Assignment 4

Due at 11:59pm on November 5.

Sagnik Chakravarty and Namit Shrivastava

Github link: https://github.com/namo507/SURV-727

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 <- "surv-727-ass4"
```

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

```r
bq_auth(email = 'sagnikchakravarty7@gmail.com')
con <- dbConnect(
  bigrquery::bigquery(),
  project = "bigquery-public-data",
  dataset = "chicago_crime",
  billing = project
)
con
```

```
<BigQueryConnection>
  Dataset: bigquery-public-data.chicago_crime
  Billing: surv-727-ass4
```

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)
```

```
[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_
269920	269920

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(*) as arrest_count
FROM crime
WHERE year = 2016 AND arrest = TRUE
GROUP BY primary_type
ORDER BY arrest_count DESC
LIMIT 20;
```

Table 2: Displaying records 1 - 10

primary_type	arrest_count
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_of_day, COUNT(*) AS arrest_count
FROM crime
WHERE year = 2016 AND arrest = TRUE
GROUP BY hour_of_day
ORDER BY arrest_count DESC
LIMIT 10;
```

Table 3: Displaying records 1 - 10

hour_of_day	arrest_count
19	3843
18	3481
20	3302
21	2961
16	2933

hour_of_day	arrest_count
22	2896
11	2895
17	2820
12	2787
14	2774

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 as year, COUNT(*) as arrest_count
FROM crime
WHERE primary_type = 'HOMICIDE' AND arrest = TRUE
GROUP BY year
ORDER BY arrest_count DESC
LIMIT 10
```

Table 4: Displaying records 1 - 10

year	arrest_count
2001	430
2002	427
2003	382
2020	349
2022	306
2004	294
2021	291
2016	289
2008	287
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 as district, COUNT(*) as arrest_count
FROM crime
WHERE year = 2015 or year = 2016 and arrest = TRUE
GROUP BY DISTRICT
ORDER BY arrest_count DESC
```

```
LIMIT 10;
```

Table 5: Displaying records 1 - 10

district	arrest_count
11	26110
8	20302
6	19526
7	19433
4	18726
25	18038
3	15453
9	15356
15	14833
10	14755

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(*) as arrest_count
FROM crime
WHERE district = 11 and year = 2016 and arrest = TRUE
GROUP BY primary_type
ORDER BY arrest_count DESC
LIMIT 10;"

result <- dbGetQuery(con, query)

print(result)
```

```
# A tibble: 10 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                                   206
 7 CRIMINAL TRESPASS                         205
 8 PUBLIC PEACE VIOLATION                    135
 9 INTERFERENCE WITH PUBLIC OFFICER          119
10 CRIMINAL DAMAGE                           106
```

Execute the query.

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')
str(crime)
```

```
List of 2
 $ src       :List of 2
  ..$ con  :Formal class 'BigQueryConnection' [package "bigrquery"] with 7 slots
  .. .. ..@ project       : chr "bigquery-public-data"
  .. .. ..@ dataset       : chr "chicago_crime"
  .. .. ..@ billing       : chr "surv-727-ass4"
  .. .. ..@ use_legacy_sql: logi FALSE
  .. .. ..@ page_size     : int 10000
  .. .. ..@ quiet         : logi NA
  .. .. ..@ bigint        : chr "integer"
  ..$ disco: NULL
  ..- attr(*, "class")= chr [1:4] "src_BigQueryConnection" "src_dbi" "src_sql" "src"
 $ lazy_query:List of 6
  ..$ x         : 'dbplyr_table_path' chr "`crime`"
  ..$ vars      : chr [1:22] "unique_key" "case_number" "date" "block" ...
  ..$ group_vars: chr(0)
  ..$ order_vars: NULL
  ..$ frame     : NULL
  ..$ is_view   : logi FALSE
  ..- attr(*, "class")= chr [1:3] "lazy_base_remote_query" "lazy_base_query" "lazy_query"
 - attr(*, "class")= chr [1:5] "tbl_BigQueryConnection" "tbl_dbi" "tbl_sql" "tbl_lazy" ...
```

```
class(crime)
```

```
[1] "tbl_BigQueryConnection" "tbl_dbi"                "tbl_sql"
[4] "tbl_lazy"               "tbl"
```

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

```
res <- crime %>%
  filter(district == 11, year == 2016, arrest == TRUE) %>%
  group_by(primary_type) %>%
  summarize(arrest_count = n()) %>%
  arrange(desc(arrest_count))
```

```
print(head(res, 10))
```

```
# Source:     SQL [10 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                                   206
 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.

```
res1 <- crime %>%
  filter(district == 11, arrest == TRUE) %>%
  group_by(primary_type, year) %>%
  summarize(arrest_count = n()) %>%
  arrange(year)
```

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

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

```
print(head(res1, 10))
```

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

```
# Source:     SQL [10 x 3]
# Database:   BigQueryConnection
# Groups:     primary_type
# Ordered by: year
   primary_type            year arrest_count
   <chr>                  <int>        <int>
 1 STALKING                2001            1
 2 CRIMINAL TRESPASS       2001          389
 3 ARSON                   2001           12
 4 MOTOR VEHICLE THEFT     2001          179
 5 BURGLARY                2001           42
 6 DECEPTIVE PRACTICE      2001           84
 7 CRIM SEXUAL ASSAULT     2001           17
 8 NARCOTICS               2001         7979
 9 WEAPONS VIOLATION       2001          236
10 PUBLIC PEACE VIOLATION  2001           34
```

Close the connection.

```
dbDisconnect(con)
```