
Centers for Medicare & Medicaid Services

# Quality Data Model (QDM)-based Clinical Quality Language (CQL) Style Guide

**Version 9.0**

**May 2025**

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 (eCQMs) that eligible clinicians and hospitals can use for reporting using certified electronic health records (EHRs) for CMS quality programs. Mathematica and its partners developed this document under a Measure and Instrument Development and Support indefinite delivery, indefinite quantity master contract vehicle task order: Behavioral Health Measures Development & Inpatient and Outpatient Measures Maintenance (CMS Contract 75FCMC18D0032, Task Order 75FCMC24F0136).

## CONTENTS

# 1. BACKGROUND

Clinical Quality Language (CQL)[1] is an HL7 standard developed as part of the Clinical Quality Framework (CQF) initiative. CQL is intended to be a clinically focused, author-friendly, and human-readable language that promotes consistency and harmonization across the CQF standards.[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:

- Electronic Clinical Quality Improvement (eCQI) Resource Center
- CQL Formatting and Usage Wiki

This guide was created under the Behavioral Health Measures Development & Inpatient and Outpatient Measures Maintenance contract in coordination with the Centers for Medicare & Medicaid Services (CMS). It is based on 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 interested parties contributed to the development of the standards in this guide.

The Measure Authoring Development Integrated Environment (MADiE) tool supports the authoring of CQL and sharing of CQL libraries. Please see the MADiE User Guide for additional MADiE-specific information.

[1] The guidance in this document is based on CQL version 1.5.3, located at https://cql.hl7.org/01-introduction.html.

[2] Raw CQL files are human readable. There is also an HTML version in the electronic clinical quality measure package exported from MADiE. The HTML human readable file provides a view consistent with the style of earlier electronic clinical quality measures based on Quality Data Model logic.

## 2. INTRODUCTION

The twofold purpose of this style guide is to (1) standardize the expression of measure concepts across Quality Data Model (QDM) electronic clinical quality measures (eCQMs) and (2) define a uniform look and feel for QDM eCQM logic using CQL. The guide focuses on common best practices 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 it 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 QDM-based Health Quality Measures Format measures. A separate guide will be defined for Fast Healthcare Interoperability 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, which is called the primary 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.

- **Local shared library.** All MADiE users can view CQL libraries, but they are typically shared and editable only among a specific group of users within MADiE. A local shared library should be used when similar functions or definitions are used across measures; it is required if five or more measures are affected.

The local shared CQL library, **`HospiceQDM`**, is used in the following CQL definition statement. The **`HospiceQDM`** library contains a CQL definition named, "Has Hospice Services." This example shows how shared libraries, when used, are represented in measure logic.

Example of CQL using a local shared library:[3] **`HospiceQDM`**

```
Hospice."Has Hospice Services"
  or exists "Malignant Neoplasm"
  or exists "Total Colectomy Performed"
  or AIFrailLTCF."Is Age 66 or Older with Advanced Illness and Frailty"
  or AIFrailLTCF."Is Age 66 or Older Living Long Term in a Nursing Home"
  or PalliativeCare."Has Palliative Care in the Measurement Period"
```

- **Global shared library.** A global library is a shared CQL library that contains CQL expressions for all measure developers to use when specifying an eCQM. All MADiE users can access the global shared library, CQMCommonQDM.[4] This library is updated and published as needed to align with the QDM and CQL standards incorporated into MADiE. Using this global shared library reduces duplication and maintains consistency of measure logic across measure specifications. Users may submit questions about the CQMCommonQDM library to the Assistant Secretary for Technology Policy/Office of the National Coordinator for Health Information Technology (ASTP/ONC) CQL Issue Tracker.

Example of CQL in the CQMCommonQDM library:[5]

```
Global.Inpatient Encounter
    ["Encounter, Performed": "Encounter Inpatient"] EncounterInpatient
      where EncounterInpatient.relevantPeriod ends during day of "Measurement Period"
```

### A. Best practices for naming measures' primary CQL libraries

As described earlier, each eCQM has at least one CQL library called the primary library. The maximum allowable length for a measure's CQL library name is 64 characters, but the

[3] Example from CMS130v13 for 2025 reporting.

[4] See the global shared library, CQMCommonQDM v9.0, at https://ecqi.healthit.gov/sites/default/files/2026-CQMCommonQDM-Library.zip.

[5] Example from CMS506v7 for 2025 reporting.

recommended maximum length is 30 characters. The library name in the 'Measure CQL Library Name' field in MADiE for each QDM eCQM should adhere to the following standards:

- eCQM Abbreviated Title + Shortened Measure Name

  Example: CMS506SafeUseofOpioids

In MADiE, the 'CMS ID' field contains the numerical identifier for the measure (that is, 506), while the 'eCQM Abbreviated Title' field contains the prefix of "CMS" along with the measure's numerical identifier (that is, CMS506). The 'Measure Name' field contains the full title of the eCQM (that is, Safe Use of Opioids - Concurrent Prescribing).

