Cooking with CQL Q&As

Session 55 – Thursday, June 24, 2021

# Queries in CQL

**Q:** When using Global Common Library v5 to calculate age, the FHIRHelpers.ToDate(Patient.birthDate) function must be explicitly used by the measure developer within the logic expression. However, in the Global Common Library v6, the FHIRHelpers.ToDate(Patient.birthDate) function does not have to be written in the logic expression to generate the equivalent result. Does Global Common Library v6 use FHIRHelpers.ToDate(Patient.birthDate) function in a tooling layer in the code?

**A:** Yes, the change to remove age calculation functions from Global Common Library v6 means that when the system-defined Age calculation functions are used, the FHIRHelpers.ToDate function will be applied as part of the tooling. More specifically, one of the functions in CQL is the ability to calculate the age of the patient. The CQL model information understands the patient context and knows how to retrieve the patient birthdate. Therefore, the measure developers do not need to write the FHIRHelpers.ToDate(Patient.birthDate) function in the logic expression, since the function is included in the generated code to transform the birthdate expressed as a FHIR date into a CQL date.

Example Logic Expression using Global Common Library v5

|  |  |
| --- | --- |
| define "HPV Test Within 5 Years for Women Age 30 and Older (124)": | |
|  | [Observation: "HPV Test"] HPVTest | |
|  | where HPVTest.value is not null | |
|  | and HPVTest.status in { 'final', 'amended', 'corrected', 'preliminary' } | |
|  | and Global."CalendarAgeInYearsAt"(FHIRHelpers.ToDate(Patient.birthDate), start of Global."Normalize Interval"(HPVTest.effective)) >= 30 | |
|  | and Global."Normalize Interval"(HPVTest.effective) ends 5 years or less on or before end of "Measurement Period" | |

Example Logic Expression using Global Common Library v6

|  |  |
| --- | --- |
| define "HPV Test Within 5 Years for Women Age 30 and Older (124) (Global v6)": | |
|  | [Observation: "HPV Test"] HPVTest | |
|  | where HPVTest.value is not null | |
|  | and HPVTest.status in { 'final', 'amended', 'corrected', 'preliminary' } | |
|  | And AgeInYearsAt(start of Global6."Normalize Interval"(HPVTest.effective)) >= 30 | |
|  | and Global6."Normalize Interval"(HPVTest.effective) ends 5 years or less on or before end of "Measurement Period" | |

**Q:** In the Influenza Vaccine measure (CMS 147), when following the negation rationale pattern for Immunization Not Done queries in Clinical Quality Language (CQL), if more than one criterion is returned in the query, does the expression stop after the first criterion is met and returned?

**A:** If a patient has multiple codes from the negation rationale value sets, the “Medical Patient or System Reason for Not Administering Influenza Vaccine” definition will return multiple items. Multiple items are then converted by the code into a Boolean for this measure because the Denominator Exclusions population criteria uses an exists operator.

|  |  |
| --- | --- |
| define "Medical Patient or System Reason for Not Administering Influenza Vaccine": | |
|  | [Immunization: "Influenza Vaccine"] NoFluVaccine |
|  | where NoFluVaccine.recorded during "Influenza Season Including August and September of the Prior Year" |
|  | and NoFluVaccine.status = 'not-done' |
|  | and ( |
|  | NoFluVaccine.statusReason in "Medical Reason" |
|  | or NoFluVaccine.statusReason in "Patient Reason" |
|  | or NoFluVaccine.statusReason in "System Reason" |
|  | or NoFluVaccine.statusReason in "Influenza Vaccination Declined" |
|  | ) |

# Value Sets and CQL

**Q:** In the library ImmunizationNotDoneExample, using Fast Healthcare Interoperability Resources® (FHIR®) version 4.0.1, does the measure developer need to specify the define “Influenza Vaccine Not Administered” in the measure since the code is happening in the background?

