Management Reporting for EKS Rules

*Darin Cozine, Adventist Health*

# Purpose:

Rules and Alerts perform an important role in the Cerner Millennium system. They run automatically without user initiation, and often without any user’s knowledge. Because these processes are mostly hidden, they can often be forgotten or out of date.

This document identifies statistics that should be monitored regularly. The queries to gather this information are included. These can be run directly from an ad-hoc query window in Discern Visual Developer or integrated into automated reports.

Table of Contents

[Purpose: 1](#_Toc504121506)

[Monthly reports: 2](#_Toc504121507)

[Monthly Executive Summary 2](#_Toc504121508)

[Rule processing summary 3](#_Toc504121509)

[Unreconciled rules 4](#_Toc504121510)

[Rule Performance Problems 5](#_Toc504121511)

[Alert Outcomes 6](#_Toc504121512)

[Weekly reports: 8](#_Toc504121513)

[Updated rules 8](#_Toc504121514)

[Updated rule executions before and after 9](#_Toc504121515)

[Rule execution changes 11](#_Toc504121516)

# Monthly reports:

## Monthly Executive Summary

These are some general statistics that can provide a general understanding of your rules environment. This format presents well to management.

1. Number of active rules:
2. Number of active rules with failures:
3. Number of active rules with 0 runs.
4. Number of rules with reconciliation errors.
5. Top 10 alerts
6. Top 10 non-alerts

Number of active rules with 0 runs can vary greatly. Some rules are designed to run infrequently, when a rare case occurs. It may be better to set a pivot point of 10 or 20 runs etc.

These stats can be gathered using the other queries listed in this document. This could be consolidated into a single DVD report if one had the time.

## Rule processing summary

This lists all the active production rules and the run and failure counts for each.

SELECT

em.module\_name

,lastModifiedBy=p.name\_full\_formatted

,lastmodifiedDate=em.updt\_dt\_tm

,Runs=sum(t.records)

, count\_logic\_false=sum(evaluate( t.conclude,0 , (t.records),0))

, count\_logic\_true=sum(evaluate( t.conclude,1 , (t.records),0))

, count\_success=sum(evaluate( t.conclude,2 , (t.records),0))

, count\_fail=sum(evaluate( t.conclude,3 ,(t.records),0))

FROM

eks\_module em

,prsnl p

,((

select records=count(\*)

,e2.module\_name

,e2.conclude

from EKS\_MODULE\_AUDIT e2

where e2.begin\_dt\_tm BETWEEN cnvtdatetime(curdate-30,0) and cnvtdatetime(curdate,0)

GROUP BY

e2.module\_name

,e2.conclude

WITH sqltype("f8","vc","i4"),TIME=1

)T)

WHERE

em.active\_flag = "A"

AND em.maint\_validation = "PRODUCTION"

AND em.maint\_dur\_begin\_dt\_tm < sysdate

AND em.maint\_dur\_end\_dt\_tm > sysdate

AND p.person\_id=em.updt\_id

AND t.module\_name = OUTERJOIN(em.module\_name)

GROUP BY

em.module\_name

,p.name\_full\_formatted

,em.updt\_dt\_tm

ORDER BY

em.module\_name

Action items:

Rules with more than 1 or 2 failures a month should be reviewed. Some rules are ‘designed to fail’, or allowed to fail as a way to exit the code processing. This should be avoided unless there is no other option. -If a rule is failing normally, how will you know if it fails abnormally? In some cases resolving these issues may only be possible with custom CCL code, or by requesting changes to the templates themselves.

Rules with 0 or few successful runs should be reviewed periodically. Rules that should be running regularly have a high priority. Rules based from old DTA’s or workflows should be expired, but can be done as time allows.

Note that count\_logic\_true and count\_logic\_false are only counted when a rule is set to FULL\_AUDIT mode. This should be sitched off if it is not actively being used for debugging purposes.

## Unreconciled rules

These are rules that use a DTA, Event Code, Order, or other item that is no longer valid. Many rules will still run correctly when a DTA is changed. There is some logic to lookup the correct item based on name, ID, or CKI. I am not exactly sure of the details. But if you have many rules with reconciliation errors, how will you know which are working correctly and which are not?

