# Clinical Quality Language (CQL) Style Guide

Version 3.0 – Last Revised: May 2019

## RECORD OF CHANGES

| Version | Date | Description of Change |
|---|---|---|
| 2.0 | August 21, 2018 | Initial publication |
| 3.0 | May 2019 | Removed references related to retired QDM logic |
| | | Updated examples of logic as needed to reflect most recent use |
| | | Added additional examples of logic in each section to provide context |
| | | Added clarifying language to content in each section |
| | | Updated language/content to align with standards changes<br>QDM 5.4 and CQL 1.3 |
| | | Removed reference to Keyword-Distinct |
| | | Updated examples using birthdate to reflect the addition of birth date, a direct reference code |

The Centers for Medicare & Medicaid Services (CMS) has contracted with Mathematica Policy Research and its partners, including The Joint Commission, to develop this style guide to support the electronic specification and maintenance of electronic clinical quality measures (eCQMs) that eligible professionals, eligible clinicians, eligible hospitals, and critical access hospitals can potentially use for reporting using certified electronic health records (EHRs) for CMS quality programs. Mathematica and its partners developed this document under two Measure and Instrument Development and Support (MIDS) indefinite delivery, indefinite quantity master contract vehicle task order contracts, Electronic Clinical Quality Measures Development and Maintenance for Eligible Professionals (CMS contract No. HHSM-500-2013-13011I, Task Order No. HHSM-500-T0001) and the Hospital Inpatient and Outpatient Process and Structural Measure Development and Maintenance contract (CMS contract No. HHSM-500-2013-13011I, Task Order No. HHSM-500-T0003).

## CONTENTS

# 1. PREREQUISITE

CQL is an HL7 standard developed as part of the Clinical Quality Framework (CQF) initiative. CQL is intended to promote standardization and harmonization; it is also intended to be clinically focused, author friendly, and human readable[1]. Users of this style guide are expected to have a baseline knowledge of CQL. Please refer to the following links for more information on CQL:

- eCQI Resource Center
- CQL Formatting and Usage Wiki

MIDS contractors, in coordination with CMS, created this document using guidelines from the CQL Formatting and Usage Wiki to promote consistency and reusability in measure specification. Measure developers and stakeholders contributed to the development of the standards included in this guide.

[1] Raw CQL files are human readable, but there is also an HTML version in the eCQM package exported from MAT. HTML human readable provides a view consistent with the style provided with earlier eCQMs based on QDM logic.

## 2. INTRODUCTION

The purpose of the style guide is to standardize expression of measure concepts across eCQMs and define a uniform "look and feel" to eCQM logic using CQL. The style guide focuses on a set of common best practices that have been implemented across CQL-based eCQMs in CMS reporting programs. The style guide also promotes the use of consistent language within the framework of CQL, including libraries, aliases, definitions, and functions, as well as guidance on other conventions, such as operator precedence. Measure stewards or developers who are developing or specifying eCQMs for future inclusion in CMS programs should align with these best practices.

## 3. STANDARDS – LIBRARIES

Libraries, the basic units of sharing CQL, consist of a foundation of CQL statements used within an eCQM. Every eCQM has at least one CQL library. eCQMs can also use CQL expressions stored in shared libraries. Using shared libraries allows similar logic to be repeated across multiple measures, thus improving standardization and harmonization across eCQMs. Shared libraries can exist at the **local** or **global** level.

- ***Local* library**: CQL libraries can be shared among several eCQMs or a specific group of eCQMs.
  - The local CQL library **'`Hospice`'** is used in a CQL definition statement below. The **'`Hospice`'** library contains multiple expressions of CQL definitions that are all hospice-related; for example, '`Discharge to Hospice`,' and '`Hospice Admissions`.'

  Example CQL Library: **`Hospice`**

```
Denominator Exclusions

  Hospice."Has Hospice"
        or exists "Malignant Neoplasm"
        or exists "Total Colectomy Performed"
```

- ***"Global" common* library**: a large shared CQL library created in the Measure Authoring Tool (MAT) that contains shared CQL expressions for use by all measure developers in the specification of an eCQM. The "Global" common library, **MATGlobalCommonFunctions**, is accessible to all MAT users and is maintained and updated by Mathematica Policy Research. Using the "Global" common library helps reduce duplication and maintains consistency across measure specifications.
- The CQL library **'`Global`'** is used in the definition below. Note: The global common library definition **`Inpatient Encounter`** is recommended for use in hospital measures.