Following the negation rationale pattern

|  |  |
| --- | --- |
| define "Medical Patient or System Reason for Not Administering Influenza Vaccine": | |
|  | [Immunization: "Influenza Vaccine"] FluVaccine |
|  | where FluVaccine.recorded during "Influenza Season Including August and September of the Prior Year" |
|  | and FluVaccine.status = 'not-done' |
|  | and ( |
|  | FluVaccine.statusReason in "Medical Reason" |
|  | or FluVaccine.statusReason in "Patient Reason" |
|  | or FluVaccine.statusReason in "System Reason" |
|  | or FluVaccine.statusReason in "Influenza Vaccination Declined" |
|  | ) |

Logic happening underneath within the Expression Logical Model (ELM)

|  |  |
| --- | --- |
| define "Influenza Vaccine Not Administered": | |
|  | [Immunization] I |
|  | where I.vaccineCode in "Influenza Vaccine" |
|  | or ToValueSet(Global.GetBaseExtension(I.vaccineCode, 'valueset-reference').value as FHIR.uri) = "Influenza Vaccine" |

|  |  |
| --- | --- |
| define function ToValueSet(uri FHIR.uri): | |
|  | System.ValueSet { id: uri.value } |

**A:** Correct, the define "Influenza Vaccine Not Administered" and define function ToValueSet(uri FHIR.uri) illustrate what is happening in the ELM and do not need to be written into the logic expression. Measure developers should always utilize the negation rationale pattern and the logic expressions to derive the appropriate values.

Negation Rationale Pattern:

|  |  |
| --- | --- |
| define "Medical Patient or System Reason for Not Administering Influenza Vaccine": | |
|  | [Immunization: "Influenza Vaccine"] FluVaccine |
|  | where FluVaccine.recorded during "Influenza Season Including August and September of the Prior Year" |
|  | and FluVaccine.status = 'not-done' |
|  | and ( |
|  | FluVaccine.statusReason in "Medical Reason" |
|  | or FluVaccine.statusReason in "Patient Reason" |
|  | or FluVaccine.statusReason in "System Reason" |
|  | or FluVaccine.statusReason in "Influenza Vaccination Declined" |
|  | ) |
|  |  |

# Functions in CQL

**Q:** What are the differences between the functions "GetBaseExtensions", "GetExtensions", and "GetExtension" in the current Global Common Library?

**A:** "GetExtension" and "GetExtensions" are used to return extensions from FHIR elements and domain resources. Measure developers should use the plural if the extension is allowed to be specified multiple times (i.e. has a cardinality greater than 1 in the profile definition. The singular function ensures that only one extension will be returned to the measure developer as a result of the function. If the singular function is used on an extension that is allowed to be specified multiple times, it will result in an error if it is used with an instance that has multiple appearances of the extension. The function "GetBaseExtensions" provides a default for the base Uniform Resource Locator (URL), instead of writing out the entire URL. Extensions in FHIR are defined by a complete URL, for example <http://hl7.org/fhir/StructureDefinition/valueset-source>. The GetBaseExtension functions allow only the “tail” of this URL to be used, defaulting the portion that is common for all base FHIR extensions: <http://hl7.org/fhir/StructureDefinition/>.

|  |  |  |
| --- | --- | --- |
| define function "GetExtensions"(element Element, url String ): | | |
|  | element.extension E |
|  | where E.url = (url) |
|  | return E |

|  |  |  |
| --- | --- | --- |
| define function "GetExtension"(element Element, url String ): | | |
|  | singleton from "GetExtensions"(element, url) |

|  |  |
| --- | --- |
| define function "GetBaseExtensions"(domainResource DomainResource, url String ): | |
|  | domainResource.extension E | |
|  | where E.url = ('http://hl7.org/fhir/StructureDefinition/' + url) | |
|  | return E | |

|  |  |
| --- | --- |
| define function "GetBaseExtension"(domainResource DomainResource, url String ): | |
|  | singleton from "GetBaseExtensions"(domainResource, url) |