMIT 5")
```

```
##    screen_name                                id from_name       date
## 1 1.066316e+14 106631626049851_1273804329332569   Ted Poe 2017-01-03
## 2 1.066316e+14 106631626049851_1274017475977921   Ted Poe 2017-01-03
## 3 1.066316e+14 106631626049851_1275444719168530   Ted Poe 2017-01-05
## 4 1.066316e+14 106631626049851_1275484255831243   Ted Poe 2017-01-05
## 5 1.066316e+14 106631626049851_1276290242417311   Ted Poe 2017-01-06
##              datetime
## 1 2017-01-03 17:16:40
## 2 2017-01-03 23:11:15
## 3 2017-01-05 14:36:24
## 4 2017-01-05 15:38:22
## 5 2017-01-06 14:31:19
##
## 1
## 2
## 3
## 4
## 5 TERRORISM UPDATE:  ·ISIS fighters attacked an Iraqi army outpost and a police station near the city
##    type
## 1  link
## 2  link
## 3 photo
## 4  link
## 5  link
##
## 1                     https://www.c-span.org/video/?420804-1%2Fus-house-meets-elect-speaker-swear-mem
## 2                                  http://poe.house.gov/2017/1/congressman-poe-introduces-the-smart-b
## 3 https://www.facebook.com/JudgeTedPoe/photos/a.137207746325572.19816.106631626049851/127544471916853
## 4                                  http://poe.house.gov/press-releases?ID=443ADE64-03E2-461A-BA56-FFF
## 5                                                        http://poe.house.gov/index.cfm?p=terrori
##          domain likes_count comments_count shares_count love_count
```

```
## 1    c-span.org         121             15            6          5
## 2 poe.house.gov         182             16           31          8
## 3          <NA>         137             13           25          9
## 4 poe.house.gov        1125            166          170        244
## 5 poe.house.gov          24              8           12          1
##   haha_count wow_count sad_count angry_count
## 1          0         0         0           0
## 2          0         0         1           2
## 3          0         0         0           0
## 4          0        10         0           0
## 5          0         5         1           3
```

UPPER

```
dbGetQuery(db, "SELECT UPPER(message) FROM posts LIMIT 5")
```

```
##
## 1
## 2
## 3
## 4
## 5 TERRORISM UPDATE:  ·ISIS FIGHTERS ATTACKED AN IRAQI ARMY OUTPOST AND A POLICE STATION NEAR THE CIT
```

```
# We have a lower link which we made Upper adding expressions.
dbGetQuery(db, "SELECT from_name, likes_count/comments_count, UPPER(type) FROM posts LIMIT 5")
```

```
##   from_name likes_count/comments_count UPPER(type)
## 1   Ted Poe                         8        LINK
## 2   Ted Poe                        11        LINK
## 3   Ted Poe                        10       PHOTO
## 4   Ted Poe                         6        LINK
## 5   Ted Poe                         3        LINK
```

```
# Adding aliases to the new columns : AS
dbGetQuery(db, "SELECT from_name, likes_count/comments_count AS lIKE_RATIO FROM posts LIMIT 5")
```

```
##   from_name lIKE_RATIO
## 1   Ted Poe          8
## 2   Ted Poe         11
## 3   Ted Poe         10
## 4   Ted Poe          6
## 5   Ted Poe          3
```

```
# Best way to write the code in multiple level so that user can read with ease.

# You can modify how to show the column name
dbGetQuery(db, "SELECT LOWER(from_name),
           likes_count/comments_count AS like_ratio
           FROM posts
           LIMIT 5")
```

```
##   LOWER(from_name) like_ratio
## 1          ted poe          8
## 2          ted poe         11
## 3          ted poe         10
```

```
## 4          ted poe          6
## 5          ted poe          3
```

Distinct

```
# Unique values
dbGetQuery(db, "SELECT DISTINCT from_name
           FROM posts
           LIMIT 5")
```

```
##                   from_name
## 1                   Ted Poe
## 2 Congressman Wm. Lacy Clay
## 3   Congressman David Scott
## 4  Congressman Hank Johnson
## 5        Rep. Steve Stivers
```

