

CONGRATULATIONS, YOU'RE THE NEW ORACLE DBA!

Elizabeth Boss, Boss Consulting Services, Inc.

INTRODUCTION

As a new database administrator, the wonderful world of Oracle may seem exciting and overwhelming at the same time. The purpose of this paper is to provide a new database administrator with a skill set that will enable effective and efficient operation the database. One of Oracle's most distinguishing features is the control that the DBA has over the manner in which the RDBMS runs. While this is an advantage, it also adds to the complexity of managing the database. As new entrants into the Oracle arena it is important to realize the opportunity for exploration and the tremendous opportunity for gaining a wealth of knowledge pertaining to the Oracle RDBMS.

USER MANAGEMENT

As a new DBA, probably one of the first tasks that you will perform will be user management. Under the genera of user management you will be required to perform such rudimentary tasks as creating new users and assigning them appropriate access privileges to database objects. The Oracle Enterprise Manager Security Manager tool provides an easy-to-use interface for creating new users. The CREATE USER command can also be issued. It is important to assign the new user a default and temporary tablespace so that they do not default to the SYSTEM tablespace.

```
CREATE USER jjohnson
IDENTIFIED by jjohnson
DEFAULT TABLESPACE user_data
TEMPORARY TABLESPACE temporary_data;
```

An advantage of using he Create User window in the Security Manager tool is that a list of available tablespaces is provided allowing a imple selection for the default and temporary tablespaces. The purpose of the temporary tablespace is to provide a location for tables temporary tables required during the processing of SQL statements. Statements requiring temporary space include ORDER BY, GROUP BY, and table joins performed through a SORT-MERGE. Temporary space is allocated when an SQL statement is initiated and de-allocated upon completion of the statement. The default tablespace for temporary segments is the SYSTEM tablespace. Since the segments are continuously allocated and de-allocated, this causes unnecessary fragmentation of the SYSTEM tablespace. To overcome this problem, a temporary tablespace should be created and users assigned to this tablespace.

The default tablespace is used when a user creates an object (including tables, indexes, and snapshots) and does not explicitly specify the storage location for the object. If a user is not assigned a default tablespace, the SYSTEM tablespace is used. The primary problem with this is that the SYSTEM tablespace contains the data dictionary information, which is crucial to the functioning of the database. Mixing data dictionary tables and user defined tables is not a good practice for several reasons, including fragmentation, physical file contention, backup, and recovery.

The user's password may be changed by the database administrator by issuing either a CREATE USER or ALER USER command. A user may change their own password by issuing an ALTER USER command.

```
CREATE USER msmith identified by msmith;

ALTER USER jjohnson identified by change_me;
```

ROLE MANAGEMENT

The task of assigning appropriate privileges on database objects to users can be simplified if *Roles* are used to group together sets of privileges. A *Role* consists of a set of privileges that have been granted to a named object (the Role). To create a role issue a CREATE ROLE command.

```
CREATE ROLE end_user;
```

The Enterprise Manager Security Manager Create Role window can also be used to create a new role. The role now exists as a database object, but does not contain any privileges. Two types of privileges can be granted to a role: system privileges and object privileges. System privileges provide the right to issue a system command like CREATE, ALTER, and DROP. Object privileges allow access to specifically named objects like tables, indexes, snapshots, and views. The GRANT command is used to provide users with both system and object privileges.

```
GRANT SELECT, INSERT, UPDATE ON dept TO end_user;
GRANT CREATE SYNONYM to end_user;
```

An existing role can be assigned to users using another form of the grant command:

```
GRANT end_user TO msmith;
```

The following query can be used to determine what roles have been created in the database:

```
COLUMN "Password?" format a10
SELECT role "Role Name",
       password_required "Password?"
FROM dba_roles
ORDER BY role;
```

SAMPLE OUTPUT:

Role Name	Password?
AQ_ADMINISTRATOR_ROLE	NO
AQ_USER_ROLE	NO
CONNECT	NO
CTXADMIN	NO
CTXAPP	NO
CTXUSER	NO
DBA	NO
DELETE_CATALOG_ROLE	NO
END_USER	NO
EXECUTE_CATALOG_ROLE	NO
EXP_FULL_DATABASE	NO

The role name is the user-defined name of the role. The password column indicates whether a password is required to enable the role. Roles that do not require a password can be enabled automatically by setting the user's default role to ALL. The following script displays all roles and the privileges that have been assigned to the roles:

```
BREAK ON "Role Name" SKIP 2
COLUMN "Admin?" format a6
SELECT role "Role Name",
       privilege "Privilege",
       admin_option "Admin?"
FROM dba_roles, dba_sys_privs
WHERE role = grantee
ORDER BY role;
```

