So Just What is a Big Database These Days?
By Craig S. Mullins

With the current level of interest in Big Data and analytics many of you may be asking yourself "What exactly does it mean to be Big?" As it turns out, that is a very good question!

Sometimes terms like Big Data and large objects get bandied about without a lot of thought being put into it. You know what I mean? Every now and then a consultant or an article will offer up some advice like "***Large*** table spaces should always be partitioned" or "Relational databases cannot adequately support ***Big*** Data." But how useful is this advice? What do they mean by large and big? Terms such as these are nebulous and always changing.

So let's take a minute or two to think about this issue. Just what is a ***large*** database today? Of course, the standard answer of "it depends" applies – it depends on your site, your mixture of data types, and your particular requirements. But is that any more useful? Let's try to define the term and put some hard numbers around it.

First of all, the question begs to be asked: "Large in terms of what?" The first thing that springs to mind is the actual size of the database. But are we talking about a single table, a single table space, or an entire database? And sometimes we might even be referring to the size of an entire subsystem or instance (such as for SAP ERP implementations).

If we are at the table level, what is the granularity of measurement for determining what is large? Do we talk in terms of number of rows or number of pages (or blocks)? Or just the amount of disk space consumed? And do we count just the base data or add up the space used by indexes on that data as well? Do we measure before and after compressing the data? Or just measure what is in use (regardless of whether it is compressed or not)?

And what about the type of data? Is a 40 GB database consisting solely of traditional data (that is, numbers and characters; dates and times) bigger than an 80 GB database that contains non-traditional BLOBs and CLOBs? From a purely physical perspective the answer is obvious, but from a management perspective the answer can be more nebulous. It may indeed be more difficult to administer the 40 GB database of traditional data than the 80 GB database of media objects because traditional data consists of more disparate attributes and is likely to change more frequently. But then again, if the multimedia data tends to change (that is, it gets modified), then even a smaller amount of that type of data can be more difficult to manage because of the different ways it is stored and managed.

Another issue is: just what are we counting when we say we have a large database? Do we want to count copied and denormalized data? And what about free space; should that count, too? There are two schools of thought: one says if it is in the database, then it counts. Another way to look at it would be to only count the core data. From the perspective of the DBA though, you have to count everything that needs to be managed – and doesn't ***everything*** need to be managed?

One useful measure of large databases is offered by Winter Corp., a research and consulting firm specializing in database scalability. Winter Corp. applies its research and analysis resources to measure industry database implementations in terms of size, rows/records and workload. Every other year for a decade (2002 through 2012) Winter surveyed the largest databases it could find and reported the results. As of their last report, the largest database they could find in production was approaching 25 petabytes in size. Now there is no denying that that one is Big!

But the biggest of the big is not the only database that deserves to be called big. There are many examples of production database implementations over a petabyte in size. For example, The Panoramic Survey Telescope and Rapid Response System, (Pan-STARRS) is used to store 1.1 petabytes of data using Microsoft SQL Server.[1] Another example comes from Yahoo!, which has implemented a 2 petabyte data warehouse using a "heavily-modified PostgreSQL engine."[2] Or perhaps most impressively, is eBay's claim to process over 10 billion rows per day on systems holding over 5 petabytes of data in Oracle, with the single largest system bigger than 1.4 petabytes.[3]

But instead of looking just at the extremes, let's bring our focus back to the center. What do you need to do to get your arms around the question of "What is Big?" at your organization? At a high level, first you need to be prepared with the *criteria* for what establishes database BIGness at your shop. Is it a management issue? A planning issue? It better be both of those, but sometimes it is a braggadocio issue, too! You know, being able to say "My database can beat up your database!" Vendors play that game sometimes.

Furthermore, the granularity of the object being discussed is important, too. Many industry publications talk about size at the database or instance level, but DBAs should be more interested in managing at the table or table space level, because that is where the administrative difficulties arise.

OK, let's bring this back around to a more specific question. My primary focus is DB2, so let's ask "What is a large DB2 table space?" A good place to start is probably 4 GB. In DB2 for z/OS, if you want to specify a value greater than 4GB there are issues that change the internal structure of the table space and it require additional resources to manage. Of course, depending on your shop and its requirements this might be too high… or even too low.

So here we are, near the end of this article and we seem to have more questions than answers. So how about some advice?

---

[1] http://news.softpedia.com/news/Microsoft-Applauds-1-1-Petabytes-SQL-Server-2008-Database-97388.shtml

[2] http://www.computerworld.com/article/2535825/business-intelligence/size-matters--yahoo-claims-2-petabyte-database-is-world-s-biggest--busiest.html

[3] http://www.dba-oracle.com/oracle_news/news_ebay_petafiles.htm

First of all, when determining "What is Big?" for your shop, do it in terms of the number of pages, not the number of rows. And be sure to standardize on the size of page you use for comparison purposes. Again, using DB2 for z/OS as an example, page sizes can be $ K, 8 K, 16 K, and 32 K. But the most common size is 4 K, so standardize on that by calculating the number of 4 K pages being used even if the page size is greater… at least when trying to evaluate and compare the size of your table spaces. You can use the number of 4 K pages to easily compare the size of one table space to another, whereas you cannot if using number of rows because row size can vary dramatically from table to table.

And when evaluating at the database or subsystem/instance level, count everything that is being persistently stored: data, indexes, free space, etc. If it is being stored it must be managed, and therefore impacts TCO. Stripping out everything but normalized data doesn't really matter if the data is not actually stored that way.

The Bottom Line

One thing can be said for sure, though – and that is this: our databases are getting bigger. And all signs point to things getting even larger. We are storing more data this year than we did last year… and we'll be storing even more next year.

Big Data systems and applications are delivering results at many organizations and with the Internet of Things on the immediate horizon it is assured that data growth will continue. Many organizations have begun talking more frequently in terms of petabytes rather than terabytes or gigabytes. Let's face it, database bigness is a fact of life these days.

Who said life as a DBA is boring? It certainly wasn't me!