Cases from EHR

```
library(tidyverse)
library(bigrquery)

# This query represents dataset "prostate_cancer_cases" for domain
"condition" and was generated for All of Us Controlled Tier Dataset v7
dataset_87448626_condition_sql <- paste("
    SELECT
        c_occurrence.person_id,
        c_occurrence.condition_concept_id,
        c_standard_concept.concept_name as standard_concept_name,
        c_standard_concept.concept_code as standard_concept_code,
        c_standard_concept.vocabulary_id as standard_vocabulary,
        c_occurrence.condition_start_datetime
    FROM
        ( SELECT
            *
        FROM
            `condition_occurrence` c_occurrence
        WHERE
            (
                condition_concept_id IN (SELECT
                    DISTINCT c.concept_id
                FROM
                    `cb_criteria` c
                JOIN
                    (SELECT
                        CAST(cr.id as string) AS id
                    FROM
                        `cb_criteria` cr
                    WHERE
                        concept_id IN (200962, 36712762, 37016740,
4163261, 4196262)
                        AND full_text LIKE '%_rank1]%'      ) a
                        ON (c.path LIKE CONCAT('%.', a.id, '.%')
                        OR c.path LIKE CONCAT('%.', a.id)
                        OR c.path LIKE CONCAT(a.id, '.%')
                        OR c.path = a.id)
                WHERE
                    is_standard = 1
                    AND is_selectable = 1)
            )
            AND (
                c_occurrence.PERSON_ID IN (SELECT
                    distinct person_id
                FROM
```

```
                    `cb_search_person` cb_search_person
                WHERE
                    cb_search_person.person_id IN (SELECT
                        person_id
                    FROM
                        `cb_search_person` p
                    WHERE
                        has_whole_genome_variant = 1 )
                    AND cb_search_person.person_id IN (SELECT
                        person_id
                    FROM
                        `cb_search_person` p
                    WHERE
                        has_ehr_data = 1 ) )
            )) c_occurrence
    LEFT JOIN
        `concept` c_standard_concept
            ON c_occurrence.condition_concept_id =
c_standard_concept.concept_id", sep="")

# Formulate a Cloud Storage destination path for the data exported
from BigQuery.
# NOTE: By default data exported multiple times on the same day will
overwrite older copies.
#       But data exported on a different days will write to a new
location so that historical
#       copies can be kept as the dataset definition is changed.
condition_87448626_path <- file.path(
  Sys.getenv("WORKSPACE_BUCKET"),
  "bq_exports",
  Sys.getenv("OWNER_EMAIL"),
  strftime(lubridate::now(), "%Y%m%d"),  # Comment out this line if
you want the export to always overwrite.
  "condition_87448626",
  "condition_87448626_*.csv")
message(str_glue('The data will be written to
{condition_87448626_path}. Use this path when reading ',
                 'the data into your notebooks in the future.'))

# Perform the query and export the dataset to Cloud Storage as CSV
files.
# NOTE: You only need to run `bq_table_save` once. After that, you can
#       just read data from the CSVs in Cloud Storage.
bq_table_save(
  bq_dataset_query(Sys.getenv("WORKSPACE_CDR"),
dataset_87448626_condition_sql, billing =
Sys.getenv("GOOGLE_PROJECT")),
  condition_87448626_path,
  destination_format = "CSV")
```

```r
# Read the data directly from Cloud Storage into memory.
# NOTE: Alternatively you can `gsutil -m cp {condition_87448626_path}`
to copy these files
#       to the Jupyter disk.
read_bq_export_from_workspace_bucket <- function(export_path) {
  col_types <- cols(standard_concept_name = col_character(),
standard_concept_code = col_character(), standard_vocabulary =
col_character())
  bind_rows(
    map(system2('gsutil', args = c('ls', export_path), stdout = TRUE,
stderr = TRUE),
        function(csv) {
          message(str_glue('Loading {csv}.'))
