Assignment 4

Isabel Shaheen O'Malley

Include the GitHub link for the repository containing these files.

https://github.com/isabelshaheen/assignment-4.git

Install and load packages

```
if(!require(bit)) install.packages("bit")
```

```
Loading required package: bit
```

```
Attaching package: 'bit'
```

```
The following object is masked from 'package:base':

    xor
```

```
if(!require(bigrquery)) install.packages("bigrquery")
```

```
Loading required package: bigrquery
```

```
#| include: false
library(bit)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.3     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.0
v ggplot2   3.4.4     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.0
v purrr     1.0.2

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter()  masks stats::filter()
x dplyr::lag()     masks stats::lag()
x dplyr::symdiff() masks bit::symdiff()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```
library(DBI)
library(dbplyr)
```

```
Attaching package: 'dbplyr'

The following objects are masked from 'package:dplyr':

    ident, sql
```

```
library(bigrquery)
```

In this notebook we will use Google BigQuery, "Google's fully managed, petabyte scale, low cost analytics data warehouse". Some instruction on how to connect to Google BigQuery can be found here: https://db.rstudio.com/databases/big-query/.

You will need to set up a Google account with a project to be able to use this service. We will be using a public dataset that comes with 1 TB/mo of free processing on Google BigQuery. As long as you do not repeat the work in this notebook constantly, you should be fine with just the free tier.

Go to https://console.cloud.google.com and make sure you are logged in a non-university Google account. **This may not work on a university G Suite account because of restrictions on those accounts.** Create a new project by navigating to the dropdown menu at the top (it might say "Select a project") and selecting "New Project" in the window that pops up. Name it something useful.

After you have initialized a project, paste your project ID into the following chunk.

```
project <- "assignment-4-403119"
```

We will connect to a public database, the Chicago crime database, which has data on crime in Chicago.

```
con <- dbConnect(
  bigrquery::bigquery(),
  project = "bigquery-public-data",
  dataset = "chicago_crime",
  billing = project
)
con
```

```
<BigQueryConnection>
  Dataset: bigquery-public-data.chicago_crime
  Billing: assignment-4-403119
```

List some information about our connection setup

```
dbGetInfo(con)
```

```
$db.version
[1] NA

$dbname
[1] "bigquery-public-data.chicago_crime"

$username
[1] NA

$host
[1] NA

$port
[1] NA
```

We can look at the available tables in this database using `dbListTables`.

Note: When you run this code, you will be sent to a browser and have to give Google permissions to Tidyverse API Packages. **Make sure you select all to give access or else your code will not run.**

```
dbListTables(con)
```

```
! Using an auto-discovered, cached token.

  To suppress this message, modify your code or options to clearly consent to
  the use of a cached token.

  See gargle's "Non-interactive auth" vignette for more details:

  <https://gargle.r-lib.org/articles/non-interactive-auth.html>

i The bigrquery package is using a cached token for
  'isabel.s.omalley@gmail.com'.

Auto-refreshing stale OAuth token.

[1] "crime"
```

Information on the ‘crime‘ table can be found here:

https://cloud.google.com/bigquery/public-data/chicago-crime-data

Use code chunks with {sql connection = con} in order to write SQL code within the document.

Write a first query that counts the number of rows of the ‘crime‘ table in the year 2016.

```
SELECT COUNT (*) AS n_rows
FROM crime
WHERE year = 2016
LIMIT 10
```

Table 1: 1 records

n_rows
269841

Next, count the number of arrests grouped by `primary_type` in 2016. Note that is a somewhat similar task as above, with some adjustments on which rows should be considered. Sort the results, i.e. list the number of arrests in a descending order.

```
SELECT
  primary_type,
  COUNTIF(year = 2016) AS arrests_2016,
FROM crime
WHERE
arrest = TRUE
AND year IN (2016)
GROUP BY primary_type
ORDER BY count(*) DESC
LIMIT 50;
```

