



# CHAPTER 5

## Database Down! Bring It Back Alive!

## 202 Everyday Oracle DBA



Whenever I hear “database down” I think of the 1978 movie *Gray Lady Down* where the nuclear submarine Neptune sinks after hitting a freighter and the folks up top have 48 hours to rescue the crew. It’s at those times that I can hear the sound effects associated with submarines in my head, as well as the background music, and a sense of fear grips my heart. Okay, so I should get a life, not just rent one from Blockbuster. But the analogy is still close.

You rarely have the luxury of 48 hours when it comes to rescuing your database, however. What usually happens is you have a dozen or so managers in suits standing in your cube doorway at six in the morning wondering how long it will be before the database is back up and running. Rather nerve-racking. So in this chapter, we’ll look at some of the best ways to get those pesky managers out of your cube as gently and quickly as possible.

### Database Down

While it really doesn’t happen often, there are times when your database does crash and burn and you find yourself looking at a SQL prompt that says a shared memory realm doesn’t exist or that Oracle is unavailable. Of course, this is when you’re lucky enough to find out that the database is down before your users do. When they find out first, you find yourself scrambling to answer questions while furiously typing and misspelling words that you know you know how to spell (like `sqlplus`, or `sysdba`). I’m a pro at consistently misspelling “select” any time I find myself under pressure, either from me trying to get things back under control as quickly as possible or due to those dozen pairs of manager eyes boring into the back of my head.

The most important thing when confronted with a down database is to get it back up and running. Then afterward, you need to figure out how, if possible, to keep it from happening again.

### Restarting

The first thing to do is check the alert logs. See if anything jumps out at you as a reason for the database being down. For example, did one of the DBAs in your shop do maintenance last night and forget to bring the database back up? Not that this would ever happen, but just for grins check and see if

## Chapter 5: Database Down! Bring It Back Alive! **203**

the end of the alert logs might show this is the case. Of course, if someone with just the wrong access decided to go out and kill a bunch of background processes, maybe because they seemed to be taking a lot of the resources on the box, and it was too late at night to bother the DBA with stuff like that, there won't be anything in the alert log or, at best, not much of one.

If there's nothing glaring in the alert log that tells you something horrible happened (like someone deleted a bunch of data files or maybe all of the control files are gone) simply try restarting the database. You might be surprised. Whatever caused the database to crash might turn out to be a simple and transient thing, and your database will be revived simply by using the startup command.

The most important thing to users and to management is getting the database restarted. However, sometimes a restart will wipe out important information, including evidence of what happened, and you won't be able to find out what happened. Try to at least dump out the contents (if possible) of some of the v\$ views to help in your analysis before restarting. Once the database is accessible, you can worry about getting to the root of the problem (sometimes referred to by cranky upper management as Root Cause Analysis, or RCA).

### **If It Doesn't Start**

Okay, you've tried the simplest and most straightforward solution—simply restarting the database—but it didn't start. Now what?

Well, you start what could arguably be seen as the fun part of being a DBA (if you're a truly warped individual, which I am, and if you enjoy a real challenge). You have to try to figure out why it isn't starting (hopefully as quickly as possible) and get it back up and running.

### **If It Doesn't Stop**

Yeah, sometimes the database gets stuck ... up. Not only stuck in the up position, but since "stuck up" implies that you were trying to shut down the database, no one can now connect to it because a database shutdown is in process.

It's normal for shutdown to sometimes get stuck, that's a given. Since "normal" implies you're willing to wait for all connections to disconnect, there have to be connections out there waiting for someone to do something. The solution? Kill off all sessions connected to shutdown immediate or (gulp) do a shutdown abort.

## 204 Everyday Oracle DBA

Okay, so that's no big deal, right? Sure, but what happens when shutdown immediate gets stuck? The emn0 background process sometimes forgets it's running, goes to sleep, and just won't wake up. Sometimes Oracle weirds out and refuses to be cooperative for some other reason.


If shutdown immediate gets stuck, there are only two ways to bring down the database. One is to kill -9 on Unix, or kill Task Manager in Windows. This is usually used only as a last resort, or by overly anxious operators with just a little too much knowledge. The other is shutdown abort. Yes, this is a valid way of shutting down the database. Of course, so is pulling the power plug or pressing the reset button, but Oracle will assure you that it's a valid shutdown method. It still makes my stomach knot, but I've actually done it. Of course, I start up as soon as the database is down and then perform shutdown immediate again so the shutdown is in as stable a state as possible.

