```
/*
Title: Asthma 1: Emergency Room Visits for Children between 3 and 21 with
       identifiable asthma
Original Author: Dr Lawrence Kleinman and the CAPQuaM Center for Excellence.
Modifications: Dr. Michael Cabana and Dr. Naomi Bardach and the IMPLEMENT team
Version written by: Robert Thombley
Last Edited Date: Oct 31, 2018
Description: SAS code to calculate Asthma 1 measure using
             administrative claims data.

Required Input Data Files: (see input file specification documentation
                           for more information) All input data should be
                           in SAS7BDAT format and conform to the
                           specifications listed.

        1) Eligibility Table: A list of all periods of continuous enrollment
                              periods longer than 3 months by member and
                              payer (aka insurance provider). Each row is a
                              continuous enrollment period for a given member
                              and insurance provider/payer. Records should not
                              overlap in time, but each member may have
                              multiple records due to enrollment with different
                              payers.

        2) Medical Claim Table: All medical claims for the relevant population.
                                Claims will be filtered by time, diagnosis code,
                                and age at service but can be pre-filtered to
                                cut down on size.
                                Data should be in long form such that each
                                record is at the patient-claim-diagnosis code
                                level (ie - for a claim that has 9 diagnosis
                                codes, you should have 9 records, indexed
                                by the DX order variable. All claim data other
                                than diagnosis code and DX order will be
                                repeated).

        3) Pharmacy Claim Table: All pharmacy claims for the relevant
                                 population. Claims should be at the prescription
                                 level, such that each record represents a
                                 single pharmacy prescription claim.
```

```
        4) HEDIS Measure asthma management medication lists.  These are the
                lists of NDC codes and drug information
                for "Asthma Medications" from the Hedis measure
                "Medication Management for People with Asthma" from
                BOTH the lookback year and the evaluation year. They can be
                accessed here:
                http://www.ncqa.org/hedis-quality-measurement/hedis-measures.

        5) Comorbidity Claims Table: All (unfiltered) medical claims for at
                                     least the patients in the measure cohort.
                                     These claims are in wide form and have 1
                                     record per claim.  Note - this is distinct
                                     from the formatting of the medical claim
                                     table (ie, for a claim that has 9 diagnosis
                                     codes, you should have 1 record, with all
                                     DX variables in one record)

Output:  ast1_out.asthma_1_data_&evalyear: This table is the SAS version of the
        long-form analytic file which has one
        record for every patient with identifiable asthma's eligible month
        during the eval year.

Comments: The age variable used is the age at the time of most recent
        qualifying denominator event for identifiable asthma.

*/

/********* BEGIN USER EDITABLE CONFIGURATION *********/

* Define the library name asmed for use throughout the codebase;

    libname ast1_in "INPUT/DIRECTORY/PATH";
    libname ast1_out "OUTPUT/DIRECTORY/PATH";

* Define analysis years;
    %let lookyear = 2014; * The lookback year;
    %let evalyear = 2015; * The evaluation year;
    %let file_suff = _15TR;

* Define beginning and ending dates;
    %let lookstart = 01Jan&lookyear.; *First date of the lookback year;
```

```
    %let evalstart = 01Jan&evalyear.; *First date of the evaluation year;
    %let evalend = 31Dec&evalyear.; *Last date of the evaluation year;


* PMCA (Comorbidity) configuration information;

    * SET NUMBER OF ICD9 FIELDS IN YOUR COMORBIDITY CLAIMS DATA;
    %let icdnum=4;

    *SET NAME OF SOURCE UNIQUE PERSON ID FOR THE COMORBDIITY CLAIMS DATA;
    %let sid=member_id;

    * SET NAME OF VARIABLES CONTAINING DIAGNOSIS CODES
      Assumes that multiple fields have the same root
      name plus numbers 1-&icdnum.  e.g. DX1 to DX25;
    %let icdvar=DX;

    * SET NAME OF UNIQUE CLAIM CODE;
    %let claimid=claim_header_id;


* Source Data Tables;

    %let eligibility_table = ast1_in.eligible_15tr;
    %let medical_claims = ast1_in.claims_15tr;
    %let pharmacy_claims = ast1_in.pclaims_15tr;
    %let asthma_meds_yearEvaluation = ast1_in.asm_2015;
    %let asthma_meds_yearLookback = ast1_in.asm_2014;
    %let comorbidity_claims = ast1_out.como_15tr_pmca;


/* Define the maximum number of diagnosis positions to include for the
    "DX any positions" events */
    %let maximum_num_dx = 4;


/********* END USER EDITABLE CONFIGURATION *********/

/********* BEGIN INTERNAL CONFIGURATION - DO NOT EDIT *********/

* Measure inclusion criteria. Do not edit;
```

```
* CPT & Revenue Codes;
%let hospital_cpt = '99221','99222','99223','99231','99238','99239',
                    '99356','99357','99218','99219','99220','99232',
                    '99233','99234','99235','99236';
%let hospital_rev_code = '0110','0111','0112','0113','0114','0117','0119',
                           '0120','0121','0122','0123','0124','0127',
                           '0129','0130','0131','0132','0133','0134',
                           '0137','0139','0150','0151','0152','0153',
                           '0154','0157','0159','0200','0201','0202',
                           '0203','0204','0206';
%let office_visit_cpt = '99201','99202','99203','99204','99205','99211',
                        '99212','99213','99214','99215';
%let ed_visit_cpt = '99281','99282','99283','99284','99285';
%let ed_visit_rev_code = '0450','0451','0452','0456','0459','0981';

* ICD9/10 Inclusion Codes;
* ICD9 = 493.*;
%let icd9_inclusion_asthma ='49300','49301','49302','49310','49311',
                            '49312','49320','49321','49322',
                            '49381','49382','49390','49391','49392';
* icd10 = J45.*;
%let icd10_inclusion_asthma = 'J4520','J4521','J4522','J4530','J4531',
                               'J4532','J4540','J4541','J4542','J4550',
                               'J4551','J4552','J45901','J45902','J45909',
                               'J45990','J45991','J45998';

* ICD9/10 Exclusion codes.;
%let icd9_exclusion = '27700','27701','27702','27703','27709','4920',
                      '4928','5182','5064','5181','7702';
%let icd10_exclusion = 'E849','E8411','E840','E841','E8419','E848','J439',
                       'J438','J430','J431','J432','J983','J684','J982',
                       'P250';

* The minimum and maximum ages to include;

%let min_included_age_at_service = 3;
%let max_included_age_at_service = 21;

/* Define the age used in the Identifiable Asthma: Critieria B of the
technical specifications (default is 5 years)*/
```

```
    %let criteria_b_age = 5;


/* Identify the last date in February for the evaluation year, since it may
    change if it is a leapyear. The value is stored as the global
    variable &last_feb_day.
*/
    data _NULL_;
        end_day = day(intnx('MONTH',mdy(2,1,&evalyear.),0,'end'));
        call symput('last_feb_day', put(end_day, 2.));
    run;


/********* END INTERNAL CONFIGURATION *********/

/********* BEGIN TABLE SETUP *********/


/* Create reference tables for use throughout the measure. */

    /* Create list of all unique member/payer enrollment periods
       within the study period.*/
    proc sql;
        CREATE TABLE enrollment_log as
        SELECT DISTINCT member_id,
                        payer_id,
                        medicaid_indicator,
                        insurance_type,
                        gender_code,
                        payer_start_dt,
                        payer_end_dt
        FROM &eligibility_table.
        WHERE payer_end_dt between "&lookstart."d AND "&evalend."d;
    quit;


    /* Filter all medical claims to include only those having:
        - diag. codes matching the measure inclusion critieria diagnosis codes
        - member/payer ids match those in the enrollment_log table
        - claim service date is within the measure time frame
        - claim service date is within the payer enroll period for the member
```

```
*/
proc sql;
    CREATE TABLE all_ast_clms_plus_enroll AS
    SELECT DISTINCT a.member_id,
        a.service_dt,
        a.claim_header_id,
        a.zip_code,
        a.diagnosis_code,
        a.dx_order,
        a.procedure_code,
        a.revenue_code,
        a.payer_id,
        a.age_at_service,
        b.payer_start_dt,
        b.payer_end_dt
    FROM &medical_claims. a
    INNER JOIN enrollment_log b
    ON a.member_id = b.member_id AND a.payer_id = b.payer_id
    WHERE a.diagnosis_code IN (&icd9_inclusion_asthma.,
                               &icd10_inclusion_asthma.)
        AND a.service_dt BETWEEN "&lookstart"d AND "&evalend"d
        AND a.service_dt BETWEEN b.payer_start_dt AND b.payer_end_dt;
quit;

/* Create a reference list of asthma medications by merging
    HEDIS ASM-C/ MMA-A NDC code lists. We choose to include
    all unique NDCs during both the lookback year and the eval year,
    excluding any NDCs for leukotriene modifiers, short acting inhaled
    beta agonists and the specific drug indacaterol.
*/

data asthma_reference_rx;
    set &asthma_meds_yearLookback (keep=NDC_CODE
                                       GENERIC_PRODUCT_NAME
                                       DESCRIPTION
                                       DRUG_ID)
        &asthma_meds_yearEvaluation (keep=NDC_CODE
                                         GENERIC_PRODUCT_NAME
                                         DESCRIPTION
                                         DRUG_ID);
```

```
        /* keep record if it does not match one of the following exclusions:
            - A product name like 'indacaterol'
            - A descrioption like 'leukotriene modifier'
            - A description like 'short acting inhaled beta'
        */
        if find(GENERIC_PRODUCT_NAME, 'indacaterol', 'i')
            + find(DESCRIPTION, 'leukotriene modifier', 'i')
            + find(DESCRIPTION,  'short-acting inhaled beta', 'i') = 0;
run;

* Remove duplicate NDC_CODES;
proc sort data=asthma_reference_rx nodupkey;
    BY NDC_CODE;
run;

/* Extract all pharmacy claims that include one of the NDC codes from
    the HEDIS reference lists and occur within the measure time frame.
*/
proc sql;
    CREATE TABLE asthma_rx_tmp as
    SELECT claims.* FROM &pharmacy_claims. claims
    INNER JOIN asthma_reference_rx ref
    ON ref.ndc_code = claims.ndc_code
    WHERE service_dt BETWEEN "&lookstart."d AND "&evalend."d;
quit;


/* Filter all pharmacy claims to include only those having:
    - member/payer ids match those in the enrollment_log table
    - claim service date is within the payer enroll period for the member
*/
proc sql;
    CREATE TABLE all_rx_clms_plus_enroll as
    SELECT DISTINCT rx.member_id,
                    rx.service_dt,
                    rx.age_at_service,
                    rx.ndc_code,
                    rx.payer_id,
                    enr.payer_start_dt,
                    enr.payer_end_dt
    FROM asthma_rx_tmp rx
```

```
        INNER JOIN enrollment_log enr
        ON rx.member_id = enr.member_id AND rx.payer_id = enr.payer_id
        AND rx.service_dt BETWEEN enr.payer_start_dt AND enr.payer_end_dt;
    quit;


    /* Filter all medical claims to include only those having:
        - diag codes matching the measure exclusion critieria diagnosis codes
    */
    proc sql;
        CREATE TABLE all_excluded as
        SELECT DISTINCT clms.member_id as member_id,
                        clms.service_dt as excl_service_dt,
                        clms.diagnosis_code
        FROM &medical_claims. clms
        WHERE clms.service_dt between "&lookstart."d AND "&evalend."d
        AND clms.diagnosis_code in (&icd9_exclusion., &icd10_exclusion.);
    quit;

/********* END TABLE SETUP *********/

/********* BEGIN DENOMINATOR CALCULATION *********/

/*  Calculating Denominator - Identify for each month of the reporting year,
                              those kids aged min_included_age_at_service
                              to max_included_age_at_service (default: 3-21,
                              at the time of service) who meet the
                              definition(s) of having identifiable asthma.
*/

* Extract some of the qualifying events from the processed claims.
* Note: These are the raw events,
* qualification logic will be performed in a later step.;


    /* Denominator Qualifying Event #1: Prior hospitalization with asthma as
                                        the primary or secondary diagnosis.*/
    proc sql;
        CREATE TABLE hosp_asth_top2_tmp as
        SELECT a.* FROM all_ast_clms_plus_enroll a
        WHERE (a.procedure_code IN (&hospital_cpt.)
```

```
            OR a.revenue_code IN (&hospital_rev_code.)
