Virtual User Group

Rules 101 with Jeff Piepenburg

Data Innovations

Agenda

I. Intro
II. COVID-19 News Summary
III. Rules 101
IV. Conclusion

Reminders

- Phone lines will be muted
- Chat questions into me at any time
- Must have an 83% participation in order to receive PACE credits
- PACE certificates will be emailed to you Wednesday

Upcoming Webinars & Virtual Events

Virtual User Group
High Availability & Disaster Recovery
Sept. 29, 2020
11:00 AM PT /2:00 PM ET

Virtual User Group
Lab Intelligence
Oct. 6, 2020
11:00 AM PT /2:00 PM ET

Virtual Thought Leadership Event
The Lab's "New Normal": Thought Leadership to Navigate and Embrace Industry Change
November 12, 2020
10:00 AM PT /2:00 PM ET

COVID-19 News Summary

Current trending topics

Data Innovations

COVID-19 News

Current trending topics

Laboratory Testing Volumes

EUAs for pooled COVID-19 testing

Treatment Plans and Test Utilization

Multiplex COVID-19 / Flu A/B

Getting Started with Rules - Rules 101

Getting Started with Instrument Manager Rules

Jeff Piepenburg, MT (ASCP)

Data Innovations

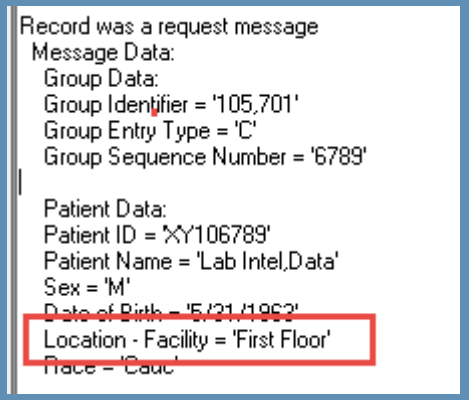
Agenda

I. The Basics
II. Creating Rules
III. Writing Rules

The Basics

Getting Started with Rules

- Rules are configuration-based
  - Allows for rules to be shared across multiple analyzers (one to many)
- Rules can be written against the information in IM
  - Review the Specimen Event Log (SEL) for available data elements
    - Example: Location-based rules



The Basics Continued

- Order of Operation
  - IM does not adhere to PEMDAS (Parenthesis, Exponents, Multiplication, Division, Addition Subtraction)
    - Will perform operations in the order that they appear
      - 3+5*3-2 = 22 in IM rather than 16 if you follow PEMDAS
    - IM uses parentheses to force order of operation
      - 3+(5*3)-2 = 16 in IM
    - Parentheses also used to group “thoughts” in IM together to make sure statements are properly evaluated
- Organization of Rules
  - Rules fire from top to bottom, left to right
  - Rules should be organized in a logical fashion
    - Example: Having a rule fire to perform a calculation on an invalid result

The Basics Continued

Overview of the Rules Screen

- Test/In Validation versus Live
- Red versus Black
- If/Then/Else
  - Initially, lots of “demand” for Else
  - Very rare to actually use Else…very few “it’s this or it’s that” in the Laboratory
- Request or Result Rule?
- Location of Rules
  - Lots of places to write rules
  - The vast majority of rules still written in Incoming Result>Before Message Queued Internally

The Basics Continued

Three basic levels of data elements in Rules

- Patient, Specimen and Test
- General Data Elements as well that do not fit in to those categories
- Patient and Specimen data elements are usually persistent
- Test Data Elements can be persistent but some/most are not

Creating Rules

Creating Rules

- Three ways to write rules
  - Drag and Drop
  - Double-Click
  - Free Text

- Likely to use a combination of those methods
  - Free text is the fastest

Creating Rules Continued

- Some users may find “sketching” out their rules helps
  - Write your rules out on paper first
- Parentheses are only used primarily in the IF statement
  - Used in the THEN statement only if doing a calculation to force order of operation
  - Also used to encase “conditions” that need to be true
- Add versus Set
  - Add adds on to the field you specify and leaves the original contents in place. Set replaces what was there with what you tell it to set
- Add Test versus Order Test
  - Add Test adds that test on to your existing message
  - Used primarily in Result messages to add on a test that was not part of the original result message, such as a calculated value

