3. Creating a Database and Data Dictionary

Abstract: This practical lesson covers the practical creation of a database and data dictionary in Oracle9i database server. The students also learn to use parameters and set up the correct environment for these activities.

Contents

Objective:

- Create a database
- Prepare the necessary resources for this activity
- Set up the parameters and environment variable
- Correctly identify and use the CREATE DATABASE command
- Create a SPFILE
- Learn about the Data Dictionary
- Create the Dictionary
- Administer Stored Procedures and Packages
- Query the Dictionary

1. CREATING A DATABASE

Creating a database requires planning and preparation. The DB administrator need to prepare the operating system, decide on the configuration parameters, and lay out the physical files of the database for optimum performance. He also needs to create the data dictionary and, for example, the Oracle-supplied PL/SQL packages.

Creating and managing a database requires planning and is done in multiple steps. The database is a collection of physical files that work together with an area of allocated memory and background processes. You create the database only once, but you generally can change the configuration (except the block size) or add more files to the database. Before creating the database, you must have the following:

- Necessary hardware resources such as memory and disk space
- Operating system and SYSDBA privileges
- A plan to lay out the files and their sizes
- A parameter file
- Defined environment variables
- The DBMS software installed
- A backup of existing databases

After you complete these steps, you can create the database using the CREATE DATABASE command. Once the database is created, it is recommended that you create an SPFILE to replace the PFILE parameter file.

2. PREPARING RESOURCES

Preparing the operating system resources is an important step. Depending on the operating system, you may have to adjust certain configuration parameters. For example, on Unix platforms, you must configure the shared memory parameters, because Oracle uses a single shared memory segment for the SGA (System Global Area). Since a major share of Oracle databases are created on Unix platforms, we will discuss certain operating system parameters that must be configured before you can create any Oracle database. The following list itemizes the Unix kernel parameters and describes their purposes. The super-user administers these kernel parameters.

SHMMAX - The maximum size of a shared memory segment

SHMMNI - The maximum number of shared memory identifiers in the system

SHMSEG - The maximum number of shared memory segments to which a user process can attach

SEMMNI - The maximum number of semaphore identifiers in the system

SHMMAX * SHMSEG - The total maximum shared memory that can be allocated

Allocate enough memory for creating the SGA when creating the database and for future database operation. It is better to fit the SGA in real memory, rather than using virtual memory, to avoid paging. Paging will degrade performance.

The Oracle software should be installed on the machine on which you will be creating the database. The user account that installs the software needs certain administrative privileges on Windows NT/2000/XP, but on Unix platforms, the user account that installs the software need not have superuser privileges. The super-user privilege is required only to set up the Oracle

account and to complete certain post-installation tasks such as creating the oratab file. The oratab file lists all the database instance names on that machine, their Oracle home locations, and whether the database should be started automatically at boot time. Certain Oracle scripts and Enterprise Manager discovery services use this file. The oratab file resides in the /etc directory under AIX, HP-UX, and Tru64 or the /var/opt/oracle directory under Solaris and Linux.

The user account that owns the Oracle software should have the necessary privileges to create the data files, redo log files, and control files. The database administrator must make sure that enough free space is available to create these files, and (s)he must follow certain Oracle guidelines about where to create the files.

The parameter file lists the parameters that will be used for creating and configuring the database. The common parameters are discussed in the next section.

Anyone can make mistakes; before performing any major task, ensure that you have methods to fix the mistakes. If you are already running databases on the server where you want to create the new database, make a full backup of all of them. If you overwrite an existing database file when creating the new database, the existing database will become useless.

3. PARAMETERS

Oracle uses the parameter file to start up the instance before creating the database. The database administrator must specify some database configuration values via the parameter file. The purpose and format of the parameter file were discussed in the previous lecture. The following parameters affect database configuration and creation:

CONTROL_FILES Specifies the control file location(s) for the new database with the full pathname. Specify at least two control files on different disks. You can specify a maximum of eight control file names. Oracle creates these control files when the database is created. Be careful when specifying the control file; if you specify the control file name of an existing database, Oracle could overwrite the control file, which will damage the existing database. If you do not use this parameter, Oracle uses the default filename, which is operating system dependent.

DB_BLOCK_SIZE Specifies the database block size as a multiple of the operating system block size — this value cannot be changed after the database is created. The default block size is 4KB on most platforms. Oracle allows block sizes from 2KB to 32KB, depending on the operating system.