Table 2: Displaying records 1 - 10

primary_type	arrests_2016
NARCOTICS	13327
BATTERY	10332
THEFT	6522
CRIMINAL TRESPASS	3724
ASSAULT	3492
OTHER OFFENSE	3415
WEAPONS VIOLATION	2511
CRIMINAL DAMAGE	1669
PUBLIC PEACE VIOLATION	1116
MOTOR VEHICLE THEFT	1097

We can also use the `date` for grouping. Count the number of arrests grouped by hour of the day in 2016. You can extract the latter information from `date` via `EXTRACT(HOUR FROM date)`. Which time of the day is associated with the most arrests?

```
SELECT
  count(*) AS arrest_number,
  EXTRACT(HOUR FROM date) AS hour
FROM crime
WHERE arrest = TRUE
  AND year = 2016
GROUP BY hour
```

```
ORDER BY arrest_number DESC
LIMIT 50;
```

Table 3: Displaying records 1 - 10

arrest_number	hour
5306	10
5200	11
4942	12
4900	7
4735	8
4675	9
4288	1
4261	6
4029	2
3750	3

Focus only on `HOMICIDE` and count the number of arrests for this incident type, grouped by year. List the results in descending order.

```
SELECT
  year,
  COUNTIF(primary_type = "HOMICIDE") AS homicide_arrests,
FROM crime
WHERE
  primary_type = "HOMICIDE"
  AND arrest = TRUE
GROUP BY year
ORDER BY COUNTIF(primary_type = "HOMICIDE") DESC
LIMIT 20;
```

Table 4: Displaying records 1 - 10

year	homicide_arrests
2001	430
2002	424
2003	379
2020	339

year	homicide_arrests
2004	293
2016	286
2008	286
2005	281
2006	281
2021	275

Find out which districts have the highest numbers of arrests in 2015 and 2016. That is, count the number of arrests in 2015 and 2016, grouped by year and district. List the results in descending order.

```
SELECT
  district,
  COUNTIF(year = 2015) AS arrests_2015,
  COUNTIF(year = 2016) AS arrests_2016
FROM crime
WHERE arrest = TRUE
GROUP BY district
ORDER BY arrests_2015 DESC
LIMIT 50;
```

Table 5: Displaying records 1 - 10

district	arrests_2015	arrests_2016
11	8974	6575
7	5549	3654
15	4514	3072
6	4473	3447
25	4448	2947
4	4325	2837
8	4112	2948
10	3621	2951
9	3468	2592
5	3085	2701

Writing queries from within R via the DBI package

Lets switch to writing queries from within R via the `DBI` package.

Create a query object that counts the number of arrests grouped by `primary_type` of district 11 in year 2016. The results should be displayed in descending order. Execute the query.

```
sql <- "SELECT primary_type, COUNTIF(arrest = TRUE) AS arrests
        FROM `crime`
        WHERE district = 11 AND year = 2016
        GROUP BY primary_type
        ORDER BY arrests DESC
        LIMIT 100"
```

```
dbGetQuery(con, sql)
```

```
# A tibble: 32 x 2
   primary_type                     arrests
   <chr>                              <int>
 1 NARCOTICS                           3634
 2 BATTERY                              635
 3 PROSTITUTION                         511
 4 WEAPONS VIOLATION                    303
 5 OTHER OFFENSE                        255
 6 ASSAULT                              206
 7 CRIMINAL TRESPASS                    205
 8 PUBLIC PEACE VIOLATION               135
 9 INTERFERENCE WITH PUBLIC OFFICER     119
10 CRIMINAL DAMAGE                      106
# i 22 more rows
```

Try to write the very same query, now using the `dbplyr` package.

