

Is 24x7 a Myth?

Ask the Oracles!



Gaja Krishna

Vaidyanatha: Lose weight instantly with this magical pill...no exercise or diet required! Transform your life completely within 12 minutes with this “self-help” video! Become a millionaire in less than 30 days without

leaving your home...and more. Our day-to-day lives are pummeled with many such unbelievable claims. The media does a great job of propagating this.

If HA is defined by 99.999% uptime on any given year, the “available downtime” both for unscheduled and scheduled events is a paltry 5.2596 minutes. That is all the time that you have for all of your system maintenance including routine database administration, operating system and Oracle software patching. I don’t think I have to even attempt to convince you that this uptime goal is impossible and downright crazy for most systems!

There is a much higher probability that the DBAs supporting these systems will scale Mt. Everest.

No matter how well you have automated your environment, 5 ¼ minutes of downtime a year to perform a plethora of required system maintenance and database administration activities in most

systems (especially running Oracle databases) is a lofty task. For most systems out there, there is a much higher probability that the DBAs supporting those systems will scale Mt. Everest without oxygen versus the system achieving 99.999% uptime!!! Remember the last time you applied an Oracle patch? How long did it take? Did everything go as planned? How long was your application (system) unavailable? Remember the last time, a simple “zero risk” effort on your database caused downtime? Need I say more?

P.S. High availability in an Oracle environment is almost always equated to Oracle RAC. If you are thinking of achieving HA with Oracle RAC and you have not designed your application for a clustered environment...think again!!! Slapping Oracle RAC into

your environment without expending the necessary time and effort to design your application is a guaranteed recipe for disaster...at least in the realm of performance management.

Gaja Krishna Vaidyanatha has over thirteen years of industry technical expertise working with Oracle systems and providing strategic and technical direction to companies such as Veritas Corporation, Oracle Corporation and Quest Software. He is the primary author of “Oracle Performance Tuning 101” published by Oracle Press and one of the co-authors of "Oracle Insights : Tales of the Oak Table". Gaja is a member of the Oak Table Network and can be reached at gaja@dbperfin.com.



Mogens Norgaard: I have known of many systems running VMS or Linux or UNIX with Oracle on them that would run for hundreds of days without interference and without problems.

But when I look back, I think those systems that just keep running day in and day out are either very simple or very, very complex (guess what happened to costs, too).

The very simple systems, where you just have a stable box with some standard Oracle installed, and where you don’t fiddle all the time with parameters, setup, etc. are very often the ones that will keep running for close to a year at a time and in rare cases even longer.

The very simple systems are very often the ones that will keep running for close to a year at a time.

The very complex ones where you have mirrored SANs, Data Guard, RAC, iAS installations with all High Availability features enabled (in other words Oracle’s High Availability Architecture and more), and redundant

power supplies and all that can also be made to work for prolonged periods, but chances are smaller due to the higher stack and the sheer complexity of all those products and options interacting with each other.

Then there are the occasional bugs; for instance, the one that that caused Oracle 8 to shut down after 248 days due to an internal counter. Refer to Metalink note 118228.1.

Do most businesses need 24x7? Probably not. When we had a huge power-out on the island of Zealand last year, with most of Copenhagen and its surroundings blacked out, no businesses had to close as far as I know.

But when the MBA-boys from the huge consultancies tell you that a few minutes downtime will cost you 42 billion dollars, the conclusion is obvious: You must invest heavily (really, you cannot invest too much!) in so-called Business Continuity, as they call High Availability these days.

Nah, I don't think so.

Buy a standard box with standard CPUs in it. Make sure it's produced in the same factory as your home computer, so you know it has zero defects. Then take a good backup and test it now and then.

Use the Disposable Computing Architecture (James Morle invented that term), which means you should have another standard box standing ready (powered off to save energy) in the corner.

Invest in a couple of NAS boxes or even better attach disks directly, and you're ready and open and highly available with simple and cheap (and therefore efficient and easy-to-understand) technology.

