Article

[Muhammad Waseem](#) · Oct 15, 2021   2m read

 Open Exchange

How to update result reference range and abnormal flag accordingly using HL7

In this article I will demonstrate the following :

- Update ReferencesRange(OBX:7) against ObservationIdentifier(OBX:3.1)[TestCode] from database by using custom utility function
- Update Abnormal Flag(OBX:8) against ObservationIdentifier(OBX:3.1)[TestCode] and ObservationValue(OBX:5)[Result] from database utility function
- Route Message based on Abnormal Flag(OBX:8)

Below is the primary and transformed HL7 2.5 ORU_R01 message:

Step 1: First of all we need to save reference ranges in database, for this I used TestRanges Persistent class:

Table: resultVer.TestRanges ○ Table Info ◉ Fields ○ Maps

Field Name	Datatype	Column #	Required	Unique
ID	%Library.BigInt	1	Yes	Yes
TestCode	%Library.Integer	2	No	No
MinRange	%Library.Double	3	No	No
MaxRange	%Library.Double	4	No	No

Step 2 :We need to create a custom function to get reference range, for this I created GetReferenceRange() ClassMethod function in utility class. Please note this class should be Extends from EnsRule.FunctionSet

```
// Get reference ranges from TestRanges table against test code
Debug this method
ClassMethod GetReferenceRange(TestCode As %String) As %String
{
    &sql(SELECT MinRange, MaxRange into :minRange,:maxRange
    FROM resultVer.TestRanges where TestCode = :TestCode)
    set minRange = $DECIMAL(minRange,3)
    set maxRange = $DECIMAL(maxRange,3)

    Set refRange = minRange _"-"_ maxRange
    //Set refRange = "Test"
    Return refRange
}
```

We need to create an other function to set Abnormal Flag, for this I created SetAbnormalFlag() ClassMethod function in utility class

```
// Get Abnormal flag based on Test Code and Resutl
Debug this method
ClassMethod SetAbnormalFlag(TestCode As %String, Result As %Decimal) As %String
{
    &sql(SELECT MinRange, MaxRange into :minRange,:maxRange
    FROM resultVer.TestRanges where TestCode = :TestCode)
    //Set Flag to L in case of Low Range
    if Result < minRange
    {
        Return "L"
    }
    //Set Flag to H in case of High Range
    elseif Result > maxRange
    {
        Return "H"
    }
    //Return empty strign in case of normal result
    Return ""
}
```

Step 3: Now we will use these functions in Data Transformation (UpdateReferenceRangesDTL). We need to use foreach loop against the repeating segments:

| #	Action	Condition	Property	Value	Key / Transform
1	for each ▼		source.{PIDgrpgrp()}		k1
2	for each ▼		source.{PIDgrpgrp(k1).ORCgrp()}		k2
3	for each ▼		source.{PIDgrpgrp(k1).ORCgrp(k2).OBXgrp()}		k3
4	set		target.{PIDgrpgrp(k1).ORCgrp(k2).OBXgrp(k3)...	##class(resultVer.Utility).GetReferenceRang...	""
5	set		target.{PIDgrpgrp(k1).ORCgrp(k2).OBXgrp(k3)...	##class(resultVer.Utility).SetAbnormalFlag(...	""
6	endeach				
7	endeach				

Action
```
set  ▾
```
Property
```
target.{PIDgrpgrp(k1).ORCgrp(k2).OBXgrp(k3).OBX:ReferencesRange}
```
Property whose value will be set. Double-clicking on a target property in the diagram will place that property in this field.

Value
```
##class(resultVer.Utility).GetReferenceRange(source.{PIDgrpgrp(k1).ORCgrp(k2).OBXgrp(k3).OBX:ObservationIdentifier.Identifier})
```
Value to assign to the property. Double-clicking on a property in the diagram will place that property in this field.

Please note that in order to get Abnormal flag we have to pass OBX:ObservationValue (Result) along with OBX:ObservationIdentifier(TestCode)