SAMPLE OUTPUT:

Role Name	Privilege	Admin?
CONNECT	ALTER SESSION	NO
	CREATE CLUSTER	NO
	CREATE DATABASE LINK	NO
	CREATE SEQUENCE	NO
	CREATE SESSION	NO
	CREATE SYNONYM	NO
	CREATE TABLE	NO
	CREATE VIEW	NO
	SELECT ANY TABLE	NO
	CTXADMIN	ALTER ANY CLUSTER
DBA	ALTER ANY INDEX	

To obtain a list of what roles have been granted to which users the following query can be used:

```

BREAK ON "User Name" SKIP 2
COLUMN "Admin?" format a6
COLUMN "Default?" format a8

SELECT grantee "User Name",
       granted_role "Role Name",
       admin_option "Admin?",
       default_role "Default?"
FROM dba_role_privs
ORDER BY grantee;
    
```

SAMPLE OUTPUT:

User Name	Role Name	Admin?	Default?
ADMIN	CONNECT	NO	YES
	DBA	NO	YES
	RESOURCE	NO	YES
CTXADMIN	CTXAPP	NO	YES
CTXAPP	CTXUSER	NO	YES
CTXSYS	CONNECT	NO	YES
	DBA	NO	YES

The following script shows all the roles and privileges granted to those roles as well as roles granted to other roles.

```

COLUMN "Privilege" format a24 word_wrapped
COLUMN "Granted Role" format a20 word_wrapped
COLUMN "Role" format a20
BREAK ON "Role" skip 2
SELECT grantee "Role",
       privilege "Privilege",
       ' ' "Granted Role"
FROM dba_sys_privs,dba_roles
WHERE role = grantee
UNION
SELECT grantee "Role",
       ' ' "Privilege",
       granted_role
FROM dba_role_privs, dba_roles
WHERE grantee = role
ORDER BY 1,2,3;
    
```

SAMPLE OUTPUT:

Role	Privilege	Granted Role
CONNECT	ALTER SESSION	
	CREATE CLUSTER	
	CREATE DATABASE LINK	
	CREATE SEQUENCE	
	CREATE SESSION	
	CREATE SYNONYM	

```
CREATE TABLE
CREATE VIEW
```

```
CTXADMIN                                CTXAPP
SELECT ANY TABLE
```

```
CTXAPP                                CTXUSER
```

MONITORING USER OBJECTS

As data is loaded into tables and extents are filled, additional extents are dynamically allocated by Oracle. The maximum number of extents that can be allocated for a given object is determined by the value for MAXEXTENTS in either the CREATE TABLE or CREATE TABLESPACE command. To determine the limit on extents and number of extents currently allocated for a table, query the data dictionary table dba_segments.

```
COLUMN "Table Name" format a20
COLUMN "Owner" format a14
SELECT segment_name "Table Name",
       owner "Owner",
       extents "Num. Extents",
       max_extents "Max. Extents"
FROM dba_segments
WHERE segment_type = 'TABLE'
and OWNER <> 'SYS'
ORDER BY owner, segment_name;
```

SAMPLE OUTPUT:

Table Name	Owner	Num. Extents	Max. Extents
CLASS	ADMIN	1	121
COURSE	ADMIN	121	121
DEMO	ADMIN	15	121
DEPT	ADMIN	1	121
GRADE	ADMIN	1	121
GRADE_HISTORY	ADMIN	121	121
STUDENT	ADMIN	1	121

If a user attempts to enter additional data and an extent cannot be allocated, the following error message is returned, and a rollback is performed:

```
ORA-01631: max # extents (121) reached in table ADMIN.GRADE_HISTORY
```

To resolve the problem, increase the value for MAXEXTENTS to a higher number or set it to UNLIMITED by issuing the following command:

```
ALTER TABLE grade_history STORAGE(maxextents 200);
```

The disadvantage of unlimited extents is that the database administrator relinquishes control over the growth of tables. With regular monitoring, problems due to table growth and extent allocations can be avoided.