Example CQL Library Name: **`Global`**

```
        Global.Inpatient Encounter

         ["Encounter, Performed": "Encounter
         Inpatient"]EncounterInpatient
           where"LengthInDays"(EncounterInpatient.relevantPeriod)<=
           120
                 and EncounterInpatient.relevantPeriod ends during
                 "Measurement Period"
```

## A. Best practices for using CQL libraries

A global or local library should be used when similar functions or definitions are used across measures; it is required if *five or more* measures are impacted.

## B. Standards for naming a local CQL library[2]

When naming a new local CQL library, adhere to the following standards:

- **DO** use PascalCase[3] (capitalizing the first letter of every word, without spaces between words).

  Example CQL library name: **`AdultOutpatientEncounters`**

- **DO** use names that are short, descriptive, easy to read, and accurately inform the reader of the contents of the library.

  Example CQL library name: **`Hospice`**

- **Do NOT** start the library name with a special character,[4] number, or underscore.[5]

  *Example:* *AdultEncounters

- **Do NOT** use spaces or special characters[4] within the library name.

  Exception: Use of underscore "_" is occasionally allowed, though not typical.

  - Example: Adult+PediatricEncounters

[2] A library can also have an alias. For standards on naming aliases, refer to section 5.

[3] See Appendix B for a complete list of case type definitions.

[4] Symbols such as +, -, *, and /.

[5] Library names are not quoted identifiers; CQL does not allow those characters in non-quoted identifiers.

## 4. STANDARDS – DEFINITIONS

Definitions are concise logical CQL expressions that provide detail about the meaning of measure concepts. Definitions are also referenced in the measure population logic. Definitions should be reused and referenced within other CQL expressions whenever appropriate.

### A. Best practices for CQL definitions

When naming definitions, use the following standards[6]:

- **DO** use Title Case[3] (capitalizing principal words only; not articles, conjunctions, or prepositions, and allow for spaces between words).

  Example CQL Definition name:

  - **B**aseline **D**EXA **S**can **T**wo **Y**ears **P**rior to the **S**tart of or **L**ess than **T**hree **M**onths **A**fter the **S**tart of **A**DT

- **DO** create names that are easy to read, describe the contents of the logical expression and provide context as to what makes the statement unique and clinically meaningful.

  Example CQL Definitions:

  - **`End Stage Renal Disease Encounter`**
  - **`Estimated Gestational Age Captured on the Day of Delivery`**
  - **`Initial Congestive Heart Failure Functional Assessment`**

- **DO** use only abbreviations or acronyms that are consistent with terminology used in the measure's narrative.

  Example CQL Definitions:

  - `Encounter With Prior or Present Diagnosis of Atrial Fibrillation or` **`VTE`**
  - **`ASCVD`** `Procedure Before End of Measurement Period`

- **DO** create definition names that are clear and indicate result type[7] (a yes/no should be named like a question [for example, "Is" or "Has"], a list of Encounters should be named "Encounter..", a list of Procedures should not be named "Encounter with xxx," and so on).

  Example CQL Definition names:

  - **`Has`** `Initial Major Depression Diagnosis`
  - **`Has`** `Complete Hearing Screening`
  - **`Has`** `Continuation Of Treatments`

[6] When entering the name of a definition in the Measure Authoring Tool (MAT), quotes are not required, nor do they appear in the human readable title for the definition, but they do appear in the source CQL declarations, as well as whenever they are referenced in CQL, and are required to be used for definition names that include spaces.

[7] The MAT will display the "Return Type" in the definition form (when there are no CQL errors).

- **Has** Liver Disease
- Qualifying **Encounters**
- **Procedures** Identifying Sexual Activity
- Ischemic Stroke **Encounters** with Discharge Status

- **Do NOT** name a definition the same as a value set name.[8]

  *Example:*

  **Cognitive Assessment**

  ```
  ["Intervention, Performed": "Cognitive Assessment"]
  ```

- **Do NOT** use special characters[4] in definition names.

  *Example:* `%Patients with Test Result`

<u>Additional examples of CQL Definitions (used in logic):</u>
using descriptive and clinically meaningful text, title case, and appropriate spacing.