            )
        AND (a.diagnosis_code IN (&icd9_inclusion_asthma.)
            OR a.diagnosis_code IN (&icd10_inclusion_asthma.)
            )
        AND dx_order IN (1,2)
    ORDER BY member_id, claim_header_id, service_dt;
quit;

* De-duplicate by claim_header_id;
data hosp_asth_top2;
    set hosp_asth_top2_tmp;
    by member_id claim_header_id;
    if first.claim_header_id;
run;


/* Denominator Qualifying Event 2: Ambulatory (ED or Outpatient) event
                                    in which asthma was the primary dx.*/
proc sql;
    CREATE TABLE amb_asth_top1 as
    SELECT a.* FROM all_ast_clms_plus_enroll a
    WHERE (a.procedure_code IN (&office_visit_cpt.)
            OR a.revenue_code IN (&ed_visit_rev_code.)
            OR a.procedure_code IN (&ed_visit_cpt.)
            )
        AND a.diagnosis_code IN (&icd9_inclusion_asthma.,
                                 &icd10_inclusion_asthma.)
        AND dx_order IN (1)
    ORDER BY member_id, service_dt;
quit;

/* Denominator Qualifying Events #3: Ambulatory (ED or Outpatient) event in
                                    which asthma dx codes appear in any
                                    position. The definition of
                                    "any position" is any dx code with an
                                    order lower than the &maximum_num_dx
                                    variable (default value is 9).
*/
proc sql;
    CREATE TABLE amb_asth_all as
```

```
    SELECT a.* FROM all_ast_clms_plus_enroll a
    WHERE (a.procedure_code IN (&office_visit_cpt.)
            OR a.revenue_code IN (&ed_visit_rev_code.)
            OR a.procedure_code IN (&ed_visit_cpt.)
            )
        AND a.diagnosis_code IN (&icd9_inclusion_asthma.,
                                 &icd10_inclusion_asthma.)
        AND dx_order <= &maximum_num_dx.
    ORDER BY member_id, service_dt;
quit;

* Denominator Qualifying Events #4: Prescription drug claims.
* These do not need to be extracted and are already
* available in the table work.all_rx_claims_plus_enroll.;

* Combine all qualifying events into one large events table;
data all_events;
    set hosp_asth_top2 (keep=member_id
                            service_dt
                            zip_code
                            age_at_service
                            payer_id IN=a)
        amb_asth_top1 (keep=member_id
                            service_dt
                            zip_code
                            age_at_service
                            payer_id IN=b)
        amb_asth_all (keep=member_id
                            service_dt
                            zip_code
                            age_at_service
                            payer_id IN=c)
        all_rx_clms_plus_enroll (keep=member_id
                                      service_dt
                                      age_at_service
                                      payer_id
                                      rename=(payer_id=rx_payer_id) IN=d);
        attrib event length=$4;

        if a then event = 'Inpt'; *Inpatient;
        if b then event = 'EOA1'; *Ed/Outpatient Asthma: Primary DX Only;
```

```
            if c then event = 'EOAd'; *Ed/Outpatient Asthma: any Dx order;
            if d then do;
                zip_code = .;
                event = 'RX';   *Pharmacy derived;
            end;
    run;


    /* De-duplicate based on member_id, event type, and service date to create
        unique service events for each member_id. Multiple claims for the same
        event type on a single day will only be counted once. */

    proc sort data = all_events out = all_events_clean nodupkey;
        by member_id service_dt event;
    run;


/* Build monthly events tables based on month of service date.  These are
    cumulative such that January's events table includes all claims from
    1/1/lookback_year to 12/31/lookback_year, while December's events table
    includes all claims from 1/1/lookback_year to 11/30/eval_year.
    These tables do not include data from the month in question.

    This macro also filters kids on age.
*/

    %MACRO ServMonth(mth, start_date, end_date);

        * Build monthly member/service/event tables;
        data &Mth;
            set all_events_clean;
            where (&start_date. le service_dt le &end_date.);
            if age_at_service lt &min_included_age_at_service.
                or age_at_service gt &max_included_age_at_service. then delete;
        run;
    %MEND ServMonth;

    * Run for each month in the evaluation year;
    %ServMonth (Jan,"&lookstart."d,"31Dec&lookyear."d);
    %ServMonth (Feb,"&lookstart."d,"31Jan&evalyear."d);
```

```
    *last day of Feb changes with year (see config section);
    %ServMonth (Mar,"&lookstart."d,"&last_feb_day.Feb&evalyear."d);
    %ServMonth (Apr,"&lookstart."d,"31Mar&evalyear."d);
    %ServMonth (May,"&lookstart."d,"30Apr&evalyear."d);
    %ServMonth (Jun,"&lookstart."d,"31May&evalyear."d);
    %ServMonth (Jul,"&lookstart."d,"30Jun&evalyear."d);
    %ServMonth (Aug,"&lookstart."d,"31Jul&evalyear."d);
    %ServMonth (Sep,"&lookstart."d,"31Aug&evalyear."d);
    %ServMonth (Oct,"&lookstart."d,"30Sep&evalyear."d);
    %ServMonth (Nov,"&lookstart."d,"31Oct&evalyear."d);
    %ServMonth (Dec,"&lookstart."d,"30Nov&evalyear."d);



/* Exclude kids with a previous diagnosis in the exclusion list,
    as of the last day of
    the preceeding month.
*/
    %MACRO Exclude (Mth, end_date);
        * Identify kids to exclude;
        proc sql;
            CREATE TABLE &Mth._Exclude AS
            SELECT DISTINCT clms.member_id
            FROM all_excluded clms
            WHERE clms.excl_service_dt <= &end_date.;
        quit;

        * Exclude them from the evaluation claims;
        proc sql;
            CREATE TABLE &Mth._Clean AS
            SELECT * FROM &Mth.
            WHERE member_id NOT IN (SELECT member_id FROM &Mth._Exclude);
        quit;

    %MEND Exclude;

    * Run for each month in the evaluation year;
    %Exclude (Jan,"31Dec&lookyear"d);
    %Exclude (Feb,"31Jan&evalyear"d);
    %Exclude (Mar,"&last_feb_day.Feb&evalyear"d);
    %Exclude (Apr,"31Mar&evalyear"d);
```

```
    %Exclude (May,"30Apr&evalyear"d);
    %Exclude (Jun,"31May&evalyear"d);
    %Exclude (Jul,"30Jun&evalyear"d);
    %Exclude (Aug,"31Jul&evalyear"d);
    %Exclude (Sep,"31Aug&evalyear"d);
    %Exclude (Oct,"30Sep&evalyear"d);
    %Exclude (Nov,"31Oct&evalyear"d);
    %Exclude (Dec,"30Nov&evalyear"d);

* Next, in each month's table, create a tally of events by type for each member;

    %MACRO Cnt (Mth);

        * Create dummy variables for each event;
        data &Mth._count (drop=i);
            set &Mth._Clean (keep = member_id age_at_service zip_code service_dt
                                    payer_id rx_payer_id event);
            array ind[4] num_inpatient num_amb_ast_1 num_amb_ast_any num_rx;
            array ev[4] $ _TEMPORARY_ ("Inpt", "EOA1","EOAd","RX");
            do i=1 to 4;
                ind[i] = ifn(event = ev[i], 1, 0);
            end;
        run;

        * Count event type totals by member;
        proc sql;
        create table &Mth._mem_tots as
            select MEMBER_ID,
                    sum(num_inpatient) as tot_inpatient,
                    sum(num_amb_ast_1) as tot_amb_ast_1,
                    sum(num_amb_ast_any) as tot_amb_ast_any,
                    sum(num_rx) as tot_rx
            from &Mth._count
            group by MEMBER_ID;
        quit;

        /* Since measure is attributed to the most recent qualifying payer,
            identify the claim level payer details
            associated with that payer's most recent qualifying event,
            for each member.
        */
```

```
proc sql;
    create table &Mth._payer_details as
    select a.member_id,
            a.payer_id,
            a.zip_code,
            a.age_at_service,
            a.service_dt
    from &Mth._count a
    inner join
    (
        select member_id, max(service_dt) as last_medical_service_dt
        from &Mth._count
        where not missing(payer_id)
        group by member_id
    ) b
    on a.member_id = b.member_id
        and a.service_dt = b.last_medical_service_dt
    where not missing(payer_id)
    order by member_id, payer_id, service_dt desc;
quit;

* In case there are any ties from above,
* output only the 1st row for each member;
data &Mth._payer_unique;
    set &Mth._payer_details;
    by member_id;
    if first.member_id then output;
run;

/* Join the member level totals with the most recent member-payer
    level details. This creates a table that identifies event totals
    per member and attributes them to the most recent payer
    for that member.
*/
proc sql;
    create table &Mth._total_mem_level as
    select a.member_id,
           a.payer_id,
           a.zip_code,
           a.age_at_service,
           a.service_dt as last_medical_service_dt,
```

```
                    b.tot_inpatient,
                    b.tot_amb_ast_1,
                    b.tot_amb_ast_any,
                    b.tot_rx
            from &Mth._payer_unique a
            inner join &Mth._mem_tots b
            on a.member_id = b.member_id;
        quit;
    %mend;

    * Run the macro for each month;
    %Cnt(Jan);
    %Cnt(Feb);
    %Cnt(Mar);
    %Cnt(Apr);
    %Cnt(May);
    %Cnt(Jun);
    %Cnt(Jul);
    %Cnt(Aug);
    %Cnt(Sep);
    %Cnt(Oct);
    %Cnt(Nov);
    %Cnt(Dec);


/* Using the member-level event totals table, apply the logic outlined in the
    tech specs to evaluate whether each member qualifies as having identifiable
    asthma. This is where the patient is determined to have identifiable asthma
    or not.
*/

    %MACRO Ident_asth (Mth);
    /* 2 data sets - identifiable asthma & non-identifiable asthma */
    data &Mth._Ident &Mth._NoIdent;
        set &Mth._Total_mem_level;
        /* criteria a */
        if tot_inpatient ge 1 then output &Mth._Ident;
        /* criteria c.i & criteria c.ii */
        else if (tot_amb_ast_any ge 3)
                or
                ((tot_amb_ast_any ge 2) and tot_rx ge 1)
```

```
                then output &Mth._Ident;

        else if age_at_service ge &criteria_b_age. then do;
            /* criteria b.i */
            if tot_amb_ast_1 ge 1 then output &Mth._Ident;
            /* criteria b.ii */
            else if tot_amb_ast_any ge 2 then output &Mth._Ident;
            /* criteria b.iii */
            else if (tot_amb_ast_any ge 1 and tot_rx ge 1)
                then output &Mth._Ident;
            else output &Mth._NoIdent;
        end;
        else output &Mth._NoIdent;
    run;

    %MEND Ident_asth;

    * Identify patients with "identifiable asthma" by month;
    %Ident_asth(Jan);
    %Ident_asth(Feb);
    %Ident_asth(Mar);
    %Ident_asth(Apr);
    %Ident_asth(May);
    %Ident_asth(Jun);
    %Ident_asth(Jul);
    %Ident_asth(Aug);
    %Ident_asth(Sep);
    %Ident_asth(Oct);
    %Ident_asth(Nov);
    %Ident_asth(Dec);

/* Finally, this macro checks whether patients are continuously enrolled with
    the attributed payer for 3 months. This macro is run monthly, so the three
    month enrollment window is rolling such that patients are required to be
    enrolled at least two months before the current month though the end of
    the current month.
*/

    %macro checkEligibility(Mth, ce_start_date, ce_end_date);
        proc sql;
            CREATE TABLE &mth._ident_CE as
```

```
            SELECT a.*, "&mth." as month, b.payer_id as eligibility_payer,
                   b.payer_start_dt,
                   b.payer_end_dt
            FROM &mth._ident a
            inner join enrollment_log b
            on a.payer_id = b.payer_id and a.member_id = b.member_id
            WHERE b.payer_start_dt <= &ce_start_date.
            and b.payer_end_dt >= &ce_end_date.
