Configuring the RCHC Relevant QIP ECDS Reports

(eCW Health Centers, Version 1)

Author: Ben Fouts, Informatics
Redwood Community Health Coalition
1310 Redwood Way, Petaluma, California 94954
support@rchc.net
Document Last Updated: 7/27/2022 4:55 PM

Configuring the RCHC Relevant QIP ECDS Reports

Table of Contents

Configuring the RCHC Relevant QIP ECDS Reports

Introduction

Project Overview

Partnership Healthplan is moving towards a model where health centers securely submit detailed patient data, which is then combined with administrative and other data sources Partnership possesses to more accurately evaluate the HEDIS measures reported to the State of California. This is referred to as Electronic Medical Record Data Exchange (ECDS) and Partnership is beginning with a number of measures that are designated by HEDS as electronic measures. There are four groups of measures that correspond to four Relevant reports that will be used to submit four data files to Partnership. These are:

1. Breast Cancer Screening (one measure)
2. Follow-Up Care for Children Prescribed ADHD Medication (one measure, with two aspects)
3. Depression Related Measures (five measures)
4. Unhealthy Alcohol Use Screening and Follow Up (one measure, with two aspects)

RCHC is developing a set of ECDS Quality Measures in Relevant that can be used to track the electronic measures based on data from the health center EHR. In contrast, the data submission reports are designed to suppliment data that Partnership already has from claims, pharmacies, outside screening providers, etc. The Relevant data reports themselves cannot be used to define the denominator or numerator of the Quality Measures.

Partnership will pay a $5,000 incentive to health centers that submit a set of test files and a set of final files for the 2022 calendar year. The deadlines and other details are described in the Supporting Documents below.

Supporting Documents

This project was developed in collaboration with Partnership Healthplan of California. Therefore, the basic documents come from their website. As of the writing of these instructions, the link to the instroduction presentation slides and recording are on the Partnership Perinatal QIP webpage[1]. There are screenshots and additional detail in the 7/2022 RCHC Workgroup presentation described below.

[1] http://www.partnershiphp.org/Providers/Quality/Pages/Perinatal-QIP.aspx

Configuring the RCHC Relevant QIP ECDS Reports

The are important Partnership documents and templates available through the eReports Portal (for PCP QIP). Health centers are strongly encouraged to download these materials to get a better understanding of the project and how to submit the data. These materials include:

1. The measure requirements (i.e., the standards and specifications)
2. The Value Set lists (for your information)
3. Four reporting templates (data will be copied to these and sent securely to Partnership)
4. A document containing SQL code examples for both eCW and NextGen. Even though RCHC health centers can adapt this SQL for their own reports, the SQL code specific to Relevant described below is better suited for this project.

On the RCHC Website (under Initiatives \ Population Health \ Data Analytics and Governance \ Program \ Data Workgroup and Report Documentation \ Additional Resources and Companion Documents \ Partnership ECDS Reporting [2]), there are two sets of slides and webinars:

1. The first set (both files marked 3/2022) is a project overview and description of the ECDS measures themselves. This presentation is named "Partnership Electronic Clinical Data Systems (ECDS) Measures."
2. The second set (both files marked 7/2022) describes the ECDS reports and is more relevant to the instructions described in sections below. This presentation is named "New ECDS Measures: Focus on Reports."

The 2022 RCHC Analytics Academy featured a presentation on how to copy reports from the RCHC Instance of Relevant and adapt them to the health center's instance of Relevant. This training was called "Can't Wait for the SQL" and featured slides, handouts and a webinar recording. These materials can also be found on the RCHC website (under Initiatives \ Population Health \ Data Analytics and Governance \ RCHC Analytics Academy \ 2022 Analytics Academy [3]).

[2] https://www.rchc.net/population-health/data-analytics-and-governance/#toggle-id-2
[3] https://www.rchc.net/population-health/data-analytics-and-governance/#toggle-id-3

Measurement Period

To keep things simple and straightforward, all four reports will use the same measurement period for both the test file and the final file: 1/1/2022 to 12/31/2022. The report SQL contains parameters, so the user will be prompted to enter a measurement period date range. The reports automatically make calculations for look-back periods. Note that this is a different approach than described in the general Partnership specifications and SQL templates.

The measurement period is defined on the report by the parameters {{start_date}} and {{end_date}}.

Partnership Patients

Data submission must include Partnership Managed Care patients only. The same approach to identifying these patients should be used in the SQL code of all four reports. The universe of Partnership patients (along with other aspects of the denominator) is identified in the universe TEMPORARY TABLE of the report SQL code.

The SQL default code for the reports identifies Partnership patients based on an insurance group ID for Partnership Managed Care. This is written as **primary_insurance_group_id** = 17 in the WHERE clause of the universe TEMPORARY TABLE. If your health center is going to use this default, identify the unique insurance group ID for Partnership Managed Care patients and replace the default group ID[4].

If a Partnership Managed Care insurance group has not been defined in your EHR (or Relevant), another option is to use individual insurance identification numbers associated with Partnership Managed Care from the field patients.primary_insurance_id

If your health center has integrated monthly Partnership patient lists into Relevant, then add SQL code to JOIN these patients (or add a statement to the WHERE clause, etc., depending how they appear in your instance). With this approach, also remove the default SQL code in the WHERE statement for primary_insurance_group_id and the visit date range of the universe TEMPORARY TABLE.

Because the list of patients enrolled in MediCal Managed Care (through Partnership) changes regularly, the term “anchor date” referrs to the list of enrolled patients at a particular time. The test files are due on October 1, 2022, so use an anchor date of September 1, 2022. The final files are due January 28, 2023, so use an anchor date of December 1, 2022 (which is the same as the other QIP reports).

[4] You can use this SQL to look for the group number(s) at your health center: SELECT * FROM payer_groups

Submitting the Files

There are two data submissions. The measurement period for both submissions (entered into the parameters) is 1/1/2022 to 12/31/2022.

The first submission is a "test" file that Partnership will use to assess their own data integration systems. They may even provide feedback to health centers that can be used to modify the Relevant report. This file is due by October 1, 2022 and should be prepared some time in September 2022. If the health center has integrated current Partnership patient lists into Relevant, the September 1, 2022 patient list from eReports should be used as the "anchor."

The second submission is the "final" report and should be run after January 8, 2023 (leaving enough time for visits at the end of December 2022 to be closed and mammogram images reviewed and completed by providers). The final report must be submitted to Partnership before January 28, 2023. If the health center has integrated current Partnership patient lists into Relevant, the December 1, 2022 patient list from eReports should be used as the "anchor."

Make sure the data copied into the files follow the column formatting that is described in row 2 of the template. It should be possible to do this using the Paste Special function in Excel because the template columns are already formatted by default. NOTE that sometimes the CIN automatically converts to scientific notation when it contains numbers along with the letter E. Scan the list before submitting it to make sure no CIN in the column Member_Key appear in scientific notation.

Naming the Submission Templates

Once the data has been validated, the report data should be copied to the appropriate template. There are four templates corresponding to the four Relevant reports.

The blank submission templates have been given general names. When a service provider has entered data and is ready to submit a template, it should be re-named according to the standard on the next page. Utilizing a unique and descriptive report name enables Partnership to ingest the report into their data analysis system.

The following instructions come from the Partnership SQL template document. The introduction presentation (see the section Supporting Documents above) from June 2022 describes how the files should be sent to Partnership.

The file name should follow the convention MEASUREID_SITENAME_DATE where:

- **MEASUREID** corresponds to the measure abbreviation and is one of the following:

MEASUREID	Measure name
ADD	Follow-Up Care for Children Prescribed ADHD Medication
BCS	Breast Cancer Screening
ASF	Unhealthy Alcohol Use Screening and Follow-Up
DEP	Five depression measures (submitted as one file)

- **SITENAME** is the name of the health center. The health center should choose a brief and easily recognizable name or abbreviation and use it consistently with all data submissions

- **DATE** is the submission date in the format YYYYMMDD with no dashes, periods or slashes

Examples: ASF_SHASTA_20220930
ADD_PETALUMA_20230115

Appendices

The next four sections of this document describe the RCHC ECDS reports and what is needed to adapt the code to the health center's instance of Relevant. The appendices contain the first draft of the basic SQL code that can be copied to DataGrip or Relevant and customized there. Alternatively, health centers can copy the code directly from the reports section of the RCHC Instance of Relevant. Note that reports running on the Staging Database will not function in the RCHC Instance.

Health center programmers are encouraged to send feedback on the SQL code (comments on the approach, logic, expressions, etc.) to Ben Fouts (bfouts@rchc.net) so that the code can be improved during the validation process. Changes will be discussed on the RCHC Slack Population health channel and the report SQL code on the RCHC Instance updated, if necessary. Therefore, reports on the RCHC Instance reports are the best and most updated source of SQL code.

General Comments on the SQL Code

The references to SQL code in this document (including the appendices) is for health centers using Relevant and the eCW Electronic Health Record. NextGen SQL code will be developed separately, although the approach and many of the ideas are the same.

The report SQL code will need to be inspected and adapted to each health center's instance. This is especially true for the two reports that are based on the Staging Database. See the materials from the 2022 RCHC Analytics Academy mentioned above in the section Supporting Documents for an explanation on how this can be done by the SQL programmer. The two reports based on the Production Database will probably require less customization, although one of them requires new Data Elements.

The outline of the SQL code is generally the same for all four reports. Temporary tables are used to define the universe and different aspects of the measure, which are all brought together in the results query. The basic flow of the temporary tables begins with the patient universe and moves on to other aspects of the measure displayed in the columns. Then exclusions are evaluated and displayed in the required columns. The results query that displays the report columns in the same order and format as the reporting template is the final section of the code.

Where the descriptive codes[5] needed by the report follow exactly the HEDIS Value Set codes, a JOIN is made directly to the HEDIS Value Set table in Relevant (the table name is hedis_value_set_codes, which exists on both the Staging and Production Databases). On two of the four reports, the first TEMPORARY TABLE contains additional codes specified by Partnership or codes not included on the HEDIS table.

There are grey comments/notes embedded in the SQL that help explain the objective of the temporary tables or specific aspects of the code. The descriptions of each report below also direct the programmer to locations in the code that must be customized or otherwise scrutinized. It is unlikely that any of the reports besides the Breast Cancer Screening report will run immediately after copying the code to Relevant.

[5] Such as procedure (CPT) and diagnosis (ICD) codes

Description of ECDS Measure Report: Breast Cancer Screening (BCS-E)

This report is based on the Relevant Production Database and so no further mapping is needed. Health centers generally have all Data Elements already mapped for the related UDS and QIP Quality Measures.

Below are some comments about sections of the SQL code. These comments reference the specific temporary tables or the final results table contained in the code of the appendix.

TEMPORARY TABLE universe_temp_raw

See notes in the Introduction above on identifying patients with Partnership Insurance

TEMPORARY TABLE mammograms_temp

Since most health centers do not perform mammograms themselves, the CPT and HCPCS Value Set options are not used. Instead, a SNOMED code is displayed for all mammograms to signify that they come from structured data (i.e., diagnostic images).

Note on Exclusion TEMPORARY TABLES

Partnership wants to know if the patient has any exclusions and provides four columns for codes and dates. The report gathers the Value Set diagnosis codes from two locations in the medical record, prioritizes them (in case there are more than 4 codes at one time, which is possible but likely rare), and displays them in the columns. Both the diagnosis code and the associated date are displayed.

The locations and a description of the data are as follows:

1) Diagnosis codes associated with a visit (from Assessments). These are associated with visits during the 27 months before the end of the measurement period, which is the same period from which mammograms are extracted. In the Results Query, these are displayed by assiciation with a particular date of service (column Date_of_Service).
2) Diagnosis codes on the patient's Problem List. These are not associated with a particular visit and so no date is displayed in the column Date_of_Service. These codes can have a start date any time before the end of the measurement period.

It is possible (and even likely) that exclusion codes will be duplicated among rows displaying visits and Problem List results. The intent is for the report to cast a wide net to identify any exclusions. Partnership will determine if the patient should be excluded based on the diagnosis code and associated date.

Results Query

Three groups of records are added together using UNION statements. They are:

1) Mammograms in the 27 months before the and of the measurement period end date. These are associated with a date in the column Date_of_Service
2) Exclusions from Assessments associated with a date in the column Date_of_Service
3) Exclusions from the Problem List not associated with a date of service (the column Date_of_Service will be blank)

The facility where the visit was performed is displayed (in column Site_Name) when an exclusion is identified from an Assessment. In these rows, the Partnership Site ID number should be added to the report manually (to column Site_ID_Number) because it does not exist in Relevant. This can be done in the second subquery (most likely using a CASE WHEN statement depending on site name) or copied into the template directly. The Site_ID_Number is the NPI number assigned to the site where care was provided (i.e., the site in the column Site_Name). Since mammograms and Problem List exclusion codes are not associated with a particular visit to the health center, no Site Name or Site ID Number are needed for these rows.

Description of ECDS Measure Report: Follow-Up Care for Children Prescribed ADHD Medication (ADD-E)

This report is based on the Relevant Staging Database. Because this Database does not have a high degree of standardization among health centers, some of the SQL code will need to be adapted by the health center programmer using the unique table and field names in the health center instance as well as any "best practices" the health center has already developed to extract particular sets of data.

TEMPORARY TABLE temp_value_sets

This table provides codes that are not part of the HEDIS Value Sets but are referenced by Partnership. These consist of some additional visit codes and also the ADHD diagnosis codes.

TEMPORARY TABLE universe_temp_raw

See notes in the Introduction above on identifying patients with Partnership Insurance. Visits displayed in the universe must have a combination of a visit code (a code from the six HEDIS Value Sets and the additional Partnership codes) and an ADHD diagnosis code.

The visit measurement period is calculated. To keep things simple for all reports in the ECDS set, the measurement period entered by the user is 1/1/2022 to 12/31/2022. However, the report will make a calculation for the start date from which visits are extracted (the end date is the same as the end date entered by the user).

So, for example, if the measurement period entered by the user is 1/1/2022 to 12/31/2022, the visit span begins on 3/1/2021. The earliest visit extracted by the report corresponds to the start of the measure's "Intake Period" which is defined as "March 1 of the year prior to the Measurement Period" (the reference is the HEDIS specifications). The report can be run for any year-long measurement period (not just a proper calendar year), but the visit extraction period will be from 10 months before the start of the measurement period to the end of the measurement period.

Results Query

There are two ID numbers needed on this report that likely do not exist in your EHR. However, if they do exist in the EHR, the programmer can reference the field in the SQL code. If they do not exist, they can be added manually to the results in the Excel template or to the SQL code using a CASE WHEN statement.

These ID numbers are:

1) Field “Site_ID_Number.” This is the NPI number assigned to the site where care was provided (i.e., the site displayed in the field Site_Name)
2) Field "Provider_Key." This is the PCP ID assigned by Partnership to the provider who provided care during the visit (i.e., the provider displayed in Clinician_First_Name and Clinician_Last_Name)

Description of ECDS Measure Report: Depression Related Measures (DMS-E, DSF-E, DRR-E, PND-E, PDS-E)

Note that some health centers might already have this report in their instance of Relevant because it was part of a Partnership pilot program last year. If the data on that report has been thoroughly validated, the health center may continue to use it. The SQL code in Appendix C has been improved from the initial version and so programmers may consider using the new code or parts of the new code.

The depression measures report is built on the Relevant Staging Database. Because this Database does not have a high degree of standardization among health centers, some of the SQL code will need to be adapted by the health center programmer using the unique table and field names in the health center instance as well as any "best practices" the health center has already developed to extract particular sets of data.

This report first requires some configuration of Transformers. Although Data Elements are not used directly by the report, the related Quality Measures rely on the Data Elements. Therefore, Transformers and Data Elements will be described here together as pairs (where applicable) and should be configured together.