<u>Example CQL Definition:</u>

**Newborn Fed Breast Milk Only Since Birth**

```
"Single Live Birth Encounter With Gestational Age 37 Weeks or More"
QualifyingEncounter
   with ["Substance, Administered": "Breast Milk"]
   BreastMilkFeeding
      such that BreastMilkFeeding.relevantPeriod starts during
      QualifyingEncounter.relevantPeriod
   without ["Substance, Administered": "Dietary Intake Other than
   Breast Milk"] OtherFeeding
      such that OtherFeeding.relevantPeriod starts during
      QualifyingEncounter.relevantPeriod
```

<u>Example CQL Definition:</u>

**Has Most Recent HbA1c Without Result**

```
Last(["Laboratory Test, Performed": "HbA1c Laboratory Test"]
HbA1cNoResult
   where HbA1cNoResult.relevantPeriod during "Measurement
   Period"
   sort by resultDatetime
).result is null
```

[8] In CQL, a definition name and value set name cannot share the same identifier.

Use the following table as a guide for naming definitions. The left column lists examples of definition statements. The right column provides example alternatives that offer improved description and readability.

### Table 1: Making good definition names better (more clear and concise)

| Good definition name | Better definition name |
|---|---|
| Anticoagulant Not Given at Discharge | Reason for Not Giving Anticoagulant at Discharge |
| In Demographic | Single Live Birth Encounter with Gestational Age 37 Weeks or More |
| Lab Test with Result | Most Recent Elevated HbA1c with Result |

### B. Standards for naming definitions across measures

Use the table below as a guide for naming definitions that use common concepts across measures. The left column represents the concept, and the right column represents the recommended standard naming convention.

### Table 2: Standard definition names for use across measures

| Concept | Recommended definition name |
|---|---|
| Hospice Exclusions<br>Exclusions for Hospice | Has Hospice |
| Encounters *or*<br>Eligible Encounters *or*<br>Valid Encounters | Qualifying Encounters |

## 5. STANDARDS – ALIASES OR ARGUMENT NAMES

Aliases, or argument names, are identifiers that reference individual CQL expressions or libraries. Aliases should correlate clearly to their source and can be reused to avoid restating key expressions. This allows for a more fluid, concise, and standardized CQL expression. Alias names should maintain meaning and uniformity within and across measures. Authors can develop aliases for libraries, functions, and definitions.

### A. Best practices for using CQL aliases and argument names

When naming aliases, use the following standards:

- **DO** use PascalCase[9] (capitalizing the first letter of every word, even if the name is a single word, and without spaces between words).

[9] See Appendix B for a complete list of case type definitions.

*Example CQL aliases*:

- `["Medication, Order": "High Risk Medications for the Elderly"]` **`HighRiskMedicationOrder`**
- `(["Intervention, Order": "Referral to Alternative Provider / Primary Care Provider"]` **`Referral`**

- **DO** use names that are short, descriptive, easy to read, and provide an accurate representation of the identified concept.

  Example CQL alias:

  - `["Diagnosis": "Allergy to Eggs"]` **`EggAllergy`**
  - `["Diagnosis": "Malignant Neoplasm of Colon"]` **`ColorectalCancer`**
  - `["Procedure, Performed": "General or Neuraxial Anesthesia"]` **`AnesthesiaProcedure`**
  - `["Procedure, Performed": "Influenza Vaccination"]` **`FluVaccination`**

- **DO** create alias names that are clinically focused.

  Example CQL alias:

  - `["Physical Exam, Performed": "Diastolic blood pressure"]` **`DiastolicBP`**
  - `["Medication, Order": "Beta Blocker Therapy for LVSD"]` **`BetaBlockerOrdered`**
  - `(["Encounter, Performed": "Alcohol and Drug Dependence Treatment"]`<br>`union ["Encounter, Performed": "Psych Visit - Psychotherapy"])` **`InitiatedTreatment`**

- **DO** use only abbreviations or acronyms that are consistent with terminology used in the measure's narrative sections.

  Example CQL alias:

  - `["Assessment, Performed": "VR12 Physical Component T Score"]` **`VR12PhysicalScore`**
  - `"No VTE Prophylaxis Medication Administered or Ordered"` **`NoVTEMedication`**

- **Do NOT** reuse aliases, unless a single alias carries the same content and scope as its original definition when reused within a measure.

  - See the examples to avoid below. The alias **`HeartRate`** is reused in two different definitions within the same measure, each with a different scope.