## B. Best practices for naming shared CQL libraries

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

- **Do** capitalize the first letter of each compound word in a variable (PascalCase).[6]

  Example: **`AdultOutpatientEncountersQDM`**

- **Do** use names that are short, descriptive, and easy to read and that accurately reflect the contents of the shared library. There is a 64-character limit for shared library names.

  Example: **`HospiceQDM`**

- **Do** append "QDM" to QDM shared library names.

  Example: **`AdultOutpatientEncountersQDM`**

- **Do not** start the library name with a special character,[7] number, or underscore. The following is an example to avoid:

  Example: `*AdultEncounters`

- **Do not** use spaces or special characters in the library name. The following is an example to avoid:

  Example: `Adult+PediatricEncounter`

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

[7] Special characters are symbols such as a plus sign (+), an underscore (_), a minus sign (-), an asterisk (*), and a slash mark (/). This list is non-exhaustive.

## 4. STANDARDS: DEFINITIONS

Definitions are concise logical CQL expressions that describe the meaning of measure concepts. Measure population logic also references definitions. Measure developers should reuse and reference definitions in other CQL expressions, whenever appropriate.

### A. Best practices for writing CQL definitions

When writing definitions, use the following best practices:[8]

- **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':[9]

**Encounter With Antibiotic Ordered Within Three Days**

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

- **Do** use a 'from' statement when comparing more than two sources of information.

  Example of CQL definition using 'from':[10]

**Most Recent Adult Depression Screening Positive and Follow Up Provided**

```
from
  "Most Recent Adult Depression Screening" LastAdultScreen,
  "Follow Up Intervention for Positive Adult Depression Screening" FollowUpPositiveAdultScreen,
  "Qualifying Encounter During Measurement Period" QualifyingEncounter
  where Global."NormalizeInterval" ( LastAdultScreen.relevantDatetime, LastAdultScreen.relevantPeriod ) 14 days or less on or before day of start of
QualifyingEncounter.relevantPeriod
    and LastAdultScreen.result ~ "Depression screening positive (finding)"
    and ( start of Global."NormalizeInterval" ( FollowUpPositiveAdultScreen.relevantDatetime, FollowUpPositiveAdultScreen.relevantPeriod ) during
QualifyingEncounter.relevantPeriod
      or FollowUpPositiveAdultScreen.authorDatetime 2 days or less on or after day of end of QualifyingEncounter.relevantPeriod
    )
    and Coalesce(start of Global."NormalizeInterval"(FollowUpPositiveAdultScreen.relevantDatetime, FollowUpPositiveAdultScreen.relevantPeriod),
FollowUpPositiveAdultScreen.authorDatetime) during "Measurement Period"
```

[8] For further information on best practices for writing CQL definitions, see the sections on Queries (http://cql.hl7.org/02-authorsguide.html#queries) and Full Query (http://cql.hl7.org/02-authorsguide.html#full-query) of the CQL Specification Author's Guide.

[9] Example from CMS146v13 for 2025 reporting.

[10] Example from CMS2v14 for 2025 reporting.

## B. Best practices for naming CQL definitions

When naming definitions, use the following best practices:

- **Do** capitalize the first letter of every word (initial case) [11] or capitalize the first and last word and all major words in between (title case).

  Examples of CQL definition names using initial case:[12]

  - **Low Glucose Test Followed By Glucose Test Result Greater Than 80**
  - **Encounter With Antibiotic Ordered Within Three Days**

  Example of CQL definition name using title case:[13]

  - **Has 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. Please note that long definition names can impair readability of the logic.

  Examples of CQL definition names:[14]

  - **First ADHD Medication Prescribed During Intake Period**
  - **Delivery Encounters with Calculated Gestational Age Greater than or Equal to 20 Weeks**
  - **HeartFailure.Heart Failure Outpatient Encounter with History of Moderate or Severe LVSD**

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

  Examples of CQL definition names:[15]

  - **Encounter with Prior or Present Diagnosis of Atrial Fibrillation or Prior Diagnosis of VTE**
  - **Is Currently Taking Beta Blocker Therapy for LVSD**

- **Do** create definition names that are clear and indicate the return.[16] For example, a yes or no return should be named like a question, using the words "is" or "has."

  Examples of CQL definition names:[17]

  - **Has Most Recent Adult Screening Negative**
  - **Has Total Hip Arthroplasty with 1 or More Lower Body Fractures**
  - **Has Normal BMI**

[11] See Appendix B for a complete list of case type definitions and for more information on the recommended use of initial case. The adoption of initial case is still in early stages. Other definition name examples that use initial case will be added to the guide in a future version, following wider adoption of this best practice.

