knowledge14®

April 27–May 1, 2014 • Moscone Center • San Francisco, CA

Lab Guide

Performance Tuning

Garrett Prochnow & Matt Kilbride

Lab instance: https://**<<name>>**.lab.service-now.com

Login / Passwords:

admin / Knowledge14

itil / Knowledge14

employee / Knowledge14

Lab Goal

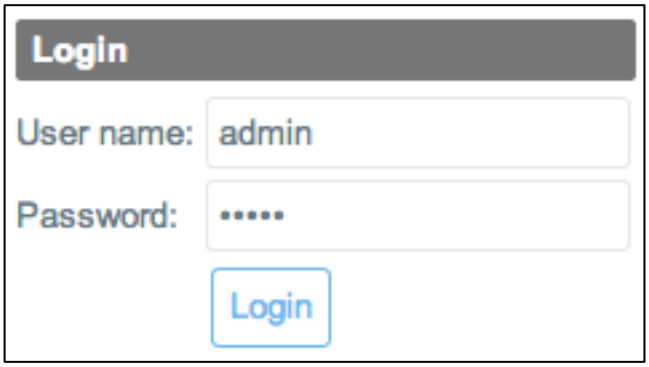
Utilize instance diagnostics, graphs, and queues to diagnose performance-related issues.

Lab 1
Diagnose current instance status

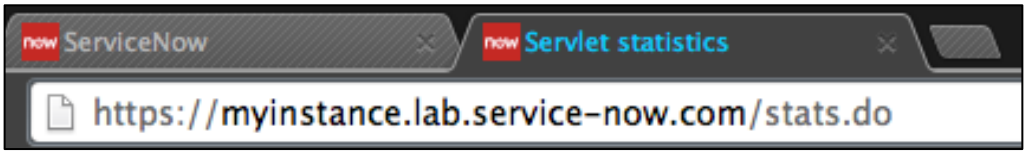
Check stats.do

1. The instructor provides you with an instance URL.

2. Log into your instance, using the username **admin** and the password **admin**.

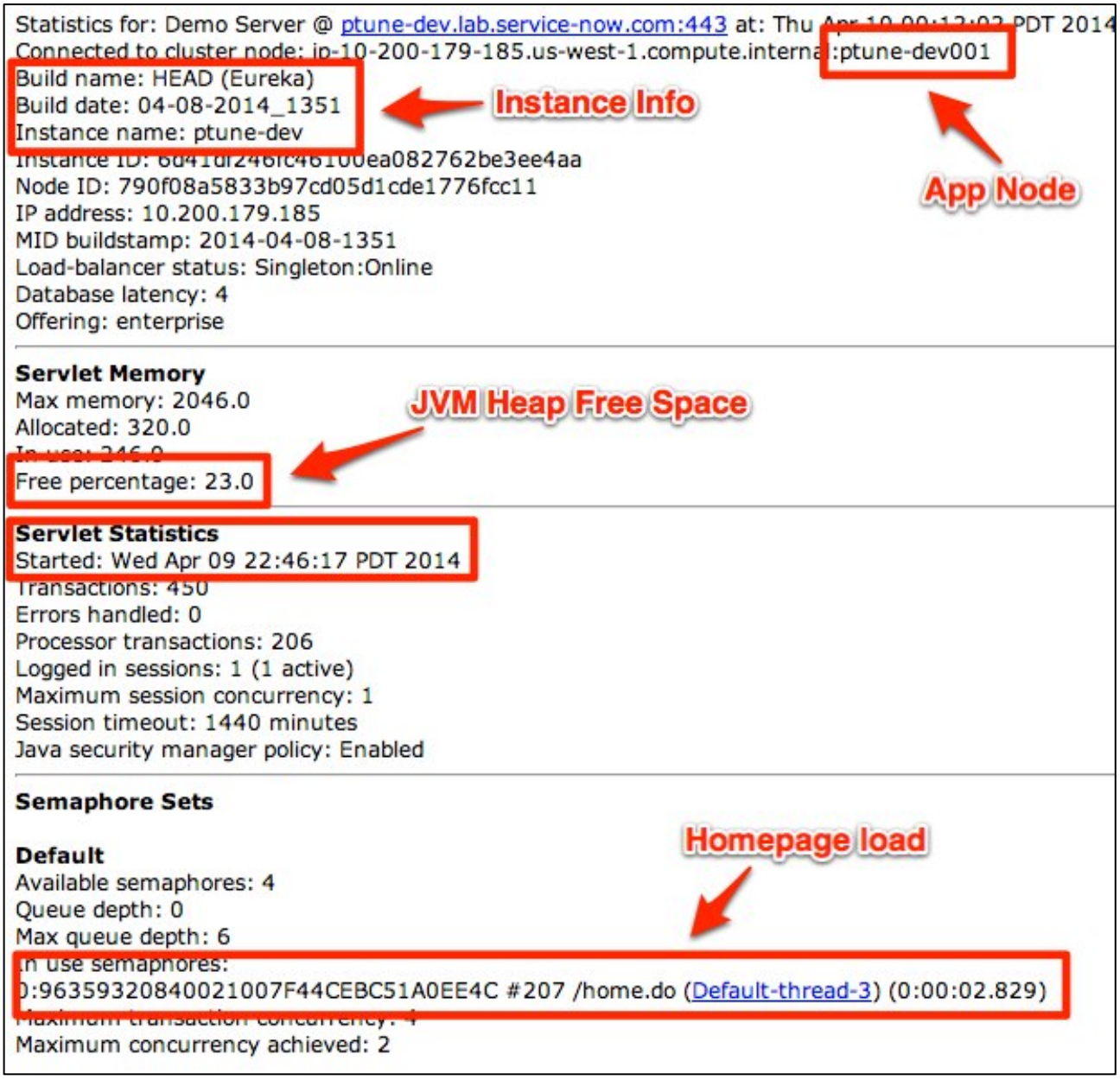


3. Navigate to the **/stats.do** page in another tab for a quick look at the status of the currently connected node.



©2014 ServiceNow, Inc. All rights reserved.

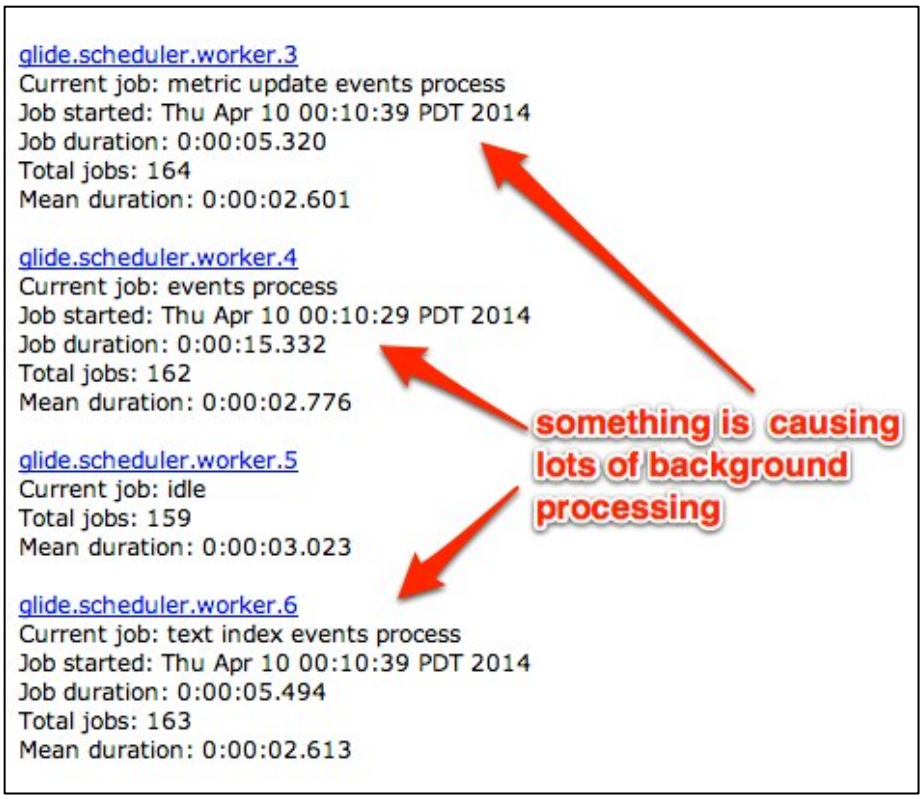
4. Observe any active user transactions in the **default** semaphore set, semaphore waits, active scheduler workers, and Server Response Time metrics. This serves as a launching point for investigation. With no user traffic, the semaphores and response times will not be very interesting.



©2014 ServiceNow, Inc. All rights reserved.

Note: The cluster node information is the fastest way to identify the active datacenter for the instance. The domain name contains a sub-domain that identifies the datacenter, which is named after the closest airport.

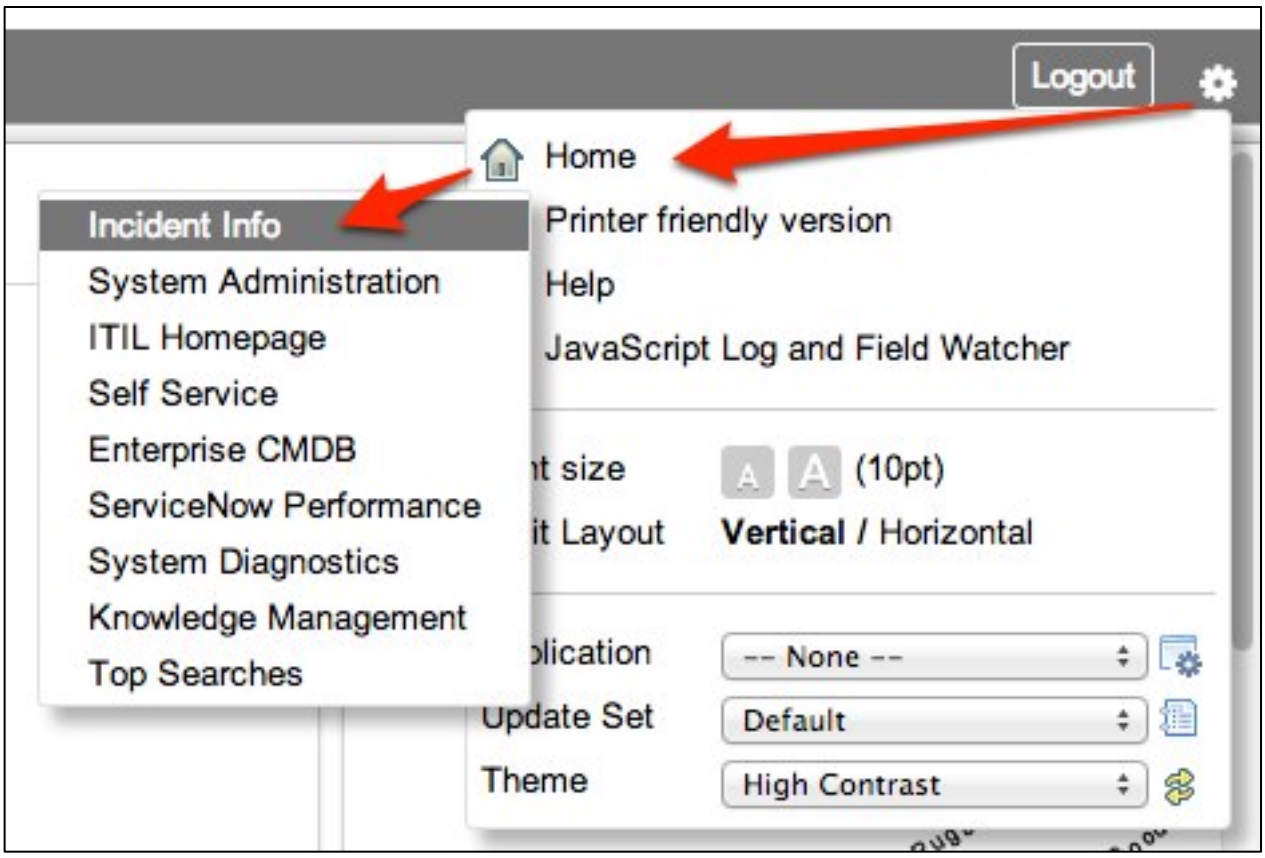




Note: All production and sub-production instances utilize multiple app nodes for load-balancing and redundancy.

©2014 ServiceNow, Inc. All rights reserved.

5. Switch back to the first tab, and visit the **Incident Info** homepage. While the homepage is loading, switch back to **stats.do** and refresh the page. Note the user transaction for **/home.do**. Refresh the page several times to get frequent updates on the status.

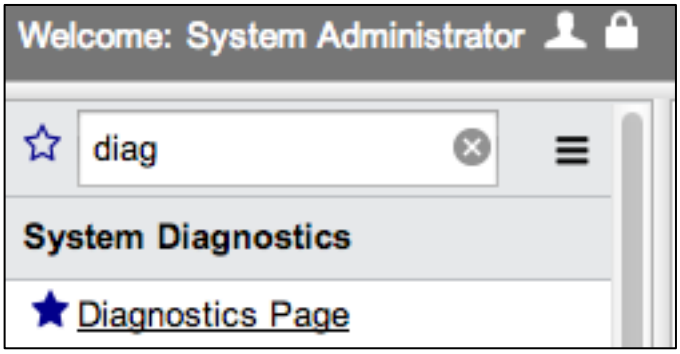




©2014 ServiceNow, Inc. All rights reserved.

Check System Diagnostics

6. Navigate to the **Diagnostics Page** module,



NOTE: When clicked, the star will turn blue to indicate addition to the navigation favorites. Favorites can be toggled using the star to the left of the filter.

©2014 ServiceNow, Inc. All rights reserved.

7. Observe the node status for each of node (in this case there will be only one node online). Also observe the data contained in the **system overview**, which includes current email, event, and logging details.

