2 Managing Security for Oracle Database Users

This chapter contains:

About User Security

Each Oracle database has a list of valid database users. To access a database, a user must run a database application, and connect to the database instance using a valid user name defined in the database. Oracle Database enables you to set up security for your users in a variety of ways. When you create user accounts, you can specify limits to the user account. You can also set limits on the amount of various system resources available to each user as part of the security domain of that user. Oracle Database provides a set of database views that you can query to find information such as resource and session information. This chapter also describes profiles. A profile is collection of attributes that apply to a user. It enables a single point of reference for any of multiple users that share those exact attributes.

Another way to manage user security is to assign users privileges and roles. Chapter 4, "Configuring Privilege and Role Authorization," provides detailed information.

Creating User Accounts

This section contains:

For guidelines about creating and managing user accounts and passwords, see the following sections:

- "Guidelines for Securing User Accounts and Privileges"

- "Guidelines for Securing Passwords"

Creating a New User Account

You create a database user with the `CREATE USER` statement. To create a user, you must have the `CREATE USER` system privilege. Because it is a powerful privilege, a database administrator or security administrator is usually the only user who has the `CREATE USER` system privilege.

Example 2-1 creates a user and specifies the user password, default tablespace, temporary tablespace where temporary segments are created, tablespace quotas, and profile. It also grants the user the minimum privilege, `CREATE SESSION`, to log in to the database session.

Example 2-1 Creating a User Account with the CREATE SESSION Privilege

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk;
GRANT CREATE SESSION TO jward;
```

Replace *password* with a password that is secure. See "Minimum Requirements for Passwords" for more information.

A newly created user cannot connect to the database until you grant the user the `CREATE SESSION` system privileges. So, immediately after you create the user account, use the `GRANT` SQL statement to grant the user these privileges. If the user must access Oracle Enterprise Manager, you should also grant the user the `SELECT ANY DICTIONARY` privilege.

Note:
As a security administrator, you should create your own roles and assign only those privileges that are needed. For example, many users formerly granted the `CONNECT` privilege did not need the additional privileges `CONNECT` used to provide. Instead, only `CREATE SESSION` was actually needed, and in fact, that is the only privilege `CONNECT` presently retains.

Creating organization-specific roles gives an organization detailed control of the privileges it assigns, and protects it in case Oracle Database changes the roles that it defines in future releases. For example, both `CONNECT` and `RESOURCE` roles will be deprecated in future Oracle Database releases. Chapter 4, "Configuring Privilege and Role Authorization," discusses how to create and manage roles.

Specifying a User Name

Within each database, a user name must be unique with respect to other user names and roles. A user and role cannot have the same name. Furthermore, each user has an associated schema. Within a schema, each schema object must have a unique name. In the following, the text in **bold** shows how to create the user name.

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk;
```

Assigning the User a Password

In Example 2-1, the new user is to be authenticated using the database. In this case, the connecting user must supply the correct password to the database to connect successfully. To specify a password for the user, use the `IDENTIFIED BY` clause in the `CREATE USER` statement.

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk;
```

See Also:

- "Minimum Requirements for Passwords" for the minimum requirements for creating passwords
- "Guidelines for Securing Passwords" for additional ways to secure passwords
- Chapter 3, "Configuring Authentication," for information about authentication methods that are available for Oracle Database users

---

Assigning a Default Tablespace for the User

Each user should have a default tablespace. When a schema object is created in the user's schema and the DDL statement does not specify a tablespace to contain the object, Oracle Database stores the object in the default user's tablespace.

The default setting for the default tablespaces of all users is the `SYSTEM` tablespace. If a user does not create objects, and has no privileges to do so, then this default setting is fine. However, if a user is likely to create any type of object, then you should specifically assign the user a default tablespace, such as the `USERS` tablespace. Using a tablespace other than `SYSTEM` reduces contention between data dictionary objects and user objects for the same data files. In general, do not store user data in the `SYSTEM` tablespace.

You can use the `CREATE TABLESPACE` SQL statement to create a permanent default tablespace other than `SYSTEM` at the time of database creation, to be used as the database default for permanent objects. By separating the user data from the system data, you reduce the likelihood of problems with the `SYSTEM` tablespace, which can in some circumstances cause the entire database to become nonfunctional. This default permanent tablespace is not used by system users, that is, `SYS`, `SYSTEM`, and `OUTLN`, whose default permanent tablespace is `SYSTEM`. A tablespace designated as the default permanent tablespace cannot be dropped. To accomplish this goal, you must first designate another tablespace as the default permanent tablespace. You can use the `ALTER TABLESPACE` SQL statement to alter the default permanent tablespace to another tablespace. Be aware that this will affect all users or objects created after the `ALTER` DDL statement commits.

You can also set a user default tablespace during user creation, and change it later with the `ALTER USER` statement. Changing the user default tablespace affects only objects created after the setting is changed.

When you specify the default tablespace for a user, also specify a quota on that tablespace.

In the following `CREATE USER` statement, the default tablespace for user `jward` is `data_ts`, and his quota on that tablespace is `500K`:

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk;
```