[12] Examples from CMS816v4 and CMS146v13 for 2025 reporting.

[13] Example from CMS645v8 for 2025 reporting.

[14] Examples from CMS136v14, CMS1028v3, and CMS135v13 for 2025 reporting.

[15] Examples from CMS108v13 and CMS144v13 for 2025 reporting.

[16] MADiE will display the return type in the definition form (when there are no CQL errors).

[17] Examples from CMS2v14, CMS56v13, and CMS69v13 for 2025 reporting.

- **Do not** give a definition the same name as a value set,[18] as this can reduce the clarity and readability of the logic and raise the risk of logical errors. The following is an example to avoid.

  Example:

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

- **Do not** give a definition the same name as a CQL operator.[19] For example, "union" is a CQL operator used to combine all elements from multiple lists of values. Do not use "union" as a name for a definition statement.
- **Do not** use special characters[20] in definition names. The following is an example to avoid.

  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 are clearer and more readable.

**Table 1. Making good definition names better and clearer**

| 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 |

The following are more examples of CQL definitions that are descriptive and clinically meaningful.

Example of CQL definition:[21]

Delivery Encounter with Cesarean Birth

```
"Singleton Delivery Encounters at 37 Plus Weeks Gravida 1 Parity 0, No Previous Births" ThirtysevenWeeksPlusEncounter
  with ["Procedure, Performed": "Cesarean Birth"] CSection
    such that Global."NormalizeInterval" ( CSection.relevantDatetime, CSection.relevantPeriod ) during PCMaternal."HospitalizationWithEDOBTriageObservation" (
ThirtysevenWeeksPlusEncounter )
```

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

[19] See the CQL Specification Author's Guide for a full list of operators: https://cql.hl7.org/STU4/02-authorsguide.html#operations.

[20] Special characters are symbols such as a plus sign (+), an underscore (_), a minus sign (-), an asterisk (*), and a slash mark (/). This list is non-exhaustive.

[21] Example from CMS334v6 for 2025 reporting.

Example of CQL definition:[22]

Most Recent Glycemic Status Date

```
Last(("Glycemic Status Assessment" QualifyingGlycemicStatus
   return date from start of Global."NormalizeInterval"(QualifyingGlycemicStatus.relevantDatetime, QualifyingGlycemicStatus.relevantPeriod))
QualifyingGlycemicStatusDays
   sort asc
)
```

## 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 *or*<br>Exclusions for Hospice | Has Hospice Services |
| Encounters *or*<br>Eligible Encounters *or*<br>Valid Encounters | Qualifying Encounters (Note: generally used for clinician measures) |

[22] Example from CMS122v13 for 2025 reporting.

## 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. Alias names should maintain their meaning and uniformity within and across measures. This allows for a more fluid, concise, and standardized CQL expression. Measure developers can create 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.[23]

  Examples of CQL aliases:[24]

```
High BMI Interventions Performed

( ["Intervention, Performed": "Follow Up for Above Normal BMI"] HighInterventionsPerformed
   where HighInterventionsPerformed.reason in "Overweight or Obese"
     or ( exists ["Diagnosis": "Overweight or Obese"] OverweightObese
        where OverweightObese.prevalencePeriod starts before or on day of Global."NormalizeInterval" ( HighInterventionsPerformed.relevantDatetime,
HighInterventionsPerformed.relevantPeriod )
          and not ( OverweightObese.prevalencePeriod ends before day of Global."NormalizeInterval" ( HighInterventionsPerformed.relevantDatetime,
HighInterventionsPerformed.relevantPeriod ) )
     )
)
```

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

  Examples of CQL aliases:[25]

  1. ["Assessment, Not Performed": "Adolescent depression screening assessment"] NoAdolescentScreen
  2. ["Intervention, Performed": "Tobacco Use Cessation Counseling"] TobaccoCessationCounseling
  3. ["Diagnosis": "Prostate Cancer"] ProstateCancer
  4. with ["Diagnosis": "Atrial Fibrillation or Flutter"] AtrialFibrillationFlutter

- **Do** use names that clearly distinguish two different events that have the same logic but occur at different times.

  For example, consider logic that is looking for a lab result that occurred before and after a clinical action.

| Good alias name | Better alias name |
|---|---|
| Lab1 | LabWithResultBeforeProcedure |
| Lab2 | LabWithResultAfterProcedure |

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

[24] Examples from CMS69v13 for 2025 reporting.

[25] Examples from CMS2v14, CMS138v13, CMS129v14, and CMS71v14 for 2025 reporting.