*Example*:

```
First(["Encounter, Performed": "Heart Rate Visit"] HeartRate)
      with ["Diagnosis": "Essential Hypertension"] Hypertension
      such that HeartRate.relevantPeriod overlaps
      Hypertension.prevalencePeriod
```

*And Example:*

```
Last(["Physical Exam, Performed": "Heart Rate Exam"] HeartRate
      with "Initial Blood Pressure Visit" InitialEncounter
      such that HeartRate.relevantPeriod during
      InitialEncounter.relevantPeriod
      where HeartRate.result is not null
      sort by start of relevantPeriod
   )
```

- **Do NOT** give an alias the same name as the definition name.

  *Example:*

  - Lower Back Procedure
    - [“Procedure, Performed”: “Lumbar Surgical Procedures”] **LowerBackProcedure**

- **Do NOT** use an alias if the definition statement does not require additional logic.

  *Example:*

  - Blood Transfusion

    [“Substance, Order”: “Blood Administration”]
    ~~BloodTransfusion~~

<u>Additional examples</u> of CQL aliases (used in logic):

Names that are meaningful and descriptive, and use proper case.

<u>Example CQL alias (used in logic):</u> **ComfortMeasures**

- Intervention Comfort Measures on Day of or Day After Start of Hospitalization

```
"Encounter with Age Range and without VTE Diagnosis or Obstetrical
Conditions" QualifyingEncounter
   with "Intervention Comfort Measures" ComfortMeasures
     such that Coalesce(start of ComfortMeasures.relevantPeriod,
     ComfortMeasures.authorDatetime)1 day or less on or after day
     of start of Global."Hospitalization"(QualifyingEncounter)
```

Example CQL alias (used in logic):

**`InfluenzaAdministration`** and **`InfluenzaVaccination`**

```
Influenza Immunizations or Procedures
```

```
( ["Immunization, Administered": "Influenza Vaccine"]
      union ( ["Procedure, Performed": "Influenza Vaccine
      Administered"] InfluenzaAdministration
             return "Immunization, Administered" { authorDatetime:
         start of InfluenzaAdministration.relevantPeriod }
      ) ) InfluenzaVaccination
      with ["Patient Characteristic Birthdate":"Birth date"]
      BirthDate
            such that (
             Global."CalendarAgeInDaysAt"(BirthDate.birthDatetime,
            InfluenzaVaccination.authorDatetime[10])>= 180)
             and
            (Global."CalendarAgeInDaysAt"(BirthDate.birthDatetime,
            InfluenzaVaccination.authorDatetime)<= 730)
```

Use the following table as a guide for naming aliases. The left column lists examples of alias names that measure developers should avoid. The example alternatives in the right column offer improved description and readability.

**Table 3: Making aliases better (easy to read and clinically focused)**

| Alias names to avoid | Better alias names |
|---|---|
| D *or*<br>Dx *or*<br>Diagnosis | HeartFailure<br>Pregnancy<br>Asthma<br>Bradycardia |
| Med<br>Medication | BetaBlocker<br>Antidepressant |
| P *or*<br>Proc *or*<br>Procedure | CardiacSurgery<br>Dialysis |
| Lab *or*<br>LabTest | HepBAntigenTest<br>MumpsTiter<br>PregnancyTest |
| E *or*<br>Enc | Encounter* *(use with caution if referring to several types of encounters in measure)*<br>InpatientEncounter<br>HeartFailureEncounter<br>Psychotherapy |
| ["Physical Exam, Performed": "Heart Rate"] **Exam** | HeartRate |
| ["Diagnostic Study, Performed": "Ejection Fraction"] **Study** | EjectionFraction |

[10] The “authorDatetime” actually should refer to “Immunization.dateTime”. Please see QDM Known Issues log at: https://github.com/esacinc/CQL-Formatting-and-Usage-Wiki/wiki/eCQM-Known-Issues

Use the following table as a guide for improving alias names even further. The left column lists examples of aliases. The right column includes example alternatives that offer improved description and clarity.

### Table 4: Making good alias names better (add description)

| Good alias name | Better alias name |
|---|---|
| HeartRate | FirstHeartRate |
| AntithromboticNotGiven | NoAntithrombotic |
| VisualExam | VisualFootExam |
| Fracture | LowerBodyFracture |
| THAProcedure | TotalHip |
| HeightExam | Height |

Use the table below as a guide for creating distinction between two aliases with similar characteristics within a measure (left column) and a guide for adding specificity (right column).

