

Re: Datenbank SQL-Server vs. MySQL

Source:

<http://www.tech-archive.net/Archive/German/Entwicklung/microsoft.public.de.german.entwickler.dotnet.datenbank/2>

From: Klaus Oberdalhoff [MVP] (*kobd_at_gmx.de*)

Date: 11/10/04

Date: Thu, 11 Nov 2004 00:16:12 +0100

Hi,

> *SAP-DB*

jibbet es nich mehr

Die heisst jetzt MAXDB und hat sich wohl nach langer Irrfahrt unter die Haube von www.mysql.com begeben.

Hier eine Antwort (vom 28.7.2003) auf meine Frage zu der damaligen SAPDB – welche Argumente für den SQL-Server sprechen ...

mfg

Klaus Oberdalhoff(Access MVP) KObd@gmx.de

> *if looking at the features for the sapdb database (www.sapdb.org) which is
> freeware under GNU licence it seems impressive – All the features of MS
> SQL
> server <incl. OLTP and OLAP, 64bit version, multi-plattformung etc.>, but
> freeware. Also the sponsor (SAP) is "not bad".
>
> How on heaven and hell is MS is planning to "fight back" against that sort
> of
> application?
>
> If i get asked from a customer how to argue against that and "pro SQL
> Server" ?
> Which arguments can you provide me ?
>
> OK to argue against mySQL is easy, you always can argue of missing
> features and
> "toy", but here <it aims at first step to emulate Oracle 7 and sponsor
> SAP, so
> it can't be a "toy"> ?
>
> I'm in need of arguments, because i already got asked vaguely... (I*

- > *haven't used*
- > *sapdb so far, so i haven't a clue if the written arguments are just*
- > *marketing*
- > *"bla bla" or if the DB is really that good.)*
- >
- > *Please help me.*

SAP DB is ADABAS D. Software AG was failing and decided to exit the database business. SAP had a large number of R/3 customers, primarily in Germany, using ADABAS D and decided they had to purchase the technology in order to protect their R/3 customer base. However, SAP did not want to be in the database business and started to look for ways to keep SAP DB going without a major investment on their part (the three major DB vendors each have between 10x and 25x the number of development people working on their products as SAP has working on SAP DB). SAP's first attempt at doing this was to take SAP DB open source. That doesn't seem to have worked out. Those who want an open source database have pretty much gone with MySQL and ignored all the other (technically better) choices. Those who don't care about open source, but care about the level of support (both vendor technical support and ISV support including applications, tools, trained people available on the market, etc.) have gone with Oracle, DB2, or SQL Server.

SAP has now switched gears and partnered with MySQL. SAP DB will be offered as a "version" of MySQL under MySQL's licensing model. I think it's a totally open question if this move will allow SAP DB to gain any ground in the database market. It further distances SAP DB from the theoretical backing of SAP, which was pretty weak to begin with. And having two incompatible offerings will confuse those looking to use MySQL. The strategic problem for MySQL remains pretty clear. Would you bet the existence of your business on products that don't have the backing of a major company? For all MySQL's success to date, it is primarily used in applications where it is not the primary data store. For example, if you want to cache your catalog data at a web server then you might use MySQL. But you can always recreate that cache from the primary data in your Oracle, DB2, or SQL Server database.

When a major stock exchange in the U.S. is out of commission for a few minutes they get a call from the U.S. Treasury Secretary. If the outage is long enough, the President of the United States will call. Now, if it's a database problem who does the CEO of NASDAQ call for backup, Monty (the founder of MySQL)? Would he rather know that he can call Steve Ballmer or Bill Gates, and that they have 50,000 people (including contingents within minutes of all Nasdaq data centers) to throw at the problem. Of course. And this security blanket issue extends throughout large, medium, and even small enterprises.

To show just how complicated the situation is start with MySQL. The MySQL storage engine is not even industrial strength. To get MySQL to approach industrial strength you have to use the InnoDB storage engine. InnoDB comes on the MySQL distribution, but is produced by another company. So already

you have a complexity that doesn't exist with SQL Server, Oracle, or DB2. And that doesn't even get to where various APIs and tools come from. With the SAP DB situation it looks every bit as bad. You'll acquire "MySQL SAPDB" from MySQL AB but the product development responsibility remains with SAP. Just who looks you in the eye and says "we promise to make you succeed", and can they live up to the promise?

And now a disaster story from the MySQL world. Progress Software decided to take its industrial strength storage engine and package it with MySQL. They formed a subsidiary to market this industrial MySQL offering. There was then a spat with MySQL AB over various things, which appeared to be some combination of the interpretation of the GPL and the use of the MySQL trademark. Anyway, from what I can tell Progress just abandoned the whole thing and folded the subsidiary. How would you have liked to have been a customer who purchased this great industrial strength MySQL offering only to have a spat between partners leave you out in the cold with a dead product? What happens if/when SAP AG and MySQL AB have a spat?