Examples of CQL aliases:[26]

```
More Than One Order(Medication List<"Medication, Order">)

"Medication" OrderMedication1
 with "Medication" OrderMedication2
  such that ( OrderMedication1.authorDatetime during "Measurement Period"
     and OrderMedication1.refills >= 1
  )
    or ( date from OrderMedication1.authorDatetime !~ date from OrderMedication2.authorDatetime
       and OrderMedication1.authorDatetime during "Measurement Period"
       and OrderMedication2.authorDatetime during "Measurement Period"
    )
    or ( date from OrderMedication1.authorDatetime ~ date from OrderMedication2.authorDatetime
       and OrderMedication1.authorDatetime during "Measurement Period"
       and date from start of OrderMedication1.relevantPeriod !~ date from start of OrderMedication2.relevantPeriod
       and start of OrderMedication1.relevantPeriod during "Measurement Period"
       and start of OrderMedication2.relevantPeriod during "Measurement Period"
    )
 return OrderMedication1
```

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

  Examples of CQL aliases:[27]

  1. EncounterLastDiastolicBP: Last(["Physical Exam, Performed": "Diastolic blood pressure"] DiastolicBP
  2. exists ["Medication, Order": "Beta Blocker Therapy for LVSD"] BetaBlockerOrdered
  3. "Documented Low BMI during Measurement Period" LowBMI

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

  Examples of CQL aliases (Note: In each example, the acronym is defined in the narrative.):[28]

  1. ["Assessment, Performed": "VR-12 Physical component summary (PCS) score – oblique method T-score"] VR12PhysicalAssessment
  2. "No VTE Prophylaxis Medication Administered or Ordered" NoVTEMedication

- **Do not** reuse the same alias for different data elements, because this can reduce the clarity and readability of the logic. See the following for an example to avoid. The alias **`HeartRate`** is reused in two different definitions in the same measure, each with a different scope.

[26] Example from CMS156v13 for 2025 reporting.

[27] Examples from CMS22v13, CMS144v13, and CMS69v13 for 2025 reporting.

[28] Examples from CMS56v13 and CMS108v13 for 2025 reporting.

Example:

```
First(["Encounter, Performed": "Heart Rate Visit"] HeartRate
       with ["Diagnosis": "Essential Hypertension"]
       Hypertension such that HeartRate.relevantPeriod overlaps
       Hypertension.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 or value set title, because this can reduce the clarity and readability of the logic and raise the risk of logical errors.
- **Do not** use an alias if the definition statement does not require additional logic, because it is unnecessary and does not add any value.

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 are clearer and more readable.

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

| Alias names to avoid | Better alias names |
|---|---|
| D *or*<br>Dx *or*<br>Diagnosis | HeartFailure<br>Pregnancy<br>Asthma<br>Bradycardia |
| Med *o*<br>Medication | BetaBlockerOrdered<br>AntidepressantAdministered |
| 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 a measure)<br>InpatientEncounter<br>HeartFailureEncounter<br>Psychotherapy |
| ["Physical Exam, Performed": "Heart Rate"] **Exam** | HeartRateExam |
| ["Diagnostic Study, Performed": "Ejection Fraction"] **Study** | EjectionFractionStudy |

Use Table 4 as a guide for further improving alias names. The left column lists examples of aliases. The right column shows alternatives that are clearer and more descriptive.

**Table 4. Making good alias names more descriptive**

| Good alias name | Better alias name |
|---|---|
| HeartRate | FirstHeartRate |
| AntithromboticNotGiven | NoAntithrombotic |
| VisualExam | VisualFootExam |
| Fracture | LowerBodyFracture |

Use Table 5 as a guide for creating distinctions between two aliases that have 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<br>*and*<br>Heart failure diagnosis | HeartFailureEncounter<br>*and*<br>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 |
|---|---|
| Hospice discharge | DischargeToHospice |
| Hospice care order | HospiceOrder |
| Hospice intervention performed | HospicePerformed |

See the following example of these standard alias names.

Examples of CQL aliases:[29] **`HospiceOrder and HospicePerformed`**

```
or exists ( ["Intervention, Order": "Hospice Care Ambulatory"] HospiceOrder
    where HospiceOrder.authorDatetime during day of "Measurement Period"
)
or exists ( ["Intervention, Performed": "Hospice Care Ambulatory"] HospicePerformed
    where Global."NormalizeInterval" ( HospicePerformed.relevantDatetime, HospicePerformed.relevantPeriod ) overlaps day of "Measurement Period"
)
```

Accurate CQL alias names can aid in reading logic that models similar concepts. See the following example that uses the concept of hospice.

Examples of CQL aliases:[30] **`HospiceEncounter, HospiceAssessment, HospiceOrder, HospicePerformed`**