1. The report displays the actual numerical score of the PHQ-2 and PHQ-9 instruments. Depending how your Transformers are configured, you may already have tables with all PHQ-2 and PHQ-9 dates and scores. The sample report SQL in the appendix (see TEMPORARY TABLE depression_screenings_temp) references two tables that were configured at one of the health centers[6]. Note that the common Transformer relevant_depression_screens may or may not display the raw score and may or may not contain/distinguish between PHQ-2 and PHQ-9 instruments. The numerical screening score is not needed for the Quality Measure (and is therefore not added to any Data Element).

2. Identify other depression screening instruments used by your health center. There is a list of acceptable instruments in the Partnership measure standards document. You may need to communicate with the medical officer or behavioral health department lead at your health center for help. The report RCHC List All Structured Data Items (for eCW users) can be used to find items in structured data (most commonly HPI) where depression screening scores are entered. This report also shows category, symptom and detail ID numbers which are unique to the health center. The presentation with the title "New ECDS Measures: Focus on Reports"

[6] The sample SQL code was designed in the Santa Rosa instance. The Transformers are named relevant_phq2 and relevant_phq9

mentioned in the section Supporting Documents (above, at the beginning of this document) discusses an approach to finding these screens in structured data.

A. The most common non-PHQ screening instrument is the Edinburgh Depression Screen, which is used with prenatal and postnatal patients (two of the related depression Quality Measures focus on these populations). There is a new Data Element (Edinburgh Depression Screens) that can be configured and enabled if your health center uses this instrument. Example SQL code for the Transformer/Data Element Pair is displayed in Appendix C.

B. Health centers may use additional depression screening instruments. Currently, there are no other standard Transformers/Data Elements for these instruments. However, they can be added to the TEMPORARY TABLE depression_screenings_temp in the report using UNION statements and SQL to pull them directly from structured data. There is an example in Appendix C.

3. The report displays the Last Menstrual Period (LMP) and the Estimated Due Date (EDD). These fields should be added to the standard Transformer relevant_pregnancy and the standard Data Element named Pregnancies (these fields have recently been added to the Relevant Data Model). Example SQL for these fields is displayed in Appendix C.

TEMPORARY TABLE office_cpt_codes_temp

This table provides codes for office visits referenced by Partnership.

TEMPORARY TABLE initial_denom_codes_temp

See notes in the Introduction above on identifying patients with Partnership Insurance

TEMPORARY TABLE preg_mp_temp

The report requires that calculations be made for gestational age. Therefore, two new fields need to be added to the Transformer relevant_pregnancy. These fields are:

- last_menstrual_period
- estimated_due_date

See Appendix C for suggested SQL that can be added to the Transformer. However, note that since this change is being made on the non-standardized Staging database, the programmer may need to customize the SQL code according to data location or unique approaches.

These two fields should also be added to the standard Data Element named Pregnancies. Appendix C also contains suggested SQL for the Data Element.

TEMPORARY TABLE future_delivery_temp

It is possible that there could be more than one pregnancy overlapping the measurement period, as identified in the TEMPORARY TABLE preg_mp_temp. In cases where the patient did not deliver before the end of the measurement period and where the Estimated Due Date (EDD) is after the end of the measurement period, the TEMPORARY TABLE future_delivery_temp will choose the pregnancy with an EDD closest to the end of the measurement period.

TEMPORARY TABLE past_delivery_temp

It is possible that there could be more than one pregnancy overlapping the measurement period, as identified in the TEMPORARY TABLE preg_mp_temp. In cases where there is at least one known delivery date during the measurement period, the TEMPORARY TABLE past_delivery_temp will choose the earliest delvery date.

Note that it is possible that a single patient can have a delivery in the measurement period (and thus a date in the column "Actual (past) date of delivery within 12 months") and a future due date period (and thus a date in the column "Estimated date of delivery (future)").

TEMPORARY TABLE preg_columns_temp

Note that the report will not display a delivery date unless the delivery date was entered into the EHR. Therefore, patients who did not deliver (i.e., pregnancy ended for some reason before delivery) or patients missing this information will have visits where the patient was apparently pregnant (column Pregnant at time of Visit = “Yes”) but with no EDD (column “Estimated date of delivery (future)”) and no delivery date (column “Actual (past) date of delivery within 12 months”).

TEMPORARY TABLE depression_screenings_temp

This table gathers the results of all of the screening instruments your health center uses. The PHQ-2 and PHQ-9 are almost universal among health centers, but the programmer should ensure that all other instrument results are also gathered by the code. Each individual instrument is added to the table with a UNION expression.

A list of HEDIS-approved instruments appears in the Partnership specifications document after the section “Additional Background Information.” Also see point #2 above in the section on configurating Transformers for this measure. The list in the Partnership document contains the instrument’s official LOINC code that goes into the column loinc_code_screen. The instrument abbreviation from the specifications document goes into the report column tool_name. The result of the depression screen (either 'Positive' or 'Negative') goes into the report column interp_result, which depends on the numerical score of the screen (column screen_score). The calculation used to obtain a result from a score in the report should correspond to the calculation for a positive finding in the table of the specifications document (and should also correspond to how the health center EHR interprets the score).

The SQL code contains SELECT queries for the PHQ-2 and PHQ-9 (see point #1 above in the section on configurating Transformers for this measure). There is also a greyed-out (in other words, “noted”) SELECT query for the Edinburgh Depression Screen (based on the new Transformer, if your health center uses the Edinburgh) and a greyed-out SELECT query for any other screening instrument your health center might have. These can be activated if your health center uses additional depression screens other than the PHQ-2 and PHQ-9. Otherwise, the greyed-out SQL can remain noted or be deleted from the code. Add as many UNION queries as necessary to get all of the depression instruments used. Data for instruments other than the Edinburgh can be extracted directly from Structured Data in the TEMPORARY TABLE or a custom Transformer created (there would be no corresponding Data Element, though).

Description of ECDS Measure Report: Unhealthy Alcohol Use Screening and Follow Up (ASF-E)

This report is based on the Relevant Production Database, and so a higher degree of standardization here means that the programmer does not need to customize the SQL code as much. However, this report requires two new Transformer-Data Element pairs.

Similar to the other screening/follow-up measures (like depression screening/follow-up or tobacco screening/follow-up), this measure focuses on unhealthy alcohol use screenings for all patients in the denominator and then, for patients with a positive screen, evaluates whether a follow-up activity was performed. The presentation with the title "New ECDS Measures: Focus on Reports" mentioned in the section Supporting Documents (above) contains a flow-chart for this measure.

Since this is a new measure among RCHC health centers, unhealthy alcohol use screens and alcohol follow-up (i.e., counseling in some form) must be identified in the EHR (normally in structured data, if they are already being entered) and extracted by the new Transformer/Data Element pair.

The Partnership ECDS measure specifications document defines three different instruments that can be used for alcohol screening. These are:

- Audit-C
- Audit
- Single Question Screen. Note that this screen must adhere to specific language in the document or be approved by Partnership.

A health center can have one or any combination of these screens. The first step is to work with a clinical or behavioral health contact at your health center to help identify the alcohol use screens being used. Then, identify the screens in your structured data[7]. Each screening type should have its own Transformer/Data Element pair.

Instrument Name	Suggested Transformer Name	Data Element Name
Audit-C	relevant_audit_c_screens	AUDIT C Screens
Audit	relevant_audit_screens	AUDIT Screens
Single Question Screen	relevant_single_q_alc_screens	Single Question Alcohol use Screens

[7] Health centers with eCW can use the report "RCHC List All Structured Data Items" to identify structured data items, along with the unique ID numbers that will be used in the SQL code

Appendix D contains example SQL for the Audit-C Transformer/Data Element pair. If your health center uses the Audit screen and/or single question alcohol use screen, the Transformer/Data Element pair will have similar SQL code. The programmer will need to customize the SQL with the unique internal ID numbers and location of data.

Appendix D contains example SQL for the alcohol counseling Transformer/Data Element pair.

Suggested Transformer Name	Data Element Name
relevant_alcohol_counseling	Alcohol Counseling Or Other Followups

TEMPORARY TABLE universe_temp_raw

See notes in the Introduction above on identifying patients with Partnership Insurance

The visits are not restricted to medical visits or any other particular kind of visit because we want as many instances of alcohol screenings and counselings as possible.

TEMPORARY TABLE alc_followup_struct_raw_temp

Partnership will identify alcohol follow-up counseling from structured data through SNOMED codes displayed in the output column Counseling_SNOMED. The EHR systems used by RCHC health centers do not automatically associate structured data items with SNOMED codes, so the SNOMED code must be added in the SQL. In the section “Additional Background Information” of the Partnership measure specifications document (see section “Supporting Documents” above) is a table with some recommeneded SNOMED codes for counseling. The sample SQL in Appendix D uses the SNOMED code 413473000 (corresponding to “Counseling about alcohol consumption”), but any of the codes in the table can be used if one better describes the exact kind of counseling at your health center.

Results Query

Two groups of records are added together using UNION statements. They are:

1) Individual visits in the measurement period where there was an alcohol screen and/or an alcohol counseling activity. These records feature the visit date in the column Date_of_Service and other visit-associated information.
2) Denominator patients with exclusions. These are not associated with a visit date in the column Date_of_Service but have dates and codes in the exclusion columns (those with names that begin with “Exclusions_”).

Note that the data for two output columns associated with individual visits are likely not present in the EHR. These columns are “Site ID number” and “Provider Site Name.” The data in these fields should align with the Partnership ID and recognized name of the site. In the default SQL, a NULL value is present for these fields.

If the site data cannot be extracted from the EHR, there are at least two options. First, a programmer can use a CASE WHEN statement to code the fields based on the Relevant location where the service was given (in the results query, the text location is in the field universe_temp_raw.Site_Name). Alternately, the location can be displayed in the query results and then replaced manually in Excel.

Appendix A: SQL Code for the Report "ECDS: Breast Cancer Screening (BCS-E)"

Database: Production Database

Description: This report displays data for the ECDS Breast Cancer Screening (BCS-E) measure that could be submitted to Partnership HealthPlan of California (2022 Edition)