```
# selecting based on values of a column
dbGetQuery(db, "SELECT name, party
           FROM congress
           WHERE party = 'Republican'
           LIMIT 5")
```

```
##                 name      party
## 1 Robert B. Aderholt Republican
## 2    Lamar Alexander Republican
## 3       Justin Amash Republican
## 4     Mark E. Amodei Republican
## 5      Rick W. Allen Republican
```

```
# working with dates greater than
dbGetQuery(db, "SELECT from_name, type, date
           FROM posts
           WHERE date > '2017-01-01'
           LIMIT 5")
```

```
##   from_name  type       date
## 1   Ted Poe  link 2017-01-03
## 2   Ted Poe  link 2017-01-03
## 3   Ted Poe photo 2017-01-05
## 4   Ted Poe  link 2017-01-05
## 5   Ted Poe  link 2017-01-06
```

```
# Between two dates
dbGetQuery(db, "SELECT from_name, type, date
           FROM posts
           WHERE date BETWEEN '2017-01-01' AND '2017-01-03'
           LIMIT 5")
```

```
##                  from_name  type       date
## 1                  Ted Poe  link 2017-01-03
## 2                  Ted Poe  link 2017-01-03
## 3 Congressman Hank Johnson video 2017-01-03
## 4       Rep. Steve Stivers photo 2017-01-01
## 5       Rep. Steve Stivers photo 2017-01-02
```

```
# AND operator
dbGetQuery(db, "SELECT from_name, type, date, likes_count
           FROM posts
           WHERE date > '2017-06-01' AND type != 'photo'AND likes_count > 500
           ORDER by likes_count DESC
           LIMIT 5")
```

```
##                    from_name  type       date likes_count
## 1 U.S. Senator Bernie Sanders video 2017-08-01      313536
## 2 U.S. Senator Bernie Sanders video 2017-08-01      313536
## 3 U.S. Senator Bernie Sanders video 2017-08-01      313536
## 4 U.S. Senator Bernie Sanders video 2017-08-01      313536
## 5 U.S. Senator Bernie Sanders video 2017-08-01      313536
```

```
# OR operator
dbGetQuery(db, "SELECT from_name, type, date, comments_count
           FROM posts
           WHERE comments_count>100 AND (type = 'photo' OR type = 'video')
           LIMIT 5")
```

```
##                  from_name  type       date comments_count
## 1                  Ted Poe video 2017-10-03            123
## 2                  Ted Poe photo 2017-11-24            176
## 3                  Ted Poe photo 2017-12-19            164
## 4 Congressman Wm. Lacy Clay photo 2017-01-04            317
## 5 Congressman Wm. Lacy Clay video 2017-01-12            163
```

```
# IN

dbGetQuery(db, "SELECT from_name, type, date, comments_count
           FROM posts
           WHERE type IN ('video', 'event')
           LIMIT 5")
```

```
##   from_name  type       date comments_count
## 1   Ted Poe video 2017-01-06              7
## 2   Ted Poe video 2017-01-24             23
## 3   Ted Poe video 2017-01-26             14
## 4   Ted Poe video 2017-01-31             19
## 5   Ted Poe video 2017-02-02             44
```

Matching conditions _ %:

```
# _ matches exactly one character:
dbGetQuery(db, "SELECT from_name, type, date, comments_count
           FROM posts
           WHERE date LIKE '2017-01-__'
           LIMIT 5")
```

```
##   from_name  type       date comments_count
## 1   Ted Poe  link 2017-01-03             15
## 2   Ted Poe  link 2017-01-03             16
## 3   Ted Poe photo 2017-01-05             13
## 4   Ted Poe  link 2017-01-05            166
```

```
## 5   Ted Poe  link 2017-01-06              8
```

```
# % matches any number of characters:
dbGetQuery(db, "SELECT from_name, type, date, comments_count
           FROM posts
           WHERE date LIKE '2017-03%'
           LIMIT 5")
```