Assigning a Tablespace Quota for the User

You can assign each user a tablespace quota for any tablespace (except a temporary tablespace). Assigning a quota accomplishes the following:

- Users with privileges to create certain types of objects can create those objects in the specified tablespace.
- Oracle Database limits the amount of space that can be allocated for storage of a user's objects within the specified tablespace to the amount of the quota.

By default, a user has no quota on any tablespace in the database. If the user has the privilege to create a schema object, then you must assign a quota to allow the user to create objects. At a minimum, assign users a quota for the default tablespace, and additional quotas for other tablespaces in which they can create objects.

The following `CREATE USER` statement assigns the following quotas for the `test_ts` and `data_ts` tablespaces:

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk;
```

You can assign a user either individual quotas for a specific amount of disk space in each tablespace or an unlimited amount of disk space in all tablespaces. Specific quotas prevent a user's objects from using too much space in the database.

You can assign quotas to a user tablespace when you create the user, or add or change quotas later. (You can find existing user quotas by querying the `USER_TS_QUOTAS` view.) If a new quota is less than the old one, then the following conditions remain true:

- If a user has already exceeded a new tablespace quota, then the objects of a user in the tablespace cannot be allocated more space until the combined space of these objects is less than the new quota.
- If a user has not exceeded a new tablespace quota, or if the space used by the objects of the user in the tablespace falls under a new tablespace quota, then the user's objects can be allocated space up to the new quota.

Restricting the Quota Limits for User Objects in a Tablespace

You can restrict the quota limits for user objects in a tablespace by using the `ALTER USER` SQL statement to change the current quota of the user to zero. After a quota of zero is assigned, the objects of the user in the tablespace remain, and the user can still create new objects, but the existing objects will not be allocated any new space. For example, you could not insert data into one of this user's exiting tables. The operation will fail with an `ORA-1536 space quota exceeded for tables` error.

Granting Users the UNLIMITED TABLESPACE System Privilege

To permit a user to use an unlimited amount of any tablespace in the database, grant the user the `UNLIMITED TABLESPACE` system privilege. This overrides all explicit tablespace quotas for the user. If you later revoke the privilege, then you must explicitly

grant quotas to individual tablespaces. You can grant this privilege only to users, not to roles.

Before granting the UNLIMITED TABLESPACE system privilege, you must consider the consequences of doing so.

Advantage:

You can grant a user unlimited access to all tablespaces of a database with one statement.

Disadvantages:

- The privilege overrides all explicit tablespace quotas for the user.
- You cannot selectively revoke tablespace access from a user with the UNLIMITED TABLESPACE privilege. You can grant selective or restricted access only after revoking the privilege.

Assigning a Temporary Tablespace for the User

You should assign each user a temporary tablespace. When a user executes a SQL statement that requires a temporary segment, Oracle Database stores the segment in the temporary tablespace of the user. These temporary segments are created by the system when performing sort or join operations. Temporary segments are owned by SYS, which has resource privileges in all tablespaces.

In the following, the temporary tablespace of jward is temp_ts, a tablespace created explicitly to contain only temporary segments.

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk;
```

To create a temporary tablespace, use the CREATE TEMPORARY TABLESPACE SQL statement.

If you do not explicitly assign the user a temporary tablespace, then Oracle Database assigns the user the default temporary tablespace that was specified at database creation, or by an ALTER DATABASE statement at a later time. If there is no default temporary tablespace explicitly assigned, then the default is the SYSTEM tablespace or another permanent default established by the system administrator. Do not store user data in the SYSTEM tablespace. Assigning a tablespace to be used specifically as a temporary tablespace eliminates file contention among temporary segments and other types of segments.

> **Note:**
> If your SYSTEM tablespace is locally managed, then users must be assigned a specific default (locally managed) temporary tablespace. They may not be allowed to default to using the SYSTEM tablespace because temporary objects cannot be placed in locally managed permanent tablespaces.

