Official Publication of the Northern California Oracle Users Group

NoCOUG JOURNAL

VOL. 18, No. 3 · AUGUST 2004 $15

Ride with the NoCOUG Team!

Teamwork Makes a Difference (see page 27)

NoCOUG Goes One-on-One with Tom Kyte

Find out what happens behind the scenes of the AskTom website

Stayin' Alive: High Availability Without Breaking the Bank

Get the ins and outs of high availability from Jeremiah Wilton

Where's the SQL?

Joel Thompson discusses the lasting value of the best practices he learned on a project for a biotech firm

Much More Inside . . .

For Anyone Who Needs to Do Hands-on SQL Tuning

Book reviewed by Brian Hitchcock

SQL Tuning—Generating Optimal Execution Plans
by Dan Tow; published by O'Reilly, 2004

Summary

Overall review: Excellent
Target audience: Anyone who needs to do hands-on SQL statement tuning.
Would you recommend to others? Yes, if they need to do SQL tuning.
Who will get the most from this book? Readers should have a general working knowledge of SQL syntax and be aware of the high-level issues involved with SQL tuning.
Is this book platform specific? Yes and no. The overall approach is not platform specific, but the examples provided cover Oracle, SQL Server, and DB2.

Overall Review

I had been asked to review the SQL for a 17-table join that was running slowly in a CRM system. My task was to verify whether Oracle was properly optimizing the SQL statement. Therefore, I had a very specific need to find information relevant to SQL statement tuning. While I am experienced with the overall concepts of SQL tuning, I had never been totally comfortable with the way it is generally done. My experience is that after a lot of analysis, the result is a lot of big words that cover possible causes and possible solutions and maybe this and maybe that, but nothing really clear-cut as to what's good, what's bad, and what could be fixed. I was at the bookstore (yes, a physical bookstore!) looking for books on a different database topic when I saw this book on the shelf. I assumed it would be nothing more than the usual, generic review of all the possible issues with SQL statements running in commercially available RDBMS products. I read the back cover of the book first and quickly scanned over what I felt was the usual stuff, until I found the following statement: "How do you decide which execution plan a query should use?" I've never really had a good answer to this, so I read further.

This is exactly what I think has been missing in the past. Namely, some conclusive evidence that we're doing the right thing. Theory is fine, and theory gets a lot of print, but you'd really like to deliver a concrete answer to a specific question. This book promised to teach me how to do just that. Most of the SQL tuning information I'd seen over many years tends to focus on the mechanics of what the Oracle optimizer was doing and/or the details of SQL syntax. However, the answer I really want is how to determine that what I'm getting from the database is the best possible solution.

The Technique You Will Learn

The technique author Dan Tow provides is summed up on the back cover of the book as a diagram-based method for tuning SQL statements. This means that you look at the SQL statement, then generate a diagram of the tables involved and how they interact. You then examine some simple statistics about each of the tables. From the diagram and the statistics, you can quickly see which tables are the most important to the performance of the query. You then compare the results from your diagram with the plan that your database optimizer has chosen. For Oracle, this means comparing your diagram with the output of the explain plan. From this comparison, you can tell if the database execution plan is the best that can be done.

This technique sounds too good to be true, but it worked for me. Several advantages of this technique (besides the fact that it works!) need to be highlighted. First, you generate the diagram from the SQL just the way it was given to you. This means you don't have to have any insight or years of experience or intuition to apply the technique. You simply take the SQL and move through it to build the diagram of the tables and the interactions.

Second, you gather some simple statistics about the tables, such as row counts and the selectivity of indexes. You use these numbers to compute some ratios that you apply to your diagram. Again, no need for world-class expertise—the author explains exactly how to gather the statistics and how to compute the ratios needed.

Third, with the diagram drawn and populated you can quickly see which tables are defining, or limiting, the performance of the query. This means you can stop worrying about most of the tables and focus on the one or two that are really worth your time. You will know where to apply your time to get the greatest performance benefit.

And finally, note that you don't need full DBA access to the database. This is a real issue for production systems. The people who know the application and the schema best don't always have full access to all parts of the database. This technique only requires database access to the schema that owns the objects in the query being reviewed. You don't need to have access to system tables to apply this technique.

This technique also saves time in that you don't have to worry about all the possible combinations (join order) of tables that you might have to look at for a complex query. The order in which tables are joined has a big impact on the performance of the query. For my 17-table join, the number of possible join orders is huge. Applying this approach eliminates almost all of the tables as being the cause of performance problems and usually (almost always) identifies the one table that must be accessed first to get the best possible performance.

Your users will also like this technique because it cuts through the techno-babble that most technical persons feel compelled to provide to users. End-users don't really care that there are many millions of ways to join 17 tables. They do like getting a single diagram that shows all the tables (this makes it look like you did at least look at the entire query!) and which one or two are the most important for performance. It is more meaningful to them when you can provide a simple explanation as to why one table is so important to performance and all the others aren't. This also means you can answer many related questions all at once without spending any more time. For example, once I had analyzed the 17-table join, it was trivial to answer questions from the developers as to the effect of joining one more (or less) table. Again, saving time and leveraging your analysis are great benefits of this approach. Typically, an application will have groups of queries that are run quite frequently. Once you have diagrammed these common queries, you can answer many more questions about related queries without repeating most of your work.

Chapter by Chapter

Chapter 1 covers why you need to tune queries, something most NoCOUG members already know.

Chapter 2 reviews tables, indexes how joins work, etc. Again, it's basic background, but some of the diagrams are cool.