Cluster Nodes Status

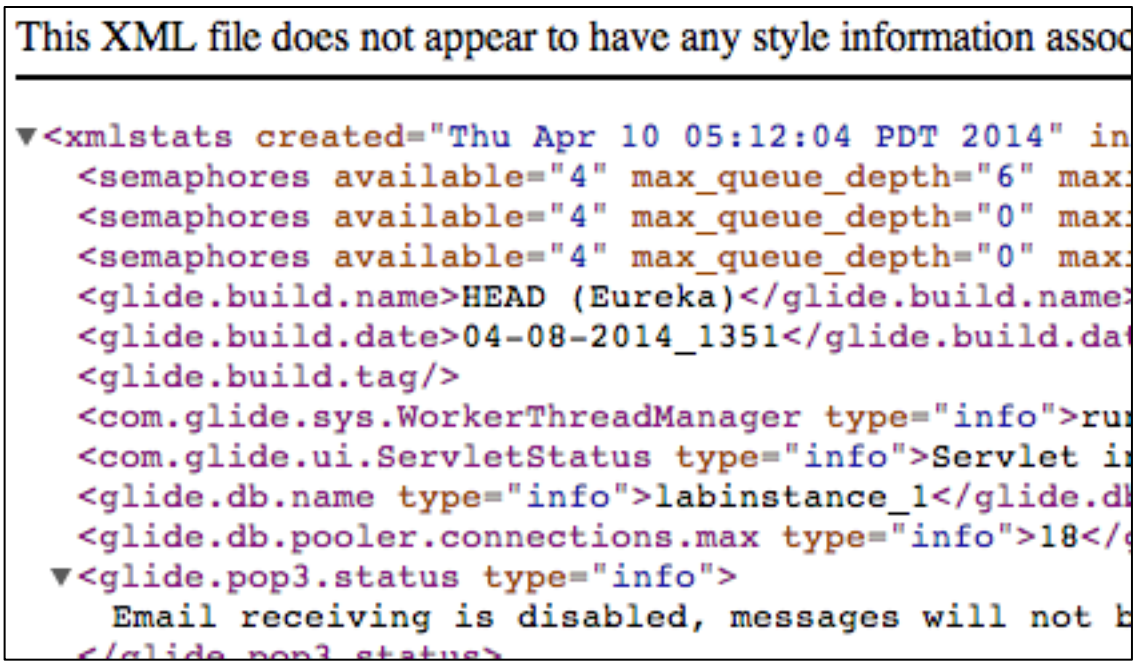
Name	ip-10-200-179-185.us-west-1.compute.internal:ptune-dev001
Status	online
Logged in users	2
Last reported	0 seconds
Last reported (seconds ago)	0
JVM UP time	5 hours 38 minutes
JVM CPU time	1 hour 33 minutes
Scheduler running	true
Scheduler queue length	0
Memory (MB)	218.0 of 2046.0
PermGen (MB)	163.0 of 256.0
JVM Classes	26109.0 loaded, 106.0 unloaded
Transactions	780
Errors	0
GC.ConcurrentMarkSweep.Count	153 (2 per 5 minutes)
GC.ConcurrentMarkSweep.TotalTime	2535 (2 seconds)
GC.ConcurrentMarkSweep.AvgTime	0.016568627
GC.Copy.Count	97717 (1444 per 5 minutes)
GC.Copy.TotalTime	723034 (12 minutes)
GC.Copy.AvgTime	0.007399265

System overview

Emails (recv) last 60 minutes	0
Emails (sent) last 60 minutes	0
Events pending	310
Log entries last 60 minutes	6461
POP3 Status	Email receiving is disabled, messages will not be processed
SMTP Status	Email sending is inactive, messages will not be sent

©2014 ServiceNow, Inc. All rights reserved.

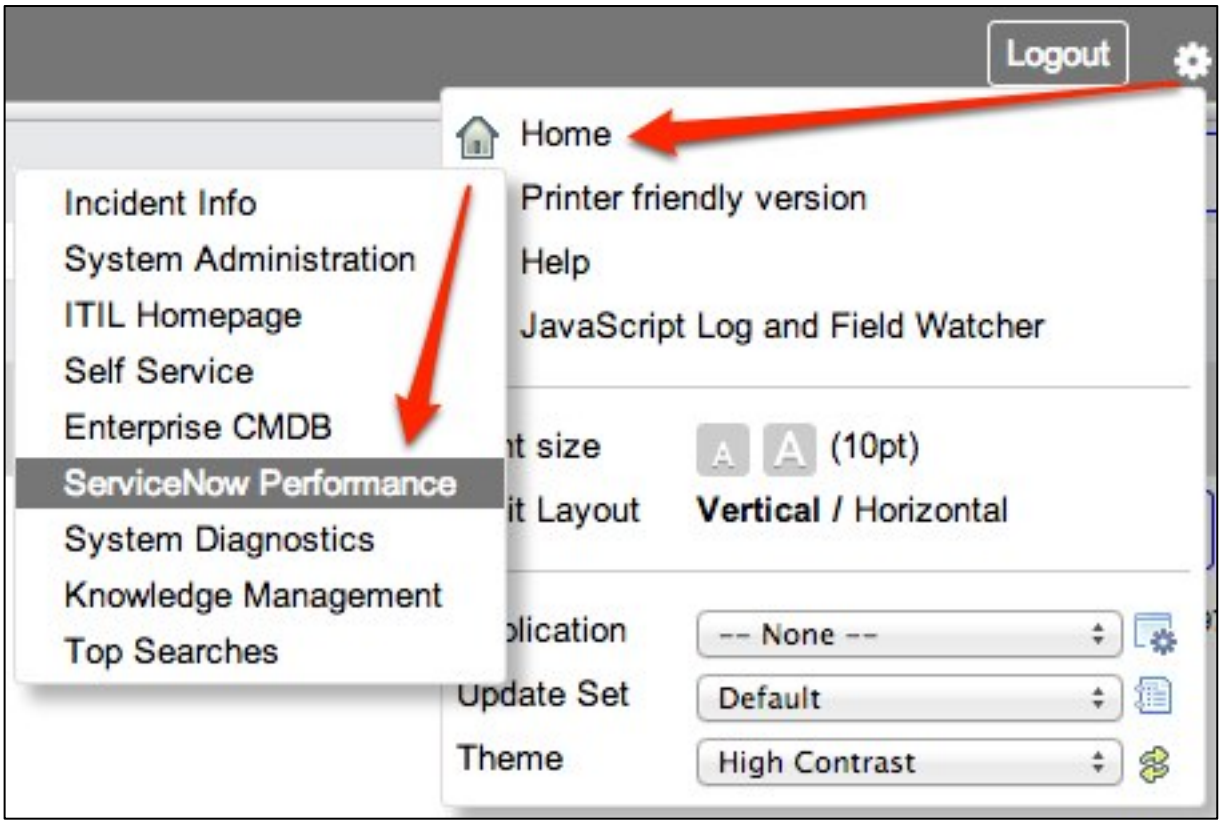
8. Click on the **name** of your online node in the **Cluster Nodes Status** section to bring up that node's xmlstats.do results. Each app node saves this information to the database every 60 seconds. This provides visibility to the status of other app nodes.



®2014 ServiceNow, Inc. All rights reserved.

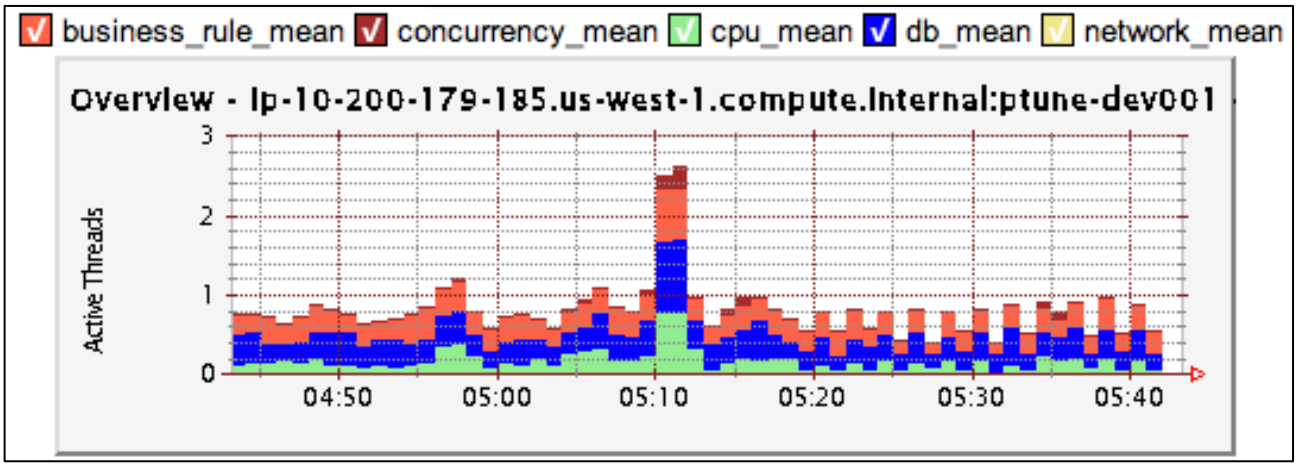
View the Performance Graphs

9. Next, use the gear in the upper-right corner to bring up the homepage icon, hover over it, and select **ServiceNow Performance**.

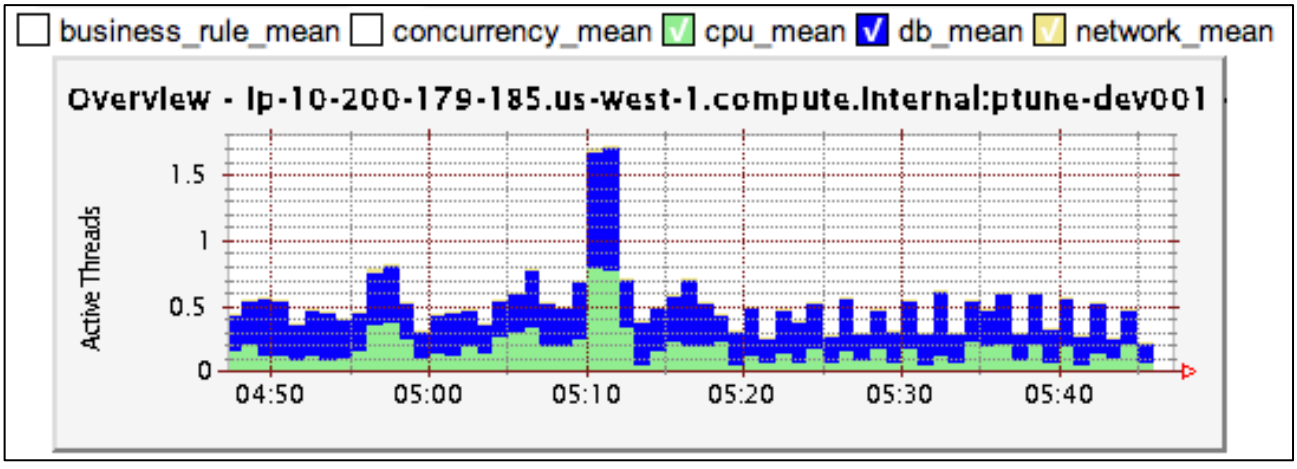


©2014 ServiceNow, Inc. All rights reserved.

10. The **set** should be **Service-now Servlet**, and the CI should be selected (there will be only one). The **Time span** should be **1 hour**. Observe the CPU, database load, and business rule load.

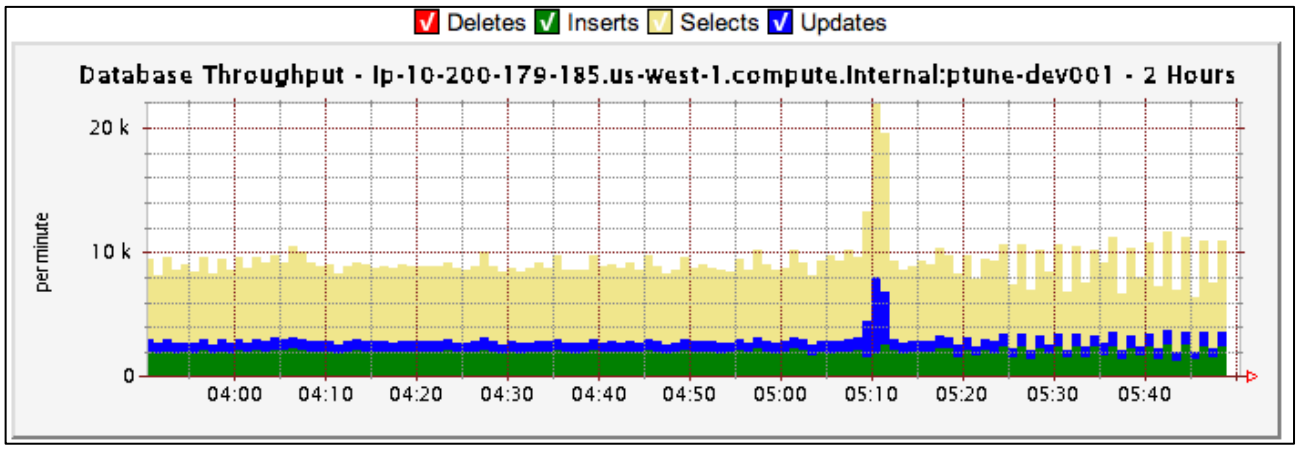


11. Remove **business_rule_mean** and **concurrency_mean** from the Overview graph by clicking their associated swatches.



©2014 ServiceNow, Inc. All rights reserved.

12. Next change the **Time span** to **2 hours**, the **Graph** size to **Extra Large**, and the **Set** to **Database**.



Conclusion

There is definitely a lot of activity on this instance. We have determined from stats.do that there is almost no user activity, but there is background activity. From the performance graphs, we see that there is significant load. We also see from the diagnostics page that there is a decent amount of logging and flurries of event activity. So lets dig a big further!

©2014 ServiceNow, Inc. All rights reserved.

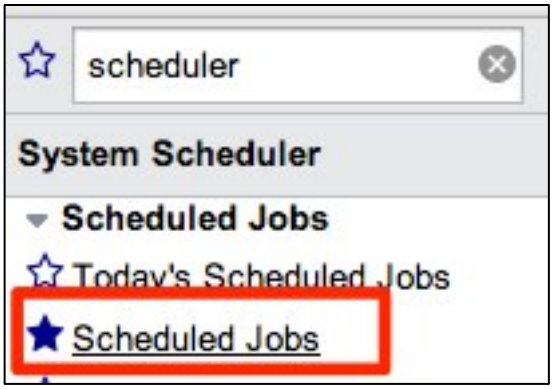
Lab Goal

Track down impacting scheduled jobs using the System Scheduler and the Transaction Log.