SELECT

em.module\_name

,em.updt\_dt\_tm

,last\_update\_by = pl.name\_full\_formatted

,dm\_reconcile\_flag=dm.description

FROM

eks\_module em

,prsnl pl

,dm\_flags dm

WHERE

em.active\_flag = "A"

AND em.maint\_dur\_begin\_dt\_tm < sysdate

AND em.maint\_dur\_end\_dt\_tm > sysdate

AND em.maint\_validation = "PRODUCTION"

AND em.reconcile\_flag = 2 ;issues with instantiation

AND pl.person\_id=em.updt\_id

AND dm.flag\_value=em.reconcile\_flag

AND dm.table\_name = "EKS\_MODULE"

AND dm.column\_name = "RECONCILE\_FLAG"

ORDER BY

last\_update\_by

,em.module\_name

Action Items:

Review the identified rules. Templates with errors will be identified with a blue  icon. Replace or remove the old items.

Note that templates that use custom code are not included in the reconciliation flags!   
(CALC\_SAVE, EXEC\_CCL, EVAL)

## Rule Performance Problems

This query highlights any rules which execute in 2 seconds or greater. Rules that take over 5 seconds are just grouped together as “over 5”.

This is similar to the lighthouse report, but filters on server 175. Rules that run on server 175 actually force the user to wait , so are the highest priority. All alerts run on server 175, but not all rules that run on 175 create an alert. Rules that run on servers 150, 151, and 152 do not interrupt the user. However they are still important for overall system performance.

Note that rules are not logged if they resolve to false (create no action). So if you are seeing some performance problems with a rule, it is likely the tip of the iceberg. You can set a rule to FULL AUDIT to see its true performance impact.

The log only records full seconds, no fractions. So if a rule starts on one second and ends on the next, you don’t know if that rule ran for .10 second or 1.90 seconds. You can still determine a correct average from a group though.

SELECT

t.module\_name

,count2=sum(evaluate(t.testtime,2,1))

,count3=sum(evaluate(t.testtime,3,1))

,count4=sum(evaluate(t.testtime,4,1))

,count5=sum(evaluate(t.testtime,5,1))

,countOver5=sum(evaluate(t.testtime,6,1))

FROM

((

SELECT

ma.module\_name

,testtime=evaluate2(

IF(datetimediff(ma.end\_dt\_tm, ma.begin\_dt\_tm)\*24\*60\*60>5.0) 6

ELSE datetimediff(ma.end\_dt\_tm, ma.begin\_dt\_tm)\*24\*60\*60

ENDIF )

FROM EKS\_MODULE\_AUDIT ma

WHERE ma.begin\_dt\_tm BETWEEN cnvtdatetime(curdate-30,0) AND cnvtdatetime(curdate,0)

AND ma.server\_number=175

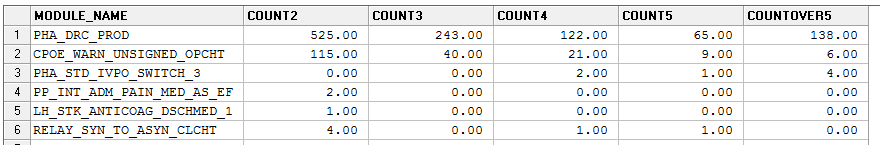
AND datetimediff(ma.end\_dt\_tm, ma.begin\_dt\_tm)\*24\*60\*60>1

WITH maxrec=10, time=1, FORMAT(DATE,"@SHORTDATETIME"), SQLTYPE("vc","i4")

)T)

GROUP BY t.module\_name

ORDER BY countOver5 DESC



Action Items:

Any rule with a substantial count of runs over 5 seconds should be addressed immediately.

## Alert Outcomes

This space is reserved for alert outcomes reporting.

When users are presented an alert, there are often options given. It will be important to understand how often the options are used vs how often an alert is ignored. I believe Lighthouse already includes this type of reporting. However, I have not has a chance to delve into it yet.

The alert outcomes are listed in the EKS\_DLG tables. There is no direct link from these tables to the MODULE\_AUDIT tables. Reports on this table to not match exactly to the MODULE\_AUDIT table. Here are some useful queries:

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;top alerts

SELECT alert\_count=count(ede.dlg\_event\_id)

,ede.dlg\_name

FROM eks\_dlg\_event ede

,eks\_dlg ed

,eks\_module m

WHERE ed.dlg\_name=ede.dlg\_name

AND m.module\_name=ed.program\_name

AND ede.dlg\_dt\_tm BETWEEN CNVTDATETIME(curdate-30,0)

AND CNVTDATETIME(curdate,0)

AND ede.action\_flag!=5

AND m.active\_flag="A"

AND m.maint\_validation ="PRODUCTION"

AND m.maint\_dur\_end\_dt\_tm > sysdate

GROUP BY ede.dlg\_name

ORDER BY 1 DESC

WITH maxrec=20, time=100

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;users with most frequent alerts

SELECT record\_count=count(\*)

