CONGRATULATIONS! YOU'RE THE NEW DBA. DON'T PANIC!

Greg Loughmiller, BellSouth Technical Services, Inc
Rachel Carmichael, GetMusic LLC

ABSTRACT

Yesterday you hadn't even seen Oracle. Today you're the Oracle DBA. Before panicking, stop and take a deep breath. This presentation provides a step-by-step plan to understand the database and manage problems before they occur. The presentation provides valuable information to position a production environment that welcomes the new world of Oracle.

INTRODUCTION

All too often a DBA is thrown into an environment and expected to “hit the ground running”. Whether you are a newbie DBA or an experienced one, there are standard steps you can take to quickly come up to speed on the environment, applications and responsibilities you now serve.

LEARNING THE ENVIRONMENT

First things first. Before you can learn anything about the daily, weekly and monthly procedures you will need to follow, you need to learn about the world in which they exist. The environment is composed of the database, the hardware, operating system and applications. We will discuss each of these in turn.

THE ORACLE DATABASE ENVIRONMENT

As a new DBA, or a DBA to a new environment, it is imperative that you know all that you can about your Oracle Software Installation. You will need to know:

- How many versions of the Oracle Software are installed
- Where each version is located
- What products are installed for each version
- Which databases are using which version of the software

The first task in learning your environment is to find all of the installations of the Oracle software. This takes a little investigative work. Before we do that, let’s define a necessary environment variable used by all Oracle installations. The environment variable *ORACLE_HOME* will define the location of the software used for a database instance.

One way to find the location of the Oracle software installation is to see if the *ORACLE_HOME* environment variable is already defined. In UNIX you can do:

```
Unix host> echo $ORACLE_HOME
/app/oracle/product/8.1.6
Unix host>
```

The problem with checking for the *ORACLE_HOME* environment variable is that it will only show the one Oracle installation that you are pointing to at the time. If you run multiple versions of Oracle, you will not be able to see them. A better place to start is to find the *oratab* file on the machine and review its contents. You can find the file using

```
find . –name oratab -print
```

There should be an entry for each database and its associated *ORACLE_HOME* location. This will tell you which databases are using which versions of the Oracle Software and the location of that *ORACLE_HOME*. This file also contains a flag to indicate whether or not the database should be automatically started when the system starts up. A sample oratab file is

```
# This file is used by ORACLE utilities.  It is created by root.sh
# and updated by the Database Configuration Assistant when creating
# a database.

# A colon, ':', is used as the field terminator.  A new line terminates
# the entry.  Lines beginning with a pound sign, '#', are comments.
#
# Entries are of the form:
#   $ORACLE_SID:$ORACLE_HOME:<N|Y>:
#
# The first and second fields are the system identifier and home
# directory of the database respectively.  The third filed indicates
# to the dbstart utility that the database should , "Y", or should not,
# "N", be brought up at system boot time.
#
# Multiple entries with the same $ORACLE_SID are not allowed.
#
#
sampledb1:/app/oracle/product/7.3.4.0.1:Y
sampledb2:/app/oracle/product/7.3.4.0.1:N
proddb1:/app/oracle/product/8.1.6:Y
proddb2:/app/oracle/product/8.1.6:Y
```

As you can see, this system has two versions of Oracle installed, with 4 databases. Three of those databases will be automatically started when the system is booted up.

Now that you know where all of the Oracle software is, and which databases are using those versions of Oracle, it's time to see which products are installed. A list of installed products and the version can be found in:

- WINDOWS NT - $ORACLE_HOME\orainst\nt.rgs
- UNIX - $ORACLE_HOME/orainst/unix.rgs

Note that for Oracle 8.1.6 these files have moved to the install directory, not the orainst directory.

Now that we know where the software is installed, let's take a look at the database itself.

JUST WHAT IS THE DATABASE?

An Oracle database is a repository of information that is managed by the Oracle software. The Oracle RDBMS software will allow you to create, populate, alter and query an Oracle database. An Oracle database is constructed of many different types of physical files and memory structures. The database uses *datafiles* to store the information managed by Oracle. Oracle will use *control files* to understand where all of the datafiles are located, how they are used, and information about the current state of the database environment. The database also contains *redo log files* to log "transaction information" in order to reconstruct pieces of the repository of information in the event of a database failure.