```
or exists ( ["Encounter, Performed": "Hospice Encounter"] HospiceEncounter
    where HospiceEncounter.relevantPeriod overlaps day of "Measurement Period"
)
or exists ( ["Assessment, Performed": "Hospice care [Minimum Data Set]"] HospiceAssessment
    where HospiceAssessment.result ~ "Yes (qualifier value)"
      and Global."NormalizeInterval" ( HospiceAssessment.relevantDatetime, HospiceAssessment.relevantPeriod ) overlaps day of "Measurement Period"
)
or exists ( ["Intervention, Order": "Hospice Care Ambulatory"] HospiceOrder
    where HospiceOrder.authorDatetime during day of "Measurement Period"
)
or exists ( ["Intervention, Performed": "Hospice Care Ambulatory"] HospicePerformed
    where Global."NormalizeInterval" ( HospicePerformed.relevantDatetime, HospicePerformed.relevantPeriod ) overlaps day of "Measurement Period"
)
```

[29] Example from CMS125v13 for 2025 reporting.

[30] Example from CMS125v13 for 2025 reporting.

## 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 from the global shared library. Functions perform operations on input arguments, while definitions operate only on the expressions in the definition.

### A. Selecting functions

To differentiate similar functions, choose from the predefined list of functions from the global shared library. Select the function that is most appropriate to meet the measure's intent. See the following examples of preferred functions from the global shared library.

- **`Global."NormalizeInterval"()`** function:[31]

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

    if pointInTime is not null then Interval[pointInTime, pointInTime]
      else if period is not null then period
      else null as Interval<DateTime>
```

  - Use the **`NormalizeInterval`** function for QDM datatypes that have use cases for both a relevantDateTime and a relevantPeriod, which will reduce the implementation burden associated with variable use of timing attributes across measures.[32]
  - The **`NormalizeInterval`** function might be needed when using a sort clause.

  Example of NormalizeInterval function with sort clause (used in logic):[33]

```
Encounter with Elevated Blood Pressure Reading SBP 120 to 129 AND DBP less than 80

  "Qualifying Encounter during Measurement Period" QualifyingEncounter
   let EncounterLastSystolicBP: Last(["Physical Exam, Performed": "Systolic blood pressure"] SystolicBP
      where Global."NormalizeInterval"(SystolicBP.relevantDatetime, SystolicBP.relevantPeriod) during day of QualifyingEncounter.relevantPeriod
     sort by start of Global."NormalizeInterval"(relevantDatetime, relevantPeriod)
   ),
```

- Age functions:

  Example of CQL age function: **`"AgeInYearsAt"()`**

```
"AgeInYearsAt" (date from start of)
```

  - The **`AgeInYearsAt`** function calculates age using **birth date** and time. By adding "date from" within the parenthetical statement, the measure developer can augment the function to not use time in the calculation.

[31] For more information on the NormalizeInterval function, please review the "Teach Me CQL: Normalize Interval" video at https://www.youtube.com/watch?v=PKKta_fKKh8.

[32] Review the QDM documentation on the eCQI Resource Center for a full list of QDM datatypes and their attributes: https://ecqi.healthit.gov/qdm.

[33] Example from CMS22v13 for 2025 reporting.

- Please note that the pattern previously used to capture birth date—["Patient Characteristic Birthdate": "Birth date"]—is no longer best practice.

Example of AgeInYearsAt function (used in logic):[34]

```
"Major Depressive Disorder Encounter" MDDEncounter
  where "AgeInYearsAt" (date from start of "Measurement Period")>= 6
  and "AgeInYearsAt" (date from start of "Measurement Period")<= 16
```

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

  Example of CQL length-of-stay function: **`Global."LengthInDays"()`**

  - The **`LengthInDays`** function calculates the difference in midnight-to-midnight 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.
  - This function can be used to calculate the length of a hospital stay for an inpatient encounter, from admission to discharge.

  Example of LengthInDays function (used in logic):[35]

  **◢ Encounter with First ICU Location Stay Less Than 1 Day**

  ```
  "Encounter with ICU Location" QualifyingEncounterICU
    where Global."LengthInDays" ( VTE. "FirstICULocationPeriod" ( QualifyingEncounterICU ) ) < 1
  ```

  Example of CQL length-of-stay function:
  **`Global."HospitalizationWithObservationLengthOfStay"()`**

  - This function returns the length in days from the start of any immediately prior emergency department visit through the observation visit to the discharge for the given encounter.

