Mindtree

Performance Test Report for Execution of Consumed Batch Job for 90% consumed status

Date: 13 June 2019

Author: Anand Babaleshwar

Summary

This report presents the observations and findings of the Consumed batch job execution in which 27000 consumed status (90% of the peak enrollments and 10% cancelled appointments as per the workload model document peak enrollments is 30000)

The objective of this batch job execution was to observe and record the behavior of the batch job for consuming the 27000 consumed status appointments.

Proprietary and Confidential © MindTree Limited

Below are the scenario details:

Batch job execution Name:

Consumed batch job execution in which 27000 consumed status (90% of the peak enrollments and 10% cancelled appointments as per the workload model document peak enrollments is 30000)

Steps:

1. Creating Pre Registrations test data of 27,000 with booked status (which is 90% of peak enrollments per day) using Jmeter tool
2. Execution of calling data sync API script with 27,000 Pre RegIds input data and each time 50 preRegIds will sent for processing
3. Once all the 27000 Pre-Regids passed without any errors and execute the consumed batch job
4. Calculate the batch job execution time
5. Checking the functionality of batch job
6. Consumed batch job failed and below jira ticket is raised
   https://mosipid.atlassian.net/browse/MOS-25723

Proprietary and Confidential © MindTree Limited

Test Environment

	Common proxy server (NGINX)	(Kubernets cluster) apache Tomcat 8.5.31	DB Postgress SQL 10.2
Number Of nodes	1	4	1
RAM	4 GB	112 GB	16GB
PROCESSOR	2 cores	16 core	4 cores

Before running expiry batch job below are the details:

- Total number of Booked appointments =30000
- Total number booking appointments changed status from “booked” to “consumed” = 27000(90% of Peak enrollments)

Batch job execution status:

instance_id	Create_Time	Start_Time	End_Time	Execution Time	Status	Exit code
1087	**2019-06-13 08:53:24.149**	**2019-06-13 08:53:24.16**	**2019-06-13 12:15:55.207**	**03:22:31**	**UNKNOWN**	**FAILED**

Proprietary and Confidential © MindTree Limited

Exit message:

org.springframework.transaction.TransactionSystemException: Could not roll back JPA transaction; nested exception is org.hibernate.TransactionException: Unable to rollback against JDBC Connection

at org.springframework.orm.jpa.JpaTransactionManager.doRollback(JpaTransactionManager.java:567)

at org.springframework.transaction.support.AbstractPlatformTransactionManager.processRollback(AbstractPlatformTransactionManager.java:838)

at org.springframework.transaction.support.AbstractPlatformTransactionManager.rollback(AbstractPlatformTransactionManager.java:812)

at org.springframework.transaction.support.TransactionTemplate.rollbackOnException(TransactionTemplate.java:168)

at org.springframework.transaction.support.TransactionTemplate.execute(TransactionTemplate.java:144)

at org.springframework.batch.core.step.tasklet.TaskletStep$2.doInChunkContext(TaskletStep.java:272)

at org.springframework.batch.core.scope.context.StepContextRepeatCallback.doInIteration(StepContextRepeatCallback.java:81)

at org.springframework.batch.repeat.support.RepeatTemplate.getNextResult(RepeatTemplate.java:375)

at org.springframework.batch.repeat.support.RepeatTemplate.executeInternal(RepeatTemplate.java:215)

at org.springframework.batch.repeat.support.RepeatTemplate.iterate(RepeatTemplate.java:145)

at org.springframework.batch.core.step.tasklet.TaskletStep.doExecute(TaskletStep.java:257)

Proprietary and Confidential © MindTree Limited

at org.springframework.batch.core.step.AbstractStep.execute(AbstractStep.java:200)

at org.springframework.batch.core.job.SimpleStepHandler.handleStep(SimpleStepHandler.java:148)

at org.springframework.batch.core.job.AbstractJob.handleStep(AbstractJob.java:394)

at org.springframework.batch.core.job.SimpleJob.doExecute(SimpleJob.java:135)

at org.springframework.batch.core.job.AbstractJob.execute(AbstractJob.java:308)

at org.springframework.batch.core.launch.support.SimpleJobLauncher$1.run(SimpleJobLauncher.java:141)

at org.springframework.core.task.SyncTaskExecutor.execute(SyncTaskExecutor.java:50)

at org.springframework.batch.core.launch.support.SimpleJobLauncher.run(SimpleJobLauncher.java:134)

at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)

at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)

at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)