You can set the temporary tablespace for a user at user creation, and change it later using the ALTER USER statement. If you are logged in as user SYS, you can set a quota for the temporary tablespace, and other space allocations. (Only user SYS can do this, because all space in the temporary tablespace belongs to user SYS.) You can also establish tablespace groups instead of assigning individual temporary tablespaces.

See Also:

- "Temporary Tablespaces" in *Oracle Database Administrator's Guide*
- "Multiple Temporary Tablespaces: Using Tablespace Groups" in *Oracle Database Administrator's Guide*

Specifying a Profile for the User

You can specify a profile when you create a user. A profile is a set of limits on database resources and password access to the database. If you do not specify a profile, then Oracle Database assigns the user a default profile.

The following example demonstrates how to assign a user a profile.

```
CREATE USER jward
 IDENTIFIED BY password
 DEFAULT TABLESPACE data_ts
 QUOTA 100M ON test_ts
 QUOTA 500K ON data_ts
 TEMPORARY TABLESPACE temp_ts
 PROFILE clerk;
```

See Also:
"Managing Resources with Profiles"

Setting a Default Role for the User

A role is a named group of related privileges that you grant as a group to users or other roles. A default role is automatically enabled for a user when the user creates a session. You can assign a user zero or more default roles.

You cannot set default roles for a user in the `CREATE USER` statement. When you first create a user, the default role setting for the user is `ALL`, which causes all roles subsequently granted to the user to be default roles. Use the `ALTER USER` statement to change the default roles for the user. For example:

```
GRANT USER jward clerk_role;

ALTER USER jward DEFAULT ROLE clerk_role;
```

Before a role can be made the default role for a user, that user must have been already granted the role.

See Also:
"Managing User Roles"

Altering User Accounts

Users can change their own passwords. However, to change any other option of a user security domain, you must have the `ALTER USER` system privilege. Security administrators are typically the only users that have this system privilege, as it allows a modification of *any* user security domain. This privilege includes the ability to set tablespace quotas for a user on any tablespace in the database, even if the user performing the modification does not have a quota for a specified tablespace.

You can alter user security settings with the `ALTER USER` SQL statement. Changing user security settings affects the future user sessions, not current sessions.

Example 2-2 shows how to use the ALTER USER statement to alter the security settings for the user avyrros:

Example 2-2 Altering a User Account

```
ALTER USER avyrros
 IDENTIFIED EXTERNALLY
 DEFAULT TABLESPACE data_ts
 TEMPORARY TABLESPACE temp_ts
 QUOTA 100M ON data_ts
 QUOTA 0 ON test_ts
 PROFILE clerk;
```

The ALTER USER statement here changes the security settings for the user avyrros as follows:

- Authentication is changed to use the operating system account of the user avyrros.
- The default and temporary tablespaces are explicitly set for user AVYRROS.
- The user avyrros is given a 100M quota for the DATA_TS tablespace.
- The quota on the test_ts is revoked for the user avyrros.
- The user avyrros is assigned the clerk profile.

Changing the User Password

Most users can change their own passwords with the PASSWORD statement, as follows:

```
PASSWORD andy
Changing password for andy
New password: password
Retype new password: password
```

No special privileges (other than those to connect to the database and create a session) are required for a user to change his or her own password. Encourage users to change their passwords frequently. "Guidelines for Securing Passwords" provides advice on the best ways to secure passwords. You can find existing users for the current database instance by querying the ALL_USERS view.

Users can also use the ALTER USER SQL statement change their passwords. For example:

```
ALTER USER andy
 IDENTIFIED BY password
```

However, for better security, use the PASSWORD statement to change the account's password. The ALTER USER statement displays the new password on the screen, where it can be seen by any overly curious coworkers. The PASSWORD command does not display the new password, so it is only known to you, not to your co-workers. In both cases, the password is encrypted on the network.

Users must have the PASSWORD and ALTER USER privilege to switch between methods of authentication. Usually, only an administrator has this privilege.

See Also:

- "Minimum Requirements for Passwords" for the minimum requirements for creating passwords
- "Guidelines for Securing Passwords" for additional ways to secure passwords
- Chapter 3, "Configuring Authentication," for information about authentication methods that are available for Oracle Database

users

Configuring User Resource Limits

This section contains:

- About User Resource Limits
- Types of System Resources and Limits
- Determining Values for Resource Limits of Profiles
- Managing Resources with Profiles

About User Resource Limits

You can set limits on the amount of various system resources available to each user as part of the security domain of that user. By doing so, you can prevent the uncontrolled consumption of valuable system resources such as CPU time. To set resource limits, you use Database Resource Manager, which is described in *Oracle Database Administrator's Guide*.