Mogens is the CEO of Miracle A/S (www.miracleas.dk), a database knowledge center and consulting/training company based in Denmark, and is the co-founder and "father figure" of the Oak Table network. He is a renowned speaker at Oracle conferences all over the world and organizes some highly respected events through Miracle A/S, including the annual Master Class and the Miracle Database Forum. He is also the co-founder of the Danish Oracle User Group (OUGKD) and was voted "Educator of the year" in Oracle Magazine's Editor's Choice Awards, 2003. Mogens can be reached at mno@miracleas.dk.



Jeremiah Wilton:

For the individual database, yes.
For a service sufficiently abstracted from the database, no.

High availability is no myth for many organizations. But while you can take many steps to create a stable database, true availability is measured from the end user's perspective, several layers removed from the database.

High-availability solutions at the database level, such as

RAC and DataGuard, while potentially important tools in creating stability for a given database, are not the starting point of any larger software-based high availability solution. The best availability in the industry comes from application software that makes a counterintuitive assumption: *The databases upon which the software relies will inevitably fail*. The better the software's ability to continue operating in such a situation, the higher the overall service's availability will be.

But isn't Oracle unbreakable? At the database level, regardless of the measures you take to improve availability, you inevitably will incur some kind of outage from time to time. An outage may be from a required upgrade or a bug. Knowing this, if you engineer application software to handle this eventuality, then a database outage will have less or no impact on end users.

The best availability comes from application software that makes a counterintuitive assumption

One way of engineering applications to maximize availability in a down-database situation is through availability testing. Imagine an ordering system, with two databases - one for

shopping carts and one database for orders. Whenever the application software times out or encounters a fatal error accessing the database, it writes a common piece of local state, such as a shared memory segment, with a flag indicating a problem on the target database. Then other threads of the application, before attempting to make a call against the database, would check this state object, and forego the database call in order to avoid encountering the same error.

In such an example, failure of the ordering database would not cause hanging or termination of the application thread, and normal use of the shopping cart could continue. Conversely, if the shopping cart database became unavailable, then those users already in the order pipeline would be able to complete their orders without the application failing.

Another way to avoid failure-prone database access is to employ caching. An application should not keep going back to the database for the same data over and over again. For instance, a well-engineered application might only query a currency exchange rate once an hour. It would then maintain that data in memory, and use it to service many requests instead of repeatedly and mindlessly querying the database. Beyond improving availability, caching has the added benefit of reducing load on the database.

In summary, there are many ways to improve a single database's availability. But the highest availability comes from thoughtful engineering of the entire architecture.

Jeremiah Wilton was the first DBA at Amazon.com. He now runs ORA-600 Consulting in Seattle where he specializes in high availability, complicated recoveries and seminars. Jeremiah's full whitepaper on high availability is available at <http://www.ora-600.net>.



Tim Gorman: 24x7 (a.k.a. 100% uptime) is not a myth. There is a production system running Oracle Rdb on DEC/Compaq/HP OpenVMS at the HP hosting center in Colorado Springs which has been available continuously since 1992 or so,

through hardware and software upgrades. So, 100% uptime is not widespread, but one or two examples do exist on this planet.

There remain important single points of failure within RAC.

24x7 might be considered a "myth" for the Oracle RDBMS, at least at the present time. Oracle RAC is primarily designed to be a scalability solution that (of necessity)

incorporates certain features that are construed (and marketed) to be "high-availability", such as automatic instance failure detection and recovery. But the presence of some important redundancies does not overcome the fact that there remain important single-points-of-failure within RAC, so it is not a high-availability solution by design. Oracle Data Guard is a solution truly designed for high-availability and the elimination of single points-of-failure, but I don't know if the product has been around long enough to bear out the designation of 24x7 over any appreciable period of time. This is something that time will tell.

Tim has worked with databases since 1984 and with the Oracle RDBMS since 1990. He is currently with SageLogix (www.sagelogix.com), a small group of consultants based near Denver, Colorado. He is an active member and past president of the Rocky Mountain Oracle Users Group (www.rmoug.org). Tim can be reached at tim@sagelogix.com.



James Koopman: Is a 24x7 database system a myth? Let me just say for the record, YES it is. At the root of the question is the definition of availability which has

the connotations of being ready for use, at hand, and accessible. With 24x7, that means 100% availability and completely accessible.

