Database - Feature #2274

improve performance of new database features

03/30/2014 02:34 PM - Eric Faulhaber

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	Performance and Scalability Improvements		
billable:	No	**vendor_id:**	GCD

Description

Subtasks:

Feature # 2275: cache of runtime-converted, dynamic queries and temp-tables	**Closed**
Feature # 2276: create a work queue of PatternEngine instances for runtime conversion t...	**Closed**
Feature # 2551: replace runtime compilation of DMOs with ASM bytecode implementation	**Closed**

Related issues:

Related to Bugs - Bug #2249: PermGen leak due to dynamic conversion at runtime	**Closed**

History

#1 - 03/30/2014 02:45 PM - Eric Faulhaber

Early testing of customer code which makes heavy use of some of the newly developed database features (particularly those involving dynamic, runtime conversion) shows that performance is quite bad. Metrics built into customer code indicate the durations of some work flows are an order of magnitude or more slower than the original code.

Although we have a milestone (M17) dedicated to performance and scalability, we need to do some early performance improvement in M11 to ensure testing for M11 can proceed reasonably. At the current level, we can expect any time-sensitive tests to fail and testing in general to be unreasonably slow.

This issue is meant to be the parent of more specific sub-tasks that will address specific areas. Broad findings related to diagnosing bottleneck areas should be posted in this task, until we have enough information to open a more specific sub-task. From that point on, all history related to that particular area should be recorded in the related subtask.

The goal of this task is not to address every possible performance issue; that should be left for M17. The idea here is to find and pick the "low hanging fruit" which can get us some big, initial gains.

#2 - 03/30/2014 03:38 PM - Eric Faulhaber

In doing some basic profiling of the slow code, it is apparent the most egregious performance problems are with the areas of P2J which perform runtime conversion of dynamically defined queries and temp-tables. It seems we could benefit greatly from two main approaches to reducing work:

- minimize runtime conversion work by caching the results of previous runtime conversions, thereby avoiding runtime conversion entirely where possible, at the expense of memory and a cache lookup;
- minimize the overhead of TRPL by using a shared work queue of pattern engines, rather than constantly instantiating new instances in individual sessions and reloading TRPL configuration.

I will open new sub-tasks for these approaches.

#3 - 07/30/2014 12:22 PM - Eric Faulhaber

- Target version changed from Milestone 11 to Milestone 17

#4 - 07/12/2016 12:03 PM - Eric Faulhaber

- Status changed from New to Closed

#5 - 11/16/2016 12:31 PM - Greg Shah

- Target version changed from Milestone 17 to Performance and Scalability Improvements