Stack Overflow Makes Slow Pages 100x Faster by Simple SQL Tuning

Monday, May 2, 2011 at 8:45AM

General Chicken in Strategy

The most common complaint against NoSQL is that if you know how to write good SQL queries then SQL works fine. If SQL is slow you can always tune it and make it faster. A great example of this incremental improvement process was written up by StackExchange's Sam Saffron, in A day in the life of a slow page at Stack Overflow, where he shows through profiling and SQL tuning it was possible to reduce page load times from 630ms to 40ms for some pages and for other pages the improvement was 100x.

Sam provides a lot of wonderful detail of his tuning process, how it works, the thought process, the tools used, and the tradeoffs involved. Here's a short summary of the steps:

1. Using their mini-profiler it was shown that a badge detail page was taking 630.3 ms to load, 298.1 ms of that was spent on SQL queries, and then the tool listed the SQL queries for the page and how long each took.
2. From the historical logs, which are stored in HAProxy on a month-by-month basis, Sam was able to determine this page is accessed 26,000 times a day and takes 532ms on average to render. This is too long, yet there are probably higher value problems to be solved, but Google takes speed into account for page rank and Sam thinks this can be done faster.
3. Sam noticed:

    1. There were lots of select queries which Sam calls the N+1 Select Problem. There are many individual detail select queries to get all the data for a master record. More details at: Hibernate Pitfall: Why Relationships Should Be Lazy.
    2. Half the time was spent on the web server.
    3. There were some expensive queries.
4. Performing a code review the code uses a LINQ-2-SQL multi join. LINQ-2-SQL takes a high level ORM description and generates SQL code from it. The generated code was slow and cost a 10x slowdown in production.
5. The strategy was to remove the ORM overhead using Dapper, which is a simple .Net object mapper, to rewrite the query. The resulting code is faster and more debuggable. Linq2Sql is being removed from selects, but is still being used for writes on Stack Overflow.
6. In production this query was taking too long, so the next step was to look at the query plan, which showed a table scan was being used instead of an index. A new index was created which cut the page load time by a factor of 10.
7. The N+1 problem was fixed by changing the ViewModel to use a left join which pulled all the records in one query.

A NoSQLite might counter that this is what key-value database is for. All that data could have been retrieved in one get, no tuning, problem solved. The counter is then you lose all the benefits of a relational database and it can be shown that the original was fast enough and could be made very fast through a simple turning process, so there is no reason to go NoSQL.

Related Articles

- Stack Overflow Articles on HighScalability
- Reddit Thread

- HackerNews Thread