### Table 5: Creating distinction between aliases with similar concepts (add specificity)

| Similar concepts | Similar alias names with specificity |
|---|---|
| Heart failure encounter<br>*and*<br>Heart failure diagnosis | HeartFailureEncounter<br>*and*<br>HeartFailureDiagnosis |

## B. Standards for naming aliases across eCQMs

Use the table below as a guide for naming aliases that use common concepts across measures. The left column represents the alias concept, and the right column represents the recommended standard alias naming convention.

### Table 6: Standard alias names for use across measures

| Concept | Standardized alias |
|---|---|
| Patient date of birth | BirthDate |
| Hospice discharge | DischargeHospice |
| Hospice care order | HospiceOrder |
| Hospice intervention performed | HospicePerformed |

Reference the following examples of standard alias names.

- The alias, **`BirthDate,`** is used consistently in different CQL definition statements, and in multiple places within and across measures.
- Example CQL alias: **`BirthDate`**

```
   Initial Population

     exists ( ["Patient Characteristic Birthdate": "Birth date"]
     BirthDate
           where
      Global."CalendarAgeInYearsAt"(BirthDate.birthDatetime, start
      of "Measurement Period")>= 18
       )
```

- Example CQL alias: Hospice - related

```
   Hospice.Has Hospice

exists ( ["Encounter, Performed": "Encounter Inpatient"]
DischargeHospice
    where ( DischargeHospice.dischargeDisposition as Code[11] ~
    "Discharge to home for hospice care (procedure)"
    or DischargeHospice.dischargeDisposition as Code[11] ~ "Discharge
    to healthcare facility for hospice care (procedure)"
           )
           and DischargeHospice.relevantPeriod ends during
          "Measurement Period"
           )
    or exists ( ["Intervention, Order": "Hospice care ambulatory"]
    HospiceOrder
    where HospiceOrder.authorDatetime during "Measurement Period"
    )
    or exists ( ["Intervention, Performed": "Hospice care
    ambulatory"] HospicePerformed
    where HospicePerformed.relevantPeriod overlaps "Measurement
    Period"
    )
```

## 6. FUNCTIONS

A function is a named CQL expression that can perform any variety of calculations. Before creating new functions, measure developers should review and use, to the extent possible and applicable, the predefined functions available in the MAT or in the shared “Global” common library.

### A. Best practices and standards for naming new CQL functions

New function names should be short, descriptive, easy to read, and provide an accurate representation of the identified concept. When naming functions, use the following standards:

[11] Example of a Direct Reference Code (See section 7D for more details).

- **DO** use PascalCase.[12, 13]

  Example CQL Function name:

  - **`EmergencyDepartmentArrivalTime()`**
- **DO** use spaces after commas to separate arguments, always.

  Example CQL Function:

  - **`Global.CalendarAgeInYearsAt`**`(BirthDateTime DateTime, AsOf DateTime)`
- **Do NOT** name a function the same as a MAT pre-defined function
  - Process step: Please review the predefined operators available in the MAT to ensure the name of your function is not the same.

## B. Selecting Functions

To differentiate among similar functions, choose from the pre-defined list in the MAT or from the "Global" common library. Select the one that's most appropriate to meet the measure's intent. See the following examples of preferred functions from the "Global" common library.

- **Age Functions:**
  - Example CQL Function: **`Global.CalendarAgeInYearsAt()`**
  - Calculates age using **only birth date**, not time.
  - Use this function for measures pertaining to *adult* patients, or as applicable.
  - For example, to determine all adult patients aged 65 or older at the beginning of the `measurement period`.

  Example CQL Function (used in logic) :

```
exists ( ["Patient Characteristic Birthdate": "Birth date"]
BirthDate
       where Global.CalendarAgeInYearsAt(Birthdate.birthDatetime,
       start of "Measurement Period")>= 65
```

- **`Global.CalendarAgeInYearsAt()`** uses another Global function, **`ToDate,`** within it to ensure that the time is not used in the calculation.

- **Length of Stay Functions** (generally used for hospital measures):
  - Example CQL Function: **`Global.LengthInDays()`**

[12] See Appendix B for complete definitions of case types.

[13] Use of double-quotes in invoking is not required and should be discouraged.