DB_NAME Specifies the database name — the name cannot be changed easily after the database is created (you must re-create the control file). The DB_NAME value can be a maximum of eight characters. You can use alphabetic characters, numbers, the underscore (_), the pound symbol (#), and the dollar symbol ($) in the name. No other characters are valid. The first character should be an alphabetic character. Oracle removes double quotation marks before processing the database name. During database creation, the DB_NAME value is recorded in the data files, redo log files, and control file of the database.

Table 1 lists and describes the other parameters that can be included in the parameter file. You must at least define the DB_CACHE_SIZE, LOG_BUFFER and SHARED_POOL_SIZE to calculate the SGA, which must fit into real, not virtual, memory.

Table 1. Initialization parameters.

Parameter Name	Description
OPEN_CURSORS	The maximum number of open cursors a session can have. The default is 50.
MAX_ENABLED_ROLES	The maximum number of database roles that users can enable. The default is 20.
DB_CACHE_SIZE	The size of the default buffer cache, with blocks sized by DB_BLOCK_SIZE. This parameter can be dynamically altered.
SGA_MAX_SIZE	The maximum size allowed for all components of the SGA. Sets an upper limit to prevent dynamically altered sizes of other parameters to push the total SGA size over this limit.
SHARED_POOL_SIZE	Size of the shared pool. Can be specified in bytes or KB or MB. The default value on most platforms is 16MB.
LARGE_POOL_SIZE	The large pool area of the SGA. Default value is 0.
JAVA_POOL_SIZE	Size of the Java pool; the default value is 20,000KB. If you are not using Java, specify the value as 0.
PROCESSES	The maximum number of processes that can connect to the instance. This includes the background processes.

LOG_BUFFER	Size of the redo log buffer in bytes.
BACKGROUND_DUMP_DEST	Location of the background dump directory. The alert log file is written in this directory.
CORE_DUMP_DEST	Location of the core dump directory.
USER_DUMP_DEST	Location of the user dump directory.
REMOTE_LOGIN_PASSWORDFILE	The authentication method. When creating the database, make sure you have either commented out this parameter or set it to NONE. If you create the password file before creating the database, you can specify a different value such as EXCLUSIVE or SHARED.
COMPATIBLE	The release with which the database server must maintain compatibility. You can specify values from 9.0 to the current release number.
SORT_AREA_SIZE	Size of the area allocated for temporary sorts.
LICENSE_MAX_SESSIONS	Maximum number of concurrent user sessions. When this limit is reached, only users with RESTRICTED SESSION privilege are allowed to connect. The default is 0—unlimited.
LICENSE_SESSIONS_WARNING	A warning limit on the number of concurrent user sessions. Messages are written to the alert log when new users connect after this limit is reached. The new user is allowed to connect up to the LICENSE_MAX_SESSIONS value. The default value is 0—unlimited.

LICENSE_MAX_USERS	Maximum number of users that can be created in the database. The default is 0—unlimited.

4. ENVIRONMENT VARIABLES

If you are creating a database on Unix, be sure to set up the appropriate environment variables. The examples in the environment variables below conform with Optimal Flexible Architecture (OFA).

ORACLE_BASE – The directory at the top of the Oracle tree, for example, /u01/apps/oracle. All versions of Oracle installed on this server are stored under this directory.

ORACLE_HOME – The location of the Oracle software, relative to ORACLE_BASE. The OFA-recommended location is $ORACLE_BASE/product/<release>, which in this case would resolve to /u01/apps/oracle/product/901.

ORACLE_SID – The instance name for the database. This name must be unique for all other instances, regardless of version, running on this server.

ORA_NLS33 – The environment variable that you must set if you want to use a character set other than the default.

PATH – The standard Unix pathname that should already exist in the Unix environment; you must add the directory for the Oracle binary executables to this path variable: $ORACLE_HOME/bin.

LD_LIBRARY_PATH – Other program libraries, both Oracle and non-Oracle, that reside in this directory.

5. THE CREATE DATABASE COMMAND

You create the database using the CREATE DATABASE command. You must start up the instance (with STARTUP NOMOUNT PFILE=) before issuing the command.