Creating Rules Continued

Add Test versus Order Test Continued

- Order Test used to order a new test that was not part of the original message
- Most commonly used in Request rules, or to order a reflex test

Writing Rules

Writing Rules

- Using the Drag and Drop Method
  - Write a rule that adds on test “HIL” if a Glucose is ordered
  - First, determine if this a Request or Result rule?
  - Where are we going to write this rule?
  - User Input Value
    - Allows you to define something specific, such as a test code, error code, test result

Writing Rules Continued

- If: ({Test Ordered} “GLU”)
  Then: {Add Test} “HIL”

Writing Rules Continued

- Using the Double-Click Method
  - Write a rule that adds a test comment of “Severe Hemolysis Present – Result may be elevated” to Potassium if Hemolysis is greater than or equal to 3+
  - What type of rule is this, request or result?
  - Where do you write it?
  - What 2 things do I need to check for this rule?
    - Is my hemolysis result >= 3+?
    - Do I have a K result present?
  - Do I care what my Potassium result is?

Writing Rules Continued

- If: ({Test Resulted} “K”) {AND} ({Hemolysis} >= “3+”)
  Then: {Add} {Test Comment(s)} “Severe Hemolysis Present – Result may be elevated” {On Test} “K”

Writing Rules Continued

- Using the Free Text Method
  - If Sodium, Chloride and Bicarbonate results are present, add test AGAP and use the formula = Na – (Cl + HCO3)
  - What type of rule? Where do you write it?
  - What do you need to check?
    - Since we are doing a calculation, we need to check that all values are numeric
    - We do not need to check to make sure all results are present and doing the numeric check takes care of that as well
  - IM is not very smart when it comes to multiple items…you will need to "explain" to it what you are trying to do very literally.
    - You can't just say "If Na and Cl and HCO3 are numeric".
    - You'll need to say "If Na is numeric AND Cl is numeric and HCO3 is numeric"

Writing Rules Continued

- Using the Free Text Method Continued
  - If: (({Result} {On Test} "NA" {Is Numeric}) {AND} ({Result} {On Test} "CL" {Is Numeric}) {AND} ({Result} {On Test} "HCO3"))
    Then: {Add Test} "AGAP" {AND} {Set} {Result} {On Test} "AGAP" = {Result} {On Test} "NA" - ({Result} {On Test} "CL" + {Result} {On Test} "HCO3")

Writing Rules Continued

- Saving Rules
  - Save early, save often!
  - Save to Test/In Validation first, then move to Live
  - Saving rules checks for syntactical errors in the rules, not for “bad” rules
  - Warnings Versus Errors
    - Errors cannot be moved to Live
    - Errors can usually be deciphered, but occasionally can be quite cryptic in nature
    - Warnings are just that…something needs your attention but may be okay
  - Typical Issues
    - Parentheses
    - Is Numeric
    - Value Lists incomplete or not defined

Writing Rules Continued

- Using what you have learned so far, write a rule that changes numeric THC results to positive
  - Test Code = THC
  - Cutoff for Positive = 200 ng/mL
- Extra Credit – Add rules for indeterminate and negative values
  - Cutoff for Negative = 150 ng/mL
- Extra Extra Credit – Convert this rule to use a value list
  - Test Codes = COC, PCP
  - Same cutoffs

Testing and Validation

Data Innovations

Agenda

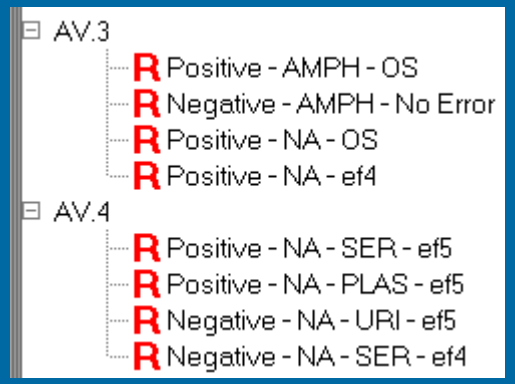
I. Testing Strategy
II. The Test Engine
III. Testing Your Rules