Parameters: {{start_date}} and {{end_date}}

Report SQL Code:

```
/*************************

Report Name: ECDS: Breast Cancer Screening (BCS-E)

Code Edited By: Ben Fouts for RCHC

Description: This report displays data for submission to Partnership Healthplan. It should not be used to
   directly evaluate a Quality Measure or for case management purposes

RCHC Started: June 2022 (based on code from the Partnership SQL specifications document)

Version Date: July 25, 2022

Revision History: 2022 Version 1

Resources Used: patients, visits, visit_set_memberships, locations, mammograms, hedis_value_set_codes,
 visit_billing_codes, billing_codes, visit_diagnosis_codes, patient_diagnoses, diagnosis_codes

*************************/

-- Universe Definition: visits with a claim that has:
--   1) Patient age between age 52 years, 0 days to 74 years, 364 days at the end of the measurement period
--   2) Patient sex = female (change this if your health center has a more precise way of defining it)
--   3) Patient currently has Partnership HealthPlan insurance -- this is the default, but can be changed
--        by the programmer to join to internal tables containing current Partnership patients
--        from the last membership file
DROP TABLE IF EXISTS universe_temp_raw;
CREATE TEMPORARY TABLE universe_temp_raw AS
SELECT DISTINCT
```

```
    id AS patient_id,
    mrn,
    subscriber_number AS Member_Key,
    first_name AS Member_First_Name,
    NULL AS Member_Middle_Name,
    last_name AS Member_Last_Name,
    date_of_birth :: DATE AS DOB,
    EXTRACT(YEAR FROM age({{end_date}} :: DATE, date_of_birth :: DATE)) AS Age_endMP,
    'F' AS Sex
FROM patients
WHERE assigned_female_at_birth IS TRUE
    AND EXTRACT(YEAR FROM age({{end_date}} :: DATE, date_of_birth :: DATE)) BETWEEN 52 and 74
    AND primary_insurance_group_id = 17 -- "PARTNERSHIP MANAGED CARE"  -- this will be different at every
health center
    AND EXISTS(
        SELECT
        FROM visits
                INNER JOIN visit_set_memberships ON visits.id = visit_set_memberships.visit_id
        WHERE visits.visit_date :: DATE BETWEEN {{start_date}} AND {{end_date}}
          AND visit_set_memberships.standard_visit_set_id = 'uds_medical'
          AND visits.patient_id = patients.id
    );


-- Collect mammograms from structured data within the lookback period.
-- This is the only source of mammograms identified by the report default code
-- Artificial SNOMED code used so Partnership can identify it
DROP TABLE IF EXISTS mammograms_temp;
CREATE TEMPORARY TABLE mammograms_temp AS
SELECT
    universe_temp_raw.patient_id,
            performed_on AS Date_of_Service,
    NULL AS CPTPX,
    NULL AS CPTMOD,
    NULL AS HCPCSPX,
    NULL AS HCPCSMOD,
    NULL AS LOINC,
    '24623002' :: VARCHAR AS SNOMED
FROM universe_temp_raw
            INNER JOIN mammograms ON mammograms.patient_id = universe_temp_raw.patient_id
WHERE performed_on BETWEEN ({{end_date}} :: DATE - INTERVAL '27 MONTHS') AND {{end_date}};


--  Make priority list of exclusion codes that appear as visit-associated diagnosis codes
DROP TABLE IF EXISTS exclusions_visit_list_temp;
CREATE TEMPORARY TABLE exclusions_visit_list_temp AS
```

```
SELECT
   patient_id,
   encounterid,
   priority,
   Exclusions_Code,
          Exclusions_DOS,
   ROW_NUMBER () OVER (
        PARTITION BY encounterid
        ORDER BY priority, Exclusions_Code) AS row_excl
FROM(SELECT
        universe_temp_raw.patient_id,
        visits.id AS encounterid,
        CASE WHEN value_set_name = 'History of Bilateral Mastectomy' THEN 1
           WHEN value_set_name = 'Absence of Left Breast' THEN 2
           WHEN value_set_name = 'Absence of Right Breast' THEN 3
           WHEN value_set_name = 'Advanced Illness' THEN 4
           WHEN value_set_name = 'Frailty Diagnosis' THEN 5
             ELSE 9 END AS priority,
        diagnosis_codes.code :: VARCHAR AS Exclusions_Code,
        visits.visit_date :: DATE AS Exclusions_DOS
      FROM universe_temp_raw
        INNER JOIN visits ON visits.patient_id = universe_temp_raw.patient_id
        INNER JOIN visit_diagnosis_codes ON visit_diagnosis_codes.visit_id = visits.id
        INNER JOIN diagnosis_codes ON diagnosis_codes.id = visit_diagnosis_codes.diagnosis_code_id
        INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = diagnosis_codes.code
            AND latest = 'TRUE'
      WHERE visits.visit_date :: DATE BETWEEN ({{end_date}} :: DATE - INTERVAL '27 MONTHS') AND {{end_date}}
        AND hedis_value_set_codes.value_set_name IN('History of Bilateral Mastectomy', 'Absence of Left
Breast',
            'Absence of Right Breast', 'Advanced Illness', 'Frailty Diagnosis', 'Frailty Symptom')
      UNION
      SELECT
        universe_temp_raw.patient_id,
        visits.id AS encounterid,
        CASE WHEN value_set_name = 'Frailty Encounter' THEN 7
            ELSE 8 END AS priority,
        billing_codes.code:: VARCHAR AS Exclusions_Code,
        visits.visit_date :: DATE AS Exclusions_DOS
      FROM universe_temp_raw
        INNER JOIN visits ON visits.patient_id = universe_temp_raw.patient_id
        INNER JOIN visit_billing_codes ON visit_billing_codes.visit_id = visits.id
        INNER JOIN billing_codes ON billing_codes.id = visit_billing_codes.billing_code_id
        INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = billing_codes.code
            AND latest = 'TRUE'
      WHERE visits.visit_date :: DATE BETWEEN ({{end_date}} :: DATE - INTERVAL '27 MONTHS') AND {{end_date}}
        AND hedis_value_set_codes.value_set_name IN('Frailty Encounter', 'Frailty Device')) AS
```

```
exclusions_visit_list_raw;


-- Place visit-associated diagnosis codes into columns, by encounter
DROP TABLE IF EXISTS exclusions_visit_list_final;
CREATE TEMPORARY TABLE exclusions_visit_list_final AS
SELECT
   enc_list.patient_id,
   enc_list.encounterid,
   enc_list.Exclusions_DOS AS Date_of_Service,
   excl_1_row.Exclusions_Code_1,
   excl_1_row.Exclusions_Code_1_DOS,
   excl_2_row.Exclusions_Code_2,
   excl_2_row.Exclusions_Code_2_DOS,
   excl_3_row.Exclusions_Code_3,
   excl_3_row.Exclusions_Code_3_DOS,
   excl_4_row.Exclusions_Code_4,
   excl_4_row.Exclusions_Code_4_DOS
FROM(SELECT DISTINCT
        patient_id,
        encounterid,
        Exclusions_DOS
     FROM exclusions_visit_list_temp) AS enc_list
INNER JOIN (SELECT DISTINCT
        patient_id,
        encounterid,
        Exclusions_Code AS Exclusions_Code_1,
        Exclusions_DOS AS Exclusions_Code_1_DOS
     FROM exclusions_visit_list_temp
     WHERE row_excl = 1) AS excl_1_row ON excl_1_row.encounterid = enc_list.encounterid
LEFT JOIN (SELECT DISTINCT
        patient_id,
        encounterid,
        Exclusions_Code AS Exclusions_Code_2,
        Exclusions_DOS AS Exclusions_Code_2_DOS
     FROM exclusions_visit_list_temp
     WHERE row_excl = 2) AS excl_2_row ON excl_2_row.encounterid = enc_list.encounterid
LEFT JOIN (SELECT DISTINCT
        patient_id,
        encounterid,
        Exclusions_Code AS Exclusions_Code_3,
        Exclusions_DOS AS Exclusions_Code_3_DOS
     FROM exclusions_visit_list_temp
     WHERE row_excl = 3) AS excl_3_row ON excl_3_row.encounterid = enc_list.encounterid
LEFT JOIN (SELECT DISTINCT
        patient_id,
```

```
        encounterid,
        Exclusions_Code AS Exclusions_Code_4,
        Exclusions_DOS AS Exclusions_Code_4_DOS
      FROM exclusions_visit_list_temp
      WHERE row_excl = 4) AS excl_4_row ON excl_4_row.encounterid = enc_list.encounterid;


-- Make priority list of exclusion diagnosis codes that appear on the patient's Problem List
DROP TABLE IF EXISTS exclusions_problem_list_temp;
CREATE TEMPORARY TABLE exclusions_problem_list_temp AS
SELECT
   patient_id,
   priority,
   Exclusions_Code,
            Exclusions_DOS,
   ROW_NUMBER () OVER (
        PARTITION BY patient_id
        ORDER BY priority, Exclusions_DOS, Exclusions_Code) AS row_excl
FROM(SELECT DISTINCT
        patient_diagnoses.patient_id,
        CASE WHEN value_set_name = 'History of Bilateral Mastectomy' THEN 1
           WHEN value_set_name = 'Absence of Left Breast' THEN 2
           WHEN value_set_name = 'Absence of Right Breast' THEN 3
           WHEN value_set_name = 'Advanced Illness' THEN 4
           WHEN value_set_name = 'Frailty Diagnosis' THEN 5
             ELSE 9 END AS priority,
        diagnosis_codes.code AS Exclusions_Code,
        patient_diagnoses.start_date AS Exclusions_DOS
      FROM patient_diagnoses
        INNER JOIN universe_temp_raw ON universe_temp_raw.patient_id = patient_diagnoses.patient_id
        INNER JOIN diagnosis_codes ON diagnosis_codes.id = patient_diagnoses.diagnosis_code_id
        INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = diagnosis_codes.code
            AND latest = 'TRUE'
     WHERE hedis_value_set_codes.value_set_name IN('History of Bilateral Mastectomy', 'Absence of Left
Breast',
            'Absence of Right Breast', 'Advanced Illness', 'Frailty Diagnosis', 'Frailty Symptom')
   ) AS raw_pl
ORDER BY patient_id, priority, Exclusions_DOS, Exclusions_Code;


-- Place problem list diagnosis codes into columns, by patient
DROP TABLE IF EXISTS exclusions_problem_list_final;
CREATE TEMPORARY TABLE exclusions_problem_list_final AS
SELECT
```

```
    enc_list.patient_id,
    excl_1_row.Exclusions_Code_1,
    excl_1_row.Exclusions_Code_1_DOS,
    excl_2_row.Exclusions_Code_2,
    excl_2_row.Exclusions_Code_2_DOS,
    excl_3_row.Exclusions_Code_3,
    excl_3_row.Exclusions_Code_3_DOS,
    excl_4_row.Exclusions_Code_4,
    excl_4_row.Exclusions_Code_4_DOS
FROM(SELECT DISTINCT
        patient_id
    FROM exclusions_problem_list_temp) AS enc_list
INNER JOIN (SELECT DISTINCT
        patient_id,
        Exclusions_Code AS Exclusions_Code_1,
        Exclusions_DOS AS Exclusions_Code_1_DOS
    FROM exclusions_problem_list_temp
    WHERE row_excl = 1) AS excl_1_row ON excl_1_row.patient_id = enc_list.patient_id
LEFT JOIN (SELECT DISTINCT
        patient_id,
        Exclusions_Code AS Exclusions_Code_2,
        Exclusions_DOS AS Exclusions_Code_2_DOS
    FROM exclusions_problem_list_temp
    WHERE row_excl = 2) AS excl_2_row ON excl_2_row.patient_id = enc_list.patient_id
LEFT JOIN (SELECT DISTINCT
        patient_id,
        Exclusions_Code AS Exclusions_Code_3,
        Exclusions_DOS AS Exclusions_Code_3_DOS
    FROM exclusions_problem_list_temp
    WHERE row_excl = 3) AS excl_3_row ON excl_3_row.patient_id = enc_list.patient_id
LEFT JOIN (SELECT DISTINCT
        patient_id,
        Exclusions_Code AS Exclusions_Code_4,
        Exclusions_DOS AS Exclusions_Code_4_DOS
    FROM exclusions_problem_list_temp
    WHERE row_excl = 4) AS excl_4_row ON excl_4_row.patient_id = enc_list.patient_id
ORDER BY patient_id;
```

-- Results Query

```
-- Note: Add "Site_ID_Number" from your system or manually to the subquery after the first UNION command.
--    This subquery displays the site name where the service was provided (other subqueries do not display
--    visit information). The Site_ID_Number is the NPI number assigned to the site
--    where care was provided (i.e., the Site_Name)
SELECT DISTINCT * FROM
(SELECT
   mammograms_temp.Date_of_Service :: DATE,
   null AS Site_Name,
   null AS Site_ID_Number,
   universe_temp_raw.Member_Key,
   null AS Member_Name,
   universe_temp_raw.Member_First_Name,
   universe_temp_raw.Member_Middle_Name,
   universe_temp_raw.Member_Last_Name,
   universe_temp_raw.DOB,
   universe_temp_raw.Sex,
   mammograms_temp.CPTPX :: VARCHAR,
   mammograms_temp.CPTMOD :: VARCHAR,
   mammograms_temp.HCPCSPX :: VARCHAR,
   mammograms_temp.HCPCSMOD :: VARCHAR,
   mammograms_temp.LOINC :: VARCHAR,
   mammograms_temp.SNOMED :: VARCHAR,
   NULL AS Exclusions_Code_1,
   NULL AS Exclusions_Code_1_DOS,
   NULL AS Exclusions_Code_2,
   NULL AS Exclusions_Code_2_DOS,
   NULL AS Exclusions_Code_3,
   NULL AS Exclusions_Code_3_DOS,
   NULL AS Exclusions_Code_4,
   NULL AS Exclusions_Code_4_DOS
FROM universe_temp_raw
   INNER JOIN mammograms_temp ON mammograms_temp.patient_id = universe_temp_raw.patient_id
UNION
SELECT
   exclusions_visit_list_final.Date_of_Service,
   locations.name AS Site_Name,
   null AS Site_ID_Number,
   universe_temp_raw.Member_Key,
   null AS Member_Name,
   universe_temp_raw.Member_First_Name,
```

```
    universe_temp_raw.Member_Middle_Name,
    universe_temp_raw.Member_Last_Name,
    universe_temp_raw.DOB,
    universe_temp_raw.Sex,
    NULL AS CPTPX,
    NULL AS CPTMOD,
    NULL AS HCPCSPX,
    NULL AS HCPCSMOD,
    NULL AS LOINC,
    NULL AS SNOMED,
    exclusions_visit_list_final.Exclusions_Code_1,
    exclusions_visit_list_final.Exclusions_Code_1_DOS,
    exclusions_visit_list_final.Exclusions_Code_2,
    exclusions_visit_list_final.Exclusions_Code_2_DOS,
    exclusions_visit_list_final.Exclusions_Code_3,
    exclusions_visit_list_final.Exclusions_Code_3_DOS,
    exclusions_visit_list_final.Exclusions_Code_4,
    exclusions_visit_list_final.Exclusions_Code_4_DOS
FROM universe_temp_raw
    INNER JOIN exclusions_visit_list_final ON exclusions_visit_list_final.patient_id = universe_temp_raw.patient_id
    INNER JOIN visits ON visits.id = exclusions_visit_list_final.encounterid
    LEFT JOIN locations ON locations.id = visits.location_id
UNION
SELECT
    NULL AS Date_of_Service,
    null AS Site_Name,
    null AS Site_ID_Number,
    universe_temp_raw.Member_Key,
    null AS Member_Name,
    universe_temp_raw.Member_First_Name,
    universe_temp_raw.Member_Middle_Name,
    universe_temp_raw.Member_Last_Name,
    universe_temp_raw.DOB,
    universe_temp_raw.Sex,
    NULL AS CPTPX,
    NULL AS CPTMOD,
    NULL AS HCPCSPX,
    NULL AS HCPCSMOD,
    NULL AS LOINC,
    NULL AS SNOMED,
```

```
    exclusions_problem_list_final.Exclusions_Code_1,
    exclusions_problem_list_final.Exclusions_Code_1_DOS,
    exclusions_problem_list_final.Exclusions_Code_2,
    exclusions_problem_list_final.Exclusions_Code_2_DOS,
    exclusions_problem_list_final.Exclusions_Code_3,
    exclusions_problem_list_final.Exclusions_Code_3_DOS,
    exclusions_problem_list_final.Exclusions_Code_4,
    exclusions_problem_list_final.Exclusions_Code_4_DOS
FROM universe_temp_raw
    INNER JOIN exclusions_problem_list_final ON exclusions_problem_list_final.patient_id =
universe_temp_raw.patient_id) AS raw_result
ORDER BY Member_Last_Name, Member_First_Name, Member_Key, Date_of_Service, Exclusions_Code_1_DOS,
Exclusions_Code_2_DOS
```

Appendix B: SQL Code for the Report "ECDS: Follow-Up Care for Children Prescribed ADHD Medication (ADD-E)"

Database: Staging Database

Description: This report displays data for the ECDS Follow-Up Care for Children Prescribed ADHD Medication (ADD-E) measure that could be submitted to Partnership HealthPlan of California (2022 Edition)