### Finding Out Why

Okay, so we haven't gotten it started, and we need to find out not only why it won't start, but what brought it down to begin with. Chances are that, when you tried to bring up the database, if there was an issue that caused it to not come up, it was either sent to the screen or the alert logs and you have some idea of what didn't work and maybe what the error was that meant it didn't work. This is an excellent place to start. You can look in the manuals to find out what the error means, or you can make use of some of Oracle's valuable resources to help you figure out what happened and where to go from here.

### The oerr Utility

The Oracle Error (oerr) utility is provided to you, free of charge on Unix and Linux, as a way to help you quickly find information about the errors you're seeing in your database without having to resort to searching with your favorite search engine or running to the manuals to find out what exactly an error message means. By using the following command at the operating system prompt, you can quickly see what Oracle is trying to tell you.

```
 oerr <prefix> <number>
```

The prefix is usually the three characters preceding the hyphen in the error that's displayed (perhaps ORA, MSG, PLS, or something else) while the number part of the command is the number to the right of the hyphen. For

## Chapter 5: Database Down! Bring It Back Alive! 205

example, did the dreaded ORA-00600 appear in your alert log? If so, you might be able to get more information by running the following command:

```
oerr ORA 00600
```

In this case, as you can see next, the information you get back will probably be less than useful, as ORA-00600 can cover many different kinds of errors, but you can still get some idea of how the command works and the format of the output, although to allow it to fit within the confines of the book, I had to take liberties with a couple of the line breaks.

```
$ oerr ora 00600
00600, 00000, "internal error code, arguments:
[%s], [%s], [%s], [%s], [%s], [%s], [%s], [%s]"
// *Cause: This is the generic internal error number for
// Oracle program
// exceptions. This indicates that a process has encountered an
// exceptional condition.
// *Action: Report as a bug - the first argument is
// the internal error number
```

While the original Oracle-provided version of this utility runs on Unix but not Windows (because it makes use of the AWK command, and not every Oracle installation assumes you've installed MKS Toolkit or CYGWIN to emulate Unix on Windows), there are ways to make a similar utility run in a Windows environment. This will make you very happy if you happen to be one of those who have Windows environments without access to a Unix alternative on which to run the command.

I like to tell SAs who want to debug errors how to use oerr. They can then look up errors themselves, which is a big help if they can field even a few user problems.

One notable utility that ports the functionality of oerr to Windows was written by Yong Huang, and uses Perl as the vehicle that allows the script to work. This script is well commented and is freeware. Don't feel like typing all that much? You can get the script from Yong's web site (<http://rootshell.be/~yong321/freeware/Windowsoerr.html>). Yong maintains an interesting site with a lot of useful information on Oracle as well as on Windows, and offers a truly geeky program that allows you to use a mouse in DOS that was written in assembler (this may not be as useful as it once was, but it's fascinating

## 206 Everyday Oracle DBA

from the perspective of someone who really doesn't have much of a life—  
not that I find reading code to be a fun pastime, you understand).

```
#!/perl -w
#Perl script in Windows simulating the Oracle oerr utility in Unix
#I assume you installed Perl for Windows on
#the same machine you installed
#Oracle Documentation. This script should be run by oerr.bat.
#See the following URL.
#This script is published as freeware at
#http://rootshell.be/~yong321/freeware/Windowsoerr.html
#(C) Copyright 2000,2004 Yong Huang (yong321@yahoo.com)
#Please modify $dir, $colon and select $fsp, and $lsp.
#On your computer, open the Oracle Documentation homepage with
#a Web browser
#and find the error message page. E.g. for Version 8.1, it may be
#Oracle8i Server -> Oracle8i Error Messages (in section References).
#Find the URL for the message page
#(if it's in an HTML frame, View Frame Info
#in Netscape, Properties in IE). Take the string before
# "\TOC.HTM". Follow my
#format below. E.g., on my machine, the URL for error message
#Table of Contents
#page is
#   for 9.2 Enterprise Ed
#   file:///C:/ora9idoc/server.920/a96525/toc.htm
#   C:\ora9idoc\server.920\a96525\toc.htm in IE
#$dir shown next should use "/" not "\", no "/" at the end
#(Additional work is needed if you use 8.1.5 documentation)
#$dir="C:/ora8idoc/server.817/a76999";
$dir="C:/ora9idoc/server.920/a96525";
#$dir="C:/ora10gdoc/server.101/b10744";
#For Oracle8i only. Ignore this paragraph if your doc is > 8i.
#$colon=":";
#Let's say you look up ORA-00600. Click it. If you see
#   ORA-00600 internal error code...
#please leave the above line commented out
#so $colon will not be set.
#For very old versions, you may see
#   ORA-00600: internal error code...
#then uncomment $colon=":".
#Error message toc.htm page searches individual error message files.
#We need to
#collect all those file names. $fsp is the file search pattern.
#If Oracle8i (except 8.1.5), use this
#$fsp='CLASS="TitleTOC"><FONT FACE="Arial, Helvetica, sans-serif"><A
```

