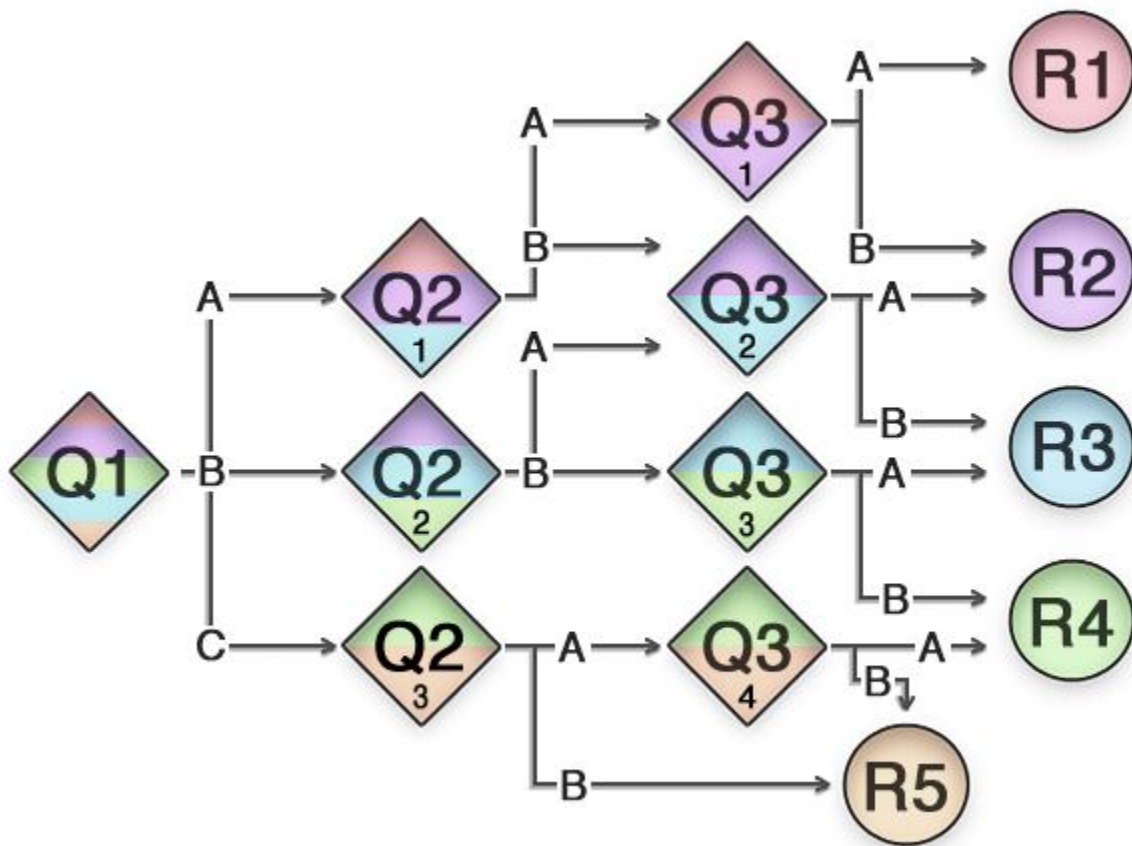


DISCERN EXPERT

Using Cerner Millennium Rules in Your Lab



Discern Expert Rule Examples For Lab	4
Overview.....	4
Solution Terms.....	5
Laboratory Events	6
General Evoke Statements	6
PathNet Specific Statements	6
Key Applications.....	6
Navigating Expert System Tool.....	7
Event Configurations.....	7
Event Bindings	7
Server Configuration	7
Server Maintenance	8
Request Processing	8
Report Auditing	8
Purging/ Archiving Tables	9
Notify Configuration.....	9
Notify Tools.....	9
Exercise – Launch the Expert System Tool and familiarize yourself with the Server Maintenance, Report Auditing and Notify Tools options	9
Navigating Discern Development Environment	9
Maintenance Section	10
Library Section	10
Knowledge Section	10
Configuring A Basic Discern Expert Rule	11
Overview	11
Starting A New Rule	12
Maintenance Tab.....	12
Exercise – Start a new rule, fill in the maintenance tab and save your progress	12
Library Tab	12
Exercise – Enter applicable information on the library tab	13
Knowledge Tab	13

Exercise – Complete the evoke, logic and action sections.....	16
Troubleshooting a Basic Discern Expert Rule	16
Testing Rules	16
Exercise – Test your rule and view the results in the EKS Monitor.....	17
Configuring a More Complex Discern Expert Rule	18
Parenthesis for ORs	18
Action Groups	18
Exercise – Adapt your current rule to check for autoverified results	19
Custom CCL Logic Statements	19
Exercise – Incorporate this CCL into a COLLECTION_EVENT rule.....	20
Example of Complex Grouping and Parenthesis	21

Notes

DISCERN EXPERT RULE EXAMPLES FOR LAB

Discern Expert provides an open avenue to managing the workflows throughout Millennium. Deciding where to start and how to attack a desired outcome can be challenging. Labpedia has example Discern Explorer, Discern Analytics, and Discern Expert templates that are updated by requests from the lab communities on uCern. You can access uCern at <http://www.ucern.com> to create an account. Once you have a sign on, follow the links for Laboratory Professionals to join the groups relating to Lab. Labpedia is a link embedded in the home pages of all PathNet Millennium groups, and is also at this address: <https://wiki.ucern.com/display/main/Labpedia>.

MAINTENANCE SECTION	
TITLE:	ADDON_CREATE_NEW_CONTAINER
FILE NAME:	ADDON_CREATE_NEW_CONTAINER
DATE:	3/14/2011
DURATION:	3/14/2011 TO 3/14/15
AUTHORS:	
VERSION:	001.000
INSTITUTION:	
SPECIALIST:	
VALIDATION:	PRODUCTION

LIBRARY SECTION

PURPOSE:
EXPLANATION:
KEY WORDS:
CITATIONS:

KNOWLEDGE

PRIORITY of Module: 50

EVOKE SECTION

Invoke on [COLLECTION_EVENT](#) where,
EKS_COLLECTION
E1 Collection event for procedure [Lytes](#)

LOGIC SECTION
EKS_COLL_STATUS
L1 [Lytes](#) collection status is [In-Lab](#)

ACTION SECTION
EKS_SETUP_ADD_ORDERS_A
A1 Setup [CreateContainer](#) to add [Urinalysis](#) with [OPT_COMMENT_TYPE](#) of [OPT_COMMENT](#) for [the same accession as Refer to L1](#)
AND
EKS_SET_ORDER_DETAIL_A
A2 SET [Container Id](#), equal to [Urinalysi->Urine>Sterile_Cup_100.000000_mL](#) or inherit it from [Refer to L1](#)
orders setup by [CreateContainer](#)
AND
EKS_CALL_ORDER_SERVER_A
A3 Call order server with [OPT_COMMENT_TYPE](#) of [OPT_COMMENT](#)

While these rules cannot be downloaded and automatically installed into your domain, the steps to recreate them are detailed in full on Labpedia. Instructions for becoming an active participant in the lab communities are all available through uCern.