          chunk <- read_csv(pipe(str_glue('gsutil cat {csv}')),
col_types = col_types, show_col_types = FALSE)
          if (is.null(col_types)) {
            col_types <- spec(chunk)
          }
          chunk
        }))
}
condition_df <-
read_bq_export_from_workspace_bucket(condition_87448626_path)

dim(condition_df)
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.4     ✔ readr     2.1.5
✔ forcats   1.0.0     ✔ stringr   1.5.1
✔ ggplot2   3.5.2     ✔ tibble    3.2.1
✔ lubridate 1.9.4     ✔ tidyr     1.3.1
✔ purrr     1.0.4
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force
all conflicts to become errors
The data will be written to gs://fc-secure-672eeb92-4859-4ed9-9f59-
d4349f3534a0/bq_exports/micah_hysong@researchallofus.org/20250825/
condition_87448626/condition_87448626_*.csv. Use this path when
reading the data into your notebooks in the future.

Loading
gs://fc-secure-672eeb92-4859-4ed9-9f59-d4349f3534a0/bq_exports/micah_h
ysong@researchallofus.org/20250825/condition_87448626/
condition_87448626_000000000000.csv.
```

```
[1] 360831      6
```

```
unique(condition_df$standard_concept_name)
```

```
 [1] "Prostate cancer metastatic to bone"
 [2] "Metastatic castration-resistant prostate cancer"
 [3] "Malignant tumor of prostate"
 [4] "Secondary malignant neoplasm of prostate"
 [5] "Carcinoma of prostate"
 [6] "Metastasis from malignant tumor of prostate"
 [7] "Adenocarcinoma of prostate"
 [8] "Hormone refractory prostate cancer"
 [9] "Recurrent malignant neoplasm of prostate"
[10] "Primary malignant neoplasm of prostate"
```

```
condition_df_counts <- condition_df |>
    group_by(person_id) |>
    summarize(number_counts = n_distinct(condition_start_datetime))

case_definition1 <-
condition_df_counts[condition_df_counts$number_counts >=2, ]
nrow(case_definition1)
```

```
[1] 6180
```

Cases from Self-reported History Survey

```
library(tidyverse)
library(bigrquery)

# This query represents dataset "pc" for domain "survey" and was
generated for All of Us Controlled Tier Dataset v8
dataset_81293648_survey_sql <- paste("
    SELECT
        answer.person_id,
        answer.survey_datetime,
        answer.survey,
        answer.question_concept_id,
        answer.question,
        answer.answer_concept_id,
        answer.answer,
        answer.survey_version_concept_id,
        answer.survey_version_name
    FROM
        `ds_survey` answer
    WHERE
        (
            question_concept_id IN (836780)
        )
```

```
        AND (
            answer.PERSON_ID IN (SELECT
                distinct person_id
            FROM
                `cb_search_person` cb_search_person
            WHERE
                cb_search_person.person_id IN (SELECT
                    person_id
                FROM
                    `cb_search_person` p
                WHERE
                    has_whole_genome_variant = 1 )
                AND cb_search_person.person_id IN (SELECT
                    person_id
                FROM
                    `cb_search_person` p
                WHERE
                    has_ehr_data = 1 ) )
        )", sep="")

# Formulate a Cloud Storage destination path for the data exported from BigQuery.
# NOTE: By default data exported multiple times on the same day will overwrite older copies.
#       But data exported on a different days will write to a new location so that historical
#       copies can be kept as the dataset definition is changed.
survey_81293648_path <- file.path(
  Sys.getenv("WORKSPACE_BUCKET"),
  "bq_exports",
  Sys.getenv("OWNER_EMAIL"),
  strftime(lubridate::now(), "%Y%m%d"),  # Comment out this line if you want the export to always overwrite.
  "survey_81293648",
  "survey_81293648_*.csv")
message(str_glue('The data will be written to {survey_81293648_path}. Use this path when reading ',
                 'the data into your notebooks in the future.'))