## Chapter 5: Database Down! Bring It Back Alive! **207**

```
HREF="([^\.]+\\.htm)';  
#If Oracle 9i, use this  
$fsp='class="TitleTOC"><a href="([^\.]+\\.htm)';  
#If Oracle 10g, use this  
#$fsp='<h2><a href="([^\.]+\\.htm)';  
#In each error message file, we identify the line that has  
#$code you intend to  
#search for e.g. ORA-01555. Different versions have different  
#HTML markup. Pick  
#a line search pattern below for your version.  
#$lsp="<STRONG>"; #if Oracle 8i (except 8.1.5)  
$lsp="<strong>"; #if Oracle 9i  
#$lsp="^<dt>.*?"; #if Oracle 10g  
##### No need to modify beyond this line. #####  
##### But hacking is welcome. #####  
if ($#ARGV!=1)  
{ print "Usage: oerr facility errornumber  
  where facility is case-insensitive and not limited to ORA  
  Please open oerr.pl with a text editor and modify  
#\ $dir if you haven't done so  
  Example: oerr ora 18\n";  
  exit 1;  
}  
open TOC, "$dir/toc.htm" or die "Can't open toc.htm: $!";  
while (<TOC>)  
{ if (/ $fsp/)  
{ $allfile{$1}=1 if defined $1;  
  #use hash to ensure uniqueness  
  #Last version uses array which contains some  
  #  filenames more than once  
  #That's very bad when running against Ver. 7.3.4 Documentation  
  #push @allfile,$1 if defined $1;  
  }  
}  
close TOC;  
$facility=uc $ARGV[0];  
$code=$facility."-".(sprintf "%05d",$ARGV[1]);  
#e.g. ORA-00600, IMP-000001  
#05/21/00 note: Found another inconsistency in Oracle doc:  
#Image Data Cartridge  
#Error Messages use "," instead of ":" after facility-errorno,  
#e.g. "IMG-00001,"  
#This is the only one I find that uses anything other than ":".  
#If you need  
#"oerr img [errorno]", better comment out the line  
# $colon=":" which may
```

## 208 Everyday Oracle DBA

```
#introduce the problem described in last paragraph.
#Inconsistent documentation
#style always causes trouble.
$code .= $colon if defined $colon;
$flag=0;
#print join("\t", sort keys %allfile), "\n"; #for debug
foreach $file (keys %allfile)
  { open INP, "$dir/$file" or warn "Error opening $file: $!";
    while (<INP>)
      { exit 0 if ($flag==1 and /$facility-/);
        #if (/<strong>$code/i) #Actually 9i uses <strong>, 8i <STRONG>.
      if (/${lsp}$code/)
        { &rawprint;
          $flag=1;
          #print "This is in file ".$file.". \n"; #for debug
        }
        elsif ($flag==1)
          { &rawprint;
            }
        }
      close INP;
    }
sub rawprint
  { s/<.*?>/g; #de-HTMLize
    print unless /^$/;
  }
```

There are, of course, other versions of this kind of utility, but I really like the way this one was written and the perspective Yong takes regarding his code.

The oerr won't actually tell you what caused the error in most cases and won't likely provide you with information on how to fix the problem, but at 3 A.M. when you're freezing in the server room trying to get out to OTN (<http://otn.oracle.com>) or Tahiti (<http://tahiti.oracle.com>) and can't, it's a handy little tool to have. I don't know about you, but for the life of me I can't remember the difference between an ORA-12345 and an ORA-01234 without a little help. For what it's worth, oerr says ORA-12345 indicates a lack of CREATE SESSION privileges, while ORA-01234 indicates someone is trying to end the online backup of a file that is busy.



## Chapter 5: Database Down! Bring It Back Alive! **209**

### **ITar**

ITar is a word that can strike fear into the heart of the most fearless DBA. An ITar is the Internet trouble action request, your personal help line to someone in Oracle Support. But there are times when Metalink can be your best friend. What, you may ask, is Metalink? Metalink is the online support venue for Oracle licenced users. You access it at <http://metalink.oracle.com> and you need to have a valid CSI number (your service number) in order to acquire an account. Believe it or not, there are people who have to support Oracle databases without the luxury of having their own CSI number and therefore without access to Metalink.