Chapters 3–4 cover how to generate execution plans and how to change them by changing the query or the indexes and so on. These chapters include specific sections for Oracle, SQL Server, and DB2. This is very good because it ties the general method back to the specifics of the syntax of each of these products.

Chapters 5–6 are, I think, the really good parts. If you don't read anything else (other than my review, of course), you should read these two chapters and then start looking at your specific problem queries. Chapter 5 starts with applying the technique to simple queries giving lots of detail on how this is done. A more complex query follows. These examples and the exercises are great. This is where I determined what I did and didn't understand. The author included solutions to the exercises he provided, and I used them. I strongly recommend that you do the exercises and make very sure you work through the solutions in detail. Again, this is where I learned what I needed to do to apply this technique. Chapter 6 then explains how to verify what the best execution plan is. This means you take the diagram of the query (chapter 5) and you move through it to determine the best way the query can be executed. Some of the special cases in this chapter were way beyond me, but that's okay. They are worth reading just so you can see how well this technique works.

Chapter 7 is where the author delves into unusual situations that I simply haven't seen in my work. To be very clear, the material is great, and I'm sure it's all very valuable, but I haven't needed to use most of it. This isn't a criticism of the material. It is great for those queries that have these unusual structures. Thankfully, I haven't had to deal with them. The basic technique works almost all of the time. For example, I haven't needed to understand cyclic join graphs or diagrams with multiple roots, but you might need to. Also, the section on generating diagrams for subqueries is good, and I did understand that part. I also got useful information from the sections on outer joins (my query had lots of them) and views. I would focus on the most useful chapters that I've identified above so you can quickly generate the diagrams of your queries. If your diagrams contain anything other than a single line between two tables, then I would review this chapter looking for help with your specific query.

Chapter 8 is a reward of sorts. After applying the technique, the reason it works is explained. That is another appealing feature of this author's technique, I can understand why it works. This chapter is brief, to the point, and very worthwhile. This should give you confidence that this approach will work in almost all cases.

Chapters 9 and 10 offer solutions to "special cases" and "seemingly unsolvable problems"—good titles, I think. I read these chapters, but I haven't seen anything like these situations in my work. I suggest you skim the section headings looking for something that applies to you if the basic technique doesn't provide a complete answer for your queries.

Appendix A has the solutions to the exercises that I've already told you to use.

Appendix B is great; it covers another fairly complex example from end to end. I strongly urge you to go through this.

What Could Be Better

Well, okay, you liked the book, but you have to say something critical just to show your readers how smart you are (and prove that it's not your brother who wrote it)—right? Well, after much review I can offer only a few minor criticisms. The technique that I learned from this book worked for me. But I didn't try a large number of queries, so it is possible that I just got lucky. I can't offer you a statistical study of thousands of queries that were analyzed and whose performance was measured to prove the effectiveness of this technique. Still, I know from my years of experience that the results I provided my users with after applying what learned from this book, were substantially better than what I used to provide.

Further, I'm sure that someone somewhere has a query that will cause this technique to break. After you read and apply this approach, I think you will agree that it works very well for the vast majority of queries that really come up. More important, most of us spend most of our time looking at queries that are pretty straightforward. My 17-table join is very common in modern commercially produced applications.

I would have liked more examples. Yet the examples shown in the book and the detailed analysis of them that is presented allowed me to work through my real-world query on the first try. There had to be a balance between the number of examples covered and the detail presented for each one. I found that this book provided just enough examples for me to learn the technique, but not so many that I never got to the end. And in the end, I was supposed to learn a technique. The number of examples that are presented in detail were enough to help me do that.

The Author

Dan Tow worked for Oracle for many years in the Applications group focusing on performance. This experience exposed him to many sets of SQL that needed to be examined. He has since started his own consulting business providing SQL tuning services.

Conclusion

Overall, this book delivers on its advertised promise. I actually read every page in this book; I didn't just skim the book to do a book review. The book is also blissfully short. I was able to read all of it in about four evenings, without missing any of my all-important television. Too many authors feel the need to write 600 pages (I should know . . .) even if they have to include copies of large sections of the Oracle documentation. And as mentioned above, I had a real need to find answers quickly for people who were paying me to support them. Therefore, I read this book with a very critical eye. And in the end I was able to provide my users with specific answers to their questions about the specific SQL they asked me to review. Best of all, they actually think I know something, which is always a pleasant surprise! ▲

About the Reviewer

Brian Hitchcock has worked at Sun Microsystems in Newark, California, for the past nine years. Before that he worked at Sybase in Emeryville, California. Even further into the past he spent 12 years at Lockheed in Sunnyvale, California, designing microwave antennas for spacecraft. He supports development databases for many different applications at Sun, handling installation, tuning, NLS, and character set issues as well as Unicode conversions. Other interests include Formula One racing, finishing a Tiffany Wisteria lamp, Springbok puzzles, Marklin model trains, Corel Painter 8, and watching TV (TiVo rules!). Brian can be reached at brian.hitchcock@aol.com or brhora@aol.com. ▲

Editor's Note: NoCOUG members receive a 20% discount on O'Reilly, No Starch, Paraglyph, Pragmatic Bookshelf, and Syngress books and O'Reilly conferences. Just use code DSUG when ordering online or by phone (800) 998-9938. http://www.oreilly.com/

*** *Free ground shipping is available for online orders of at least $29.95 that go to a single U.S. address. This offer applies to U.S. delivery addresses in the 50 states and Puerto Rico. For more details, go to: http://www.oreilly.com/news/freeshipping_0703.html*