```
##   from_name  type       date comments_count
## 1   Ted Poe photo 2017-03-01              1
## 2   Ted Poe video 2017-03-01             23
## 3   Ted Poe photo 2017-03-02             12
## 4   Ted Poe  link 2017-03-03             27
## 5   Ted Poe photo 2017-03-07              7
```

```
# SQLite does not have Regular Expressions, but we can get creative.
dbGetQuery(db, "SELECT from_name, message, date
           FROM posts
           WHERE message LIKE '%hungary%'
           LIMIT 5")
```

```
##                 from_name
## 1             Albio Sires
## 2 Congressman Doug Lamborn
## 3 Congressman Seth Moulton
## 4      Rep. Dennis A. Ross
## 5      Rep. Dennis A. Ross
##
## 1
## 2 Welcome Home Iron Brigade! I’m proud to have visited with our 3rd Armored Brigade Combat Team, 4th
## 3
## 4
## 5
##         date
## 1 2017-09-28
## 2 2017-10-05
## 3 2017-11-29
## 4 2017-05-05
## 5 2017-06-09
```

Group_by

```
dbGetQuery(db,
  "SELECT from_name, COUNT(*) AS post_count
  FROM posts
  GROUP BY from_name
  LIMIT 3")
```

```
##       from_name post_count
## 1   Albio Sires       2676
## 2    Ann Wagner       1140
## 3 Anthony Brown       2160
```

```
# sort : type_count by ORDER:
dbGetQuery(db,
  "SELECT type, COUNT(type) AS type_count
  FROM posts
  GROUP BY type
  ORDER BY type_count LIMIT 5")
```

```
##     type type_count
## 1  music        126
## 2   note        138
## 3  event       8004
## 4 status      88770
## 5  video     179568
```

```
# now in descending orders
dbGetQuery(db,
  "SELECT type, COUNT(type) AS type_count
  FROM posts
  GROUP BY type
  ORDER BY type_count DESC LIMIT 5")
```

```
##     type type_count
## 1  photo     406788
## 2   link     369072
## 3  video     179568
## 4 status      88770
## 5  event       8004
```

```
# top 3 most popular post?
dbGetQuery(db,
  "SELECT from_name, message, likes_count, datetime
  FROM posts
  ORDER BY likes_count DESC
  LIMIT 3")
```

```
##                      from_name
## 1 U.S. Senator Elizabeth Warren
## 2 U.S. Senator Elizabeth Warren
## 3 U.S. Senator Elizabeth Warren
##
## 1 Tonight we fight for American values at airports all across this country. I'm at Logan Airport toni
## 2 Tonight we fight for American values at airports all across this country. I'm at Logan Airport toni
## 3 Tonight we fight for American values at airports all across this country. I'm at Logan Airport toni
##   likes_count            datetime
## 1      421064 2017-01-29 01:59:12
## 2      421064 2017-01-29 01:59:12
## 3      421064 2017-01-29 01:59:12
```

```
# You can also specify the column number instead of the name
dbGetQuery(db,
  "SELECT from_name, message, likes_count, datetime
  FROM posts
  ORDER BY likes_count DESC
  LIMIT 2")
```

```
##                      from_name
```

```
## 1 U.S. Senator Elizabeth Warren
## 2 U.S. Senator Elizabeth Warren
##
## 1 Tonight we fight for American values at airports all across this country. I'm at Logan Airport toni
## 2 Tonight we fight for American values at airports all across this country. I'm at Logan Airport toni
##   likes_count            datetime
## 1      421064 2017-01-29 01:59:12
## 2      421064 2017-01-29 01:59:12
```

```
# what was the post with the highest comment to like ratio?
# We subset only posts with 1000 likes or more to avoid outliers.
dbGetQuery(db,
  "SELECT from_name, message, likes_count, comments_count, date,
      comments_count/likes_count AS comment_like_ratio
  FROM posts
  WHERE likes_count > 1000
  ORDER BY comment_like_ratio DESC
  LIMIT 5")
```