Oracle uses *initialization files* to decide how to execute and manage the database. These files will also determine how much memory to allocate to the Oracle memory structure (the SGA - System Global Area) and how to use that memory while the database is up and running.

You will also hear the word *instance* as you discuss Oracle databases. An *instance* is the collection of the Oracle processes managing the *datafiles*, *control files*, *redo log files*, *initialization files* and *memory structures*. Some of these processes are:

- SMON – System Monitor, the overall controller of the database and memory allocated, manages the System Global Area
- PMON – Process Monitor, the controller of each individual process that is accessing the database and the memory allocated to the process
- DBWR – Database Writer, the process which writes information back into the datafiles
- LGWR – Log Writer, the process which writes the redo information to the redo log files

Depending on which parameters you set in the initialization files, there may be other processes running for your database.

Now that we've started discussing the initialization files, let's take a look at what they are and how they are used.

IMPORTANT FILES FOR ORACLE

Where do we start? There are several important files within Oracle. But let's start with some of the basics that you will need to learn as a new DBA.

ORATAB

The *oratab* file contains information about the databases running on that system. Each entry of that file will contain the Oracle System Identifier ($ORACLE_SID), the $ORACLE_HOME, and an identifier to indicate to an Oracle utility whether or not to start the database at the time the system is booted for each database on the system.

LISTENER.ORA

The *listener.ora* is used to determine which databases the Oracle Listener will be able to communicate with on the system. If you are going to have users connect to the database remotely, the Oracle Listener will need to know the specifics of the database. This file lists the names and addresses of databases that reside locally on that specific machine, the ORACLE_HOME for the database and the network port on which to "listen" for incoming requests. The listener will support requests to connect to those databases from other machines or clients.

TNSNAMES.ORA

The *tnsnames.ora* filel contains necessary information for users to connect to databases that are local and remote. Any database that a process will try to connect to must be defined within the *tnsnames.ora* file. The syntax of this file is well documented and beyond the scope of this paper.

SQLNET.ORA

The *sqlnet.ora* file contains specific sql*net configuration information. In general you should not change this file. However, there are parameters you can set that will enable you debug connection problems.

SNMP_RO.ORA AND SNMP_RW.ORA

These files are associated with the use of the Intelligent Agent and the Oracle Enterprise Manager. They contain configuration information about the instances that will be visible to the Intelligent Agent.

INIT.ORA

The *init.ora* file will contain the parameters that configure your database. It is also known as the *initialization file*. This file contains parameters which detail the database block size, maximum number of processes, memory usage and requirements, location of directories used by Oracle, and other base configuration information. This file tells Oracle how to configure the System Global Area (SGA) when the database starts. The SGA is the memory allocated to the Oracle database. Changes to the init.ora can either improve your environment, or prevent the database from coming up. It's a good idea to have a backup of this file prior to making modifications to the original.

SAMPLE DIRECTORY TREE OF AN ORACLE INSTALLATION

The Oracle documentation refers to an installation and its OFA (Optimal Flexible Architecture) directory structure. There have been many papers written discussing the pros and cons of this architecture and we will not rehash them here. We will however, list a typical OFA directory structure to give you better understanding of what OFA involves and which of the Oracle environment variables relate to this structure.

```
/
      app/
            oracle/ ----------------------------$ORACLE_BASE
                  admin/
                        /proddb1    ------------$DBA
                                    cdump/
                                    arch/
                                    bdump/
                                    udump/
                                    exp/
                                    log/
                                    pfile/
                  product/
                        8.1.6/----------------$ORACLE_HOME
                              . . . all directories containing the Oracle 8.1.6
                                     installed products

      usr/
            local/
                  bin/
                        coraenv
                        dbhome
                        oraenv
                        dbstart
                        dbshut
                        dbswitch
                        dbsid
      var/
```

```
opt/
      oracle/
            oratab
            listener.ora
            tnsnames.ora
```

THE HARDWARE AND OPERATING SYSTEM ENVIRONMENT

By now, you are probably overwhelmed with the things you have to learn and here we are, telling you that in addition to learning all about your database, we think you should know about your hardware and the operating system! Believe it or not, we are not crazy. We don't expect you to learn enough to become the system administrator, but a working knowledge of how the machine and operating system work can only help you when problems happen.

WHAT IS YOUR OPERATING SYSTEM?