These can seem like a lot of trouble, but they can be worth their weight in gold if just one of your issues gets resolved in a reasonable amount of time (as compared with your struggling to figure out something on your own). The analysts who get assigned to your ITars have an internal knowledge base at their fingertips from which they can draw nuggets of wisdom that would often take you an eternity to stumble upon yourself in your troubleshooting.

Metalink really is a very good resource for troubleshooting your database issues. They have a search facility that provides you with answers or ways to think through issues. Forums are also available where you can post questions and issues, and talk with others who've experienced similar problems. Often Oracle employees monitor these forums and answer questions if they don't believe that support's intervention is warranted. Of course, if you have truly unique situations and you're posting questions that are so complex and unusual that no one else could possibly have the same issue, the question is usually met with dead silence or the suggestion that you log an ITar.

Note 166650.1 from Metalink offers valuable information on how best to work with Oracle support. I highly recommend that while trying to resolve an issue yourself, you should log an ITar. This way not only will you have someone helping you defuse the situation, but you'll be able to maintain a running dialog of what you've tried and what Oracle suggested so that the next time you find yourself in a similar situation, you'll have a starting point to fall back on.

It's important to note that Metalink is a wonderful tool, but it's a little quirky. Putting too many phrases into the search criteria will cause you to not get back any hits, even though the general search is for any of the words. Also, when you're creating an ITar, it's important that you know exactly where your cursor is if you're considering using the backspace key on your keyboard. If your cursor is in the wrong place and you backspace, you've

## 210 Everyday Oracle DBA

just lost all your hard work and will have to enter information into the form again from scratch. I hate typing, and more than anything I hate having to retype something I already typed in.

Something else to remember: include as much information in your ITar as you can. This will provide the analyst with the background she needs to start working on your problem. Don't think something is relevant? Don't be too sure.

Remember, you've been the one pulling your hair out; your support analyst has no way of knowing what you know. He or she hasn't been the one staring at your screen for hours, poring over log and trace files, trying and failing at every turn. Explain the problem the way you might to your mother or the way you might explain *exactly* what you want your child to do—again, include lots of details, in writing, so both of you know what you're talking about. If you have doubts about whether or not you've given enough detail, have your favorite trusted developer or systems analyst look at your explanation of the situation and see if they can make heads or tails of it from your description. The analyst who gets assigned your issue isn't familiar with everything you've tried or with the details of your systems; give all the details you can.

Your analyst can weed out the information he doesn't need as you go, and it's less frustrating for you if you put it in to begin with than to have to update a severity-one ITar with requested information when your database is still not functional. And don't be afraid to make your ITar a severity one if you have a down database that's impacting your business adversely. Support frowns on too many severity-one ITars for test or development databases, but if it is impacting your ability to move up code to production that's needed for your business, or it's keeping you from fixing a production database so it doesn't crash, they're usually very understanding.

There are two other notes relevant to ITar creation that are very handy to have as reference material. Note 280603.1 tells you how to close an ITar (or service request). Yes, always close a service request. If you don't, it will remain in your analyst's queue until she gets frustrated and soft closes it for you for lack of attention. If you know what fixed the issue, put the resolution in the verbiage of why you closed it. This will help the analyst help the next guy who has a similar problem, and it will also be in the ITar so you can retrieve the information later if you find yourself looking at the same issue again. Note 235444.1 provides you with much needed information on how to prepare information and systems for a test case with Oracle support.

## Chapter 5: Database Down! Bring It Back Alive! 211

Sometimes when Oracle support people can't reproduce a similar situation in the lab, they need your help to contrive a situation where your issue occurs. When trying to help your analyst reproduce your situation in her lab, it's important you provide her with step-by-step information. While you probably aren't thinking in these terms when you're "in the moment" so to speak, it's important to remember that your analyst doesn't have your background information in this situation and will be happy to have all the help you can give her. This note walks you through all the steps needed to help Oracle help you.

It's important to remember that, as with any support or help desk functionality, it's all potluck when it comes to the analyst you're assigned and their ability. Your analyst may be great with some issues, but not yours. So try your best to work within the system and deal with the analyst you're assigned, but if they start reading you the documentation, run screaming to their manager and request an analyst who can help you. Don't expect your analyst to be a magician or a mind reader. They each have their own strengths and weaknesses. Though they're likely to have more resources than you, they don't know everything and they certainly don't know all of the things that you have already tried unless you tell them. Not only that, but they're not just working on your issue; they're probably working on a dozen ITars simultaneously, and keeping all the details straight can be a true juggling act. They're human just like you. Bear with them. Regardless, the help system often works, and if nothing else, it gives you a second set of eyes that might see what you're missing.