OVERVIEW

Discern Expert is an event-driven, rule-based application. It can be successfully employed by health professionals to perform cost containment, quality assurance,

resource utilization, interdepartmental communication, and risk management. An organization or department can automate protocol management with Discern Expert.

Discern Expert provides the ability to perform both asynchronous and synchronous processing. With asynchronous processing, the task is performed in the background without interaction from the user. Synchronous processing allows the interaction between the user and the system to occur in real time, with the ability to require response to some of the system requests in order to proceed to the next screen. With one application license, the client receives both asynchronous and synchronous capabilities.

The main activity of Discern Expert is to monitor events of interest within the clients licensed Cerner Health Network Architecture (HNA) applications and take action(s) depending upon user-defined criteria. A small sample of events that can invoke Discern Expert includes admissions, transfers, discharges, orders, and results. Events can be captured either before or after they are processed. Discern Expert also can trigger on events that occur in non-Cerner systems that are interfaced to Cerner HNA systems using a foreign system interface. Discern Expert provides the ability to act on order events, either before or after they are processed, potentially preventing complications or the performance of unnecessary tasks. This is a key advantage of Discern Expert over many other types of knowledge systems.

Discern Expert users create rules with IF-THEN logic that enables them to apply their own criteria to a set of events and then take actions compatible with their goals.

For example:

IF:

An order for a Blood Product Request is initiated AND

The patient has not had a Type and Screen ordered within 3 day,

THEN:

Send an alert to the person placing the order that a current blood sample is not in the laboratory, giving the option to add the Type and Screen automatically.

Discern Expert employs an advanced, graphical user interface-based (GUI) editor that allows for the easy creation of syntactically correct rules. Online Help is available to simplify the process further, and CBTs and an advanced course are available. Discern Expert achieves maximal performance through advanced software design techniques. It is at the leading edge of advanced clinical decision support tools.

SOLUTION TERMS

A glossary is included in every applications Help file, gives you access to terms across all Cerner Millennium applications. The following list identifies the terms pertaining specifically to Discern Analytics:

Notes

Action – The conclusion of the rule, or the event that the system should create.

Evoke – Cerner system event that can initiate a rule. This is also referred to as the triggering event.

Logic – The grouping of if statements that query the Cerner database for true/ false responses.

Module Priority – A field which allows a numeric value between 1 and 100 to be set to determine the sequencing of the rules within a single event.

Template – The term used to describe the Cerner supplied evoke, logic, or action statements.

Validation – A field within the rule that allows the user to set if the rule is in Testing, Research, Production or Expired states.

LABORATORY EVENTS

There are a few dozen different types of events that will invoke the Discern Expert processing and trigger a rule. Several of the generic events will trigger for use in PathNet, while only a few are PathNet specific.

GENERAL EVOKE STATEMENTS

- | | |
|---------------------|-----------------|
| • Add to Scratchpad | ADDTOSCRATCHPAD |
| • Clinical Event | CLINICAL_EVENT |
| • Order Event | ORDER_EVENT |
| • Patient Event | PATIENT_EVENT |

PATHNET SPECIFIC STATEMENTS

- | | |
|-----------------------|---------------------|
| • AP Order Event | AP_ORDER_EVENT |
| • BB Result Event | BB_RESULT_EVENT |
| • Collection Event | COLLECTION_EVENT |
| • CSM Result Event | RESULT_EVENT_CSM |
| • MB Result Event | MB_RESULT_EVENT |
| • MB Scripted Workup | MB_SCRIPTED_WORKUPS |
| • MB Susceptibility | MB_SUSCEPTIBILITY |
| • Notification Ranges | PNTFLEXREFRANGE |
| • Result Event | RESULT_EVENT |

Note: Add to Scratchpad support is slated to be added to Department Order Entry for release in 2014.

KEY APPLICATIONS

The Discern Analytics suite consists of four separate applications:

- **EKM Editor** – EKMEditor.exe, is used to build Discern Rules prior to 2012. This application allows the user to graphically build a Discern Rule. A CCL object is created when the rule is saved.
- **Discern Developer** – DiscernDev.exe, replaces the EKMEditor in 2012 code. DiscernDev will work on 2010 code, but the EKMEditor was officially sunset on the 2012 code base. In addition to all of the functionality of the EKMEditor, DiscernDev adds new support for copy and paste actions within a rule or between rules, and additional support around grouping of logic statements.
- **Discern Notify** – DiscernNotify.exe is used to receive notification messages to display to a user.
- **Expert System Tool** – ExpertSysTool.exe, this tool allows configuration of events, requests, and servers needed to run for Discern Expert. This tool also allows purging and archiving of Discern Expert modules.

NAVIGATING EXPERT SYSTEM TOOL

EVENT CONFIGURATIONS

The event configurations table lists some standard setup for different evoking events for Discern Expert. In general, it is unnecessary to make any changes to the data in this table. The Event Name column correlates to the events seen in the evoke sections of Discern Expert rules. The priority defines which server class will handle the event. For example, if the RESULT_EVENT is tied to the priority of 60, and the server class of the EKS_ASYNC_02 is tied the range of 34 to 66, then a RESULT_EVENT rule will run through the ESK_ASYNC_02 server class.

EVENT BINDINGS

The event bindings table lists all of these events and the scripts that they are bound to within Cerner. This table, like the Event Configuration table, should typically not need to be changed. However, it provides some useful information when troubleshooting, or writing a rule that needs access to the request structured that triggered the rule. For example, when using a RESULT_EVENT evoke statement, it is known that the request structure sent will be from request 250074 (GLB_UPD_LAB_RESULTS). If that request number is entered into Discern Visual Developer, the information on who performed a result, can be found at `request->orders[]->assays[].perform_personnel_id`. That could then be used in a CCL to check the performing tech of a result that triggered a Discern Expert rule.

