Table Scan vs. Add Index - which is quicker?

Asked 16 years, 3 months ago Modified 11 years, 5 months ago

Viewed 3k times

7

I have a table with many millions of rows. I need to find all the rows with a specific column value. That column is not in an index, so a table scan results.

But would it be quicker to add an index with the column at the head (prime key following), do the query, then drop the index?

I can't add an index permanently as the user is nominating what column they're looking for.

sql database optimization indexing

Share

Improve this question

Follow

edited Jun 25, 2013 at 20:07

Taryn

247k ● 57 ● 370 ● 408

asked Aug 27, 2008 at 13:30

ColinYounger

6,865 ● 5 ● 33 ● 33

8 Answers Sorted by: Highest score (default)

9

Two questions to think about:

1. How many columns could be nominated for the query?
2. Does the data change frequently? A lot of it?

If you have a *small* number of candidate columns, and the data doesn't change *a lot*, then you might want to consider adding a permanent index on any or even all candidate column.

"*Blasphemy!*", I hear. Most sources tell you to "never" index every column of a table, but that advised is rooted on the generic assumption that tables are modified frequently.

You will pay a price in additional storage, as well as a performance hit when the data changes.

How small is *small* and how much is *a lot*, and is the tradeoff worth it? There is no way to tell a priory because "too slow" is usually a subjective measurement.

You will have to try it, measure the size of your indexes and then the effect they have in the searches. You will have to balance the costs against the increase in satisfaction of your customers.

[Added] Oh, one more thing: temporary indexes are not only physically slower than a table scan, but they would

destroy your concurrency. Re-indexing a table usually (always?) requires a full table lock, so in effect only one user search could be done at a time.

Good luck.

Share Improve this answer

Follow

edited Aug 27, 2008 at 18:14

answered Aug 27, 2008 at 14:28

Euro Micelli

33.9k ● 8 ● 53 ● 72

8

I'm no DBA, but I would guess that building the index would require scanning the table anyway.

Unless there are going to be multiple queries on that column, I would recommend not creating the index.

Best to check the explain plans/execution times for both ways, though!

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:36

Jarrod Dixon

15.8k ● 9 ● 61 ● 72

3

As everyone else has said, it most certainly would not be faster to add an index than it would be to do a full scan of that column.

However, I would suggest tracking the query pattern and find out which column(s) are searched for the most, and add indexes at least for them. You may find out that 3-4 indexes speeds up 90% of your queries.

Share Improve this answer

Follow

answered Aug 27, 2008 at 14:08

Dane

9,827 ● 6 ● 29 ● 23

2

Adding an index requires a table scan, so if you can't add a permanent index it sounds like a single scan will be (slightly) faster.

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:32

Bill the Lizard

405k ● 211 ● 572 ● 889

2

No, that would not be quicker. What would be quicker is to just add the index and leave it there!

Of course, it may not be practical to index every column, but then again it may. How is data added to the table?

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:33

Joel Coehoorn

415k ● 114 ● 577 ● 813

2

It wouldn't be. Creating an index is more complex than simply scanning the column, even if the computational complexity is the same.

That said - how many columns do you have? Are you sure you can't just create an index for each of them if the query time for a single find is too long?

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:34

skolima

32.6k • 27 • 118 • 152

2

It depends on the complexity of your query. If you're retrieving the data once, then doing a table scan is faster. However, if you're going back to the table more than once for related information in the same query, then the index is faster.

Another related strategy is to do the table scan, and put all the data in a temporary table. Then index THAT and then you can do all your subsequent selects, groupings, and as many other queries on the subset of indexed data. The benefit being that looking up related information in related tables using the temp table is MUCH faster.

However, space is cheap these days, so you'd probably best be served by examining how your users actually USE your system and adding indexes on those frequent columns. I have yet to see users use ALL the search parameters ALL the time.

Share Improve this answer

Follow

answered Aug 27, 2008 at 13:56

hova

2,841 ● 20 ● 19

2

Your solution will not scale unless you add a permanent index to each column, with all of the columns that are returned in the query in the list of included columns (a covering index). These indexes will be very large, and inserts and updates to that table will be a bit slower, but you don't have much of a choice if you are allowing a user to arbitrarily select a search column.

How many columns are there? How often does the data get updated? How fast do inserts and updates need to run? There are trade-offs involved, depending on the answers to those questions. Do plenty of experimentation and testing so you know for sure how things will perform.

But to your original question, adding and dropping an index for the purpose of a single query is only beneficial if you do more than one select during the query (for example, the select is in a sub-query that gets run for each row returned).

Share Improve this answer

Follow

answered Aug 27, 2008 at 14:39

Eric Z Beard

38.4k ● 27 ● 101 ● 147