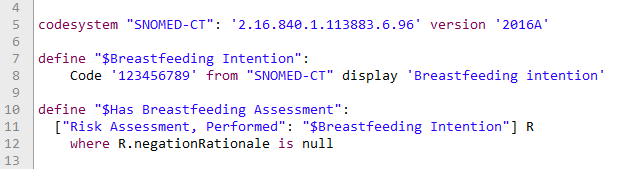
**CQL Style Guide**

1. **Syntax Element Definitions**
   1. *Whitespace* - Whitespace defines the separation between all tokens in the language
   2. *Comment* - Comments are ignored by the language, allowing for descriptive text
   3. *Literal* - Literals allow basic values to be represented within the language
   4. *Symbol* - Symbols such as +, -, \*, and /
   5. *Keyword* - Grammar-recognized keywords such as define and where
   6. *Identifier* - User-defined identifiers
2. **Case Related Conventions** 
   1. CQL is case-sensitive
      1. Encourages consistency
      2. Reduces potential naming clashes with keywords in the language
   2. Different types of casing:
      1. lowercase
      2. camelCase
      3. PascalCase
   3. Keywords are always lowercase
   4. System library functions are always PascalCase
      1. Sum({ 1, 2, 3 })
   5. System type names are always PascalCase
      1. Integer, Interval<Integer>
3. **Spacing Conventions**
   1. Use tabs to indent
      1. Indent one tab for related content (more on this in the clauses section)
   2. Always separate keywords and operators from other symbols by at least a space
   3. Always use a space after a comma
   4. Never use a space before or after a period
4. **Identifiers**
   1. Simple Identifiers:
      1. Any alphabetic character or underscore, followed by any number of alpha-numeric characters or underscores
         1. \_simpleIdentifier
         2. simpleIdentifier
         3. SimpleIdentifer24
   2. Quoted Identifiers:
      1. Any sequence of characters enclosed in double-quotes
         1. “Quoted identifiers use double-quotes”
         2. “To include a double-quote in a quoted identifier, use a backslash (\”)”
5. **QDM Measure Authoring Conventions**
   1. Named Expression Identifiers (Top-Level Expressions, or Variables)
      1. Always use quoted identifiers
      2. Use Pascal Case and appropriate spacing
      3. Provide a descriptive, meaningful name
   2. Data Type Names
      1. Always use quoted identifiers
      2. Use Pascal Case and appropriate spacing
      3. Use the friendly name
         1. [“Encounter, Performed”]
   3. Negation
      1. When building a criteria involving any QDM Data Type that has a *negationRationale* attribute, by definition, the CQL expression will return all matching data, regardless of whether they were negated or not.
      2. To ensure that the correct data is returned, by convention, any negatable QDM Data Type should include an additional restriction must be used to specify whether negation is desired:
      3. To reference a single code, rather than defining a value set, use a CQL define:



* + 1. Identify “regions” for each section of the measure using comments:

// Region: Initial Population 1

define "$Initial Population 1":  
 "$Live Birth Infants without Parenteral Nutrition"

// EndRegion: Initial Population 1

* + 1. Positive Assessments:

define "Has Breastfeeding Assessment":  
 ["Risk Assessment, Performed": "Breastfeeding Intention"] R  
 where R.negationRationale is null.

* 1. Attribute Names
     1. Never use quoted identifier
     2. Use camelCase
        1. admissionDateTime

1. **Operators and Functions**
   1. Operators are always keywords, and always lowercase
      1. Binary operators are always infix
      2. Unary operators are always prefix
      3. Always use a space before and after operators
         1. result > 5 ‘mg’
         2. not exists ([“Encounter, Performed”])
   2. Functions are never keywords, always PascalCase
      1. Functions always use parentheses (even if the function has no arguments)
      2. Never put a space between the function name and the argument list
      3. Always use spaces after commas
      4. If necessary, an argument list can be continued across multiple lines
         1. Keep the opening parenthesis with the function name
         2. Indent subsequent lines one level
2. **Literals**
   1. Boolean
      1. true false // keywords
   2. Numbers
      1. 45, 0, -20 // Integers
      2. 98.6, 25.0 // Decimals
   3. Quantities
      1. 45 ‘mg’
      2. 28 ‘mm[Hg]’
   4. DateTime
      1. ‘@’ symbol, followed by an ISO-8601 date/time value
         1. @2014
         2. @2014-01
         3. @2014-01-25
         4. @2014-01-25T12
         5. @2014-01-25T12:30
         6. @2014-01-25T12:30:30.0Z // specifies UTC timezone
         7. @2014-01-25T12:30:30.0+00:30 // specifies timezone as an offset
   5. Time
      1. ‘@T’ followed by an ISO-8601 time value
         1. @T12:30:30.0Z // same as the time portion of the DateTime literal
   6. Strings
      1. ‘String literals use single quotes’
      2. ‘To include a single quote in a string, use a backslash (\’)’
3. **Intervals**
   1. Intervals can be constructed from any type that supports ordered comparison
   2. Intervals use standard mathematical notation for open/closed
      1. Interval[1, 5]
      2. Interval(1, 9)
      3. Interval[@2015-01-01T00:00:00Z, @2016-01-01T00:00:00Z)
   3. Never put a space before or after the opening or closing boundary
   4. Always put a space after the comma
4. **Lists, Tuples**
   1. Lists can contain elements of any type
   2. Always separate the contents of the list with a space:
      1. { 1, 2, 3 } // The space helps distinguish the braces from parenthesis
      2. Sum({ 1, 2, 3 }) // Even when the list is itself an argument
   3. Tuples contain named elements of any type
   4. Always separate the contents of the tuple with a space:
      1. { name: ‘Patrick’, birthDate: @2014-01-01 }
   5. The Tuple keyword is optional
      1. { } // empty List
      2. { : } // empty Tuple
5. **Queries** 
   1. The central expression construct of CQL is the query
   2. In general, simple queries can fit on a single line
      1. [“Encounter, Performed”: “Inpatient”] E where duration in days of E.period >= 120
   3. If a clause needs more than one line, continue the clauses indented beneath the clause:
      1. “$Pharyngitis Encounters with Antibiotics” P  
          with [“Laboratory Test, Performed”: “Group A Streptococcus Test”] T  
          such that T.result is not null  
          and T.startDateTime in Interval[E.startDateTime - 3 days, E.stopDateTime + 3 days]  
          with [“Encounter, Performed”] ….
   4. Clause-based construct

<primary source> <alias>  
 <with or without clauses>  
 <where clause>  
 <return clause>  
 <sort clause>

1. **Syntax Highlighters**
   1. The following categories should be displayed using different styles in a syntax highlighter:
      1. Symbols
      2. Keywords
      3. Operators
      4. Literals
         1. Numbers
         2. Strings
         3. Dates and Times
      5. Comments
      6. Identifiers
         1. Type Identifiers
         2. Variable Identifiers
         3. Function Identifiers