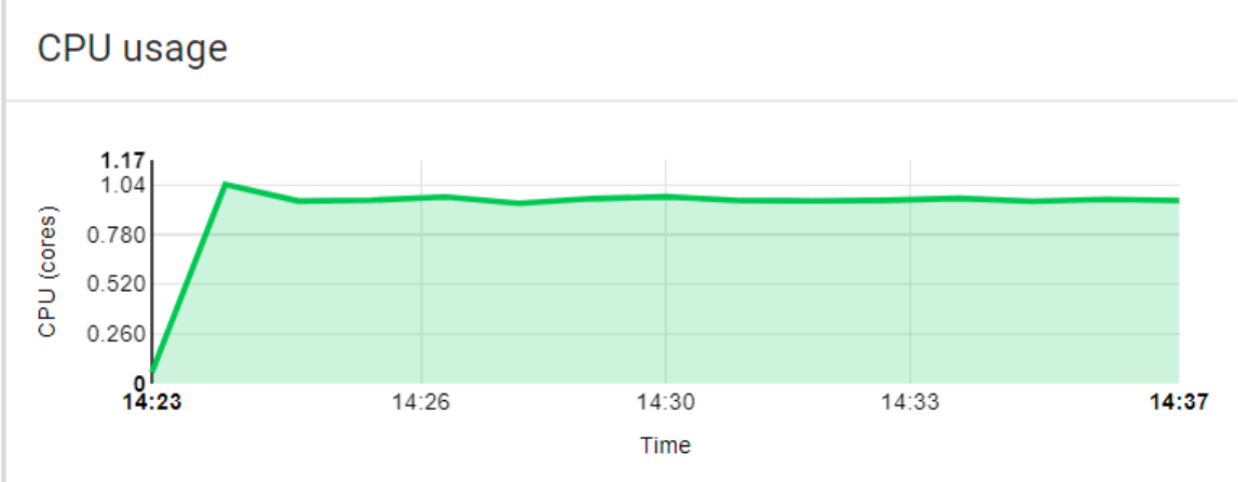
at java.lang.reflect.Method.invoke(Method.java:498)

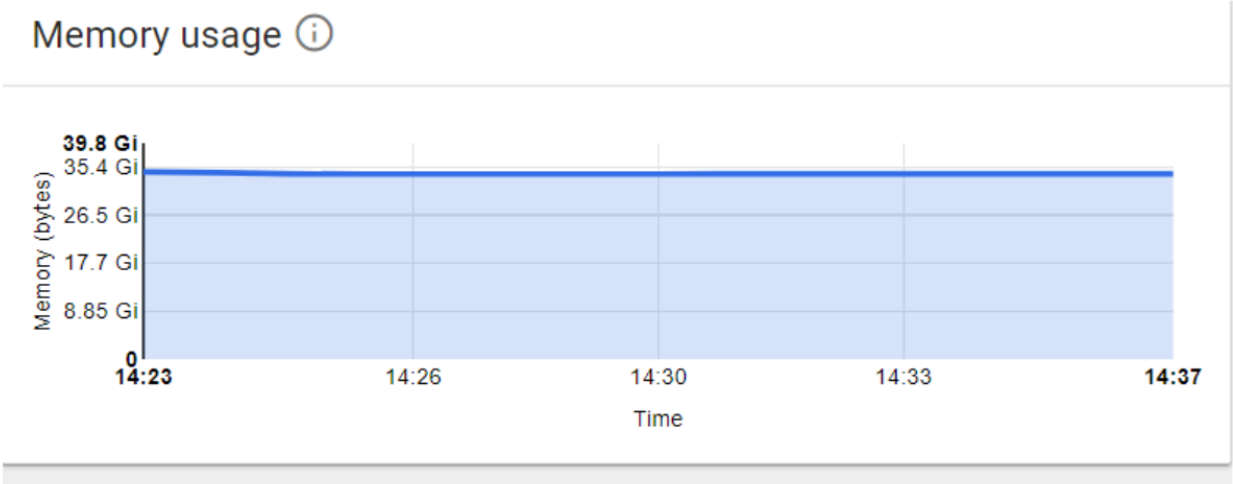
at org.springframework.aop.support.AopUtils.invokeJoinpointUsingReflection(AopUtils.java:343)

at org.sprin

Proprietary and Confidential © MindTree Limited

CPU and memory utilization





Proprietary and Confidential © MindTree Limited

After running expiry batch job execution:

- Total number of Booked appointments =3000
- Total number of records in the prereg.applicant_demographic_consumed table =27000
- Total number of records in the prereg.applicant_document_consumed table =27000

Conclusion and Next Steps:

Expiry batch job this scenario taken **03:22:31 sec** to complete execution and consumed appointments are matching as per the scenario ,So we have completed testing batch jobs as per performance testing approach with 90% consumed status and 10% failed appointments. Observed failed status of batch job in DB and raised jira ticket https://mosipid.atlassian.net/browse/MOS-25723 for this and once developers fix the issue we will retest this consumed batch job.

Proprietary and Confidential © MindTree Limited