Lab 2
Identify impacting scheduled jobs

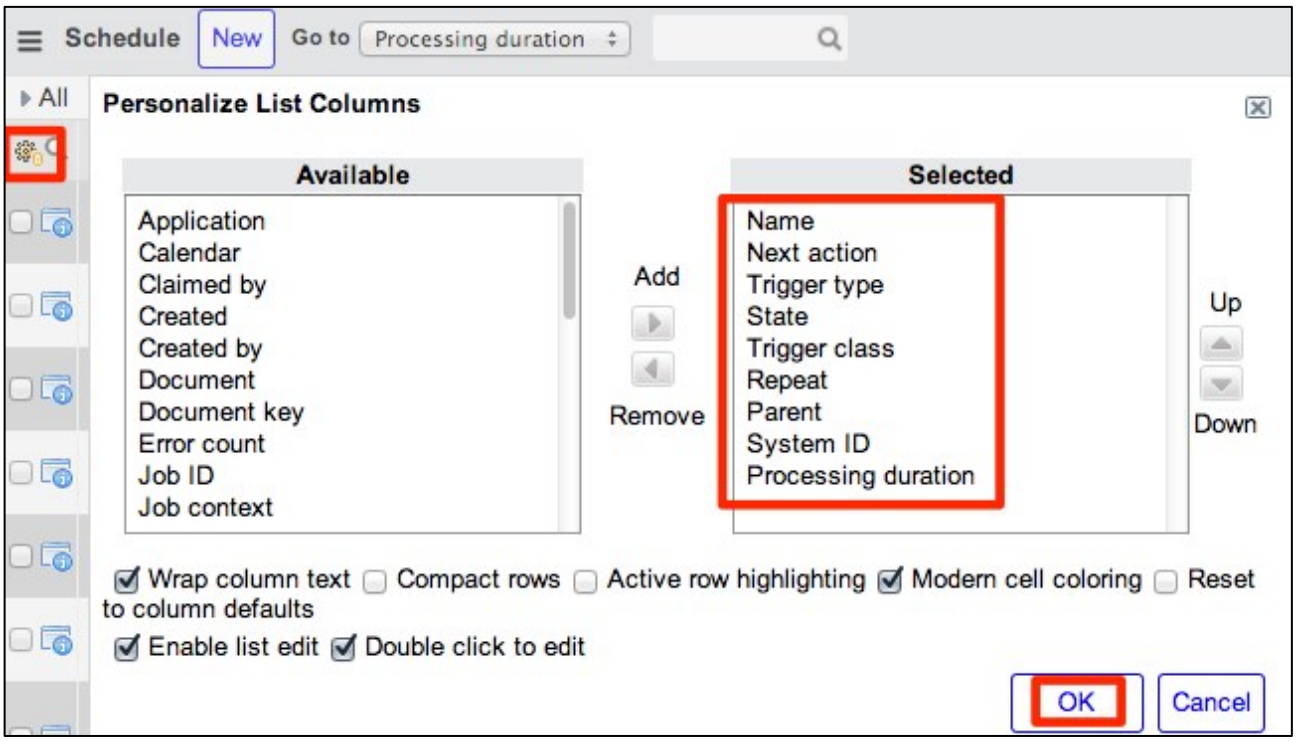
Scheduled Jobs

1. Open **Scheduled Jobs** under **System Scheduler**.

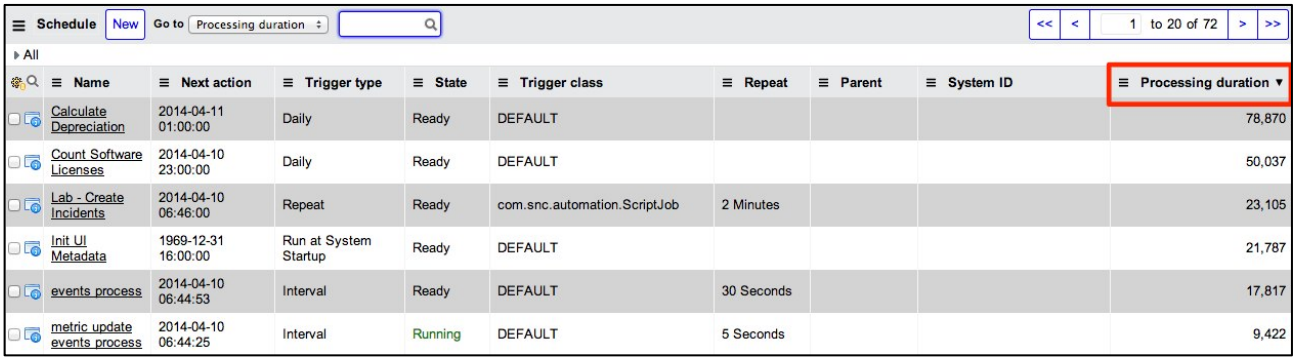


©2014 ServiceNow, Inc. All rights reserved.

2. Use the gear to configure the column list to match the screenshot below.



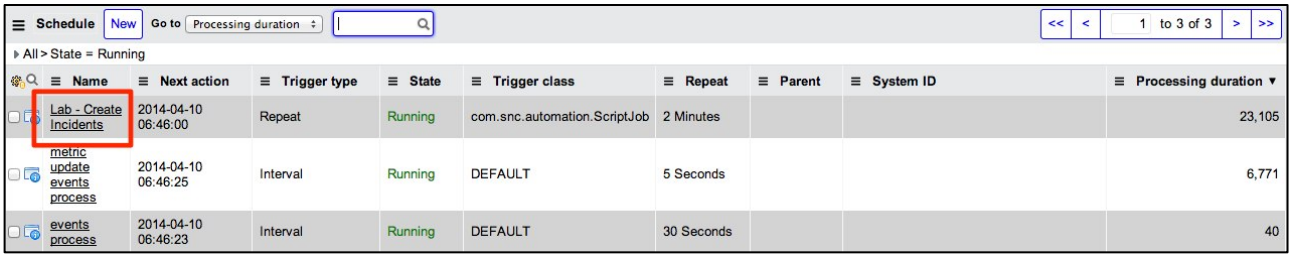
3. Sort descending on **Processing duration**.



Name	Next action	Trigger type	State	Trigger class	Repeat	Parent	System ID	Processing duration
Calculate Depreciation	2014-04-11 01:00:00	Daily	Ready	DEFAULT				78,870
Count Software Licenses	2014-04-10 23:00:00	Daily	Ready	DEFAULT				50,037
Lab - Create Incidents	2014-04-10 06:46:00	Repeat	Ready	com.snc.automation.ScriptJob	2 Minutes			23,105
Init UI Metadata	1969-12-31 16:00:00	Run at System Startup	Ready	DEFAULT				21,787
events process	2014-04-10 06:44:53	Interval	Ready	DEFAULT	30 Seconds			17,817
metric update events process	2014-04-10 06:44:25	Interval	Running	DEFAULT	5 Seconds			9,422

©2014 ServiceNow, Inc. All rights reserved.

4. Add a filter of **State is Running**. Refresh the list a few times until you catch some jobs running. **Lab - Create Incidents** stands out as the most impacting due to its **Processing duration** (the execution time of the previous run) and frequency.



But wait, maybe this was an irregular event that we just happened to catch. Lets see how we can prove that it's a frequent long running job.

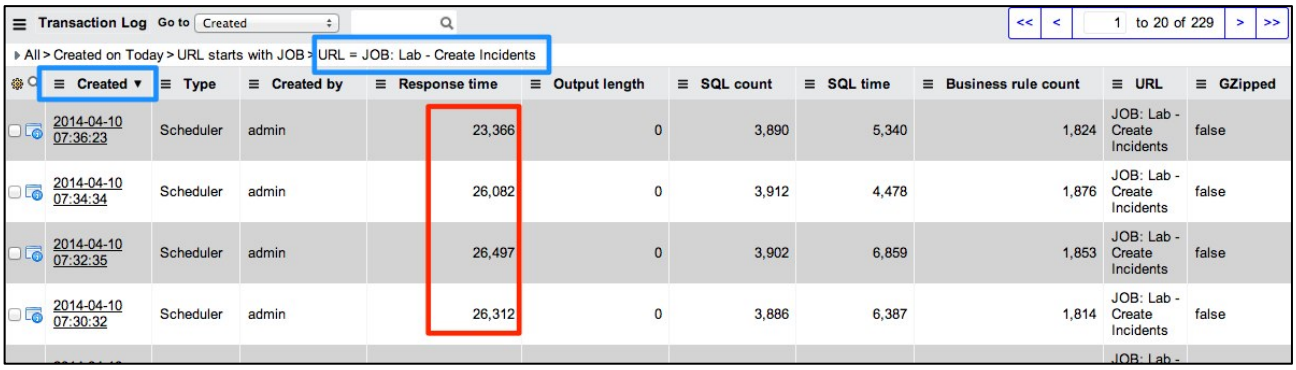
Background Transaction Log

5. Open **Transactions (Background)** under **System Logs**. The default list contains all of today's scheduler jobs that ran longer than 1 second.



©2014 ServiceNow, Inc. All rights reserved.

6. Show matching on a **URL** with the value **JOB: Lab - Create Incidents**, sort Z-A on the **Created** column, and observe the execution times (**Response Time**) and frequency. Also note the additional metrics that are recorded for these transactions. The **Response times** (run time in this case) are consistent.

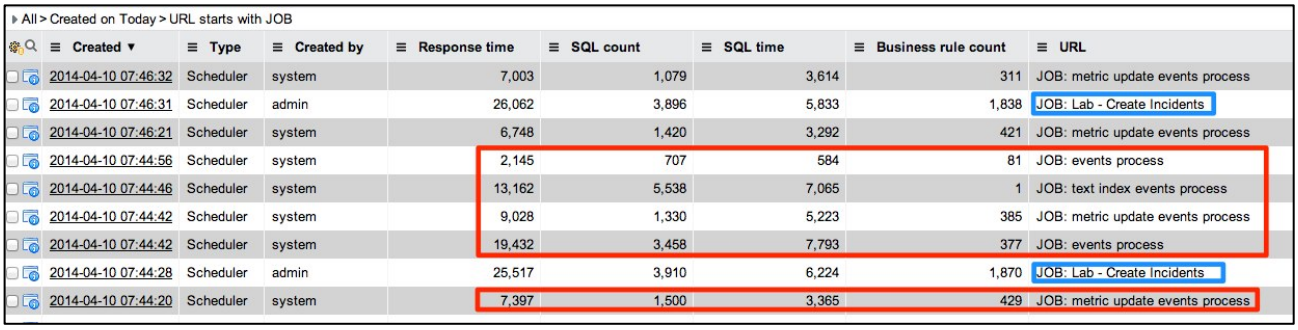


Transaction Log Go to Created << < 1 to 20 of 229 > >>

All > Created on Today > URL starts with JOB > URL = JOB: Lab - Create Incidents

Created	Type	Created by	Response time	Output length	SQL count	SQL time	Business rule count	URL	GZipped
2014-04-10 07:36:23	Scheduler	admin	23,366	0	3,890	5,340	1,824	JOB: Lab - Create Incidents	false
2014-04-10 07:34:34	Scheduler	admin	26,082	0	3,912	4,478	1,876	JOB: Lab - Create Incidents	false
2014-04-10 07:32:35	Scheduler	admin	26,497	0	3,902	6,859	1,853	JOB: Lab - Create Incidents	false
2014-04-10 07:30:32	Scheduler	admin	26,312	0	3,886	6,387	1,814	JOB: Lab - Create Incidents	false

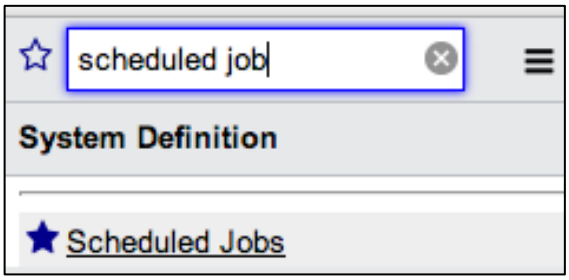
7. Remove the filter for the specific job, and have a look at the other jobs that ran between the most recent run and second most recent run. These are additional background activities that were triggered.



All > Created on Today > URL starts with JOB

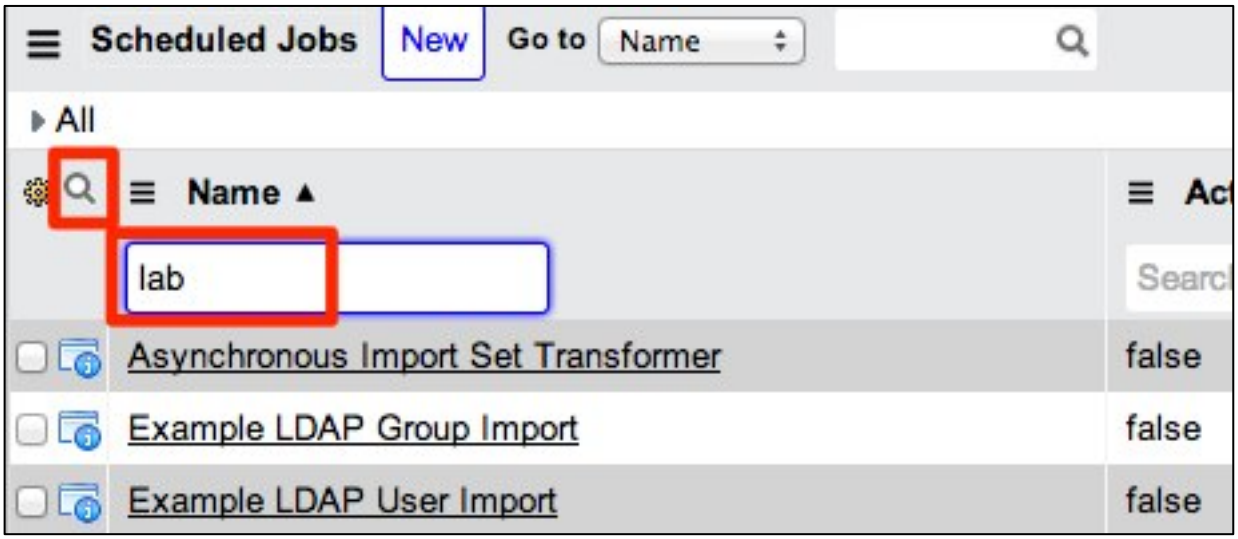
Created	Type	Created by	Response time	SQL count	SQL time	Business rule count	URL
2014-04-10 07:46:32	Scheduler	system	7,003	1,079	3,614	311	JOB: metric update events process
2014-04-10 07:46:31	Scheduler	admin	26,062	3,896	5,833	1,838	JOB: Lab - Create Incidents
2014-04-10 07:46:21	Scheduler	system	6,748	1,420	3,292	421	JOB: metric update events process
2014-04-10 07:44:56	Scheduler	system	2,145	707	584	81	JOB: events process
2014-04-10 07:44:46	Scheduler	system	13,162	5,538	7,065	1	JOB: text index events process
2014-04-10 07:44:42	Scheduler	system	9,028	1,330	5,223	385	JOB: metric update events process
2014-04-10 07:44:42	Scheduler	system	19,432	3,458	7,793	377	JOB: events process
2014-04-10 07:44:28	Scheduler	admin	25,517	3,910	6,224	1,870	JOB: Lab - Create Incidents
2014-04-10 07:44:20	Scheduler	system	7,397	1,500	3,365	429	JOB: metric update events process