Testing Strategy

- Overall Testing Strategy for IM Rules
  - Test what is needed
    - Look at the data elements the rule is using
  - Test multiple scenarios
    - Test in the affirmative, negative and the absurd
  - Testing of value lists
    - Test items in the beginning, the middle and the end of the value list
    - Testing the logic of the rule, not the value list
  - Do not test rules in a vacuum
    - Rules can be tested individually, but should also be tested as part of the whole rule set

Testing Strategy

- Overall Testing Strategy for IM Rules Continued
  - Spend the time to build a good set of test scenarios and cases
    - Saves time in the long run
  - Build your test cases to match your rule schema



  - Print/save your audit trails
    - Regulators love to see these

Agenda

I. Testing Strategy

II. The Test Engine

III. Testing Your Rules

The Test Engine

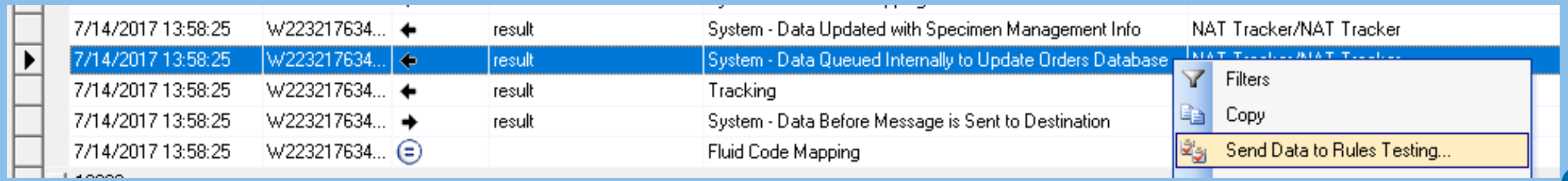
- The Test Engine
  - One of the most powerful tools in IM
  - Virtually eliminates the need to do “wet-testing” with the analyzer
  - Allows for the simulation of any result; allows for easily testing the “edges” of rules
  - Easily create and recreate testing scenarios for regulatory purposes
- Tips and Tricks for the Test Engine
  - Age
    - Items needed to calculate a patient age:
      - DOB
      - Collection Date and Time
      - How old the sample is, not the patient

The Test Engine

- Tips and Tricks for the Test Engine Continued
  - Adding or Removing Fields
    - Not all fields are displayed
    - Use the Field Chooser to add or remove unneeded fields
  - Copying/Duplicating Test Cases
    - Test cases can be copied/pasted using standard Windows commands
    - Test cases can also be copied to other configurations as needed
    - Test cases can be imported and exported as well
  - Testing Scenarios
    - Allows for multiple test cases to be run at once
    - Test cases can build on each other; Something “set” in the first case can be used by the second test case
    - For this to happen, the test cases need to match up
      - Usually requires Patient ID and possibly the request ID being set in the test cases