Oh, and don't be afraid to escalate your ITar if you aren't getting anywhere with the analyst to whom it's been assigned, and you've honestly tried to work things out. Sometimes an analyst is too busy to spend the time that your management thinks needs to be spent on an issue, or he simply doesn't have the relevant experience to provide you with the information necessary. If you've been updating your ITar and answering analyst questions to the best of your ability yet are still getting nowhere, don't be afraid to escalate the ITar, and the issue, to a higher-level analyst. You will quickly lose credibility, and get a reputation with support as a reactionary whiner if you try to escalate issues that you haven't been working actively, or that simply don't justify continuous support (like requesting installation media or information gathering), but if you're confident you're justified in escalating an issue, do so. Escalation is often looked on as a doctor's second opinion or a mechanic's assurance that the problem with your car really is the water

## 212 Everyday Oracle DBA

pump and not just a lack of antifreeze. While Note 120817.1 isn't necessarily designed with the average DBA in mind (it is, after all, entitled "Oracle Applications Welcome Basket"), it has very relevant information in it that you can use—most particularly in this case: how and when to escalate an issue.

As an aside, ITars are no longer just a way of having troubles taken care of or having questions answered. They're now the mode of choice to request from Oracle, whether concerning an enhancement of the database or one of the products and applications surrounding it. While it seems like it's taking human-to-human communication a step further out of the Oracle Support equation, it does allow you to be very specific and have a written record you can use to determine what your enhancement request is doing.

## Tools

So what tools are available to you to help determine what's causing all your heartache? What can you use to tell you why your bright shiny database went belly up? This section will give you some places to look, as well as some places you might not have thought about.

## Alert Log Monitors

One of the most obvious places to look for problems is in the alert logs. Remember back a couple of chapters when we were looking at making the alert logs into external tables so SQL could be used to scavenge information from them? Well, that isn't really practical if your database is down, so you're going to have to search through them manually. If you want to be the eternal optimist and consider your downed database as lucky, you're fortunate, thanks to the alert log's linear nature. Whatever caused the problem with your database, if it's anywhere in the logs, is likely to be found near the very bottom of the file. So running "tail" on the file for the last couple dozen lines might give you a clue to the problem (if you're running on Unix or Linux), or simply scrolling to the bottom on Windows may do the same.

If you have CRON or AT working on your system, you can set up scripts to run under these utilities as alert log monitors so the system will alert you when errors start showing up in the alert logs. One thing that's nice to have is a little script that checks the alert logs for ORA errors, e-mails you (or your cell phone or text pager) when an error occurs, and then renames the alert

## Chapter 5: Database Down! Bring It Back Alive! **213**

log to something like alert<sid>.date so you can easily find, near the bottom of the file, the error condition when you go looking for it. In this way, you not only keep your alert logs at a reasonable size, you have easy access to the error condition. Running this little script every minute or every ten minutes or every hour will allow you to catch many error conditions either before they crash your database or very near the time of the crash. They don't require that the database be up and functional at the time.

Oracle Enterprise Manager (OEM), which has gotten more and more useful over the past few releases and is now more of a tool than just a pointy clicky way to do your job, can also be set up to alert you whenever there are error conditions occurring in your database, or whenever your database is down. Sometimes these features are short circuited by the fact that the database has indeed crashed, but at least they can be set up to monitor crashes and you will (lucky dog that you are) be the first to know that your database is down.

### **Database Monitoring**

OEM is also a useful tool for just monitoring the database. You can set it up to alert you when your data files are getting low and when you're in danger of your application crashing because either it can't get to the database or the database won't let it process information. While many of these issues aren't exactly database down, they can appear to be to your user community.

You may be able to see that the database is, indeed, up, but your users might be reporting that they cannot access the database. This means that the database is down as far as they're concerned. If the database isn't responding to requests, if the listener just isn't listening, and web listeners are not listening or Oracle Names or LDAP servers are not responding the way they should, then in effect the database is down.

Whatever you use to monitor, it needs to be able to determine if the archive log destination directory is filling up. It should be able to determine if objects are getting close to their maximum number of extents or to their maximum available size on disk. Has the maximum number of user connections been reached? While with the use of locally managed tablespaces this should never be an issue, there are still organizations running with dictionary-managed tablespaces because someone somewhere heard about a case where locally managed tablespaces performed worse than dictionary managed.