,p.name\_full\_formatted

FROM eks\_dlg\_event ede

,eks\_dlg ed

,eks\_module m

,prsnl p

WHERE ed.dlg\_name=ede.dlg\_name

AND m.module\_name=ed.program\_name

AND p.person\_id=ede.active\_status\_prsnl\_id

AND ede.dlg\_dt\_tm BETWEEN CNVTDATETIME(curdate-30,0) AND CNVTDATETIME(curdate,0)

AND ede.action\_flag!=5

AND m.active\_flag="A"

AND m.maint\_validation ="PRODUCTION"

AND m.maint\_dur\_end\_dt\_tm > sysdate

GROUP BY

p.name\_full\_formatted

,p.person\_id

ORDER BY 1 DESC

WITH time=100, maxrec=10

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;top alerts by position

SELECT

record\_count=count(\*)

,position=uar\_get\_code\_description(p.position\_cd)

,m.module\_name

FROM

eks\_dlg\_event ede

,eks\_dlg ed

,eks\_module m

,prsnl p

WHERE

ed.dlg\_name=ede.dlg\_name

AND m.module\_name=ed.program\_name

AND p.person\_id=ede.active\_status\_prsnl\_id

AND ede.dlg\_dt\_tm BETWEEN CNVTDATETIME(curdate-30,0) AND CNVTDATETIME(curdate,0)

AND ede.action\_flag!=5

AND m.active\_flag="A"

AND m.maint\_validation ="PRODUCTION"

AND m.maint\_dur\_end\_dt\_tm > sysdate

AND p.position\_cd= 271051182.00

GROUP BY

p.position\_cd

,m.module\_name ORDER BY 1 DESC

WITH maxrec=10, time=100

# Weekly reports:

These reports are useful to monitor the regular changes in executions of rules and alerts. And to keep an eye on changes in the rules themselves, as often the rules are managed by a widely distributed group of developers.

I have found that the statistics vary greatly from day-today. Instead it is better to report on a weekly basis.

## Updated rules

This lists any rules that were updated in the past week and who updated them. In our environment, each support team (pharmacy, ambulatory, etc) manages their own rules. But they must submit them for review before moving to production. This process has proven to be problematic, both in execution and politically. But this is still useful to know.

SELECT

updated\_by=p.name\_full\_formatted

,m.module\_name

, m.updt\_dt\_tm

FROM

eks\_module m

,prsnl p

WHERE

p.person\_id=m.updt\_id

AND m.updt\_dt\_tm BETWEEN cnvtdatetime(curdate-7,0) and cnvtdatetime(curdate,0)

AND m.active\_flag="A"

ORDER BY

p.name\_full\_formatted

,m.module\_name

## Updated rule executions before and after

This query lists all the rules that were updated 7-14 days ago, and how often they were executed before and after the change. I send this out to the rule writers weekly as an FYI. This way they know if a minor change accidentally had a minor impact. Or if a major change had the desired effect.

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;before and after updates

SELECT

lastModifiedBy=p.name\_full\_formatted

,em.module\_name

;,em.version

,status="Action Performed"

,lastmodifiedDate=em.updt\_dt\_tm

,before=b.before

,after=a.after

FROM

prsnl p

,eks\_module em

,((

SELECT

em.module\_name

,before=count(\*)

FROM

eks\_module em

,EKS\_MODULE\_AUDIT e

WHERE

em.active\_flag = "A"

AND em.maint\_validation = "PRODUCTION"

AND em.maint\_dur\_begin\_dt\_tm < sysdate

AND em.maint\_dur\_end\_dt\_tm> sysdate

AND em.updt\_dt\_tm BETWEEN CNVTDATETIME(curdate-14,0) AND CNVTDATETIME(curdate-7,0)

AND e.module\_name=em.module\_name

AND e.end\_dt\_tm!=null

AND e.begin\_dt\_tm BETWEEN em.updt\_dt\_tm-7 AND em.updt\_dt\_tm

AND e.conclude =2

GROUP BY

em.module\_name

WITH sqltype("vc","i4")

)B)

,((

SELECT

em.module\_name

,after=count(\*)

FROM

eks\_module em

,EKS\_MODULE\_AUDIT e2

WHERE

em.active\_flag = "A"

AND em.maint\_validation = "PRODUCTION"

AND em.maint\_dur\_begin\_dt\_tm < sysdate

AND em.maint\_dur\_end\_dt\_tm> sysdate

AND em.updt\_dt\_tm BETWEEN CNVTDATETIME(curdate-14,0) AND CNVTDATETIME(curdate-7,0)

AND e2.module\_name=em.module\_name

AND e2.end\_dt\_tm!=null

AND e2.begin\_dt\_tm BETWEEN em.updt\_dt\_tm AND em.updt\_dt\_tm+7

AND e2.conclude =2

GROUP BY

em.module\_name

WITH sqltype("vc","i4")

