Re: COBOL and DB2 vs. Java and DB2

Source: http://coding.derkeiler.com/Archive/Cobol/comp.lang.cobol/2007−09/msg00794.html

- *From*: "Pete Dashwood" <dashwood@xxxxxxxxxxxxxxxxxxxxxxxxx>
- *Date*: Tue, 18 Sep 2007 04:30:37 +1200

"Judson McClendon" <judmc@xxxxxxxxxxxxx> wrote in message
news:MGuHi.52087$Y7.9502@xxxxxxxxxxxxxxxxxxxxxxxxx

> "Robert" <no@xxxxxx> wrote:
>
> > "Judson McClendon" <judmc@xxxxxxxxxxxxx> wrote:
> >
> > > In the situations I am familiar with, traditional mainframe
> > databases
> > > blow
> > > the doors off SQL type approaches. Whoever designed SQL
> > must have
> > > been entirely clueless, or indifferent, to machine efficiency.
> >
> > Everyone who's run a Google search knows that to be untrue. It displays
> > the time it spent
> > searching three billion Web pages. I just did it with four wildcard
> > words −− "for * * nail
> > the * was lost". It found 27,000 in .08 seconds. That's fast!
>
> Yes, and how many shared servers does it require to achieve that, eh?
> My point was about hardware efficiency. If it takes 20 computers to
> do what one could do using another approach, then it is less hardware
> efficient.

Judson, you may be over−reacting here (Robert's posts often have that effect on people :−)) and losing sight of the actual argument.

If those 20 computers each cost less than one twentieth the cost of the single computer, then your case is not made.

While I'm here, I would point out that your statement above regarding SQL is at best provocative, at worst, simply untrue.

If, by "traditional mainframe databases" you mean VSAM datasets connected together, or (I seem to recall your background is Unisys?) perhaps indexed random files in a FORTE style database, then it is very arguable whether these would "blow the doors off" an SQL type approach (by which I take it you mean Relational Database).

There was a time when this MIGHT have been true, but that time is long gone. Certainly, a well designed "Traditional" DB benchmarked against a poorly designed RDB will not fare well (I know, because I used the performance argument to enable a certain company to hold onto their existing VSAM and resist RDB, but that was 20 years ago...)

I've had a bit to do with Relational Databases (and, indeed, traditional databases) and enhancing the performance of both. I don't think your statement stands against a modern Relational Database (even free ones like MySQL, and PostgreSQL, which are excellent products.)

RDBMS are much smarter than they used to be; the better ones can actually monitor their own performance and optimize storage for the statistically most likely accesses being experienced. If access patterns change, so does the physical organization of the data. Stored procedures can run as asynchronous tasks, and the latest models allow result sets to be obtained and processed in cache so that connections are freed instantly and there is no need to lock a connection while running a SQL cursor, for example.

Ironically, although today's SQL is running on much more powerful platforms than yesterday's did, it still not enough for the future and new ways of accessing data (not to mention new storage technologies) are already emerging

Just one such is the emerging technology of Query Expressions (LINQ) and Lambda functions for data, which is using the Relational model to set SQL on its head and support deferred execution, load levelling across servers, and parallel processing, all transparent to the application programmer. Some of this stuff (Lambdas) is showing incredible data transfers with terabytes transmitted instantly across country. (Try GOOGLE on the terms I've mentioned.)

Here's a video interview from 2005 with Anders Hejlberg, which explains why an Object approach to Data access can allow "across the board" access to tables in memory, XML files, Documents, as well as Relational DBs, using a single OO Query language. Most of what he talks about is now available:
http://channel9.msdn.com/showpost.aspx?postid=114680

If you bring yourself up to date on database technology you may well find it blows the doors off traditional approaches...:−)

Personally, I find it very exciting and am beginning to use C# to run Query Expressions rather than SQL. It is very early days yet and I am still learning, but it is much more natural than SQL., and integrates seamlessly

into the language.

With new storage technologies just around the corner we need to find smarter ways to access data. SQL, particularly embedded SQL, may be rendered redundant by these new approaches. It is a very long way away from "traditional mainframe databases".

Pete.
−−
"I used to write COBOL...now I can do anything."

.