            ORDER BY member_id;
    quit;
%mend;

* Include only continuously eligible patients according to the supplied
* date boundaries;
%checkEligibility(Jan, "01Nov&lookyear."d, "31Jan&evalyear."d);
%checkEligibility(Feb, "01Dec&lookyear."d, "&last_feb_day.Feb&evalyear."d);
%checkEligibility(Mar, "01Jan&evalyear"d, "31Mar&evalyear"d);
%checkEligibility(Apr, "01Feb&evalyear"d, "30Apr&evalyear"d);
%checkEligibility(May, "01Mar&evalyear"d, "31May&evalyear"d);
%checkEligibility(Jun, "01Apr&evalyear"d, "30Jun&evalyear"d);
%checkEligibility(Jul, "01May&evalyear"d, "31Jul&evalyear"d);
%checkEligibility(Aug, "01Jun&evalyear"d, "31Aug&evalyear"d);
%checkEligibility(Sep, "01Jul&evalyear"d, "30Sep&evalyear"d);
%checkEligibility(Oct, "01Aug&evalyear"d, "31Oct&evalyear"d);
%checkEligibility(Nov, "01Sep&evalyear"d, "30Nov&evalyear"d);
%checkEligibility(Dec, "01Oct&evalyear"d, "31Dec&evalyear"d);

* Create a denominator reference table that concatenates all of the monthly
* denominator tables into a single table;
data ast1_out.denominator_raw_data;
    set jan_ident_ce feb_ident_ce mar_ident_ce apr_ident_ce
        may_ident_ce jun_ident_ce jul_ident_ce aug_ident_ce
        sep_ident_ce oct_ident_ce nov_ident_ce dec_ident_ce;
    format eval_start_dt date9.;
    format eval_end_dt date9.;
    eval_start_dt = input("01" || month || "&evalyear.", date9.);
    eval_end_dt = intnx('month',eval_start_dt,0,'end');
run;

proc sort data=ast1_out.denominator_raw_data;
    by member_id payer_id;
```

```
    run;

/********* END DENOMINATOR CALCULATION  *********/

/********* BEGIN NUMERATOR CALCULATION *********/

/*  Calculating Numerator - Identify, for each month of the reporting year,
                            those kids in the denominator who had an
                            inpatient hospitalization or ED visit.
*/

    %MACRO calculateNumerator(Mth, BeginDt, EndDt);
        /* ED visits: Filter all medical claims to include only those having:
            - diag codes matching the measure inclusion critieria diag codes
            - a primary or secondary diagnosis ordering
            - an ED visit CPT or Revenue code
            - claim service date is within the monthly time frame
            - an age between the minimum and maximum allowable ages
        */
        proc sql;
            create table &Mth._ed as
                select clm.*, "EDep" as eventtype
                from all_ast_clms_plus_enroll clm
                where diagnosis_code IN (&icd9_inclusion_asthma.,
                                         &icd10_inclusion_asthma.)
                        AND dx_order IN (1,2)
                        AND (procedure_code IN (&ed_visit_cpt.)
                                OR  revenue_code IN (&ed_visit_rev_code.)
                            )
                        AND clm.service_dt BETWEEN &BeginDt and &EndDt
                        AND age_at_service
                            BETWEEN &min_included_age_at_service.
                                and &max_included_age_at_service.;
        quit;

        /* Hospitalizations: Filter all med claims to include only those having:
            - diag codes matching the measure inclusion critieria diag codes
            - a primary or secondary diagnosis ordering
            - an inpatient hospitalization CPT or Revenue code
            - claim service date is within the monthly time frame
            - an age between the minimum and maximum allowable ages
```

```
*/

proc sql;
    create table &Mth._hosp as
        select clm.*, "HOSP" as eventtype
        from all_ast_clms_plus_enroll clm
        where diagnosis_code IN (&icd9_inclusion_asthma.,
                                 &icd10_inclusion_asthma.)
                AND dx_order IN (1,2)
                AND (procedure_code IN (&hospital_cpt.)
                        OR revenue_code IN (&hospital_rev_code.)
                        )
                AND clm.service_dt BETWEEN &BeginDt and &EndDt
                AND age_at_service
                    BETWEEN &min_included_age_at_service.
                        and &max_included_age_at_service.;
quit;

/* Deduplicate hospitalization dates by claim_header_id.
    This is the same logic as used in the denominator calculation for
    de-duplicating inpatient events data.
*/
proc sort data = &Mth._hosp;
    by member_id claim_header_id service_dt;
run;

data &Mth._hosp_dedup;
    set &Mth._hosp;
    by member_id claim_header_id;
    if first.claim_header_id;
run;

* Now combine ED and hospitalization results;
data &Mth._ed_hosp;
    set &Mth._ed  &Mth._hosp_dedup;
run;

/* Restrict to those kids who qualify for the denominator during this
    month, including only numerator events attributed to the
    denominator payer.
*/
```

```
    proc sql;
        create table &Mth._numerator as
        select distinct a.*
        from &Mth._ed_hosp a
        inner join &Mth._ident_ce b
        on a.payer_id = b.payer_id and a.member_id = b.member_id
        and b.payer_start_dt <= &BeginDt. and b.payer_end_dt >= &EndDt.
        order by a.member_id, a.payer_id, a.service_dt;
    quit;

    * De-duplicate so that there is only one record on a given service date
        for each event type;
    proc sort data=&Mth._numerator;
        by member_id service_dt eventtype;
    run;

    data &Mth._numerator_clean;
        set &Mth._numerator;
        by member_id service_dt eventtype;
        if first.eventtype;
    run;

/* Remove any hospitalizations that occur within one day of an ED visit or
    of one another because these are considered to be the same occurrence.
    ED visits for subsequent days are allowed.
*/
    proc sort data=&Mth._numerator_clean;
        by member_id service_dt eventtype;
    run;

    data &Mth._numerator_clean2;
        set &Mth._numerator_clean;
        by member_id service_dt eventtype;
        prev_date = lag1(service_dt);
        if first.member_id then do;
            prev_date = .;
        end;
        diff = intck('day', prev_date, service_dt);
    run;

    data &Mth._numerator_clean3;
```

```
            set &Mth._numerator_clean2;
            if eventtype = 'HOSP' and not missing(prev_date)
                and diff <=1 then delete;
        run;


        * Count the events per member in the month;
        proc sql;
            create table &Mth._numerator_cnt as
                select member_id, payer_id,
                        "&Mth" as Month,
                        count(distinct(service_dt)) as num_events,
                        max(age_at_service) as age
                from &Mth._numerator_clean3
                group by member_id, payer_id;
        quit;

    %MEND;


    * Calculate numerator counts for each member, by month;
    %calculateNumerator (Jan,"01Jan&evalyear"d,"31Jan&evalyear"d);
    %calculateNumerator (Feb,"01Feb&evalyear"d,"&last_feb_day.Feb&evalyear"d);
    %calculateNumerator (Mar,"01Mar&evalyear"d,"31Mar&evalyear"d);
    %calculateNumerator (Apr,"01Apr&evalyear"d,"30Apr&evalyear"d);
    %calculateNumerator (May,"01May&evalyear"d,"31May&evalyear"d);
    %calculateNumerator (Jun,"01Jun&evalyear"d,"30Jun&evalyear"d);
    %calculateNumerator (Jul,"01Jul&evalyear"d,"31Jul&evalyear"d);
    %calculateNumerator (Aug,"01Aug&evalyear"d,"31Aug&evalyear"d);
    %calculateNumerator (Sep,"01Sep&evalyear"d,"30Sep&evalyear"d);
    %calculateNumerator (Oct,"01Oct&evalyear"d,"31Oct&evalyear"d);
    %calculateNumerator (Nov,"01Nov&evalyear"d,"30Nov&evalyear"d);
    %calculateNumerator (Dec,"01Dec&evalyear"d,"31Dec&evalyear"d);


* Create a numerator reference table that concatenates all of the monthly
* denominator tables into a single table;
    data ast1_out.numerator_raw_data;
        set jan_numerator_cnt feb_numerator_cnt mar_numerator_cnt
            apr_numerator_cnt may_numerator_cnt jun_numerator_cnt
            jul_numerator_cnt aug_numerator_cnt sep_numerator_cnt
```

```
            oct_numerator_cnt nov_numerator_cnt dec_numerator_cnt;
        format eval_start_dt date9.;
        format eval_end_dt date9.;
        eval_start_dt = input("01" || month || "&evalyear.", date9.);
        eval_end_dt = intnx('month',eval_start_dt,0,'end');
    run;

/********* END NUMERATOR CALCULATION *********/

/********* BEGIN COMORBIDITY CALCULATION *********/

/*  'PEDIATRIC MEDICAL COMPLEXITY ALGORITHM v3.0'

    Programmer(s):  Wren Haaland, Kathryn Whitlock
                    Center for Child Health, Behavior, and Development
                    Seattle Children's Research Institute

                    Dorothy Lyons, Peter Woodcox
                    First Steps Database
                    Research and Data Analysis Division
                    Washington State Department of Social and Health Services

    Date:           April 2017
                    Revision from v2.0

    Modified from original state by Robert Thombley in Aug 2018 for use in the
    current measure by:
    - Changing which variables are output in the final step.
    - Changing SAS library names to be consistent.
    - Modified Pulmonary/Respiratory chronic condition indicator (pulresp) to
        not be triggered by asthma diagnosis codes as defined by the inclusion
        criteria for this measure.
    - Moved configuration parameters into the configuration section of this
        measure's code.
    - Significantly shortened original author comments for the sake of brevity.
        For more information please see the original algorithm sourcecode.

   The Process:

        1) 'Claims' dataset:  claims are read in.
```

2) The program identifies whether the codes provided are from the ICD-9 or ICD-10 codeset.

3) Claims are sorted by person ID and claim ID.

4) 'Flagclaims' processing steps through the multiple lines (or single line) of individual claims for a person, identifying body systems involved, progressive status of a condition, and malignancy. The final result is a dataset where identifications of body system, progressivity, and malignancy have been rolled up to a single record per claim, with a single indication of any specified occurrence (i.e. once per claim).

5) 'Results' processing rolls up to one record per child, with
   a) a single indication whether each body type is identified as affected,
   b) a sum across claims for each body type affected (i.e. how many claims for this child have this kind of indication?),
   c) indication if a progressive condition, and
   d) indication if malignancy.

   Finally, this collected information is used to calculate the child's status.

Final Condition Definitions:

This program calculates two separate variables containing condition assignments for 'Complex Chronic', 'Non-complex Chronic', and 'Non-Chronic'.

The variable 'cond_less' contains values from the less conservative algorithm, and is designed to be used with less detailed data sources, such as those without outpatient claims.

The variable 'cond_more' contains values from the more conservative algorithm, and is designed to be used with more detailed data sources such as those including inpatient and outpatient claims.

Values for both of the above variables are
'3 Complex Chronic'

```
          '2 Non-complex Chronic'
          '1 Non-Chronic'

    Definitions of the categories assigned by the Algorithm:

        The less conservative version (cond_less) calculates values as

        'Complex Chronic':  1) more than one body system is involved, or
                            2) one or more conditions are progressive, or
                            3) one or more conditions are malignant

        'Non-complex Chronic': 1) only one body system is involved, and
                            2) the condition is not progressive or malignant

        'Non-Chronic':      1) no body system indicators are present, and
                            2) the condition is not progressive or malignant


        The more conservative version (cond_more) calculates values as

        'Complex Chronic':  1) more than one body system is involved,
                               and each must be indicated in more than one
                               claim, or
                            2) one or more conditions are progressive, or
                            3) one or more conditions are malignant

        'Non-complex Chronic': 1) only one body system is indicated in more
                                  than one claim, and
                               2) the condition is not progressive or
                                  malignant

        'Non-Chronic':      1) no body system indicators are present in
                               more than one claim, and
                            2) the condition is not progressive or malignant


Body Systems of interest and the variables used to indicate them:

        cardiac                   cardiac
        craniofacial              cranio
        dermatological            derm
```

```
            endocrinological           endo
            gastrointestinal           gastro
            genetic                    genetic
            genitourinary              genito
            hematological              hemato
            immunological              immuno
            malignancy                 malign
            mental health              mh
            metabolic                  metab
            musculoskeletal            musculo
            neurological               neuro
            pulmonary-respiratory      pulresp
            renal                      renal
            ophthalmological           opthal
            otologic                   otol
            otolaryngological          otolar

     Datasets Created:

        The FLAGCLAIMS dataset contains 1 record for each claim record,
        with the condition flags added.

        The RESULTS dataset creates 1 record per client with accumulated
        indicators, and keeps final condition determinations.

        Created December 2012, revised May 2015, revised April 2017.
 */


* Read in the claims data.  Keep client ID, and diagnosis codes.