Oracle is Oracle is Oracle, right? No matter what the platform, if you are running equivalent versions of Oracle, it will have the same functionality. So why do you care about your operating system?

Well, while within the database things are the same, at minimum the way you designate where your files go differs from operating system to operating system. NT uses "\" as a directory name separator, while Unix uses "/". Oracle is started differently on NT than it is on Unix.

You should be able to at least find your way around the system, so that you can find and read your trace and log files and check on the processes that are running. A basic understanding of the OS will allow you to make better decisions on where the datafiles should go, when you need to increase kernel parameters, and how much you can change them before running out of system resources.

Oh, and we suggest that you make friends with your system administrator. You never know when you'll need help with something on the system beyond your knowledge and it helps to be on good terms with the system administrators. We've found that liberal amounts of chocolate always help.

THINGS THAT ARE IMPORTANT TO ORACLE AND THE OS

Some of the parameters you set in the init.ora file have a direct effect on the amount of operating system resources that your instance will use. If you set the processes parameter too high, and you have not set the number of semaphores (on your Unix system) high enough, the database will not start. If you set the db_block_buffers and the log_buffers high, your SGA will be larger. The larger the SGA you have, the more system memory you need. And if you have not set the shared memory segment kernel parameter to a large enough value, you will be unable to start the database. As we said, you should make friends with your system administrator, because changes to operating system level parameters, especially those that force you to rebuild the kernel and reboot the system, are not things you can do without the system administrator's help.

KNOW THE DISKS AND THE LAYOUT

As a DBA, it is very important to understand the physical and logical layout of your storage system. It's not necessary to learn the details of what theVolume Manager does, but knowing which data files are on which physical disk drives is invaluable when it comes to tuning performance and managing problems.

We have seen an environment where a physical disk drive was broken into multiple logical volumes. This is pretty common practice. But this same disk was shared between 2 separate systems and databases. And each of these databases had some of their online redo logs and control files on the same physical disk. This created 2 problems for

the environments, both of which are probably obvious to you by now. The first problem was I/O contention and performance. Both databases were fighting for the same disk drive as they were using their separate online redo logs. The 2nd problem was that each system shared a single point of failure. If there was a problem with this single disk drive, it could have taken out both databases.

So get to know the data files used in your database, where they are located, and diagram that back to the physical disk drives. This documentation will help you in the future as you begin to tune for hot spots and I/O contention.

THE APPLICATION ENVIRONMENT

Last but not least, you need to know the application environment. You need to know what the end-users expect of the application in terms of performance and availability. The programmers will expect you to make changes on some sort of regular schedule.

WHAT ARE YOUR UPTIME REQUIREMENTS

Part of managing a database environment is the need to consider the requirements of the application and/or the business. Once the requirements have been defined, then work as a DBA begins. If this is a production system, find out when you can take the database away from the end-users for maintenance. One of the things that we have done in our environments has been to negotiate a period of time each week where we could perform database maintenance, or run cold backups. The DBA has to develop a strategy for Backup/Recovery that is based upon the requirements of the application and depending on whether or not it's a 24x7 environment, cold backups may not be feasible.

You have to know the critical business hours of the application as well. We have labeled those hours as the "Golden Hours". These are the prime business hours when the system must be up and running. These hours are also the time period where the impact is the most if there is a system failure. During that time period, there should not be any changes to the database environment unless they are critical to the application and environment.

As an example, what if your systems and the users are located in different geographical areas? You as a DBA must know the hours of operation of the application, uptime requirements, and criticality of the application so that you can plan for your backups, database maintenance, and the strategy for managing the overall environment without impacting the global users.

WHAT ARE THE SPACE GROWTH PROJECTIONS

One of the challenges facing DBAs is to predict when you will run out of space for the database. There may not be any thing more embarrassing to a DBA than running out of space. You need to monitor the space usage of objects and data files on a regular basis. You should be positioned so that you won't have to explain to the CIO of your organization that the application is failing due to lack of available space.

As a new DBA, you need to be prepared to monitor, estimate, and take corrective action for your database and its space. There are several things that can be done to help you to accomplish this task. Work with the application team. Become familiar with how the application works and how the data flows through your database. Ask the business people within your organization if they anticipate a seasonal growth (for example, the wireless industry can have 40% of their business during the holiday season). Another process to use would be to capture the physical attributes of the data files and objects on a weekly basis and build graphs to reflect the growth over a period of time with the use of the historical data that is captured. You should be able to successfully project some sense of growth model with the database information and the business knowledge that you have obtained.