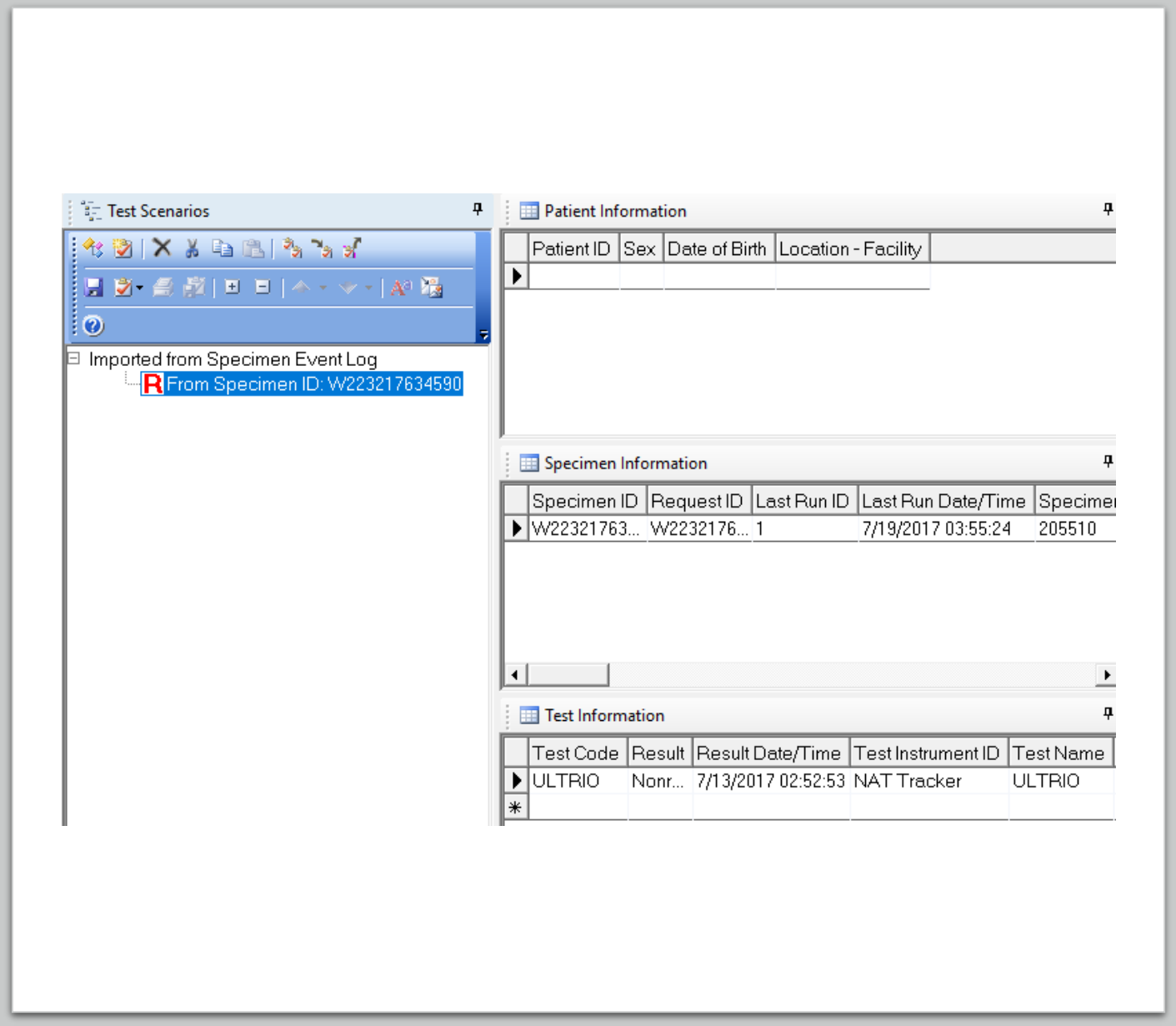
The Test Engine

- Tips and Tricks for the Test Engine Continued
  - Send Data to Rules Testing
    - Allows you to send real-world examples to the testing engine
    - Realistic examples from the analyzer eliminate any testing bias that might exist
    - From the SEL, choose the appropriate entry to send to Rules Testing
    - Most likely you will want to choose the “System - Data Queued Internally to Update Orders Database”
    - This is the entry that has all of the data possible; includes all of the information from the analyzer as well as the information that IM “knows” for the sample
    - Cannot have the Rules configuration screen open to the configuration you are trying to copy to



The Test Engine

- Tips and Tricks for the Test Engine Continued
  - Send Data to Rules Testing Continued