Parameters: {{start_date}} and {{end_date}}

Report SQL Code:

```
/*************************

Report Name: ECDS: Follow-Up Care for Children Prescribed ADHD Medication (ADD-E)

Code Edited By: Ben Fouts for RCHC

Description: This report displays data for submission to Partnership Healthplan. It should not be used to
   directly evaluate a Quality Measure or for case management purposes

RCHC Started: June 2022 (based on code from the Partnership SQL specifications document)

Version Date: July 25, 2022

Revision History: 2022 Version 1

Resources Used: relevant_patients, relevant_visits, relevant_providers, doctors, relevant_programs,
 relevant_genders, hedis_value_set_codes, edi_invoice, edi_inv_cpt, edi_inv_diagnosis,
 relevant_visit_diagnosis_codes, relevant_patient_diagnoses, relevant_diagnosis_codes

*************************/

--Define all codes needed for the measure
DROP TABLE IF EXISTS temp_value_sets;
CREATE TEMPORARY TABLE temp_value_sets AS
SELECT *
FROM (
        VALUES
    ('99492','CPT','Additional Visit Codes'), ('99493','CPT','Additional Visit Codes'), ('99494','CPT','Additional Visit
```

```
Codes'),
      ('G0071','HCPCS','Additional Visit Codes'), ('G0512','HCPCS','Additional Visit Codes'),
('G2012','HCPCS','Additional Visit Codes'),
      ('F90.0','ICD10CM','ADHD'), ('F90.1','ICD10CM','ADHD'), ('F90.2','ICD10CM','ADHD'),
('F90.8','ICD10CM','ADHD'),
      ('F90.9','ICD10CM','ADHD'), ('314','ICD9CM','ADHD'),('314.01','ICD9CM','ADHD')
    )
      AS value_set_values_values (code, code_system, code_group);


-- Universe Definition: visits with a claim that has:
--    1) An ADHD diagnosis code (temp_value_sets.code_group = 'ADHD')
--    2) An appropriate office or telephone procedure code (temp_value_sets.code_group = 'Visit Codes')
--    3) A visit in the calculated measurement period
--    4) Patient age between age 6 years, 0 days to 13 years, 363 days at the end of the measurement period
--    5) Patient currently has Partnership HealthPlan insurance -- this is the default, but can be changed
--         by the programmer to join to internal tables containing current Partnership patients
--         from the last monthly membership file
DROP TABLE IF EXISTS universe_temp_raw;
CREATE TEMPORARY TABLE universe_temp_raw AS
SELECT DISTINCT
   relevant_patients.id AS patient_id,
   relevant_patients.mrn,
   LEFT(relevant_patients.subscriber_number,10) AS Member_Key,
   relevant_patients.first_name AS Member_First_Name,
   NULL AS Member_Middle_Name,
   relevant_patients.last_name AS Member_Last_Name,
   relevant_patients.date_of_birth AS DOB,
   EXTRACT(YEAR FROM age(edi_invoice.servicedt, relevant_patients.date_of_birth)) AS Age,
   CASE WHEN relevant_genders.abbreviation IN('F','M','U') THEN relevant_genders.abbreviation
     WHEN relevant_genders.abbreviation = 'NA' THEN 'U'
     ELSE 'O' END AS Sex,
   edi_invoice.encounterid AS enc_id,
   edi_invoice.servicedt :: DATE AS Date_of_Service,
   relevant_programs.name AS visit_facility,
   relevant_providers.first_name AS Clinician_First_Name,
   NULL AS Clinician_Middle_Name,
   relevant_providers.last_name AS Clinician_Last_Name,
   relevant_providers.npi AS Clinician_NPI_Number,
   CASE WHEN doctors.deano IS NOT NULL THEN 'Y' ELSE 'N' END AS Prescriber,
   CASE WHEN temp_value_sets_cpt.code_system = 'CPT' THEN edi_inv_cpt.code END AS visit_cpt,
   CASE WHEN temp_value_sets_cpt.code_system = 'CPT' THEN edi_inv_cpt.mod1 END AS visit_cptmod,
   CASE WHEN temp_value_sets_cpt.code_system = 'CPT' THEN edi_inv_cpt.displayindex END AS
visit_cpt_displayindex,
   CASE WHEN temp_value_sets_cpt.code_system = 'HCPCS' THEN edi_inv_cpt.code END AS visit_hcpcs,
   CASE WHEN temp_value_sets_cpt.code_system = 'HCPCS' THEN edi_inv_cpt.mod1 END AS visit_hcpcsmod,
```

```
   CASE WHEN temp_value_sets_cpt.code_system = 'HCPCS' THEN edi_inv_cpt.displayindex END AS
visit_hcpcs_displayindex,
   edi_inv_diagnosis.code AS adhd_dx,
   edi_inv_diagnosis.primarycode,
   edi_inv_diagnosis.icdorder
FROM relevant_patients
   INNER JOIN edi_invoice ON edi_invoice.patientid = relevant_patients.id
      AND edi_invoice.servicedt BETWEEN ({{start_date}} :: DATE - INTERVAL '10 months' + INTERVAL '1 DAY') AND
{{end_date}}
      AND edi_invoice.deleteflag = 0
      AND edi_invoice.voidflag = 0
   INNER JOIN edi_inv_cpt ON edi_inv_cpt.invoiceid = edi_invoice.id
      AND edi_inv_cpt.deleteflag = 0
   LEFT JOIN temp_value_sets AS temp_value_sets_cpt ON temp_value_sets_cpt.code = LEFT(edi_inv_cpt.code, 5)
      AND temp_value_sets_cpt.code_group = 'Additional Visit Codes'
   LEFT JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = LEFT(edi_inv_cpt.code, 5)
      AND value_set_name IN('BH Outpatient', 'Observation', 'Health and Behavior Assessment or Intervention',
            'Online Assessments','Telephone Visits', 'Visit Setting Unspecified')
      AND code_system_name IN('CPT','HCPCS')
      AND latest = 'TRUE'
   INNER JOIN edi_inv_diagnosis ON edi_inv_diagnosis.invoiceid = edi_invoice.id
   INNER JOIN temp_value_sets AS temp_value_sets_icd ON temp_value_sets_icd.code = edi_inv_diagnosis.code
      AND temp_value_sets_icd.code_group = 'ADHD'
   INNER JOIN relevant_providers ON relevant_providers.id = edi_invoice.dosproviderid
   LEFT JOIN doctors ON doctors.doctorid = edi_invoice.dosproviderid
   LEFT JOIN relevant_programs ON relevant_programs.id = edi_invoice.invfacilityid
   LEFT JOIN relevant_genders ON relevant_genders.id = relevant_patients.gender_id
WHERE (temp_value_sets_cpt.code IS NOT NULL OR hedis_value_set_codes.code_value IS NOT NULL)
   AND EXTRACT(YEAR FROM age({{end_date}}, relevant_patients.date_of_birth)) BETWEEN 6 and 13
   AND relevant_patients.primary_insurance_group_id = 17; -- "PARTNERSHIP MANAGED CARE"


-- This query unduplicates the temporary table "universe_temp_raw" (see universe definition above)
-- so that one row is equal to one visit no matter how many CPT and ICD codes are picked up
-- If there is more than one CPT code, the display index field is used to prioritize only one
-- If there is more than one ICD code, the ICD order field is used to prioritize only one
DROP TABLE IF EXISTS universe_temp;
CREATE TEMPORARY TABLE universe_temp AS
SELECT DISTINCT
   universe_temp_raw.patient_id,
   mrn,
   Member_Key,
   Member_First_Name,
   Member_Middle_Name,
   Member_Last_Name,
   DOB,
```

```
    Age,
    Sex,
    universe_temp_raw.Date_of_Service,
    visit_facility,
    Clinician_First_Name,
    Clinician_Middle_Name,
    Clinician_Last_Name,
    Clinician_NPI_Number,
    Prescriber,
    cpt_join.CPTPX,
    cpt_join.CPTMOD,
    hcpcs_join.HCPCS,
    hcpcs_join.HCPCMOD,
    icd_join.ICDDX_10
FROM universe_temp_raw
    INNER JOIN(SELECT DISTINCT ON (patient_id, Date_of_Service)
        patient_id,
        Date_of_Service,
        visit_cpt AS CPTPX,
        visit_cptmod AS CPTMOD
      FROM universe_temp_raw
      ORDER BY patient_id, Date_of_Service, visit_cpt_displayindex, visit_cptmod) AS cpt_join
        ON cpt_join.patient_id = universe_temp_raw.patient_id
          AND cpt_join.Date_of_Service = universe_temp_raw.Date_of_Service
    INNER JOIN(SELECT DISTINCT ON (patient_id, Date_of_Service)
        patient_id,
        Date_of_Service,
        visit_hcpcs AS HCPCS,
        visit_hcpcsmod AS HCPCMOD
      FROM universe_temp_raw
      ORDER BY patient_id, Date_of_Service, visit_hcpcs_displayindex, visit_hcpcsmod) AS hcpcs_join
        ON hcpcs_join.patient_id = universe_temp_raw.patient_id
          AND hcpcs_join.Date_of_Service = universe_temp_raw.Date_of_Service
    INNER JOIN(SELECT DISTINCT ON (patient_id, Date_of_Service)
        patient_id,
        Date_of_Service,
        adhd_dx AS ICDDX_10
      FROM universe_temp_raw
      ORDER BY patient_id, Date_of_Service, primarycode DESC, icdorder) AS icd_join
        ON icd_join.patient_id = universe_temp_raw.patient_id
          AND icd_join.Date_of_Service = universe_temp_raw.Date_of_Service;


-- Exclusions from the Problem List
DROP TABLE IF EXISTS exclusions_problem_list_raw;
CREATE TEMPORARY TABLE exclusions_problem_list_raw AS
```

```
SELECT DISTINCT
        relevant_patient_diagnoses.patient_id,
        relevant_diagnosis_codes.code AS exclude_code_problst,
        relevant_patient_diagnoses.start_date AS firstDx,
        relevant_patient_diagnoses.end_date AS ended_on
FROM relevant_patient_diagnoses
  INNER JOIN relevant_diagnosis_codes ON relevant_diagnosis_codes.id =
relevant_patient_diagnoses.diagnosis_code_id
  INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = relevant_diagnosis_codes.code
    AND hedis_value_set_codes.value_set_name = 'Narcolepsy'
    AND latest = 'TRUE'
WHERE relevant_patient_diagnoses.start_date <= {{end_date}};


-- Exclusions from Visit Assessments (with first date the code appeared on an Assessment)
DROP TABLE IF EXISTS exclusions_assess_raw;
CREATE TEMPORARY TABLE exclusions_assess_raw AS
SELECT
  relevant_visits.patient_id,
        relevant_diagnosis_codes.code AS exclude_code_assess,
        MIN(relevant_visits.visit_date :: DATE) AS firstDx
FROM relevant_visits
  INNER JOIN relevant_visit_diagnosis_codes ON relevant_visit_diagnosis_codes.visit_id = relevant_visits.id
  INNER JOIN relevant_diagnosis_codes ON relevant_diagnosis_codes.id =
relevant_visit_diagnosis_codes.diagnosis_code_id
  INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = relevant_diagnosis_codes.code
    AND hedis_value_set_codes.value_set_name = 'Narcolepsy'
    AND latest = 'TRUE'
WHERE relevant_visits.visit_date :: DATE <= {{end_date}}
GROUP BY relevant_visits.patient_id, relevant_diagnosis_codes.code;


--Combine and prioritize exclusions. See Standards and Specifications Document for explanation
DROP TABLE IF EXISTS exclusions_list_temp;
CREATE TEMPORARY TABLE exclusions_list_temp AS
SELECT
  patient_id,
  code,
  code_first_date,
  resolve_date,
  priority,
  ROW_NUMBER () OVER (
      PARTITION BY patient_id
      ORDER BY priority, code_first_date) AS row_excl
FROM(
    SELECT
```

```
        patient_id,
        exclude_code_problst AS code,
        firstDx :: DATE AS code_first_date,
        ended_on :: DATE AS resolve_date,
        CASE WHEN ended_on IS NOT NULL THEN 1
          ELSE 2 END AS priority
      FROM exclusions_problem_list_raw
    UNION
      SELECT
        patient_id,
        exclude_code_assess AS code,
        firstDx :: DATE AS code_first_date,
        NULL AS resolve_date,
        3 AS priority
      FROM exclusions_assess_raw
    ) AS exclusion_union;


-- Results query.
-- Note: Add "Site_ID_Number" from your system or manually. This is the NPI number assigned to
--     the site where care was provided (i.e., the site displayed in the field Site_Name)
-- Note: Add "Provider_Key" from your system or manually. This is the PCP ID assigned by PHC to
--     the provider who provided care at the visit (i.e., the Clinician_First_Name, etc.)
-- Note: You can choose to combine the clinician names into the field "Clinician_Name"
--     and the patient/member names into the field "Member_Name"
SELECT
  Date_of_Service,
  visit_facility AS Site_Name,
  null AS Site_ID_Number,
  null AS Provider_Key,
  Clinician_NPI_Number,
  NULL AS Clinician_Name,
  Clinician_First_Name,
  Clinician_Middle_Name,
  Clinician_Last_Name,
  Prescriber,
  Member_Key,
  NULL AS Member_Name,
  Member_First_Name,
  Member_Middle_Name,
  Member_Last_Name,
  DOB,
  Sex,
  CPTPX,
  CPTMOD,
  HCPCS,
```

```
   HCPCMOD,
   ICDDX_10,
   exclude_1.code AS Exclusions_ICDDX10_1,
   exclude_1.code_first_date AS Exclusions_ICDDX10_1_DOS,
    CASE WHEN NOT exclude_1.code = exclude_2.code THEN exclude_2.code
      END AS Exclusions_ICDDX10_2,
   CASE WHEN NOT exclude_1.code = exclude_2.code THEN exclude_2.code_first_date
      END AS Exclusions_ICDDX10_2_DOS,
   exclude_1.resolve_date AS Exclusions_ICDDX10_1_Resolution,
   CASE WHEN NOT exclude_1.code = exclude_2.code THEN exclude_2.resolve_date
      END AS Exclusions_ICDDX10_2_Resolution
FROM universe_temp
   LEFT JOIN(SELECT
               patient_id,
               code,
               code_first_date,
               resolve_date
             FROM exclusions_list_temp
             WHERE row_excl = 1) AS exclude_1
         ON exclude_1.patient_id = universe_temp.patient_id
   LEFT JOIN(SELECT
               patient_id,
               code,
               code_first_date,
               resolve_date
             FROM exclusions_list_temp
             WHERE row_excl = 2) AS exclude_2
      ON exclude_2.patient_id = universe_temp.patient_id
ORDER BY Member_Last_Name, Member_First_Name, Date_of_Service
```

Appendix C: SQL Code for the Report "ECDS: Depression Related Measures (DMS-E, DSF-E, DRR-E, PND-E, PDS-E)" and Related Transformers/Data Elements

Configure Transformers/Data Elements

Check if your instance of Relevant has the Transformers relevant_phq2 and relevant_phq9 and how they are configured. There could be variations in the name and variations in the SQL code between health centers. In the report SQL code below, relevant_phq2 and relevant_phq9 each display one total score for one instrument on one date for one patient.

NEW Transformer/Data Element Pair for Edinburgh Depression Screens

Configure only if your health center uses the Edinburgh Depression Screen. Note that the ID numbers for catid, itemid, and detailed will be unique at each health center. One option to find these ID numbers is the report "RCHC List All Structured Data Items."

Transformer: Build relevant_edinburgh_screen

```
DROP TABLE IF EXISTS relevant_edinburgh_screen;
CREATE TABLE relevant_edinburgh_screen AS
SELECT DISTINCT
        enc.patientid           AS patient_id,
        enc.encounterid         AS visit_id,
        enc.date :: DATE        AS performed_on,
        structhpi.value :: INT  AS score,
        CASE WHEN structhpi.value :: INT >= 10 THEN TRUE
                ELSE FALSE END      AS result
FROM enc
        INNER JOIN structhpi ON structhpi.encounterid = enc.encounterid
                AND (structhpi.catid = 170663 AND structhpi.itemid = 10575 AND structhpi.detailid = 1549
                        OR structhpi.catid = 375124 AND structhpi.itemid = 22423 AND structhpi.detailid =
2310
                        OR structhpi.catid = 250273 AND structhpi.itemid = 18785 AND structhpi.detailid =
1516)
WHERE enc.deleteflag = 0
        AND enc.status = 'CHK'
        AND structhpi.value ~ '^\d+$'
```

Data Element: Edinburgh Depression Screens

```
SELECT DISTINCT
        patient_id,
        performed_on,
        score,
        result
FROM relevant_edinburgh_screens
```

NEW Fields for Existing Pregnancy Transformer/Data Element Pair

Transformer: Build relevant_pregnancy

This is a relatively complicated Transformer that features more than one TEMPORARY TABLE. The final subquery creates the table relevant_pregnancy. Two fields (last_menstrual_period and estimated_due_date) should be added to the final subquery. At some health centers, these fields are already available from the existing code. Check the SQL code in the FROM statement for the following references (note there may be slight variations):

```
FROM obf_pregnancy preg
        LEFT JOIN obf_edd_details AS lmp
                ON lmp.pregid = preg.pregid AND lmp.conf_lmp_date ~ '^\d+\/\d+\/\d+$'
                        AND lmp.delstat = 0
        LEFT JOIN obf_edd_details AS final_edd
                ON final_edd.pregid = preg.pregid
                        AND final_edd.final_edd_date ~ '^\d+\/\d+\/\d+$'
                        AND final_edd.delstat = 0
```

This code JOINS the alias tables “lmp” and “final_edd.” If they exist, they can be added to the SELECT statement as the final two columns:

```
SELECT DISTINCT . . .
        COALESCE(COALESCE(final_edd.final_edd_date, init_edd.init_edd_date) :: DATE - 280,
                lmp.conf_lmp_date :: DATE) AS last_menstrual_period,
        COALESCE(COALESCE(final_edd.final_edd_date, init_edd.init_edd_date) :: DATE,
                lmp.conf_lmp_date :: DATE + 280) AS estimated_due_date
```

Configuring the RCHC Relevant QIP ECDS Reports

Data Element: Pregnancies

Add the two new fields to the Data Element. An example of the final code is below.

```
SELECT DISTINCT
        id,
        patient_id,
        started_on,
        ended_on,
        prenatal_treatment_initial_trimester,
        prenatal_treatment_initiated_at_health_center,
        last_menstrual_period,
        estimated_due_date
FROM relevant_pregnancy
```

Report SQL

Database: Staging Database

Description: This report displays data for the ECDS Depression Related Measures (DMS-E, DSF-E, DRR-E, PND-E, PDS-E) that could be submitted to Partnership HealthPlan of California (2022 Edition)

Parameters: {{start_date}} and {{end_date}}

Report SQL Code:

```
/*************************

Report Name: ECDS: Depression Related Measures (DMS-E, DSF-E, DRR-E, PND-E, PDS-E)

Code Edited By: Ben Fouts for RCHC

Description: This report displays data for submission to Partnership Healthplan. It should not be used to
    directly evaluate a Quality Measure or for case management purposes

RCHC Started: June 2022 (based on code from the Partnership SQL specifications document)

Version Date: July 25, 2022

Revision History: 2022 Version 2 (previous version was developed in 2021)
```

Resources Used: relevant_patients, edi_invoice, edi_inv_cpt, relevant_payers, relevant_payer_groups, relevant_genders, relevant_providers, relevant_pregnancy, relevant_deliveries, relevant_phq2, relevant_phq9, relevant_visit_diagnosis_codes, relevant_diagnosis_codes, hedis_value_set_codes, problemlist, itemdetail

