Assignment 4

Due at 11:59pm on November 5.

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 <- "melodic-furnace-439518-p7"
```

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: melodic-furnace-439518-p7
```

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)
```

```
i Suitable tokens found in the cache, associated with these emails:

* 'meganliney@gmail.com'

* 'xinyulin@umich.edu'

  Defaulting to the first email.

! 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 'meganliney@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.

```
SELECT count(primary_type), count(*)
FROM crime
WHERE year = 2016
LIMIT 10;
```

Table 1: 1 records

f0_	f1_
269922	269922

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, count(*)
FROM crime
WHERE year = 2016 AND arrest = True
GROUP BY primary_type
ORDER BY count(*) DESC;
```

Table 2: Displaying records 1 - 10

primary_type	f0_
NARCOTICS	13327
BATTERY	10333
THEFT	6522
CRIMINAL TRESPASS	3724
ASSAULT	3492
OTHER OFFENSE	3415
WEAPONS VIOLATION	2511
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?

```
SELECT EXTRACT(HOUR FROM date) as hour, count(*)
FROM crime
WHERE year = 2016 AND arrest = True
GROUP BY hour
ORDER BY count(*) DESC;
```

Table 3: Displaying records 1 - 10

hour	f0_
19	3843
18	3481
20	3302
21	2961
16	2933
22	2896
11	2895
17	2820
12	2787
14	2774

Answer: It can be noticed that at 19:00 there are most arrests.

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, count(*)
FROM crime
WHERE primary_type ="HOMICIDE" AND arrest = True
GROUP BY year
ORDER BY count(*) DESC;
```

Table 4: Displaying records 1 - 10

year	f0_
2001	430
2002	427
2003	382
2020	349
2022	306
2004	294

year	f0_
2021	292
2016	289
2008	287
2006	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.

```sql
SELECT district, year, count(*)
FROM crime
WHERE year IN (2015, 2016) AND arrest = True
GROUP BY year, district
ORDER BY count(*) DESC;
```

Table 5: Displaying records 1 - 10

district	year	f0_
11	2015	8974
11	2016	6575
7	2015	5549
15	2015	4514
6	2015	4474
25	2015	4450
4	2015	4325
8	2015	4113
7	2016	3655
10	2015	3622

Answer: District 11 had the highest ARRESTS in 2015 & 2016. We can

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 primary_type, count(*)
FROM crime
WHERE year = 2016 AND district = 11 AND arrest = True
GROUP BY primary_type
ORDER BY count(*) DESC;"
dbGetQuery(con, query)
```

```
# A tibble: 27 x 2
   primary_type                        f0_
   <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 17 more rows
```

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.

```
crime <- tbl(con, "crime")
```

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

```
sql <- crime %>%
  filter(year == 2016, district == 11, arrest == TRUE)%>%
  group_by(primary_type)%>%
  summarise(count = n())%>%
  arrange(desc(count))
sql
```

```
# Source:     SQL [?? x 2]
# Database:   BigQueryConnection
# Ordered by: desc(count)
```

```
   primary_type                      count
   <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 more rows
```

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

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

```r
sql1 <- crime %>%
  filter(district == 11, arrest == TRUE)%>%
  group_by(primary_type, year)%>%
  summarise(count = n(), .groups = "drop")%>%
  arrange(year)%>%
  collect()
sql1
```

```
# A tibble: 613 x 3
   primary_type          year count
   <chr>                <int> <int>
 1 BATTERY               2001   962
 2 PROSTITUTION          2001   424
 3 THEFT                 2001   419
 4 CRIM SEXUAL ASSAULT   2001    17
 5 DECEPTIVE PRACTICE    2001    84
 6 GAMBLING              2001    71
 7 INTIMIDATION          2001     3
 8 CRIMINAL TRESPASS     2001   389
 9 WEAPONS VIOLATION     2001   236
10 NARCOTICS             2001  7979
# i 603 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(sql1, 10)
```

```
# A tibble: 10 x 3
   primary_type         year count
   <chr>               <int> <int>
 1 BATTERY              2001   962
 2 PROSTITUTION         2001   424
 3 THEFT                2001   419
 4 CRIM SEXUAL ASSAULT  2001    17
 5 DECEPTIVE PRACTICE   2001    84
 6 GAMBLING             2001    71
 7 INTIMIDATION         2001     3
 8 CRIMINAL TRESPASS    2001   389
 9 WEAPONS VIOLATION    2001   236
10 NARCOTICS            2001  7979
```

Close the connection.

```r
dbDisconnect(con)
```