SERVER CONFIGURATION

The server configuration table lists the servers that operate for Discern Expert. This table can be used to change the server log level, if privileges exist for the user accessing the tool. To change the log level, select the appropriate server for the event priority in question, and select then Modify button. The Logging Level is then shown in a drop

Notes

down box. A log level of 4 is only used when troubleshooting a rule. In a production environment, the log levels are typically set to 0.

Modify Server Class

Server Class: EKS_ASYNCH_02

Server Type:
☒ Asynchronous
☐ Synchronous

Begin Priority: 34
End Priority: 66

Cache Max Sizes:
Event Cache: 1000
Evoke Cache: 1000
Module Cache: 1000

Module Cache Priority:
Low Priority: 50
High Priority: 100

Server Control Panel Information:
Description: Discern Expert Server (EKS_ASYNCH_02)
Driver Path: cer_exe/srv_drvr
EXE Path: cer_exe/eks_srvexpert
Database: ORACLE
Database user: v500

Logging Level: Debug Mode (4) ☒ Enable Full Audit
Number of Instances: 1 ☒ Restart after Crash

OK Cancel

SERVER MAINTENANCE

The server maintenance tab allows the user, if privileges are granted, to cycle the necessary servers for Discern Expert. The easy way to cycle servers is to select the Cycle All button. Knowing which server to cycle however, can save time when waiting for all of the servers to cycle. When initially saving a rule, the Discern Expert Server, Discern Event Server, and the appropriate server for the evoke event being used, all need to be cycled. For example, on a RESULT_EVENT rule, servers 175, 176, and 151 would need to be cycled. When modifying a rule, only the server associated with the event needs to be cycled. In the same example, when modifying a RESULT_EVENT rule, server 151 would need to be cycled for the changes to be applied to the system.

This same theory applies across multiple nodes. The Expert System Tool can change nodes by selecting the Change Host button in this view.

REQUEST PROCESSING

This table allows the user to control which requests are routed to each expert server. The function would be used after adding or modifying server classes or event bindings.

REPORT AUDITING

The report auditing selection lists several reports that can be run against the environment.

EKS_MONITOR – This report lets you trace the execution of a Discern Expert rule. Each logic line that is evaluated is displayed with troubleshooting information and a final true or false conclusion. This report can also be executed from Discern Dev.

EKS_RPT_TEMPLATES – This report displays a listing of all Discern Expert templates in the current environment.

EKS_RPT_MODULES – This report displays a listing of all Discern Expert rules built in the current environment.

EKS_RPT_EXPIRED_EKMS – This report displays a listing of all Discern Expert rules that are due to expire within a specified number of days.

EKS_SYSTEM_AUDIT – Displays performance information across a specified timeframe.

EKS_RPT_EKM_EVENTS – This report displays a listing of all EKM modules and events per module.

EKS_PERF_AUDIT – This report displays a performance query for all rules that have been recently fired.

PURGING/ ARCHIVING TABLES

Report auditing allows the removal of unnecessary versions of Discern Expert rules, or the completed removal of outdated rules that should not have been built.

NOTIFY CONFIGURATION

The Notify Configuration Settings dialog box is divided into two sections, and allows administrators and users to enable and disable Discern Notify for their perspective areas of responsibility. The Domain Level Settings allow administrators to enable and disable Discern Notify for the current domain and to set the system defaults for how Discern Notify will function for all users. The Machine Level Settings allow users to enable and disable Discern Notify for their current PC or workstation, or globally on Windows Terminal Server (Citrix). User level settings can be set within the Discern Notify application by each user, and takes precedence over the default Domain Level Settings set by the administrator.

NOTIFY TOOLS

This option displays a listing of reports and tests that can be run with Discern Notify. For example, a test message can be sent from user to user with the Test Discern Notify option.

EXERCISE – LAUNCH THE EXPERT SYSTEM TOOL AND FAMILIARIZE YOURSELF WITH THE SERVER MAINTENANCE, REPORT AUDITING AND NOTIFY TOOLS OPTIONS

NAVIGATING DISCERN DEVELOPMENT ENVIRONMENT

The EKM Editor (EKMEditor.exe) or Discern Development (DiscernDev.exe) both reflect the same structure of a rule. Any rule built in the EKM Editor will display in Discern Dev. However, some of the functionality in Discern Dev is not supported in the EKM Editor, and a module built in Discern Dev, will not open if that functionality is used. That will be discussed further in this section. Both tools allow the definition of the following elements within a rule.

Notes

MAINTENANCE SECTION

The maintenance section provides the ability to enter some overriding information for the rule.

- **Title** – An optional field that sets a 30 character title for the rule.
- **Duration** – A required field that determines a beginning and ending date for the rule. The duration dates must be inclusive of the dates you want the rule to fire. When a module has a validation of Production, Testing or Research, but the current date is outside the duration dates, the duration overrides the validation status and the rule will not function. When a module is in an expired status, the duration dates are ignored and the rule will not function.
- **Validation** – A required field with a drop down box to choose from four values. A status of PRODUCTION means the rule will evoke, perform the logic, and take an action. A status of TESTING or RESEARCH means the rule will evoke, perform the logic, but the action will not be initiated. This is helpful in testing a rule in a live environment to validate the outcome that would have occurred. A status of EXPIRED means the rule will not evoke at all.
- **Authors** – An optional field to enter the person creating the rule.
- **Institution** – An option field to enter the institution the rule is being used for.
- **File Name** – The file name associated with the rule. When you first save the rule you will be prompted for a file name. This field is required and set to a maximum of 25 characters. An underscore, “_”, is typically used to group common rules.
- **Update Date** – The date, time and user ID of the last save of the rule.
- **Version** – An auto incrementing number that specifies how many times the rule has been saved.
- **Specialist** – A freetext optional field to enter the name of the specialist who contributed to the rule.
- **Module Priority** – A field that allows a numeric value between 1 and 100 to be set to determine the sequencing of the rules within a single event.