* If incoming diagnosis data do not have decimals, the processing within this
* data step can be commented out;

data claims(keep=&sid &claimid &icdvar.1-&icdvar.&icdnum);
    set &comorbidity_claims.;
    /*array   &icdvar(&icdnum) &icdvar.1-&icdvar.&icdnum;
    do i = 1 to &icdnum;
        &icdvar{i} = compress(&icdvar{i},".");
    end;*/
    run;
```

```
/*  Determine whether codes are from ICD-9 codeset or ICD-10 codeset    */
data icdcoded;
    set claims;

    array   &icdvar(&icdnum) &icdvar.1-&icdvar.&icdnum;
    array   firstchar(&icdnum) $ firstchar1-firstchar&icdnum;
    array   icd(&icdnum) icd1-icd&icdnum;
    array   ecode(&icdnum) ecode1-ecode&icdnum;
    array   vcode(&icdnum) vcode1-vcode&icdnum;

    do i = 1 to &icdnum;
    firstchar{i} = SUBSTR(&icdvar{i}, 1, 1);
    if firstchar{i} in: ('A' 'B' 'C' 'D' 'F' 'G' 'H' 'I' 'J' 'K' 'L' 'M' 'N'
                         'O' 'P' 'Q' 'R' 'S' 'T' 'U' 'W' 'X' 'Y' 'Z')
    then icd{i}=1;
    else if firstchar{i} in: ('0' '1' '2' '3' '4' '5' '6' '7' '8' '9')
    then icd{i}=0;

    if firstchar{i} in: ('E')
    then ecode{i}=1;
    else ecode{i}=0;
    if firstchar{i} in: ('V')
    then vcode{i}=1;
    else vcode{i}=0;

    end;

    if &icdnum=1 then do;
      countwide=icd1;
      totalwide=1;
      ecodewide=ecode1;
      vcodewide=vcode1;
    end;

    else if &icdnum>1 then do;
      countwide=0;
      totalwide=0;
      ecodewide=0;
      vcodewide=0;
      do i = 1 to &icdnum;
```

```
        if icd{i}=1 then countwide=countwide+1;
        if not missing(firstchar{i}) then totalwide=totalwide+1;
        if ecode{i}=1 then ecodewide=ecodewide+1;
        if vcode{i}=1 then vcodewide=vcodewide+1;
      end;
    end;

run;

proc sort data=icdcoded; by &sid &claimid; run;
proc means data=icdcoded noprint;
  by &sid &claimid;
  var countwide totalwide ecodewide vcodewide;
  output out=counts sum=countwl totalwl ecodewl vcodewl;
run;
data counts; set counts; keep &sid &claimid totalwl countwl ecodewl vcodewl;
run;


proc sort data=claims; by &sid &claimid; run;
proc sort data=counts; by &sid &claimid; run;
data icdcoded;
    merge claims counts;
    by &sid &claimid;

    * if at least one ICD-10 code then mark entire claim;
    if countwl>=1 then icd10codeset=1;
    else if countwl=0 then do;
      * if just one code and it's an E-code;
      if totalwl=1 & ecodewl=1 then icd10codeset=1;
      * if just one code and it's a V-code;
      if totalwl=1 & vcodewl=1 then icd10codeset=0;
      * anything else, including claim of only E- and V-codes;
      else icd10codeset=0;
    end;
    else if icd10codeset=. then icd10codeset=0;
run;



/*  Sort records by client id and claim id in order to roll up
```

```
    designations by claim.   */

proc sort data=icdcoded out=claims2;
    by &sid &claimid;
run;

/*  Assign values to the appropriate flags based on conditions found and
    roll up by claim. */

data flagclaims(keep=&sid &claimid
              cardiac cranio derm endo gastro genito hemato immuno malign
              mh metab musculo neuro genetic opthal
              otol otolar pulresp renal progressive);
    set claims2;
    by &sid &claimid;

    retain cardiac cranio derm endo gastro genetic genito hemato immuno
            malign mh metab musculo neuro opthal otol otolar pulresp
            renal progressive;

* preset all the flags to 0 for rollup to one
* record with designations per claim;

    if first.&claimid then
        do;
            cardiac = 0; cranio  = 0; derm = 0; endo  = 0; gastro  = 0;
            genetic = 0; genito = 0; hemato = 0;
            immuno  = 0; malign  = 0; mh   = 0; metab = 0;
            musculo = 0; neuro   = 0; opthal = 0;
            otol    = 0; otolar  = 0; pulresp = 0; renal = 0;
            progressive = 0;
        end;

    array   &icdvar(&icdnum) &icdvar.1-&icdvar.&icdnum;

    if icd10codeset=0 then

    do i = 1 to &icdnum;

        if &icdvar{i} in: ('010' '011' '012' '013'
                           '014' '015' '016' '017' '018') then pulresp=1;
```

```
if &icdvar{i} =:'030'    then neuro=1;

if &icdvar{i} =:'0402'   then gastro=1;

if &icdvar{i} in: ('042' '043' '044') then do; immuno=1; progressive=1;
end;

if &icdvar{i} in: ('046' '0582' '0785') then do; neuro=1; progressive=1;
end;

if &icdvar(i) =:'07953' then do; immuno=1; progressive=1; end;

if &icdvar{i} in: ('090' '094' '095') then do; neuro=1; progressive=1;
end;

if &icdvar{i} =:'135'    then immuno=1;

if &icdvar{i} =:'1363'   then pulresp=1;

if &icdvar{i} =:'137'    then do; pulresp=1;   progressive=1; end;

if &icdvar{i} in: ('138' '139') then do; neuro=1; progressive=1; end;

if &icdvar{i} in: ('14' '15' '16' '17' '18' '19' '200' '201' '202'
                   '203' '204' '205' '206' '207' '208' '2090' '2091'
                   '2092' '2093' '2097' '23877')  then malign=1;

if &icdvar{i} =:'2377'   then neuro=1;

if &icdvar{i} in: ('240' '241' '242' '243' '244' '245'
              '246' '249' '250' '2510' '252'
              '2530' '2531' '2533' '2534' '2535' '2537' '2538'
              '254' '255' '256' '257' '258')  then endo=1;

if &icdvar{i}=:'2532'    then do; endo=1; progressive=1; end;

if &icdvar{i} in: ('260' '261' '262'
                   '2630' '2632'
                   '264' '265' '266' '267'
                   '2680' '2681' '2682'
```

```
                    '273' '274')  then metab=1;

if &icdvar{i} in: ('270' '271' '272' '2750') then do;
    metab=1; progressive=1;
end;

if &icdvar{i} =:'2770'   then do; pulresp=1; progressive=1; end;

if &icdvar{i} in: ('2771' '2774' '2777')   then metab=1;

if &icdvar{i} in: ('2772' '2773' '2775' '27781' '27782' '27783'
              '27784' '27785' '27786' '27787' '27788')
                    then do; metab=1; progressive=1; end;

if &icdvar{i} =:'2776'   then immuno=1;

if &icdvar{i} =:'27801'  then metab=1;

if &icdvar{i} =:'279'    then do; immuno=1; progressive=1; end;

if &icdvar{i} =:'2810'   then hemato=1;

if (&icdvar{i} =:'282' and &icdvar{i} ^=:'2825') then hemato=1;

if &icdvar{i} in: ('2824' '2826') then do; hemato=1; progressive=1;
end;

if &icdvar{i} =:'283'    then hemato=1;

if &icdvar{i} =:'284'    then do;
    hemato=1;
    if &icdvar{i} ^=:'2841' then progressive=1;
end;

if &icdvar{i} in:('2850' '28521' '28522' '28529' '2858') then hemato=1;


if &icdvar{i} =:'286' then do;
        hemato=1;
        if &icdvar{i} in: ('2860' '2861' '2862' '2863')
            then progressive=1;
```

```
    end;

if &icdvar{i} in:('2871' '2873')    then do; hemato=1;   end;

if &icdvar{i} in:('28802' '2885')    then immuno=1;
if &icdvar{i} in:('28801' '2881' '2882' '2884') then do;
    immuno=1; progressive=1;
end;

if &icdvar{i} in:('28951' '28952' '28953'
                  '28981' '28983' '28989') then do; immuno=1; end;

if &icdvar{i} in:('2911' '2921' '2940')  then  mh=1;
if &icdvar{i} in:('2950' '2951' '2952' '2953' '2954'
                   '2955' '2956' '2957' '2958') then do;
    mh=1; progressive=1;
end;

if &icdvar{i} in: (  '296' '2971' '2973'
                    '2990' '2991' '2998'
                    '3001' '3003' '3007' '30081'
                    '3010' '3011' '3012' '3013' '3014'
                    '3015' '3016' '3017' '3018'
                    '3039'
                    '3040' '3041' '3042' '3044' '3045'
                    '3046' '3047' '3048' '3049'
                    '30722' '30723' '3073' '3077') then mh=1;

if &icdvar{i} in:('3071' '30751' )  then do; mh=1; progressive=1; end;

if &icdvar{i} in: ( '3100' '3101' '311'
                    '3120' '3121' '3122' '3123' '3124' '3125' '3126'
                    '3127' '31281' '31282' '3129'
                    '31381' '3140' '3141' '3142'
                    '31501' '31502' '3151' '3152' '31531' '31532'
                    '31534' '3154' '3155' '3158' '3159'
                    '317' '3180' '3181' '3182' '319') then mh=1;

if &icdvar{i} =:'326'    then do; neuro=1; progressive=1; end;

if &icdvar{i} in:('32720' '32721' '32723' '32724' '32725'
```

```
                    '32726' '32727' '32729' ) then
                    do;
                        pulresp=1;
                        if &icdvar{i} =:'32725' then progressive=1;
                    end;

if &icdvar{i} =:'330'    then do; neuro=1; progressive=1; end;

if &icdvar{i} in:('3313' '3314') then neuro=1;

if &icdvar{i} in:('3330' '3332' '3334' '3335'
                    '3336' '33371' '33391')  then do;
    neuro=1; progressive=1;
end;

if &icdvar{i} =:'334'    then do; neuro=1; progressive=1; end;

if &icdvar{i} in:('335' '336') then do; neuro=1; progressive=1; end;

if &icdvar{i} =: '337' then neuro=1;

if &icdvar{i} in:('340' '341') then do; neuro=1; progressive=1; end;

if &icdvar{i} =:'342' then neuro=1;

if &icdvar{i} =:'343' then do;
    neuro=1;
    if &icdvar{i} in:('3432' '3438' '3439') then progressive=1;
end;

if &icdvar{i} =:'344' then do;
    neuro=1;
    if &icdvar{i} =:'3440' then progressive=1;
end;

if &icdvar{i} in:('345' '347')   then neuro=1;

if &icdvar{i} in:('3481' '3482' '3492' '3499') then
                        do;
                            neuro=1;
                            if &icdvar{i} =:'3481' then progressive=1;
```

```
                        end;

if &icdvar{i} in:('350' '351' '352' '353'
                '354' '355' '356' '357' '358') then neuro=1;


if &icdvar{i} in:('3590' '3591' '3592') then do;
    musculo=1; progressive=1;
end;
if &icdvar{i} in:('3593' '3594' '3595'
                    '3596' '3598' '3599') then musculo=1;

if &icdvar{i} in:('3602' '3603' '3604' '361'
             '3620' '3621' '36226' '36227'
             '36230' '36231' '36232' '36233'
             '36234' '36235' '36236' '36237'
             '36240' '36241' '36242' '36243'
             '36250' '36251' '36252' '36253'
             '36254' '36255' '36256' '36257'
             '36260' '36261' '36262' '36263'