- **`LengthInDays()`** calculates the difference in calendar days between the start and end of the given interval.

  ```
  Global.LengthInDays(Value Interval<DateTime>)

   difference in days between start of Value and end of
   Value
  ```

- This function can be used to calculate length of hospital stay for an inpatient encounter from admission to discharge. See example below.

o <u>Example CQL Function (used in logic):</u>

```
Global.Inpatient Encounter

   ["Encounter, Performed": "Encounter Inpatient"]
   EncounterInpatient
          where
          LengthInDays(EncounterInpatient.relevantPeriod)<=
          120
                and EncounterInpatient.relevantPeriod ends
                during "Measurement Period"
```

o <u>Example CQL Function:</u> **`Global.HospitalizationLengthOfStay()`**

- Returns the length of stay in days (i.e., the number of days between admission and discharge) for the given encounter, or from the admission of any immediately prior emergency department visit to the discharge of the encounter.

```
Global.HospitalizationLengthOfStay(Encounter "Encounter,
Performed")

LengthInDays("Hospitalization"(Encounter))
```

o <u>Example CQL Function (used in logic):</u>

```
Encounter Less Than Two Days

TJC."Ischemic Stroke Encounter" IschemicStrokeEncounter
   where
   Global.HospitalizationLengthOfStay(IschemicStrokeEncounter)<
   2
```

o <u>Example CQL Function:</u> **`Global.Hospitalization()`**

- This function returns the total interval for admission to discharge for the given encounter, or for the admission of any immediately prior emergency department visit to the discharge of the given encounter.

```
Global.Hospitalization(Encounter "Encounter, Performed")

Encounter Visit
   let EDVisit: Last(["Encounter, Performed": "Emergency
   Department Visit"] LastED
         where LastED.relevantPeriod ends 1 hour or less on or
         before start of Visit.relevantPeriod
          sort by
          end of relevantPeriod
     )
       return Interval[Coalesce(start of EDVisit.relevantPeriod,
       start of Visit.relevantPeriod),
       end of Visit.relevantPeriod]
```

- Example CQL Function (used in logic): **`Global.Hospitalization()`**

```
Admission without VTE or Obstetrical Conditions

(Global."Inpatient Encounter" InpatientEncounter
       where not ( exists ( InpatientEncounter.diagnoses
      EncounterDiagnoses
       where ( EncounterDiagnoses in "Obstetrics"
               or EncounterDiagnoses in "Venous Thromboembolism"
               or EncounterDiagnoses in "Obstetrics VTE"
          )
       )
  )
)
intersect ( Global."Inpatient Encounter" InpatientEncounter
without ( ["Diagnosis": "Obstetrics"]
       union ["Diagnosis": "Venous Thromboembolism"]
       union ["Diagnosis": "Obstetrics VTE"] ) Diagnosis
              such that Diagnosis.prevalencePeriod starts during
             Global.Hospitalization(InpatientEncounter))
```

## 7. OTHER CQL BEST PRACTICES

### A. Population criteria

When using population criteria, be descriptive and specific, making sure names are descriptive and easy to read. Below is an example of how to improve naming of population criteria.

| Current population criteria | Improved population criteria |
|---|---|
| **Initial Population**<br>'In Demographic' | **Initial Population**<br>'Single Live Birth Encounter with Gestational Age 37 Weeks or More' |

When the denominator population criteria are equivalent to the initial population criteria, state **'Initial Population'** for the Denominator as illustrated below:

Example CQL: Initial Population = Denominator

```
Initial Population

          exists ( ["Patient Characteristic Birthdate":"Birth date"]
          BirthDate
                   where
          Global."CalendarAgeInYearsAt"(BirthDate.birthDatetime,
          start of "Measurement Period")>= 65
          )
             and exists "Qualifying Encounters"

Denominator

"Initial Population"
```

## B. Additional Timing Phrases:

Additional timing phrases, besides **day of** expressions, may be needed when making a statement such as: **A starts/ends before/after or concurrent with start of B.**

Example CQL timing phrase: **ends 1 day after day of start of**

A supplementary timing constraint is added to ensure the timing of the relevant period for the AnesthesiaProcedure ends 1 day after the start of relevant period of the QualifyingEncounter.

```
No VTE Prophylaxis Medication Due to Medical Reason on Day of or Day
after Procedure

   from
       "Encounter with Age Range and without VTE Diagnosis or
      Obstetrical Conditions" QualifyingEncounter,
       ["Procedure, Performed": "General or Neuraxial Anesthesia"]
      AnesthesiaProcedure,
       "No VTE Prophylaxis Medication Administered or Ordered"
      NoVTEMedication
       where NoVTEMedication.negationRationale in "Medical Reason"
             and AnesthesiaProcedure.relevantPeriod ends 1 day after
            day of start of QualifyingEncounter.relevantPeriod
             and NoVTEMedication.authorDatetime during
            VTEICU."CalendarDayOfOrDayAfter"(end of
            AnesthesiaProcedure.relevantPeriod)
       return QualifyingEncounter
```

