<https://github.com/esacinc/CQL-Formatting-and-Usage-Wiki/tree/master/Source/Libraries>

On the Wiki there are now documents for version 2.0 and version 3.0

We can use version 3.0 as a redline draft for future changes.

library MATGlobalCommonFunctions version '2.0.000'

using QDM version '5.3'

codesystem "LOINC:2.46": 'urn:oid:2.16.840.1.113883.6.1' version 'urn:hl7:version:2.46'

codesystem "SNOMEDCT:2016-03": 'urn:oid:2.16.840.1.113883.6.96' version 'urn:hl7:version:2016-03'

valueset "Emergency Department Visit": 'urn:oid:2.16.840.1.113883.3.117.1.7.1.292'

valueset "Encounter Inpatient": 'urn:oid:2.16.840.1.113883.3.666.5.307'

valueset "Intensive Care Unit": 'urn:oid:2.16.840.1.113762.1.4.1110.23'

code "Birthdate": '21112-8' from "LOINC:2.46" display 'Birth date'

code "Dead": '419099009' from "SNOMEDCT:2016-03" display 'Dead'

parameter "Measurement Period" Interval<DateTime>

context Patient

define "Encounter":

["Encounter, Performed": "Emergency Department Visit"]

define "Inpatient Encounter":

["Encounter, Performed": "Encounter Inpatient"] EncounterInpatient

where "LengthInDays"(EncounterInpatient.relevantPeriod)<= 120

and EncounterInpatient.relevantPeriod ends during "Measurement Period"

/\*ToDate takes a given DateTime value and returns a DateTime with the time components "zeroed", and the timezone of the input value, for example, given the DateTime @2012-01-01T06:30:00.0Z, this function will return @2012-01-01T00:00:00.0Z.\*/

define function "ToDate"(Value DateTime ):

DateTime(year from Value, month from Value, day from Value, 0, 0, 0, 0, timezone from Value)

/\*CalendarAgeInDaysAt calculates the calendar age (meaning the age without considering time components) in days.\*/

define function "CalendarAgeInDaysAt"(BirthDateTime DateTime, AsOf DateTime ):

days between ToDate(BirthDateTime)and ToDate(AsOf)

/\*CalendarAgeInDays calculates the calendar age (meaning the age without considering time components) in days as of today\*/

define function "CalendarAgeInDays"(BirthDateTime DateTime ):

CalendarAgeInDaysAt(BirthDateTime, Today())

/\*CalendarAgeInMonthsAt calculates the calendar age (meaning the age without considering time components) in months.\*/

define function "CalendarAgeInMonthsAt"(BirthDateTime DateTime, AsOf DateTime ):

months between ToDate(BirthDateTime)and ToDate(AsOf)

/\*CalendarAgeInMonths calculates the calendar age (meaning the age without considering time components) in months as of today.\*/

define function "CalendarAgeInMonths"(BirthDateTime DateTime ):

CalendarAgeInMonthsAt(BirthDateTime, Today())

/\*CalendarAgeInYearsAt calculates the calendar age (meaning the age without considering time components) in years.\*/

define function "CalendarAgeInYearsAt"(BirthDateTime DateTime, AsOf DateTime ):

years between ToDate(BirthDateTime)and ToDate(AsOf)

/\*CalendarAgeInYears calculates the calendar age (meaning the age without considering time components) in years as of today.\*/

define function "CalendarAgeInYears"(BirthDateTime DateTime ):

CalendarAgeInYearsAt(BirthDateTime, Today())

/\*Hospitalization 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.\*/

define function "Hospitalization"(Encounter "Encounter, Performed" ):

( singleton from ( ["Encounter, Performed": "Emergency Department Visit"] EDVisit

where EDVisit.relevantPeriod ends 1 hour or less on or before start of Encounter.relevantPeriod

) ) X

return if X is null then Encounter.relevantPeriod else Interval[start of X.relevantPeriod, end of Encounter.relevantPeriod]

/\*calculates the difference in calendar days between the start and end of the given interval.\*/

define function "LengthInDays"(Value Interval<DateTime> ):

difference in days between start of Value and end of Value

/\* returns list of all locations within an encounter, including locations for immediately prior ED visit. If data contains many encounters within 1 hour before the start of encounter error will occur indicating the hospitalization cannot be determined\*/

define function "Hospitalization Locations"(Encounter "Encounter, Performed" ):

( singleton from ( ["Encounter, Performed": "Emergency Department Visit"] EDVisit

where EDVisit.relevantPeriod ends 1 hour or less on or before start of Encounter.relevantPeriod

) ) EDEncounter

return if EDEncounter is null then Encounter.facilityLocations else flatten { EDEncounter.facilityLocations, Encounter.facilityLocations }

/\*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\*/

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

LengthInDays("Hospitalization"(Encounter))

/\*Returns admission time for an encounter or for immediately prior emergency department visit. If the data contain many encounters within 1 hour before the start of the encounter an error indicating hospitalization cannot be determined

\*/

define function "Hospital Admission Time"(Encounter "Encounter, Performed" ):

start of "Hospitalization"(Encounter)

/\*Hospital Discharge Time returns the discharge time for an encounter\*/

define function "Hospital Discharge Time"(Encounter "Encounter, Performed" ):

end of Encounter.relevantPeriod

/\*Returns earliest arrival time for an encounter including any prior ED visit. If the data contain multiple encounters within 1 hour before the start of the encounter, an error will occur indicating it cannot be unambiguously determined

\*/

define function "Hospital Arrival Time"(Encounter "Encounter, Performed" ):

start of First(("Hospitalization Locations"(Encounter))HospitalLocation

sort by start of locationPeriod

).locationPeriod

/\*Returns the latest departure time for encounter including any prior ED visit. If the data contain multiple encounters within 1 hour before the start of the encounter, an error will occur indicating it cannot be unambiguously determined\*/

define function "Hospital Departure Time"(Encounter "Encounter, Performed" ):

end of Last(("Hospitalization Locations"(Encounter))HospitalLocation

sort by start of locationPeriod

).locationPeriod

/\*Returns the arrival time in the ED for the encounter. If the data contain multiple encounters within 1 hour before the start of the encounter or multiple ED locations an error will occur indicating it cannot be determined

\*/

define function "Emergency Department Arrival Time"(Encounter "Encounter, Performed" ):

start of ( singleton from ( ( "Hospitalization Locations"(Encounter)) HospitalLocation

where HospitalLocation.code in "Emergency Department Visit"

)

).locationPeriod

/\*First Inpatient Intensive Care Unit returns the first intensive care unit for

the given encounter, without considering any immediately prior emergency

department visit.\*/

define function "First Inpatient Intensive Care Unit"(Encounter "Encounter, Performed" ):

First((Encounter.facilityLocations)HospitalLocation

where HospitalLocation.code in "Intensive Care Unit"

and HospitalLocation.locationPeriod during Encounter.relevantPeriod

sort by start of locationPeriod

)