When to commit changes?

Asked 16 years, 3 months ago Modified 6 years, 6 months ago

Viewed 4k times

5

Using Oracle 10g, accessed via Perl DBI, I have a table with a few tens of million rows being updated a few times per second while being read from much more frequently form another process.

Soon the update frequency will increase by an order of magnitude (maybe two). Someone suggested that committing every N updates instead of after every update will help performance.

I have a few questions:

- Will that be faster or slower or it depends (planning to benchmark both way as soon as can get a decent simulation of the new load)
- Why will it help / hinder performance.
- If "it depends ..." , on what ?
- If it helps what's the best value of N ?
- Why can't my local DBA have an helpful straight answer when I need one?
(Actually I know the answer to that one) :-)

> EDIT:

> @codeslave : Thanks, btw losing uncommited changes is not a problem, I don't delete the original data used for updating till I am sure everything is fine , btw cleaning lady did unplugs the server, TWICE :-)
>
> Some googling showed it might help because of issue related to rollback segments, but I still don't know a rule of thumb for N every few tens ? hundreds? thousand ?
>
> @diciu : Great info, I'll definitely look into that.

sql **oracle-database** **commit**

Share

Improve this question

Follow

edited Jun 6, 2018 at 8:00

wolφi

8,351 ● 3 ● 39 ● 70

asked Aug 28, 2008 at 19:40

Pat

36.7k ● 18 ● 74 ● 87

6 Answers

Sorted by: Highest score (default)

▲

4

A commit results in Oracle writing stuff to the disk - i.e. in the redo log file so that whatever the transaction being commited has done can be recoverable in the event of a power failure, etc. Writing in file is slower than writing in memory so a commit will be slower if performed for many operations in a row rather then for a set of coalesced updates.

In Oracle 10g there's an asynchronous commit that makes it much faster but less reliable: https://web.archive.org/web/1/http://articles.techrepublic%2ecom%2ecom/5100-10878_11-6158695.html

PS I know for sure that, in a scenario I've seen in a certain application, changing the number of coalesced updates from 5K to 50K makes it faster by an order of magnitude (10 times faster).

Share Improve this answer Follow

edited Aug 20, 2015 at 4:41
Community Bot
1 ● 1

answered Aug 28, 2008 at 19:55
diciu
29.3k ● 4 ● 53 ● 68

1

Reducing the frequency of commits will certainly speed things up, however as you are reading and writing to this table frequently there is the potential for locks. Only you can determine the likelihood of the same data being updated at the same time. If the chance of this is low, commit every 50 rows and monitor the situation. Trial and error I'm afraid :-)

Share Improve this answer

answered Aug 28, 2008 at 20:30

Follow

stevechol
502 ● 4 ● 8

1

As well as reducing the commit frequency, you should also consider performing bulk updates instead of individual ones.

Share Improve this answer

Follow

answered Sep 15, 2008 at 21:09

David Aldridge
52.3k ● 8 ● 72 ● 99

Can you elaborate on why that would help ? – Pat Sep 16, 2008 at 11:19

Because it allows the processing of more data with fewer SQL operations. It is faster – David Aldridge Oct 1, 2008 at 4:19

0

If you "don't delete the original data used for updating till [you are] sure everything is fine", then why don't you remove all those incremental commits in between, and rollback if there's a problem? It sounds like you effectively have built a transaction systems on top of transactions.

Share Improve this answer

Follow

answered Aug 28, 2008 at 20:32

BIBD
15.4k ● 26 ● 87 ● 141

0

@CodeSlave your your questions is answered by @stevechol , if i remove ALL the incremental commits there will be locks. I guess if nothing better comes along I'll follow his advice pick a random number , monitor the load and adjust accordingly. While applying @diciu twaks.

PS: the transaction on top of transaction is just accidental, I get the files used for updates by FTP and instead of deleting them immediately I set a cron job to deletes them a week later (if no one using the application has complained) that means if something goes wrong I have a week to catch the errors.

Share Improve this answer Follow

answered Aug 28, 2008 at 20:46

Pat

36.7k ● 18 ● 74 ● 87

0

> Faster/Slower?

It will probably be a little faster. However, you run a greater risk of running into deadlocks, losing uncommitted changes should something catastrophic happen (cleaning lady unplugs the server), FUD, Fire, Brimstone, etc.

> Why would it help?

Obviously fewer commit operations, which in turn means fewer disk writes, etc.

> DBA's and straight answers?

If it was easy, you won't need one.

Share Improve this answer

Follow

edited Mar 15, 2011 at 21:58

Jason Plank

2,336 ● 5 ● 32 ● 40

answered Aug 28, 2008 at 19:49

BIBD

15.4k ● 26 ● 87 ● 141