# Perform the query and export the dataset to Cloud Storage as CSV files.
# NOTE: You only need to run `bq_table_save` once. After that, you can
#       just read data from the CSVs in Cloud Storage.
bq_table_save(
  bq_dataset_query(Sys.getenv("WORKSPACE_CDR"),
dataset_81293648_survey_sql, billing = Sys.getenv("GOOGLE_PROJECT")),
  survey_81293648_path,
  destination_format = "CSV")
```

```
# Read the data directly from Cloud Storage into memory.
# NOTE: Alternatively you can `gsutil -m cp {survey_81293648_path}` to
copy these files
#       to the Jupyter disk.
read_bq_export_from_workspace_bucket <- function(export_path) {
  col_types <- cols(survey = col_character(), question =
col_character(), answer = col_character(), survey_version_name =
col_character())
  bind_rows(
    map(system2('gsutil', args = c('ls', export_path), stdout = TRUE,
stderr = TRUE),
        function(csv) {
          message(str_glue('Loading {csv}.'))
          chunk <- read_csv(pipe(str_glue('gsutil cat {csv}')),
col_types = col_types, show_col_types = FALSE)
          if (is.null(col_types)) {
            col_types <- spec(chunk)
          }
          chunk
        }))
}
survey_df <-
read_bq_export_from_workspace_bucket(survey_81293648_path)

dim(survey_df)
```

```
The data will be written to gs://fc-secure-672eeb92-4859-4ed9-9f59-
d4349f3534a0/bq_exports/micah_hysong@researchallofus.org/20250825/
survey_81293648/survey_81293648_*.csv. Use this path when reading the
data into your notebooks in the future.
```

```
pc<-survey_df[survey_df$answer == "Including yourself, who in your
family has had prostate cancer? - Self",]
case_definition2<-unique(pc$person_id)
length(case_definition2)
```

All Cases

```
nrow(case_definition1)
length(unique(case_definition1$person_id))
length(case_definition2)
full_list<-c(case_definition1$person_id, case_definition2)
cases<-unique(full_list)
length(cases)
```

Controls

```
library(tidyverse)
library(bigrquery)

# This query represents dataset "prostatectomy" for domain "procedure"
and was generated for All of Us Controlled Tier Dataset v7
dataset_47173302_procedure_sql <- paste("
    SELECT
        procedure.person_id,
        procedure.procedure_concept_id,
        p_standard_concept.concept_name as standard_concept_name,
        p_standard_concept.concept_code as standard_concept_code,
        p_standard_concept.vocabulary_id as standard_vocabulary
    FROM
        ( SELECT
            *
        FROM
            `procedure_occurrence` procedure
        WHERE
            (
                procedure_concept_id IN (SELECT
                    DISTINCT c.concept_id
                FROM
                    `cb_criteria` c
                JOIN
                    (SELECT
                        CAST(cr.id as string) AS id
                    FROM
                        `cb_criteria` cr
                    WHERE
                        concept_id IN (4073700, 4211496, 4235738)

                        AND full_text LIKE '%_rank1]%'      ) a
                        ON (c.path LIKE CONCAT('%.', a.id, '.%')
                        OR c.path LIKE CONCAT('%.', a.id)
                        OR c.path LIKE CONCAT(a.id, '.%')
                        OR c.path = a.id)
                WHERE
                    is_standard = 1
                    AND is_selectable = 1)
            )
            AND (
                procedure.PERSON_ID IN (SELECT
                    distinct person_id
                FROM
                    `cb_search_person` cb_search_person
                WHERE
                    cb_search_person.person_id IN (SELECT
```

```
                        person_id
                    FROM
                        `cb_search_person` p
                    WHERE
                        has_whole_genome_variant = 1 )
                    AND cb_search_person.person_id IN (SELECT
                        person_id
                    FROM
                        `cb_search_person` p
                    WHERE
                        has_ehr_data = 1 ) )
            )) procedure
    LEFT JOIN
        `concept` p_standard_concept
            ON procedure.procedure_concept_id =
p_standard_concept.concept_id", sep="")

# Formulate a Cloud Storage destination path for the data exported
from BigQuery.
# NOTE: By default data exported multiple times on the same day will
overwrite older copies.
#       But data exported on a different days will write to a new
location so that historical
#       copies can be kept as the dataset definition is changed.
procedure_47173302_path <- file.path(
  Sys.getenv("WORKSPACE_BUCKET"),
  "bq_exports",
  Sys.getenv("OWNER_EMAIL"),
  strftime(lubridate::now(), "%Y%m%d"),  # Comment out this line if
you want the export to always overwrite.