  **◢ Global.HospitalizationWithObservationLengthofStay(Encounter "Encounter, Performed")**

  ```
  "LengthInDays"("HospitalizationWithObservation"(Encounter))
  ```

  Example of HospitalizationWithObservationLengthOfStay function (used in logic):[36]

  **◢ Encounter Less Than Two Days**

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

  Example of CQL length-of-stay function:[37]
  **`Global."HospitalizationWithObservationAndOutpatientSurgeryService"()`**

  - This function returns the total interval from the start of any immediately prior emergency department visit, outpatient surgery visit, or observation visit to the discharge of the given encounter.

[34] Example from CMS177v13 for 2025 reporting.

[35] Example from CMS190v13 for 2025 reporting.

[36] Example from CMS72v13 for 2025 reporting.

[37] Example from CMS529v5 for 2025 reporting.

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

    Encounter Visit
        let ObsVisit: Last(["Encounter, Performed": "Observation Services"] LastObs
                where LastObs.relevantPeriod ends 1 hour or less on or before start of Visit.relevantPeriod
                sort by
                end of relevantPeriod
        ),
        VisitStart: Coalesce(start of ObsVisit.relevantPeriod, start of Visit.relevantPeriod),
        EDVisit: Last(["Encounter, Performed": "Emergency Department Visit"] LastED
                where LastED.relevantPeriod ends 1 hour or less on or before VisitStart
                sort by
                end of relevantPeriod
        ),
        VisitStartWithED: Coalesce(start of EDVisit.relevantPeriod, VisitStart),
        OutpatientSurgeryVisit: Last(["Encounter, Performed": "Outpatient Surgery Service"] LastSurgeryOP
                where LastSurgeryOP.relevantPeriod ends 1 hour or less on or before VisitStartWithED
                sort by
                end of relevantPeriod
        )
        return Interval[Coalesce(start of OutpatientSurgeryVisit.relevantPeriod, VisitStartWithED),
        end of Visit.relevantPeriod]
```

Example of CQL length-of-stay function:[38]

```
Global."HospitalizationWithObservation"()
```

- This function returns the total interval for (1) admission to observation for the given encounter or (2) admission of any immediately prior emergency department visit to the observation for the given encounter.

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

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

## B. 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.[39]

  Example of CQL function name:[40]

  ```
  LastHistoryPretermBirth(Encounter "Encounter, Performed")
  ```

[38] Example from CMS71v14 for 2025 reporting.

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

[40] Example from CMS334v6 for 2025 reporting.

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

  Example of CQL function:

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

- **Do not** give a function the same name as a CQL predefined function, because this can raise the risk of logical errors.

Process step: Please review the predefined operators available in MADiE[41] to ensure the name of your function is not the same as that of any of the predefined operators.

[41] See the MADiE User Guide for a list of operators: https://www.emeasuretool.cms.gov/training-resources.

## 7. OTHER CQL BEST PRACTICES

### A. Population criteria

When using population criteria, be descriptive and specific, making sure names are easy to read. The following example shows how to improve the 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" |

For hospital measures, the recommended timing pattern for Initial Population criteria is an encounter that `ends during day of "Measurement Period."` The use of `"ends"` provides a single point of time comparison within the measurement period so the receiving systems can provide reports on a quarterly basis, while `"day of"` specifies day precision to avoid time zone offset and millisecond issues.

Example CQL:[42]

**◢ Encounter with Age 18 and Older**

```
["Encounter, Performed": "Encounter Inpatient"] InpatientEncounter
  where InpatientEncounter.relevantPeriod ends during day of "Measurement Period"
    and AgeInYearsAt(date from start of InpatientEncounter.relevantPeriod) >= 18
```

When the denominator population criteria are equivalent to the initial population criteria, state "Initial Population" for the Denominator, as shown below:

Example CQL:[43] Initial Population = Denominator

**◢ Initial Population**

```
"Has Qualifying Encounter"
and exists "Order for 12 Months of ADT in 3 Months Before to 9 Months After Start of Measurement Period"
```

**◢ Denominator**

```
"Initial Population"
```

- **Do** use brackets, **[** and **]**, to represent a closed interval and parentheses, **(** and **)**, for open intervals. This pertains to definitions containing age intervals to avoid inconsistencies and to harmonize logic across measures. The CQL Specification states that "intervals in CQL are represented by specifying the low and high points of the interval and whether the boundary is inclusive (meaning the boundary point is part of the interval) or exclusive (meaning the boundary point is excluded from the interval). Following standard mathematics notation,

[42] Example from CMS826v2 for 2025 reporting.

[43] Example from CMS645v8 for 2025 reporting.

inclusive (closed) boundaries are indicated with square brackets, and exclusive (open) boundaries are indicated with parentheses."[44]

- Consider the following statement:

  ```
  Interval[3, 5)
  ```

  This expression results in an interval that contains the integers 3 and 4 because the bracket represents a closed or inclusive boundary. But the interval does not contain 5 because the parenthesis represents an open or exclusive boundary. This example is for illustration only, because as noted below, it is not best practice to use mixed-boundary notation for intervals.

In the following example, the definition uses `Interval[18, 85]` to include patients ages 18 to 85 but exclude patients ages 17 and younger and 86 and older at the end of the measurement period.

Example CQL:[45] **`Age Interval`**

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

For intervals, avoid using mixed-boundary notation—for example, Interval(18, 85]—in the same logical statement.

## B. Additional timing phrases

CQL supports precision-based date and time comparisons. Be sure to consider whether date or dateTime is being evaluated in timing phrases.[46] Additional timing phrases might be needed when making a statement such as "A starts/ends before/after or concurrent with start of B." These additional timing phrases indicate a time precision in the logic and ignore the day component.

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

Example CQL:[47] "ends 1 day after day of start of"

[44] See the CQL Specification Author's Guide for more on interval values: https://cql.hl7.org/02-authorsguide.html#interval-values.

[45] Example from CMS165v13 for 2025 reporting.

[46] See the CQL Formatting and Usage Wiki for more on timing phrases: (1) https://github.com/cqframework/CQL-Formatting-and-Usage-Wiki/wiki/Authoring-Measures-in-CQL#time-interval-calculations and (2) https://github.com/cqframework/CQL-Formatting-and-Usage-Wiki/wiki/Cooking-with-CQL-Q&A:---Date-and-Time-Calculations.

[47] Example from CMS108v13 for 2025 reporting.

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

```
Encounter with Intervention Comfort Measures on Day of or Day After Procedure

from
  VTE."Encounter with Age Range and without VTE Diagnosis or Obstetrical Conditions" QualifyingEncounter,
  ["Procedure, Performed": "General or Neuraxial Anesthesia"] AnesthesiaProcedure,
  "Intervention Comfort Measures" ComfortMeasures
  where Global."NormalizeInterval" ( AnesthesiaProcedure.relevantDatetime, AnesthesiaProcedure.relevantPeriod ) ends 1 day after day of start of
QualifyingEncounter.relevantPeriod
    and Coalesce(start of Global."NormalizeInterval"(ComfortMeasures.relevantDatetime, ComfortMeasures.relevantPeriod), ComfortMeasures.authorDatetime)during day
of TJC."CalendarDayOfOrDayAfter" (
    end of Global."NormalizeInterval" ( AnesthesiaProcedure.relevantDatetime, AnesthesiaProcedure.relevantPeriod ) )
  return QualifyingEncounter
```

In addition, when assessing date-time valued elements for comparison to the measurement period, unless time-sensitive comparison is truly desired, use the **`date from`** operator to assess only the date portion of the date-time valued element. In the following example, the **`date from`** operator is used to compare the birth date at the date level of precision.[48]

```
TJC.Non Elective Encounter with Age

["Encounter, Performed": "Nonelective Inpatient Encounter"] NonElectiveEncounter
  where AgeInYearsAt(date from start of NonElectiveEncounter.relevantPeriod) >= 18
    and NonElectiveEncounter.relevantPeriod ends during day of "Measurement Period"
```

## C. Operator precedence

Precedence in CQL expressions is determined by the order of appearance in the expression, from left to right. To ensure consistent and predictable behavior in the order of operations in CQL expressions, use parentheses around a grouping to enforce higher precedence. See Appendix Table A.1 for more details.

CQL adopts expected operator precedence to ensure consistent and predictable behavior of written expressions. For example, primary operators—such as, [ ] and ( )—have the highest operator precedence, while conjunctions and disjunctions are lower on the list.[49] As with most expression languages, parentheses can always be used to force the order of operations if the defined operator precedence does not produce the intended evaluation of a given expression.

[48] Example from CMS71v14 for 2025 reporting.

[49] See Appendix Table A.1 for the full list.

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:[50] Operator precedence

**Denominator Exclusions**

```
Hospice."Has Hospice Services"
 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 AIFrailLTCF."Is Age 66 or Older with Advanced Illness and Frailty"
or AIFrailLTCF."Is Age 66 or Older Living Long Term in a Nursing Home"
or PalliativeCare."Has Palliative Care in the Measurement Period"
```

## D. Direct reference codes

Measure developers maintain and publish value sets on the Value Set Authority Center website. Value sets and codes are listed in the terminology section of each measure specification.[51] Direct reference codes (DRCs) are single codes that can be referenced directly in CQL logic, instead of creating a value set containing a single code. DRCs are recommended for all single codes and may be used for other single-use terminology codes.[52] 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 file; it might 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.

Example:[53] DRC used in CQL logic

**Rotavirus Numerator Inclusion Conditions**