Step 4: We will transform the message from "ResultsDTRule" Business rule and send message to "ResultsRule" Business rule again to send message to "ToCriticalResultsFile" operation in case of Abnormal flag is set otherwise if will send to "ToCriticalResultsFile" operation:
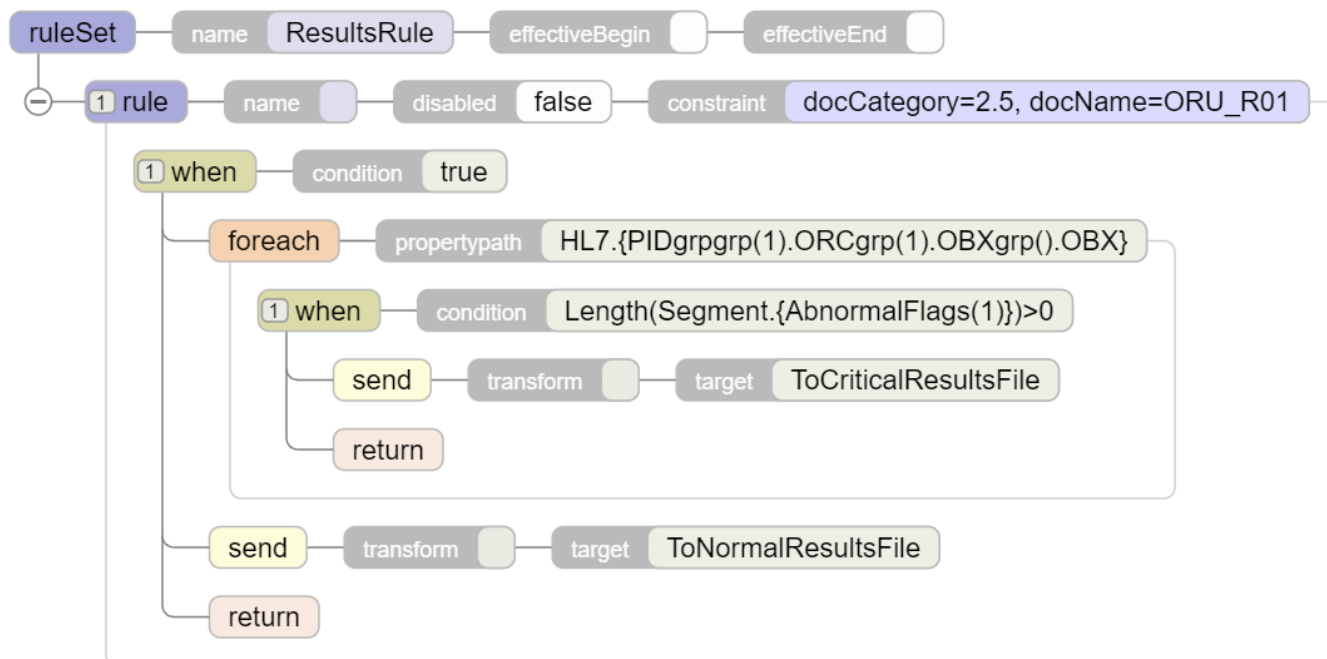
Following Visual Trace Below Visual Trace is for the message successfully send to "ToCriticalResultsFile" Operation
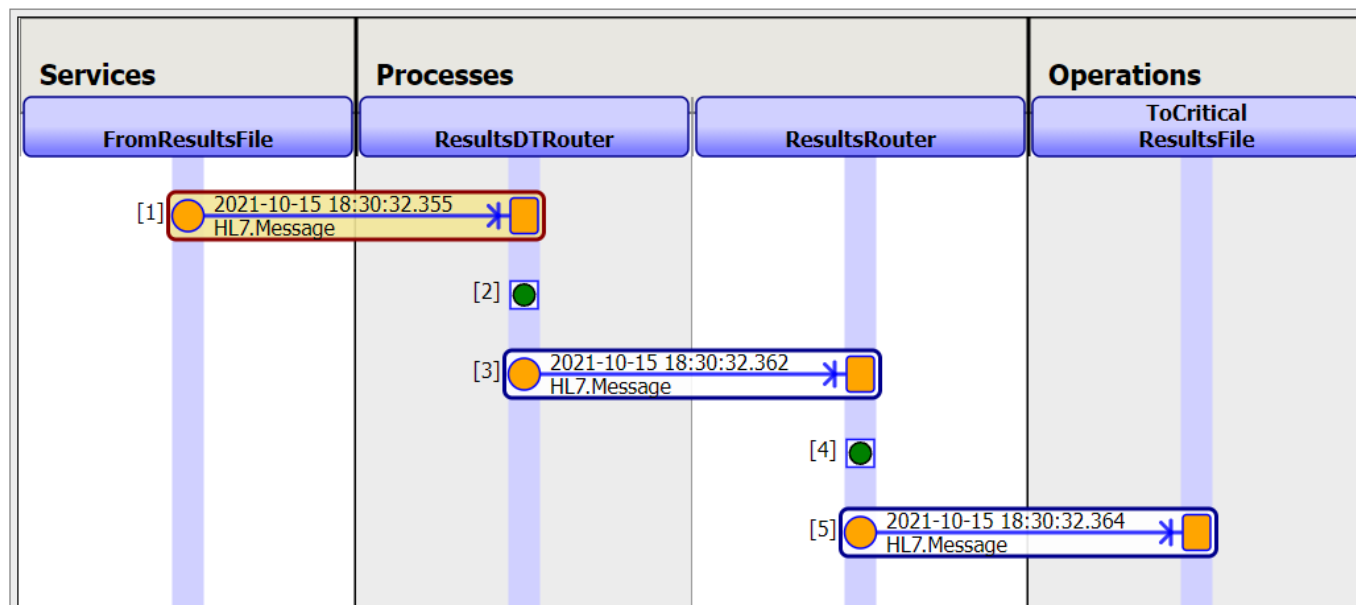
Thanks

#Business Rules #DTL #SQL #Ensemble #InterSystems IRIS for Health
Check the related application on InterSystems Open Exchange