# Assignment 4

**Due at 11:59pm on November 4.**

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.

```r
project <- "survmeth"
```

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

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

```
<BigQueryConnection>
  Dataset: bigquery-public-data.chicago_crime
  Billing: survmeth
```

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:

* 'liushengkun0128@gmail.com'

* 'shengkun@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
  'liushengkun0128@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) AS primary_count, count(*) AS overall_count -- counting non-missi
FROM crime
WHERE year = 2016
LIMIT 10;
```

Table 1: 1 records

| primary_count | overall_count |
|---:|---:|
| 269937 | 269937 |

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 DISTINCT arrest
FROM crime;
```

Table 2: 2 records

| arrest |
|:---|
| FALSE |
| TRUE |

```
SELECT
primary_type,
count(primary_type) AS arrest_count_by_type
FROM crime
WHERE year = 2016
AND arrest = TRUE
GROUP BY primary_type
ORDER BY arrest_count_by_type DESC
LIMIT 10;
```

Table 3: Displaying records 1 - 10

| primary_type | arrest_count_by_type |
|:---|---:|
| NARCOTICS | 13327 |
| BATTERY | 10334 |

| primary_type | arrest_count_by_type |
|---|---:|
| 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?

```sql
SELECT
EXTRACT(HOUR FROM date) AS hour_of_day,
COUNT(primary_type) AS arrest_count_by_date
FROM crime
WHERE year = 2016
AND arrest = TRUE
GROUP BY hour_of_day
ORDER BY arrest_count_by_date DESC
LIMIT 10;
```

Table 4: Displaying records 1 - 10

| hour_of_day | arrest_count_by_date |
|---:|---:|
| 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.

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

Table 5: Displaying records 1 - 10

| year | arrest__count__by__homicide |
|---|---:|
| 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.

```
SELECT
district,
year,
COUNT(primary_type) AS arrest_count
From crime
WHERE arrest = TRUE
  AND year IN (2015, 2016)
GROUP BY district, year
ORDER BY arrest_count DESC;
```

Table 6: Displaying records 1 - 10

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

- district No. 11 has the highest numbers of arrests in both 2015 and 2016.

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.

```
query <- "
SELECT
  primary_type,
  COUNT(primary_type) AS arrest_count
FROM crime
WHERE district = 11
  AND year = 2016
  AND arrest = TRUE
GROUP BY primary_type
ORDER BY arrest_count DESC;
"
```

Execute the query.

```
result_D11 <- dbGetQuery(con, query)
result_D11
```

```
# A tibble: 27 x 2
   primary_type                      arrest_count
   <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
# 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.

```
# map the table in R
crime_dplyr_tbl <- tbl(con, "crime")
```

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

```
query_dplyr <- crime_dplyr_tbl %>%
  filter(district == 11, year == 2016, arrest == TRUE) %>%
  group_by(primary_type) %>%
  summarise(arrest_count = n()) %>%
  arrange(desc(arrest_count))

query_dplyr
```

```
# Source:     SQL [?? x 2]
# Database:   BigQueryConnection
# Ordered by: desc(arrest_count)
   primary_type                     arrest_count
   <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
# i more rows
```

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

```r
query_dplyr_2 <- crime_dplyr_tbl %>%
  filter(district == 11, arrest == TRUE) %>%
  group_by(primary_type, year) %>%
  summarise(arrest_count = n(), .groups = "drop") %>%
  arrange(year)

query_dplyr_2
```

```
# Source:     SQL [?? x 3]
# Database:   BigQueryConnection
# Ordered by: year
   primary_type         year arrest_count
   <chr>               <int>        <int>
 1 BATTERY              2001          962
 2 DECEPTIVE PRACTICE   2001           84
 3 KIDNAPPING           2001            4
 4 CRIMINAL DAMAGE      2001          163
 5 MOTOR VEHICLE THEFT  2001          179
 6 CRIM SEXUAL ASSAULT  2001           17
 7 THEFT                2001          419
 8 SEX OFFENSE          2001           19
 9 BURGLARY             2001           42
10 CRIMINAL TRESPASS    2001          389
# i more rows
```

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

```r
result_query_dplyr_2 <- collect(query_dplyr_2)
```

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

```r
head(result_query_dplyr_2, 10)
```

```
# A tibble: 10 x 3
   primary_type                year arrest_count
   <chr>                      <int>        <int>
 1 LIQUOR LAW VIOLATION        2001           49
 2 SEX OFFENSE                 2001           19
 3 WEAPONS VIOLATION           2001          236
 4 ASSAULT                     2001          322
 5 INTIMIDATION                2001            3
 6 ROBBERY                     2001           97
 7 CRIM SEXUAL ASSAULT         2001           17
 8 OFFENSE INVOLVING CHILDREN  2001           44
 9 DECEPTIVE PRACTICE          2001           84
10 KIDNAPPING                  2001            4
```

Close the connection.

```r
dbDisconnect(con)
```