             '36264' '36265' '36266'
             '36270' '36271' '36272' '36273'
             '36274' '36275' '36276' '36277'
             '36285'
             '363'
             '3641' '3642' '3645' '3647' '3648'
             '365'
             '3690' '3691' '3692' '3693' '3694'
             '36960' '36961' '36962' '36963' '36964'
             '36965' '36966' '36967' '36968' '36969'
             '36970' '36971' '36972' '36973' '36974'
             '36975' '36976'
             '377'
             '3780' '3781' '3782' '3783' '3784' '3785' '3786'
             '3787' '3788'
             '3790' '3791' '3792' '3793'
             '3794' '3795' '3796') then opthal=1;

if &icdvar{i} in:('385' '386' '387'
             '3880' '3881' '3883' '3885') then otol=1;
```

```
if &icdvar{i} in:('3891' '3892' '3897' '3898' '3899') then neuro=1;

if &icdvar{i} =:'390'    then immuno=1;

if &icdvar{i} in:('393' '394' '395' '396'
                  '397' '398' '401' '402') then do;
     cardiac=1;
     if &icdvar{i} in:('40201' '40211' '40291') then progressive=1;
end;

if &icdvar{i} =:'403' then do;
    renal=1;
    if &icdvar{i} in:('40301' '40311' '40391') then progressive=1;
end;

if &icdvar{i} =:'404' then do;
    renal=1;
    if &icdvar{i} in:('40401' '40411' '40491'
                  '40402' '40403' '40412' '40413' '40492' '40493')
    then progressive=1;
end;

if &icdvar{i} =:'405'   then cardiac=1;

if &icdvar{i} in:('410' '411' '412')  then do;
    cardiac=1; progressive=1;
end;
if &icdvar{i} =:'414' and &icdvar{i} ^=:'4144' then do;
    cardiac=1; progressive=1;
end;
if &icdvar{i} in:('416' '417')  then do; cardiac=1; progressive=1; end;

if &icdvar{i} in:('424' '426')  then cardiac=1;

if (&icdvar{i} =:'425' and &icdvar{i} ^=:'4259')  then do;
    cardiac=1; progressive=1;
end;

if &icdvar{i} in:('4270' '4271' '4273' '4274' '42781')  then cardiac=1;

if (&icdvar{i} in:('428' '4291') and &icdvar{i} ^=:'4289') then do;
```

```
    cardiac=1; progressive=1;
end;

if &icdvar{i} =:'4293'   then cardiac=1;

if &icdvar{i} in:('433' '4372' '4373' '4374'
                '4375' '4376' '4377' '438') then do;
    neuro=1; progressive=1;
end;

if &icdvar{i} in:('441') then do; cardiac=1; progressive=1; end;

if &icdvar{i} in:('442' '443')  then cardiac=1;

if &icdvar{i} =:'446' then do;
    immuno=1;

    if &icdvar{i} in:('4460' '4462' '4463') then
        do;
            progressive=1;
        end;
 end;

if &icdvar{i} in:('447') then cardiac=1;

if &icdvar{i} in:('452' '4530') then do; cardiac=1; progressive=1; end;

if &icdvar{i} in:('45350' '45351' '45352'
          '45371' '45372' '45373' '45374'
          '45375' '45376' '45377' '45379'
          '4570' '4571' '4572')  then cardiac=1;


if &icdvar{i} in:('4940' '4941')    then do;
    pulresp=1; progressive=1;
end;

/* Original:
    if &icdvar{i} =:'493'    then pulresp=1;
*/
```

```
* Modified;
if &icdvar{i} = :'493'
    and &icdvar{i} not in:(&icd9_inclusion_asthma.) then pulresp=1;
* End modification;

if &icdvar{i} =:'495'    then pulresp=1;

if &icdvar{i} =:'496'    then do;  pulresp=1; progressive=1; end;

if &icdvar{i} in:('515' '516') then do;  pulresp=1; progressive=1; end;

if &icdvar{i} in:('5190' '5193' '5194') then pulresp=1;

if &icdvar{i} =:'526'    then musculo=1;

if &icdvar{i} in: ('5270' '5271' '5277')  then gastro=1;

if &icdvar{i} in:('5300' '53013' '5303' '5305'
                  '5306' '53083' '53084' '53085') then gastro=1;

if &icdvar{i} in: ('531' '532' '533'
                  '534' '5362' '5363')  then gastro=1;

if &icdvar{i} in:('555' '556' '5651'
            '5690' '5691' '5692' '56944' '56981' )   then gastro=1;

if &icdvar{i} in:('5714' '5715' '5716' '5718' '5719')   then do;
    gastro=1; progressive=1; end;

if &icdvar{i} in:('5723' '5724' '5730')  then do;
    gastro=1; progressive=1; end;

if &icdvar{i} in:('5732' '5734' '5738' '5739'
            '57511' '5755' '5756' '5758'
            '5760' '5761' '5764' '5765' '5768'
            '5771' '5772' '5778'
            '5790' '5791' '5792' '5794')  then gastro=1;

if &icdvar{i} =:'581'    then renal=1;

if &icdvar{i} in:('582' '583' '585' '586')    then do;
```

```
    renal=1; progressive=1; end;

if &icdvar{i} =:'5880'  then do; musculo=1; progressive=1; end;

if &icdvar{i} =:'5881'                  then renal=1;

if &icdvar{i} =:'591'                   then genito=1;

if &icdvar{i} in:('59371' '59372' '59373' '59382'
             '5960'  '5961'  '5962'   '5964'
             '59652' '59653' '59654' '59655'
             '598'
             '5991' '5992' '5993' '5994' '5995'
             '59981' '59982' '59983')   then renal=1;

if &icdvar{i} in:(
             '60785'
             '6083'
             '617' '618' '619'
             '6221'
             '6230'
             '6240'
             '62920' '62921' '62922' '62923' '62929') then genito=1;

if &icdvar{i} =:'694'   then derm=1;

if &icdvar{i} =:'6954'   then do; immuno=1; progressive=1; end;

if &icdvar{i} in:('7010' '7018' '7050' '707')  then derm=1;

if &icdvar{i} =:'710' then do;
    immuno=1;
    if &icdvar{i} in:('7100' '7108' '7109') then
        do;
            progressive=1;
        end;
end;

if &icdvar{i} in:('712' '714')    then immuno=1;

if &icdvar{i} in:('717'
```

```
                  '7180' '7182' '7183' '7184' '7185' '7186' '7187'
                  '7220' '7221' '7222' '7223' '7224' '7225' '7226'
                  '7227' '7228')  then musculo=1;

if &icdvar{i} in:('720' '721' '725')  then immuno=1;

if &icdvar{i} in:('7281' '7282' '7286' '7287') then musculo=1;
if &icdvar{i} =:'7283' then do; musculo=1; progressive=1; end;

if &icdvar{i} in:('7301'
                  '7310' '7311' '7312' '7313' '7318'
                  '7320' '7321'
                  '7330' '7333' '7334' '7337' )  then musculo=1;

if &icdvar{i} in:('73605' '73606' '73607' '73631' '73632'
                  '73671' '73672' '73673' '73674' '73675'
                  '73681') then musculo=1;

if &icdvar{i} in:('7370' '7371' '7373' '7378' '7379'
             '7384' '7385' '7386' )  then musculo=1;

if &icdvar{i} in:('7400' '7401' '7402' '741')  then do;
     neuro=1; progressive=1; end;

if &icdvar{i} =:'742' then do;
    neuro=1;
    if (&icdvar{i} ^=:'7423') then progressive=1;
end;

if &icdvar{i} in:('7430' '7431' '7432' '7434' '7435'
                  '74361' '74362' '74363' '74366' '74369') then do;
    opthal=1; end;

if &icdvar{i} in:('7440' '7442' '7443' '7444' '7449')  then do;
    otolar=1;  end;

if &icdvar{i} in:('7450' '7451' '7452' '7453'
                   '7456' '7457' '7458' '7459') then do;
    cardiac=1;
    progressive=1;
end;
```

```
 if &icdvar{i} in:('7454' '7455')   then cardiac=1;

if (&icdvar{i} =:'746' and &icdvar{i} ^=:'7469') then do;
    cardiac=1;
    if &icdvar{i} in:('7462' '7467') then progressive=1;
end;

if &icdvar{i} =:'7474' then do; cardiac=1;progressive=1; end;

if &icdvar{i} in:('7471' '74721' '74722' '74729' '7473'
                  '74781' '74783' '74789') then cardiac=1;

if &icdvar{i} =:'748' then do;
    pulresp=1;
    if &icdvar{i} in:('7484' '7485' '7486') then progressive=1;
end;

if &icdvar{i} =:'749'   then cranio=1;

if &icdvar{i} in:('7501' '7502' '7503'
                  '7504' '7507' '7509') then gastro=1;

if &icdvar{i} in:('7511' '7512' '7513' '7514'
                  '7515' '75160' '7518' '7519') then do;
    gastro=1;
end;

if &icdvar{i} in:('75161' '75162' '75169' '7517' ) then do;
    gastro=1; progressive=1;
end;

if  &icdvar{i} in: ('75261' '75262' '7527') then genito=1;

if  &icdvar{i} in:('7530' '7531') then do; genito=1; progressive=1; end;
if  &icdvar{i} in:('7532' '7534' '7535' '7536'
                  '7537' '7538' '7539') then genito=1;

if  &icdvar{i} in:('7540' '7542'
                  '75430' '75431' '75432' '75433' '75434' '75435'
                  '7547'
```

```
                    '7552' '7553' '7554'
                    '75553' '75554' '75558') then musculo=1;

if &icdvar{i} in:('7560' '7561' '7563' '7564'
                     '7565' '75683' '75689' '7569') then musculo=1;
if &icdvar{i} in:('7566' '7567') then do;
    musculo=1; progressive=1;
end;

if &icdvar{i} =:'7570' then derm=1;

if  &icdvar{i} in:('7571' '75731') then do; derm=1; progressive=1;  end;

if &icdvar{i} in:('7580' '7581' '7582' '7583' '7585'
                    '7586' '7587' '7588' '7589') then do;
    genetic=1;
    if &icdvar{i} in:('7581' '7582' '75831' '75833') then progressive=1;
end;

if &icdvar{i} =:'759' then do;
    genetic=1;
    if &icdvar{i} in:('7590' '7591' '7592'
                        '7593' '7594' '7596') then progressive=1;
end;

if &icdvar{i} =:'7707' then pulresp=1;

if &icdvar{i} in:('78001' '78003')   then do;
    neuro=1; progressive=1;
end;
if &icdvar{i} in:('78051' '78053' '78057')   then neuro=1;

if &icdvar{i} in:('78833' '78834' '78837'
                '78838' '78839') then genito=1;

if &icdvar{i} in:('887' '896' '897')    then musculo=1;

if &icdvar{i} in:('9066' '9067' '9068') then musculo=1;

if &icdvar{i} =:'952'    then do;  neuro   =1; progressive=1; end;
```

```
    if &icdvar{i} =:'V08'    then immuno  =1;

    if &icdvar{i} =:'V151'   then cardiac=1;

    if &icdvar{i} =:'V420'   then do;  renal   =1; progressive=1; end;
    if &icdvar{i} =:'V421'   then do;  cardiac =1; progressive=1; end;
    if &icdvar{i} =:'V422'   then do;  cardiac =1;               end;
    if &icdvar{i} =:'V426'   then do;  pulresp =1; progressive=1; end;

    if &icdvar{i} in:('V427' 'V4284')  then do;
        gastro=1; progressive=1;
    end;

    if &icdvar{i} =:'V4281'  then do;  hemato  =1; progressive=1; end;
    if &icdvar{i} =:'V4283'  then do;  endo    =1; progressive=1; end;

    if &icdvar{i} =:'V4322'  then do;  cardiac =1; progressive=1; end;

    if &icdvar{i} in:('V520' 'V521')    then musculo=1;

    if &icdvar{i} in:('V530' 'V532')    then neuro  =1;