*************************/*

```
--Define Office and Follow-up Visit CPT codes defined by Partnership
--These are the default codes and your health center may have additional local codes (see PHP instructions)
DROP TABLE IF EXISTS office_cpt_codes_temp;
CREATE TEMPORARY TABLE office_cpt_codes_temp AS
SELECT *
FROM (
        VALUES
    ('59400','CPT'),('59510','CPT'),('59618','CPT'),('59619','CPT'),('90791','CPT'),('90792','CPT'),
    ('90832','CPT'),('90833','CPT'),('90834','CPT'),('90836','CPT'),('90837','CPT'),('90838','CPT'),
    ('90839','CPT'),('90845','CPT'),('90846','CPT'),('90847','CPT'),('90849','CPT'),('90853','CPT'),
    ('90865','CPT'),('90867','CPT'),('90868','CPT'),('90869','CPT'),('90870','CPT'),('90875','CPT'),
    ('90876','CPT'),('90880','CPT'),('90887','CPT'),('96116','CPT'),('96121','CPT'),('96130','CPT'),
    ('96131','CPT'),('96132','CPT'),('96133','CPT'),('96136','CPT'),('96137','CPT'),('96138','CPT'),
    ('96139','CPT'),('96146','CPT'),('96151','CPT'),('96156','CPT'),('96170','CPT'),('96171','CPT'),
    ('97165','CPT'),('97166','CPT'),('97167','CPT'),('97799','CPT'),('98960','CPT'),('98961','CPT'),
    ('98962','CPT'),('98966','CPT'),('98967','CPT'),('98968','CPT'),('99078','CPT'),('99201','CPT'),
    ('99202','CPT'),('99203','CPT'),('99204','CPT'),('99205','CPT'),('99211','CPT'),('99212','CPT'),
    ('99213','CPT'),('99214','CPT'),('99215','CPT'),('99217','CPT'),('99218','CPT'),('99219','CPT'),
    ('99220','CPT'),('99241','CPT'),('99242','CPT'),('99243','CPT'),('99244','CPT'),('99245','CPT'),
    ('99304','CPT'),('99305','CPT'),('99306','CPT'),('99307','CPT'),('99308','CPT'),('99309','CPT'),
    ('99310','CPT'),('99315','CPT'),('99316','CPT'),('99318','CPT'),('99324','CPT'),('99325','CPT'),
    ('99326','CPT'),('99327','CPT'),('99328','CPT'),('99334','CPT'),('99335','CPT'),('99336','CPT'),
    ('99337','CPT'),('99339','CPT'),('99340','CPT'),('99341','CPT'),('99342','CPT'),('99343','CPT'),
    ('99344','CPT'),('99345','CPT'),('99347','CPT'),('99348','CPT'),('99349','CPT'),('99350','CPT'),
    ('99366','CPT'),('99381','CPT'),('99382','CPT'),('99383','CPT'),('99384','CPT'),('99385','CPT'),
    ('99386','CPT'),('99387','CPT'),('99391','CPT'),('99392','CPT'),('99393','CPT'),('99394','CPT'),
    ('99394','CPT'),('99395','CPT'),('99395','CPT'),('99396','CPT'),('99396','CPT'),('99397','CPT'),
    ('99397','CPT'),('99401','CPT'),('99402','CPT'),('99403','CPT'),('99404','CPT'),('99411','CPT'),
    ('99412','CPT'),('99441','CPT'),('99442','CPT'),('99443','CPT'),('99483','CPT'),('99484','CPT'),
    ('99492','CPT'),('99493','CPT'),('G0071','HCPCS'),('G0101 ','HCPCS'),('G0155','HCPCS'),('G0176','HCPCS'),
    ('G0177','HCPCS'),('G0402','HCPCS'),('G0409','HCPCS'),('G0410','HCPCS'),('G0411','HCPCS'),('G0438','HCPCS'),
    ('G0439','HCPCS'),('G0444','HCPCS'),('G0463','HCPCS'),('G0502','HCPCS'),('G0503','HCPCS'),('G0507','HCPCS'),
    ('G0511','HCPCS'),('G0512','HCPCS'),('G2012','HCPCS'),('G8431','HCPCS'),('G8510','HCPCS'),('H0002','HCPCS'),
    ('H0004','HCPCS'),('H0031','HCPCS'),('H0034','HCPCS'),('H0035','HCPCS'),('H0036','HCPCS'),('H0037','HCPCS'),
    ('H0039','HCPCS'),('H0040','HCPCS'),('H1000','HCPCS'),('H1002','HCPCS'),('H1003','HCPCS'),('H2000','HCPCS'),
    ('H2001','HCPCS'),('H2010','HCPCS'),('H2011','HCPCS'),('H2012','HCPCS'),('H2013','HCPCS'),('H2014','HCPCS'),
    ('H2015','HCPCS'),('H2016','HCPCS'),('H2017','HCPCS'),('H2018','HCPCS'),('H2019','HCPCS'),('H2020','HCPCS'),
    ('S0201','HCPCS'),('S9480','HCPCS'),('S9484','HCPCS'),('S9485','HCPCS'),('T1015','HCPCS'),('T1016','HCPCS'),
```

```
    ('T1017','HCPCS'),('T2022','HCPCS'),('T2023','HCPCS'),('Z1032','HCPCS'),('Z1032','HCPCS'),('Z1034','HCPCS'),
    ('Z1036','HCPCS'),('Z1038','HCPCS'),('Z6300','HCPCS'),('Z6302','HCPCS'),('Z6304','HCPCS'),('Z6306','HCPCS'),
    ('Z6300','HCPCS'),('Z6302','HCPCS'),('Z6304','HCPCS'),('Z6306','HCPCS'),('Z6308','HCPCS'),('Z6500','HCPCS')
  )
    AS value_set_values_values (code,code_system);


-- Universe Definition: visits with a:
--   1) Claim in the measurement period
--   2) Claim has a defined office procedure code (office_cpt_codes_temp.code)
--   3) Patient who is 12 years or older
--   4) Patient sho is enrolled in Partnership Managed Care insurance
-- Patients in the Universe currently have Partnership HealthPlan insurance -- this is the default,
--   but can be changed by the programmer to join to internal tables containing current
--   Partnership patients from the last monthly membership file
DROP TABLE IF EXISTS initial_denom_codes_temp;
CREATE TEMPORARY TABLE initial_denom_codes_temp AS
SELECT DISTINCT
  relevant_patients.id AS patient_id,
  relevant_patients.subscriber_number,
  relevant_patients.first_name AS memberfirstname,
  relevant_patients.last_name AS memberlastname,
  relevant_patients.date_of_birth :: DATE AS dob,
  CASE WHEN relevant_genders.abbreviation IN('F','M','U') THEN relevant_genders.abbreviation
    WHEN relevant_genders.abbreviation = 'NA'
        OR relevant_genders.id IS NULL
      THEN 'U'
    ELSE 'O' END AS Sex,
  edi_invoice.encounterid AS enc_id,
  edi_invoice.servicedt :: DATE AS enc_date,
  relevant_providers.last_name || ', ' || relevant_providers.first_name AS visit_clinician_full_name,
  relevant_providers.npi AS clinician_npi_number,
  office_cpt_codes_temp.code AS visit_cpt,
  edi_inv_cpt.mod1 AS cptmod,
  office_cpt_codes_temp.code_system,
  edi_inv_cpt.displayindex
FROM relevant_patients
  INNER JOIN edi_invoice ON edi_invoice.patientid = relevant_patients.id
    AND edi_invoice.servicedt BETWEEN {{start_date}} AND {{end_date}}
    AND edi_invoice.deleteflag = 0
    AND edi_invoice.voidflag = 0
  INNER JOIN edi_inv_cpt ON edi_inv_cpt.invoiceid = edi_invoice.id
  INNER JOIN office_cpt_codes_temp ON office_cpt_codes_temp.code = LEFT(edi_inv_cpt.code, 5)
  INNER JOIN relevant_payers ON relevant_payers.payer_key = relevant_patients.primary_insurance_id
  INNER JOIN relevant_payer_groups ON relevant_payer_groups.id = relevant_payers.payer_group_key
  LEFT JOIN relevant_genders ON relevant_genders.id = relevant_patients.gender_id
```

```
  LEFT JOIN relevant_providers ON relevant_providers.id = edi_invoice.dosproviderid
WHERE EXTRACT(YEAR FROM age(NOW(), relevant_patients.date_of_birth :: DATE)) >= 12
  AND relevant_patients.primary_insurance_group_id = 17; -- "PARTNERSHIP MANAGED CARE"


--Unduplicated encounter denominator
DROP TABLE IF EXISTS total_denominator_temp;
CREATE TEMPORARY TABLE total_denominator_temp AS
SELECT DISTINCT
  patient_id,
  subscriber_number,
  memberfirstname,
  memberlastname,
  dob,
  sex,
  enc_id,
  enc_date,
  visit_clinician_full_name,
  clinician_npi_number
FROM initial_denom_codes_temp;
CREATE INDEX index_total_denominator_temp_on_enc_id ON total_denominator_temp (enc_id);
CREATE INDEX index_total_denominator_temp_on_patient_id ON total_denominator_temp (patient_id);


--Produce Outpatient Visit CPT/HCPCS columns for the report output
DROP TABLE IF EXISTS cpt_columns_temp;
CREATE TEMPORARY TABLE cpt_columns_temp AS
WITH order_cpt AS(
  SELECT
    enc_id,
    visit_cpt,
    code_system,
    ROW_NUMBER() OVER (PARTITION BY enc_id, code_system
      ORDER BY displayindex, visit_cpt) AS row
  FROM (SELECT DISTINCT
          enc_id,
          visit_cpt,
          code_system,
          displayindex
        FROM initial_denom_codes_temp) AS undup
),cpt1 AS(
  SELECT enc_id, visit_cpt
      FROM order_cpt
      WHERE code_system = 'CPT'
        AND row = 1
),cpt2 AS(
```

```
   SELECT enc_id, visit_cpt
         FROM order_cpt
         WHERE code_system = 'CPT'
            AND row = 2
),cpt3 AS(
   SELECT enc_id, visit_cpt
         FROM order_cpt
         WHERE code_system = 'CPT'
            AND row = 3
),cpt4 AS(
   SELECT enc_id, visit_cpt
         FROM order_cpt
         WHERE code_system = 'CPT'
            AND row = 4
),cpt1_mod AS(
   SELECT cpt1.enc_id, initial_denom_codes_temp.cptmod
         FROM cpt1
            INNER JOIN initial_denom_codes_temp ON initial_denom_codes_temp.enc_id = cpt1.enc_id
               AND initial_denom_codes_temp.visit_cpt = cpt1.visit_cpt
         WHERE initial_denom_codes_temp.cptmod IS NOT NULL
),hcpcs1 AS(
SELECT enc_id, visit_cpt
      FROM order_cpt
      WHERE code_system = 'HCPCS'
         AND row = 1
),hcpcs1_mod AS(
   SELECT hcpcs1.enc_id, initial_denom_codes_temp.cptmod
         FROM hcpcs1
            INNER JOIN initial_denom_codes_temp ON initial_denom_codes_temp.enc_id = hcpcs1.enc_id
               AND initial_denom_codes_temp.visit_cpt = hcpcs1.visit_cpt
         WHERE initial_denom_codes_temp.cptmod IS NOT NULL
),enc_nbr_distinct AS (SELECT DISTINCT enc_id FROM order_cpt)
SELECT
   enc_nbr_distinct.enc_id,
   cpt1.visit_cpt AS cpt1,
   cpt2.visit_cpt AS cpt2,
   cpt3.visit_cpt AS cpt3,
   cpt4.visit_cpt AS cpt4,
   cpt1_mod.cptmod,
   hcpcs1.visit_cpt AS hcpcs,
   hcpcs1_mod.cptmod AS hcpcmod
FROM enc_nbr_distinct
   LEFT JOIN cpt1 ON cpt1.enc_id = enc_nbr_distinct.enc_id
   LEFT JOIN cpt2 ON cpt2.enc_id = enc_nbr_distinct.enc_id
   LEFT JOIN cpt3 ON cpt3.enc_id = enc_nbr_distinct.enc_id
   LEFT JOIN cpt4 ON cpt4.enc_id = enc_nbr_distinct.enc_id
```

```
   LEFT JOIN cpt1_mod ON cpt1_mod.enc_id = enc_nbr_distinct.enc_id
   LEFT JOIN hcpcs1 ON hcpcs1.enc_id = enc_nbr_distinct.enc_id
   LEFT JOIN hcpcs1_mod ON hcpcs1_mod.enc_id = enc_nbr_distinct.enc_id
ORDER BY enc_nbr_distinct.enc_id;


--Pregnancies overlapping measurement period for universe patients
DROP TABLE IF EXISTS preg_mp_temp;
CREATE TEMPORARY TABLE preg_mp_temp AS
SELECT DISTINCT
   total_denominator_temp.patient_id,
   relevant_pregnancy.id AS preg_id,
   relevant_pregnancy.started_on,
   relevant_pregnancy.ended_on,
   relevant_pregnancy.last_menstrual_period :: DATE AS lmp_date,
   relevant_pregnancy.estimated_due_date :: DATE AS edd_date,
   relevant_deliveries.delivered_on :: DATE AS actual_date_of_delivery,
   COALESCE(relevant_deliveries.delivered_on,relevant_pregnancy.estimated_due_date)  :: DATE AS sort_date,
   CASE WHEN COALESCE(relevant_deliveries.delivered_on,relevant_pregnancy.estimated_due_date)  :: DATE <=
{{end_date}}
      THEN TRUE ELSE FALSE END AS delivery_edd_before_endmp
FROM total_denominator_temp
   INNER JOIN relevant_pregnancy ON relevant_pregnancy.patient_id = total_denominator_temp.patient_id
   LEFT JOIN relevant_deliveries ON relevant_deliveries.pregid = relevant_pregnancy.id
WHERE (relevant_pregnancy.started_on,
      relevant_pregnancy.ended_on + 1
   ) OVERLAPS (
      {{start_date}},
      {{end_date}} :: DATE + 1    );


--Choose one future delivery for the column "Estimated date of delivery (future)"
DROP TABLE IF EXISTS future_delivery_temp;
CREATE TEMPORARY TABLE future_delivery_temp AS
SELECT DISTINCT ON (patient_id)
   patient_id,
   preg_id,
   started_on,
   ended_on,
   lmp_date,
   edd_date,
   actual_date_of_delivery,
   sort_date
FROM preg_mp_temp
WHERE delivery_edd_before_endmp IS FALSE
ORDER BY patient_id, sort_date ASC;
```

```
--Choose one past pregnancy for the column "Actual (past) date of delivery within 12 months"
DROP TABLE IF EXISTS past_delivery_temp;
CREATE TEMPORARY TABLE past_delivery_temp AS
SELECT DISTINCT ON (patient_id)
   patient_id,
   preg_id,
   started_on,
   ended_on,
   lmp_date,
   edd_date,
   actual_date_of_delivery,
   sort_date
FROM preg_mp_temp
WHERE delivery_edd_before_endmp IS TRUE
ORDER BY patient_id, sort_date DESC;


--Make weeks/days gestation calculations and display other relevant pregnancy items for the report output
DROP TABLE IF EXISTS preg_columns_temp;
CREATE TEMPORARY TABLE preg_columns_temp AS
WITH preg_dates AS (
   SELECT total_denominator_temp.patient_id,
        total_denominator_temp.enc_date,
        total_denominator_temp.enc_id,
        COALESCE(past_delivery_temp.lmp_date, future_delivery_temp.lmp_date) AS calc_lmp_date,
        COALESCE(past_delivery_temp.edd_date, future_delivery_temp.edd_date) AS calc_edd_date,
        future_delivery_temp.edd_date                    AS future_edd_date,
        past_delivery_temp.actual_date_of_delivery       AS past_date_of_delivery,
        past_delivery_temp.lmp_date                      AS past_delivery_lmp
   FROM total_denominator_temp
        LEFT JOIN past_delivery_temp ON past_delivery_temp.patient_id = total_denominator_temp.patient_id
          AND total_denominator_temp.enc_date BETWEEN past_delivery_temp.started_on AND
past_delivery_temp.ended_on
        LEFT JOIN future_delivery_temp ON future_delivery_temp.patient_id =
total_denominator_temp.patient_id
          AND total_denominator_temp.enc_date BETWEEN future_delivery_temp.started_on AND
future_delivery_temp.ended_on
   WHERE past_delivery_temp.patient_id IS NOT NULL
     OR future_delivery_temp.patient_id IS NOT NULL
)
SELECT
   patient_id,
   enc_date,
   enc_id,
```

```
    future_edd_date,
    past_date_of_delivery,
    calc_lmp_date,
    calc_edd_date,
    TRUNC(DATE_PART('day', enc_date ::timestamp - calc_lmp_date :: timestamp)/7) || 'W' ||
        DATE_PART('day', enc_date ::timestamp - calc_lmp_date :: timestamp) -
        (TRUNC(DATE_PART('day', enc_date ::timestamp - calc_lmp_date :: timestamp)/7) * 7) || 'D'
        AS gestation_weeks_and_days_at_time_of_service,
    CASE WHEN NOT past_date_of_delivery IS NULL THEN
      TRUNC(DATE_PART('day', past_date_of_delivery ::timestamp - past_delivery_lmp ::timestamp)/7) || 'W' ||
        DATE_PART('day', past_date_of_delivery ::timestamp - past_delivery_lmp ::timestamp) -
        (TRUNC(DATE_PART('day', past_date_of_delivery ::timestamp - past_delivery_lmp ::timestamp)/7) * 7) ||
'D'
        END AS weeks_and_days_gestation_at_time_of_delivery
FROM preg_dates;


--Identify depression screenings from structured data. Priority order: PHQ-2, PHQ-9, Edinburgh, GDS (or other
screens)
DROP TABLE IF EXISTS depression_screenings_temp;
CREATE TEMPORARY TABLE depression_screenings_temp AS
SELECT    -- PHQ-2 add from Transformer (if similar exists) or pull directly from structured data
    total_denominator_temp.patient_id,
    total_denominator_temp.enc_id,
    '55758-7' AS loinc_code_screen,
    'PHQ2' AS tool_name,
    total_score :: VARCHAR AS screen_score,
    CASE WHEN result IS TRUE THEN 'Positive' ELSE 'Negative' END AS interp_result,
    1 AS priority
FROM total_denominator_temp
    INNER JOIN relevant_phq2 ON relevant_phq2.visit_id = total_denominator_temp.enc_id
UNION
SELECT    -- PHQ-9 add from Transformer (if similar exists) or pull directly from structured data
    total_denominator_temp.patient_id,
    total_denominator_temp.enc_id,
    '44261-6' AS loinc_code_screen,
    'PHQ9' AS tool_name,
    score :: VARCHAR  AS screen_score,
    CASE WHEN score :: INT >= 10 THEN 'Positive' ELSE 'Negative' END AS interp_result,
    2 AS priority
FROM total_denominator_temp
    INNER JOIN relevant_phq9 ON relevant_phq9.visit_id = total_denominator_temp.enc_id;
/*
UNION
    -- Edinburgh Depression Screen add from Transformer (if similar exists) or pull directly from structured data
SELECT
```

```
   total_denominator_temp.patient_id,
   total_denominator_temp.enc_id,
   '71354-5' AS loinc_code_screen,
   'EPDS' AS tool_name,
   score :: VARCHAR  AS screen_score,
   CASE WHEN result IS TRUE THEN 'Positive' ELSE 'Negative' END AS interp_result,
   3 AS priority
FROM total_denominator_temp
   INNER JOIN relevant_edinburgh_screens ON relevant_edinburgh_screens.visit_id =
total_denominator_temp.enc_id
UNION
 -- If your health center uses additional kinds of depression screening tools, add them below using UNION sub-
queries
 -- Otherwise, delete this section
SELECT     -- [Add name of tool here for easy reference]
   total_denominator_temp.patient_id,
   total_denominator_temp.enc_id,
   '48544-1' AS loinc_code_screen,        -- Add LOINC from Partnership ECDS instructions
   'GDSL' AS tool_name,                   -- Add tool name (or abbreviation)
   total_score :: VARCHAR  AS screen_score,
   CASE WHEN total_score :: INT >= 10 THEN 'Positive'
      ELSE 'Negative' END AS interp_result,      -- You may have to use a CASE WHEN statement. See Partnership ECDS
instructions
   4 AS priority                      -- Priority should start and 4 and increase to 5, 6, etc. depending on number of screens
FROM total_denominator_temp
   INNER JOIN(
         SELECT DISTINCT
            enc.patientid        AS patient_id,
            enc.encounterid      AS enc_id,
            structhpi.value :: INT      AS total_score
         FROM enc
            INNER JOIN structhpi