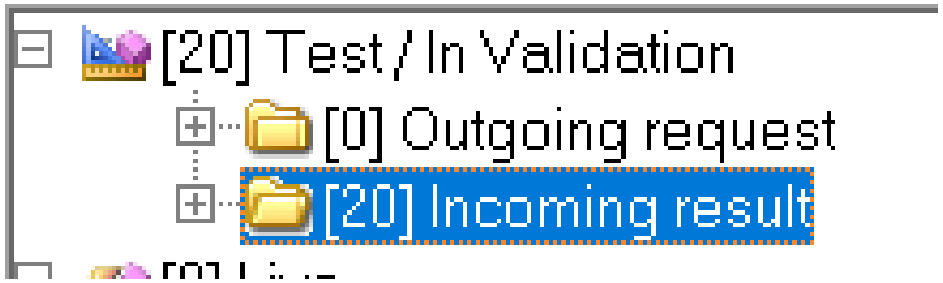


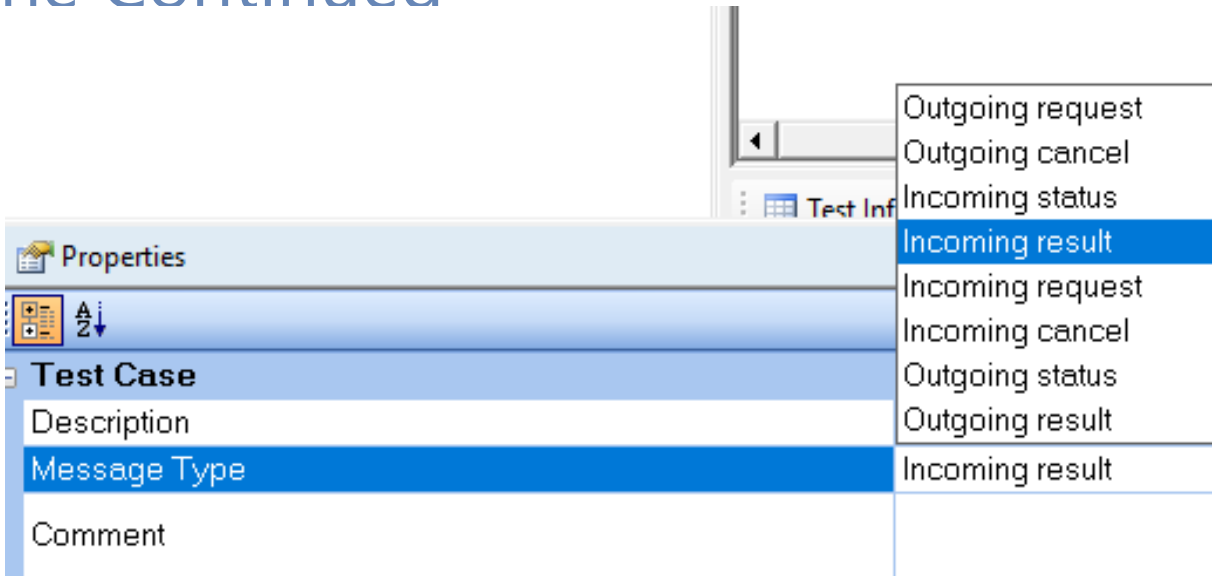
The Test Engine

- Tips and Tricks for the Test Engine Continued
  - Send Data to Test Engine Continued
    - You do not want to use Audit Trails for this purpose
    - Audit trails reflect the state of the message AFTER rules have already fired
  - Executing Test Cases
    - If you right-click, always click on the red letter of the test case
    - Right-clicking on the test case/scenario name usually causes IM to think you are trying to rename the test case
    - Depending on what you are trying to test, you may need to alter the properties of the test case
    - Always match the Message Type of the Test Case to the folder your rule is located in

