Clinical Quality Language (CQL) Style Guide

Version 2 – Last Revised: August 21, 2018

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

This page has been left blank for double-sided copying.

CLINICAL QUALITY LANGUAGE (CQL) STYLE GUIDE

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. 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

2. INTRODUCTION

The purpose of the style guide is to standardize expression of measure concepts across electronic clinical quality measures (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 Centers for Medicare & Medicaid Services (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.

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.

3. STANDARDS – LIBRARIES

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

***Local* library** – one or more CQL libraries that can be shared among several measures or a specific group of measures

- *Example:* The local CQL library name "**Hospice"** is used in a CQL definition statement below. "**Hospice"** library contains multiple expressions of hospice-related relevant CQL definitions that apply to hospice care (for example, "Discharge to Hospice," "Hospice Admissions," and so on).

- **Denominator Exclusions**

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

***"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 is accessible to all MAT users. Using the "Global" common library helps to reduce duplication and maintain consistency across measure specifications.

- *Example:* The CQL library name **"Global"** is used in the definition below. Note: The global common library definition **Inpatient Encounter** is recommended for use in all hospital measures.
  - define "**Inpatient Encounter**"

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

The version of the global library referenced in this document is **version 2.0.0**.

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

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

- Do NOT start the library name with a special character,[1] number, or underscore.[2]
- Do NOT use spaces or special characters within the library name.[2]
  - Exception: Use of underscore "_" is occasionally allowed, though not typical.
- DO use PascalCase[3] (capitalizing the first letter of every word, without spaces between words). See Appendix B for a complete list of case type definitions.

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

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

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

- DO use names that inform the reader of the contents of the library.

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:

- DO create names that are descriptive, understandable, and meaningful to the context.
- DO use Title Case.[3]
- DO allow for spaces.
- Do NOT use special characters[1] in definition names.
- DO use only abbreviations or acronyms that are consistent with terminology used in the eCQM header.
- Do NOT name a definition the same as a value set name.[4]
- DO create definition names that are clear and indicate result type (a yes/no should be named like a question [for example, "Is" or "Has"], a list of Encounters should be named "Encounters...", a list of Procedures should not be named "Encounters with xxx", and so on).

Reference the following examples that use descriptive text, title case, and appropriate spacing.

- Example 1: Newborn Fed Breast Milk Only Since Birth

```
define "Newborn Fed Breast Milk Only Since Birth"

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

- Example 2: Most Recent Elevated HbA1c

```
define "Most Recent Elevated HbA1c":
```

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

```
Last(["Laboratory Test, Performed": "HbA1c Laboratory Test"]
HighHbA1c
where HighHbA1c.relevantPeriod during "Measurement Period"
and HighHbA1c.result is not null
```

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

Table 1: Making good definition names better (more human readable)

Good definition name	Better definition name
Anticoagulant Not Given at Discharge	Reason For Not Giving Anticoagulant at Discharge
Encounter with Principal Diagnosis of Ischemic Stroke	Ischemic Stroke Encounter
In Demographic	Single Live Birth Encounter With Gestational Age 37 Weeks or More
Lab Test With Result	Most Recent Elevated HbA1c

B. Standards for naming definitions

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

Table 2: Standard definition names for use across measures

Concept	Recommended definition name
Hospice Exclusions Exclusions for Hospice	Hospice Exclusions
Qualifying Encounter *or* Eligible Encounter *or* Valid Encounter	Qualifying Encounter *or* Initial Qualifying Encounter *or* Follow Up Qualifying Encounter

5. STANDARDS – ALIASES OR ARGUMENT NAMES

Aliases, or argument names, are identifiers that reference individual or primary CQL expressions. 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. Similar 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.[5]
- DO use names that are short, if possible.
- DO use names that are descriptive and provide an accurate representation of the clinical concept.
- DO capitalize the first letter of the alias name, even if the name is a single word.
- DO create alias names that are clinically focused and human readable.
- DO use only abbreviations or acronyms that are consistent with terminology used in the eCQM header.
- Do NOT repeat aliases, unless a single alias carries the same content and scope as its original definition when reused within a measure.
- Do NOT give an alias the same name as the definition name.
- Do NOT use an alias if the definition statement does not require additional logic (see examples below).

Reference the following examples of alias names that are meaningful and descriptive, and that use PascalCase.

- Example 1: **ComfortMeasuresOnly**

```
define "Comfort Measures during Hospitalization"

  TJC."Encounter with Principal Diagnosis of Ischemic Stroke"
  NonElectiveEncounter
  with "Intervention Comfort Measures" ComfortMeasuresOnly
  such that Coalesce (start of ComfortMeasuresOnly.relevantPeriod
  ComfortMeasuresOnly.authorDatetime) during
  Global."Hospitalization"(NonElectiveEncounter)
```

- Example 2: **NoDischargeAntithrombotic**