Start with the original crime table. Use LIMIT 100 to ensure the results object is manageable.

```
sql <- "SELECT *
        FROM `crime`
        LIMIT 100"

dbGetQuery(con, sql)
```

```
# A tibble: 100 x 22
   unique_key case_number date                block           iucr  primary_type
        <int> <chr>       <dttm>              <chr>           <chr> <chr>
```

```
 1    6581153 HP653445    2007-06-01 03:00:00 001XX W 107TH ~ 0266  CRIM SEXUAL~
 2   10875659 JA183609    2017-03-09 04:30:00 106XX S WABASH~ 0281  CRIM SEXUAL~
 3   12484396 JE375603    2021-09-15 11:30:00 106XX S PERRY ~ 0281  CRIMINAL SE~
 4   11470096 JB466080    2018-10-06 12:30:00 003XX W 105TH ~ 031A  ROBBERY
 5    8671773 HV346740    2012-06-21 11:03:00 105XX S EDBROO~ 031A  ROBBERY
 6    8692961 HV368745    2012-07-05 10:22:00 105XX S PERRY ~ 031A  ROBBERY
 7    9742421 HX392693    2014-08-16 02:00:00 103XX S WENTWO~ 031A  ROBBERY
 8    9855255 HX489208    2014-10-31 10:28:00 105XX S CALUME~ 031A  ROBBERY
 9    3229506 HK247037    2004-03-18 11:16:27 103XX S DR MAR~ 031A  ROBBERY
10   12408851 JE280416    2021-06-26 04:25:00 0000X W 103RD ~ 031A  ROBBERY
# i 90 more rows
# i 16 more variables: description <chr>, location_description <chr>,
#   arrest <lgl>, domestic <lgl>, beat <int>, district <int>, ward <int>,
#   community_area <int>, fbi_code <chr>, x_coordinate <dbl>,
#   y_coordinate <dbl>, year <int>, updated_on <dttm>, latitude <dbl>,
#   longitude <dbl>, location <chr>
```

```r
# Store the result (100 observations from the crime table) as an object in our workspace

subtable <- dbGetQuery(con, sql)
str(subtable)
```

```
tibble [100 x 22] (S3: tbl_df/tbl/data.frame)
 $ unique_key          : int [1:100] 6581153 10875659 12484396 11470096 8671773 8692961 9742
 $ case_number         : chr [1:100] "HP653445" "JA183609" "JE375603" "JB466080" ...
 $ date                : POSIXct[1:100], format: "2007-06-01 03:00:00" "2017-03-09 04:30:00"
 $ block               : chr [1:100] "001XX W 107TH ST" "106XX S WABASH AVE" "106XX S PERRY
 $ iucr                : chr [1:100] "0266" "0281" "0281" "031A" ...
 $ primary_type        : chr [1:100] "CRIM SEXUAL ASSAULT" "CRIM SEXUAL ASSAULT" "CRIMINAL SE
 $ description         : chr [1:100] "PREDATORY" "NON-AGGRAVATED" "NON-AGGRAVATED" "ARMED: HA
 $ location_description: chr [1:100] "RESIDENCE" "RESIDENCE" "ALLEY" "SIDEWALK" ...
 $ arrest              : logi [1:100] FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ domestic            : logi [1:100] FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ beat                : int [1:100] 512 512 512 512 512 512 512 512 512 512 ...
 $ district            : int [1:100] 5 5 5 5 5 5 5 5 5 5 ...
 $ ward                : int [1:100] 34 9 34 34 9 34 34 9 9 34 ...
 $ community_area      : int [1:100] 49 49 49 49 49 49 49 49 49 49 ...
 $ fbi_code            : chr [1:100] "02" "02" "02" "03" ...
 $ x_coordinate        : num [1:100] 1177377 1178459 1177489 1175884 1179134 ...
 $ y_coordinate        : num [1:100] 1834005 1834456 1834243 1835298 1835078 ...
 $ year                : int [1:100] 2007 2017 2021 2018 2012 2012 2014 2014 2004 2021 ...
```

```
 $ updated_on          : POSIXct[1:100], format: "2018-02-28 03:56:25" "2018-02-10 03:50:01"
 $ latitude            : num [1:100] 41.7 41.7 41.7 41.7 41.7 ...
 $ longitude           : num [1:100] -87.6 -87.6 -87.6 -87.6 -87.6 ...
 $ location            : chr [1:100] "(41.699836467, -87.626134333)" "(41.701049625, -87.622
```