```
##                    from_name
## 1 U.S. Senator Susan Collins
## 2 U.S. Senator Susan Collins
## 3 U.S. Senator Susan Collins
## 4 U.S. Senator Susan Collins
## 5 U.S. Senator Susan Collins
##
## 1 After securing significant changes in the bill, as well as commitments to pass legislation to help
## 2 After securing significant changes in the bill, as well as commitments to pass legislation to help
## 3 After securing significant changes in the bill, as well as commitments to pass legislation to help
## 4 After securing significant changes in the bill, as well as commitments to pass legislation to help
## 5 After securing significant changes in the bill, as well as commitments to pass legislation to help
##   likes_count comments_count       date comment_like_ratio
## 1        1070          14974 2017-12-01                13
## 2        1070          14974 2017-12-01                13
## 3        1070          14974 2017-12-01                13
## 4        1070          14974 2017-12-01                13
## 5        1070          14974 2017-12-01                13
```

Join

```
library(DBI)
db <- dbConnect(RSQLite::SQLite(), "facebook-db.sqlite")

dbGetQuery(db,
  "SELECT posts.likes_count, congress.party, posts.date
    FROM posts JOIN congress
    ON congress.screen_name = posts.screen_name
    LIMIT 5")
```

```
##   likes_count      party       date
## 1         201 Republican 2017-01-03
## 2         201 Republican 2017-01-03
## 3         201 Republican 2017-01-03
```

```
## 4         201 Republican 2017-01-03
## 5         201 Republican 2017-01-03
```

ON

```
dbGetQuery(db,
  "SELECT posts.from_name, posts.message, posts.shares_count, congress.party
  FROM posts JOIN congress
    ON congress.screen_name = posts.screen_name
  WHERE party = 'Democrat'
  ORDER BY shares_count DESC
  LIMIT 3")
```

```
##                 from_name
## 1 Congressman Mark Takano
## 2 Congressman Mark Takano
## 3 Congressman Mark Takano
##
## 1 This remarkable line of questioning from Congresswoman Suzan DelBene demonstrates just a few of the
## 2 This remarkable line of questioning from Congresswoman Suzan DelBene demonstrates just a few of the
## 3 This remarkable line of questioning from Congresswoman Suzan DelBene demonstrates just a few of the
##   shares_count    party
## 1       341039 Democrat
## 2       341039 Democrat
## 3       341039 Democrat
```

```
dbGetQuery(db,
  "SELECT posts.from_name, posts.message, posts.shares_count, congress.party
  FROM posts JOIN congress
    ON congress.screen_name = posts.screen_name
  WHERE party = 'Republican'
  ORDER BY shares_count DESC
  LIMIT 3")
```

```
##     from_name
## 1 John McCain
## 2 John McCain
## 3 John McCain
##
## 1 Our government has a responsibility to defend our borders, but we must do so in a way that makes us
## 2 Our government has a responsibility to defend our borders, but we must do so in a way that makes us
## 3 Our government has a responsibility to defend our borders, but we must do so in a way that makes us
##   shares_count      party
## 1       100376 Republican
## 2       100376 Republican
## 3       100376 Republican
```

Grouping and Aggregating

```
# COUNT * = total no. of rows
```

```
dbGetQuery(db, 'SELECT COUNT(*) FROM posts')
```

```
##   COUNT(*)
## 1  1052466
```

```
dbGetQuery(db, 'SELECT COUNT(*) FROM congress')
```

```
##   COUNT(*)
## 1      518
```

```
dbGetQuery(db,
  "SELECT congress.party, COUNT(*) AS total_posts
  FROM posts JOIN congress
    ON congress.screen_name = posts.screen_name
  GROUP BY congress.party")
```

```
##         party total_posts
## 1    Democrat      510774
## 2 Independent        5358
## 3  Republican      536334
```

```
dbGetQuery(db,
  "SELECT congress.party, congress.gender, COUNT(*) AS total_posts
  FROM posts JOIN congress
    ON congress.screen_name = posts.screen_name
  GROUP BY congress.party, congress.gender")
```