5.1. Create a database manually

The following is a sample database creation command:

```
CREATE DATABASE "PROD01"
CONTROLFILE REUSE
```

```
LOGFILE GROUP 1
   ('/oradata02/PROD01/redo0101.log',
    '/oradata03/PROD01/redo0102.log) SIZE 5M REUSE,
   GROUP 2
    ('/oradata02/PROD01/redo0201.log',
    '/oradata03/PROD01/redo0202.log) SIZE 5M REUSE
MAXLOGFILES 4
MAXLOGMEMBERS 2
MAXLOGHISTORY 0
MAXDATAFILES 254
MAXINSTANCES 1
NOARCHIVELOG
CHARACTER SET "WE8MSWIN1252"
NATIONAL CHARACTER SET "AL16UTF16"
DATAFILE '/oradata01/PROD01/system01.dbf' SIZE 80M
   AUTOEXTEND ON NEXT 5M MAXSIZE UNLIMITED
UNDO TABLESPACE UNDOTBS
DATAFILE '/oradata04/PROD01/undo01.dbf' SIZE 35M
DEFAULT TEMPORARY TABLESPACE TEMP
TEMPFILE '/oradata05/PROD01/temp01.dbf' SIZE 20M;
```

Let's look at the clauses used in the CREATE DATABASE command. The only mandatory portion in this command is the CREATE DATABASE clause. If you omit the database name, Oracle takes the default value from the parameter DB_NAME defined in the initialization parameter file. The value specified in the parameter file and the database name in this command should be the same.

The CONTROLFILE REUSE clause overwrites an existing control file. Normally you use this clause only when re-creating a database. If you omit this clause, and any of the files specified by the CONTROL_FILES parameter exist, Oracle returns an error.

The LOGFILE clause specifies the location of the online redo log files. If you omit the GROUP clause, Oracle creates the files specified in separate groups with one member in each. A database must have at least two redo groups. In the example, Oracle creates two redo log groups with two members in each. It is recommended that all redo log groups be the same size. The REUSE clause overwrites an existing file, if any, provided the sizes are the same.

The next five clauses specify limits for the database. The control file size depends on these limits, because Oracle pre-allocates space in the control file. MAXLOGFILES specifies the maximum number of redo log groups that can ever be created in the database. MAXLOGMEMBERS specifies the maximum number or redo log members (copies of redo log files) for each redo log group. The MAXLOGHISTORY is used only for the Real Application Cluster configuration. It specifies the maximum number of archived redo log files for automatic media recovery. MAXDATAFILES specifies the maximum number of data

files that can be created in this database. Data files are created when you create a tablespace or add more space to a tablespace by adding a data file. MAXINSTANCES specifies the maximum number of instances that can simultaneously mount and open this database. If you want to change any of these limits after the database is created, you must re-create the control file.

Tip:

The initialization parameter DB_FILES specifies the maximum number of data files accessible to the instance. The MAXDATAFILES clause in the CREATE DATABASE command specifies the maximum number of data files allowed for the database. The DB_FILES parameter cannot specify a value larger than MAXDATAFILES.

You can specify NOARCHIVELOG or ARCHIVELOG to configure the redo log archiving. The default is NOARCHIVELOG; you can change the database to ARCHIVELOG mode by using the ALTER DATABASE command after the database is created.

The CHARACTER SET clause specifies the character set used to store data. The default is WE8MSWIN1252 on Windows platforms. The character set cannot be changed after database creation. The NATIONAL CHARACTER SET clause specifies the national character set used to store data in columns specifically defined as NCHAR, NCLOB, or NVARCHAR2. If not specified, the national character set defaults to the database character set.

The unqualified DATAFILE clause in this example specifies one or more files created for the SYSTEM tablespace. You can optionally specify the AUTOEXTEND clause.

The UNDO TABLESPACE clause specifies an undo tablespace with one or more associated data files. This tablespace contains undo segments when automatic undo management is enabled with the initialization parameter UNDO_MANAGEMENT=AUTO.

The DEFAULT TEMPORARY TABLESPACE clause defines the tablespace location for all temporary segments. If you create a user without specifying a temporary tablespace, this one is used.

Now that you have seen what is involved in creating a database, let's put this all together:

1. Be sure you have enough resources available and the necessary privileges.
2. Decide on a database name, control file locations, and a database block size, and prepare a parameter file including other necessary parameters.
3. Decide on the locations for control files, data files, and redo log files. If at all possible, spread out data files that may compete for the same resources to different physical volumes. For example, updates to a table will generate I/O against both the table and the index. Therefore, placing the indexes in a different tablespace on a different physical volume may improve performance.

4. Decide on the version of the database and the instance name. Set the environment variables ORACLE_HOME with the directory name of the Oracle software installation and ORACLE_SID with the instance name. Normally the instance name and database name are the same. Set up the ORA_NLS33 environment variable if you are using a character set other than the default.
5. Start the instance. Using SQL*Plus, connect using a SYSDBA account and issue STARTUP NOMOUNT.
6. Create the database by using the CREATE DATABASE command.