This resource limit feature is very useful in large, multiuser systems, where system resources are very expensive. Excessive consumption of these resources by one or more users can detrimentally affect the other users of the database. In single-user or small-scale multiuser database systems, the system resource feature is not as important, because user consumption of system resources is less likely to have a detrimental impact.

You manage user resource limits by using Database Resource Manager. You can set password management preferences using profiles, either set individually or using a default profile for many users. Each Oracle database can have an unlimited number of profiles. Oracle Database allows the security administrator to enable or disable the enforcement of profile resource limits universally.

Setting resource limits causes a slight performance degradation when users create sessions, because Oracle Database loads all resource limit data for each user upon each connection to the database.

See Also:
Oracle Database Administrator's Guide for detailed information about managing resources

Types of System Resources and Limits

Oracle Database can limit the use of several types of system resources, including CPU time and logical reads. In general, you can control each of these resources at the session level, call level, or both, as discussed in the following sections:

- Limiting the User Session Level
- Limiting Database Call Levels
- Limiting CPU Time
- Limiting Logical Reads
- Limiting Other Resources

Limiting the User Session Level

Each time a user connects to a database, a session is created. Each session uses CPU time and memory on the computer that runs Oracle Database. You can set several resource limits at the session level.

If a user exceeds a session-level resource limit, then Oracle Database terminates (rolls back) the current statement and returns a message indicating that the session limit has been reached. At this point, all previous statements in the current transaction are intact, and the only operations the user can perform are `COMMIT`, `ROLLBACK`, or disconnect (in this case, the current transaction is committed). All other operations produce an error. Even after the transaction is committed or rolled back, the user cannot accomplish any more work during the current session.

Limiting Database Call Levels

Each time a user runs a SQL statement, Oracle Database performs several steps to process the statement. During this processing, several calls are made to the database as a part of the different execution phases. To prevent any one call from using the system excessively, Oracle Database lets you set several resource limits at the call level.

If a user exceeds a call-level resource limit, then Oracle Database halts the processing of the statement, rolls back the statement, and returns an error. However, all previous statements of the current transaction remain intact, and the user session remains connected.

Limiting CPU Time

When SQL statements and other types of calls are made to Oracle Database, a certain amount of CPU time is necessary to process the call. Average calls require a small amount of CPU time. However, a SQL statement involving a large amount of data or a runaway query can potentially use a large amount of CPU time, reducing CPU time available for other processing.

To prevent uncontrolled use of CPU time, you can set fixed or dynamic limits on the CPU time for each call and the total amount of CPU time used for Oracle Database calls during a session. The limits are set and measured in CPU one-hundredth seconds (0.01 seconds) used by a call or a session.

Limiting Logical Reads

Input/output (I/O) is one of the most expensive operations in a database system. SQL statements that are I/O-intensive can monopolize memory and disk use and cause other database operations to compete for these resources.

To prevent single sources of excessive I/O, you can limit the logical data block reads for each call and for each session. Logical data block reads include data block reads from both memory and disk. The limits are set and measured in number of block reads performed by a call or during a session.

Limiting Other Resources

Oracle Database provides for limiting several other resources at the session level:

- **You can limit the number of concurrent sessions for each user.** Each user can create only up to a predefined number of concurrent sessions.
- **You can limit the idle time for a session.** If the time between calls in a session reaches the idle time limit, then the current transaction is rolled back, the session is terminated, and the resources of the session are returned to the system. The next call receives an error that indicates that the user is no longer connected to the instance. This limit is set as a number of elapsed minutes.

Note:
Shortly after a session is terminated because it has exceeded an idle time limit, the process monitor (PMON) background process cleans up after the terminated session. Until PMON completes this process, the terminated session is still counted in any session or user resource limit.

- **You can limit the elapsed connect time for each session.** If the duration of a session exceeds the elapsed time limit, then the current transaction is rolled back, the session is dropped, and the resources of the session are returned to the system. This limit is set as a number of elapsed minutes.

Note:
Oracle Database does not constantly monitor the elapsed idle time or elapsed connection time. Doing so reduces system performance. Instead, it checks every few minutes. Therefore, a session can exceed this limit slightly (for example, by 5 minutes) before Oracle Database enforces the limit and terminates the session.