  "procedure_47173302",
  "procedure_47173302_*.csv")
message(str_glue('The data will be written to
{procedure_47173302_path}. Use this path when reading ',
                 'the data into your notebooks in the future.'))

# Perform the query and export the dataset to Cloud Storage as CSV
files.
# NOTE: You only need to run `bq_table_save` once. After that, you can
#       just read data from the CSVs in Cloud Storage.
bq_table_save(
  bq_dataset_query(Sys.getenv("WORKSPACE_CDR"),
dataset_47173302_procedure_sql, billing =
Sys.getenv("GOOGLE_PROJECT")),
  procedure_47173302_path,
  destination_format = "CSV")


# Read the data directly from Cloud Storage into memory.
# NOTE: Alternatively you can `gsutil -m cp {procedure_47173302_path}`
```

```
to copy these files
#       to the Jupyter disk.
read_bq_export_from_workspace_bucket <- function(export_path) {
  col_types <- cols(standard_concept_name = col_character(),
standard_concept_code = col_character(), standard_vocabulary =
col_character())
  bind_rows(
    map(system2('gsutil', args = c('ls', export_path), stdout = TRUE,
stderr = TRUE),
        function(csv) {
          message(str_glue('Loading {csv}.'))
          chunk <- read_csv(pipe(str_glue('gsutil cat {csv}')),
col_types = col_types, show_col_types = FALSE)
          if (is.null(col_types)) {
            col_types <- spec(chunk)
          }
          chunk
        }))
}
procedure_df <-
read_bq_export_from_workspace_bucket(procedure_47173302_path)

exclude1<-unique(procedure_df$person_id)
length(exclude1)

library(tidyverse)
library(bigrquery)

# This query represents dataset "prostate_psa" for domain
"measurement" and was generated for All of Us Controlled Tier Dataset
v7
dataset_05885865_measurement_sql <- paste("
    SELECT
        measurement.person_id,
        measurement.measurement_concept_id,
        m_standard_concept.concept_name as standard_concept_name,
        m_standard_concept.concept_code as standard_concept_code,
        m_standard_concept.vocabulary_id as standard_vocabulary
    FROM
        ( SELECT
            *
        FROM
            `measurement` measurement
        WHERE
            (
                measurement_concept_id IN (SELECT
                    DISTINCT c.concept_id
                FROM
                    `cb_criteria` c
                JOIN
```

```
                    (SELECT
                        CAST(cr.id as string) AS id
                    FROM
                        `cb_criteria` cr
                    WHERE
                        concept_id IN (40779420, 4272032)
                        AND full_text LIKE '%_rank1]%'      ) a
                        ON (c.path LIKE CONCAT('%.', a.id, '.%')
                        OR c.path LIKE CONCAT('%.', a.id)
                        OR c.path LIKE CONCAT(a.id, '.%')
                        OR c.path = a.id)
                WHERE
                    is_standard = 1
                    AND is_selectable = 1)
            )
            AND (
                measurement.PERSON_ID IN (SELECT
                    distinct person_id
                FROM
                    `cb_search_person` cb_search_person
                WHERE
                    cb_search_person.person_id IN (SELECT
                        person_id
                    FROM
                        `cb_search_person` p
                    WHERE
                        has_whole_genome_variant = 1 )
                    AND cb_search_person.person_id IN (SELECT
                        person_id
                    FROM
                        `cb_search_person` p
                    WHERE
                        has_ehr_data = 1 ) )
            )) measurement
    LEFT JOIN
        `concept` m_standard_concept
            ON measurement.measurement_concept_id =
m_standard_concept.concept_id", sep="")

# Formulate a Cloud Storage destination path for the data exported
from BigQuery.
# NOTE: By default data exported multiple times on the same day will
overwrite older copies.
#       But data exported on a different days will write to a new
location so that historical
#       copies can be kept as the dataset definition is changed.
measurement_05885865_path <- file.path(
  Sys.getenv("WORKSPACE_BUCKET"),
  "bq_exports",
```

```
  Sys.getenv("OWNER_EMAIL"),
  strftime(lubridate::now(), "%Y%m%d"),  # Comment out this line if
you want the export to always overwrite.