## C. Operator precedence

Precedence in CQL expressions is determined by the order of appearance in the expression, left to right. To ensure consistent and predictable behavior in the order of operations within CQL expressions, use parentheses around a grouping to enforce higher precedence. Refer to the table in Appendix A for more detail on operator precedence.

Refer to the example below, which uses parentheses to promote operator precedence around **exists ‘Right Mastectomy’** and **exists ‘Left Mastectomy’** to make the grouping clear.

Example CQL Operator Precedence:

```
Denominator Exclusions

Hospice."Has Hospice"
       or ( Count("Unilateral Mastectomy
      Procedure")= 2 )
       or ( exists "Right Mastectomy"
                    and exists "Left Mastectomy"
       )
       or exists "History Bilateral Mastectomy"
       or exists "Bilateral Mastectomy Procedure"
```

## D. Direct Reference Codes

Direct Reference Codes (DRCs) are single terminology codes that can be referenced by name directly within CQL logic, instead of creating a value set for a single code. DRCs are recommended for all single-use LOINC codes, and may be used for other single-use terminology codes. The DRC’s specific code will always be called out in the Terminology section of the measure, and may be referenced in the Data Criteria section, if used as part of a QDM element.

Example of Direct Reference Code (used in CQL logic):

```
Denominator Exclusions

"Initial Population" LiveBirthEncounter
       where LiveBirthEncounter.dischargeDisposition ~ "Patient
      deceased during stay (discharge status = dead) (finding)"
              and not exists "Has Complete Hearing Screening"
```

Example of Direct Reference Code (used in CQL logic):

```
Rotavirus Numerator Inclusion Conditions

( ( ["Diagnosis": "Anaphylaxis due to rotavirus vaccine (disorder)"]
       union ["Diagnosis": "Severe Combined Immunodeficiency"]
       union ["Diagnosis": "Intussusception"] ) RotavirusConditions
              with ["Patient Characteristic Birthdate":"Birth date"]
      BirthDate
              such that
             Global."CalendarAgeInDaysAt"(BirthDate.birthDatetime,
             start of RotavirusConditions.prevalencePeriod)<= 730)
```

# APPENDIX A:

# THE ORDER OF OPERATOR PRECEDENCE IN CQL FROM HIGHEST TO LOWEST

### Table A.1. The order of operator precedence in CQL from highest to lowest

| Category | Operators |
|---|---|
| Primary | . [] () |
| Conversion Phrase | convert..to |
| Unary Arithmetic | unary +/- |
| Extractor | start/end/difference/duration/width/successor/predecessor of component/singleton from |
| Exponentiation | ^ |
| Multiplicative | * / div mod |
| Additive | + - |
| Conditional | if..then..else<br>case..else..end |
| Unary List | distinct collapse flatten |
| Unary Test | is null/true/false |
| Type Operators | is as cast..as |
| Unary Logical | not exists |
| Between | between<br>*precision* between<br>duration in *precision* between<br>difference in *precision* between |
| Comparison | <= < > >= |
| Timing Phrase | same as<br>includes<br>during<br>before/after<br>within |
| Interval Operators | meets overlaps starts ends |
| Equality | = != ~ !~ |
| Membership | in contains |
| Conjunction | and |
| Disjunction | or xor |
| Implication | implies |
| Binary List | union intersect except |

Source: https://cql.hl7.org/03-developersguide.html#operator-precedence

# APPENDIX B:

# CASE TYPE DEFINITIONS

## Case Type Definitions:

*(Note: CQL is a case-sensitive language)*

- **lowercase** – All letters are lowercase
- **camelCase** – First letters of words are capitalized, except the first word, with no whitespace characters allowed
- **PascalCase** – First letters of words are capitalized, including words not capitalized in Title Case like "and" and "of," with no whitespace characters allowed
- **Title Case** – Standard title casing including spaces and tabs, but no other whitespace characters allowed