LIBRARY SECTION

All fields in the library section are optional, but entering information in these fields will allow the rules to be searched and queried against. It also provides a space to enter the specifications for how the rule should be used. All of the following fields are freetext:

- Purpose
- Explanation
- Key Words
- Citations
- Query
- Impact

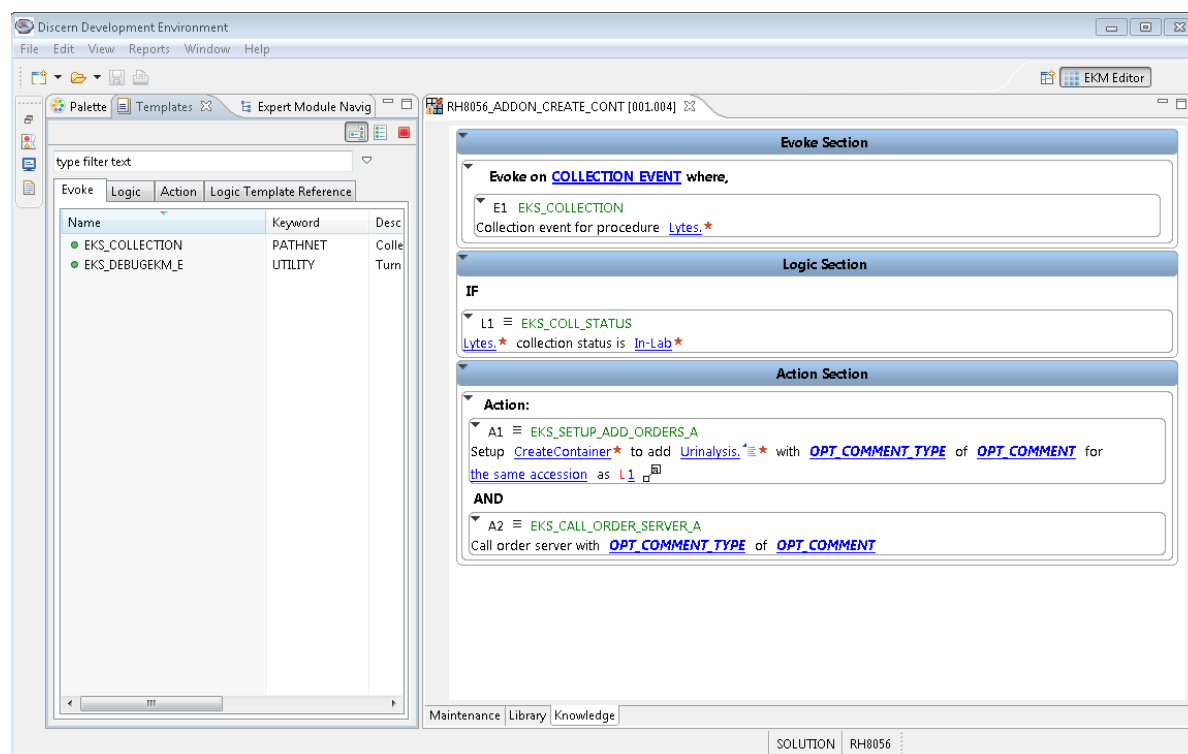
KNOWLEDGE SECTION

- **Evoke Section** – The evoke section determines what events or actions in Cerner will cause this rule to be evaluated. The evoke section consists of both the evoke event and the evoke statements.
- **Logic Section** – The logic section consists of templates that evaluate short statements, resulting in a true or false conclusion. The range of logic templates available flexes on the evoke event chosen.
- **Action Section** – The action section determines what the rule should do if the proper set of logic statements have evaluated as expected. The action templates flex based on the evoke event chosen.

CONFIGURING A BASIC DISCERN EXPERT RULE

OVERVIEW


Upon first opening Discern Dev, the default perspective will appear. The perspective is defined as the configuration of the smaller windows, or views, and where they appear on the screen. Depending on the resolution of the monitor in use, you may choose to setup the perspective in a different manner. The views can be turned on or off by selecting the X in the upper corner, minimized by selecting the horizontal line, or dragged and dropped to lie over another section as tabs. Once you have established the perspective to your liking, it can be saved in the Window > Save Perspective As... option. Below is an example of a perspective that maximizes the space for the rule itself, while still showing the template list:

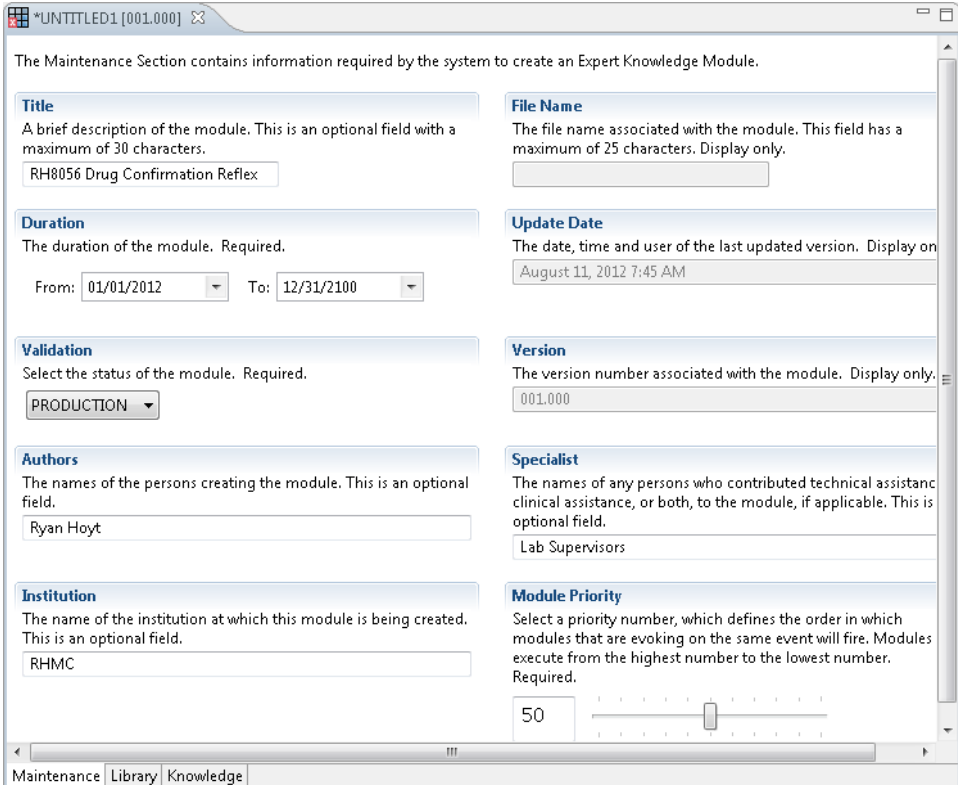


Notes

STARTING A NEW RULE

MAINTENANCE TAB

To create a new rule, select the new button () or File > New > Expert Knowledge Module. This will then default the cursor to the Title box on the Maintenance Tab. For starters, our first rule will be a very simple reflex scenario. A presumptive positive Drug Screen, will trigger a new order to be placed, on the same accession number, for a Drug Confirmation. In the Title field, create a title that is meaningful to you. This is a free text field. Set the Duration to have a wider timeframe than one day. Set the validation status to PRODUCTION. Enter applicable information in the Authors, Institution, and Specialist fields. For now, leave the priority at the default level of 50.