               ON structhpi.encounterid = enc.encounterid AND structhpi.catid = 250273  -- ID numbers are from your
system
                  AND structhpi.itemid = 18785 AND structhpi.detailid = 1516
                  AND structhpi.value ~ '\d+'
         WHERE enc.deleteflag = 0
            AND enc.status = 'CHK'
            AND structhpi.encounterid IS NOT NULL) AS gdsl_given ON gdsl_given.enc_id =
total_denominator_temp.enc_id
*/

--Unduplicate depression screening results and add row number
DROP TABLE IF EXISTS depression_screen_priorities_temp;
CREATE TEMPORARY TABLE depression_screen_priorities_temp AS
SELECT
```

```
   enc_id,
   patient_id,
   loinc_code_screen,
   tool_name,
   screen_score,
   interp_result,
   priority,
   ROW_NUMBER() OVER(PARTITION BY enc_id ORDER BY priority) AS row
FROM(SELECT DISTINCT ON (enc_id, priority)
      enc_id,
      patient_id,
      loinc_code_screen,
      tool_name,
      screen_score,
      interp_result,
      priority
   FROM depression_screenings_temp
   ORDER BY enc_id, priority, screen_score DESC) AS depression_screenings_temp_distinct_on;


--Prepare depression screening results so they can be entered into columns of the report output
DROP TABLE IF EXISTS depression_screen_columns_temp;
CREATE TEMPORARY TABLE depression_screen_columns_temp AS
SELECT DISTINCT
   depression_screen_priorities_temp.patient_id,
   depression_screen_priorities_temp.enc_id,
   total_denominator_temp.enc_date,
   first_columns.loinc_code_screen AS loinc_tool_1,
   first_columns.tool_name AS name_tool_1,
   first_columns.screen_score AS score_tool_1,
   first_columns.interp_result AS result_tool_1,
   second_columns.loinc_code_screen AS loinc_tool_2,
   second_columns.tool_name AS name_tool_2,
   second_columns.screen_score AS score_tool_2,
   second_columns.interp_result AS result_tool_2
FROM depression_screen_priorities_temp
   INNER JOIN total_denominator_temp ON total_denominator_temp.enc_id =
depression_screen_priorities_temp.enc_id
   INNER JOIN (SELECT * FROM depression_screen_priorities_temp
      WHERE row = 1) AS first_columns ON first_columns.enc_id = depression_screen_priorities_temp.enc_id
   LEFT JOIN (SELECT * FROM depression_screen_priorities_temp
      WHERE row = 2) AS second_columns ON second_columns.enc_id =
depression_screen_priorities_temp.enc_id
ORDER BY depression_screen_priorities_temp.enc_id;
```

```
--Identify a depression or other behavioral health condition code associated with the present visit Assessment
DROP TABLE IF EXISTS depression_dx_column_temp;
CREATE TEMPORARY TABLE depression_dx_column_temp AS
SELECT DISTINCT ON (total_denominator_temp.enc_id)
   total_denominator_temp.patient_id,
   total_denominator_temp.enc_id,
   relevant_diagnosis_codes.code AS icd_code
FROM total_denominator_temp
   INNER JOIN relevant_visit_diagnosis_codes ON relevant_visit_diagnosis_codes.visit_id =
total_denominator_temp.enc_id
   INNER JOIN relevant_diagnosis_codes ON relevant_diagnosis_codes.id =
relevant_visit_diagnosis_codes.diagnosis_code_id
   INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = relevant_diagnosis_codes.code
      AND hedis_value_set_codes.value_set_name = 'Depression or Other Behavioral Health Condition'
      AND latest = 'TRUE'
ORDER BY total_denominator_temp.enc_id, relevant_diagnosis_codes.code;


--Identify exclusion diagnosis codes (including depression) that occurred in the past two years
--Take the earliest visit date for each code and order all codes by their dates
DROP TABLE IF EXISTS exclusion_dx_temp;
CREATE TEMPORARY TABLE exclusion_dx_temp AS
SELECT
   patient_id,
   icd_code,
   first_visit_date,
   ROW_NUMBER() OVER(PARTITION BY patient_id ORDER BY first_visit_date, icd_code) AS row
FROM(SELECT
      total_denominator_temp.patient_id,
      relevant_diagnosis_codes.code AS icd_code,
      MIN(relevant_visits.visit_date) :: DATE AS first_visit_date
   FROM total_denominator_temp
      INNER JOIN relevant_visits ON relevant_visits.patient_id = total_denominator_temp.patient_id
      INNER JOIN relevant_visit_diagnosis_codes ON relevant_visit_diagnosis_codes.visit_id = relevant_visits.id
      INNER JOIN relevant_diagnosis_codes ON relevant_diagnosis_codes.id =
relevant_visit_diagnosis_codes.diagnosis_code_id
      INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = relevant_diagnosis_codes.code
         AND hedis_value_set_codes.value_set_name IN('Bipolar Disorder','Other Bipolar Disorder',
            'Personality Disorder','Pervasive Developmental Disorder','Psychotic Disorders',
            'Depression or Other Behavioral Health Condition')
         AND latest = 'TRUE'
   WHERE relevant_visits.visit_date BETWEEN ({{end_date}} :: DATE - INTERVAL '2 YEARS') AND {{end_date}}
   GROUP BY total_denominator_temp.patient_id, relevant_diagnosis_codes.code) AS raw_codes;
```

```
--Prepare the exclusion diagnosis codes by date of service for the report output
DROP TABLE IF EXISTS exclusion_columns_temp;
CREATE TEMPORARY TABLE exclusion_columns_temp AS
SELECT
   denom.patient_id,
   exclusions_1.icd_code AS exclusions_icddx10_1,
   exclusions_1.first_visit_date AS exclusions_icddx10_1_dos,
   exclusions_2.icd_code AS exclusions_icddx10_2,
   exclusions_2.first_visit_date AS exclusions_icddx10_2_dos,
   exclusions_3.icd_code AS exclusions_icddx10_3,
   exclusions_3.first_visit_date AS exclusions_icddx10_3_dos,
   exclusions_4.icd_code AS exclusions_icddx10_4,
   exclusions_4.first_visit_date AS exclusions_icddx10_4_dos,
   exclusions_5.icd_code AS exclusions_icddx10_5,
   exclusions_5.first_visit_date AS exclusions_icddx10_5_dos,
   exclusions_6.icd_code AS exclusions_icddx10_6,
   exclusions_6.first_visit_date AS exclusions_icddx10_6_dos,
   exclusions_7.icd_code AS exclusions_icddx10_7,
   exclusions_7.first_visit_date AS exclusions_icddx10_7_dos,
   exclusions_8.icd_code AS exclusions_icddx10_8,
   exclusions_8.first_visit_date AS exclusions_icddx10_8_dos,
   exclusions_9.icd_code AS exclusions_icddx10_9,
   exclusions_9.first_visit_date AS exclusions_icddx10_9_dos,
   exclusions_10.icd_code AS exclusions_icddx10_10,
   exclusions_10.first_visit_date AS exclusions_icddx10_10_dos,
   exclusions_11.icd_code AS exclusions_icddx10_11,
   exclusions_11.first_visit_date AS exclusions_icddx10_11_dos,
   exclusions_12.icd_code AS exclusions_icddx10_12,
   exclusions_12.first_visit_date AS exclusions_icddx10_12_dos,
   exclusions_13.icd_code AS exclusions_icddx10_13,
   exclusions_13.first_visit_date AS exclusions_icddx10_13_dos,
   exclusions_14.icd_code AS exclusions_icddx10_14,
   exclusions_14.first_visit_date AS exclusions_icddx10_14_dos,
   exclusions_15.icd_code AS exclusions_icddx10_15,
   exclusions_15.first_visit_date AS exclusions_icddx10_15_dos
FROM (SELECT DISTINCT patient_id FROM exclusion_dx_temp) AS denom
   INNER JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
         FROM exclusion_dx_temp
         WHERE row = 1) AS exclusions_1
            ON exclusions_1.patient_id = denom.patient_id
   LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
         FROM exclusion_dx_temp
         WHERE row = 2) AS exclusions_2
            ON exclusions_2.patient_id = denom.patient_id
   LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
         FROM exclusion_dx_temp
```

```
        WHERE row = 3) AS exclusions_3
          ON exclusions_3.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 4) AS exclusions_4
          ON exclusions_4.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 5) AS exclusions_5
          ON exclusions_5.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 6) AS exclusions_6
          ON exclusions_6.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 7) AS exclusions_7
          ON exclusions_7.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 8) AS exclusions_8
          ON exclusions_8.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 9) AS exclusions_9
          ON exclusions_9.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 10) AS exclusions_10
          ON exclusions_10.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 11) AS exclusions_11
          ON exclusions_11.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 12) AS exclusions_12
          ON exclusions_12.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 13) AS exclusions_13
          ON exclusions_13.patient_id = denom.patient_id
LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
        FROM exclusion_dx_temp
        WHERE row = 14) AS exclusions_14
          ON exclusions_14.patient_id = denom.patient_id
```

```
   LEFT JOIN (SELECT DISTINCT patient_id, first_visit_date, icd_code
            FROM exclusion_dx_temp
            WHERE row = 15) AS exclusions_15
              ON exclusions_15.patient_id = denom.patient_id
ORDER BY denom.patient_id;


--Determine if any of the exclusion diagnosis codes displayed in the exclusion columns have been resolved since
--    the time of service
DROP TABLE IF EXISTS exclusion_dx_resolved_temp;
CREATE TEMPORARY TABLE exclusion_dx_resolved_temp AS
SELECT
   exclusion_dx_temp.patient_id,
   icd_code AS exclusion_code,
   problemlist.resolvedon :: DATE AS date_resolved,
   exclusion_dx_temp.row AS exclusion_row_orig,
   ROW_NUMBER() OVER(PARTITION BY patient_id ORDER BY row) AS resolution_row
FROM exclusion_dx_temp
   INNER JOIN problemlist ON problemlist.patientid = exclusion_dx_temp.patient_id
   INNER JOIN itemdetail ON itemdetail.itemID = problemlist.asmtid AND propID = 13
      AND itemdetail.value = exclusion_dx_temp.icd_code
WHERE row BETWEEN 1 and 15
   AND resolvedon IS NOT NULL;


--Prepare the resolution columns for the report output
DROP TABLE IF EXISTS exclusion_resolved_columns_temp;
CREATE TEMPORARY TABLE exclusion_resolved_columns_temp AS
SELECT
   denom.patient_id,
   excl_resolve_1.exclusion_code AS exclusions_icddx10_1_resolution,
   excl_resolve_1.date_resolved AS exclusions_icddx10_1_resolution_date,
   excl_resolve_2.exclusion_code AS exclusions_icddx10_2_resolution,
   excl_resolve_2.date_resolved AS exclusions_icddx10_2_resolution_date,
   excl_resolve_3.exclusion_code AS exclusions_icddx10_3_resolution,
   excl_resolve_3.date_resolved AS exclusions_icddx10_3_resolution_date,
   excl_resolve_4.exclusion_code AS exclusions_icddx10_4_resolution,
   excl_resolve_4.date_resolved AS exclusions_icddx10_4_resolution_date
FROM(SELECT DISTINCT patient_id FROM exclusion_dx_resolved_temp) AS denom
   INNER JOIN (SELECT patient_id, exclusion_code,date_resolved
         FROM exclusion_dx_resolved_temp
         WHERE resolution_row = 1) AS excl_resolve_1
           ON excl_resolve_1.patient_id = denom.patient_id
   LEFT JOIN (SELECT patient_id, exclusion_code,date_resolved
         FROM exclusion_dx_resolved_temp
         WHERE resolution_row = 2) AS excl_resolve_2
```

```
        ON excl_resolve_2.patient_id = denom.patient_id
  LEFT JOIN (SELECT patient_id, exclusion_code,date_resolved
      FROM exclusion_dx_resolved_temp
      WHERE resolution_row = 3) AS excl_resolve_3
        ON excl_resolve_3.patient_id = denom.patient_id
  LEFT JOIN (SELECT patient_id, exclusion_code,date_resolved
      FROM exclusion_dx_resolved_temp
      WHERE resolution_row = 4) AS excl_resolve_4
        ON excl_resolve_4.patient_id = denom.patient_id;


--Report output
SELECT DISTINCT
   total_denominator_temp.enc_date AS date_of_service,
   NULL AS site_id_number,
   NULL AS provider_site_name,
   total_denominator_temp.visit_clinician_full_name,
   total_denominator_temp.clinician_npi_number,
   total_denominator_temp.subscriber_number AS member_cin,
   NULL AS membername,
   total_denominator_temp.memberfirstname,
   NULL AS membermiddlename,
   total_denominator_temp.memberlastname,
   total_denominator_temp.dob,
   total_denominator_temp.sex,
   cpt_columns_temp.cpt1,
   cpt_columns_temp.cpt2,
   cpt_columns_temp.cpt3,
   cpt_columns_temp.cpt4,
   cpt_columns_temp.cptmod,
   cpt_columns_temp.hcpcs,
   cpt_columns_temp.hcpcmod,
   CASE WHEN preg_columns_temp.enc_id IS NULL
     THEN 'No' ELSE 'Yes' END AS pregnant_at_time_of_visit,
   preg_columns_temp.future_edd_date AS estimated_date_of_delivery,
   preg_columns_temp.past_date_of_delivery AS actual_date_of_delivery,
   preg_columns_temp.gestation_weeks_and_days_at_time_of_service,
   CASE WHEN NOT del_date_postpart.past_date_of_delivery IS NULL
     AND del_date_postpart.past_date_of_delivery <= total_denominator_temp.enc_date
     THEN TRUNC(DATE_PART('day', total_denominator_temp.enc_date ::timestamp -
del_date_postpart.past_date_of_delivery :: timestamp)/7) || 'W' ||
       DATE_PART('day', total_denominator_temp.enc_date ::timestamp -
del_date_postpart.past_date_of_delivery :: timestamp) -
       (TRUNC(DATE_PART('day', total_denominator_temp.enc_date ::timestamp -
del_date_postpart.past_date_of_delivery :: timestamp)/7) * 7) || 'D'
       END AS weeks_and_days_postpartum,
```

```
preg_columns_temp.weeks_and_days_gestation_at_time_of_delivery,
depression_screen_columns_temp.loinc_tool_1 AS loinc_code_1,
depression_screen_columns_temp.name_tool_1 AS depression_screening_tool_name_1,
depression_screen_columns_temp.loinc_tool_2 AS loinc_code_2,
depression_screen_columns_temp.name_tool_2 AS depression_screening_tool_name_2,
depression_screen_columns_temp.score_tool_1 AS score_screening_tool_1,
depression_screen_columns_temp.result_tool_1 AS result_of_screening_1,
depression_screen_columns_temp.score_tool_2 AS score_screening_tool_2,
depression_screen_columns_temp.result_tool_2 AS result_of_screening_2,
depression_dx_column_temp.icd_code AS icddx10,
exclusion_columns_temp.exclusions_icddx10_1,
exclusion_columns_temp.exclusions_icddx10_1_dos,
exclusion_columns_temp.exclusions_icddx10_2,
exclusion_columns_temp.exclusions_icddx10_2_dos,
exclusion_columns_temp.exclusions_icddx10_3,
exclusion_columns_temp.exclusions_icddx10_3_dos,
exclusion_columns_temp.exclusions_icddx10_4,
exclusion_columns_temp.exclusions_icddx10_4_dos,
exclusion_columns_temp.exclusions_icddx10_5,
exclusion_columns_temp.exclusions_icddx10_5_dos,
exclusion_columns_temp.exclusions_icddx10_6,
exclusion_columns_temp.exclusions_icddx10_6_dos,
exclusion_columns_temp.exclusions_icddx10_7,
exclusion_columns_temp.exclusions_icddx10_7_dos,
exclusion_columns_temp.exclusions_icddx10_8,
exclusion_columns_temp.exclusions_icddx10_8_dos,
exclusion_columns_temp.exclusions_icddx10_9,
exclusion_columns_temp.exclusions_icddx10_9_dos,
exclusion_columns_temp.exclusions_icddx10_10,
exclusion_columns_temp.exclusions_icddx10_10_dos,
exclusion_columns_temp.exclusions_icddx10_11,
exclusion_columns_temp.exclusions_icddx10_11_dos,
exclusion_columns_temp.exclusions_icddx10_12,
exclusion_columns_temp.exclusions_icddx10_12_dos,
exclusion_columns_temp.exclusions_icddx10_13,
exclusion_columns_temp.exclusions_icddx10_13_dos,
exclusion_columns_temp.exclusions_icddx10_14,
exclusion_columns_temp.exclusions_icddx10_14_dos,
exclusion_columns_temp.exclusions_icddx10_15,
exclusion_columns_temp.exclusions_icddx10_15_dos,
exclusion_resolved_columns_temp.exclusions_icddx10_1_resolution,
exclusion_resolved_columns_temp.exclusions_icddx10_1_resolution_date,
exclusion_resolved_columns_temp.exclusions_icddx10_2_resolution,
exclusion_resolved_columns_temp.exclusions_icddx10_2_resolution_date,
exclusion_resolved_columns_temp.exclusions_icddx10_3_resolution,
exclusion_resolved_columns_temp.exclusions_icddx10_3_resolution_date,
```

```
    exclusion_resolved_columns_temp.exclusions_icddx10_4_resolution,
    exclusion_resolved_columns_temp.exclusions_icddx10_4_resolution_date
FROM total_denominator_temp
    LEFT JOIN cpt_columns_temp ON cpt_columns_temp.enc_id = total_denominator_temp.enc_id
    LEFT JOIN preg_columns_temp ON preg_columns_temp.enc_id = total_denominator_temp.enc_id
    LEFT JOIN preg_columns_temp AS del_date_postpart ON del_date_postpart.patient_id =
total_denominator_temp.patient_id
        AND del_date_postpart.past_date_of_delivery IS NOT NULL
    LEFT JOIN depression_screen_columns_temp ON depression_screen_columns_temp.enc_id =
total_denominator_temp.enc_id
    LEFT JOIN depression_dx_column_temp ON depression_dx_column_temp.enc_id =
total_denominator_temp.enc_id
    LEFT JOIN exclusion_columns_temp ON exclusion_columns_temp.patient_id =
total_denominator_temp.patient_id
    LEFT JOIN exclusion_resolved_columns_temp ON exclusion_resolved_columns_temp.patient_id =
total_denominator_temp.patient_id
ORDER BY total_denominator_temp.memberlastname, total_denominator_temp.memberfirstname,
    total_denominator_temp.dob, total_denominator_temp.enc_date
```