)A)

WHERE

em.active\_flag = "A"

AND em.maint\_validation = "PRODUCTION"

AND em.maint\_dur\_begin\_dt\_tm < sysdate

AND em.maint\_dur\_end\_dt\_tm> sysdate

AND em.updt\_dt\_tm BETWEEN CNVTDATETIME(curdate-14,0) AND CNVTDATETIME(curdate-7,0)

AND p.person\_id=em.updt\_id

AND a.module\_name=OUTERJOIN(em.module\_name)

AND b.module\_name=OUTERJOIN(em.module\_name)

ORDER BY

lastModifiedBy

,em.module\_name

## Rule execution changes

These querys will show significant changes to rule execution counts regardless of when they were updated.

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;rule success counts per day have signifigantly changed

SELECT

em.module\_name

,status="Action Performed"

,pivotdate=format(cnvtdatetime(curdate-8,0),"@SHORTDATE")

,before=cnvtint(t.records)

,after=cnvtint(t2.records)

,pctChange=cnvtint(100\*(t2.records-t.records)/(t.records+1))

,lastModifiedBy=p.name\_full\_formatted

,lastmodifiedDate=em.updt\_dt\_tm

FROM

((

SELECT records=count(\*)

,e.module\_name

FROM EKS\_MODULE\_AUDIT e

WHERE e.begin\_dt\_tm BETWEEN cnvtdatetime(curdate-22,0) and cnvtdatetime(curdate-15,0)

AND e.conclude =2

GROUP BY

e.module\_name

WITH sqltype("f8","vc")

)T)

,((

SELECT records=count(\*)

,e2.module\_name

FROM EKS\_MODULE\_AUDIT e2

WHERE e2.begin\_dt\_tm BETWEEN cnvtdatetime(curdate-8,0) and cnvtdatetime(curdate,0)

AND e2.conclude =2

GROUP BY

e2.module\_name

WITH sqltype("f8","vc")

)T2)

,eks\_module em

,prsnl p

WHERE

t2.module\_name=outerjoin(em.module\_name)

AND t.module\_name = outerjoin (em.module\_name)

AND (t.records>100 or t2.records>100)

AND (abs((t2.records-t.records)/(t.records+1))>0.9

OR t.records is null

OR t2.records is null)

AND em.active\_flag = "A"

AND em.maint\_validation = "PRODUCTION"

AND em.maint\_dur\_begin\_dt\_tm < sysdate

AND em.maint\_dur\_end\_dt\_tm > sysdate

;AND em.reconcile\_flag = 2 ;issues with instantiation

AND p.person\_id=em.updt\_id

ORDER BY

pctChange DESC

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;rule failure counts per day have signifigantly changed

SELECT

em.module\_name

,status="Failed"

,pivotdate=format(cnvtdatetime(curdate-8,0),"@SHORTDATE")

,before=cnvtint(t.records)

,after=cnvtint(t2.records)

,pctChange=cnvtint(100\*(t2.records-t.records)/(t.records+1))

,lastModifiedBy=p.name\_full\_formatted

,lastmodifiedDate=em.updt\_dt\_tm

FROM

((

SELECT records=count(\*)

,e.module\_name

FROM EKS\_MODULE\_AUDIT e

WHERE e.begin\_dt\_tm BETWEEN cnvtdatetime(curdate-22,0) and cnvtdatetime(curdate-15,0)

AND e.conclude =3

GROUP BY

e.module\_name

WITH sqltype("f8","vc")

)T)

,((

SELECT records=count(\*)

,e2.module\_name

FROM EKS\_MODULE\_AUDIT e2

WHERE e2.begin\_dt\_tm BETWEEN cnvtdatetime(curdate-8,0) and cnvtdatetime(curdate,0)

AND e2.conclude =3

GROUP BY

e2.module\_name

WITH sqltype("f8","vc")

)T2)

,eks\_module em

,prsnl p

WHERE

t2.module\_name=outerjoin(em.module\_name)

AND t.module\_name = outerjoin (em.module\_name)

AND (t.records>100 or t2.records>100)

AND (abs((t2.records-t.records)/(t.records+1))>0.9

OR t.records is null

OR t2.records is null)

AND em.active\_flag = "A"

AND em.maint\_validation = "PRODUCTION"

AND em.maint\_dur\_begin\_dt\_tm < sysdate

AND em.maint\_dur\_end\_dt\_tm > sysdate

;AND em.reconcile\_flag = 2 ;issues with instantiation

AND p.person\_id=em.updt\_id

ORDER BY

pctChange DESC