    if &icdvar{i} =:'V533'              then cardiac=1;
    if &icdvar{i} =:'V535'              then gastro =1;

    if &icdvar{i} =:'V550'              then pulresp=1;
    if &icdvar{i} in:('V551' 'V552' 'V553' 'V554')  then gastro =1;
    if &icdvar{i} in:('V555' 'V556' 'V557')         then genito =1;

    if &icdvar{i} in:('V560' 'V561' 'V562' 'V568')  then do;
        renal=1; progressive=1; end;

    if &icdvar{i} in:('V5781' 'V5789')  then musculo=1;


end;


else if icd10codeset=1 then

do i = 1 to &icdnum;
    if &icdvar{i} =:'A15'    then pulresp=1;
```

```
    if &icdvar{i} in: ('A171' 'A1781' 'A1783')  then neuro=1;

    if &icdvar{i} in: ('A170' 'A1782' 'A1789' 'A179')  then do;
        neuro=1; progressive=1; end;

    if &icdvar{i} =:'A180'   then musculo=1;

    if &icdvar{i} =:'A181'   then genito=1;

if &icdvar{i} in:('A182' 'A1885')   then immuno=1;

if &icdvar{i} =:'A183'   then gastro=1;

if &icdvar{i} =:'A184'   then derm=1;

if &icdvar{i} =:'A185'   then opthal=1;

if &icdvar{i} =:'A186'   then otol=1;

if &icdvar{i} in: ('A187' 'A1881' 'A1882')  then endo=1;

if &icdvar{i} in: ('A1889' 'A19')   then pulresp=1;

if &icdvar{i} =:'A30'   then neuro=1;

if &icdvar{i} in: ('A50' 'A521' 'A522' 'A523' 'A527' 'A528'
        'A529' 'A53' 'A81' 'B100' 'B900' 'B91' )
        then do; neuro=1; progressive=1; end;

if &icdvar{i} =:'B20'    then do; immuno=1; progressive=1; end;

if &icdvar{i} =:'B901'   then genito=1;

if &icdvar{i} =:'B902'   then musculo=1;

if &icdvar{i} in:('B908' 'B909')   then pulresp=1;

if &icdvar{i} =:'B9735'   then do; immuno=1; progressive=1; end;

if &icdvar{i} in: ('C0' 'C1' 'C2' 'C3' 'C40' 'C41'
```

```
    'C43' 'C4A' 'C44' 'C46' 'C47' 'C48'
    'C49' 'C5' 'C6' 'C7' 'C8' 'C90'
    'C910' 'C911' 'C913' 'C914' 'C916'
    'C919' 'C91Z' 'C92' 'C93' 'C940'
    'C942' 'C943' 'C948' 'C95' 'C96'
    'D03' 'D45' 'D474' 'D47Z1')   then malign=1;

if &icdvar{i} in: ('D510' 'D511' 'D55' 'D565' 'D568'
    'D569' 'D58' 'D591' 'D594' 'D599' 'D600'
    'D608' 'D609' 'D63' 'D640' 'D641' 'D642' 'D643'
    'D6489' 'D680' 'D681' 'D682' 'D68312'
    'D68318' 'D685' 'D6861' 'D6862' 'D688'
    'D689' 'D691' 'D693' 'D6941' 'D6949' 'D7589' ) then hemato=1;

if &icdvar{i} in: ('D560' 'D561' 'D570' 'D571' 'D572'
    'D574' 'D578' 'D610' 'D61818' 'D6182'
    'D6189' 'D619' 'D644' 'D66' 'D67'
    'D68311' 'D6942' 'D7581') then do; hemato=1; progressive=1; end;

if &icdvar{i} in: ('D704' 'D720' 'D763' 'D802' 'D803'
    'D804' 'D805' 'D808' 'D838' 'D839'
    'D841' 'D849' 'D86' 'D890' 'D891'
    'D892' 'D8989' 'D899') then immuno=1;

if &icdvar{i} in: ('D700' 'D71' 'D761' 'D800' 'D801'
    'D810' 'D811' 'D812' 'D814' 'D8189' 'D819'
    'D831' 'D89811' 'D89813') then do; immuno=1; progressive=1; end;

if &icdvar{i} in: ('D820' 'D8982')   then genetic=1;

if &icdvar(i) =:'D821' then do; genetic=1; progressive=1; end;

if &icdvar{i} in: ('E018' 'E030' 'E031' 'D032' 'E034'
    'E038' 'E039' 'E04' 'E062' 'E063' 'E064' 'E065'
    'E069' 'E070' 'E071' 'E0789' 'E079' 'E08'
    'E09' 'E10' 'E11' 'E13' 'E209' 'E21'
    'E220' 'E228' 'E229' 'E232' 'E236' 'E243'
    'E248' 'E249' 'E25' 'E260' 'E261' 'E2681'
    'E269' 'E270' 'E271' 'E274' 'E275' 'E278'
    'E279' 'E28' 'E29' 'E310' 'E318' 'E319'
    'E45')   then endo=1;
```

```
    if &icdvar(i) in: ('E00' 'E230') then do; endo=1; progressive=1; end;

    if &icdvar{i} =: 'E312'   then malign=1;

if &icdvar{i} in: ('E40' 'E43' 'E440' 'E50' 'E52' 'E53' 'E54'
        'E550' 'E643' 'E6601' 'E800' 'E8020'
        'E8029' 'E805' 'E880' 'E888')   then metab=1;

    if &icdvar{i} =: 'E45'   then neuro=1;

    if &icdvar(i) in: ('E700' 'E7021' 'E7029' 'E7040' 'E705'
        'E708' 'E710' 'E71120' 'E7119' 'E712'
        'E7131' 'E7141' 'E7142' 'E7144' 'E7150'
        'E71510' 'E71511' 'E71522' 'E71529'
        'E71548' 'E7200' 'E7203' 'E7204' 'E7209'
        'E7210' 'E7211' 'E7219' 'E7220' 'E7222'
        'E7223' 'E7229' 'E723' 'E728' 'E729' 'E7400'
        'E7401' 'E7404' 'E7409' 'E7421' 'E7439'
        'E744' 'E748' 'E749' 'E7502' 'E7519' 'E7521'
        'E7522' 'E7523' 'E75249' 'E7525' 'E7529'
        'E754' 'E7601' 'E7603' 'E761' 'E76219'
        'E7622' 'E7629' 'E763' 'E770' 'E771'
        'E786' 'E7870' 'E7871' 'E7872' 'E788' 'E789'
        'E791' 'E798' 'E83' 'E85' 'E881' 'E884') then do;
        metab=1; progressive=1; end;

    if &icdvar{i} in: ('E84') then do; pulresp=1; progressive=1; end;

    if &icdvar{i} in: ('E890' 'E894' 'E895')   then endo=1;

    if &icdvar{i} in: ('F04' 'F070' 'F1020' 'F1021' 'F1096'
        'F1120' 'F1121' 'F1320' 'F1321' 'F1420'
        'F1421' 'F1520' 'F1521' 'F1620' 'F1621'
        'F1820' 'F1821' 'F1920' 'F1921' 'F21'
        'F22' 'F24' 'F31' 'F32'
        'F33' 'F340' 'F341' 'F39' 'F429'
        'F444' 'F445' 'F446' 'F447' 'F448'
        'F449' 'F450' 'F4521' 'F4522' 'F60'
        'F63' 'F6812'
        'F840' 'F843' 'F845' 'F848' 'F849'
```

```
        'F88' 'F89' 'F90' 'F911' 'F912'
        'F913' 'F918' 'F919' 'F951' 'F952'
        'F981' 'F984' )   then mh=1;

if &icdvar{i} in: ('F200' 'F201' 'F202' 'F203' 'F205' 'F208' 'F25'
        'F50' ) then do; mh=1; progressive=1; end;

    if &icdvar{i} in: ('F7' 'F801' 'F802' 'F804' 'F81' 'F82'
        'G110' 'G255' 'G371' 'G372' 'G400'
        'G401' 'G402' 'G403' 'G404' 'G405'
        'G4080' 'G4082' 'G409' 'G40A' 'G40B'
        'G474' 'G50' 'G510' 'G511' 'G512'
        'G518' 'G519' 'G52' 'G54' 'G56' 'G57'
        'G587' 'G588' 'G589' 'G600' 'G603'
        'G608' 'G609' 'G6181' 'G6189' 'G619'
        'G620' 'G621' 'G622' 'G63' 'G7000'
        'G701' 'G702' 'G7080' 'G7089' 'G709'
        'G7114' 'G7119' 'G712' 'G722' 'G723'
        'G7289' 'G729' 'G733' 'G737' 'G801'
        'G802' 'G803' 'G804' 'G81' 'G822'
        'G825' 'G830' 'G831' 'G832' 'G833'
        'G834' 'G835' 'G8381' 'G8389' 'G839'
        'G900' 'G904' 'G905' 'G908' 'G909'
        'G910' 'G911' 'G932' 'G9389' 'G939'
        'G969' 'G990' )   then neuro=1;

    if &icdvar{i} in: ('F842' 'G09' 'G10' 'G111' 'G113'
        'G114' 'G118' 'G119' 'G12'
        'G14' 'G23' 'G241' 'G253' 'G2582'
        'G3181' 'G3182' 'G319' 'G320'
        'G3281' 'G35' 'G360' 'G370' 'G375'
        'G378' 'G379' 'G4081' 'G601' 'G710'
        'G7111' 'G7112' 'G7113' 'G800' 'G808'
        'G809' 'G901' 'G931' 'G950' 'G9519'
        'G9589' 'G959' 'G992'  ) then do; neuro=1; progressive=1; end;

    if &icdvar{i} in: ('G4730' 'G4731' 'G4733' 'G4734' 'G4736'
        'G4737' 'G4739' )   then pulresp=1;

    if &icdvar{i} in: ('G4735' ) then do; pulresp=1; progressive=1; end;
```

```
    if &icdvar{i} in: ('H150' 'H158' 'H201' 'H202'
        'H208' 'H209' 'H212' 'H2150' 'H2151'
        'H2152' 'H2154' 'H2155' 'H2156'
        'H218' 'H270' 'H2711' 'H2712' 'H2713'
        'H278' 'H300' 'H301' 'H302' 'H3081'
        'H309' 'H310' 'H3110' 'H3112' 'H312'
        'H313' 'H314' 'H318' 'H319' 'H330'
        'H3310' 'H3319' 'H332' 'H333' 'H334'
        'H338' 'H34' 'H350' 'H3515' 'H3516' 'H3517'
        'H352' 'H3530' 'H3533' 'H3534' 'H3535' 'H3536'
        'H3537' 'H3538' 'H3540' 'H3541' 'H3542'
        'H3543' 'H3545' 'H3546' 'H355' 'H357'
        'H3589' 'H36' 'H401' 'H402' 'H403'
        'H404' 'H405' 'H406' 'H408' 'H409' 'H42'
        'H430' 'H432' 'H433' 'H4381' 'H4389'
        'H4430' 'H4440' 'H4450' 'H46' 'H4701'
        'H4703' 'H4709' 'H4714' 'H472' 'H4731'
        'H4732' 'H4739' 'H474' 'H475' 'H476'
        'H479' 'H490' 'H491' 'H492' 'H493'
        'H494' )   then opthal=1;

if &icdvar{i} in: ('H4891' ) then do; metab=1; progressive=1; end;

    if &icdvar{i} in: ('H4988' 'H5000' 'H5005' 'H5006'
        'H5007' 'H5008' 'H5010' 'H5015' 'H5016'
        'H5017' 'H5018' 'H5030' 'H5032' 'H5034'
        'H5040' 'H5042' 'H5043' 'H505' 'H5060'
        'H5069' 'H5089' 'H51' 'H540' 'H541'
        'H542' 'H543' 'H548' 'H550' 'H57')   then opthal=1;


    if &icdvar{i} in: ('H71' 'H74' 'H80' 'H81' 'H83'
        'H903' 'H905' 'H906' 'H908'
        'H913' 'H918X3' 'H918X9' 'H9190'
        'H9193' 'H93093' 'H93099' 'H9313'
        'H9319' 'H9325' 'H933X3' 'H933X9')   then otol=1;

    if &icdvar{i} in: ('I00' 'I05' 'I06' 'I07'
        'I080' 'I088' 'I089' 'I09'
        'I10' 'I119' 'I340' 'I348' 'I35'
        'I370' 'I378' 'I44' 'I4510' 'I4519' 'I452'
```

```
    'I453' 'I454' 'I455' 'I456' 'I458'
    'I459' 'I471' 'I472' 'I48' 'I4901'
    'I517' 'I720' 'I721' 'I722' 'I723'
    'I724' 'I728' 'I729' 'I7300' 'I7301'
    'I7381' 'I7389' 'I739' 'I770' 'I771'
    'I773' 'I774' 'I775' 'I776' 'I778'
    'I779' 'I798' 'I822' 'I825'
    'I82709' 'I82719' 'I82729' 'I82891'
    'I82A29' 'I82B29' 'I82C29' 'I890')   then cardiac=1;

if &icdvar{i} in: ('I110' 'I200' 'I21' 'I24'
    'I2510' 'I252' 'I253' 'I254' 'I255'
    'I258' 'I259' 'I127'
    'I128' 'I421' 'I422' 'I423' 'I424'
    'I425' 'I426' 'I428' 'I43' 'I4902'
    'I50' 'I515' 'I712' 'I714' 'I716'
    'I719' 'I81' 'I820') then do; cardiac=1; progressive=1; end;


if &icdvar{i} in: ('I150' 'I158' )   then renal=1;

if &icdvar{i} in: ('I12' 'I13' ) then do; renal=1; progressive=1; end;


if &icdvar{i} in: ('I69898' 'I699')   then neuro=1;

if &icdvar{i} in: ('I630' 'I631' 'I632' 'I65' 'I671'
    'I674' 'I675' 'I676' 'I677' ) then do; neuro=1; progressive=1; end;

/* ORIGINAL:
    if &icdvar{i} in: ('J45' 'J47' 'J84842' 'J950'
    'J985' 'J986' )   then pulresp=1;
*/

* Modified;
if &icdvar{i} in: ('J45' 'J47' 'J84842' 'J950' 'J985' 'J986' )
    and &icdvar{i} not in (&icd10_inclusion_asthma.) then pulresp=1;
* End modification;

if &icdvar{i} in: ('J840' 'J8410' 'J84111' 'J84112'
    'J84117' 'J842' 'J8483' 'J84841' 'J84843'
```

```
    'J84848' 'J8489' 'J849') then do; pulresp=1; progressive=1; end;


if &icdvar{i} in: ('K110' 'K111' 'K117' 'K200' 'K220'
    'K224' 'K225' 'K227' 'K228' 'K254'
    'K255' 'K256' 'K257' 'K264' 'K265'
    'K266' 'K267' 'K274' 'K275' 'K276'