The Maintenance Section contains information required by the system to create an Expert Knowledge Module.

Title
A brief description of the module. This is an optional field with a maximum of 30 characters.
RH8056 Drug Confirmation Reflex

Duration
The duration of the module. Required.
From: 01/01/2012 To: 12/31/2100

Validation
Select the status of the module. Required.
PRODUCTION

File Name
The file name associated with the module. This field has a maximum of 25 characters. Display only.

Update Date
The date, time and user of the last updated version. Display only.
August 11, 2012 7:45 AM

Version
The version number associated with the module. Display only.
001.000

Authors
The names of the persons creating the module. This is an optional field.
Ryan Hoyt

Specialist
The names of any persons who contributed technical assistance, clinical assistance, or both, to the module, if applicable. This is optional field.
Lab Supervisors

Institution
The name of the institution at which this module is being created. This is an optional field.
RHMC

Module Priority
Select a priority number, which defines the order in which modules that are evoking on the same event will fire. Modules execute from the highest number to the lowest number. Required.
50

Maintenance Library Knowledge

At this point, the rule has not been saved, and when attempting to save, multiple warning messages appear. However, if attempting to save, it will save the work that has been done so far, but the rule will not be able to evoke because no information has been entered on the Knowledge tab.

EXERCISE – START A NEW RULE, FILL IN THE MAINTENANCE TAB AND SAVE YOUR PROGRESS

LIBRARY TAB

All of the information on this tab is optional, however, it can be quite useful when troubleshooting a rule written by someone else. Here, information can be entered about what the rule is supposed to do in plain English. Without having to decipher

different logic statements, someone could quickly review the information on this tab and have a working knowledge of the purpose for the rule. It is standard practice to enter as much information as possible on this tab for this reason.

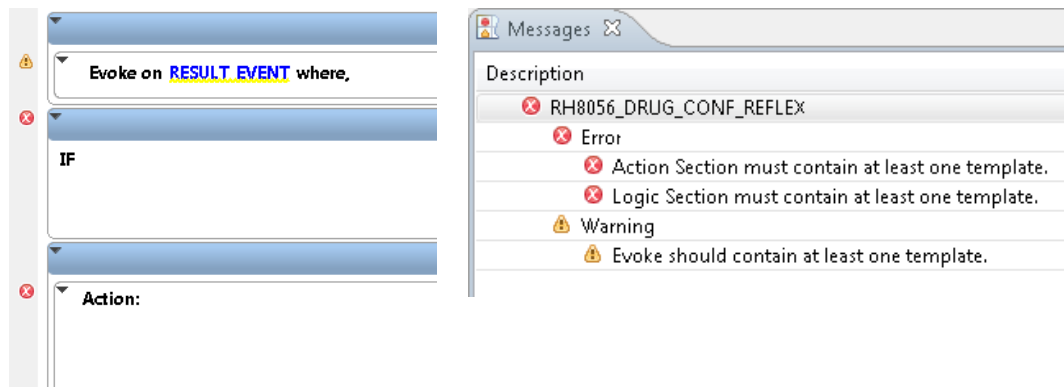
EXERCISE – ENTER APPLICABLE INFORMATION ON THE LIBRARY TAB

KNOWLEDGE TAB

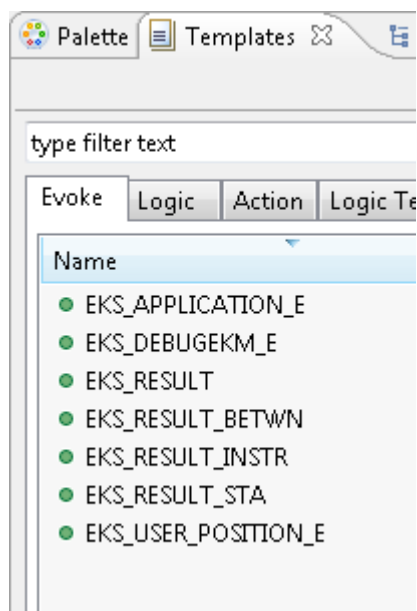
The knowledge tab contains the bulk of the information about the rule and how it functions. For this particular example, the rule should fire when a result for the Drug Screen orderable is completed. There are a few different evoke statements that could be used for this purpose. The `CLINICAL_EVENT`, `ORDER_EVENT` and `RESULT_EVENT` evoke statements all have the ability to check to see if an order is completed. However, part of the rule also specifies the action must be taken on a Presumptive Positive result in that completed order. This eliminates the `ORDER_EVENT` evoke. The `ORDER_EVENT` does not inherently have the ability to check the results within that order. That leaves the `CLINICAL_EVENT` and the `RESULT_EVENT`. Either of these would work for this rule, however, the `RESULT_EVENT` has specific functions in PathNet that will come in handy later on this rule. The understanding of what evoke statements to use is best learned through setting up examples. Not every logic template will work with every evoke statement, so knowing which options are available for each evoke will help.

EVOKE TEMPLATES

Once an evoke statement is chosen, the warning lights on the left-hand side of the Editor view will turn from a red x to a yellow triangle. The Messages view will explain what those errors are:



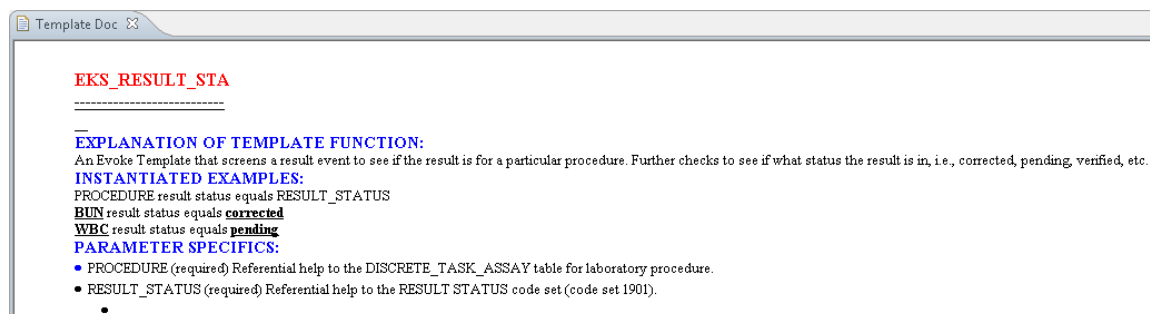
Notes



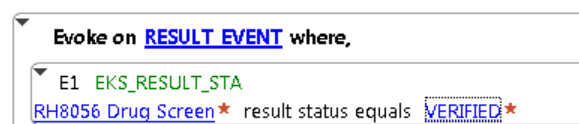
In this case, the warning message just indicates that the rest of the evoke has not yet been completed. Select the Evoke Section within the editor, and the Evoke Section will be highlighted in a blue color. The Templates view will also flip to the Evoke tab, displaying applicable templates that can be used within this section. In the upper right hand corner of the Templates view are three buttons that determine which templates, whether evoke, logic, or action, show. The first button filters the list of templates that are applicable for the selected evoke. The second button shows all possible templates. The third button shows templates that are no longer recommended.