Now get the notion of a full database up and running out of your head. You see a database is composed of many pieces. There are the Oracle processes, networking components such as TNS, tables, indexes, and procedural code. If any of these pieces are unavailable does that mean our database system is unavailable? I tend to think YES. We have all administered databases that required us to fix some piece of data, rebuild an index, or restart our TNS listeners. Is this down time? I tend to think YES. Downtime by definition is a window of opportunity taken by administrators in order for them to fix something and directly affects availability.

We have all administered databases that required us to fix some piece of data, rebuild an index, or restart TNS listeners.

You see, 24x7 is not just a hardware issue but in reality is a user issue. Even the best DBAs, if honest, will tell you countless stories about how they have snuck in some form of maintenance that they are proud of because it went undetected by an end user. Outages will always occur in database systems. It might not be

today, it might not be tomorrow, someone may notice it, and most may not. But the fact is, it is impossible for us to have a truly 24x7 system when we take a look at availability from individual usage patterns, administrative tasks, and the end user perspective.

James is founder of Pine Horse, Inc. (www.pinehorse.com), and creator of (www.dbcorral.com). Over the years James has worked with a variety of database-centric software and tools vendors as strategist, architect, DBA, and performance expert. He is an accomplished author with articles appearing in publications including Oracle Professional, Database Trends and Applications, DatabaseJournal.com, and various vendor newsletters. James can be contacted at jkoopmann@pinehorse.com.



Chris Lawson: A goal of 24x7 is a fine objective for critical information systems, but I believe many businesses are fooling themselves when they claim they are meeting this goal. Ironically, the problems with most high availability solutions aren't really due to flaws in the technology, but how they are

implemented. In particular, I have noticed that "hot standby" systems are especially weak links.

In my experience, most high availability systems fail for these reasons:

Critical components are missed: In order for a failover solution to work, every part must failover properly (or have some type of redundancy). If any piece is missed, you don't really have high availability. Unlike school, a score of 99% correct isn't good enough--it gets an "F" just like the system that is only 1% correct. When your online system is down and customers are mad, management doesn't really care that you got it almost right. At one large insurance company, a critical server failed. As planned, the database failed over just fine, and was 100% usable. Unfortunately, a major part of the application didn't, because the designers neglected to include it in the process.

Incomplete testing of the failover process: In the above case, a simple failover simulation would have immediately uncovered the deficiency. Incredibly, technical management assumed everything would work fine--without testing. In another case I witnessed, a failover didn't work because file systems had been modified on one node (the current node), but not on the failover node. This flaw was only uncovered when the failover didn't work.

How do we design a system that provides this assurance: "People will never make a mistake"?

Overlooking likelihood of human error: By this I mean blunders made after the high availability system is implemented. I think this is the toughest problem to solve. The key focus in high availability solutions is usually reliability of the hardware. In reality, however, failures are often the result of human

error. Several years ago, I helped setup a Veritas Cluster Server high availability solution. We tested everything many times. Failover worked flawlessly. A few months later, however, a real outage occurred, and the failover didn't work! Investigation showed that the system administrator (a very competent individual by the way) had recently deleted a directory that he thought wasn't needed. Of course, the directory contained files vital to VCS. The most obvious answer to this problem is to add strict configuration control.

In summary, true 100% system uptime will likely remain a tough goal, but proper planning and testing can increase the uptime dramatically. To really achieve 24x7 however, we need to focus on the root cause of most system outages. The really difficulty problem is not how to increase

hardware reliability, but how to increase human reliability. How do we design a system that provides this assurance: "People will never make a mistake"?

Chris is an Oracle DBA consultant in the San Francisco Bay area, where he specializes in performance tuning of data warehouse and financial applications. He is a frequent speaker at the Northern California Oracle Users Group, and has written for a number of publications such as Oracle Internals, Exploring Oracle, SELECT, Oracle Informant, and Intelligent Enterprise. Chris has held a variety of positions in the IT field; ranging from systems engineer to department manager; and is an instructor for the University of Phoenix. He can be contacted via www.oraclemagician.com.