5.2. Using OMF to Create a Database

In contrast to using the CREATE DATABASE command, using Oracle Managed Files (OMF) can make the process of creating a database much simpler. If the initialization parameters DB_CREATE_FILE_DEST and DB_CREATE_ONLINE_DEST_n are defined with the desired operating system locations for the data files and online redo log files, creating a database can be as simple as the following:

```
CREATE DATABASE DEFAULT TEMPORARY TABLESPACE TMP;
```

6. CREATING AN SPFILE

Using an SPFILE instead of a PFILE for database initialization has many distinct benefits for the DBA, including but not limited to "on-the-fly" modifications to SPFILE contents, with the effect of any parameter change taking place immediately or after the next instance restart. After you configure the init.ora file correctly, create the SPFILE when connected as SYSDBA:

```
SQL> CREATE SPFILE FROM PFILE;
```

By default, both the PFILE and SPFILE reside in the same location. At instance startup, the Oracle Server looks for a file named spfileSID.ora first. If it doesn't exist, the file spfile.ora is used next. If that file does not exist, initSID.ora (a PFILE) is used.

7. THE DATA DICTIONARY

The most important part of the Oracle database is the data dictionary. The data dictionary is a set of tables and views that hold the database's metadata information. You cannot update the dictionary directly; Oracle updates the dictionary when you issue any Data Definition Language (DDL) commands. The dictionary is provided as read-only for users and administrators. The contents of the data dictionary and obtaining information from the dictionary are discussed in the section "Querying the Dictionary."

The data dictionary consists of *base tables* and user-accessible views. The base tables are normalized and contain cryptic, version-specific information. You use the views to query the dictionary and extract meaningful information. To create the views, install the additional Oracle-supplied scripts after the database is created.

The base tables contain information such as the users of the database and their permissions, the amount of the used and unused space for database objects, constraint information, and so on. Users and administrators rarely, if ever, need to access the base tables, with the exception of tables such as AUD$, which contains auditing information for objects in the database.

When the database is created, Oracle creates two users, SYS and SYSTEM. SYS is the owner of the data dictionary, and SYSTEM is a DBA account. The initial password for SYS is CHANGE_ON_INSTALL; the initial password for SYSTEM is MANAGER. Change these passwords once the database is created.

8. CREATING THE DICTIONARY

The Oracle database is functional only when you create the dictionary views and additional tablespaces, rollback segments, users, and so on. Creating the dictionary views is the next step after you create the database by using the CREATE DATABASE command. Running certain Oracle-supplied scripts creates the dictionary views. We'll discuss all these topics in this section as well as give you some basics of how PL/SQL packages are created and maintained in the data dictionary.

8.1. Data Dictionary Scripts

The data dictionary base tables are created under the SYS schema in the SYSTEM tablespace when you issue the CREATE DATABASE command. Oracle automatically creates the tablespace and tables using the sql.bsq script found under the $ORACLE_HOME/rdbms/admin directory. This script creates the following:

- The SYSTEM tablespace by using the data file(s) specified in the CREATE DATABASE command
- A rollback segment named SYSTEM in the SYSTEM tablespace
- The SYS and SYSTEM user accounts
- The dictionary base tables and clusters
- Indexes on dictionary tables and sequences for dictionary use
- The roles PUBLIC, CONNECT, RESOURCE, DBA, DELETE_CATALOG_ROLE, EXECUTE_CATALOG_ROLE, and SELECT_CATALOG_ROLE
- The DUAL table

Don't modify the definitions in the sql.bsq script — for example, by adding columns, removing columns, or changing the data types or width.

You can change these storage parameters: INITIAL, NEXT, MINEXTENTS, MAXEXTENTS, PCTINCREASE, FREELISTS, FREELIST GROUPS, and OPTIMAL.

> Tip:
>
> The DUAL table is a dummy table owned by SYS and accessible to all users of the database. The table has only one column, named DUMMY, and only one row. Do not add more rows to this table.

Running the script catalog.sql creates the data dictionary views. This script creates synonyms on the views to allow users easy access to the views. Before running any data dictionary script, connect to the database as SYS. The dictionary creation scripts are under the $ORACLE_HOME/rdbms/admin directory on most platforms.

The script catproc.sql creates the dictionary items necessary for PL/SQL functionality. The other scripts necessary for creating dictionary objects depend on the operating system and the functionality you want in the database. For example, if you are not using Real Application Clusters (RACs), you need not install any RAC-related dictionary items. At a minimum, run the catalog.sql and catproc.sql scripts after creating the database.

