Assignment 4

Kangrui Liu

2023-10-27

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.

Github Link –> https://github.com/krliu67/Assignment_SURV727/tree/main/a4

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 <- "surv-727-test-403117"
```

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: surv-727-test-403117
```

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 'kangruiliusz@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(*)
FROM crime;
```

Table 1: 1 records

f0_
7921096

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 group by primary_type;
```

Table 2: Displaying records 1 - 10

primary_type	f0_
CRIMINAL SEXUAL ASSAULT	7666
BATTERY	1446344
STALKING	5074
CRIMINAL DAMAGE	902905
CRIMINAL TRESPASS	216607
LIQUOR LAW VIOLATION	14993
OBSCENITY	839
NON - CRIMINAL	38
DOMESTIC VIOLENCE	1

primary_type	f0_
HOMICIDE	12897

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 date, count(*) counts_date FROM crime WHERE date >= "2016-01-01 00:00:00" and date <= "2017-01-0
```

Table 3: Displaying records 1 - 10

date	counts_date
2016-04-01 12:00:00	69
2016-01-14 01:00:00	16
2016-11-02 05:00:00	29
2016-03-15 09:00:00	31
2016-04-29 09:00:00	34
2016-06-24 10:00:00	28
2016-10-05 09:00:00	45
2016-07-10 12:00:00	24
2016-04-02 05:00:00	19
2016-01-22 06:00:00	25

```
query <- "SELECT date, count(*) counts_date FROM crime WHERE date >= '2016-01-01 00:00:00' and date <=
count_date_2016 <- dbGetQuery(con, query)
count_date_2016$date <- as.character(count_date_2016$date)
```

```
count_date_2016 %<>%
  separate(date, c("date","time"), sep = " ")

count_date_2016 <- count_date_2016[,-1]

count_date_2016 <- aggregate(counts_date ~ time, data = count_date_2016, sum)

count_date_2016 %>% subset(counts_date==max(count_date_2016$counts_date))
```

```
##         time counts_date
## 831 09:00:00       11176
```

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

```
SELECT date, count(*) counts_date FROM crime WHERE primary_type = "HOMICIDE" GROUP BY date ORDER BY cou
```

Table 4: Displaying records 1 - 10

date	counts_date
2003-08-27 08:35:00	6
2016-02-04 01:00:00	6
2008-04-23 06:15:00	5
2007-03-10 06:56:00	4
2017-03-30 04:32:00	4
2010-04-14 04:25:00	4
2015-07-04 04:00:00	4
2010-09-02 08:30:00	4
2016-08-23 01:34:00	4
2010-04-21 09:05:00	3

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, count(*) counts_district FROM crime WHERE date >= "2015-01-01 00:00:00" and date <= "20
```

Table 5: Displaying records 1 - 10

district	counts_district
11	38189
8	34914
6	32298
4	30916
7	30012
25	29660
12	26441
1	25750
9	25452
3	25450

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.

```
SELECT primary_type, count(*) counts FROM crime WHERE date >= "2016-01-01 00:00:00" and date <= "2017-01
```

Table 6: Displaying records 1 - 10

primary_type	counts
BATTERY	3906
NARCOTICS	3635
THEFT	2043
CRIMINAL DAMAGE	1775
ASSAULT	1330
OTHER OFFENSE	1045
ROBBERY	1007

primary_type	counts
MOTOR VEHICLE THEFT	776
DECEPTIVE PRACTICE	609
PROSTITUTION	511

```
query <- "SELECT primary_type, count(*) counts FROM crime WHERE date >= '2016-01-01 00:00:00' and date <
```

Execute the query.

```
dbGetQuery(con,query)
```

```
## # A tibble: 10 x 2
##    primary_type        counts
##    <chr>                <int>
##  1 BATTERY               3906
##  2 NARCOTICS             3635
##  3 THEFT                 2043
##  4 CRIMINAL DAMAGE       1775
##  5 ASSAULT               1330
##  6 OTHER OFFENSE         1045
##  7 ROBBERY               1007
##  8 MOTOR VEHICLE THEFT    776
##  9 DECEPTIVE PRACTICE     609
## 10 PROSTITUTION           511
```

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")
```

```
## Warning: <BigQueryConnection> uses an old dbplyr interface
## i Please install a newer version of the package or contact the maintainer
## This warning is displayed once every 8 hours.
```

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

```
crime %>% filter(district == 11, date >= '2016-01-01 00:00:00' && date <= '2017-01-01 00:00:00')%>% grou
```

```
## # Source:   SQL [?? x 2]
## # Database: BigQueryConnection
## # Groups:   primary_type
##    primary_type               n
##    <chr>                  <int>
##  1 BATTERY                 3906
##  2 CRIMINAL DAMAGE         1775
##  3 CRIMINAL TRESPASS        323
##  4 LIQUOR LAW VIOLATION      11
##  5 STALKING                  10
##  6 NON - CRIMINAL             1
```

```
##  7 CRIMINAL SEXUAL ASSAULT     8
##  8 OBSCENITY                   2
##  9 DECEPTIVE PRACTICE        609
## 10 HOMICIDE                   95
## # i more rows
```

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

```
crime %>% filter(district == 11) %>% group_by(year, primary_type) %>%   summarize(arrest_count = n()) %
```

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

```
## # Source:     SQL [?? x 3]
## # Database:   BigQueryConnection
## # Groups:     year
## # Ordered by: year
##     year primary_type         arrest_count
##    <int> <chr>                       <int>
##  1  2001 HOMICIDE                      72
##  2  2001 KIDNAPPING                    36
##  3  2001 LIQUOR LAW VIOLATION          49
##  4  2001 CRIMINAL DAMAGE             2193
##  5  2001 STALKING                       5
##  6  2001 CRIMINAL TRESPASS            515
##  7  2001 THEFT                       3098
##  8  2001 BURGLARY                     866
##  9  2001 ASSAULT                     1667
## 10  2001 BATTERY                     5938
## # i more rows
```

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

```
year_primary <- crime %>% filter(district == 11) %>% group_by(year, primary_type) %>%   summarize(arres
```

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

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

```
year_primary[c(1:10),]
```

```
##    year               primary_type arrest_count
## 1  2001                    ROBBERY         1243
## 2  2001        MOTOR VEHICLE THEFT         1183
## 3  2001 OFFENSE INVOLVING CHILDREN          140
## 4  2001              OTHER OFFENSE         1150
## 5  2001                     ARSON           47
## 6  2001               SEX OFFENSE           67
```

```
## 7  2001                    BATTERY        5938
## 8  2001             CRIMINAL DAMAGE       2193
## 9  2001          CRIMINAL TRESPASS         515
## 10 2001                 KIDNAPPING          36
```

Close the connection.

```
dbDisconnect(con)
```