The Test Engine

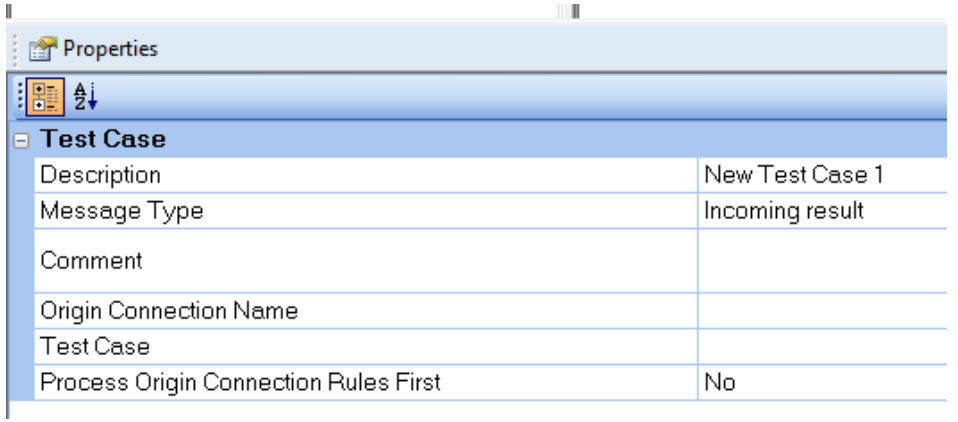
- Tips and Tricks for the Test Engine Continued
  - Executing Test Cases Continued





The Test Engine

- Tips and Tricks for the Test Engine Continued
  - SM rules
    - "Difficult" to test
    - Requires defining the connections used in the properties of the test case



The Test Engine

- Tips and Tricks for the Test Engine Continued
  - Testing complicated rules
    - May require that you break down the rule in to smaller pieces first to test
    - Add back in additional pieces once you have the smaller piece(s) working
    - Cut/Copy/Paste
  - Parentheses
    - Should always have an even number of parentheses
    - For every open parentheses, you need a close parentheses
    - If you get a parentheses error, add/remove 1 at a time
    - Best method I have found for “counting” parentheses?

Agenda

I. Testing Strategy
II. The Test Engine
III. Testing Your Rules

Testing Your Rules

- Let's go back and test the rules we wrote earlier
- The first rule was this:
  - If: {Test Ordered} "GLU"
    Then: {Add Test} "HIL"
  - What type of test case is this? Request or Result?
  - What data elements do we need in our test case?
    - DOB?
    - Collection Date/Time?
    - Patient ID?

Testing Your Rules

- The second rule was:
  - If: ({Test Resulted} “K”) {AND} ({Hemolysis >= “3+”)
    Then: {Add} {Test Comment(s)} “Severe Hemolysis Present – Result may be elevated” {On Test} “K”
  - What type of test case is this? Request or Result?
  - What data elements do we need in our test case?

Testing Your Rules

- The third rule was:
  - If: (({Result} {On Test} "NA" {Is Numeric}) {AND} ({Result} {On Test} "CL" {Is Numeric}) {AND} ({Result} {On Test} "HCO3")) Then: {Add Test} "AGAP" {AND} {Set} {Result} {On Test} "AGAP" = {Result} {On Test} "NA" - ({Result} {On Test} "CL" + {Result} {On Test} "HCO3")
  - What type of test case is this? Request or Result?
  - What data elements do we need in our test case?

Testing Your Rules

- Testing the Delta Check What data elements are needed?
  - Patient ID
  - DOB
  - Collection D/T
  - Test Code
  - Test Result
  - “Previous” fields will be populated once we run the test cases
- We will need to build two cases
  - 1st Test Case will pass through without any issues
  - 2nd Test Case will be the one that triggers your delta check rule
    - Second test will have the same patient identifiers as the first test case, but a different collection date and time and specimen ID than the first test case
  - Both Test Cases will be under the same scenario and executed together

Common Issues and Tips and Tricks

Data Innovations

Common Issues and Tips and Tricks

- Age
  - Items needed to calculate a patient age:
    - DOB
    - Collection Date and Time
  - IM uses days to calculate age
    - Fractions of days can be used. 0.5 days = 12 hours
    - To keep things simple, use Age in Days for all of your age-based rules
    - Months and Years data elements also available
      - These data elements do not recognize fractions. 0.5 years = 0 years
- Is Numeric
  - Lots of issues with calculations using non-numeric values
    - IM will often evaluate text as a zero

