Clinical Quality Language Style Guide

Version 6.0

Last Revised: May 2022

The Centers for Medicare & Medicaid Services (CMS) contracted with Mathematica and its partners, including The Joint Commission, to develop this style guide to support the electronic specification and maintenance of electronic clinical quality measures that eligible professionals, eligible clinicians, eligible hospitals, and critical access hospitals can 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 indefinite delivery, indefinite quantity master contract vehicle task order contracts: Electronic Clinical Quality Measures Development and Maintenance for Eligible Clinicians (CMS Contract #75FCMC18D0032, Task Order #75FCMC19F0004) and Behavioral Health Measures Development and Inpatient and Outpatient Measure Maintenance (CMS Contract #75FCMC18D0032, Task Order #75FCMC19F0003).

CONTENTS

1. PREREQUISITE

Clinical Quality Language (CQL)[1] is an HL7 standard developed as part of the Clinical Quality Framework (CQF) initiative. CQL is intended to promote standardization and harmonization across the CQF standards; it is also intended to be clinically focused, author friendly, and human readable.[2]

Users of this style guide are assumed 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
- Benefits of CQL
- CQL for Implementers

The Electronic Clinical Quality Measures Development and Maintenance for Eligible Clinicians contract (EC) and Behavioral Health Measures Development and Inpatient and Outpatient Measure Maintenance contract (EH), in coordination with the Centers for Medicare & Medicaid Services (CMS), created this document using guidance from the CQL Formatting and Usage Wiki. The purpose of the guide is to promote consistency and reusability of the specifications in measures included in federal reporting programs. Measure developers and stakeholders contributed to the development of the standards in this guide.

The Measure Authoring Tool (MAT) supports the authoring of CQL and sharing of CQL libraries. Please see the MAT User Guide for additional MAT-specific information.

[1] The guidance in this document is based on CQL STU 1.55, located at: https://cql.hl7.org/01-introduction.html.

[2] Raw CQL files are human readable, but there is also an HTML version in the eCQM package exported from the MAT. HTML human readable provides a view consistent with the style provided with earlier electronic clinical quality measures based on Quality Data Model logic.

2. INTRODUCTION

The twofold purpose of this style guide is to (1) standardize expression-of-measure concepts across electronic clinical quality measures (eCQMs) and (2) define a uniform "look and feel" for eCQM logic using CQL. The guide focuses on common best practices that have been implemented across CQL-based eCQMs in CMS reporting programs. It also promotes the use of consistent language within the framework of CQL, including libraries, aliases, definitions, and functions, and provides guidance on other conventions, such as operator precedence. Measure stewards or developers who are developing or specifying eCQMs for potential inclusion in CMS reporting programs should follow these best practices. This guide applies to Health Quality Measures Format (HQMF) measures; a separate guide may be defined for Fast Healthcare Interoperable Resources (FHIR) based measures when appropriate.

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 use CQL expressions stored in shared libraries. Shared libraries enable similar logic to be used across multiple measures, improving standardization and harmonization across eCQMs. Shared libraries can exist at the ***local*** or ***global*** level in the MAT.

- ***Local* library**: CQL libraries can be shared among several eCQMs or a specific group of eCQMs within the MAT.

  The local CQL library **'Hospice'** is used in a CQL definition statement below. The **'Hospice'** library contains one CQL definition named "Has Hospice".

  Example CQL library: **Hospice**

```
Hospice. "Has Hospice"
  or exists "Malignant Neoplasm"
  or exists "Total Colectomy Performed"
  or FrailtyLTI."Advanced Illness and Frailty Exclusion Not
  Including Over Age 80"
```

- ***Global* library**: A global library is a shared CQL library created in the MAT that contains CQL expressions for all measure developers to use when specifying an eCQM. The common library, **MATGlobalCommonFunctions**, is accessible to all MAT users and is maintained and updated by Mathematica. The common library is published with each update to the Quality Data Model (QDM) and CQL standards incorporated into the MAT. Using this common library reduces duplication and maintains consistency across measure specifications. The version of the common library to be used with each measure update is specified on the eCQI Resource Center site.
- The CQL library **'Global'** is used in the definition below. Note: The common library definition "Inpatient Encounter" is recommended for use in hospital measures.

  Add CQL Library with Alias:

```
include MATGlobalCommonFunctions version '7.0.000' called Global
```

  Example CQL Library Name: **Global**

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

A. Best practices for using CQL libraries

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

B. Best practices for naming CQL libraries

The measure name and measure library name do *not* have to be the same. When naming a new measure CQL library, adhere to the following standards:

- **DO** use PascalCase.[3]

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

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

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

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

  Example*:* *AdultEncounters

- **DO NOT** use spaces or special characters[4] in the library name.

