```
############################################################

--  ER Analysis Assignment SQL Code

--  Author: Jason Massey (JM)

--  Last Edited: 3/11/2025

############################################################

##############################

#   Part I - Data Cleaning

##############################

#####################################################################
# Step 1: In explorer bar, create dataset "well_data" to store all data tables
#####################################################################

#####################################################################
# Step 2: Clean Demography Data
#####################################################################

# Person table has N = 2326856 rows
# Join concept to person as table (demog, N = 2326856)
```

```sql
CREATE TABLE well_data.demog AS
SELECT concept_id, concept_name, gender_concept_id, person_id, year_of_birth
FROM bigquery-public-data.cms_synthetic_patient_data_omop.concept as x
INNER JOIN bigquery-public-data.cms_synthetic_patient_data_omop.person as y
ON x.concept_id = y.gender_concept_id;

# Create new table (demog2) adding age at exam column and removing rows with missing
conceptID info
CREATE TABLE well_data.demog2 AS
SELECT concept_id, concept_name, gender_concept_id, person_id, year_of_birth,
(2008 - year_of_birth) AS age_at_exam
FROM well_data.demog as x
WHERE concept_id != 0;

# Create new table (demog3) adding age category column by age quartiles
CREATE TABLE well_data.demog3 AS
WITH Quartiles AS (
  SELECT APPROX_QUANTILES(age_at_exam, 4) AS age_quartiles
  FROM well_data.demog2)
SELECT *,
CASE WHEN age_at_exam <= (
    SELECT age_quartiles[OFFSET(1)]
    FROM Quartiles ) THEN 'Q1'
  WHEN age_at_exam > (
    SELECT age_quartiles[OFFSET(1)]
    FROM Quartiles )
  AND age_at_exam <= (
    SELECT age_quartiles[OFFSET(2)]
    FROM Quartiles ) THEN 'Q2'
  WHEN age_at_exam > (
    SELECT age_quartiles[OFFSET(2)]
    FROM Quartiles )
  AND age_at_exam <= (
    SELECT age_quartiles[OFFSET(3)]
    FROM Quartiles ) THEN 'Q3'
  ELSE 'Q4'
```

```
END AS age_quartile
FROM well_data.demog2;


#####################################################################
# Step 3: Clean Procedure Data
#####################################################################

# Procedure_occurence table has N = 278769529 rows
# Join concept to procedure_occurrence as new table (proc) N = 278769529
CREATE TABLE well_data.proc AS
SELECT concept_id, domain_id, vocabulary_id, concept_code, procedure_concept_id,
person_id, procedure_dat,
visit_occurrence_id, procedure_occurrence_id, quantity
FROM bigquery-public-data.cms_synthetic_patient_data_omop.concept  as x
INNER JOIN bigquery-public-data.cms_synthetic_patient_data_omop.procedure_occurrence
as y
ON x.concept_id = y.procedure_concept_id;

# Looking to see the duplicates of person_id (278743124)
# (Total - duplicates) = (278769529 - 278743124) = 26,405 patients with only 1
procedure
SELECT * FROM (
SELECT *, COUNT(1) OVER(PARTITION BY person_id) dup_count
FROM well_data.proc )
WHERE dup_count > 1 ;

# Looking to see the duplicates of procedure_occurence_id (0 - most granular)
SELECT * FROM (
SELECT *, COUNT(1) OVER(PARTITION BY procedure_occurrence_id) dup_count
FROM well_data.proc )
WHERE dup_count > 1 ;

# (1) Create new table (proc2) with indicator columns "exam" and (2) "ER"
# (3) Extract year from date
# (4) Remove rows with missing conceptid info
```

```
CREATE TABLE well_data.proc2 AS
SELECT concept_id, domain_id, vocabulary_id, concept_code, procedure_concept_id,
person_id, procedure_dat,
      visit_occurrence_id, procedure_occurrence_id, quantity,
CASE WHEN domain_id = "Procedure" AND vocabulary_id = "CPT4" AND concept_code  IN
("99385", "99386", "99387", "99395","99396", "99397") THEN 1 ELSE 0 END
  AS exam,
CASE WHEN domain_id = "Procedure" AND vocabulary_id = "CPT4" AND concept_code BETWEEN
"99281" AND "99285" THEN 1 ELSE 0 END
  AS ER,
EXTRACT(YEAR FROM procedure_dat ) AS year #(3)
FROM well_data.proc
WHERE concept_id != 0;

# Create a new table (proc3) that filters the analysis to the years with values only
in 2008 or 2009. I also filtered this table to include patients who had at least one
exam coded as 1 per patient_id in 2008 as 1 for all of their rows.
