Case Definition 1: At least one occurrence of the breast cancer diagnostic codes

```
library(tidyverse)
library(bigrquery)

# This query represents dataset "breast_cancer_condition" for domain
"condition" and was generated for All of Us Controlled Tier Dataset v7
dataset_57829836_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
    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 (4091464, 4091469, 4112853,
4162253, 81250)
                        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_57829836_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_57829836",
  "condition_57829836_*.csv")
message(str_glue('The data will be written to
{condition_57829836_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_57829836_condition_sql, billing =
Sys.getenv("GOOGLE_PROJECT")),
  condition_57829836_path,
  destination_format = "CSV")


# Read the data directly from Cloud Storage into memory.
# NOTE: Alternatively you can `gsutil -m cp {condition_57829836_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_57829836_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_57829836/condition_57829836_*.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_57829836/
```

```
condition_57829836_000000000000.csv.

[1] 864695      5
```

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

```
 [1] "Intraductal carcinoma in situ of breast"
 [2] "Malignant neoplasm of male breast"
 [3] "Primary malignant neoplasm of female right breast"
 [4] "Carcinoma of female breast"
 [5] "Malignant neoplasm of upper-inner quadrant of female breast"
 [6] "Malignant neoplasm of axillary tail of female breast"
 [7] "Malignant neoplasm of nipple and areola of male breast"
 [8] "Infiltrating lobular carcinoma of left female breast"
 [9] "Primary malignant neoplasm of central portion of female breast"
[10] "Secondary malignant neoplasm of breast"
[11] "Infiltrating duct carcinoma of left female breast"
[12] "Hereditary breast and ovarian cancer syndrome"
[13] "Carcinoma of breast - lower, inner quadrant"
[14] "Primary malignant neoplasm of male breast"
[15] "Carcinoma in situ of female breast"
[16] "Carcinoma in situ of right breast"
[17] "Hormone receptor positive malignant neoplasm of breast"
[18] "Inflammatory carcinoma of breast"
[19] "Primary malignant neoplasm of female left breast"
[20] "Malignant phyllodes tumor of breast"
[21] "Intraductal carcinoma in situ of bilateral breasts"
[22] "Malignant neoplasm of axillary tail of breast"
[23] "Malignant tumor of breast"
```

```
[24] "Primary malignant neoplasm of areola of female breast"

[25] "Carcinoma of breast - upper, outer quadrant"

[26] "Primary malignant neoplasm of breast lower outer quadrant"

[27] "Papillary carcinoma in situ of breast"

[28] "Carcinoma in situ of left breast"

[29] "Overlapping malignant neoplasm of male breast"

[30] "Secondary malignant neoplasm of female breast"

[31] "Malignant neoplasm of lower-outer quadrant of female breast"

[32] "Infiltrating duct carcinoma of breast"

[33] "Primary malignant neoplasm of axillary tail of right female
breast"
[34] "Carcinoma of male breast"

[35] "Carcinoma of central portion of breast"

[36] "Primary malignant neoplasm of breast lower inner quadrant"

[37] "Malignant neoplasm of breast upper outer quadrant"

[38] "Primary malignant neoplasm of axillary tail of left female
breast"
[39] "Triple-negative breast cancer"

[40] "Infiltrating duct carcinoma of right female breast"

[41] "Primary malignant neoplasm of breast upper inner quadrant"

[42] "Invasive carcinoma of breast"

[43] "Malignant neoplasm of breast upper inner quadrant"

[44] "Carcinoma of breast"

[45] "Lobular carcinoma in situ of breast"

[46] "Primary malignant neoplasm of skin of breast"

[47] "Recurrent primary malignant neoplasm of left female breast"

[48] "Intraductal carcinoma in situ of right breast"
```

```
[49] "Infiltrating lobular carcinoma of right female breast"

[50] "Primary malignant neoplasm of lower inner quadrant of female
breast"
[51] "Metastatic human epidermal growth factor 2 positive carcinoma of
breast"
[52] "Local recurrence of malignant tumor of breast"

[53] "Primary malignant neoplasm of breast"

[54] "Infiltrating lobular carcinoma of breast"

[55] "Carcinoma in situ of breast"

[56] "Malignant neoplasm of lower-inner quadrant of female breast"

[57] "Malignant neoplasm of female breast"

[58] "Malignant neoplasm of central part of female breast"

[59] "Paget's disease of nipple"

[60] "Human epidermal growth factor 2 negative carcinoma of breast"

[61] "Intraductal carcinoma in situ of left breast"

[62] "Infiltrating ductal carcinoma of upper outer quadrant of right
female breast"
[63] "Malignant lymphoma of breast"

[64] "Overlapping malignant neoplasm of female breast"

[65] "Lobular carcinoma in situ of left breast"

[66] "Malignant neoplasm, overlapping lesion of breast"

[67] "Malignant melanoma of skin of breast"

[68] "Primary malignant neoplasm of lower outer quadrant of female
breast"
[69] "Mucinous carcinoma of breast"

[70] "HER2-positive carcinoma of breast"

[71] "Malignant neoplasm of upper-outer quadrant of female breast"

[72] "Primary malignant neoplasm of upper outer quadrant of female
breast"
[73] "Lobular carcinoma in situ of right breast"

[74] "Primary malignant neoplasm of breast upper outer quadrant"
```

```
[75] "Primary malignant neoplasm of axillary tail of breast"

[76] "Primary malignant neoplasm of upper inner quadrant of female
breast"
[77] "Malignant neoplasm of nipple and areola of female breast"

[78] "Primary malignant neoplasm of female breast"

[79] "Adenocarcinoma of breast"

[80] "Infiltrating duct carcinoma of female breast"

case_definition1<-unique(condition_df$person_id)
length(case_definition1)

[1] 10697
```