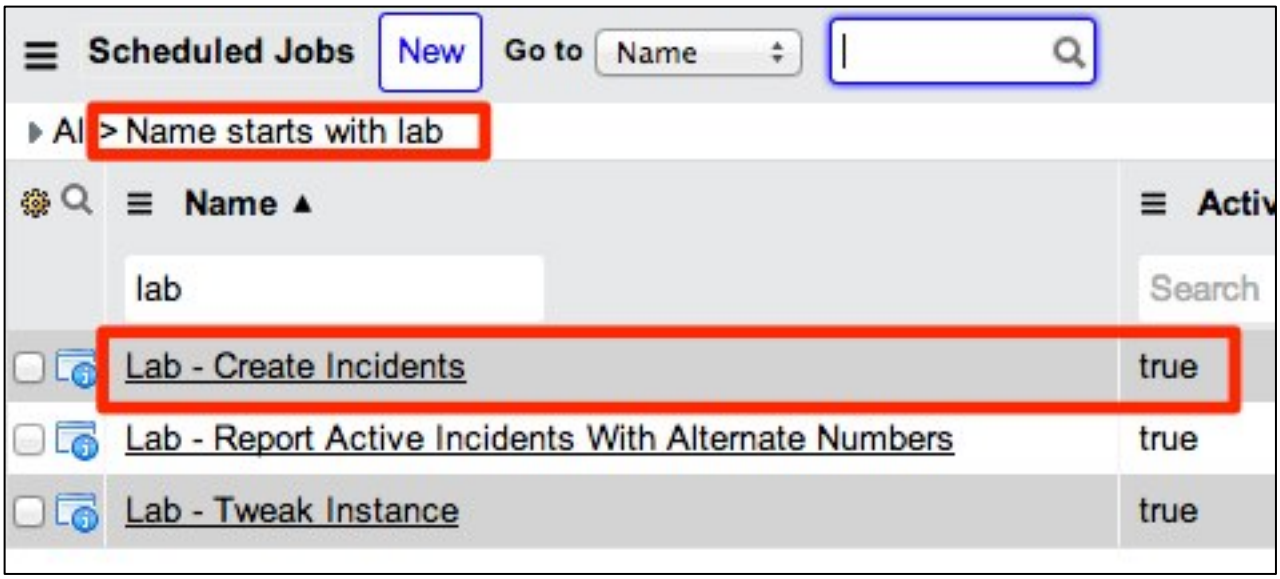
8. Lets go ahead and disable that job. Find **Scheduled Jobs** under **System Definition**.



©2014 ServiceNow, Inc. All rights reserved.

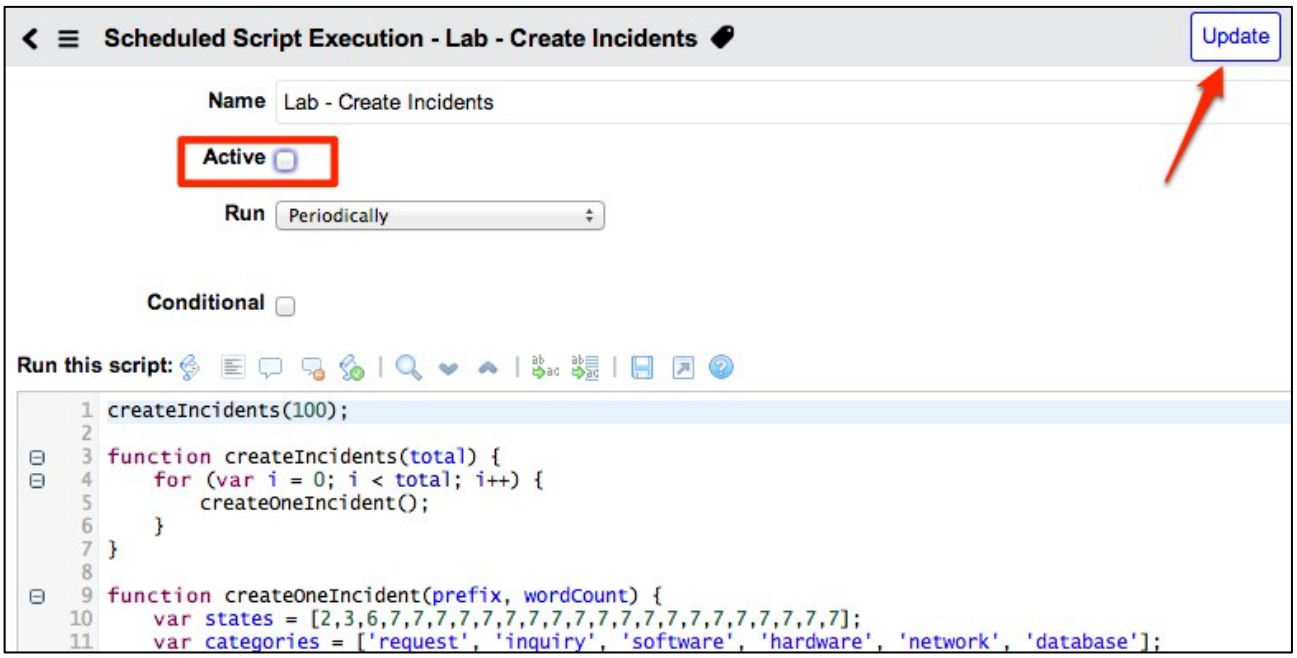
9. Click on the magnifying glass to expand the column search box, and type ***lab*** **[ENTER]** to do a "starts with" search for the job.





©2014 ServiceNow, Inc. All rights reserved.

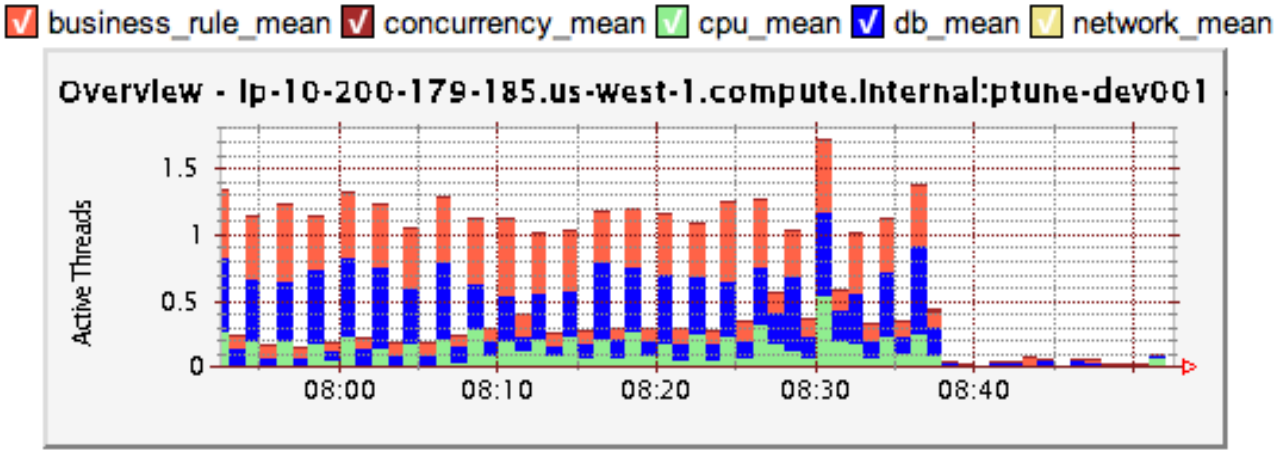
10. Un-check **active**, and **update** the record.



©2014 ServiceNow, Inc. All rights reserved.

Verification

Visit the **ServiceNow Performance** homepage again, and confirm that the activity has ceased.



©2014 ServiceNow, Inc. All rights reserved.

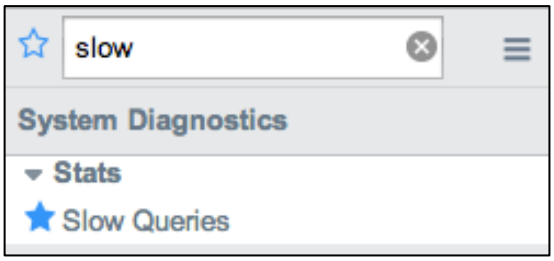
Lab Goal

Use the new query patterns feature to identify sql that may require tuning.

Lab 3
Investigate Slow SQL Queries

Unearth a slow query and investigate it

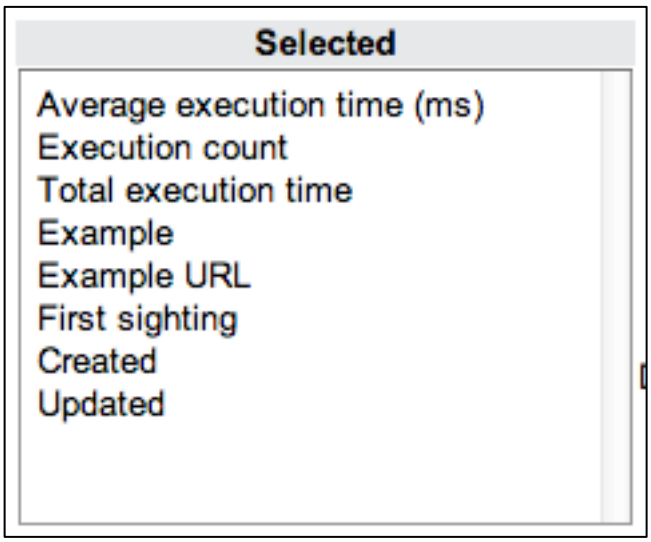
1. Locate the **Slow Queries** module under **System Diagnostics -> Stats**.



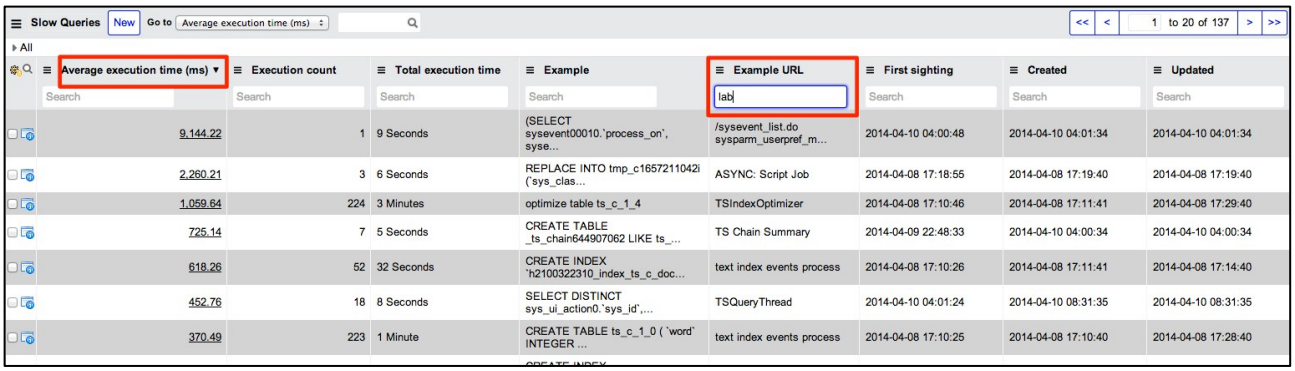
NOTE: The slow SQL statements in previous releases are replaced in Calgary by query patterns.

©2014 ServiceNow, Inc. All rights reserved.

2. Use **Personalize List** to set up the fields to match the screenshot below.

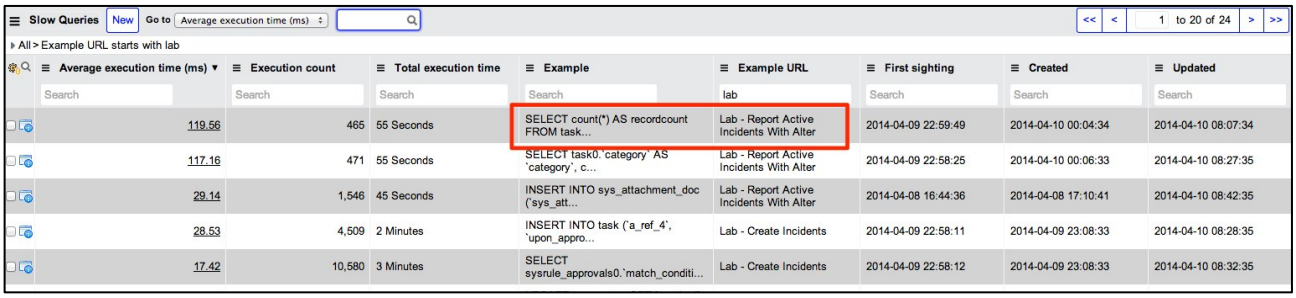


3. Sort Z-A on **Average execution time (ms)**. Filter by **Example URL** *startswith* **lab**.

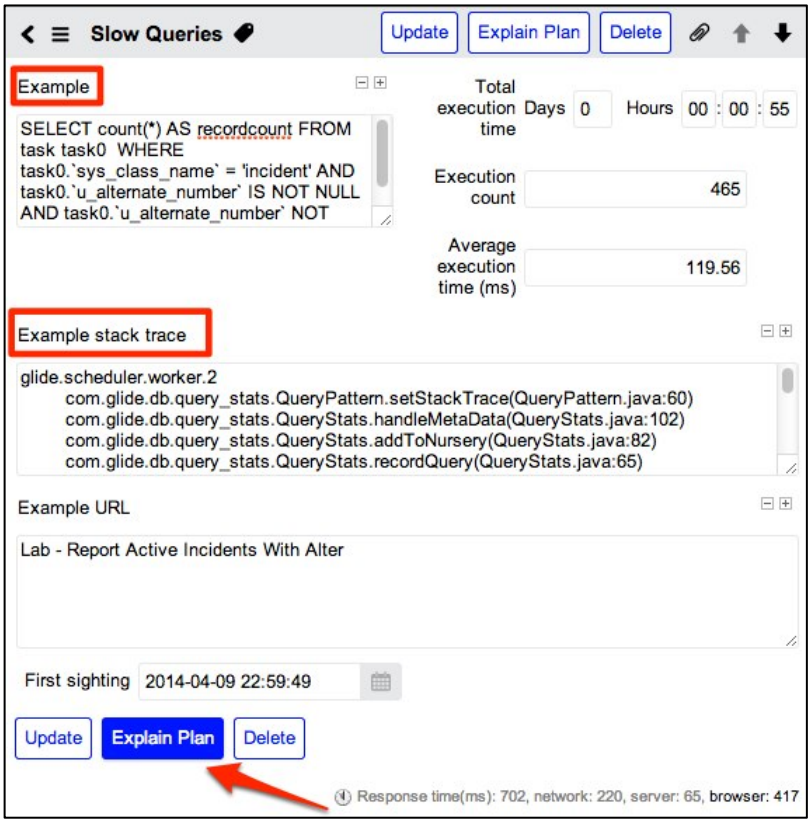


©2014 ServiceNow, Inc. All rights reserved.

