Assignment 4

Due at 11:59pm on November 4.

Jamila Sani

This is an individual assignment. Turn in this assignment as an HTML or PDF file to ELMS. Make sure to include the R Markdown or Quarto file that was used to generate it. Include the GitHub link for the repository containing these files.

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 <- "jamilas-727-project"
```

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: jamilas-727-project
```

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 'sanijamila2016@gmail.com'.

[1] "crime"
```

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

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

Write a first query that counts the number of rows of the ‘crime‘ table in the year 2016. Use code chunks with {sql connection = con} in order to write SQL code within the document.

- Number of rows of the ‘crime’ table in the year 2016 = 269,933

```
SELECT count(primary_type) AS primary_count, count(*) AS overall_count -- counting non-missi
FROM crime
WHERE year = 2016             #filter where year is 2016
```

Table 1: 1 records

primary_count	overall_count
269934	269934

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.

- Top three arrests were:

  NARCOTICS 13,327
  BATTERY 10,334
  THEFT 6,522

```
SELECT primary_type, COUNT(*) AS arrests
FROM crime
WHERE year = 2016
  AND arrest = TRUE
GROUP BY primary_type
ORDER BY arrests DESC;
```

Table 2: Displaying records 1 - 10

primary_type	arrests
NARCOTICS	13327
BATTERY	10334
THEFT	6522
CRIMINAL TRESPASS	3724
ASSAULT	3494
OTHER OFFENSE	3416
WEAPONS VIOLATION	2510
CRIMINAL DAMAGE	1669
PUBLIC PEACE VIOLATION	1116
MOTOR VEHICLE THEFT	1098

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?

- Based on the top three arrests (>3,000 arrests), most arrests occurred in the evening between 6 p.m. and 8 p.m.

```
SELECT EXTRACT(HOUR FROM date) AS hour,
       COUNT(*) AS arrests
FROM crime
WHERE year = 2016
  AND arrest = TRUE
GROUP BY hour
ORDER BY arrests DESC;
```

Table 3: Displaying records 1 - 10

hour	arrests
19	3843
18	3482
20	3303
21	2962
16	2933
22	2896
11	2893
17	2821
12	2788
14	2775

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

- The top five number of arrests (>300 arrests) for homicide occurred in 2001, 2002, 2003, 2020, and 2022 respectively.

```
SELECT year, COUNT(*) AS arrests
FROM crime
WHERE primary_type = 'HOMICIDE'
  AND arrest = TRUE
GROUP BY year
ORDER BY arrests DESC;
```

Table 4: Displaying records 1 - 10

year	arrests
2001	431
2002	428
2003	386
2020	356
2022	321
2021	296
2004	294
2016	292
2008	288
2005	284

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.

- District 11 had the highest number of arrests in both 2015 and 2016.

```
SELECT year, district, COUNT(*) AS arrests
FROM crime
WHERE year IN (2015, 2016)
  AND arrest = TRUE
GROUP BY year, district
ORDER BY arrests DESC;
```

Table 5: Displaying records 1 - 10

year	district	arrests
2015	11	8975
2016	11	6578
2015	7	5549
2015	15	4514
2015	6	4476
2015	25	4451
2015	4	4326
2015	8	4115
2016	7	3656
2015	10	3628

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.

```
query <- "
  SELECT year, primary_type, COUNT(*) AS arrests
  FROM crime
  WHERE district = 11
    AND year = 2016
    AND arrest = TRUE
GROUP BY year, primary_type
ORDER BY arrests DESC;
"
district11_data <- dbGetQuery(con, query)
head (district11_data, 10)
```

```
# A tibble: 10 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                            207
 7  2016 CRIMINAL TRESPASS                  205
 8  2016 PUBLIC PEACE VIOLATION             135
 9  2016 INTERFERENCE WITH PUBLIC OFFICER   119
10  2016 CRIMINAL DAMAGE                    106
```

Try to write the very same query, now using the `dbplyr` package. For this, you need to first map the `crime` table to a tibble object in R.

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

- As expected, the counts/results are the same as the previous question.

```
crime_tbl <- tbl (con, 'crime')

district11_tbl <- crime_tbl %>%
  filter(district == 11, year == 2016, arrest == TRUE) %>%
  group_by(primary_type) %>%
  summarise(arrests = n()) %>%
  arrange (desc(arrests))

head (district11_tbl, 10)
```

```
# Source:     SQL [?? x 2]
# Database:   BigQueryConnection
# Ordered by: desc(arrests)
   primary_type                      arrests
   <chr>                               <int>
 1 NARCOTICS                          3634
 2 BATTERY                             635
 3 PROSTITUTION                        511
 4 WEAPONS VIOLATION                   303
 5 OTHER OFFENSE                       255
 6 ASSAULT                             207
 7 CRIMINAL TRESPASS                   205
 8 PUBLIC PEACE VIOLATION              135
 9 INTERFERENCE WITH PUBLIC OFFICER    119
10 CRIMINAL DAMAGE                     106
```

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

```
tibble_crime <- tbl (con, 'crime')

tibble_11_year <- tibble_crime %>%
  filter(district == 11, arrest == TRUE) %>%
  group_by(primary_type, year) %>%  # grouped by 'primary_type' and 'year'
  summarise(arrests = n()) %>%
  arrange (desc(year))
head (tibble_11_year)
```

```
`summarise()` has grouped output by "primary_type". You can override using the
`.groups` argument.
```

```
# Source:     SQL [?? x 3]
# Database:   BigQueryConnection
# Groups:     primary_type
# Ordered by: desc(year)
  primary_type       year arrests
  <chr>             <int>   <int>
1 ARSON              2025       1
2 NARCOTICS          2025    2044
3 CRIMINAL TRESPASS  2025      59
4 OTHER OFFENSE      2025     116
5 THEFT              2025      27
6 BATTERY            2025     348
```

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

```
district11_local <- collect (tibble_11_year)
```

```
`summarise()` has grouped output by "primary_type". You can override using the
`.groups` argument.
```

```
district11_local
```

```
# A tibble: 638 x 3
# Groups:   primary_type [31]
   primary_type                       year arrests
   <chr>                             <int>   <int>
 1 BURGLARY                           2025       7
 2 WEAPONS VIOLATION                  2025     301
 3 HOMICIDE                           2025       7
 4 NARCOTICS                          2025    2044
 5 STALKING                           2025       1
 6 SEX OFFENSE                        2025       3
 7 LIQUOR LAW VIOLATION               2025       1
 8 CRIMINAL TRESPASS                  2025      59
 9 THEFT                              2025      27
10 CONCEALED CARRY LICENSE VIOLATION  2025       8
# i 628 more rows
```

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

```r
head (district11_local, 10)
```

```
# A tibble: 10 x 3
# Groups:   primary_type [10]
   primary_type                       year arrests
   <chr>                             <int>   <int>
 1 BURGLARY                           2025       7
 2 WEAPONS VIOLATION                  2025     301
 3 HOMICIDE                           2025       7
 4 NARCOTICS                          2025    2044
 5 STALKING                           2025       1
 6 SEX OFFENSE                        2025       3
 7 LIQUOR LAW VIOLATION               2025       1
 8 CRIMINAL TRESPASS                  2025      59
 9 THEFT                              2025      27
10 CONCEALED CARRY LICENSE VIOLATION  2025       8
```

Close the connection.

```r
dbDisconnect(con)
```