    'K277' 'K284' 'K285' 'K286' 'K287'
    'K3184' 'K50' 'K510' 'K512' 'K513'
    'K518' 'K519' 'K624' 'K6282' 'K632'
    'K763' 'K7689' 'K769' 'K77' ' K811'
    'K823' 'K824' 'K828' 'K830' 'K833'
    'K834' 'K835' 'K838' 'K861' 'K862'
    'K863' 'K868' 'K900' 'K901' 'K902'
    'K903' 'K9081' 'K915')   then gastro=1;

if &icdvar{i} in: ('K73' 'K74' 'K754' 'K761' 'K766'
    'K767' ) then do; gastro=1; progressive=1; end;



if &icdvar{i} in: ('L100' 'L101' 'L102' 'L104' 'L109'
    'L120' 'L121' 'L122' 'L128' 'L13'
    'L574' 'L744' 'L89' 'L904' 'L940'
    'L943' 'L97' 'L984' 'L988')   then derm=1;


if &icdvar{i} in: ('M050' 'M051' 'M0530' 'M0560' 'M060'
    'M062' 'M063' 'M068' 'M069' 'M08' 'M111'
    'M112' 'M118' 'M119' 'M120'
    'M303' 'M311' 'M313' 'M314' 'M316'
    'M33' 'M3500' 'M3501' 'M353' 'M45'
    'M461' 'M468' 'M469' 'M481')   then immuno=1;

if &icdvar{i} in: ('L930' 'L932' 'M300' 'M310' 'M312'
    'M321' 'M340' 'M341' 'M349' 'M355'
    'M358' 'M359') then do; immuno=1; progressive=1; end;

if &icdvar{i} in: ('M100' 'M103' 'M104' 'M109'
    'M1A0' 'M1A3' 'M1A4' 'M1A9')   then metab=1;
```

```
 if &icdvar{i} in: ('M2105' 'M2115' 'M2133' 'M2137' 'M215'
     'M216X' 'M2175' 'M2176' 'M232' 'M233'
     'M241' 'M242' 'M243' 'M244' 'M245'
     'M246' 'M247' 'M248' 'M278' 'M400'
     'M4020' 'M41' 'M420' 'M430' 'M431'
     'M438' 'M460' 'M471' 'M4781' 'M482'
     'M483' 'M489' 'M498' 'M500' 'M502'
     'M503' 'M5106' 'M513' 'M514' 'M519'
     'M61' 'M625' 'M6289' 'M720' 'M722'
     'M816' 'M818' 'M852' 'M863' 'M864'
     'M865' 'M866' 'M870' 'M88'
     'M890' 'M894' 'M897' 'M908'
     'M918' 'M955' 'M961' 'M998')   then musculo=1;

if &icdvar{i} in: ('M623' 'M906' 'N250') then do;
     musculo=1; progressive=1;
 end;


if &icdvar{i} in: ('N02' 'N04' 'N05' 'N08' 'N13' 'N1372' 'N251'
     'N2889' 'N312' 'N318' 'N319' 'N320'
     'N321' 'N322' 'N3501' 'N35028' 'N351'
     'N358' 'N359' 'N360' 'N361' 'N362'
     'N364' 'N365' 'N368' 'N37' 'N3942'
     'N3945' 'N3946' 'N39490' 'N39498')   then renal=1;

if &icdvar{i} in: ('N03' 'N18' 'N19') then do;
     renal=1; progressive=1;
end;


if &icdvar{i} in: ('N500' 'N80' 'N810' 'N811' 'N812'
     'N813' 'N814' 'N815' 'N816' 'N8181'
     'N8182' 'N8183' 'N8184' 'N8189' 'N819'
     'N820' 'N821' 'N824' 'N825' 'N828'
     'N829' 'N87' 'N880' 'N893' 'N894' 'N900' 'N901'
     'N904' 'N9081' 'N99110' 'N99111'
     'N99112' 'N99113' 'N99114' )   then genito=1;

if &icdvar{i} in: ('P270' 'P271' 'P278' )   then pulresp=1;
```

```
if &icdvar{i} in: ('P293')   then cardiac=1;

if &icdvar{i} in: ('Q030' 'Q031' 'Q038' 'Q0702')   then neuro=1;

if &icdvar{i} in: ('Q00' 'Q01' 'Q02' 'Q041' 'Q042'
     'Q043' 'Q045' 'Q048' 'Q05' 'Q060'
     'Q061' 'Q062' 'Q063' 'Q064' 'Q068'
     'Q0701' 'Q0703' 'Q078' 'Q079') then do; neuro=1; progressive=1; end;

if &icdvar{i} in: ('Q100' 'Q103' 'Q107' 'Q110' 'Q111'
     'Q112' 'Q130' 'Q131' 'Q132' 'Q133'
     'Q134' 'Q135' 'Q1381' 'Q1389' 'Q140'
     'Q141' 'Q142' 'Q143' 'Q148' 'Q150')   then opthal=1;


if &icdvar{i} in: ('Q16' 'Q171' 'Q172' 'Q174' 'Q178'
     'Q179' 'Q180' 'Q181' 'Q182' 'Q189')   then otol=1;


if &icdvar{i} in: ('Q210' 'Q211' 'Q220' 'Q221' 'Q222'
     'Q223' 'Q229' 'Q230' 'Q231' 'Q232'
     'Q233' 'Q238' 'Q242' 'Q243' 'Q244'
     'Q245' 'Q246' 'Q248' 'Q251' 'Q252'
     'Q253' 'Q254' 'Q255' 'Q256' 'Q257'
     'Q269' 'Q282' 'Q283' 'Q288')   then cardiac=1;

if &icdvar{i} in: ('Q200' 'Q201' 'Q202' 'Q203' 'Q204' 'Q205'
     'Q208' 'Q212' 'Q213' 'Q218' 'Q219'
     'Q225' 'Q234') then do; cardiac=1; progressive=1; end;


if &icdvar{i} in: ('Q300' 'Q301' 'Q308' 'Q310' 'Q311'
     'Q318' 'Q321' 'Q324' 'Q332' 'Q338'
     'Q339' 'Q34')   then pulresp=1;

if &icdvar{i} in: ('Q330' 'Q333' 'Q334' 'Q336' ) then do;
     pulresp=1; progressive=1;
 end;

if &icdvar{i} in: ('Q351' 'Q353' 'Q355' 'Q359' 'Q36'
     'Q37' 'Q385')   then cranio=1;
```

```
 if &icdvar{i} in: ('Q382' 'Q383' 'Q384' 'Q388' 'Q390'
      'Q391' 'Q392' 'Q393' 'Q394' 'Q395'
      'Q396' 'Q398' 'Q402' 'Q409' 'Q41'
      'Q42' 'Q431' 'Q433' 'Q437' 'Q438'
      'Q441' 'Q458' 'Q459')   then gastro=1;

 if &icdvar{i} in: ('Q442' 'Q443' 'Q445' 'Q446' 'Q447'
      'Q450') then do; gastro=1; progressive=1; end;

 if &icdvar{i} in: ('Q540' 'Q541' 'Q542' 'Q543' 'Q548'
      'Q549' 'Q563' 'Q564' 'Q6101' 'Q6210'
      'Q6211' 'Q6212' 'Q6231' 'Q6239' 'Q624'
      'Q625' 'Q6261' 'Q6262' 'Q6263' 'Q628'
      'Q640' 'Q6410' 'Q6419' 'Q642' 'Q644'
      'Q645' 'Q646' 'Q6471' 'Q6473' 'Q6474'
      'Q6475' 'Q6479' 'Q649')   then genito=1;

 if &icdvar{i} in: ('Q600' 'Q601' 'Q602' 'Q603' 'Q604'
      'Q605' 'Q6100' 'Q6102' 'Q6119'
      'Q612' 'Q613' 'Q614' 'Q615' 'Q618'
      'Q619') then do; genito=1; progressive=1; end;

 if &icdvar{i} in: ('Q650' 'Q651' 'Q652' 'Q653' 'Q654'
      'Q655' 'Q667' 'Q6689' 'Q670' 'Q671'
      'Q672' 'Q673' 'Q674' 'Q675' 'Q688'
      'Q710' 'Q711' 'Q712' 'Q713' 'Q714'
      'Q715' 'Q716' 'Q7189' 'Q719' 'Q720'
      'Q721' 'Q722' 'Q723' 'Q724' 'Q725'
      'Q726' 'Q727' 'Q7289' 'Q73' 'Q740'
      'Q760' 'Q761' 'Q762' 'Q763' 'Q764'
      'Q774' 'Q776' 'Q778' 'Q780'
      'Q781' 'Q782' 'Q783' 'Q784' 'Q788'
      'Q789' 'Q796' 'Q798' 'Q799')   then musculo=1;

 if &icdvar{i} in: ('Q771' 'Q790' 'Q791' 'Q792' 'Q793'
      'Q794' 'Q7959') then do; musculo=1; progressive=1; end;

if &icdvar{i} in: ('Q750' 'Q759') then cranio=1;

if &icdvar{i} in: ('Q7951') then genito=1;
```

```
if &icdvar{i} in: ('Q803' 'Q804' 'Q809' 'Q824') then do;
  derm=1; progressive=1;
end;

if &icdvar{i} in: ('Q820') then derm=1;

if &icdvar{i} in: ('Q850') then neuro=1;

if &icdvar{i} in: ('Q851' 'Q858' 'Q871' 'Q872' 'Q873'
      'Q8740' 'Q875' 'Q8789' 'Q897' 'Q898'
      'Q899' 'Q90' 'Q933' 'Q937' 'Q9381'
      'Q9389' 'Q96' 'Q970' 'Q971' 'Q972' 'Q978'
      'Q980' 'Q981' 'Q984' 'Q985' 'Q987'
      'Q988' 'Q992' 'Q998' 'Q999') then genetic=1;

if &icdvar{i} in: ('Q8781' 'Q8901' 'Q891' 'Q892' 'Q893'
      'Q894' 'Q913' 'Q917' 'Q928' 'Q934'
      'Q9388') then do; genetic=1; progressive=1; end;


if &icdvar{i} in: ('R4020' 'R403' 'S1410' 'S1411' 'S1412'
      'S1413' 'S1415' 'S2410' 'S2411' 'S2413'
      'S2415' 'S3410' 'S3411' 'S3412' 'S3413'
      'S343') then do; neuro=1; progressive=1; end;

if &icdvar{i} in: ('S4801' 'S4802' 'S4811' 'S4812'
      'S4891' 'S4892' 'S5801' 'S5802'
      'S5811' 'S5812' 'S5891' 'S5892'
      'S6841' 'S6842' 'S6871' 'S6872'
      'S7801' 'S7802' 'S7811' 'S7812'
      'S7891' 'S7892' 'S8801' 'S8802'
      'S8811' 'S8812' 'S8891' 'S8892'
      'S9801' 'S9802' 'S9831' 'S9832'
      'S9891' 'S9892') then musculo=1;

if &icdvar{i} in: ('Z21') then immuno=1;

if &icdvar{i} in: ('Z430') then pulresp=1;

if &icdvar{i} in: ('Z431' 'Z432' 'Z433' 'Z434' 'Z465') then gastro=1;
```

```
      if &icdvar{i} in: ('Z435' 'Z436' 'Z437') then genito=1;

      if &icdvar{i} in: ('Z440' 'Z441') then musculo=1;

      if &icdvar{i} in: ('Z450' 'Z953') then cardiac=1;

      if &icdvar{i} in: ('Z4531') then opthal=1;

      if &icdvar{i} in: ('Z45328' 'Z454' 'Z461' 'Z462') then neuro=1;

      if &icdvar{i} in: ('Z49' 'Z940') then do; renal=1; progressive=1; end;

      if &icdvar{i} in: ('Z941' 'Z95812') then do;
        cardiac=1; progressive=1;
      end;

      if &icdvar{i} in: ('Z942') then do; pulresp=1; progressive=1; end;

      if &icdvar{i} in: ('Z944' 'Z9481' 'Z9482') then do;
        gastro=1; progressive=1;
      end;

      if &icdvar{i} in: ('Z9483') then do; endo=1; progressive=1; end;

    end;


    if last.&claimid then output;
    run;


/*  Roll up to one record per child, with a single flag for each body type,
    a sum across claims for each body type, and presence of a progressive
    condition or malignancy. Calculate final condition determinations. */


data ast1_out.results_pmca;
    set flagclaims;
    by &sid;
    retain  anycardiac anycranio anyderm anyendo anygastro anygenetic
```

```
        anygenito anyhemato anyimmuno anymalign
        anymetab anymusculo anyneuro anyopthal anyotol anyotolar
        anypulresp anyrenal anymh anyprogressive

        anycardiac2 anycranio2 anyderm2 anyendo2 anygastro2 anygenetic2
        anygenito2 anyhemato2 anyimmuno2 anymalign2
        anymetab2 anymusculo2 anyneuro2 anyopthal2
        anyotol2 anyotolar2 anypulresp2 anyrenal2 anymh2

         cardiac2h  cranio2h  derm2h  endo2h  gastro2h  genetic2h  genito2h
         hemato2h  immuno2h  malign2h metab2h  musculo2h  neuro2h  opthal2h
         otol2h  otolar2h  pulresp2h  renal2h  mh2h ;

if first.&sid then
    do;
        anycardiac = 0; anycranio = 0; anyderm = 0; anyendo = 0;
        anygastro  = 0; anygenetic = 0;
        anygenito  = 0; anyhemato = 0; anyimmuno = 0;
        anymalign  = 0; anymetab = 0; anymusculo = 0;
        anyneuro   = 0; anyopthal = 0; anyotol = 0;
        anyotolar  = 0; anypulresp = 0; anyrenal = 0;
        anymh      = 0; anyprogressive = 0;

        anycardiac2 = 0; anycranio2 = 0; anyderm2    = 0;
        anyendo2    = 0; anygastro2  = 0; anygenetic2 = 0;
        anygenito2  = 0; anyhemato2 = 0; anyimmuno2  = 0;
        anymalign2  = 0; anymetab2   = 0; anymusculo2 = 0;
        anyneuro2   = 0; anyopthal2 = 0; anyotol2    = 0;
        anyotolar2  = 0; anypulresp2 = 0; anyrenal2   = 0;
        anymh2      = 0;

        cardiac2h = 0; cranio2h = 0; derm2h    = 0;
        endo2h    = 0; gastro2h  = 0; genetic2h = 0;
        genito2h  = 0; hemato2h = 0; immuno2h  = 0;
        malign2h  = 0; metab2h   = 0; musculo2h = 0;
        neuro2h   = 0; opthal2h = 0; otol2h    = 0;
        otolar2h  = 0; pulresp2h = 0; renal2h   = 0;
        mh2h      = 0;

    end;
```

```
    /*if a body system is indicated, create
        1) a flag indicating the presence of that body system
            involvement (indicator y/n), and
        2) the number of claims with that body system
            indicated (sum across claims).