MONITORING DATABASE OBJECTS

The database administrator is responsible for the creation and maintenance of storage containers for database objects called tablespaces. Tablespaces allow objects of the same type to be grouped together and stored in the same logical container. In order to maintain tablespaces the DBA must be aware of what tablespaces have been created and what objects are assigned to each tablespace. The following script creates a list of tablespaces:

```
SELECT tablespace_name,
       status
FROM dba_tablespaces
ORDER BY tablespace_name;
```

SAMPLE OUTPUT:

TABLESPACE_NAME	STATUS
-----------------	--------

```

-----
ENROLL                                ONLINE
ROLLBACK_DATA                        ONLINE
SYSTEM                                ONLINE
TEMPORARY_DATA                       ONLINE
USER_DATA                             ONLINE

```

The status column indicates whether a tablespace is currently online or offline. Tablespaces typically remain online, but can be taken offline manually if desired, to perform a backup or when tablespace recovery is required. When a tablespace is created, at least one physical file is assigned to the tablespace. The size of the file is based upon the size specification in the CREATE TABLESPACE command. As objects are assigned to the tablespace, space is allocated in the data file assigned to the tablespace. Space usage in a tablespace can be monitored through the data dictionary table `dba_free_space`.

```

SELECT f.tablespace_name,
       round(sum(f.bytes)/d.bytes,3)*100 "% Free"
FROM   dba_free_space f,
       dba_data_files d
WHERE  f.file_id = d.file_id
GROUP BY f.tablespace_name, d.bytes;

```

SAMPLE OUTPUT:

```

TABLESPACE_NAME          % Free
-----
ENROLL                   98
ROLLBACK_DATA            17.9
SYSTEM                   9.9
TEMPORARY_DATA           99.9
USER_DATA                24.8

```

When the underlying data file for a tablespace is full, no additional data can be added to the objects assigned to that tablespace. The user attempting to add data to a file that does not have space will receive the following error message: `ORA-01653: unable to extend table ADMIN.GRADE_HISTORY by 1 in tablespace ENROLL`

Upon receiving this error message, the database administrator should take the time necessary to determine the amount of free space available in the tablespace. It is possible that the free space is available in the tablespace, but simply not coalesced. If a query against `dba_free_space` shows enough free space to perform the intended operation, the following command can be issued to manually coalesce the free space in the tablespace:

```
ALTER TABLESPACE enroll COALESCE;
```

If the query against `dba_free_space` indicates that there is truly not enough space in the tablespace, the problem can be resolved by adding a data file to the tablespace affected.

```
ALTER TABLESPACE enroll ADD DATAFILE 'enroll2.dbf' size 10M;
```

After the problem has been resolved (either by addition a new data file or by coalescing the free space), the transaction that caused the error can be re-executed, and should succeed.

Another critical type of database objects that the DBA is responsible for are rollback segments. Rollback segments contain the data necessary to “undo” or rollback a transaction and return the record to the state prior to changes. The data contained in a rollback segment includes the before image of an update or delete transaction and block information including block ID and file number. Rollback segments are used for each INSERT, UPDATE, and DELETE statement issued. Rollback segments are also accessed to provide a user with a read-consistent version of data due to a very long running query. To determine what rollback segments have been created query `dba_rollback_segs`.

```

SELECT segment_name "Rollback Segment",
       tablespace_name "Tablespace",
       status
FROM   dba_rollback_segs
ORDER BY segment_name;

```

SAMPLE OUTPUT:

Rollback Segment	Tablespace	STATUS
RB1	ROLLBACK_DATA	ONLINE
RB2	ROLLBACK_DATA	ONLINE
RB3	ROLLBACK_DATA	ONLINE
RB4	ROLLBACK_DATA	ONLINE
RB5	ROLLBACK_DATA	ONLINE
RB6	ROLLBACK_DATA	ONLINE
RB7	ROLLBACK_DATA	ONLINE
RB_TEMP	SYSTEM	OFFLINE
SYSTEM	SYSTEM	ONLINE

All rollback segments other than the one or two SYSTEM rollback segments should be assigned to a tablespace specifically dedicated to rollback segments. This ensures continuous availability of the rollback segments and reduces backup/recovery requirements for the tablespace. The STATUS column indicates whether a rollback segment is online or offline. An offline rollback segment has been created, but is not currently available for use. The status of a rollback segment is changed with the ALTER ROLLBACK SEGMENT command. Just like any other database object, when rollback segments are created the initial and maximum number of extents are specified. To determine what values were specified for these two parameters, the data dictionary tables dba_rollback_segs and v\$rollstat can be used.

```
SELECT segment_name "RBS",
       min_extents "Min. Extents",
       max_extents "Max. Extents",
       optsize "Optimal"
FROM dba_rollback_segs,
     v$rollstat
WHERE usn = segment_id;
```