```
##         party gender total_posts
## 1    Democrat      F      182724
## 2    Democrat      M      328050
## 3 Independent      M        5358
## 4  Republican      F       59358
## 5  Republican      M      476976
```

```
dbGetQuery(db,
  "SELECT congress.party, domain, COUNT(*) AS domain_count
  FROM posts JOIN Congress
    ON congress.screen_name = posts.screen_name
  WHERE congress.party = 'Democrat'
  GROUP BY domain
  ORDER BY domain_count DESC
  LIMIT 5")
```

```
##      party             domain domain_count
## 1 Democrat               <NA>       323574
## 2 Democrat        nytimes.com        12954
## 3 Democrat             bit.ly        12258
## 4 Democrat washingtonpost.com        11670
## 5 Democrat        thehill.com         5394
```

```
dbGetQuery(db,
  "SELECT congress.party, domain, COUNT(*) AS domain_count
  FROM posts JOIN Congress
    ON congress.screen_name = posts.screen_name
  WHERE congress.party = 'Republican'
  GROUP BY domain
  ORDER BY domain_count DESC
```

```
  LIMIT 5")
```

```
##        party                 domain domain_count
## 1 Republican                   <NA>       355170
## 2 Republican                 bit.ly        10338
## 3 Republican             foxnews.com         4248
## 4 Republican             thehill.com         3984
## 5 Republican washingtonexaminer.com         2784
```

```
# Average # of posts by party
dbGetQuery(db,
  "SELECT congress.party, AVG(posts.likes_count), COUNT(*)
  FROM posts JOIN congress
    ON congress.screen_name = posts.screen_name
  GROUP BY congress.party")
```

```
##         party AVG(posts.likes_count) COUNT(*)
## 1    Democrat               404.5807   510774
## 2 Independent             17207.3303     5358
## 3  Republican               171.2773   536334
```

```
# DIFFERENT WAY:
dbGetQuery(db,
  "SELECT congress.party, SUM(posts.likes_count)/COUNT(*) AS average
  FROM posts JOIN congress
    ON congress.screen_name = posts.screen_name
  GROUP BY congress.party")
```

```
##         party average
## 1    Democrat     404
## 2 Independent   17207
## 3  Republican     171
```

```
# most popular post by party
dbGetQuery(db,
  "SELECT from_name, message, congress.party, MAX(posts.likes_count), COUNT(*)
  FROM posts JOIN congress
    ON congress.screen_name = posts.screen_name
  GROUP BY congress.party")
```

```
##                      from_name
## 1 U.S. Senator Elizabeth Warren
## 2   U.S. Senator Bernie Sanders
## 3                   John McCain
##
## 1
## 2
## 3 Our government has a responsibility to defend our borders, but we must do so in a way that makes us
##         party MAX(posts.likes_count) COUNT(*)
## 1    Democrat                 421064   510774
## 2 Independent                 335572     5358
## 3  Republican                 288231   536334
```

```
# number of posts by day of the month
dbGetQuery(db,
  "SELECT SUBSTR(date, 9, 10) AS day_of_month, COUNT(*) as post_count
  FROM posts
```

```
  GROUP BY day_of_month")
```

```
##    day_of_month post_count
## 1            01      37188
## 2            02      32946
## 3            03      32958
## 4            04      34212
## 5            05      31836
## 6            06      38970
## 7            07      37986
## 8            08      33906
## 9            09      30822
## 10           10      33066
## 11           11      34836
## 12           12      36366
## 13           13      40332
## 14           14      37692
## 15           15      35256
## 16           16      35598
## 17           17      34104
## 18           18      34392
## 19           19      33048
## 20           20      36738
## 21           21      34482
## 22           22      31968
## 23           23      31152
## 24           24      34524
## 25           25      32586
## 26           26      32580
## 27           27      36084
## 28           28      37842
## 29           29      30546
## 30           30      27162
## 31           31      21288
```

```
library(sqldf)

sqldf("SELECT count(*) from congress")
```

```
##   count(*)
## 1      518
```