    */
                                  *indicator y/n;       *sum across claims;
    if cardiac       = 1   then do; anycardiac       = 1; anycardiac2 + 1; end;
    if cranio        = 1   then do; anycranio        = 1; anycranio2  + 1; end;
    if derm          = 1   then do; anyderm          = 1; anyderm2    + 1; end;
    if endo          = 1   then do; anyendo          = 1; anyendo2    + 1; end;
    if gastro        = 1   then do; anygastro        = 1; anygastro2  + 1; end;
    if genetic       = 1   then do; anygenetic       = 1; anygenetic2 + 1; end;
    if genito        = 1   then do; anygenito        = 1; anygenito2  + 1; end;
    if hemato        = 1   then do; anyhemato        = 1; anyhemato2  + 1; end;
    if immuno        = 1   then do; anyimmuno        = 1; anyimmuno2  + 1; end;
    if metab         = 1   then do; anymetab         = 1; anymetab2   + 1; end;
    if musculo       = 1   then do; anymusculo       = 1; anymusculo2 + 1; end;
    if neuro         = 1   then do; anyneuro         = 1; anyneuro2   + 1; end;
    if pulresp       = 1   then do; anypulresp       = 1; anypulresp2 + 1; end;
    if renal         = 1   then do; anyrenal         = 1; anyrenal2   + 1; end;
    if opthal        = 1   then do; anyopthal        = 1; anyopthal2  + 1; end;
    if otol          = 1   then do; anyotol          = 1; anyotol2    + 1; end;
    if otolar        = 1   then do; anyotolar        = 1; anyotolar2  + 1; end;
    if mh            = 1   then do; anymh            = 1; anymh2      + 1; end;
    if progressive   = 1   then do; anyprogressive   = 1;                    end;
    if malign        = 1   then do; anymalign        = 1;                    end;

*roll up to last observation;

if last.&sid then
    do;

    length cond_less cond_more  $24.;

*************************************************************************
    CONDITION DETERMINATION--calculate condition type
                          based on two different algorithms
*************************************************************************;

    *~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
 LESS CONSERVATIVE ALGORITHM
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

        The less conservative version (cond_less) calculates values as

        'Complex Chronic':  1) more than one body system is involved, or
                            2) one or more conditions are progressive, or
                            3) one or more conditions are malignant

        'Non-complex Chronic': 1) only one body system is involved, and
                               2) the condition is not progressive
                                    or malignant

        'Non-Chronic':      1) no body system indicators are present, and
                            2) the condition is not progressive or malignant

        *count number of different body systems involved;
        scount_less = anycardiac + anycranio + anyderm
                    + anyendo  + anygastro  + anygenetic
                    + anygenito  + anyhemato + anyimmuno
                    + anymetab + anymusculo + anyneuro
                    + anypulresp + anyrenal  + anyopthal
                    + anyotol  + anyotolar  + anymh;

        *set condition based on less conservative algorithm;
        if scount_less     >= 2 or
           anyprogressive   = 1 or
           anymalign        = 1 then cond_less = '3 Complex Chronic';

        else

        if scount_less      = 1 then cond_less = '2 Non-complex Chronic';
        else cond_less = '1 Non-Chronic';

*~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 MORE CONSERVATIVE ALGORITHM
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

        The more conservative version (cond_more) calculates values as

        'Complex Chronic':  1) more than one body system is involved,
```

```
                        and each must be indicated in more than
                        one claim, or
                     2) one or more conditions are progressive, or
                     3) one or more conditions are malignant

'Non-complex Chronic': 1) only one body system is indicated in
                          more than one claim, and
                       2) the condition is not progressive or malignant

'Non-Chronic':      1) no body system indicators are present in
                       more than one claim, and
                    2) the condition is not progressive or malignant

*id body systems with indications in at least two different claims;
if anycardiac2 >= 2 then cardiac2h = 1;
if anycranio2  >= 2 then cranio2h  = 1;
if anyderm2    >= 2 then derm2h    = 1;
if anyendo2    >= 2 then endo2h    = 1;
if anygastro2  >= 2 then gastro2h  = 1;
if anygenetic2 >= 2 then genetic2h = 1;
if anygenito2  >= 2 then genito2h  = 1;
if anyhemato2  >= 2 then hemato2h  = 1;
if anyimmuno2  >= 2 then immuno2h  = 1;
if anymetab2   >= 2 then metab2h   = 1;
if anymusculo2 >= 2 then musculo2h = 1;
if anyneuro2   >= 2 then neuro2h   = 1;
if anyopthal2  >= 2 then opthal2h  = 1;
if anyotol2    >= 2 then otol2h    = 1;
if anyotolar2  >= 2 then otolar2h  = 1;
if anypulresp2 >= 2 then pulresp2h = 1;
if anyrenal2   >= 2 then renal2h   = 1;
if anymh2      >= 2 then mh2h      = 1;


* count number of body systems that are indicated in more than one claim ;
scount_more = cardiac2h + cranio2h + derm2h   + endo2h
                + gastro2h  + genetic2h + genito2h
                + hemato2h  + immuno2h + metab2h
                + musculo2h + neuro2h  + pulresp2h
                + renal2h  + otol2h    + otolar2h
                + opthal2h + mh2h;
```

```
            *set condition based on more conservative algorithm;
            if scount_more    >= 2 or
               anyprogressive  = 1 or
               anymalign       = 1 then cond_more = '3 Complex Chronic';
            else

            if scount_more     = 1 then cond_more = '2 Non-complex Chronic';

            else cond_more = '1 Non-Chronic';


            output;
        end;
    run;

/********* END COMORBIDITY CALCULATION *********/

/********* BEGIN OUTPUT FILE CREATION *********/

/* Create output file - The denominator and numerator data are brought together
                        to create a member-payer-month level
                        analytic table. This is the output of this measure.
*/
    * Add demographic information and payer details
        to the monthly denominator data;
    proc sql;
        create table denom_plus_demo as
            select a.member_id,
                   a.payer_id,
                   a.age_at_service,
                   b.gender_code,
                   b.payer_start_dt,
                   b.payer_end_dt,
                   b.medicaid_indicator,
                   b.insurance_type,
                   a.eval_start_dt, a.eval_end_dt
            from ast1_out.denominator_raw_data a
            inner join &eligibility_table. b
            on a.member_id = b.member_id and a.payer_id = b.payer_id
            and a.eval_start_dt between b.payer_start_dt and b.payer_end_dt;
```

```
quit;

* Merge denominator and numerator into one table;
proc sql;
    create table output_long as
    select a.member_id, a.payer_id,
        a.eval_start_dt, a.eval_end_dt, a.gender_code,
            a.age_at_service as age_at_last_denom_event,
            a.medicaid_indicator,
            a.insurance_type, a.payer_start_dt, a.payer_end_dt,
            coalesce(b.num_events, 0) as qual_events_this_month
    from denom_plus_demo a
    left outer join ast1_out.numerator_raw_data b
    on a.member_id = b.member_id and a.eval_end_dt = b.eval_end_dt
        and a.payer_id = b.payer_id
    ORDER BY a.member_id, a.eval_end_dt;
quit;


* Add in the comorbidity indicators to the output
* and create the final output dataset;
* This output only includes the less conservative PMCA variable,
* based on recommendations
* from the PMCA developers that this would be more reliable for this use.;

proc sql;
    create table ast1_out.asthma_1_data_&evalyear. as
    select a.*,
            coalesce(b.anycardiac,0) as anycardiac,
            coalesce(b.anycranio,0) as anycranio,
            coalesce(b.anyderm, 0) as anyderm,
            coalesce(b.anyendo,0) as anyendo,
            coalesce(b.anygastro,0) as anygastro,
            coalesce(b.anygenetic,0) as anygenetic,
            coalesce(b.anygenito,0) as anygenito,
            coalesce(b.anyhemato,0) as anyhemato,
            coalesce(b.anyimmuno,0) as anyimmuno,
            coalesce(b.anymalign,0) as anymalign,
            coalesce(b.anymetab,0) as anymetab,
            coalesce(b.anymusculo,0) as anymusculo,
            coalesce(b.anyneuro,0) as anyneuro,
```

```
            coalesce(b.anyopthal,0) as anyopthal,
            coalesce(b.anyotol,0) as anyotol,
            coalesce(b.anyotolar,0) as anyotolar,
            coalesce(b.anypulresp,0) as anypulresp,
            coalesce(b.anyrenal,0) as anyrenal,
            coalesce(b.anymh,0) as anymh,
            coalesce(b.anyprogressive,0) as anyprogressive,
            coalesce(b.scount_less,0) as scount_less,
            b.cond_less
        from output_long a
        left outer join ast1_out.results_pmca b
        on a.member_id = b.member_id;
    quit;

/********* END OUTPUT FILE CREATION *********/
```