Try a more specific query, *without* limiting to 100 observations

```
sql <- "SELECT year, primary_type, COUNTIF(arrest = TRUE) AS arrests
        FROM `crime`
        WHERE district = 11
        GROUP BY primary_type, year
        ORDER BY arrests DESC"

dbGetQuery(con, sql)
```

```
# A tibble: 638 x 3
    year primary_type arrests
   <int> <chr>          <int>
 1  2005 NARCOTICS       9718
 2  2003 NARCOTICS       9562
 3  2002 NARCOTICS       9232
 4  2004 NARCOTICS       9083
 5  2006 NARCOTICS       8185
 6  2001 NARCOTICS       7978
 7  2007 NARCOTICS       7395
 8  2013 NARCOTICS       7234
 9  2014 NARCOTICS       6801
10  2009 NARCOTICS       5941
# i 628 more rows
```

```
# Store the result as an object in our workspace

crime_tibble <- dbGetQuery(con, sql)
str(crime_tibble)
```

```
tibble [638 x 3] (S3: tbl_df/tbl/data.frame)
 $ year        : int [1:638] 2005 2003 2002 2004 2006 2001 2007 2013 2014 2009 ...
 $ primary_type: chr [1:638] "NARCOTICS" "NARCOTICS" "NARCOTICS" "NARCOTICS" ...
 $ arrests     : int [1:638] 9718 9562 9232 9083 8185 7978 7395 7234 6801 5941 ...
```

Again, count the number of arrests grouped by `primary_type` of district 11 in year 2016, now using `dplyr` syntax.

```
crime_tibble %>%
  filter(year == 2016)
```

```
# A tibble: 32 x 3
    year primary_type                     arrests
   <int> <chr>                              <int>
 1  2016 NARCOTICS                          3634
 2  2016 BATTERY                             635
 3  2016 PROSTITUTION                        511
 4  2016 WEAPONS VIOLATION                   303
 5  2016 OTHER OFFENSE                       255
 6  2016 ASSAULT                             206
 7  2016 CRIMINAL TRESPASS                   205
 8  2016 PUBLIC PEACE VIOLATION              135
 9  2016 INTERFERENCE WITH PUBLIC OFFICER    119
10  2016 CRIMINAL DAMAGE                     106
# i 22 more rows
```

Count the number of arrests grouped by `primary_type` and `year`, still only for district 11. Arrange the result by `year`.

```
crime_tibble %>%
  arrange(desc(year))
```

```
# A tibble: 638 x 3
    year primary_type                     arrests
   <int> <chr>                              <int>
 1  2023 NARCOTICS                          1237
 2  2023 WEAPONS VIOLATION                   403
 3  2023 BATTERY                             300
 4  2023 OTHER OFFENSE                       177
 5  2023 PROSTITUTION                        139
 6  2023 ASSAULT                             102
 7  2023 CRIMINAL DAMAGE                      51
 8  2023 MOTOR VEHICLE THEFT                  48
 9  2023 CRIMINAL TRESPASS                    43
10  2023 INTERFERENCE WITH PUBLIC OFFICER     39
# i 628 more rows
```

Assign the results of the query above to a local R object.

```
crime_tibble <- crime_tibble %>%
  arrange(desc(year))
```

Confirm that you pulled the data to the local environment by displaying the first ten rows of the saved data set.

```
head(crime_tibble, 10)
```

```
# A tibble: 10 x 3
    year primary_type                    arrests
   <int> <chr>                             <int>
 1  2023 NARCOTICS                         1237
 2  2023 WEAPONS VIOLATION                  403
 3  2023 BATTERY                            300
 4  2023 OTHER OFFENSE                      177
 5  2023 PROSTITUTION                       139
 6  2023 ASSAULT                            102
 7  2023 CRIMINAL DAMAGE                     51
 8  2023 MOTOR VEHICLE THEFT                 48
 9  2023 CRIMINAL TRESPASS                   43
10  2023 INTERFERENCE WITH PUBLIC OFFICER    39
```

Close the connection.

```
dbDisconnect(con)
```