- **You can limit the amount of private System Global Area (SGA) space (used for private SQL areas) for a session.** This limit is only important in systems that use the shared server configuration. Otherwise, private SQL areas are located in the Program Global Area (PGA). This limit is set as a number of bytes of memory in the SGA of an instance. Use the characters **K** or **M** to specify kilobytes or megabytes.

See Also:
For instructions about enabling or disabling resource limits:

- "Finding Information About Database Users and Profiles"
- "Managing User Roles"
- *Oracle Database Administrator's Guide* for detailed information about managing resources

Determining Values for Resource Limits of Profiles

Before creating profiles and setting the resource limits associated with them, you should determine appropriate values for each resource limit. You can base these values on the type of operations a typical user performs. For example, if one class of user does not usually perform a high number of logical data block reads, then use the `ALTER RESOURCE COST` SQL statement to set the `LOGICAL_READS_PER_SESSION` setting conservatively.

Usually, the best way to determine the appropriate resource limit values for a given user profile is to gather historical information about each type of resource usage. For example, the database or security administrator can use the `AUDIT SESSION` clause to gather information about the limits `CONNECT_TIME`, `LOGICAL_READS_PER_SESSION`.

You can gather statistics for other limits using the Monitor feature of Oracle Enterprise Manager (or SQL*Plus), specifically the Statistics monitor.

See Also:

- "Using Data Dictionary Views to Find Information About Users and Profiles"

- Chapter 9, "Verifying Security Access with Auditing"
- *Oracle Database 2 Day DBA* for more information about Database Control
- Enterprise Manager online Help for more information about the Monitor feature

---

Managing Resources with Profiles

A **profile** is a named set of resource limits and password parameters that restrict database usage and instance resources for a user. You can assign a profile to each user, and a default profile to all others. Each user can have only one profile, and creating a new one supersedes an earlier version.

You need to create and manage user profiles only if resource limits are a requirement of your database security policy. To use profiles, first categorize the related types of users in a database. Just as roles are used to manage the privileges of related users, profiles are used to manage the resource limits of related users. Determine how many profiles are needed to encompass all types of users in a database and then determine appropriate resource limits for each profile.

In general, the word profile refers to a collection of attributes that apply to a user, enabling a single point of reference for any of multiple users that share those exact attributes. User profiles in Oracle Internet Directory contain attributes pertinent to directory usage and authentication for each user. Similarly, profiles in Oracle Label Security contain attributes useful in label security user administration and operations management. Profile attributes can include restrictions on system resources. You can use Database Resource Manager to set these types of resource limits.

Profile resource limits are enforced only when you enable resource limitation for the associated database. Enabling this limitation can occur either before starting up the database (using the `RESOURCE_LIMIT` initialization parameter) or while it is open (using the `ALTER SYSTEM` statement).

Though password parameters reside in profiles, they are unaffected by `RESOURCE_LIMIT` or `ALTER SYSTEM` and password management is always enabled. In Oracle Database, Database Resource Manager primarily handles resource allocations and restrictions.

See Also:

- *Oracle Database Administrator's Guide* for detailed information on managing resources
- "Finding Information About Database Users and Profiles" for viewing resource information
- *Oracle Database SQL Language Reference* for information about `ALTER SYSTEM` or `RESOURCE_LIMIT`

Creating Profiles

Any authorized database user can create, assign to users, alter, and drop a profile at any time (using the `CREATE USER` or `ALTER USER` statement). Profiles can be assigned only to users and not to roles or other profiles. Profile assignments do not affect current sessions, instead, they take effect only in subsequent sessions. To find information about current profiles, query the `DBA_PROFILES` view.

See Also:

- *Oracle Database SQL Language Reference* for more information about the SQL statements used for managing profiles, such as `CREATE PROFILE`, and for information about how to calculate composite limits.
- *Oracle Database Administrator's Guide* for detailed information about managing resources
- "Creating User Accounts"
- "Altering User Accounts"

Dropping Profiles

To drop a profile, you must have the `DROP PROFILE` system privilege. You can drop a profile (other than the default profile) using the SQL statement `DROP PROFILE`.To successfully drop a profile currently assigned to a user, use the `CASCADE` option.

The following statement drops the profile `clerk`, even though it is assigned to a user:

```
DROP PROFILE clerk CASCADE;
```

Any user currently assigned to a profile that is dropped is automatically assigned to the `DEFAULT` profile. The `DEFAULT` profile cannot be dropped. When a profile is dropped, the drop does not affect currently active sessions. Only sessions created after a profile is dropped use the modified pro file assignments.