For the rule example, the EKS_RESULT_STA template will allow the functions that are required. To see more information about that template, select it from the list, right-click and choose Template Doc. That will open the Template Doc view with information about the template and, typically, a few real life examples.



To add the EKS_RESULT_STA template to the rule, double click the template from the Templates view. When the template is added to the rule, the parameters are shown as blue hyperlinks. Some parameters are fill in the blank (numeric ranges), drop down boxes (IS/ IS NOT) or lookup boxes (both of the parameters in this template). Select the **PROCEDURE** parameter and click the HELP button to open a new window to find the appropriate assay. In this example, we want to evoke on a Verified Drug Screen. We will discuss later the combination of Verified and Autoverified events. Notice that to help navigate the rule, each template that we insert gets a unique identifier for that rule. This template is given the identifier of E1 because it is the first template within the Evoke Section.



LOGIC TEMPLATES

Once we have started the rule with the evoke section, selecting within the Logic Section, will automatically change the tab in the Templates window to show the logic templates for that evoke statement. Each template is indicated with a green dot, a yellow triangle, or a red square. These indicators note the efficiency of a particular template. While not recommended templates will work, there is often a better alternative available to the user building the rule.

Templates in the logic section are often linked together, to pull a result from one template into consideration in another, or to tie an order that was found on a previous encounter to a template that looks specifically at order details. With that in mind, it is generally good practice to restate in the logic section, what was already stated in the evoke section. In this example, that means we will want to include a template that checks that the result in question was a verified drug screen. For this case, we can combine that idea with the checking of the result value by using the EKS_RESULT_EVAL template.

To link in another line of logic, find the template necessary in the Templates view and double click it as well. An AND will automatically be added between the two statements. In this case, before our action of adding on a confirmation test, the rule should validate that a confirmation test does not already exist on that accession number. To do this, use the EKS_ORDERS_FIND_L template. This template looks to find an order, and we can tell it to check the same accession number as the triggering action by linking it to that logic line 1. Because the template is looking for an order, and we want to verify an order does not exist, right click on the AND and select Insert NOT. Notice that many of the parameters are left with their defaults. Generally, a default parameter that starts with OPT is optional and does not need to be filled out.

IF

L1 EKS_RESULT_EVAL

RH8056 Drug Screen VERIFIED result is equal to Presumptive Pos

AND

NOT

L2 EKS_ORDERS_FIND_L

orders exist whose primary mnemonic is RH8056 Drug Confirmation with order status of Ordered, Completed, InProcess on the same accession as L1 whose date criteria is OPT_QUAL OPT_TIME_NUM OPT_TIME_UNIT from the anchor OPT_ANCHOR_DT_TM

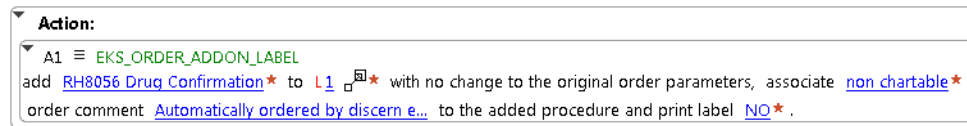
Additional logic templates could be added to check for any number of things, the patient's location, if an acceptance of a reflex was acknowledged by the ordering provider as a prompt value, if the patient's age is above or below cutoff values, etc.

ACTION SECTION

Selecting in the Action Section will change the Templates view to show only action templates. The actions of a rule are the outcomes that we want to automatically occur. Actions can be placing a new order, cancelling an order, adding a result, printing a report, sending an email, produce a notification, and so on. Actions almost always refer back to a template in the logic section to determine which order, accession, encounter, or person context to use for the action. In the above example, the action is to place an add-on order. There are three main templates to use for an add-on lab order; the EKS_ORDER_ADDON, EKS_ORDER_ADDON_SPECIMEN, and the

Notes

EKS_ORDER_ADDON_LABEL. The ADDON_LABEL template allows the builder of the rule to specify if a label should print for the new order or not.



Action sections can also have multiple conclusions. For instance, if the user who resulted the test should be notified that an add-on for the confirmation was placed, the EKS_NOTIFY template could be used with an AND joiner. EKS_NOTIFY leaves a few gaps in responding to the user however. For example, if this rule were to apply to autoverified results as well, who is there to notify? If this rule applies to performed results, how is the person managing the instrument notified? Because of this, it is important to think through the process of notification prior to using an EKS_NOTIFY template.

EXERCISE – COMPLETE THE EVOKE, LOGIC AND ACTION SECTIONS

TROUBLESHOOTING A BASIC DISCERN EXPERT RULE

TESTING RULES

In order to have a newly created rule work for the first time, servers must be cycled on each node for that environment. The first time we create a new rule this is necessary because the rule creates a CCL object that is executed. The Discern Event server needs to be cycle to see the new event. Every time a rule is updated, the applicable server for that evoke event must be cycled. This information is discussed further in the section about the Expert System Tool.

Once the proper servers have been cycled, a new rule can be tested. The EKS Monitor, run from the Expert System Tool > Report Auditing, or Discern Dev > Reports, will show step by step information that will help in determining what happened when a rule attempted to fire.

Notes

Let's examine the EKS Monitor for the example rule built above:

EXPERT KNOWLEDGE MODULE AUDIT							Page: 1	
Date/Time		Module	Conclude	Person Name	Server	AUG/11/12 14:34:27		
Type	Template Name		Person ID	Encntr ID	Order ID	Task Assay Cd	Time Secs	

AUG/11/12 14:34:03	RH8056_DRUG_CONF_REFLEX	(2)	Hoyt, Ryan		EKS_ASYNC_02	151	131	Elapsed: 5
Triggering request number: 250074								
Logic (1)	EKS_RESULT_EVAL	True	9175767	15477327	5010242123	3046014517		
Accession: 00000-12-224-000018 accession_id: 13103267								
In A, Value:Presumptive Pos, = upper nomenclaturelupn:PRESUMPTIVE POS								
Logic (2)	EKS_ORDERS_FIND_L	False	9175767	15477327	5010242123	0		
NO Orders exist whose primary mnemonic is RH8056 Drug Confirmation with order status of Ordered , Completed ,								
or InProcess on the same accession as linked to logic template 1 (0.06s)								
Action (1)	EKS_ORDER_ADDON_LABEL	True	9175767	15477327	5010242142	0		
RH8056 Drug Confirmation was ordered successfully with status of Ordered (order status cd = 2550)								

We can see the unique numbers assigned to each line of the rule and the template name that was included. For example, L2 was the EKS_ORDERS_FIND_L template. In this case that one evaluated to False, but the rule was built with AND NOT logic, so a False evaluation means the Drug Confirmation was not found on that same accession.