```
define "Denominator Exceptions For No Antithrombotic"

  TJC."Encounter with Principal Diagnosis of Ischemic Stroke"
  IschemicStrokeEncounter
  with "Antithrombotic Not Given at Discharge"
  NoDischargeAntithrombotic
  such that NoDischargeAntithrombotic.authorDatetime during
  Encounter.relevantPeriod
```

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

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

Table 3: Making aliases clinically focused and human readable

Alias names to avoid	Better alias names
D *or* Dx *or* Diagnosis	HeartFailure Pregnancy Asthma Bradycardia
Med Medication	BetaBlocker Antidepressant
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 the following table as a guide for improving alias names even further. The left-hand column lists examples of aliases. The right-hand column includes example alternatives that offer improved description and human readability.

Table 4: Making good alias names better (more human readable)

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-hand column) and a guide for adding specificity (right-hand column).

Table 5: Creating distinction between similar alias names

Similar alias names within a measure	Similar alias names with specificity
Heart failure encounter Heart failure diagnosis	HeartFailureEncounter*and* HeartFailureDiagnosis

B. Standards for naming aliases

Use the table below as a guide for naming aliases that use common concepts across measures. The left-hand column represents the alias concept, and the right-hand 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 on alias names.

- Example 1: The "**BirthDate**" alias is used consistently in different CQL definition statements, and in multiple places within and across measures.

```
define "Patient Demographic"
      ["Patient Characteristic Birthdate": "Birthdate"] BirthDate
      where BirthDate.birthDatetime occurs 5 months or less before
      start of "Measurement Period"
```

- Example 2: Hospice aliases

```
Hospice Exclusions
    exists ( ["Encounter, Performed": "Encounter Inpatient"]
    DischargeHospice
        where ( DischargeHospice.dischargeDisposition in
        "Discharged to Home for Hospice Care"
```

```
            or DischargeHospice.dischargeDisposition in "Discharged to
            Health Care Facility for Hospice Care"
            )
            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 the shared CQL "Global" common library, **MATGlobalCommonFunctions**, maintained and updated by Mathematica Policy Research.

To differentiate among similar functions in the MAT (predefined) or "Global" common library, choose the one that's most appropriate for your needs and that meets the measure intent.

Reference the following examples of functions available in the MAT:

- Example 1: Selecting between age calculation functions **AgeInYearsAt()** vs. **Global.CalendarAgeInYearsAt()**
    - **AgeInYearsAt()** calculates age using both **Birth Date *and* Time**.
        - Use this function when a more granular level of age calculation is needed (for example, *pediatric or neonatal* measures).
        - To determine the time from birth to an intervention in the NICU, **AgeInYearsAt()** is more useful, as it provides both **Birth Date *and* Time**.
    - **Global.CalendarAgeInYearsAt()** calculates age using **only Birth Date**.
        - Use this function for measures pertaining to *adult* patients, or as applicable.
        - For example, use this function to determine all adult patients aged 65 or older at the beginning of the measurement period.

          ```
          Global.CalendarAgeInYearsAt(start of "Measurement
          Period")>= 65
          ```

- Example 2: Selecting between length of stay calculations (generally used for hospital measures): **Global.LengthInDays()** vs. **Global.“HospitalizationLengthOfStay”()**

    - **Global.LengthInDays()**

        - Use this function for measures that calculate length of hospital stay from admission to discharge.

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

        - This function is replacing the QDM attribute that will be retired “LengthOfStay” to calculate the length of stay from Admission Datetime to Discharge Datetime.

        - For example, use the function **Global.LengthInDays()** for length of hospital stay for inpatient encounter from admission to discharge.

```
Inpatient Encounter

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

    - **Global.“Hospitalization Length of Stay”()**

        - This function returns the difference in days from ED admission Datetime to inpatient discharge Datetime if the given Encounter contains an ED visit.

        - For example, define function "Hospitalization Length of Stay"(Encounter "Encounter, Performed"):

            LengthInDays("Hospitalization"(Encounter))

- Example 3: A comparison of QDM logic vs. CQL function logic using **Global.Hospitalization()**

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

- The table below shows that one CQL expression using **Global.Hospitalization()** efficiently covers two timing phrases in original QDM UNIONs.

Table 7: Comparison of QDM and CQL expressions using Global.Hospitalization()

QDM expression	CQL expression
Union of: "Diagnosis: Obstetrics" $DiagnosisVTE starts during Occurrence A of $EncounterInpatient Union of: "Diagnosis: Obstetrics" $DiagnosisVTE starts during ("Encounter, Performed: Emergency Department Visit" <= 1 hour(s) ends before or concurrent with start of Occurrence A of $EncounterInpatient)	define "Admissions for Obstetrical Conditions with VTE" "Inpatient Encounter" Encounter with (["Diagnosis": "Obstetrics"] union ["Diagnosis": "Venous Thromboembolism"] union ["Diagnosis": "Obstetrics VTE"]) Diagnosis such that Diagnosis.prevalencePeriod starts during **Global."Hospitalization"(Encounter)**

A. Best practices and standards for naming new CQL functions

New function names should be clinically focused and human readable. When naming functions, use the following standards:

- DO use PascalCase.[6]
- DO use parentheses at the end, even if the function has no arguments.
- DO use spaces after commas to separate arguments, always.
- DO continue an argument list across multiple lines if necessary, but keep the opening parenthesis on the same line as the function identifier, and indent subsequent lines one level.
- Do NOT put a space between the opening and closing parentheses if the function has no arguments.
- Do NOT put a space between the function name and the argument list, or between the opening and closing parentheses and the arguments.

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

- Do NOT attempt to right-align indented content when continuing an argument list, as this leads to unnecessary maintenance to preserve the alignment.

7. OTHER CQL BEST PRACTICES

A. Population criteria

When using population criteria, be descriptive and specific, and make sure names are human readable. Below is an example of how to improve population criteria.

Current population criteria	Improved population criteria
Initial Population o "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, call the "Initial Population" definition as illustrated below:

```
define "Initial Population":
      "Encounter Age and No Diagnosis of VTE or Obstetrics"
define "Denominator":
      "Initial Population"
```

B. Keyword: distinct

The example below shows that the keyword "distinct" is not required, as the use of UNION eliminates duplicate results. Also, in this example, to ensure a "day" is a unit in the calculation, the time part of the Datetime should NOT be included, and the additional words "day of" are needed. Also, be aware that the start of an Intervention could be before the start of Hospitalization as long as both are on the same date.

Previous use of "distinct"	Revised CQL expression and phrasing (time is NOT included)
distinct (TJC."Encounter with Principal Diagnosis of Ischemic Stroke" Encounter where Encounter.dischargeDisposition in "Discharge To Acute Care Facility" or Encounter.dischargeDisposition in "Left Against Medical Advice" or Encounter.dischargeDisposition in "Patient Expired")	Intervention Comfort Measures During Hospitalization And on Day of or Day After Admission "Encounter Age and No Diagnosis of VTE or Obstetrics" QualifyingEncounter with "Intervention Comfort Measures" Intervention such that Coalesce(start of Intervention.relevantPeriod, Intervention.authorDate)1 day or less on or after **day of** start of Global."Hospitalization"(QualifyingEncounter)

An additional timing phrase is needed when stating "A starts after / before or concurrent with start of B" besides "day of" expression.

The following example shows use of a "**1 day or less on or after day of start"** CQL expression. An additional timing constraint is added to ensure **NoVTEProphylaxisPatientRefusal** starts after or is concurrent with the start of Encounter.

```
No VTE Prophylaxis Patient Refusal on Day of or Day After Admission

from
     "Encounter Age and No Diagnosis of VTE or
Obstetrics" QualifyingEncounter,
     "No VTE Prophylaxis Patient Refusal"
NoVTEProphylaxisPatientRefusal
     where
NoVTEProphylaxisPatientRefusal.authorDate 1
day or less on or after day of start of
QualifyingEncounter.relevantPeriod
            and
NoVTEProphylaxisPatientRefusal.authorDatetime
on or after start of
QualifyingEncounter.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.

QDM Expression	CQL Expression
• Denominator Exclusions = o OR: Count= 2 : "Procedure, Performed: Unilateral Mastectomy" ends before end of "Measurement Period" o OR: ▪ AND: Union of: ▪ "Diagnosis: Status Post Right Mastectomy" ▪ "Diagnosis: Unilateral Mastectomy, Unspecified Laterality (anatomical location site: Right)" ▪ starts before end of "Measurement Period" ▪ AND: Union of: ▪ "Diagnosis: Status Post Left Mastectomy" ▪ "Diagnosis: Unilateral Mastectomy, Unspecified Laterality (anatomical location site: Left)" ▪ starts before end of "Measurement Period" o OR: "Diagnosis: History of bilateral mastectomy" starts before end of "Measurement Period"	**Denominator Exclusions** Hospice."Has Hospice Exclusions" or (Count("Unilateral Mastectomy ")= 2) or **(exists "Right Mastectomy" and exists "Left Mastectomy")** or exists "History Bilateral Mastectomy" or exists "Bilateral Mastectomy " *Added parentheses around "exists 'Right Mastectomy'" and "exists 'Left Mastectomy'" promote operator precedence*

This page has been left blank for double-sided copying.

APPENDIX A:

THE ORDER OF OPERATOR PRECEDENCE IN CQL FROM HIGHEST TO LOWEST

Table 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/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
Unary Test	is null/true/false
Type Operators	is as cast..as
Unary Logical	not exists
Between	between 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
Binary List	union intersect except

Source: HL7 CQL Specification “Operator Precedence” section.

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