CREATE TABLE well_data.proc3 AS
SELECT person_id, exam, ER, year,
 CASE WHEN EXISTS (
     SELECT 1
     FROM well_data.proc2 AS t2
     WHERE t2.person_id = t1.person_id AND t2.exam = 1 ) THEN 1 ELSE 0 END
     AS exam_yesno
FROM well_data.proc2 AS t1
WHERE year IN (2008, 2009) ;

# (1) Create new table (proc4) by transforming from long to wide format
# (2) Create columns per patient: collapse >=1 exams in 2008 and sum total ER visits
in 2009
CREATE TABLE
well_data.proc4 AS
SELECT person_id,
SUM(exam) AS exam_total,
ANY_VALUE(exam_yesno) AS exam_yesno,
SUM(ER) AS ER_total
```

```
FROM well_data.proc3
GROUP BY person_id ;


#####################################################################
# Step 4: Create Analytic Table
#####################################################################

# (1) Transform both demog2 and (2) plan period from long to wide as
deduplicated_tables
# (3) Left join both to proc4 to create final analytic table (analysis)
#  NOTE: Gender, age, and insurance dates don't vary by patient; therefore use
any_value() to collapse
#  NOTE: payer_plan_period had many concept_id variables that would be likely be
included in further analyses
CREATE TABLE well_data.analysis AS WITH
deduplicated_table2 AS (
  SELECT person_id,
    ANY_VALUE(concept_name) AS gender,
    ANY_VALUE(age_at_exam) AS age_at_exam,
    ANY_VALUE(age_quartile) AS age_quartile
  FROM well_data.demog3
  GROUP BY person_id),
deduplicated_table3 AS (  #(2)
  SELECT person_id,
    ANY_VALUE(payer_plan_period_start_date) AS insurance_begins,
    ANY_VALUE(payer_plan_period_end_date) AS insurance_ends
  FROM bigquery-public-data.cms_synthetic_patient_data_omop.payer_plan_period
  GROUP BY person_id)
SELECT x.person_id, exam_yesno, exam_total, ER_total, gender, age_at_exam,
age_quartile, insurance_begins, insurance_ends  #(3)
FROM well_data.proc4 as x
LEFT JOIN deduplicated_table2 as y ON x.person_id = y.person_id
LEFT JOIN deduplicated_table3 as z ON x.person_id = z.person_id ;

# N = 1903990 rows/patients in analysis
```

```
#####################################

#   Part II - Descriptive Questions

#####################################


################
#  Question 1  #
################

# Calculate the average ER total by those who had a preventative exam vs. those who
did not
WITH MeanData AS (
SELECT exam_yesno,
COUNT(*) AS N,
  SUM(ER_total) AS ER_total_cat,
  AVG(ER_total) AS mean_ER_total
FROM well_data.analysis
GROUP BY exam_yesno )
SELECT exam_yesno, N, ER_total_cat, mean_ER_total
FROM MeanData
ORDER BY exam_yesno;

# List of each category #exams (1,2,3,4, 5) by mean #ER visits
WITH MeanData AS (
SELECT exam_total,
COUNT(*) AS N,
  SUM(ER_total) AS ER_total_cat,
  AVG(ER_total) AS mean_ER_total
FROM well_data.analysis
GROUP BY exam_total )
SELECT exam_total, N, ER_total_cat, mean_ER_total
```

```
FROM MeanData
ORDER BY exam_total;

#SOLUTION:
# For looking at those who get exams vs. those who do not no -  in fact the average number of total ER visits per person is slightly higher in those who received at least one preventative exam than those who did not.
# Looking among those who get exams we do see a slight gradual decrease in ER visits as the number of preventative exams goes up; however some of the power is low with small data.

################
#  Question 2  #
################

#  Stratify Q1 list by gender M/F
SELECT gender, exam_yesno,
COUNT(*) AS N,
AVG(ER_total) AS mean_ER_total
FROM well_data.analysis
GROUP BY gender, exam_yesno
ORDER BY gender, exam_yesno;

SELECT gender, exam_total,
COUNT(*) AS N,
AVG(ER_total) AS mean_ER_total
FROM well_data.analysis
GROUP BY gender, exam_total
ORDER BY gender, exam_total;

#  Stratify Q1 list by age categories
SELECT age_quartile, exam_yesno,
COUNT(*) AS N,
AVG(ER_total) AS mean_ER_total
FROM well_data.analysis
GROUP BY age_quartile, exam_yesno
```

```sql
ORDER BY age_quartile, exam_yesno;

SELECT age_quartile, exam_total,
COUNT(*) AS N,
AVG(ER_total) AS mean_ER_total
FROM well_data.analysis
GROUP BY age_quartile, exam_total
ORDER BY age_quartile, exam_total;

# SOLUTION:
# No major changes in preventative exams when stratified by sex or age for 0 vs 1
# Among those who get exams, no major changes when stratified by sex or age
# There are higher ER visits overall in females
# There are higher ER visits in the more elderly as to be expected
```