YOU NEED TO DEVELOP SOME STRATEGIES

Okay, stop and take a breath. Once you have learned the environment, the next thing you need to understand is your backup and recovery strategies. No matter what else happens with your database, if you are able to restore it to a point in time, you will be able to handle anything else.

BACKUP AND RECOVERY

Start by looking into what the users expect. Is this a 24x7 system where the database must be available at all times? Is this a reporting system, with definite times that the database is not in use? The more information you have about the availability of the database, the better able you will be to plan an effective backup. You need to know how frequently you will be able to take a backup, how long the users can function without the database available and how long it will take you to complete a backup.

The second part of the title of this section is called "recovery". It doesn't matter how frequently you take backups of your system, if you cannot recover the database from these backups, they are useless. Test the backups you have taken frequently. Make sure you can read the tapes you have used. It doesn't help to have countless backups, if you have had the tape drive adjusted and can no longer read the tapes. You need to not only develop a backup plan, but also a recovery plan. Document the steps you will need to take to recover the database in the event of a disaster, and TEST those steps. It's much easier to do a recovery that has been rehearsed and had all the steps debugged and timed than it is to first attempt to recover your database under pressure with all your management breathing down your neck.

T O ARCHIVE, OR NOT TO ARCHIVE

One of the decisions you will need to make is whether or not you want to put your database into archivelog mode. If you do not need to be able to recover to as close a point in time as possible to when the database went down, you may not want to put your database into archivelog mode. A decision support system, where the data is loaded once and then not updated until the next data load, is a good candidate for not using archivelog mode. A 24x7 website, where the information in the database is constantly updated, should be in archivelog mode.

WHEN CAN I TAKE THE DATABASE DOWN

Okay, you've planned out your backups, tested them and have a good recovery plan in place. Now, how do you do needed maintenance in your database? You need to work a deal with the users, to give you the time and opportunity to install upgrades of the database software, install changes to the application that they've requested or just reorganize and restructure things in the database for better performance.

Think about ways you can make the database partially available, or create a scaled down version of the database and application that will allow you to move the users to this version while you make changes to the full system. Be prepared to work the off-hours, weekends and the middle of the night, in order to put your changes into place with the least impact on your end-users.

DAILY PROCEDURES

Haven't we given you enough to think about yet? Time for another deep breath and then we'll go on. Now that you have the security of knowing that you can recover from a disaster, it's time to deal with the day-to-day things you should be looking into to ensure that the database performs at a level that is acceptable to your users.

VERIFY ALL INSTANCES ARE UP

Make sure that all database instances are available. Log into each database locally and from a remote node using both a privileged and non-privileged account. These checks should be made on a regular interval and should be automated. A script that runs on a periodic basis that logs into each database, tnsping the instance to check the listener, and ping the host name could notify support of any failures in those steps.

CHECK ALERT LOG

For each managed instance of Oracle, go to the location of the alert log for that instance. This will typically be in $DBA/bdump. You will need to login to each machine with an account that has the rights to view the Oracle Operational directory for that instance. You will need to make sure that your environment is properly defined for those machines that are running multiple instances of Oracle. You may want to automate this so that you mail yourself the overnight alert logs each morning.

DBSNMP PROCESSES

If you are using the OEM and its Intelligent Agents, you will need to log into those machines and check to make sure that the individual agents are up and running. The DBSNMP processes are collectors for the OEM products. You can also write scripts to access the SNMP MIB without OEM. There is an agent on the OEM machine that which will communicate to these SNMP agents/MIBS via the network.

There are different methods to see if the agents are running. One way is to see if the process is actually there:

```
ps -ef|grep dbsnmp
oracle 27841   312  0         Mar 03 ?        0:33 dbsnmp
oracle   312     1  0         Feb 22 ?        6:05 dbsnmp
```

Another way is to check the status using Oracle's listener:

```
LSNRCTL> dbsnmp_status
The db subagent is already running.
LSNRCTL>
```

VERIFY BACKUPS WERE SUCCESSFUL

Every environment will have a different backup strategy. This strategy must be documented and tested frequently to ensure that the backups are readable and you can successfully recover using them.