Common Issues and Tips and Tricks Continued

- “Persistent” data elements
  - There is often a need to remember information from one run to another of a sample (Criticals, repeats, “2 of 3” testing)
  - Prior versions of IM, this was difficult to do
    - Fields like Previous Test Dilution or Previous Test Error were unique in IM. Test-level data elements that were persistent from run to run
    - Only two available!
  - Several different options to store and recall information now
    - Previous Run data elements
    - Specimen User Fields
    - Test User Persistent Fields

Common Issues and Tips and Tricks Continued

- Any/That
  - "You can have an Any without a That, but you can't have a That without an Any"
  - If you want to use a That, you have to have used an Any in the IF statement
  - Only one Any per IF statement; Can have multiple That in your IF and THEN statement
- Any/That Bad Example:

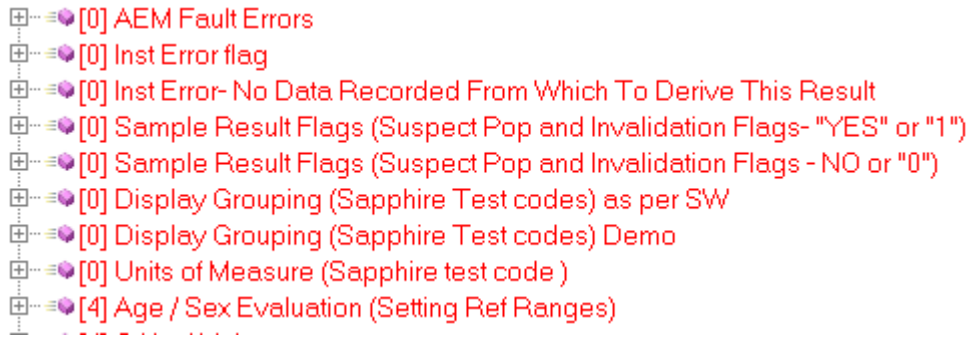
  - De AND} ({Re Else: - 'On Any Test' can not be used multiple times in an If statement

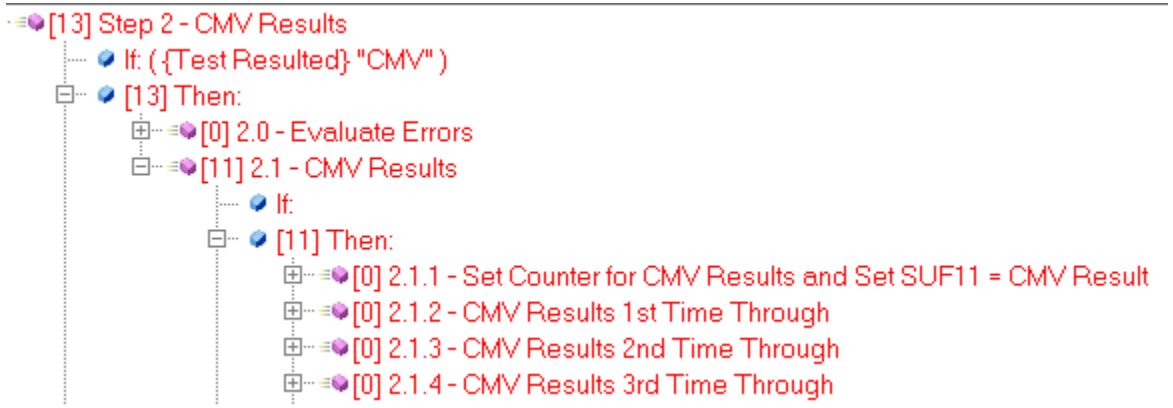
Common Issues and Tips and Tricks Continued

- [0] Any/That Example 1
  - If: ({Result} {On Any Test} > "100")
  - [0] Then: {Hold That Test for Verification}
  - [0] Else:
- [0] Any/That Example 2
  - If: (({Result} {On Any Test} > "100") {AND} ({Result} {On Test} "GLU" {NOT} = ""))
  - [0] Then: {Hold Test for Verification} "GLU"
  - [0] Else:
- [0] Any/That Example 3
  - If: (({Result} {On Any Test} {NOT} = "") {AND} ({Result} {On That Test} {NOT} = "Positive"))
  - [0] Then: {Hold That Test for Verification}
  - [0] Else:

Common Issues and Tips and Tricks Continued

- Organization of Rules
  - Rules can be used for organization purposes





  - "Blank" rules can be read as IF: Always
  - "Blank" rules can also be used to store value lists for the child rules underneath the parent rule

Common Issues and Tips and Tricks Continued

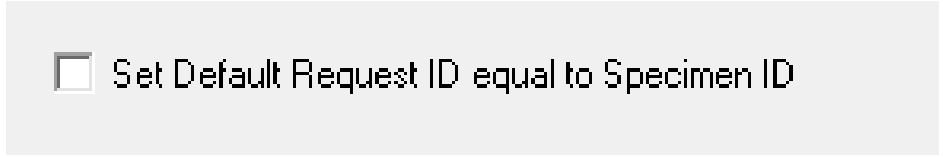
- SM rules
  - Limited functionality - Primarily a left-over function from earlier releases of IM
  - Used for comparing/evaluating results that came from different analyzers or at different times from the same analyzer
  - Requires that you specify the instrument connection(s) in the rule itself
  - Cannot change or edit a result, only add new tests and results
  - Almost all functionality can be done with regular rules now

Common Issues and Tips and Tricks Continued

- Request ID
  - This is an internal ID used by to keep samples unique
  - Typically consists of the Specimen ID + the year and the Julian day (1234-2018120)
  - If you ever see the same sample ID on different rows on the workspace, this is why
  - Created to prevent overwriting information on samples where they do not use unique Specimen IDs
  - If your LIS uses unique IDs (never repeats), Request ID can be manipulated by rules to help with this issue

Common Issues and Tips and Tricks Continued

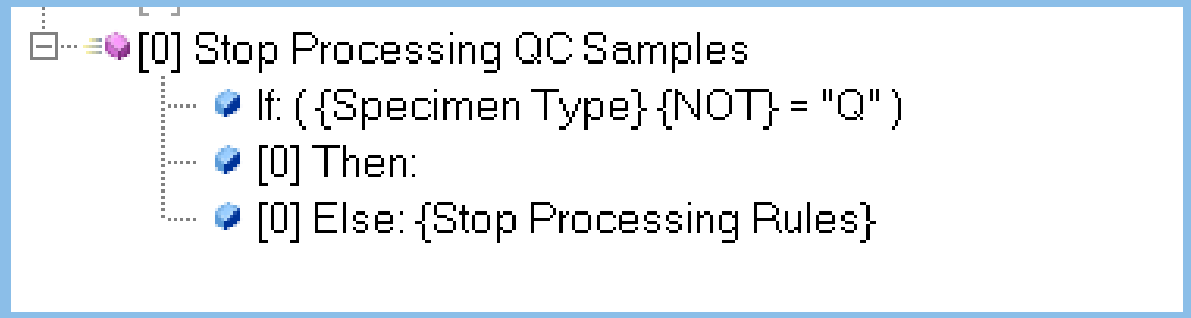
- Request ID Continued
  - As of 8.15.10, setting Request ID = Specimen ID is a setting within the application
    - Configuration>Specimen Management Configuration>Specimen ID Algorithm



  - Will not work for unsolicited results; needs an order for this feature to work, otherwise rules will still be needed to set the Request ID

Common Issues and Tips and Tricks Continued

- Else
  - As previously mentioned, there are not many use cases for Else
  - Example:



Questions?

- Thank you for your time!

Upcoming Webinars & Virtual Events

Virtual User Group
High Availability & Disaster Recovery
Sept. 29, 2020
11:00 AM PT /2:00 PM ET

Virtual User Group
Lab Intelligence
Oct. 6, 2020
11:00 AM PT /2:00 PM ET

Virtual Thought Leadership Event
The Lab's "New Normal": Thought Leadership to Navigate and Embrace Industry Change
Sept. 29, 2020
10:00 AM PT /2:00 PM ET