  Example: Adult+PediatricEncounter

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

[4] Special characters are symbols such as +, -, *, and /.

4. STANDARDS—DEFINITIONS

Definitions are concise logical CQL expressions that explain the meaning of measure concepts. Definitions are also referenced in the measure population logic. They should be reused and referenced in other CQL expressions, whenever appropriate.

A. Best practices for writing CQL definitions

When writing definitions, use the following best practices:

- **DO** use a 'with' or 'without' statement with a 'such that' statement when comparing two different data types or data sources.
- Example of CQL definition using 'with':

Encounter with Antibiotic Ordered within Three Days

```
"Qualifying Encounter" EDOrAmbulatoryVisit
with ["Medication, Order": "Antibiotic Medications for
Pharyngitis"]"] AntibioticOrdered
such that ( EDOrAmbulatoryVisit.relevantPeriod starts 3 days or
less on or before EDOrAmbulatoryVisit.authorDatetime )
```

- **DO** use a 'from' statement when comparing more than two sources of information.
- Example of CQL definition using 'from:

Most Recent Adult Depression Screening Positive and Follow Up Provided

```
from
"Most Recent Adult Depression Screening" LastAdultScreen,
"Follow Up for Positive Adult Depression Screening"
FollowUpPositiveAdultScreen,
"Qualifying Encounter During Measurement Period"
QualifyingEncounter
where LastAdultScreen.relevantDatetime 14 days or less on or
before day of start of QualifyingEncounter.relevantPeriod
    and LastAdultScreen.result in "Positive Depression Screening"
    and ( FollowUpPositiveAdultScreen.authorDatetime same day as
      end of QualifyingEncounter.relevantPeriod
      or FollowUpPositiveAdultScreen.relevantDatetime same day as
      end of QualifyingEncounter.relevantPeriod
    )
```

B. Best practices for naming CQL definitions

When naming definitions, use the following best practices:

- **DO** use Title Case.[5]

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

Example CQL definition name:

```
Baseline DEXA Scan Two Years Prior to the Start of or Less
than Three Months After the Start of ADT
```

- **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 definition names:

  ```
  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 definition names:

  ```
  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 the return.[6] For example, a yes/no return should be named like a question, using the words "Is" or "Has". A list of encounters should be named "Encounters…", while a list of procedures should be named "Procedures…"

  Example CQL definition names:

  ```
  Has Initial Major Depression Diagnosis
  Has Complete Hearing Screening
  Has Continuation of Treatments
  Has Liver Disease
  Qualifying Encounters
  Procedures Identifying Sexual Activity
  Ischemic Stroke Encounters with Discharge Status
  ```

- **DO NOT** give a definition the same name as a value set.[7]

  Example:

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

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

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

- **DO NOT** give a definition the same name as a CQL operator.[8] For example, 'Union' is a CQL operator used to combine all the elements from multiple lists of values. Do not use 'Union' as a name for a definition statement.
- **DO NOT** use special characters[4] in definition names.

  Example:

  ```
  % Patients with Test Result
  ```

Use Table 1 as a guide for naming definitions. The left column provides examples of definition names, and the right column shows 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

Below are more examples of CQL definitions that use descriptive and clinically meaningful text, title case, and appropriate spacing.

Example CQL definition:

Single Live Term Newborn Encounter With Newborn Fed Breast Milk Only Since Birth

```
PCNewborn."Single Live Term Newborn Encounter
During Measurement Period" QualifyingEncounter
          with ["Substance, Administered":
          "Breast Milk"] BreastMilkFeeding
              such that
          Global."NormalizeInterval" (
          BreastMilkFeeding.relevantDatetime,
          BreastMilkFeeding.relevantPeriod )
          starts during
          QualifyingEncounter.relevantPeriod
            without ["Substance, Administered":
          "Dietary Intake Other than Breast
          Milk"] OtherFeeding
              such that
          Global."NormalizeInterval" (
          OtherFeeding.relevantDatetime,
          OtherFeeding.relevantPeriod ) starts
          during
          QualifyingEncounter.relevantPeriod
```

[8] Please see the CQL specification for a full list of operators: https://cql.hl7.org/STU4/02-authorsguide.html#operations

Example CQL definition:

Most Recent HbA1c

```
Last(["Laboratory Test, Performed": "HbA1c
Laboratory Test"] RecentHbA1c
            where RecentHbA1c.relevantDatetime
           during "Measurement Period"
               sort by relevantDatetime
```

C. Standards for naming definitions across measures

Use Table 2 as a guide for naming definitions that use common concepts across measures. The left column presents the concept, and the right column shows the recommended standard naming convention.