We can also see some logging information below each template line. For example, below L1 it spells out the value that it in fact was looking for when it evaluated (Presumptive Pos).

The numbers in the columns on the right can be used to troubleshoot. For example, we know the person_id for the patient is 9175767, the encntr_id is 15477327, the order_id of the screen is 5010242123, and the order_id of the new order is 5010242142.

It also spells out the Discern Expert server that did the processing of the rule. In this example, the EKS_ASYNC_02 (151) was responsible for handling the logic of the rule. With that information, we could check the actual server logs for further information on the processing of the rule. That log file would be located in CCLUSERDIR and named rtlsrv0151_*.log. The contents of that log file are often difficult to decipher, but using the search function in DVDev will help find the proper spot within the file.

The EKS Monitor will always show the Logic and Action lines from the current rule. If this rule is updated with a different L2, the monitor from this rule will still show as False, even though that particular logic was not applied. Because of that, it is important if major changes are made to rules, that the archived versions be available for a certain period of time.

EXERCISE – TEST YOUR RULE AND VIEW THE RESULTS IN THE EKS MONITOR

Notes

CONFIGURING A MORE COMPLEX DISCERN EXPERT RULE

RULE

The initial example was a very basic example of a Discern Expert rule that uses only one evoke and only two logic statements. No grouping is required and when processing the rule, Discern Expert servers would stop as soon as a condition determined it. What if we wanted to check for both verified and autoverified Drug Screen results?

PARENTHESIS FOR ORs

Groups, or parenthesis, can be added at any level of a Discern Expert rule. Given the above changes, modifications would be needed to both the evoke and logic sections. To insert another evoke statement that will essentially be the same as the first, right click the E1 line, select copy, then right click and paste. When inserting a second evoke template, Discern Dev automatically adds an AND statement. To change this to an OR, simply double click the word AND. With the logic section, this presents a different challenge however. If the rule can fire on either a verified or autoverified result, the current L1 could be true or false, and the rule would need to continue to evaluate in order to be successful. If the action did not refer to L1, the use of parenthesis in the logic section could be used as well.

ACTION GROUPS

An option exists in Discern Expert to allow the processing to continue even if a logic line evaluates as false. By inserting an Action Group into a rule, Discern Expert will change the way it evaluates the logic lines. Action groups can be used when different outcomes are needed for slight variances in logic. Because of that, the logic section can be joined with all ANDs without worrying about the individual qualifications.

IF

L1 EKS_RESULT_EVAL

RH8056 Drug Screen VERIFIED result is equal to Presumptive Pos

AND

L2 EKS_RESULT_EVAL

RH8056 Drug Screen AUTOVERIFIED result is equal to Presumptive Pos

AND

L3 EKS_ORDERS_FIND_L

orders exist whose primary mnemonic is RH8056 Drug Confirmation with order status of Ordered, Completed, InProcess on the same accession as L1 whose date criteria is OPT_QUAL OPT_TIME_NUM OPT_TIME_UNIT from the anchor OPT_ANCHOR_DT_TM

AND

L4 EKS_ORDERS_FIND_L

orders exist whose primary mnemonic is RH8056 Drug Confirmation with order status of Ordered, Completed, InProcess on the same accession as L2 whose date criteria is OPT_QUAL OPT_TIME_NUM OPT_TIME_UNIT from the anchor OPT_ANCHOR_DT_TM

To add an action group, right click in the action section and choose Insert Action Group.

Action:

Insert Template

Insert Grouping

Insert Action Group

Undo

Redo

Cut

Copy

Paste

Select the line with the IF in the new group that's added and the Logic Template Reference will be highlighted in the Templates view. Double click the templates that need to be evaluated for the particular action. In this example, the IF should read:

Action Group #1

IF L1 AND NOT L3

The same should be completed to add a second Action Group that will evaluate if the results are autoverified. Clicking on the three lined equals sign will let an alias for that group to be named.

Action Section

Action Group #1 Add on confirmation if verified ...

Action Group #2

IF L2 AND NOT L4

Action:

A2 = EKS_ORDER_ADDON_LABEL

add RH8056 Drug Confirmation* to L2 with no change to the original order parameters, associate non chartable* order comment Automatically ordered by discern e...

Now when the EKS Monitor is run, all 4 of the logic lines have been evaluated:

EXPERT KNOWLEDGE MODULE AUDIT									
						Page: 1			
						AUG/11/12 15:33:49			
Date/Time	Module	Conclude	Person Name	Server	Time	Task	Assay	Cd	
Type	Template Name		Person ID	Order ID	Secs				

AUG/11/12 15:33:35	RH8056_DRUG_CONF_REFLEX	(2)	Hoyt, Ryan	EKS_ASYNC_02	151	30	Elapsed: 4		
Triggering request number: 250074									
Logic (1)	EKS_RESULT_EVAL	True	9175767	15477327	5010245275	3046014517			
Accession: 00000-12-224-000020 accession_id: 13103270									
In A, Value:Presumptive Pos, = upper nomenclaturelupn:PRESUMPTIVE POS									
Logic (2)	EKS_RESULT_EVAL	False	9175767	15477327	5010245275	3046014517			
EKS-I- Returning 0 - EKS GoodReturn2									
Logic (3)	EKS_ORDERS_FIND_L	False	9175767	15477327	5010245275	0			
NO Orders exist whose primary mnemonic is RH8056 Drug Confirmation with order status of Ordered , Completed ,									
or InProcess on the same accession as linked to logic template 1 (0.04s)									
Logic (4)	EKS_ORDERS_FIND_L	False	9175767	15477327	5010245275	0			
NO Orders exist whose primary mnemonic is RH8056 Drug Confirmation with order status of Ordered , Completed ,									
or InProcess on the same accession as linked to logic template 2 (0.05s)									
Action (1)	EKS_ORDER_ADDON_LABEL	True	9175767	15477327	5010245554	0			
RH8056 Drug Confirmation was ordered successfully with status of Ordered (order_status_cd = 2550)									
Action (2)	EKS_ORDER_ADDON_LABEL	NotRun	0	0	0	0			

EXERCISE – ADAPT YOUR CURRENT RULE TO CHECK FOR AUTOVERIFIED RESULTS

CUSTOM CCL LOGIC STATEMENTS

Cerner provides numerous templates for logic evaluation, but sometimes an additional qualifier may be needed that is not available in the template list. The EKS_EXEC_CCL_L template allows the a CCL to be directly integrated into the logic statements of a rule. The Template Doc for the EKS_EXEC_CCL_L template provides the basic instructions for executing and returning a CCL value.