Appendix D: SQL Code for the Report "ECDS: Unhealthy Alcohol Use Screening and Follow Up (ASF-E)" and Related Transformers/Data Elements

NEW Transformer/Data Element Pair(s) for Alcohol Screens

Configure depending on which screen(s) are used at your health center. There are three possible screens and three possible Transformer/Data Element pairs. Note that the ID numbers for catid, itemid, and detailed will be unique at each health center. One option to find these ID numbers is the report "RCHC List All Structured Data Items" (available on the RCHC Aggregate for eCW health centers).

The sample code for the Transformer/Data Element pair below is for the Audit C screen. The Audit screen and the single-question screen should take a similar approach. See the above section for this measure for a table with Transformer and Data Element names for the three screens. In the code below, the screen is located in Social History, but it may be in another structured data location at your health center.

Transformer: Build relevant_audit_c_screens

```
DROP TABLE IF EXISTS relevant_audit_c_screens;
CREATE TABLE relevant_audit_c_screens AS
SELECT DISTINCT
      enc.patientid AS patient_id,
      enc.date :: DATE AS performed_on,
      audit_c_score.value AS score,
      CASE WHEN audit_c_interpret.value = 'Positive' THEN TRUE ELSE FALSE END AS positive
FROM enc
   INNER JOIN structsocialhistory AS audit_c_score ON audit_c_score.encounterid = enc.encounterid
      AND audit_c_score.catid = 262111
      AND audit_c_score.itemid = 262128
      AND audit_c_score.detailid = 1004 --Use a unique combination of codes to identify the AUDIT-C score
      AND audit_c_score.value ~ '^\d'
   LEFT JOIN structsocialhistory AS audit_c_interpret ON audit_c_interpret.encounterid = enc.encounterid
      AND audit_c_interpret.catid = 262111
      AND audit_c_interpret.itemid = 262128
      AND audit_c_interpret.detailid = 1005 --Use a unique combination of codes to identify the AUDIT-C
interpretation
      AND NOT audit_c_interpret.value IS NULL
      AND NOT audit_c_interpret.value = ' '
WHERE enc.deleteflag = 0
```

```
    AND enc.status = 'CHK';
```

Data Element: AUDIT C Screens

```
SELECT DISTINCT
   patient_id,
   performed_on,
   score,
   positive
FROM relevant_audit_c_screens
```

NEW Transformer/Data Element Pair for Alcohol Follow-up Counseling

The report output displays alcohol follow-up counseling entered as a CPT code, a HCPCS code, an ICD-10 code, or into structured data. The CPT and HCPCS codes are extracted directly from claims and the ICD-10 codes from visit assessments by the report so no separate Transformers or Data Elements are needed for them. The alcohol follow-up Transformer/Data Element pair identifies alcohol counseling only from structured data. Similar to the alcohol screening Transformer, the location and ID numbers for the counseling item will be unique at each health center.

Transformer: Build relevant_alcohol_counseling

```
DROP TABLE IF EXISTS relevant_alcohol_counseling;
CREATE TABLE relevant_alcohol_counseling AS
SELECT DISTINCT
   enc.patientid AS patient_id,
   enc.date :: DATE AS performed_on
FROM enc
        INNER JOIN structpreventive ON structpreventive.encounterid = enc.encounterid
WHERE structpreventive.catid = 306460
   AND structpreventive.itemid = 306462
   AND structpreventive.detailid = 1021 --Use a unique combination of codes to identify the alcohol counseling item
   AND NOT structpreventive.value IS NULL
   AND NOT structpreventive.value = ' '
   AND NOT structpreventive.value ILIKE 'No%'
   AND enc.deleteflag = 0
        AND enc.status = 'CHK';
```

Configuring the RCHC Relevant QIP ECDS Reports

Data Element: Alcohol Counseling Or Other Followups

```
SELECT
   patient_id,
   performed_on
FROM relevant_alcohol_counseling
```

Report SQL

Database: Production Database

Description: This report displays data for the ECDS Unhealthy Alcohol Use Screening and Follow Up (ASF-E) measure that could be submitted to Partnership HealthPlan of California (2022 Edition)