## 214 Everyday Oracle DBA

If you want to see if your issues are connected to free space in the tablespaces of your database, you can set up this script to run automatically (or on command) to check space issues.

```
select tbs.tablespace_name,
tot.bytes/1024 total_bytes,
tot.bytes/1024-sum(nvl(fre.bytes,0))/1024 bytes_used,
sum(nvl(fre.bytes,0))/1024 free_space,
(1-sum(nvl(fre.bytes,0))/tot.bytes)*100 pct,
decode(
greatest((1-sum(nvl(fre.bytes,0))/tot.bytes)*100, 90),
90, '', '*') pct_warn
from dba_free_space fre,
(select tablespace_name, sum(bytes) bytes
from dba_data_files
group by tablespace_name) tot,
dba_tablespaces tbs
where tot.tablespace_name = tbs.tablespace_name
and fre.tablespace_name(+) = tbs.tablespace_name
group by tbs.tablespace_name, tot.bytes/1024, tot.bytes
order by 5, 1 ;
```

This script will show you all tablespaces, their free space, used space, and available space, flagging those tablespaces that have less than 90 percent free space with an asterisk in the last column. While this doesn't bring anything back, it can help you keep the database from going down to begin with.

But tablespaces don't just run out of free space. Sometimes there's sufficient free space in the tablespace, but you could be nearing the maximum usable number of extents available to the object if you're using dictionary-managed tablespaces. You'll want to watch numbers as they decrease in this case because once the available number of extents reaches 0, you'll begin to get errors in your application, and users will notice and start complaining.

```
col name noprint new_value dbname
col value noprint new_value block_size
set verify off
SELECT db.name, value
FROM v$database db, v$parameter pr
WHERE pr.name = 'db_block_size' ;
SELECT
```

## Chapter 5: Database Down! Bring It Back Alive! 215

```
fre.tablespace_name,  
SUM(fre.bytes/1024) free_space,  
COUNT(*) num_free,  
MAX(fre.bytes/1024) largest,  
/*AVG(fre.bytes/1024) avg_size,*/  
GREATEST(NVL(mnt.max_next_extent,&block_size),  
NVL(mni.max_next_extent,&block_size))/1024 grt_extent,  
SUM(DECODE(GREATEST  
GREATEST(NVL(mnt.max_next_extent,&block_size),  
NVL(mni.max_next_extent,&block_size)),  
fre.bytes), fre.bytes,  
TRUNC(fre.bytes/greatest(NVL(mnt.max_next_extent,&block_size),  
NVL(mni.max_next_extent,&block_size))),0) min_usable  
FROM  
dba_free_space fre,  
(SELECT tab.tablespace_name,  
MAX(tab.next_extent) max_next_extent  
FROM dba_tables tab  
GROUP BY tab.tablespace_name) mnt,  
(SELECT idx.tablespace_name,  
MAX(idx.next_extent) max_next_extent  
FROM dba_indexes idx  
GROUP BY idx.tablespace_name) mni  
WHERE  
fre.tablespace_name = mnt.tablespace_name(+) and  
fre.tablespace_name = mni.tablespace_name(+)  
GROUP BY  
fre.tablespace_name,  
GREATEST(NVL(mnt.max_next_extent,&block_size),  
NVL(mni.max_next_extent,&block_size))  
ORDER BY 6 desc,1 ;
```

The output from either of these scripts can easily be parsed using a shell script, or Perl, and scheduled using cron. If you don't rely on Oracle Enterprise Manager to automate the monitoring of your database, and you're running on a UNIX operating system, cron is a good way to automate monitoring.

Another critical part of monitoring is at the file system level. One of the most important file systems to monitor is the archive log destination. Another, depending on how diligent you are with cleanup, is the directory into which the logs are written. The following is a simple script you can use to monitor a file system (in this case called /archives) that sends you an e-mail whenever the file system reaches 90 percent full:

## 216 Everyday Oracle DBA

```
USED=`df -Pk /archives  
      | tail -1 | awk '{print $5}' | cut -d % -f 1`  
if [ $USED -ge 90 ]  
then  
echo Filesystem /archives is at $USED percent | mail  
you@your.co.com  
fi
```

Another simple script to run, this one to make sure your database is up and running, employs the time-honored tradition of using GREP to see if background processes are running.

```
running =`ps -ef |grep smon |grep <SID>`  
if [ $running != <SID> ]  
then  
echo DATABASE DOWN |mail you@your.co.com  
fi
```

## History

Okay, now, consider that you're going to start receiving information on crashes, space, and other monitoring information. What will you do with it? You can just react to the information when it comes, and then go on with your day-to-day activity, or you could start to compile this information, along with the steps you took to alleviate the condition. You can build yourself a maintenance schema for each instance, or create a central instance and into that repository store all of the situation that you come up against and what you did to rectify the situation. In this way, you can have a running history of not only what's happened in the database, but an affidavit that shows users and clients that you're both proactive as well as reactive, and that you do have a clue what's been going on in your database.