The dictionary creation scripts all begin with cat. Many of the scripts call other scripts. For example, when you execute catalog.sql, it calls the following scripts:

standard.sql Creates a package called STANDARD, which contains the SQL functions to implement basic language features

cataudit.sql Creates data dictionary views to support auditing

catexp.sql Creates data dictionary views to support import/export

catldr.sql Creates data dictionary views to support direct-path load of SQL*Loader

catpart.sql Creates data dictionary views to support partitioning

catadt.sql Creates data dictionary views to support Oracle objects and types

catsum.sql Creates data dictionary views to support Oracle summary management

From the name of a script, you can sometimes identify its purpose. The following list indicates the categories of scripts.

cat*.sql Catalog and data dictionary scripts

dbms*.sql PL/SQL administrative package definitions

prvt*.plb PL/SQL administrative package code, in wrapped (encrypted) form

uNNNNNN.sql Database upgrade/migration scripts

dNNNNNN.sql Database downgrade scripts

utl*.sql Additional tables and views needed for database utilities

9. ADMINISTERING STORED PROCEDURES AND PACKAGES

The PL/SQL stored programs are stored in the data dictionary. They are treated like any other database object. The code used to create the procedure, package, or function is available in the dictionary views DBA_SOURCE, ALL_SOURCE, and USER_SOURCE—except when you create them with the WRAP utility. The WRAP utility generates encrypted code, which only the Oracle server can interpret.

You manage the privileges on these stored programs by using regular GRANT and REVOKE statements. You can GRANT and REVOKE execute privileges on these objects to other users of the database.

The DBA_OBJECTS, ALL_OBJECTS, and USER_OBJECTS views give information about the status of the stored program. If a procedure is invalid, you can recompile it by using the following statement:

```
ALTER PROCEDURE <procedure_name> COMPILE;
```

To recompile a package, compile the package definition and then the package body as in the following statements:

```
ALTER PACKAGE <package_name> COMPILE;

ALTER PACKAGE <package_name> COMPILE BODY;
```

To compile the package, procedure, or function owned by any other schema, you must have the ALTER ANY PROCEDURE privilege.

10. COMPLETING THE DATABASE CREATION

After creating the database and creating the dictionary views, you must create additional tablespaces to complete the database creation process. Oracle recommends creating the following tablespaces if they were not created in the CREATE DATABASE script or with DBCA (Database Configuration Assistant), discussed later. You can create additional tablespaces depending on the requirements of your application.

UNDOTBS Holds the undo segments for automatic undo management. When you create the database, Oracle creates a SYSTEM undo segment in the SYSTEM tablespace. For the database that has multiple tablespaces, you must have at least one

undo segment that is not in the SYSTEM tablespace for manual undo management or one undo tablespace for automatic undo management.

TEMP Holds the temporary segments. Oracle uses temporary segments for sorting and for any intermediate operation. Oracle uses these segments when the information to be sorted will not fit in the SORT_AREA_SIZE parameter specified in the initialization file.

USERS Contains the user tables.

INDX Contains the user indexes.

TOOLS Holds the tables and indexes created by the Oracle administrative tools.

After creating these tablespaces, you must create additional users for the database. As soon as the database is created, back it up, and then immediately change the passwords for SYS and SYSTEM.

11. QUERYING THE DICTIONARY

You can query the data dictionary views and tables in the same way that you query any other table or view. From the prefix of the data dictionary views, you can determine for whom the view is intended. Some views are accessible to all Oracle users; others are intended for database administrators only.

The data dictionary views can be classified into the following categories based on their prefix:

DBA_ These views contain information about all structures in the database - they show what is in all users' schemas. Accessible to the DBA or anyone with the SELECT_CATALOG_ROLE privilege, they provide information on all the objects in the database and have an OWNER column.

ALL_ These views show information about all objects that the user has access to. They are accessible to all users. Each view has an OWNER column, providing information about objects accessible by the user.

USER_ These views show information about the structures owned by the user (in the user's schema). They are accessible to all users and do not have an OWNER column.

V$ These views are known as dynamic performance views, because they are continuously updated while a database is open and in use, and their contents relate primarily to performance. The actual dynamic performance views are identified by the prefix V_$. Public synonyms for these views have the prefix V$.

GV$ For almost all V$ views, Oracle has a corresponding GV$ view. These are the global dynamic performance views and are useful if you are running Oracle Real

Application Clusters. The corresponding GV$ view has an additional column identifying the instance number called INST_ID.

The ALL_ views and USER_ views contain almost identical information except for the OWNER column, but the DBA_ views often contain more information useful for administrators.