Parameters: {{start_date}} and {{end_date}}

Report SQL Code:

```
/*************************

Report Name: ECDS: Unhealthy Alcohol Use Screening and Follow Up (ASF-E)

Code Edited By: Ben Fouts for RCHC

Description: This report displays data for submission to Partnership Healthplan. It should not be used to
   directly evaluate a Quality Measure or for case management purposes

RCHC Started: June 2022 (based on code from the Partnership SQL specifications document)

Version Date: July 25, 2022

Revision History: 2022 Version 1

Resources Used: patients, visits, providers, locations, genders, hedis_value_set_codes, visit_billing_codes,
   billing_codes, visit_diagnosis_codes, patient_diagnoses, diagnosis_codes, audit_c_screens, audit_screens,
   single_question_alcohol_use_screens, alcohol_counseling_or_other_followups

*************************/

-- Universe Definition:
```

```
-- 1) Visit date in the measurement period
-- 2) Patient age 12 years and older at the start of the measurement period
-- 3) Patient currently has Partnership HealthPlan insurance -- this is the default, but can be changed
--      by the programmer to join to internal tables containing Partnership patients from
--      monthly membership files during the measurement period
DROP TABLE IF EXISTS universe_temp_raw;
CREATE TEMPORARY TABLE universe_temp_raw AS
SELECT DISTINCT
   patients.id AS patient_id,
   patients.mrn,
   LEFT(patients.subscriber_number,10) AS Member_Key,
   patients.first_name AS Member_First_Name,
   patients.middle_name AS Member_Middle_Name,
   patients.last_name AS Member_Last_Name,
   patients.date_of_birth :: DATE AS DOB,
   EXTRACT(YEAR FROM age({{start_date}}, patients.date_of_birth :: DATE)) AS Age_startMP,
  CASE WHEN genders.name = 'Female' THEN 'F'
     WHEN genders.name = 'Male' THEN 'M'
     WHEN genders.name = 'Unknown' OR genders.name = 'Choose not to disclose' THEN 'U'
     WHEN genders.name IS NULL THEN 'U'
     ELSE 'O' END AS Sex,
   visits.id AS encounter_id,
   visits.visit_date :: DATE AS Date_of_Service,
   locations.name AS Site_Name,
   providers.first_name AS Clinician_First_Name,
   providers.middle_name AS Clinician_Middle_Name,
   providers.last_name AS Clinician_Last_Name,
   providers.npi AS Clinician_NPI_Number
FROM patients
   INNER JOIN visits ON visits.patient_id = patients.id
   INNER JOIN providers ON providers.id = visits.provider_id
   LEFT JOIN genders ON genders.id = patients.gender_id
   LEFT JOIN locations ON locations.id = visits.location_id
WHERE EXTRACT(YEAR FROM age({{start_date}}, patients.date_of_birth :: DATE)) >=12
   AND visits.visit_date BETWEEN {{start_date}} AND {{end_date}}
   AND patients.primary_insurance_group_id = 17; -- "PARTNERSHIP MANAGED CARE"


-- List alcohol counseling and other followup in structured data (one code per encounter ID)
DROP TABLE IF EXISTS alc_followup_struct_raw_temp;
CREATE TEMPORARY TABLE alc_followup_struct_raw_temp AS
SELECT DISTINCT
   universe_temp_raw.patient_id,
   alcohol_counseling_or_other_followups.performed_on,
   '413473000' :: VARCHAR AS snomed_code --This is the SNOMED code that most closely describes health center
counseling
```

```
FROM alcohol_counseling_or_other_followups
        INNER JOIN universe_temp_raw ON universe_temp_raw.patient_id =
alcohol_counseling_or_other_followups.patient_id;


--List alcohol counseling and other followup by Value Set CPT or HCPCS Value Set Codes
-- (one code per type per encounter ID)
DROP TABLE IF EXISTS alc_followup_claim_raw_temp;
CREATE TEMPORARY TABLE alc_followup_claim_raw_temp AS
SELECT DISTINCT ON (universe_temp_raw.encounter_id, hedis_value_set_codes.code_system_name)
   universe_temp_raw.encounter_id,
   universe_temp_raw.patient_id,
   hedis_value_set_codes.code_system_name AS code_system,
   billing_codes.code AS claim_code
FROM universe_temp_raw
   INNER JOIN visits ON visits.patient_id = universe_temp_raw.patient_id
   INNER JOIN visit_billing_codes ON visit_billing_codes.visit_id = visits.id
   INNER JOIN billing_codes ON billing_codes.id = visit_billing_codes.billing_code_id
   INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = billing_codes.code
WHERE visits.visit_date :: DATE BETWEEN {{start_date}} AND {{end_date}}
   AND hedis_value_set_codes.value_set_name = 'Alcohol Counseling or Other Follow Up Care'
   AND hedis_value_set_codes.latest = 'TRUE'
ORDER BY universe_temp_raw.encounter_id, hedis_value_set_codes.code_system_name, billing_codes.code;


-- List alcohol counseling and other followup by Value Set ICD-10 Value Set Codes (one code per encounter ID)
DROP TABLE IF EXISTS alc_followup_assess_raw_temp;
CREATE TEMPORARY TABLE alc_followup_assess_raw_temp AS
SELECT DISTINCT ON (universe_temp_raw.encounter_id)
   universe_temp_raw.encounter_id,
   universe_temp_raw.patient_id,
   diagnosis_codes.code AS assess_code
FROM universe_temp_raw
   INNER JOIN visits ON visits.patient_id = universe_temp_raw.patient_id
   INNER JOIN visit_diagnosis_codes ON visit_diagnosis_codes.visit_id = visits.id
   INNER JOIN diagnosis_codes ON diagnosis_codes.id = visit_diagnosis_codes.diagnosis_code_id
   INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = diagnosis_codes.code
WHERE visits.visit_date :: DATE BETWEEN {{start_date}} AND {{end_date}}
  AND hedis_value_set_codes.value_set_name = 'Alcohol Counseling or Other Follow Up Care'
  AND hedis_value_set_codes.latest = 'TRUE'
ORDER BY universe_temp_raw.encounter_id, hedis_value_set_codes.code_system_name, diagnosis_codes.code;


-- Identify start date of exclusion diagnoses from the Problem List and Assessments
DROP TABLE IF EXISTS exclusions_raw_temp;
CREATE TEMPORARY TABLE exclusions_raw_temp AS
```

```
SELECT
   patient_id,
   code_group,
   exclude_code,
   MIN(dx_date) AS first_dx_date,
   MAX(ended_on) AS resolution_date
FROM (
      SELECT universe_temp_raw.patient_id AS patient_id,
           hedis_value_set_codes.value_set_name AS code_group,
           diagnosis_codes.code AS exclude_code,
           patient_diagnoses.start_date AS dx_date,
           CASE WHEN patient_diagnoses.end_date <= {{end_date}}
             THEN patient_diagnoses.end_date END AS ended_on
      FROM universe_temp_raw
        INNER JOIN patient_diagnoses ON patient_diagnoses.patient_id = universe_temp_raw.patient_id
        INNER JOIN diagnosis_codes ON diagnosis_codes.id = patient_diagnoses.diagnosis_code_id
        INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = diagnosis_codes.code
      WHERE hedis_value_set_codes.value_set_name IN('Alcohol Use Disorder', 'Dementia')
        AND hedis_value_set_codes.latest = 'TRUE'
      UNION
      SELECT universe_temp_raw.patient_id AS patient_id,
           hedis_value_set_codes.value_set_name AS code_group,
           diagnosis_codes.code AS exclude_code,
           visits.visit_date :: DATE AS dx_date,
           NULL AS ended_on
      FROM universe_temp_raw
        INNER JOIN visits ON visits.patient_id = universe_temp_raw.patient_id
        INNER JOIN visit_diagnosis_codes ON visit_diagnosis_codes.visit_id = visits.id
        INNER JOIN diagnosis_codes ON diagnosis_codes.id = visit_diagnosis_codes.diagnosis_code_id
        INNER JOIN hedis_value_set_codes ON hedis_value_set_codes.code_value = diagnosis_codes.code
      WHERE visits.visit_date :: DATE BETWEEN {{start_date}} AND {{end_date}}
        AND hedis_value_set_codes.value_set_name IN('Alcohol Use Disorder', 'Dementia')
        AND hedis_value_set_codes.latest = 'TRUE'
    ) AS raw_exclusions
WHERE dx_date <= {{end_date}}
GROUP BY patient_id, code_group, exclude_code;


-- Add row numbers to exclusions so they can be put into output columns
DROP TABLE IF EXISTS exclusions_raw_final;
CREATE TEMPORARY TABLE exclusions_raw_final AS
SELECT
   patient_id,
   code_group,
   exclude_code,
   first_dx_date,
```

```
   resolution_date,
   ROW_NUMBER () OVER (
        PARTITION BY patient_id
        ORDER BY code_group DESC, resolution_date DESC, first_dx_date) AS row_excl
FROM exclusions_raw_temp;


-- Results query
-- First part displays individual visits with unhealthy alcohol screening and/or counseling/followup
-- Second part (after UNION) displays the exclusions independent of any visit
SELECT * FROM
   (
SELECT
   universe_temp_raw.Date_of_Service,
   universe_temp_raw.Site_Name,
   NULL AS Site_ID_Number,  --Needs to be added from service provider's system
   NULL AS Provider_Key,   --Needs to be added from service provider's system
   universe_temp_raw.Clinician_NPI_Number,
   NULL AS Clinician_Name,  --Optional
   universe_temp_raw.Clinician_First_Name,
   universe_temp_raw.Clinician_Middle_Name,
   universe_temp_raw.Clinician_Last_Name,
   universe_temp_raw.Member_Key,
   NULL AS Member_Name,    --Optional
   universe_temp_raw.Member_First_Name,
   universe_temp_raw.Member_Middle_Name,
   universe_temp_raw.Member_Last_Name,
   universe_temp_raw.DOB,
   universe_temp_raw.Sex,
   audit_score_temp.Audit_Test_Score,
   audit_score_temp.Audit_Score_Interpretation,
   auditc_score_temp.AuditC_Test_Score,
   auditc_score_temp.AuditC_Score_Interpretation,
   single_q_score_temp.Single_Q_Score,
   single_q_score_temp.Single_Q_Interpretation,
   couns_cpt_temp.Counseling_CPT,
   couns_hcpcs_temp.Counseling_HCPCS,
   couns_icd_temp.Counseling_ICD10,
   couns_struct_temp.Counseling_SNOMED,
   NULL AS Exclusions_ICDDX10_1,
   NULL AS Exclusions_ICDDX10_1_DOS,
   NULL AS Exclusions_ICDDX10_2,
   NULL AS Exclusions_ICDDX10_2_DOS,
   NULL AS Exclusions_ICDDX10_3,
   NULL AS Exclusions_ICDDX10_3_DOS,
   NULL AS Exclusions_ICDDX10_4,
```

```
  NULL AS Exclusions_ICDDX10_4_DOS,
  NULL AS Exclusions_ICDDX10_5,
  NULL AS Exclusions_ICDDX10_5_DOS,
  NULL AS Exclusions_ICDDX10_1_Resolution,
  NULL AS Exclusions_ICDDX10_2_Resolution,
  NULL AS Exclusions_ICDDX10_3_Resolution,
  NULL AS Exclusions_ICDDX10_4_Resolution,
  NULL AS Exclusions_ICDDX10_5_Resolution
FROM universe_temp_raw
  LEFT JOIN (SELECT patient_id,
            performed_on,
            score AS Audit_Test_Score,
            CASE WHEN positive THEN 'Positive'
              ELSE 'Negative' END AS Audit_Score_Interpretation
          FROM audit_screens) AS audit_score_temp
            ON audit_score_temp.patient_id = universe_temp_raw.patient_id
              AND audit_score_temp.performed_on :: DATE = universe_temp_raw.Date_of_Service :: DATE
   LEFT JOIN (SELECT patient_id,
            performed_on,
            score AS AuditC_Test_Score,
            CASE WHEN positive THEN 'Positive'
              ELSE 'Negative' END AS AuditC_Score_Interpretation
          FROM audit_c_screens) AS auditc_score_temp
            ON auditc_score_temp.patient_id = universe_temp_raw.patient_id
              AND auditc_score_temp.performed_on :: DATE = universe_temp_raw.Date_of_Service :: DATE
    LEFT JOIN (SELECT patient_id,
            performed_on,
            score AS Single_Q_Score,
            CASE WHEN positive THEN 'Positive'
              ELSE 'Negative' END AS Single_Q_Interpretation
          FROM single_question_alcohol_use_screens) AS single_q_score_temp
            ON single_q_score_temp.patient_id = universe_temp_raw.patient_id
              AND single_q_score_temp.performed_on :: DATE = universe_temp_raw.Date_of_Service :: DATE
  LEFT JOIN (SELECT encounter_id,
            claim_code AS Counseling_CPT
          FROM alc_followup_claim_raw_temp
          WHERE code_system = 'CPT') AS couns_cpt_temp
       ON couns_cpt_temp.encounter_id = universe_temp_raw.encounter_id
  LEFT JOIN (SELECT encounter_id,
            claim_code AS Counseling_HCPCS
          FROM alc_followup_claim_raw_temp
          WHERE code_system = 'HCPCS') AS couns_hcpcs_temp
       ON couns_hcpcs_temp.encounter_id = universe_temp_raw.encounter_id
  LEFT JOIN (SELECT encounter_id,
            assess_code AS Counseling_ICD10
          FROM alc_followup_assess_raw_temp) AS couns_icd_temp
```

```
        ON couns_icd_temp.encounter_id = universe_temp_raw.encounter_id
    LEFT JOIN (SELECT patient_id,
                      performed_on,
                      snomed_code AS Counseling_SNOMED
                 FROM alc_followup_struct_raw_temp) AS couns_struct_temp
                      ON couns_struct_temp.patient_id = universe_temp_raw.patient_id
                        AND couns_struct_temp.performed_on :: DATE = universe_temp_raw.Date_of_Service :: DATE
WHERE audit_score_temp.patient_id IS NOT NULL
    OR auditc_score_temp.patient_id IS NOT NULL
    OR single_q_score_temp.patient_id IS NOT NULL
    OR couns_cpt_temp.encounter_id IS NOT NULL
    OR couns_hcpcs_temp.encounter_id IS NOT NULL
    OR couns_icd_temp.encounter_id IS NOT NULL
    OR couns_struct_temp.patient_id IS NOT NULL
UNION
SELECT  -- This part is for exclusions that are not attatched to a specific visit
    NULL AS Date_of_Service,
    NULL AS Site_Name,
    NULL AS Site_ID_Number,
    NULL AS Provider_Key,
    NULL AS Clinician_NPI_Number,
    NULL AS Clinician_Name,
    NULL AS Clinician_First_Name,
    NULL AS Clinician_Middle_Name,
    NULL AS Clinician_Last_Name,
    undup_pts.Member_Key,
    NULL AS Member_Name,    --Optional
    undup_pts.Member_First_Name,
    undup_pts.Member_Middle_Name,
    undup_pts.Member_Last_Name,
    undup_pts.DOB,
    undup_pts.Sex,
    NULL AS Audit_Test_Score,
    NULL AS Audit_Score_Interpretation,
    NULL AS AuditC_Test_Score,
    NULL AS AuditC_Score_Interpretation,
    NULL AS Single_Q_Score,
    NULL AS Single_Q_Interpretation,
    NULL AS Counseling_CPT,
    NULL AS Counseling_HCPCS,
    NULL AS Counseling_ICD10,
    NULL AS Counseling_SNOMED,
    exclusions_1a.Exclusions_ICDDX10_1,
    exclusions_1a.Exclusions_ICDDX10_1_DOS,
    exclusions_2a.Exclusions_ICDDX10_2,
    exclusions_2a.Exclusions_ICDDX10_2_DOS,
```

```
    exclusions_3a.Exclusions_ICDDX10_3,
    exclusions_3a.Exclusions_ICDDX10_3_DOS,
    exclusions_4a.Exclusions_ICDDX10_4,
    exclusions_4a.Exclusions_ICDDX10_4_DOS,
    exclusions_5a.Exclusions_ICDDX10_5,
    exclusions_5a.Exclusions_ICDDX10_5_DOS,
    exclusions_1a.Exclusions_ICDDX10_1_Resolution,
    exclusions_2a.Exclusions_ICDDX10_2_Resolution,
    exclusions_3a.Exclusions_ICDDX10_3_Resolution,
    exclusions_4a.Exclusions_ICDDX10_4_Resolution,
    exclusions_5a.Exclusions_ICDDX10_5_Resolution
FROM (SELECT DISTINCT patient_id,
                Member_Key,
                Member_First_Name,
                Member_Middle_Name,
                Member_Last_Name,
                DOB,
                Sex
            FROM universe_temp_raw) AS undup_pts
    LEFT JOIN (SELECT patient_id,
                exclude_code AS Exclusions_ICDDX10_1,
                first_dx_date AS Exclusions_ICDDX10_1_DOS,
                resolution_date AS Exclusions_ICDDX10_1_Resolution
            FROM exclusions_raw_final
            WHERE row_excl = 1) AS exclusions_1a
        ON exclusions_1a.patient_id = undup_pts.patient_id
    LEFT JOIN (SELECT patient_id,
                exclude_code AS Exclusions_ICDDX10_2,
                first_dx_date AS Exclusions_ICDDX10_2_DOS,
                resolution_date AS Exclusions_ICDDX10_2_Resolution
            FROM exclusions_raw_final
            WHERE row_excl = 2) AS exclusions_2a
        ON exclusions_2a.patient_id = undup_pts.patient_id
    LEFT JOIN (SELECT patient_id,
                exclude_code AS Exclusions_ICDDX10_3,
                first_dx_date AS Exclusions_ICDDX10_3_DOS,
                resolution_date AS Exclusions_ICDDX10_3_Resolution
            FROM exclusions_raw_final
            WHERE row_excl = 3) AS exclusions_3a
        ON exclusions_3a.patient_id = undup_pts.patient_id
    LEFT JOIN (SELECT patient_id,
                exclude_code AS Exclusions_ICDDX10_4,
                first_dx_date AS Exclusions_ICDDX10_4_DOS,
                resolution_date AS Exclusions_ICDDX10_4_Resolution
            FROM exclusions_raw_final
            WHERE row_excl = 4) AS exclusions_4a
```

```
        ON exclusions_4a.patient_id = undup_pts.patient_id
    LEFT JOIN (SELECT patient_id,
                exclude_code AS Exclusions_ICDDX10_5,
                first_dx_date AS Exclusions_ICDDX10_5_DOS,
                resolution_date AS Exclusions_ICDDX10_5_Resolution
              FROM exclusions_raw_final
              WHERE row_excl = 5) AS exclusions_5a
        ON exclusions_5a.patient_id = undup_pts.patient_id
WHERE exclusions_1a.patient_id IS NOT NULL
    OR exclusions_2a.patient_id IS NOT NULL
    OR exclusions_3a.patient_id IS NOT NULL
    OR exclusions_4a.patient_id IS NOT NULL
    OR exclusions_5a.patient_id IS NOT NULL
    ) AS all_records
ORDER BY Member_Last_Name, Member_First_Name, Member_Middle_Name, Member_Key,
Exclusions_ICDDX10_1_DOS DESC, Date_of_Service
```