RBS	Min. Extents	Max. Extents	Optimal
SYSTEM	2	121	
RB1	2	121	20480
RB2	2	121	20480
RB3	2	121	
RB4	2	121	
RB5	2	121	
RB6	2	121	
RB7	2	121	

Rollback segments have the ability to extend until MAXEXTENTS has been reached. At that point in time, the “guilty” transaction (i.e., the one that caused the allocation which exceeded maxextents) would receive an error message, and a rollback would be performed.

```
ORA-01562: failed to extend rollback segment number 18
ORA-01628: max # extents (4) reached for rollback segment TEST_RBS
```

The error message indicates the rollback segment id and segment name. At this point, the DBA has several options for problem resolution. If there are active transactions, when the transactions are committed the rollback space will be available. The first action taken by the DBA could be to determine which rollback segments are being used by which users. That information is stored in x\$ tables.

```
select b.ksuudnam "User Name",
       c.name "RBS Name"
FROM x$ktcxb a,
     x$ksuse b,
     undo$ c
WHERE a.ksspaown = b.addr
AND    c.us# = a.kxidusn
ORDER BY c.name;
```

SAMPLE OUTPUT:

User Name	RBS Name
ADMIN	RB3
MSMITH	RB5

After identification of users accessing the rollback segment, the users could be contacted and asked to perform commits on active transactions. If enough space is de-allocated, no further work is required, the failed transaction can be re-executed. If this is a non-typical transaction, a large rollback segment could be created, and the transaction set to use the large rollback segment.

```
SET TRANSACTION USE ROLLBACK SEGMENT large_rbs;
```

Upon completion of the large transaction, Oracle's default distribution of transactions takes place once again. At that point in time, the large rollback segment could be taken offline until needed again. If frequent errors are encountered when trying to allocate space within rollback segments, either add more rollback segments or increase the size of the existing rollback segments. If the transaction is a typical transaction, consider increasing the size of all rollback segments to accommodate larger transactions. The size that a rollback segment can grow to is modified by increasing the parameter MAXEXTENTS.

PRIVILEGE ERRORS

Another frequently encountered error is ORA-00942: table or view does not exist. The first action the administrator should take is to check object level privileges against the table or view. The following script provides a list of object-level privileges:

```
COLUMN "Object" format a24
COLUMN "Privilege" format a12
COLUMN grantee format a12
SELECT owner || '.' || table_name "Object",
       ' ' "Column",
       privilege "Privilege",
       grantee "Grantee"
FROM dba_tab_privs
UNION
SELECT owner || '.' || table_name,
       column_name,
       privilege,
       grantee
FROM dba_col_privs
```

SAMPLE OUTPUT:

Object	Column	Privilege	Grantee
ADMIN.CLASS		SELECT	MSMITH
ADMIN.COURSE		SELECT	MSMITH
ADMIN.DEPT		INSERT	MSMITH
ADMIN.DEPT		SELECT	MSMITH
ADMIN.DEPT		UPDATE	MSMITH
ADMIN.GRADE		SELECT	MSMITH
ADMIN.GRADE	GRADE	UPDATE	MSMITH
ADMIN.STUDENT		SELECT	MSMITH

The grantee and table name can be specified in the WHERE clause to limit the rows returned. If appropriate privileges have been granted to the user, check to see that either a synonym for the table has been created, or that the user is specifying owner.tablename in the SQL statement. To verify a synonym, query dba_synonyms or user_synonyms depending on whether the synonym is public or private.

```

SELECT synonym_name "Synonym Name",
       table_owner "Table Owner",
       table_name "Table Name"
FROM user_synonyms
ORDER BY synonym_name;

```

SAMPLE OUTPUT:

Synonym Name	Table Owner	Table Name
-----	-----	-----
CLASS	ADMIN	CLASS
COURSE	ADMIN	COURSE
DEPT	ADMIN	DEPT
GRADE	ADMIN	GRADE
STUDENT	ADMIN	STUDENT

CONCLUSION

The Oracle RDBMS is a comprehensive and powerful database that can seem quite overwhelming to a new database administrator. By understanding some of the most common errors encountered, and being prepared to resolve common problems, the administrator will be able to quickly handle the majority of problems that arise. Because Oracle provides so much control to the administrator, it requires a more in-depth understanding of how Oracle functions than other databases may require. One of the best qualities that a new DBA can have is the desire to explore. New knowledge can be gained continually by investigating the data dictionary views provided by Oracle, including the dynamic performance tables and the fixed tables (x\$ tables). Since the Oracle RDBMS changes frequently, so must the DBA's skills and knowledge level.