You can use the data dictionary information to generate the source code for all the objects created in the database. For example, the information on tables is available in the dictionary views DBA_TABLES, DBA_TAB_COLUMNS, or ALL_TABLES, ALL_TAB_COLUMNS, or USER_TABLES, USER_TAB_COLUMNS.

The dictionary view DICTIONARY contains names and descriptions of all the data dictionary views in the database. DICT is a synonym for DICTIONARY view; the dynamic performance view V$FIXED_TABLE contains similar information to that found in DICTIONARY. The DICT_COLUMNS dictionary view contains the description of all columns in the dictionary views.

If you want to know all the dictionary views that provide information about tables, you can run a query similar to the following.

```
SQL> COL TABLE_NAME FORMAT A25
SQL> COL COMMENTS FORMAT A40
SQL> SELECT * FROM DICT WHERE TABLE_NAME LIKE '%TAB%';
```

The dictionary views ALL_OBJECTS, DBA_OBJECTS, and USER_OBJECTS provide information about the objects in the database. These views contain the timestamp of object creation and the last DDL timestamp. The STATUS column shows whether the object is invalid; this information is especially useful for PL/SQL–stored programs and views.

Tip:

Query the data dictionary view PRODUCT_COMPONENT_VERSION or V$VERSION to see the version of the database and installed components. Oracle product versions have five numbers. For example, in the version number 9.0.1.0.1, 9 is the version, the first zero is the new features' release, the first 1 is the maintenance release, the second zero is the generic patch set number, and the last 1 is the platform-specific patch set number.

Data Dictionary Views vs. Dynamic Performance Tables

The usual distinction between data dictionary views and dynamic performance views is that data dictionary views are relatively static and that dynamic performance tables are primarily related to performance.

In reality, though, there are exceptions to this rule! The dynamic performance view V$VERSION may not change for months, and the data dictionary view DBA_PENDING_TRANSACTIONS may change constantly in a distributed environment.

The experienced DBA, therefore, cannot always rely on view prefixes and sometimes just has to know where to look for information about the running database.

12. THE DATABASE CONFIGURATION ASSISTANT

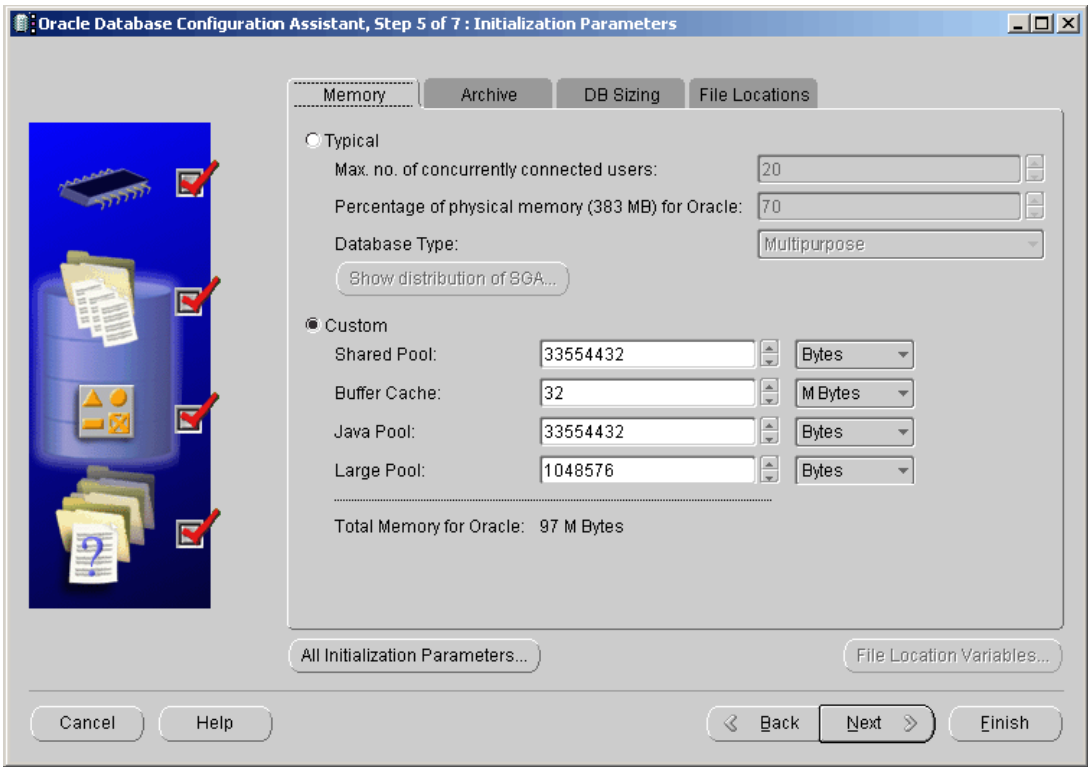
The Database Configuration Assistant (DBCA) is Oracle's GUI DBA tool for creating, modifying, or deleting a database. After you answer a few questions, this tool can create a database, save a template for future use, or give you the scripts to create the database. It is a good idea to generate the scripts by using this tool and then customize the script files, if needed, and create the database. You can create the database with a Shared Server configuration (MTS) or a dedicated server configuration. You can also choose the additional options you want to install in the database, such as Oracle InterMedia and Oracle JVM.