There are other factors specific to SAP DB. If its so good, why did Software AG fail? Remember, they were a major player in the pre-relational era. I knew a lot of senior IT people who thought they were a fantastic company, and preferred their product to IMS DB, DL/1, Cullinet, or other players of that era. Why does Adabase D/SAP DB have little penetration outside Germany? I think the only reason it has any penetration at all is that German companies (as is the case in any country) have a preference for home grown products. Why do so few SAP R/3 installations use SAP DB? If SAP really backs SAP DB, why don't they really push it more strongly for use with R/3? They don't seem to be pushing it outside of the Linux environment. SQL Server has dominated new SAP installations over the last 5 years (though I don't know the most recent trend). IBM is a very strong SAP partner, with much of that partnership centered around DB2. Despite SAP's animosity with Oracle, most large SAP installations are still on Oracle.

There are few objective measures when comparing databases, but performance and scalability certainly is one. The largest R/3 SD 3-tier benchmark # of users for SQL Server 2000 is currently 26,000. The largest for SAP DB is 5500. The largest R/3 SD 2-tier benchmark # of users with SQL Server 2000 is currently 2750. The largest for SAP DB is 470. If SAP DB was so great, and SAP was really standing behind it, you would expect them to have gotten together with a hardware partner and published the top benchmark results using their own database system. Obviously there are reasons, be they scalability itself or other reasons they wouldn't want high end installations to use SAP DB right now, that SAP hasn't done this.

On the technical level SAP DB may be able to check a lot of boxes, but that doesn't mean it is qualitatively as good as the checkboxes make it out. Oracle was #1 in OLAP just a few years ago, but now barely registers in the marketshare numbers. IBM packaged a leading third-party product so they could check the OLAP box, but again their share numbers don't register. As far as I can tell SAP DB is a decent product, but many decent products have failed in the market (Informix, Sybase, Ingres, Interbase, Sharebase,

numerous OODB products, etc.).

So how do Microsoft, Oracle, and IBM compete with MySQL, SAP DB, and other theoretically free products? One way, obviously, is to stay a step ahead on the functionality, reliability, availability, performance, etc. fronts.

Another is to focus on the true total cost of ownership and make sure that is lower than products that just offer lower initial cost of acquisition. A third is to focus heavily on completeness of overall product offerings, such as development tools, application servers, and management infrastructure rather than just on the database engine. A fourth is to focus on support and service.

Back in the 1980s I was on the bandwagon that database engines were a commodity and should be given away. At DEC we went so far as to do that, giving the Rdb engine away with the system purchase and charging for the Development Environment and Management Tools. We even wedged the investment model to reduce the investment in the engine and increase it for all the surrounding (non-commodity) pieces. We were premature to say the least and database engines continue to evolve rapidly enough to maintain non-commodity status (and prices) 15 years later. But eventually database engines will become a commodity and the price, for the engine alone, will approach \$0.

Microsoft is no stranger to commoditization. While a lot is made of the "free" Linux vs the \$50 or so that OEMs pay for Windows XP, you still have to be amazed at what \$50 buys you. (A little off topic: My classic argument with Linux people over why OEMs don't package in Linux is this: The reason OEMs only pay \$50 for Windows is that they assume first line support. Even though Linux is free, they also have to provide first line support. If the rate of calls they'd get on Linux is double that of what they get on Windows, and it would be since so few non-geeks know anything about Linux, the effective cost of Linux to the OEM would be much higher than Windows.) Microsoft turned \$5000 workstation operating systems into a \$50 product. It turned \$10,000 development tools into \$100 products. And it turned \$100,000 databases into \$5000 products. It also learned that if you add enough capability you can have \$1000 versions of your development tools and \$20000 versions of your database product and people will buy them despite having the availability of lower cost alternatives.

The next step in Microsoft database commoditization is already happening with MSDE, which follows the model of you buy the development tool and that grants you free run-time redistribution, bringing the database engine price effectively to \$0. So with SQL Server you pay for higher-end scalability, availability, Analysis Services, Notification Services, Reporting Services, some Replication features, and the Management Tools. You already pay enormously less for all of this added value than you would have to pay by combining the individual capabilities from multiple sources. I think this suggests a long term model for Microsoft (and other vendors). Give away that which has become a commodity, charge a reasonable amount for that which adds (non-commodity) value. If you can stay sufficiently ahead of the commoditization curve then you'll have a healthy revenue stream. If you can't, you don't deserve to stay in the business.

microsoft.public.de.german.entwickler.dotnet.datenbank: Re: Datenbank SQL-Server vs. MySQL

Hal Berenson, SQL Server MVP
True Mountain Group LLC