Table 2. Standard definition names for use across measures

Concept	Recommended definition name
Hospice Exclusions Exclusions for Hospice	Has Hospice
Encounters *or* Eligible Encounters *or* Valid Encounters	Qualifying Encounters

5. STANDARDS—ALIASES OR ARGUMENT NAMES

Aliases, or argument names, are identifiers that refer to 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 their 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 and argument names, use the following standards:

- **DO** use PascalCase.[9]

  Example CQL aliases*:*

  ```
  "Most Recent Documented BMI" MostRecentBMI

       with ["Intervention, Performed": "Follow Up for Above Normal BMI"] AboveNormalFollowUp

  (["Intervention, Order": "Referral to Alternative Provider / Primary Care Provider"] Referral
  ```

- **DO** use names that are short, descriptive, and easy to read and that accurately reflect the identified concept.

  Example CQL aliases:

  ```
  ["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 aliases:

  ```
  ["Physical Exam, Performed": "Diastolic blood pressure"] DiastolicBP

  ["Medication, Order": "Beta Blocker Therapy for LVSD"] BetaBlockerOrdered

   ["Intervention, Performed": "Follow Up for Below Normal BMI"] BelowNormalFollowUp
  ```

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

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

  Example CQL aliases (note: in each of these examples, the acronym is defined in the narrative):

  ```
  ["Assessment, Performed": "Physical component summary (PCS)
  score - oblique method T-score"] VR12PhysicalAssessment

  "No VTE Prophylaxis Medication Administered or Ordered"
  NoVTEMedication
  ```

- **DO NOT** reuse aliases.

  See below for an example to avoid. The alias **HeartRate** is reused in two different definitions in 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
         overlapsHypertension.prevalencePeriod)

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

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

  Example:

  ```
  Lower Back Procedure

  ["Procedure, Performed": "Lumbar Surgical Procedures"]
  LowerBackProcedure

  where LowerBackProcedure.relevantDateTime overlaps
  "Measurement Period"
  ```

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

  Example:

  ```
  Blood Transfusion

  ["Substance, Order": "Blood Administration"] BloodTransfusion
  ```

Use Table 3 as a guide for naming aliases. The left column lists examples of alias names that measure developers should avoid. The alternatives in the right column offer improved descriptions and readability.

Table 3. Making aliases easier to read and more clinically focused

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

Use Table 4 as a guide for improving alias names even further. The left column lists examples of aliases. The right column includes alternatives that offer improved descriptions and clarity.

Table 4. Making good alias names more descriptive

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

Use Table 5 as a guide for creating distinctions between two aliases with similar characteristics within a measure by adding specificity.

Table 5. Differentiating between aliases with similar concepts by adding specificity

Similar concepts	Similar alias names with specificity
Heart failure encounter *and* Heart failure diagnosis	HeartFailureEncounter *and* HeartFailureDiagnosis

B. Standards for naming aliases across measures

Use Table 6 as a guide for naming aliases that use common concepts across measures. The left column presents the alias concept, and the right column presents the recommended standard alias naming convention.

Table 6. Standard alias names recommended for use across measures

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

Also see 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`**

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

- Accurate CQL alias names can aid in reading logic that models similar concepts. An example with the concept of hospice illustrates this below.
- Example CQL alias: DischargeToHospice, HospiceOrder, HospicePerformed

```
exists ( ["Encounter, Performed": "Encounter Inpatient"]
DischargeToHospice
    where ( DischargeToHospice.dischargeDisposition ~ "Discharge to
    home for hospice care (procedure)"
    or DischargeToHospice.dischargeDisposition  ~ "Discharge to
    healthcare facility for hospice care (procedure)"
    )
          and DischargeToHospice.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"
 )
```

Note: Use the equivalence operator (~) for code comparison and the ‘in’ operator to compare to value sets.

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—to the extent possible and applicable—use the predefined functions available in the MAT or in the shared "Global" common library. Functions act on the input arguments passed to them, whereas definitions operate only on the expressions in the definition.

A. Best practices and standards for naming new CQL functions

New function names should be short, descriptive, and easy to read and should provide an expression that accurately represents the identified concept. When naming functions, use the following standards:

- **DO** use PascalCase.[10]

  Example CQL function name:

  ```
  "EmergencyDepartmentArrivalTime"(Encounter "Encounter,
  Performed")
  ```

- **DO** use spaces after commas to separate arguments.

  Example CQL function:

  ```
  Global.NormalizeInterval(pointInTime DateTime, period
  Interval<DateTime>)
  ```

- **DO NOT** give a function the same name as a MAT predefined function.

  Process step: Please review the predefined operators available in the MAT[11] to ensure that the name of your function is not the same.

B. Selecting functions

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

- "NormalizeInterval" function:

  Example: Global CQL function:

- **Global."NormalizeInterval"**(pointInTime DateTime, period Interval<DateTime>):
- if pointInTime is not null then Interval[pointInTime, pointInTime]
- else if period is not null then period

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

[11] See the MAT user guide for a list of operators: https://www.emeasuretool.cms.gov/training-resources/user-guide.

- else null as Interval<DateTime>
    - Use the “NormalizeInterval” function for QDM datatypes that have use cases for both a relevantDatetime and a relevantPeriod to reduce implementation burden associated with variable use of timing attributes across measures.
    - Use the “NormalizeInterval” function for the following QDM datatypes:
        - Assessment, Performed
        - Device, Order
        - Diagnostic Study, Performed
        - Diagnostic Study, Order
        - Intervention, Performed
        - Intervention, Order
        - Laboratory Test, Performed
        - Laboratory Test, Order
        - Medication, Active
        - Medication, Administered
        - Medication, Dispensed
        - Physical Exam, Performed
        - Procedure, Performed
        - Substance, Administered
        - Encounter, Performed
        - Condition/Diagnosis/Problem
        - Allergy/Intolerance
        - Symptom
        - Immunization, Administered
    - The “NormalizeInterval” function may be needed when using a sort clause. An example is provided below:

```
"First BMI in Measurement Period":
 { First(["Physical Exam, Performed": "BMI Ratio"] BMI
   where start of
Global."NormalizeInterval"(BMI.relevantDatetime,
BMI.relevantPeriod)during "Measurement Period"
       and BMI.result is not null
     sort by start of
Global."NormalizeInterval"(relevantDatetime, relevantPeriod)
```

-)}
- Age functions:

Example CQL function:

- **"AgeInYearsAt"(**date from start of**)**
    - This function calculates age using **only birth date**, not time.

Example CQL function (used in logic):

```
AgeInYearsAt(date from start of "measurement Period") in Interval
[18,85]
    And exists "Essential Hypertension Diagnosis"
    And exists AdultOutpatientEncounters."Qualifying Encounters"
```

- **Length-of-stay functions** (generally used for hospital measures):

  Example CQL function: **Global."LengthInDays"()**

    - **LengthInDays()** calculates the difference in calendar days between the start and end of the given interval. Timing intervals should always be noted in chronological order as [start, finish] to avoid negative time intervals.

      ```
      Global."LengthInDays"(Value Interval<DateTime>)
      difference in days between start of Value and end of Value
      ```

    - This function can be used to calculate the length of a hospital stay for an inpatient encounter from admission to discharge. Also see the examples below.

  Example CQL function (used in logic):

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

  Example CQL function: **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
      ))
      ```

  Example CQL function (used in logic):

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

Example CQL function: **`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):

```
 "Ischemic Stroke Encounter" IschemicStrokeEncounter
     with "Intervention Comfort Measures" ComfortMeasure
          such that Coalesce(start of
 ComfortMeasure.relevantPeriod,
 ComfortMeasure.authorDatetime)during
 Global."Hospitalization"(IschemicStrokeEncounter)
```

7. OTHER CQL BEST PRACTICES

A. Population criteria

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

Current population criteria	Improved population criteria
Initial Population 'In Demographic'	**Initial Population** '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 shown below:

Example CQL: Initial Population = Denominator

```
Initial Population
      "Patient is Male"
         and "Has Qualifying Encounter"
         and "Androgen Deprivation Therapy Start Date" is not
      null
Denominator
"Initial Population"
```

- **DO** use brackets, **[** and **]**, to represent a closed interval and parenthesis, **(** and **)**, for open intervals. This pertains to definitions containing age intervals to avoid inconsistencies and to harmonize logic across measures. In the example below, the definition uses Interval[18, 85] to include patients ages 18 to 85 but exclude patients ages 17 and younger and 85 and older at the start of the measurement period.

  Example in CQL of Age Interval:

```
   AgeInYearsAt(date from start of "measurement Period") in Interval
[18,85]
      And exists "Essential Hypertension Diagnosis"
            And exists AdultOutpatientEncounters."Qualifying
        Encounters"
```

B. Additional timing phrases

CQL supports precision-based date/time comparisons. Be sure to consider whether day or time should be considered in 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.**

When comparing date-time valued elements to the measurement period, unless time-sensitive comparison is truly desired, use the day of modifier to indicate that the comparison should be performed to the day.

In addition, when accessing date-time valued elements for comparison to the measurement period, unless time-sensitive comparison is truly desired, use the date from extractor to access only the date portion of the date-time valued element.

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.