4. Look for a **SELECT count(*)** query who's Example URL is **Lab - Report Active Incidents With Alter**. Open the record.

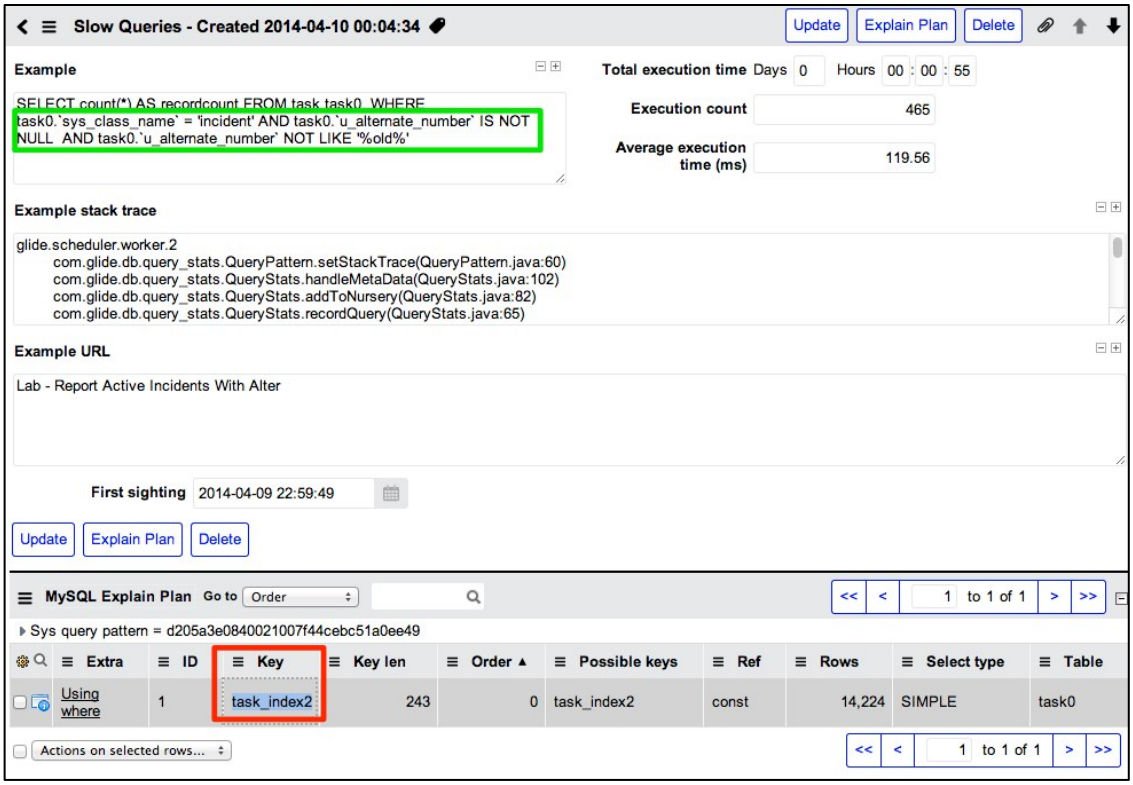


5. In the form view, we get an example query that uses the query pattern. We also get a stack trace for that example so that we have a better idea of where it came from. Click on **Explain Plan**.

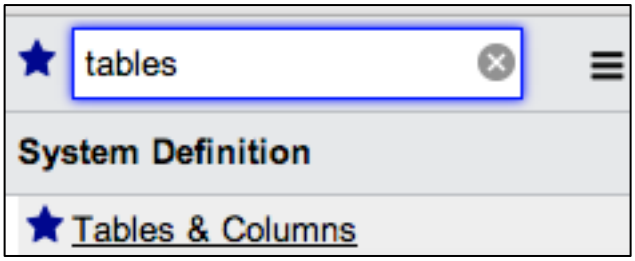


©2014 ServiceNow, Inc. All rights reserved.

6. Note the **Possible keys** and **Key** columns. The term “key” translates to “index” in mysql, and in this example mysql chose its only index option, which was ***task_index2***. That rather generic name doesn’t tell us much, so we need to find out what columns that index contains.

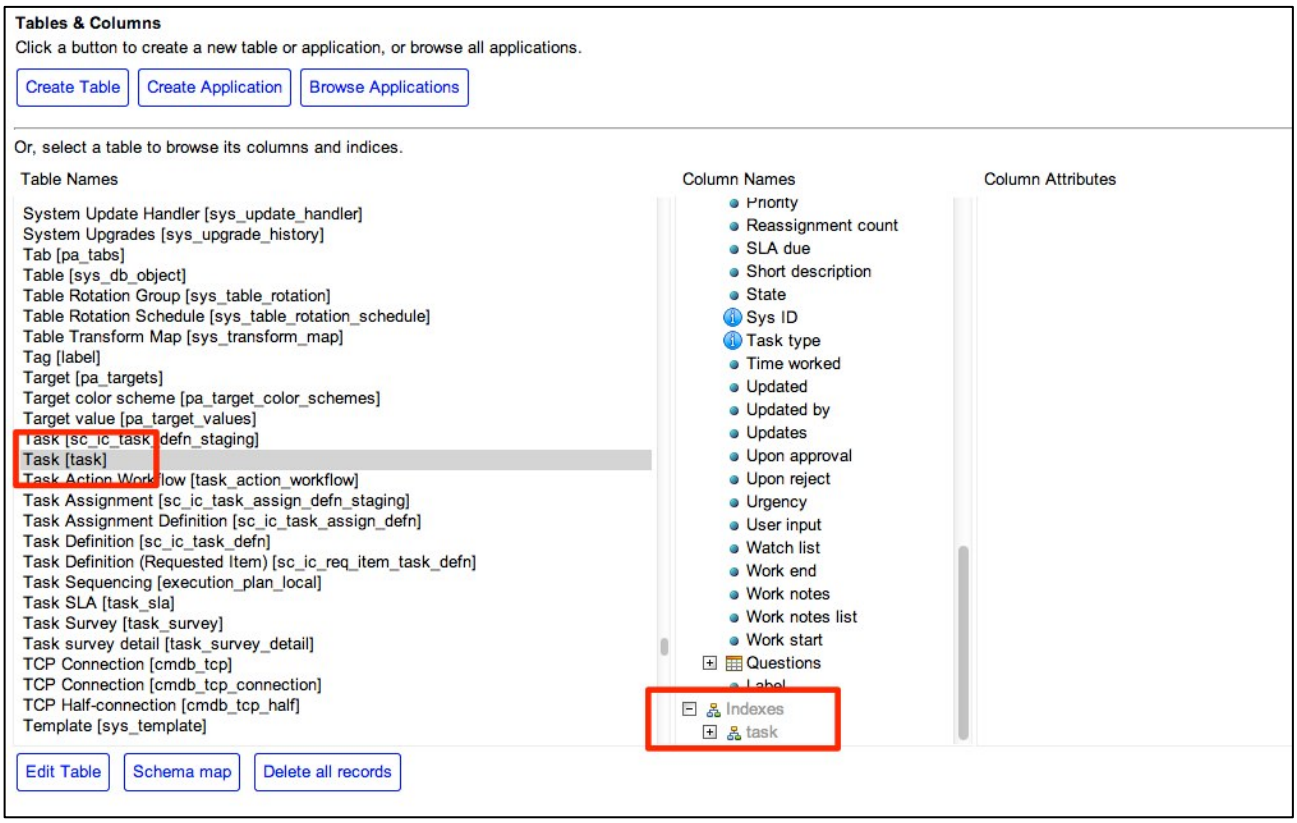


7. Navigate to **Tables & Columns**, and select **Task [task]**.



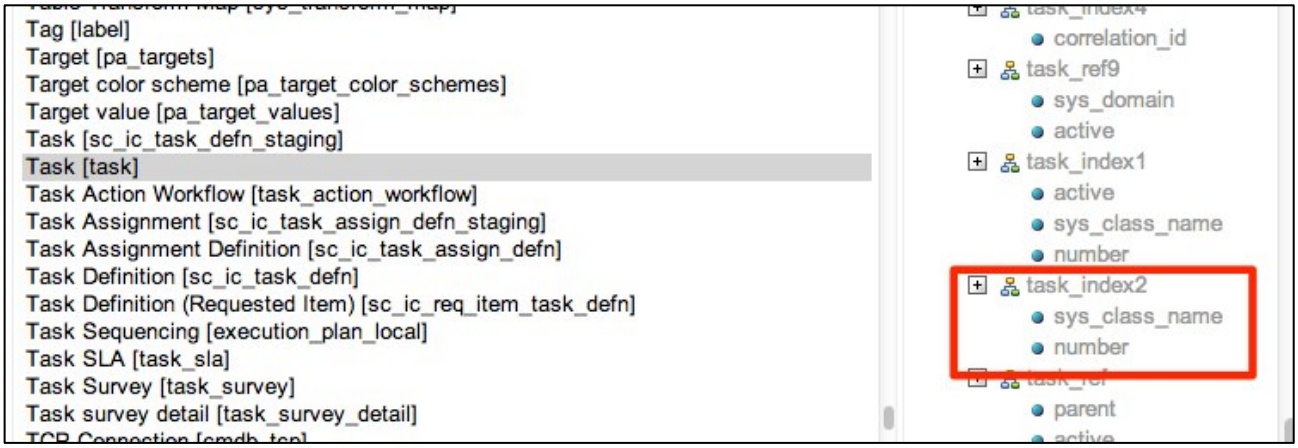
©2014 ServiceNow, Inc. All rights reserved.

8. Expand the indexes to locate **task_index2**.



®2014 ServiceNow, Inc. All rights reserved.

9. Here we see that we do hit an index for sys_class_name, but it doesn't reduce the resulting set of rows very much. Mysql still has to retrieve the full row for each incident record to determine the count. Two ways that we might improve this query would be to add ***active = true*** to the report if the business would allow it (this should make use of task_index1 instead), or add a new index that contains *sys_class_name* and *u_alternate_number*. Adding an index would only be recommended if this was a frequently executed query, as there is additional overhead on insert/update from maintaining the index. If a scheduled report is the only source of this query, it would usually be best to leave it alone.



©2014 ServiceNow, Inc. All rights reserved.

10. Return to the **Slow Queries** module. Sort Z-A on **Total execution time**, and set the criteria like that in the screenshot below. This provides a list of the most impacting select statements that have been detected in the past two hours. Notice that many of the instance initialization statements have been weeded out, as are those that are infrequently used.



Conclusion

The Slow Queries module is a very powerful tool for maintaining platform performance. This tool has made it much more difficult for a bad query to go un-noticed. It should be checked during development, closely watched in UAT, and regularly monitored in production.

©2014 ServiceNow, Inc. All rights reserved.

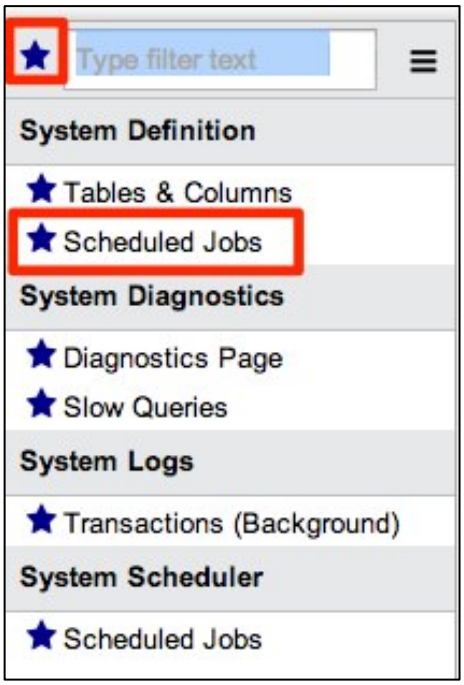
Lab Goal

Use a poorly performing homepage to demonstrate how quotas, render debugging, and render caching can be used to improve performance.

Lab 4
Quotas & Homepage Debugging

Re-activate scheduled job for additional load

1. Click the Star next to the filter to show only the favorites. Click **Scheduled Jobs** under **System Definition**.

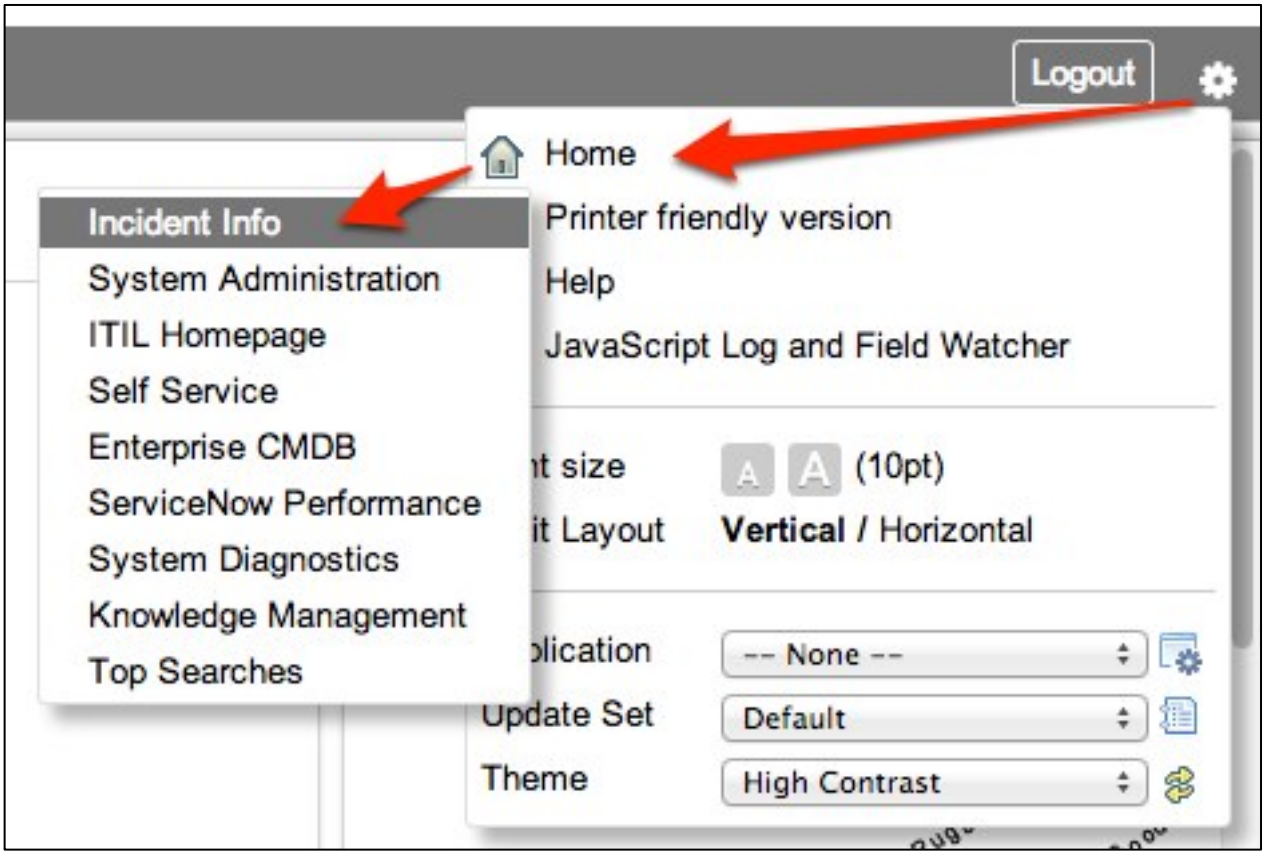


2. Set active = true for the scheduled job **Lab - Create Incidents**.

©2014 ServiceNow, Inc. All rights reserved.