  "measurement_05885865",
  "measurement_05885865_*.csv")
message(str_glue('The data will be written to
{measurement_05885865_path}. Use this path when reading ',
                 'the data into your notebooks in the future.'))

# Perform the query and export the dataset to Cloud Storage as CSV
files.
# NOTE: You only need to run `bq_table_save` once. After that, you can
#       just read data from the CSVs in Cloud Storage.
bq_table_save(
  bq_dataset_query(Sys.getenv("WORKSPACE_CDR"),
dataset_05885865_measurement_sql, billing =
Sys.getenv("GOOGLE_PROJECT")),
  measurement_05885865_path,
  destination_format = "CSV")


# Read the data directly from Cloud Storage into memory.
# NOTE: Alternatively you can `gsutil -m cp
{measurement_05885865_path}` to copy these files
#       to the Jupyter disk.
read_bq_export_from_workspace_bucket <- function(export_path) {
  col_types <- cols(standard_concept_name = col_character(),
standard_concept_code = col_character(), standard_vocabulary =
col_character())
  bind_rows(
    map(system2('gsutil', args = c('ls', export_path), stdout = TRUE,
stderr = TRUE),
        function(csv) {
          message(str_glue('Loading {csv}.'))
          chunk <- read_csv(pipe(str_glue('gsutil cat {csv}')),
col_types = col_types, show_col_types = FALSE)
          if (is.null(col_types)) {
            col_types <- spec(chunk)
          }
          chunk
        }))
}
has_psa <-
read_bq_export_from_workspace_bucket(measurement_05885865_path)

dim(has_psa)
length(unique(has_psa$person_id))
```

```r
controls<-setdiff(has_psa$person_id, exclude1)
controls<-setdiff(controls, cases)
length(controls)

df_cases <- data.frame(
  person_id = cases,
  status = 1
)

df_controls <- data.frame(
  person_id = controls,
  status = 0
)

final_df <- rbind(df_cases, df_controls)
nrow(final_df)
n_distinct(final_df$person_id)
```

Remove cis_females

```r
# This snippet assumes that you run setup first

# This code copies a file from your Google Bucket into a dataframe

# replace 'test.csv' with the name of the file in your google bucket
(don't delete the quotation marks)
name_of_file_in_bucket <- 'Demographic_and_ancestry_covariates.csv'

################################################################################
##
##
################## DON'T CHANGE FROM HERE
################################
##
################################################################################
##

# Get the bucket name
my_bucket <- Sys.getenv('WORKSPACE_BUCKET')

# Copy the file from current workspace to the bucket
system(paste0("gsutil cp ", my_bucket, "/data/",
name_of_file_in_bucket, " ."), intern=T)

# Load the file into a dataframe
demographics  <- read_csv(name_of_file_in_bucket)

cis_female_ids <- demographics %>%
  filter(SexGender == "Cis_female") %>%
  select(person_id)
```

```r
final_df <- final_df[!(final_df$person_id %in%
cis_female_ids$person_id), ]
n_distinct(final_df$person_id)

# This snippet assumes that you run setup first

# This code saves your dataframe into a csv file in a "data" folder in
Google Bucket

# Replace df with THE NAME OF YOUR DATAFRAME
my_dataframe <- final_df

# Replace 'test.csv' with THE NAME of the file you're going to store
in the bucket (don't delete the quotation marks)
destination_filename <- 'eMERGE_prostate_cancer_case_control_df.csv'

################################################################################
##
##
################## DON'T CHANGE FROM HERE
################################
##
################################################################################
##

# store the dataframe in current workspace
write_excel_csv(my_dataframe, destination_filename)

# Get the bucket name
my_bucket <- Sys.getenv('WORKSPACE_BUCKET')

# Copy the file from current workspace to the bucket
system(paste0("gsutil cp ./", destination_filename, " ", my_bucket,
"/data/"), intern=T)

# Check if file is in the bucket
system(paste0("gsutil ls ", my_bucket, "/data/*.csv"), intern=T)
```