It is very critical to ensure that the backups have been successful each evening. There are several ways to determine this.

- The backup process should send out an email for notification that the job has completed
- A DBA should log into the database and look at several V$ tables to ensure all data files were backed up
- The DBA should check for entries in the alert log for hot backups
- The backup process itself should create a log file of its processing and this log file should be reviewed
- If the backup process has failed, the DBA has to make sure that there are no tablespaces labeled as "in backup". This will generate more redo activity and create problems at instance restart if there is a crash. From Oracle7.3 and forward, you can use the table V$BACKUP to see if a tablespace is still in backup mode and you *can alter tablespace <tablespace_name> end backup* before opening the database.

In addition to the database files being backed up, the backup procedure must collect all of the archived redo logs. Redo logs should be treated as gold to the environment. You should NEVER have a gap in the redo logs, or you will not be able to perform a complete recovery of your system.

There also should be 2 different types of backups for the control files. One for the actual Oracle control files, and another file which contains the necessary SQL to re-create the control file. Part of the backup strategy could also be individual table exports each evening. If the environment is performing individual table exports each evening, then the export files should be placed on the backup tape as well.

REVIEW AND VALIDATE ALL SCHEDULED JOBS HAVE RUN SUCCESSFULLY

All scheduled jobs via cron, AT, OEM, or other products will need to be reviewed on a daily basis after they are scheduled to run. All scheduled jobs should be configured to send an email about the results to the DBA team. If any of the jobs are not successful, then escalation and corrective actions should be taken for each of the jobs.

THE ORACLE MAIL ACCOUNT

Another daily task should be to archive the oracle mail file. Each day should start with a clean mailbox for ease of tracking jobs,etc.. By archiving the mail file-one can check the output and status of CRON/AT jobs more quickly.

MONITOR ORACLE ARCHIVE REDO LOG FILE SYSTEMS

The file system which holds the archived redo logs should be treated very cautiously. In a typical environment, you need to make sure that you are notified when that file system reaches a certain threshold of use. While the threshold will vary based on how often you create archived redo logs and the size of the logs, you must ensure that the file system never reaches 100% full. If it does, the only symptom is that the database will simply stop the next time Oracle has to archive a redo log and wait for space to be made available. Until you free up that space, no one will be able to do anything that impacts the redo logs. So it is essential that these file systems be monitored and files moved automatically so that this situation does not arise.

Once that threshold is reached, the oldest archived redo logs should be moved to another file system or backed up to tape and deleted to alleviate the space issue. There should be a process that runs multiple times per day via an automated mechanism. Never remove any files from either the archive or backup directory unless you are absolutely sure that the file is on a backup tape. It is typical to see 2 days worth of archived redo logs in both of these directories. This would allow a database recovery to be performed without restoring redo logs from a backup tape.

MONITOR DISK CAPACITY

All file systems related to Oracle should be monitored to determine if any space usage thresholds have been reached. There can be multiple reasons for file systems to reach their capacity. If a file system has reached a threshold then escalation and corrective action needs to be performed. For example, if the $DBA file system is reaching capacity it's possible that several core or trace files have been created. If the $ORACLE_HOME file system is reaching capacity, it's likely that someone is performing a sql*net trace, and that the listener log file is rather large. Each file system has a different cause and effect for reaching its capacity.

REMOVE ANY UNNECESSARY FILES

There are several files and directories that need to be monitored on a daily basis.

- Oracle trace files in either the udump or bdump directories can be removed once you have reviewed them

- Core files/directories should be removed on a daily basis once you have reviewed them
- Listener.log can only be removed by shutting down the listener and either deleting or renaming the log file. It is a good idea to do this on a regular basis so that the file doesn't grow too large.
- Alert_<oracle sid>.log is the log of all actions within the database. This file can be renamed or deleted while the database is up and Oracle will simply create a new one as needed. It is a good idea to rename this file on a daily basis and review it each day.

If Oracle Trace is turned on, the files which are generated can grow to be quite large. These files reside in $ORACLE_HOME/otrace/admin. You will see the process.dat and regid.dat files within that directory and growing if Oracle Trace is turned on. Oracle Trace should be disabled and these files removed.