Case Definition 2: Personal Health History Survey Indication

```
library(tidyverse)
library(bigrquery)

# This query represents dataset "bc" for domain "survey" and was
generated for All of Us Controlled Tier Dataset v8
dataset_28116944_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 (836772)
        )
        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_28116944_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_28116944",
  "survey_28116944_*.csv")
message(str_glue('The data will be written to {survey_28116944_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_28116944_survey_sql, billing = Sys.getenv("GOOGLE_PROJECT")),
  survey_28116944_path,
  destination_format = "CSV")


# Read the data directly from Cloud Storage into memory.
# NOTE: Alternatively you can `gsutil -m cp {survey_28116944_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_28116944_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_28116944/survey_28116944_*.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/survey_28116944/
survey_28116944_000000000000.csv.
```

```
[1] 104257      9
```

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

```
[1] 6778
```

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

```
[1] 10697
```

```
[1] 6778
```

```
[1] 12447
```

Get Controls

```
# 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)
```

```
character(0)
```

```
table(demographics$SexGender[demographics$person_id %in% cases])

controls <- demographics$person_id[!(demographics$person_id %in%
cases)]
table(demographics$SexGender[demographics$person_id %in% controls])
length(controls)

#Control for Sex/Gender
# Filter the demographic dataframe to remove rows where SexGender is
"Cis_male"
non_male_ids <- demographics %>%
  filter(SexGender != "Cis_male") %>%
  select(person_id)

# Filter the cases dataframe by retaining only rows with person_id in
cis_woman_ids
```

```r
cases <- cases[cases %in% non_male_ids$person_id]
length(cases)

# Filter the controls dataframe by retaining only rows with person_id
in cis_woman_ids
controls <- controls[controls %in% non_male_ids$person_id]
length(controls)

library(tidyverse)
library(bigrquery)

# This query represents dataset "mastectomy" for domain "procedure"
and was generated for All of Us Controlled Tier Dataset v8
dataset_81554391_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,
        procedure.procedure_datetime,
        procedure.procedure_type_concept_id,
        p_type.concept_name as procedure_type_concept_name,
        procedure.modifier_concept_id,
        p_modifier.concept_name as modifier_concept_name,
        procedure.quantity,
        procedure.visit_occurrence_id,
        p_visit.concept_name as visit_occurrence_concept_name,
        procedure.procedure_source_value,
        procedure.procedure_source_concept_id,
        p_source_concept.concept_name as source_concept_name,
        p_source_concept.concept_code as source_concept_code,
        p_source_concept.vocabulary_id as source_vocabulary,
        procedure.modifier_source_value
    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 (4286804)
                        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
    LEFT JOIN
        `concept` p_type
            ON procedure.procedure_type_concept_id = p_type.concept_id

    LEFT JOIN
        `concept` p_modifier
            ON procedure.modifier_concept_id = p_modifier.concept_id
    LEFT JOIN
        `visit_occurrence` v
            ON procedure.visit_occurrence_id = v.visit_occurrence_id
    LEFT JOIN
        `concept` p_visit
            ON v.visit_concept_id = p_visit.concept_id
    LEFT JOIN
        `concept` p_source_concept
```

```
            ON procedure.procedure_source_concept_id =
p_source_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_81554391_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_81554391",
  "procedure_81554391_*.csv")
message(str_glue('The data will be written to
{procedure_81554391_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_81554391_procedure_sql, billing =
Sys.getenv("GOOGLE_PROJECT")),
  procedure_81554391_path,
  destination_format = "CSV")


# Read the data directly from Cloud Storage into memory.
# NOTE: Alternatively you can `gsutil -m cp {procedure_81554391_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(), procedure_type_concept_name = col_character(),
modifier_concept_name = col_character(), visit_occurrence_concept_name
= col_character(), procedure_source_value = col_character(),
source_concept_name = col_character(), source_concept_code =
col_character(), source_vocabulary = col_character(),
modifier_source_value = 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_81554391_path)

unique(procedure_df$standard_concept_name)

head(procedure_df, 5)

length(controls)
controls <- controls[!controls %in% procedure_df$person_id]
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)

# 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_breast_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)
```