You can run the DBCA as part of the Oracle Universal Installer (OUI) or as a stand-alone application. As with the manual creation of the database, be sure to define the environment variables before creating the database.

If you choose the typical installation option, you have only a few questions to answer. You also have the option of copying a preconfigured database (template) or creating a new database. The tool generates the initialization parameters based on the type of database you create; the options are Online Transaction Processing (OLTP), Data Warehousing, or Multipurpose.

If you choose the custom installation option, you have full control of the SGA sizing. Figure 1 shows the SGA memory configuration screen of the DBCA.

Figure 1. DBCA SGA Sizing



To run the DBCA, use the dbca command under Unix, or find DBCA in the Windows Start menu under Oracle/Configuration and Migration Tools/Database Configuration Assistant.

You can easily configure many, if not all, of the other initialization parameters from within the DBCA. Following are some typical parameters:

- File locations: pathname for parameter and trace files
- Character set values: NLS-related
- Archiving: location and format of the archived log files
- Trace files: location of the user and system trace files
- Storage: block size, sizing control files, datafiles and tablespaces, redo log groups
- Sorting: SORT_AREA_SIZE

Figure 2 shows a sample DBCA screen in which the character set options and sort area size can be adjusted. If you are an advanced DBA, you can modify virtually every initialization parameter, as shown in Figure 3.

Figure 2. DBCA Sort Area Sizing

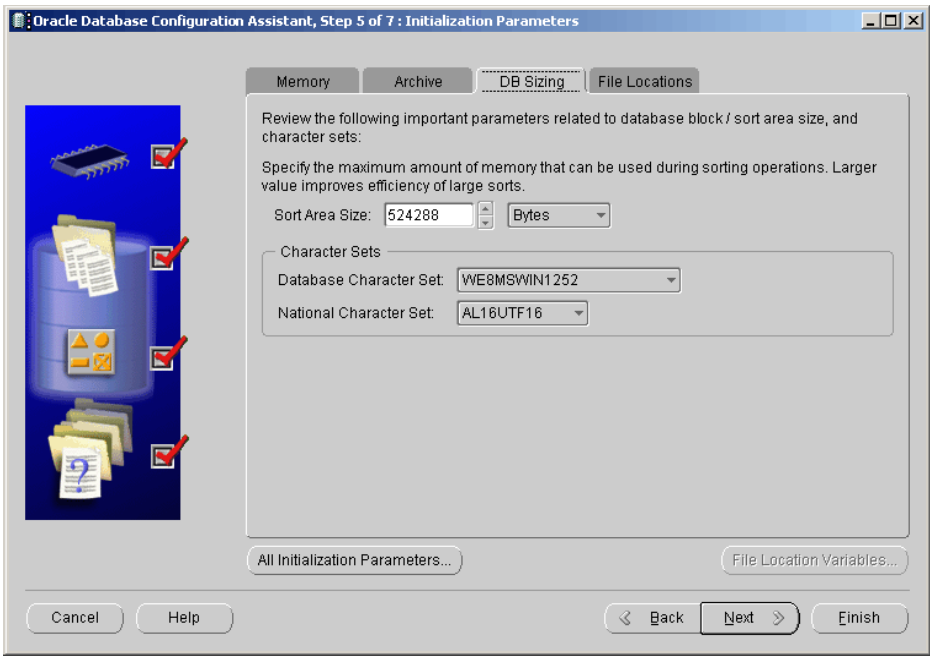
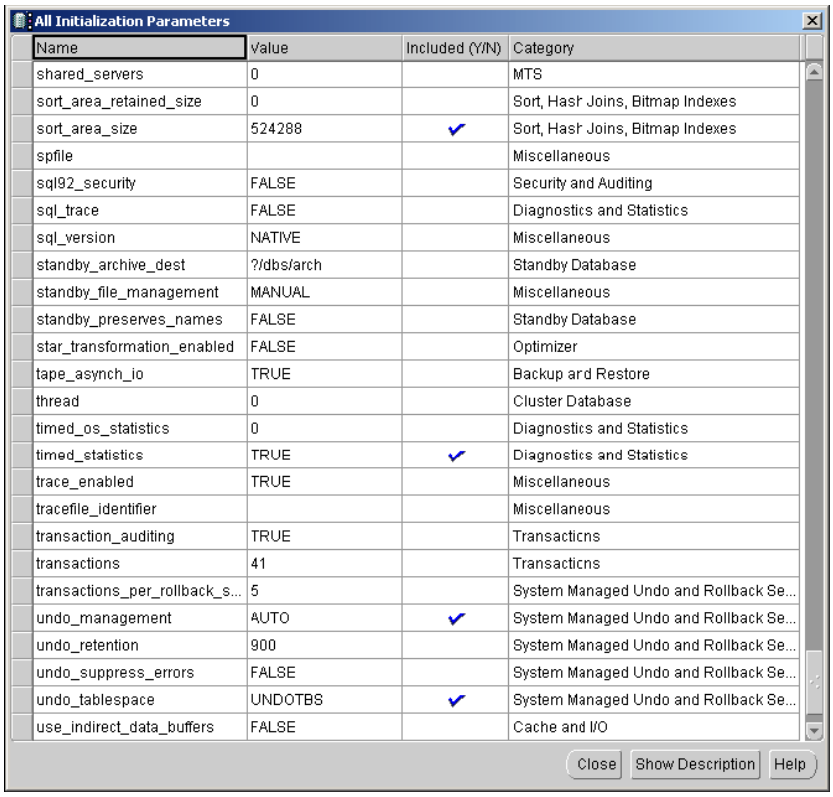