Demonstrate quota & render debugger behavior

3. Visit the **Incident Info** homepage.

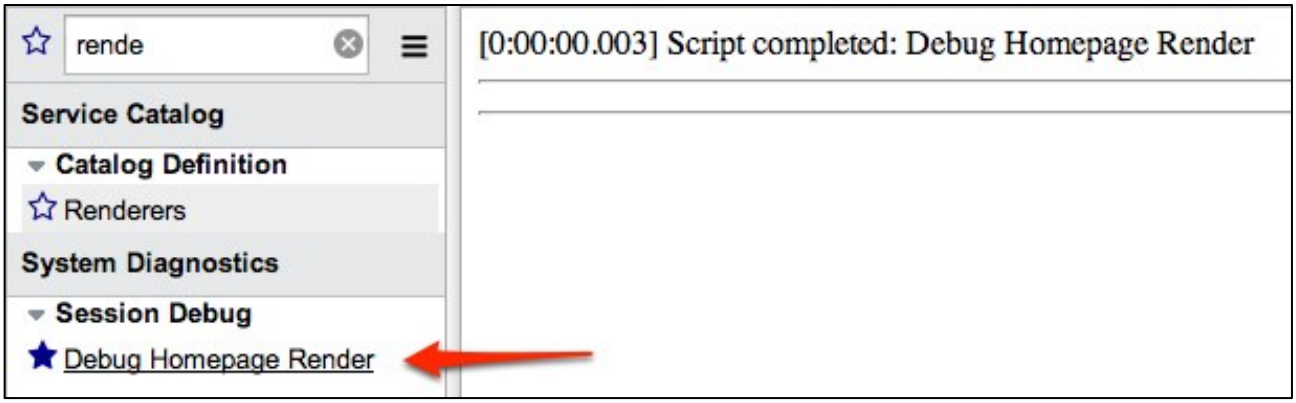


©2014 ServiceNow, Inc. All rights reserved.

4. One or more of the elements on the homepage should have been cancelled due to the homepage widget quota. If not, try refreshing the homepage a few more times until you do.

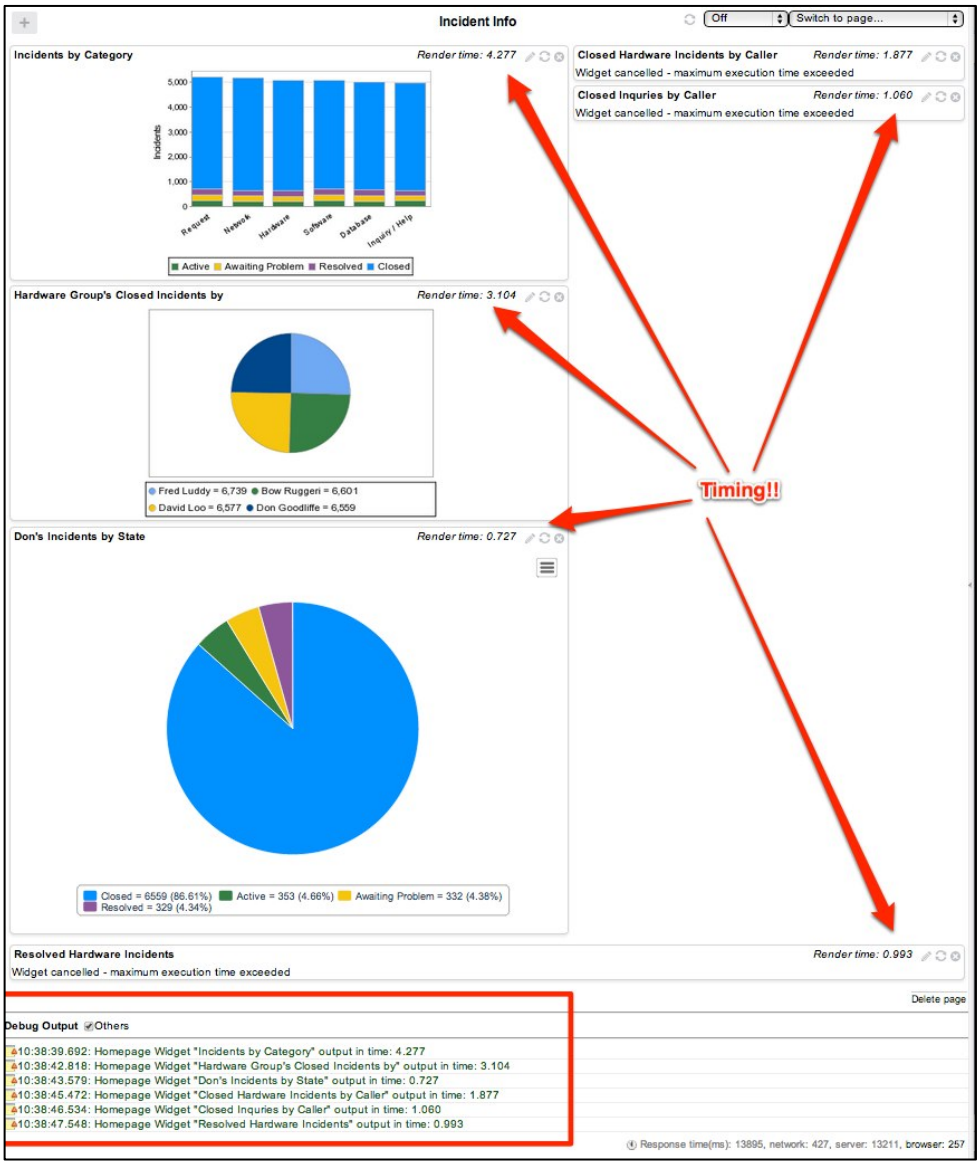


5. Click on **Debug Homepage Render** under **System Diagnostics -> Session Debug**.



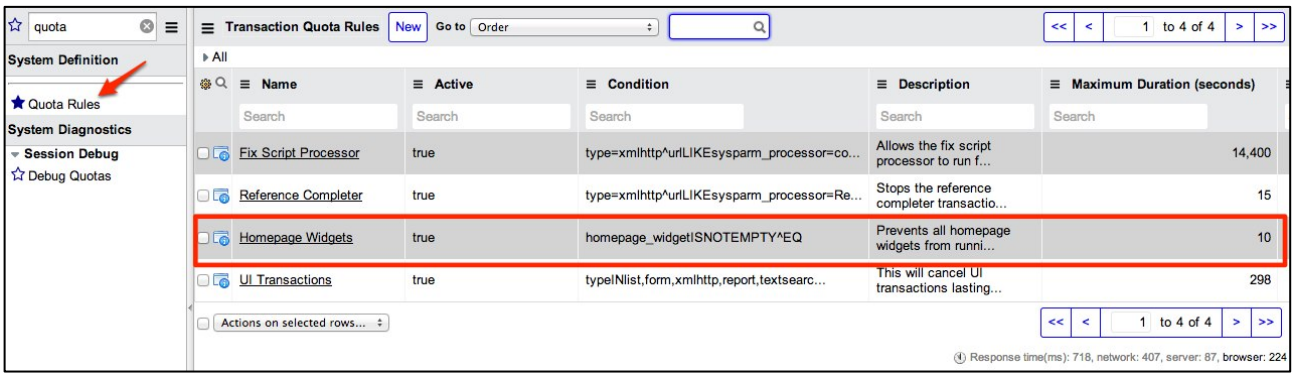
©2014 ServiceNow, Inc. All rights reserved.

6. Navigate to the **Incident Info** home page again. Some elements still cancelled, but this time we have timing info.



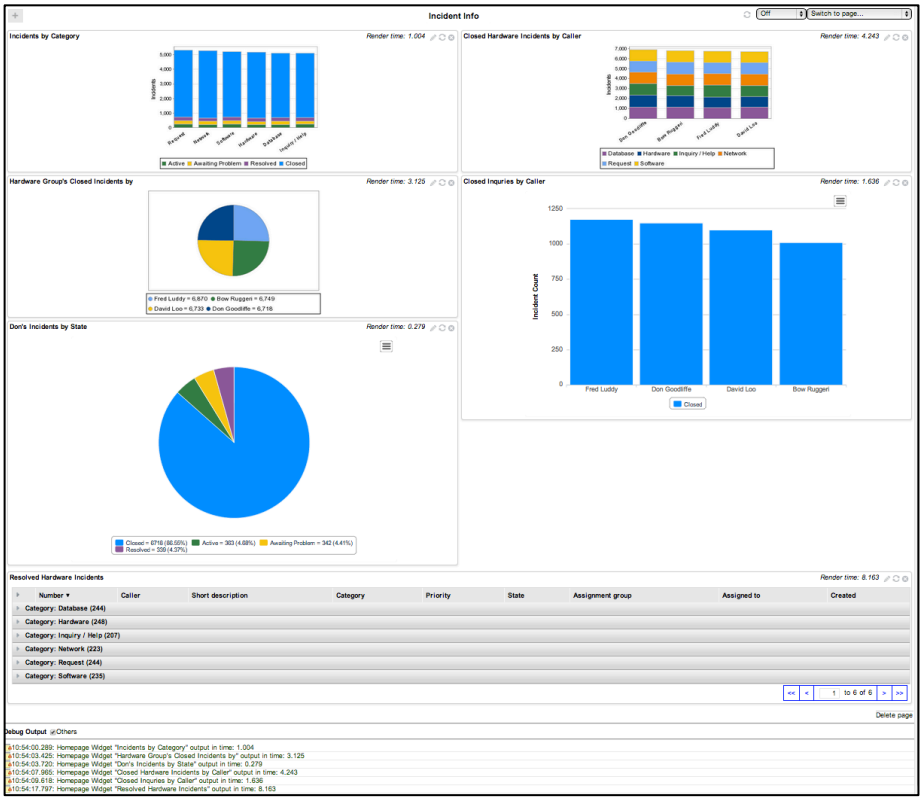
®2014 ServiceNow, Inc. All rights reserved.

7. Navigate to **Quota Rules**, and observe the **Homepage Widgets**' value for **Maximum Duration**.



8. Increase the quota back to the default value of 30.

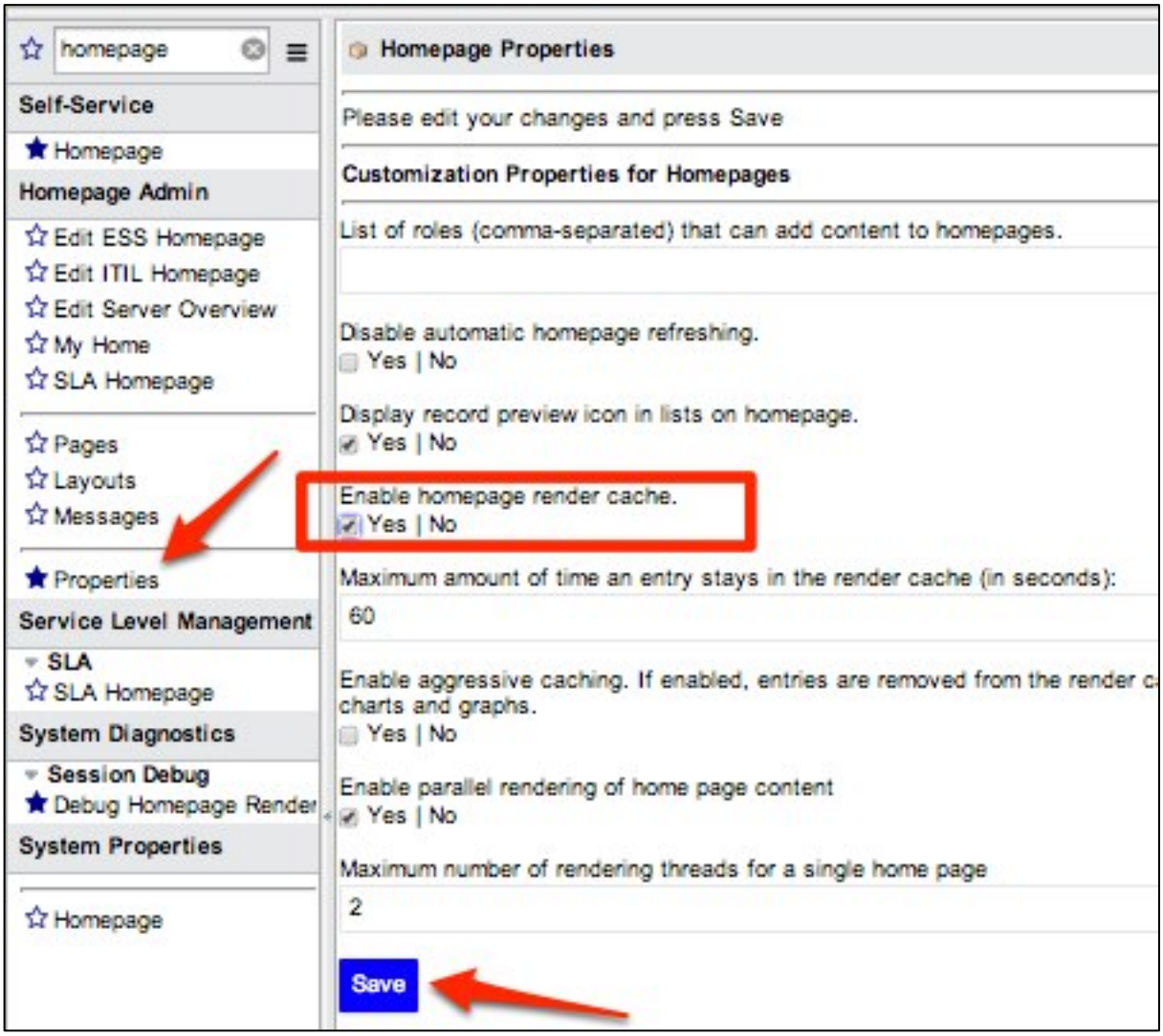
9. Navigate back to the **Incident Info** home page. This time they all loaded!