MONITOR LOCKING FOR EACH INSTANCE

Each instance should be monitored on a daily basis for any user blocking another user. This gives the appearance of a slow system when they are actually waiting behind another user. There are several mechanisms to check locking sessions or to be notified of locking. These should be implemented as desired for the application/environment. There are many reasons why locking would occur, therefore these scenarios need to be documented carefully so the development and DBA teams can resolve the primary issue.

OBJECT MANAGEMENT (EXTENTS)

On a daily basis there are several pieces of information to review about the objects within the database for the applications.

MAX EXTENTS

Each object needs to be checked to see if it is reaching the maximum number of extents allowed for the object. Several levels of warnings need to be generated and sent to the DBA for corrective action.

UNABLE TO CREATE NEXT EXTENT

Each object needs to be checked to see if there is free space available for its next extent. There will be an application outage if any object is unable to allocate a next extent as defined in its storage parameters. This issue can be handled in phases. The object's next extent can be reduced to meet the space requirements initially. Then you can either move objects to another tablespace or add more space for that particular tablespace.

CHECK EXTENTS

By monitoring objects that exceed a threshold, you will be able to extrapolate the possible date where you will exhaust the available space for that environment.

TABLESPACE USAGE

The total space allocated to the database by datafile/tablespace should be monitored on a daily basis. Those tablespaces where the space used is 80% of the total available should be reviewed to see if objects should be moved or if another datafile should be added. This information along with extent/segment growth can be used to determine if an object is growing at an alarming rate.

This information can be retained to assist in the capacity planning of the environment as well. You can review historical data and decide when additional disk should be allocated to the environment or when objects should be moved to a new tablespace.

SESSION AND USER INFORMATION

Users can have an enormous impact on the performance of the database. You therefore have to monitor and maintain control over user access as well as the physical world of the database.

MONITORING TOTAL SESSIONS

Each instance will have an upper limit on the number of processes and sessions allowed into the database. You can review the number of users accessing the database by querying the V$LICENSE table. A predefined threshold should be used to allow you enough time to reset the maximum number and avoid any type of database outage.

REVOKE DBA PRIVILEGES FROM ACCOUNTS

No account other than those controlled by the DBA team should have the DBA role. This role grants total access to the database to the user and should not be given out indiscriminately.

TOTAL ACTIVE SESSIONS

The total number of ACTIVE sessions should be monitored as well. A high number of total active sessions can be indicative of performance, locking, or latch contention issues. This threshold is built over a period of time as you collect data on the characteristics of each instance.

TOTAL SESSIONS BY ACCOUNT

Another item to monitor is the total number of sessions by user. It may be necessary to limit the resources used by an Oracle user account if an environment becomes resource constrained. This can be done via profiles for the Oracle accounts. We have typically monitored the total number of sessions by account to set a threshold that would not create any application issues or place any stress on the environment. With this information, you can adjust the profiles used in the environment.

OPEN CURSORS PER SESSION

Monitoring sessions that reach a threshold against the MAX_OPEN _CURSOR parameter will assist the developers and prevent application errors. It can be used as a tool to see if the applications have any problems where they do not release resources.

STATISTICAL INFORMATION BY SESSION

Monitoring certain statistical information about sessions can lead you to sessions that are consuming more than their fair share of the system resources. For example, you could monitor the disk reads, buffer gets, et al of the open cursors of sessions. This information can direct you to poorly tuned SQL or run-away processes.

STORED OBJECTS

Yet another area to be monitored is the status of the objects within your database.

INVALID STORED OBJECTS

Stored database objects (packages, procedures, functions, views and triggers) may become invalid in your environment due to changes in to the underlying tables or other stored programs in the database. Oracle will normally correct this problem on its own by recompiling the object the next time it is accessed. But there are times when the problem will require manual intervention. You will see stored objects like packages, procedures, and triggers become invalid after an installation of changes for an application. It is always a good habit to recompile invalid objects every time there is an installation or modification to stored objects.

It is also a good habit to run a script that will recompile the invalid objects on a periodic basis. If objects remain in an invalid status, the applications will have errors and the users will be unable to work.

Generated DDL for Stored Objects

There should be several scripts that will generate the necessary SQL to rebuild your stored objects. The generated SQL should rebuild packages, stored procedures, synonyms, views, and triggers. The scripts should be broken down by object type, and run on a periodic basis. The output of the scripts should be placed in a location where they can not be edited. The strategy for this process can vary. The output can be used by development as their source control environment. It is also a good way to document your database.