```
( ["Diagnosis": "Anaphylaxis caused by rotavirus vaccine (disorder)"] RotavirusConditions
   where date from start of RotavirusConditions.prevalencePeriod during day of "First Two Years"
 )
```

**Terminology**

- code "Anaphylaxis caused by rotavirus vaccine (disorder)" ("SNOMEDCT Code (428331000124103)")

[50] Example from CMS125v13 for 2025 reporting.

[51] For more information on value sets, please review the eCQI Resource Center's Value Set Information page: https://ecqi.healthit.gov/value-set-information?qt-tabs_vsg=about.

[52] For more information on code systems, vocabularies, and terminologies, please review the MMS Hub's Specify the Code resource: https://mmshub.cms.gov/measure-lifecycle/measure-specification/specify-code/code-systems-vocabularies-terminologies.

[53] Example from CMS117v13 for 2025 reporting.

## E. Other style considerations

Using the following best practices will enhance the readability and usability of measure specifications:

- **Do not** copy and paste bullets into measure header text in MADiE. The bullets will not format correctly. Instead, use hyphens (-) to improve the readability of the text. If including citations for clinical recommendations, consider replacing bullet formatting with hyphens after moving text into the measure header to improve readability. **Any Word-based formatting can cause issues with exporting or the export artifacts when pasted into the measure header text in MADiE.**
- Use the equivalence operator (~) to indicate equivalence between two concepts. Use the inequivalence operator (!~) to indicate inequivalence between two concepts.

  Example:[54]

  **Qualifying Encounter During Measurement Period**

```
( ["Encounter, Performed": "Office Visit"]
  union ["Encounter, Performed": "Ophthalmological Services"]
  union ["Encounter, Performed": "Outpatient Consultation"]
  union ["Encounter, Performed": "Care Services in Long Term Residential Facility"]
  union ["Encounter, Performed": "Nursing Facility Visit"] ) QualifyingEncounter
  where QualifyingEncounter.relevantPeriod during "Measurement Period"
    and QualifyingEncounter.class !~ "virtual"
```

  Example:[55]

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

[54] Example from CMS142v13 for 2025 reporting.

[55] Example from CMS146v13 for 2025 reporting.

## VERSION HISTORY

| Version | Date | Description of change |
|---|---|---|
| 2.0 | August 2018 | Initial publication |
| 3.0 | May 2019 | Removed references related to retired QDM logic |
| 3.0 | May 2019 | Updated examples of logic to reflect most recent use |
| | | Added 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<br>QDM 5.4 and CQL 1.3 |
| | | Removed reference to Keyword-Distinct |
| | | Updated examples using birth date to reflect the addition of birth date, a direct reference code |
| 4.0 | May 2020 | Updated language and content to align with standards changes<br>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 'NormalizeInterval' function to the 'Selecting functions' section |
| 6.0 | May 2022 | Updated examples of logic to reflect most recent use |
| 7.0 | May 2023 | Updated examples of logic to reflect most recent use |
| 8.0 | May 2024 | Updated examples of logic to reflect most recent use |
| 9.0 | May 2025 | Updated examples of logic to reflect most recent use |
| | | Updated all tooling references from MAT to MADiE |
| | | Added new examples, references, and guidance on case types |
| | | Updated title of guide from 'CQL Style Guide' to 'QDM-based CQL Style Guide' |

# APPENDIX A:

# THE ORDER OF OPERATOR PRECEDENCE IN CQL

## The Order of Operator Precedence in CQL

Operator precedence determines the order in which operators are evaluated in expressions. Parentheses can be used to override the default precedence and ensure expressions are evaluated in the desired order.

### 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"**<br>*component* **"singleton from"** |
| **Exponentiation** | ^ |
| **Multiplicative** | * / **"div mod"** |
| **Additive** | + - & |
| **Conditional** | **"if..then..else**<br>**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"**<br>*precision* **"between"**<br>**"duration in"** *precision* **"between"**<br>**"difference in"** *precision* **"between"** |
| **Binary List** | **"union intersect except"** |
| **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"** |

Source: CQL Specification Developer's Guide, Operator Precedence Section (https://cql.hl7.org/N1A/03-developersguide.html#operator-precedence).

Note: Bolded text enclosed by quotes is used to identify CQL operator language.

# 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 white spaces between 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 white spaces between characters allowed.
- **Title Case.** Standard title casing, including spaces and tabs, but no other white spaces between characters allowed.
- **Initial Case.** First letter of every word is capitalized (for example, "Includes Or Starts During"), as opposed to title case, which traditionally does not capitalize conjunctions and prepositions (for example, "Includes or Starts During").
    - Please note: Use of initial case is a best practice recommended by the Using CQL with FHIR v1.0.0 resource. To support future transition from QDM to FHIR, early adoption of initial case in both QDM and FHIR measure specifications is encouraged.