©2014 ServiceNow, Inc. All rights reserved.

Test homepage render caching

10. You may have noticed that every time you refresh your home page, the graphs are regenerated. This is because caching was disabled for testing. Lets turn it back on. Click on **Properties** under **Homepage Admin**, check ***Enable homepage render cache***, and **save.**



11. Navigate back to the **Incident Info** home page, click the refresh icon for the homepage after it loads

12. Observe the time to refresh with caching now on. It should be immediate.

©2014 ServiceNow, Inc. All rights reserved.

Takeaways

- Quotas can help protect the overall instance performance by limiting the impact of long-running individual transactions
- Homepage render debugging is a very useful tool for isolating slow reports and widgets
- Disabling render caching is a handy way to do rapid testing (DON'T FORGET TO TURN IT BACK ON)

©2014 ServiceNow, Inc. All rights reserved.

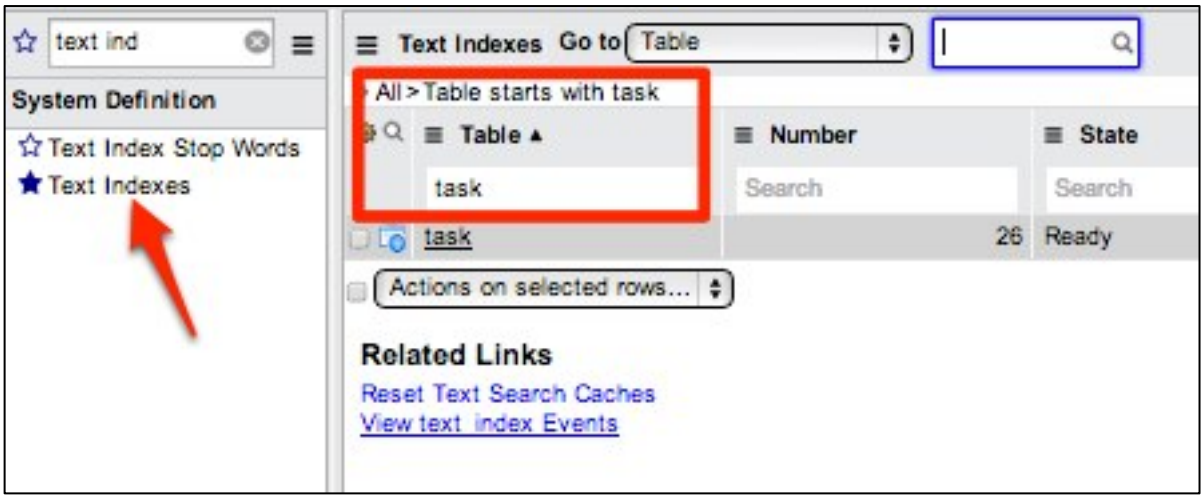
Lab Goal

Optimize text index performance by removing over-populated words.

Lab 5
Text Index Auto-stop Thresholds

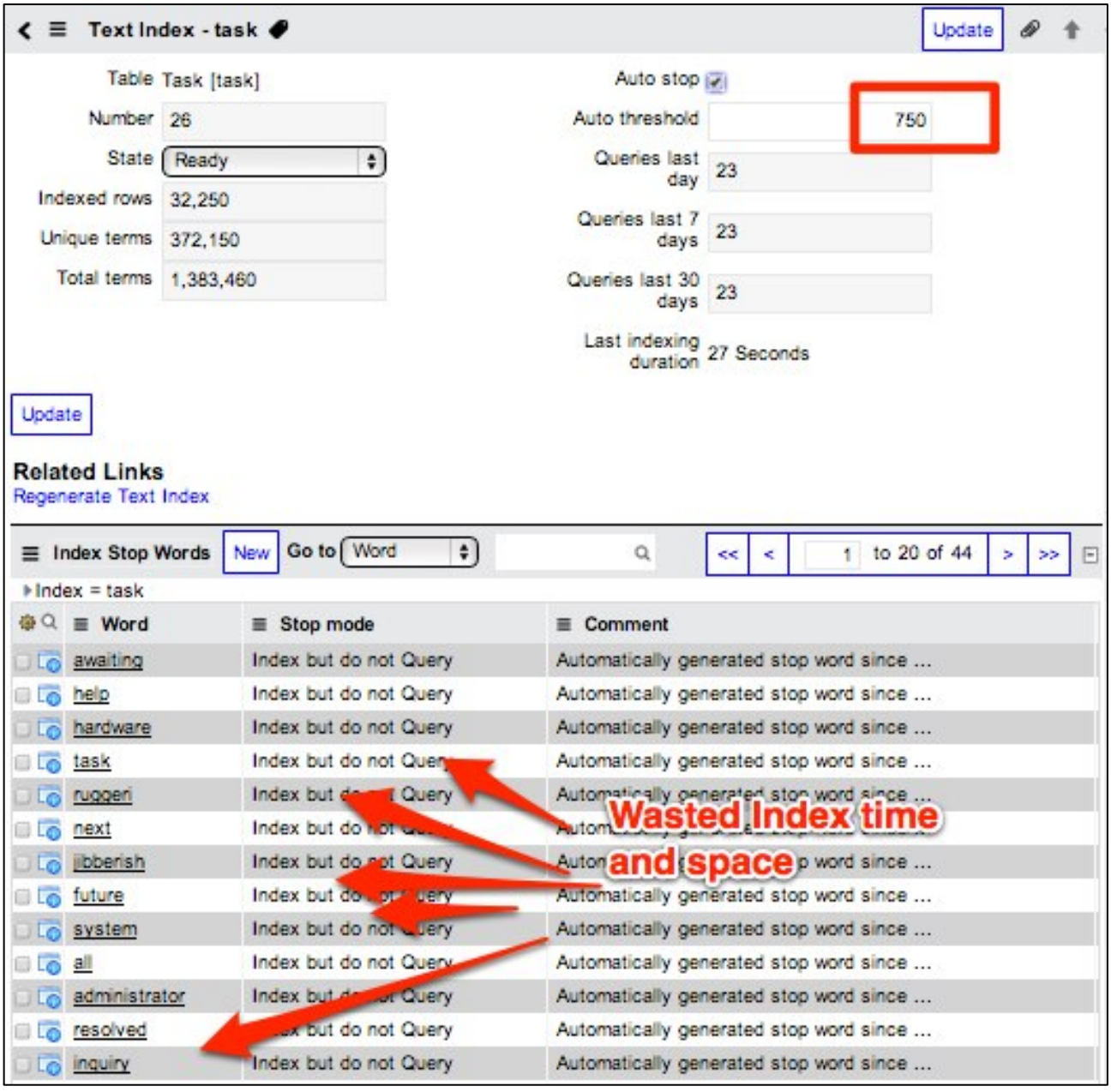
Regenerate the Task table index without auto-stop words

1. Click on the **Text Indexes** module under **System Definition**. Find the ***task*** table index definition, and it.



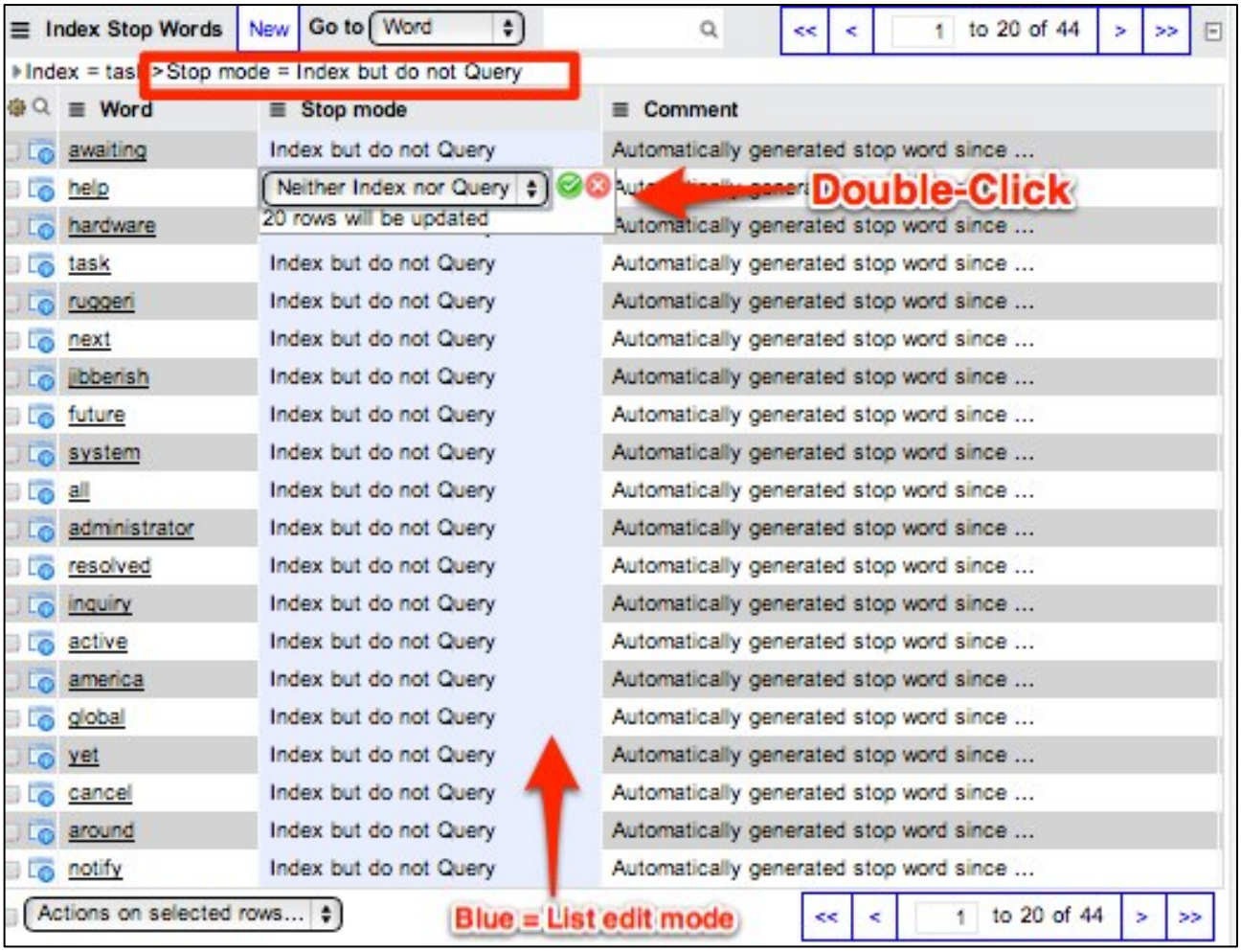
©2014 ServiceNow, Inc. All rights reserved.

2. Note the **Auto threshold** value. This is the number of occurrences of the word before the index will start to ignore it. The default value is 250,000, but we changed it to 750 so that auto-stop words could be generated in a reasonable amount of time. Note that we have several auto-stop words in the related list.

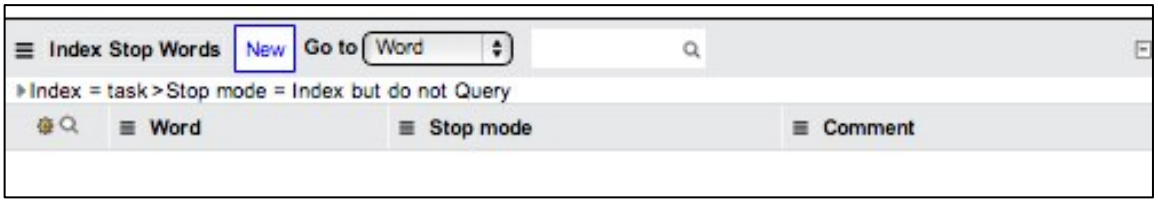


©2014 ServiceNow, Inc. All rights reserved.

3. In the **Index Stop Words** related list, show matching on ***Index but do not Query***. Next, list edit (shift+click-drag-unclick, then double-click) the **Stop mode** for all of the words in the list, and change them to ***Neither Index nor Query***. *Refresh the list, and repeat until they are all changed (three edits for me).*

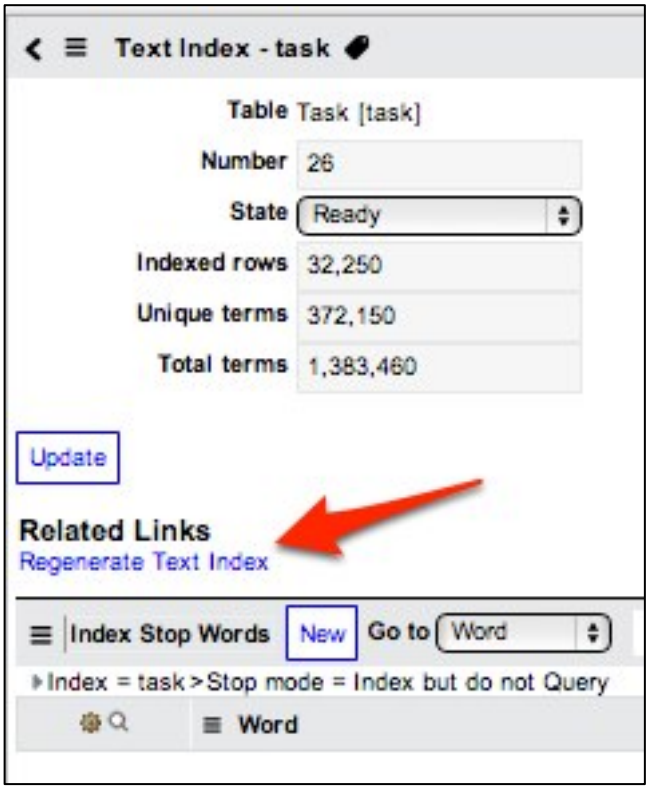


After completed:



©2014 ServiceNow, Inc. All rights reserved.