```
from
 VTE."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 Global."NormalizeInterval" (
AnesthesiaProcedure.relevantDatetime,
AnesthesiaProcedure.relevantPeriod ) ends 1 day after day of
start of QualifyingEncounter.relevantPeriod
and NoVTEMedication.authorDatetime during
TJC."CalendarDayOfOrDayAfter" (
 end of Global."NormalizeInterval" (
AnesthesiaProcedure.relevantDatetime,
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. See the table in Appendix A for more details.

In the example below, parentheses are used to promote operator precedence around **exists "Left Mastectomy Diagnosis"** and **exists "Left Mastectomy Procedure"** and to make the groupings clear.

Example CQL operator precedence:

```
Hospice."Has Hospice"
  or ( ( exists ( "Right Mastectomy Diagnosis" )
        or exists ( "Right Mastectomy Procedure" )
    )
      and ( exists ( "Left Mastectomy Diagnosis" )
          or exists ( "Left Mastectomy Procedure" )
      )
  )
  or exists "Bilateral Mastectomy Diagnosis"
  or exists "Bilateral Mastectomy Procedure"
```

```
  or FrailtyLTI."Advanced Illness and Frailty Exclusion Not
Including Over Age 80"
```

D. Direct Reference Codes

Measure developers maintain and publish value sets on the Value Set Authority Center web site. Value sets and codes are listed in the terminology section of each measure specification. Direct reference codes (DRCs) are single terminology codes that can be referenced directly within CQL logic, instead of creating a single code value set. DRCs are recommended for all single-use LOINC codes and may be used for other single-use terminology codes. When included in a definition, a DRC is incorporated in the CQL syntax through use of the code descriptor. The DRC's specific code and corresponding descriptor will always be included in the Terminology section of the human readable; it may also be referenced in the Data Criteria section if used as part of a QDM element, not just as an attribute of a previously defined QDM element.

- **DO NOT** include the version of the DRC directly in the CQL. This causes downstream issues in MAT and other eCQM tools like the Bonnie testing tool.

  Example of DRC without a version number:

  - `code "Birth date": '21112-8' from "LOINC" display 'Birth date'`

Examples of DRCs used in CQL logic:

```
( ( ["Diagnosis": "Anaphylaxis due to rotavirus vaccine
(disorder)"]
    union ["Diagnosis": "Severe Combined Immunodeficiency"]
    union ["Diagnosis": "Intussusception"] ) RotavirusConditions
    where ( start of RotavirusConditions.prevalencePeriod during
Interval[Patient.birthDatetime, Patient.birthDatetime + 2 years])
)
)
```

VERSION HISTORY

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 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 and content to align with standards changes 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
4.0	May 2020	Updated language and content to align with standards changes QDM 5.5 and CQL 1.4
		Updated examples of logic to reflect most recent use
5.0	May 2021	Removed references related to retired QDM logic
		Updated examples of logic to reflect most recent use
		Added section on best practices for writing definitions
		Added guidance on creating age intervals
		Added 'Normalize Interval' function to the 'Selecting functions' section
6.0	May 2022	Updated examples of logic to reflect most recent use

APPENDIX A:

THE ORDER OF OPERATOR PRECEDENCE IN CQL

Table A.1. Order of operator precedence in CQL (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 case..else..end
Unary List	distinct collapse flatten expand
Unary Test	is null true false
Type Operators	is as cast..as
Unary Logical	not exists
Between	between *precision* between duration in *precision* between difference in *precision* between
Comparison	<= < > >=
Timing Phrase	same as includes during before/after 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 for the first word, with no whitespace characters allowed (used for QDM attributes)
- **PascalCase** – First letters of words are capitalized, including words not capitalized in Title Case such as “and” and “of,” with no whitespace characters allowed
- **Title Case** – Standard title casing including spaces and tabs, but no other whitespace characters allowed