A good example of this is on an COLLECTION_EVENT trigger, if it is necessary to flex the outcome based on the current location of the specimen, this information is not available within the Cerner templates. However, we know from checking the COLLECTION_EVENT request (265060), that the current location of the specimen is available at the highest level of the request. With this knowledge, a short CCL script can be written to evaluate the specimen login location entered in the rule, and the location of the specimen:

Notes

```
drop program rule_gl_check_login_location go
create program rule_gl_check_login_location

set retval = -1
set msg = "The script did not complete successfully"

declare login_loc_cd = vc
set login_loc_cd = trim(cnvstring(request->curr_loc_cd))

if (login_loc_cd = OPT_PARAM)
    set msg = "Login Location matches OPT_PARAM value"
    set retval = 100
else
    set msg = "Login location did not match"
    set retval = 0
endif

set log_message = msg

end go
```

This is this included in a rule as a logic statement and the parameter that is passed is the location_cd of the login location that should be compared to the current location.

l1   EKS_EXEC_CCL_L
Execute [rule_gl_check_login_location](#)★ with parameters [28327192](#)

EXERCISE – INCORPORATE THIS CCL INTO A COLLECTION_EVENT RULE

EXAMPLE OF COMPLEX GROUPING AND PARENTHESIS

Evoke Section

Evoke on [RESULT_EVENT_CSM](#) where,

E1 CSM_ORD_E_ORGANIZATION
Client [IS](#)★ among: [ORGANIZATION2](#), [ORGANIZATION3](#), [ORGANIZATION4](#), [ORGANIZATION5](#), [ORGANIZATION6](#), [ORGANIZATION7](#), [ORGANIZATION8](#), [ORGANIZATION9](#), [ORGANIZATION10](#), [ORGANIZATION11](#), [ORGANIZATION12](#), [ORGANIZATION13](#), [ORGANIZATION14](#), [ORGANIZATION15](#), [ORGANIZATION16](#), [ORGANIZATION17](#), [ORGANIZATION18](#), [ORGANIZATION19](#)

AND

E2 CSM_ORD_E_COMPLETED_IND
The order [IS](#)★ complete

AND

E3 CSM_ORD_E_RPT_PRIORITY
Report priority [IS NOT](#)★ among: [SI-Stat](#)★, [PRIORITY2](#), [PRIORITY3](#), [PRIORITY4](#), [PRIORITY5](#), [PRIORITY6](#), [PRIORITY7](#), [PRIORITY8](#), [PRIORITY9](#), [PRIORITY10](#), [PRIORITY11](#), [PRIORITY12](#), [PRIORITY13](#), [PRIORITY14](#), [PRIORITY15](#), [PRIORITY16](#), [PRIORITY17](#), [PRIORITY18](#), [PRIORITY19](#)

AND

(

E4 CSM_DTA_E_RESULT_STATUS
Result status is [Verified](#)★

OR

E5 CSM_DTA_E_RESULT_STATUS
Result status is [Autoverified](#)★

)

AND

NOT

(

E6 CSM_DTA_E_RESULT_CODE
Result code is [1740 Critical Numeric](#)★

OR

E7 CSM_DTA_E_RESULT_CODE
Result code is [1739 Critical Alpha](#)★

)

AND

E8 CSM_DTA_E_TASK_ASSAY
Task assay [IS](#)★ among: [HGB](#)★, [PLT](#), [NA](#), [K](#), [INR](#), [GLU](#), [BUN](#), [CA](#), [WBC](#), [TASK_ASSAY10](#), [TASK_ASSAY11](#), [TASK_ASSAY12](#), [TASK_ASSAY13](#), [TASK_ASSAY14](#), [TASK_ASSAY15](#), [TASK_ASSAY16](#), [TASK_ASSAY17](#), [TASK_ASSAY18](#)

Logic Section

IF

L1 CSM_ORD_I_COMPLETED_IND
The order [IS](#)★ complete

AND

(

(

L2 CSM_DTA_I_TASK_ASSAY
Task assay [IS](#)★ among: [HGB](#), [TASK_ASSAY2](#), [TASK_ASSAY3](#), [TASK_ASSAY4](#), [TASK_ASSAY5](#), [TASK_ASSAY6](#), [TASK_ASSAY7](#), [TASK_ASSAY8](#), [TASK_ASSAY9](#), [TASK_ASSAY10](#), [TASK_ASSAY11](#), [TASK_ASSAY12](#), [TASK_ASSAY13](#), [TASK_ASSAY14](#), [TASK_ASSAY15](#), [TASK_ASSAY16](#), [TASK_ASSAY17](#), [TASK_ASSAY18](#) link to [L2](#) □

AND

L3 CSM_DTA_I_RESULT_VALUE
Result value is [≤](#)★ [10.0](#)★ link to [L2](#) □

)

OR

(

L4 CSM_DTA_I_TASK_ASSAY
Task assay [IS](#)★ among: [PLT](#), [TASK_ASSAY2](#), [TASK_ASSAY3](#), [TASK_ASSAY4](#), [TASK_ASSAY5](#), [TASK_ASSAY6](#), [TASK_ASSAY7](#), [TASK_ASSAY8](#), [TASK_ASSAY9](#), [TASK_ASSAY10](#), [TASK_ASSAY11](#), [TASK_ASSAY12](#), [TASK_ASSAY13](#), [TASK_ASSAY14](#), [TASK_ASSAY15](#), [TASK_ASSAY16](#), [TASK_ASSAY17](#), [TASK_ASSAY18](#) link to [L2](#) □

AND

(

L5 CSM_DTA_I_RESULT_VALUE
Result value is [≤](#)★ [100.0](#)★ link to [L2](#) □

OR

L6 CSM_DTA_I_RESULT_VALUE
Result value is [≥](#)★ [600.0](#)★ link to [L2](#) □

)

)

OR

Notes



Notes