4. Click **Regenerate Text Index** to rebuild the text indexes from scratch, eliminating them from the index tables. **In a production environment, this would be done during the lowest period of user activity, as the index will be incomplete for the duration of the rebuild, which could be several hours. If upgrading the indexes are automatically regenerated. Simply change the Stop modes prior to the upgrade, and they will take effect once the upgrade finishes the rebuild.**



©2014 ServiceNow, Inc. All rights reserved.

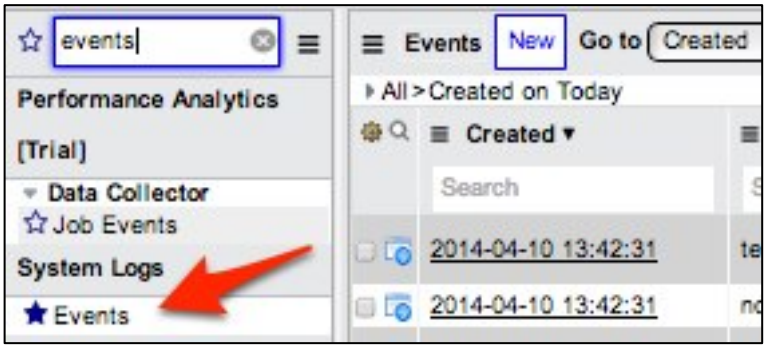
Lab Goal

Understand how to monitor the status of the event queues. The Events table is home to three separate queues, each processed by a different scheduled job. The queues are distinguished by the value in the Queue column for each record. The options are text_index, metrick_update, and everything else will have no queue defined.

Lab 6
Monitor Event Queue

Make sense of the Events table

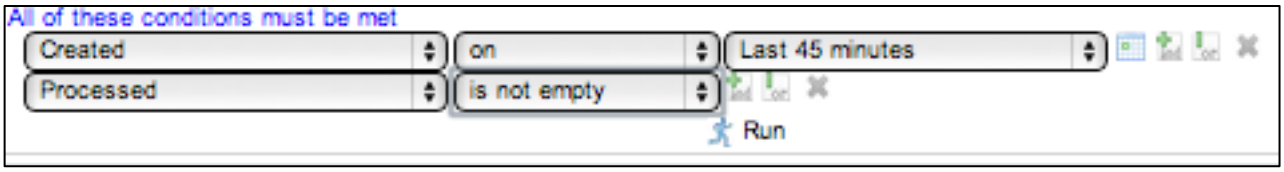
1. Navigate to the **Event Log** module under **System Logs**.

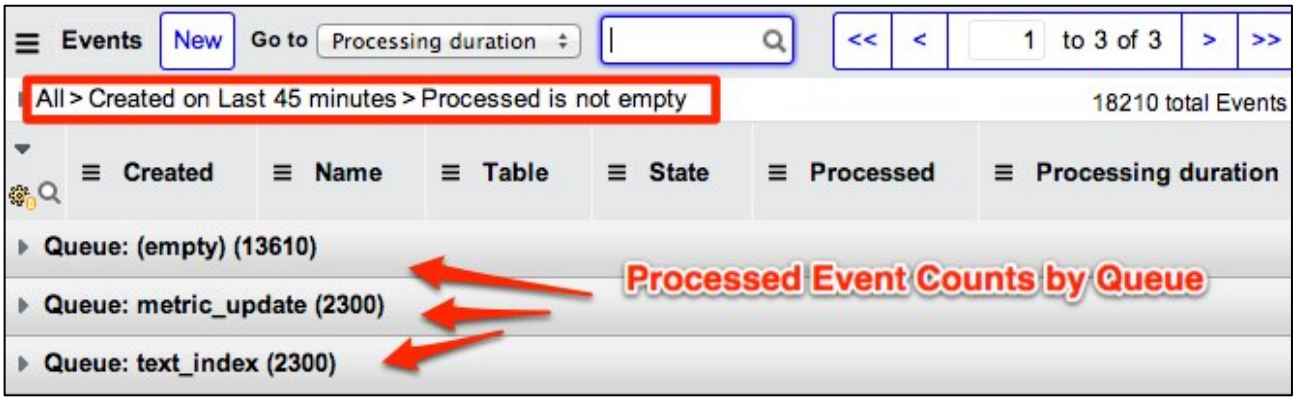


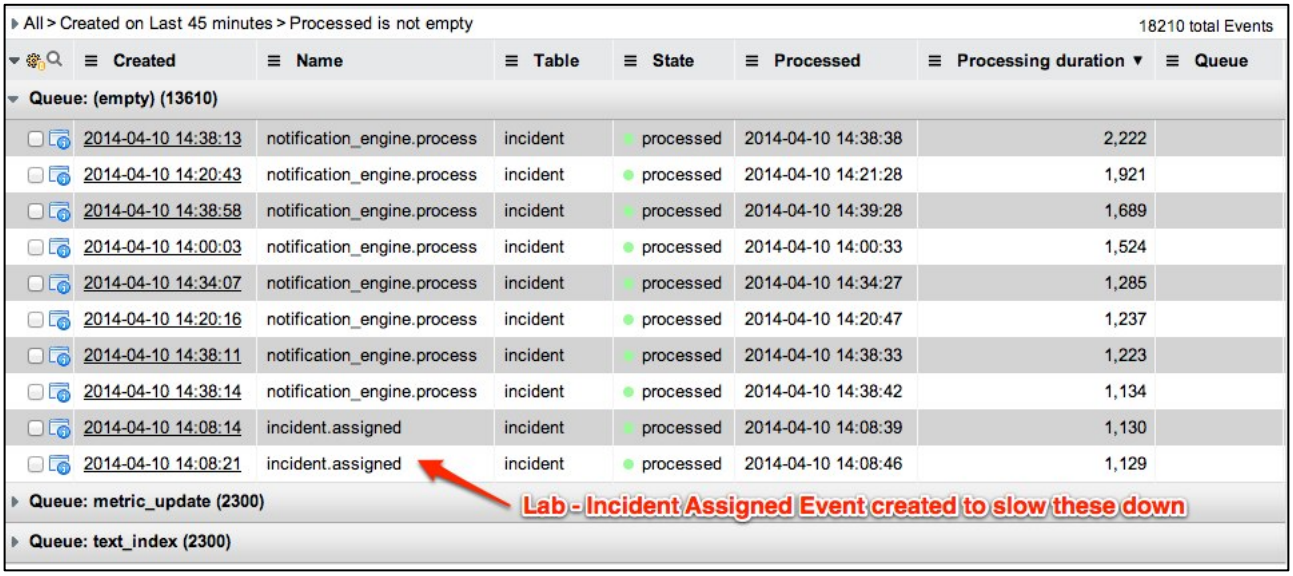
2. Add the ***State*** field to the personalized list. Sort Z-A on ***Processing Duration***.

©2014 ServiceNow, Inc. All rights reserved.

3. Show the number of completed events in the last 45 minutes for each queue, with specific events sorted by slowest. Set the criteria as depicted in the screenshot. Then right-click on the Queue column header, and select ***Group By Queue***.



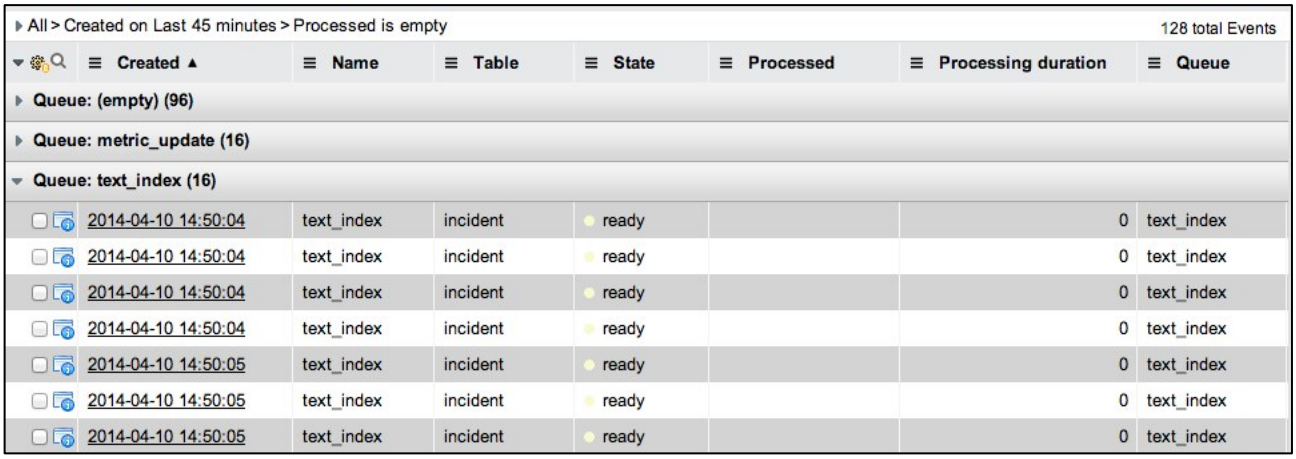




Tip: If the list is refreshed by clicking the filter, the grouping will go away. To retain the grouping, either right-click the table header and choose ***Refresh List***, or view the list directly rather than within the framework, and a browser refresh will retain it.

©2014 ServiceNow, Inc. All rights reserved.

4. Next we want to see what events haven't been processed, and hopefully how far behind we are. To do this, we change the filter slightly to ***Processed is empty***, and sort A-Z on **Created**. In the below example, we are not behind. The processors are just in-between runs. If we were behind, the ***Created*** time of the first record in the list would be further into the past than a few seconds.



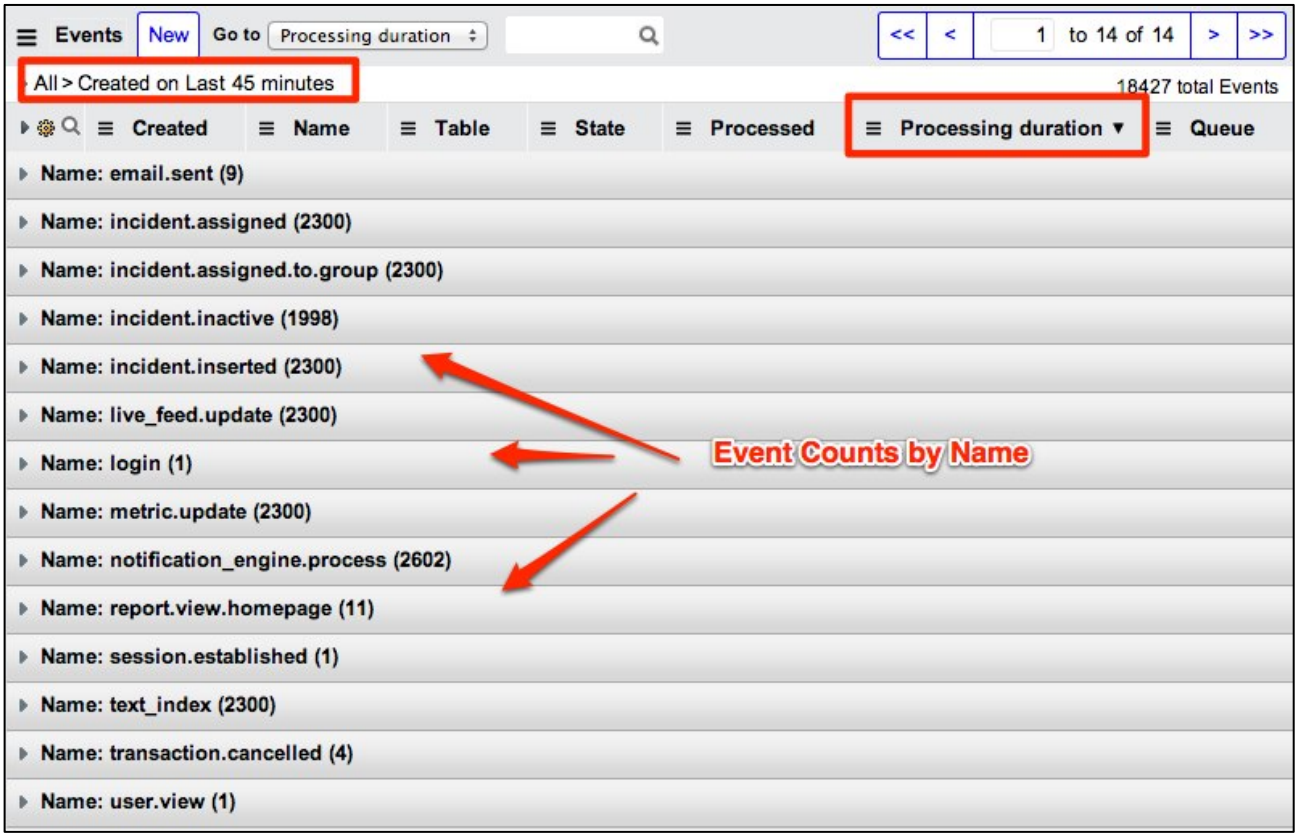
All > Created on Last 45 minutes > Processed is empty — 128 total Events

Created ▲	Name	Table	State	Processed	Processing duration	Queue
Queue: (empty) (96)						
Queue: metric_update (16)						
Queue: text_index (16)						
2014-04-10 14:50:04	text_index	incident	ready		0	text_index
2014-04-10 14:50:04	text_index	incident	ready		0	text_index
2014-04-10 14:50:04	text_index	incident	ready		0	text_index
2014-04-10 14:50:04	text_index	incident	ready		0	text_index
2014-04-10 14:50:05	text_index	incident	ready		0	text_index
2014-04-10 14:50:05	text_index	incident	ready		0	text_index
2014-04-10 14:50:05	text_index	incident	ready		0	text_index

Note: If a queue is further behind than the filter that was used, the oldest record will be at most, the time in the filter. Maybe obvious, but worth mentioning.

©2014 ServiceNow, Inc. All rights reserved.

5. Get a count of events for each event name for the last 45 minutes.



Other bits

- If an event queue is unable to keep up with the influx of events, confirm that events for that queue are actually getting processed, by looking for changes ***Created on today***, ***Processed is not empty***, and ordered by ***Updated***.
- Support is well versed in troubleshooting events, and should be contacted with queue lag issues.
- The Events table (sys_event) is a rotated table. If not queried with ***Created*** in the criteria (preferably 24 hours or less), madness will likely ensue as you wait for all of the rotations to look for your data, and union the results.
- The wiki has a lot of additional detail about how events are created and used. http://lmsnwtfy.com/

©2014 ServiceNow, Inc. All rights reserved.