## Panic Mode

Panic mode is not a place where you want to be. Let's say you've tried to restart the database and found you can't simply start it with the startup.

Now what? What if someone dropped a data file or a whole tablespace without your knowing it? What happens if your users have production



## Chapter 5: Database Down! Bring It Back Alive! **217**

passwords and have managed to drop an interface table to Oracle E-Business Suite and they can't process payments for certain kinds of vendors? What if the sky is falling and Henny Penny and Foxy Loxy have, in their panic to clean up the server, dropped active database files but didn't tell you for fear of upsetting you, the database god or goddess? Relax. Take a deep breath and get a cup of coffee or tea and settle in for the adventure.

Panic isn't going to accomplish anything and it may well get you into more trouble than taking the extra time to approach the issue with a calm open mind. You are in control and you can do this.

If nothing else, there are people out there who can, and will, help you. <http://www.freelists.org/webpage/oracle-l> is the URL for a very useful and helpful listserv, the Oracle-L list. There are very knowledgeable DBAs on that list who are more than willing to help with problems and issues. I'm not sure about all of them, but many look on helping as a way to learn something new, or to help someone else not make the same mistakes they have. There's expertise on this list on everything from Perl with Oracle, to RAC, to tuning, to how to get the right results from the most horrendous query imaginable. These resources are available to you even when you aren't having database issues and just want to discuss differences in doing things, but when you're looking at a database full of helpless data and have no way to get to it, they can be a far more invaluable resource.

## Hot Standby Database

So, management wants to make sure that, if the database does go down, you have a backup database to use, a database that you can quickly bring back to life to return the organization to its previous condition. More detailed information on Data Guard and physical and logical standby databases can be found in Chapter 6. Suffice it to say that you can make use of Data Guard technology to make sure you have an available and viable database for your users.

## Problem Resolution

Okay, so you have a problem. Since you're the DBA, they'll at some point expect you to resolve it—which means someone is going to know you exist.

## 218 Everyday Oracle DBA

Keep in mind though, the faster the resolution, the less people who'll ever know there was a problem.

### No Oracle Connectivity

Okay, so you can connect from the server prompt on the database server, but the users can't seem to connect. Perform some obvious steps first. Check the log files for your connectivity (Net8, SQL Net, *name du jour*). Ensure your environmental variables are set and inherited by the connection running the listener. These kinds of errors are almost always caused by a misconfiguration in the user's environment.

Use Sql\*Plus both from your client computer and at the server level to connect to the database with a valid user ID and password as well as the service name. Make sure no one has suddenly decided they needed to update their own version of the client software. Sometimes installing a new version, regardless of reason or version, can cause brand-new issues. Sometimes the issue concerns an incompatibility between two versions of the client software installed on the same computer, while sometimes versions of tnsnames.ora and other configuration files have gotten overlaid by the software. Either way, it's always because the installation of the software has messed something up and the user is often very reticent to fess up to having changed anything. Have them check their PATH variables as well. Sometimes something has gotten updated, changed, or deleted that used to be in the PATH when the user was last able to connect.

Tnsping <sid> from the database server, and then do it again from a client machine. Ping the database server from a client machine. Telnet from the client machine to the database server. Even if you can't log in, as long as there are no errors returned from the telnet attempt, there's little chance there aren't any network connectivity issues from where you are to the server. Now have the users who are raising issues do the same thing.

Start the listener. At worst, you'll find that the listener is already running. At best, you'll discover the listener is down and that restarting it will fix the problem.

### Database Links

One of the most aggravating issues that DBAs work with are database links that just don't seem to work. Sometimes they're user-defined database links that used to work but that now have all of a sudden stopped, or they are

## Chapter 5: Database Down! Bring It Back Alive! **219**

database links that users are trying to define that simply won't work as they should. One common problem with database errors is when you think you should be connecting just fine, but you start getting ORA-12154. Often, this is caused by the user misunderstanding which configuration files are being used and assuming that the local version of the file is what's currently employed. When a client issues a database link connection command, the address is not resolved on the client; it's resolved on the server that the connected user's session is connected to.