Figure 3. DBCA Initialization Parameter Editing



The script generated by the DBCA does the following (this is a good template for you to use when creating new databases):

1. Creates a parameter file, starts up the database in NOMOUNT mode, and creates the database by using the CREATE DATABASE command.

2. Runs catalog.sql.

3. Creates tablespaces for tools (TOOLS), undo (UNDOTBS), temporary (TEMP), user (USERS), and index (INDX).

4. Runs the following scripts:

   a. catproc.sql—sets up PL/SQL.

   b. caths.sql—installs the heterogeneous services (HS) data dictionary, providing the ability to access non-Oracle databases from the Oracle database. HS is an integrated component in the 9i database.

   c. otrcsvr.sql—sets up Oracle Trace server stored procedures.

   d. utlsampl.sql—sets up sample user SCOTT and creates demo tables.

   e. pupbld.sql—creates product and user profile tables. This script is run as SYSTEM.

5. Runs the scripts necessary to install the other options chosen.

You can also use the DBCA to manage templates, which makes it easier to create a similar database on the same or a different server. You can create the template from scratch, or generate it from an existing database. You can create these derived templates with or without the data from the original database. In essence, this template management feature let you easily clone a database.

When you create a template that contains both the structure and the data from an existing database, any database you create with this template must contain all datafiles, tablespaces and undo segments in the template. You cannot add or remove any datafiles, tablespaces or undo segments before the database is created, nor can you change any initialization parameters. You can change control files, log groups and data file destinations.

SUMMARY

In this lesson, you learned how to create a database. The Oracle database is created by using the command CREATE DATABASE. This command runs the sql.bsq script, which in turn creates the data dictionary tables. The three parameters that you should pay particular attention to before creating a database are CONTROL_FILES, DB_NAME, and DB_BLOCK_SIZE. You cannot change the block size of the database once it is created.

Running the script catalog.sql creates the data dictionary views. DBAs and users use these views to query the information from the data dictionary. The data dictionary is a set of tables and views that hold the metadata. The views prefixed with DBA_ are accessible only to the

DBA or user with the SELECT_CATALOG_ROLE privilege. The views prefixed with ALL_ have information about all objects in the database that the user has any privilege on. The USER_ views show information about the objects that the user owns.

Before creating the database, be sure that you have enough resources, such as disk space and memory. You also need to prepare the operating system by setting the resource parameters, if any, and making sure that you have enough privileges. Oracle Managed Files (OMF) aids in the database creation by centralizing default operating system file locations.

PL/SQL has several administrative packages that are useful to the DBA as well as developers. To install these packages you run the script catproc.sql. Most of the administrative scripts are located under the directory $ORACLE_HOME/rdbms/admin.

Oracle has a graphical database creation tool, DBCA, designed to ease the administrative burden of creating scripts manually. DBCA also allows the DBA to create database templates, with and without data files.

References

[1] Oracle9i DBA Fundamentals I.