Daily Exports

Each environment should use the Oracle export mechanism as a function of their backup strategy. Exports will give table level backups in the event a table is dropped.

Exports with data

Exports can be performed on an incremental basis. Once a week, take a full database. Then as the week progresses, only export objects that have changed. This will allow you to spend less time performing exports each evening.

Exports with No Rows

An export file of the system without any rows or data will provide a library of the DDL for the complete database. This file should be used in addition to the scripts described in the Stored Objects section. The file can be used as a backup for re-creating database objects in the event of a failure.

Collection of Database Statistical Information

Each Oracle instance should have a mechanism to collect performance and information statistics. There are 3 methods discussed in this section.

Basic Performance Scripts

There are several basic performance scripts available from the internet or books. The typical information they contain is library cache hit ratios, db_buffer_cache, excessive I/O, etc. The purpose is to take those scripts and gather that information for each instance that is being monitored. This data should be available to use via email, spreadsheet, or inserted into a special database to maintain this information.

Modified Estat/Bstat scripts

The DBAs should run the bstat/estat performance scripts on a periodic basis. These scripts should be run daily or 2 times per day, based on the Oracle instance environment. The scripts should be modified so that the results of the estat script are placed into a database. The data will be available in the typical results.txt file as well as in a database for research and review. By placing the data in a database you can do trending analysis. This data can be very useful to determine if any configuration changes made an impact on the environment.

Analyzing database objects

If a database is using the Oracle Cost Based Optimizer , then the DBAs should analyze the database objects to gather statistics on a regular basis. As the job gathers statistics about the database objects, the information is placed in the DBA_TABLES view. This information should be placed into repository database after the analyze has completed. This repository should have an image of the DBA_TABLES view to hold all of the collected statistical information over time. The collection of this data can be very valuable for estimating capacity planning, growth analysis, application database trends, and any specific growth patterns required.

Periodic Procedures

By now you are sitting there thinking you have more than enough work to occupy your day. Well, there are still a few things left for you to do in your non-existent spare time. Hang in there, we're almost done.

On a regular basis, you should be testing the backups you make. Try restoring them to another machine and opening the database. If you back the files up to tape, verify that you can read the tapes. You do not want to be in a situation where you have to go back several weeks on your backup because the latest tapes are unusable.

Weekly, sit down with the development staff (even if it is only by phone) and review the status of anything that either you need from them or they need from you. Part of your job as a DBA is to suggest ways for the developers to write code that makes better use of the database.

In order to be able to make those suggestions, you are going to have to keep up with the latest updates to the software. Read the documentation, or at least the release notes, to get an overview of the new functionality within the database. Check the Oracle MetaLink website for patch release information and product updates. Alert messages about "undocumented procedures" (otherwise known as bugs) will also be placed on MetaLink.

Finally, spend time once a week reviewing the statistical information you've been collecting. On a daily basis you should be looking for unusual changes (changes of greater than 10% in the size of an object, as an example), but you should still track the changes over a week and month to be able to predict the changes (to both hardware and software) that you will need to implement BEFORE you have problems.

SUMMARY

While stepping in and suddenly becoming the DBA is overwhelming, by taking it one step at a time, reviewing the procedures we have outlined and taking charge of the database and environment, you will be able to handle all problems and situations easily.

ABOUT THE AUTHORS:

Greg Loughmiller has been an Oracle DBA for over 5 years. He has worked in a large scale Oracle shop for the past 5 years managing a centralized DBA team to provide Production Support, Performance tuning, and Architecture standards for an Oracle environment. He is currently working on a large scale architecture and performance tuning initiative. Greg is also OCP certified.

Rachel Carmichael has been an Oracle DBA for over nine years. She chairs the DBA Special Interest Group for the New York Oracle Users Group, is both Chauncey and OCP certified and has presented at various international and national conferences, presenting at the ECO all-day DBA seminar for the past 4 years. She co-authored the Oracle Press book, Oracle SQL and PL/SQL Annotated Archives, 1999 with Kevin Loney, and the Oracle Press book, Oracle DBA 101, 2000 with Marlene Theriault and James Viscusi. She can be reached via email at rachel.carmichael@getmusic.com.