For example, let's assume that the client has the following in their `tnsnames.ora` file:

```
DEVDB = (DESCRIPTION=  
  (ADDRESS= (PROTOCOL=tcp) (HOST=myhost1) (PORT=11234))  
  (CONNECT_DATA= (SID=DEVSID))  
)
```

And the server has the following:

```
DEVLDB = (DESCRIPTION=  
  (ADDRESS= (PROTOCOL=tcp) (HOST=myhost1) (PORT=11234))  
  (CONNECT_DATA= (SID=DEVSID))  
)
```

If the user creates the following database link:

```
SQL> CREATE DATABASE LINK MYDBLINK  
CONNECT TO scott IDENTIFIED BY tiger USING 'DEVDB';
```

the link is likely to fail and will need to have its definition changed to USING DEVLDB. If there is a `tnsnames.ora` file on the server, it should either be in the `%ORACLE_HOME%\network\admin` directory, or a symbolic link to the file should be located there. If one doesn't exist there, copy one to the directory and modify it so it contains the appropriate entries.

Verify the information on the link from the DBA views.

```
SELECT DB_LINK, HOST FROM DBA_DB_LINKS;
```

If all else fails, rename the `sqlnet.ora` file on the client. If that doesn't work, try renaming the one on the server. Double-check the entries in the `tnsnames.ora` file. If you're still getting nowhere, and the database link still

## 220 Everyday Oracle DBA

fails, try pretending that the tnsnames file doesn't exist anywhere and build the tnsnames entry into the database link itself.

```
CREATE DATABASE LINK MYDBLINK  
CONNECT TO scott IDENTIFIED BY tiger  
USING '(DESCRIPTION= ADDRESS=(PROTOCOL=tcp) (HOST=myhost1) (PORT=11234))  
(CONNECT_DATA=(SID=DEVSID))';
```

By defining the database link in this manner, you'll not only be sure about what user ID and password the link will connect as, but you can be certain about what server definition will be used for the connection and what entries are on any given tier (since the person defining the link likely won't be the only one using it) and that the link will function.

## RDA

Oracle Remote Diagnostic Agent (RDA) is a set of scripts, customized to each platform, that are designed to provide information on the overall Oracle environment and assist Oracle Support with problem diagnosis.

While Oracle Support encourages the use of RDA as a means of gathering information so they can debug issues, it can help you in the same regard. Use of this tool greatly reduces ITar resolution time by minimizing the number of requests from Oracle Support Services for more information. If for no other reason, it's a beneficial tool that should be used whenever possible prior to opening an ITar, and as a way of debugging an issue yourself.

For a list of all available RDA versions and platforms, please see Note 175853.1 on Metalink. Currently, it's supported on VMS, Windows, Solaris, HP-UX, Tru64, AIX, SuSE, and Red Hat Linux. Naturally, it can be adapted to other platforms with only a little tinkering. Errors will indicate utilities that aren't supported on different platforms.

RDA collects useful information for overall system configuration as well as data that's useful for corrective issues related to the following products:

- Oracle RDBMS Server
- Oracle RAC Cluster
- Oracle Application Server (*iAS 1.0.2.x, 9iAS 9.0.2.x/9.0.3.x/10.1.2, HTTP Server*)

## Chapter 5: Database Down! Bring It Back Alive! **221**

- Oracle Management Server and Intelligent Agent (Grid Server, Agent Server, DB Control)
- OLAP products (Express Server, Financial Analyzer, and Demand Planning Server)
- Oracle forms and reports
- Oracle networking products

### Test Cases

Okay, so maybe you need to provide Oracle with a test case or maybe you just want to have a place where you can re-create the database so you can figure out why things broke.

Cloning is a good way to not only provide test cases on cleansed data but also to help you fix the problems in a production-like test bed.

Copy the code tree for the database binaries along with the data files and all data associated with the database to a different location or a different server. Own them as another user so you can be sure you don't have any way to mess up your other databases. Change the values in \$ORACLE\_HOME/rdbms/lib/config.c to reflect the new owner of the binaries and relink the Oracle binaries to work in the new location.

```
cd $ORACLE_HOME/bin  
relink all
```

If you're on the same server, you'll need to rename the database. If you're on a different server, you could start monkeying around, but feel free to leave the database name the same. I, personally, am never that confident and always change the name to something outlandish so that I know beyond a shadow of a doubt where I am at the time.

### Summary

Recovering from disasters isn't much different than recovering from any backup. Practice when it comes to the art of recovery is important. Clone your production database to a backup location and use it as a practice arena so you can practice fixing things as they break. Having your own private

## **222** Everyday Oracle DBA

playground is often one of the best tools a DBA has. If you can break a database in every way imaginable (or better yet, have someone else think up ways to break it), and then recover or restart or do whatever is called for in the given situation, you'll better know what to look for the next time your database goes down.