Deleting User Accounts

When you drop a user account, Oracle Database removes the user account and associated schema from the data dictionary. It also immediately drops all schema objects contained in the user schema, if any.

Notes:

- If a user schema and associated objects must remain but the user must be denied access to the database, then revoke the `CREATE SESSION` privilege from the user.
- Do not attempt to drop the `SYS` or `SYSTEM` user. Doing so corrupts your database.

A user that is currently connected to a database cannot be dropped. To drop a connected user, you must first terminate the user sessions using the SQL statement `ALTER SYSTEM` with the `KILL SESSION` clause. You can find the session ID (SID) by querying the `V$SESSION` view.

Example 2-3 shows how to query `V$SESSION` and displays the session ID, serial number, and user name for user `ANDY`.

Example 2-3 Querying V$SESSION for the Session ID of a User

```
SELECT SID, SERIAL#, USERNAME FROM V$SESSION;

    SID         SERIAL#     USERNAME
------- --------------- ----------------------
    127           55234      ANDY
...
```

Example 2-4 shows how to stop the session for user `andy`.

Example 2-4 Killing a User Session

```
ALTER SYSTEM KILL SESSION '127, 55234';
```

You can drop a user from a database using the `DROP USER` statement. To drop a user and all the user schema objects (if any), you must have the `DROP USER` system privilege. Because the `DROP USER` system privilege is powerful, a security administrator is typically the only type of user that has this privilege.

If the schema of the user contains any dependent schema objects, then use the `CASCADE` option to drop the user and all associated objects and foreign keys that depend on the tables of the user successfully. If you do not specify `CASCADE` and the user schema contains dependent objects, then an error message is returned and the user is not dropped.

Before dropping a user whose schema contains objects, thoroughly investigate which objects the schema contains and the implications of dropping them. You can find the objects owned by a particular user by querying the `DBA_OBJECTS` view.

Example 2-5 shows how to find the objects owned by user `andy`.

Example 2-5 Finding Objects Owned by a User

```
SELECT OWNER, OBJECT_NAME FROM DBA_OBJECTS WHERE OWNER LIKE 'ANDY';
```

(Enter the user name in capital letters.) Pay attention to any unknown cascading effects. For example, if you intend to drop a user who owns a table, then check whether any views or procedures depend on that particular table.

Example 2-6 drops the user `andy` and all associated objects and foreign keys that depend on the tables owned by `andy`.

Example 2-6 Dropping a User Account

```
DROP USER andy CASCADE;
```

See Also:
Oracle Database Administrator's Guide for more information about terminating sessions

---

Finding Information About Database Users and Profiles

This section contains:

- Using Data Dictionary Views to Find Information About Users and Profiles
- Listing All Users and Associated Information
- Listing All Tablespace Quotas
- Listing All Profiles and Assigned Limits
- Viewing Memory Use for Each User Session

Using Data Dictionary Views to Find Information About Users and Profiles

Table 2-1 lists data dictionary views that contain information about database users and profiles. For detailed information about these views, see *Oracle Database Reference*.

Table 2-1 Data Dictionary Views That Display Information about Users and Profiles

View	Description
ALL_OBJECTS	Describes all objects accessible to the current user
ALL_USERS	Lists users visible to the current user, but does not describe them
DBA_PROFILES	Displays all profiles and their limits
DBA_TS_QUOTAS	Describes tablespace quotas for users
DBA_OBJECTS	Describes all objects in the database
DBA_USERS	Describes all users of the database
DBA_USERS_WITH_DEFPWD	Lists all user accounts that have default passwords
PROXY_USERS	Describes users who can assume the identity of other users
RESOURCE_COST	Lists the cost for each resource in terms of CPUs for each session, reads for each session, connection times, and SGA
USER_PASSWORD_LIMITS	Describes the password profile parameters that are assigned to the user
USER_RESOURCE_LIMITS	Displays the resource limits for the current user
USER_TS_QUOTAS	Describes tablespace quotas for users
USER_OBJECTS	Describes all objects owned by the current user
USER_USERS	Describes only the current user
V$SESSION	Lists session information for each current session, includes user name
V$SESSTAT	Lists user session statistics
V$STATNAME	Displays decoded statistic names for the statistics shown in the V$SESSTAT view

The following sections present examples of using these views. These examples assume that the following statements have been run:

```
CREATE PROFILE clerk LIMIT
    SESSIONS_PER_USER 1
    IDLE_TIME 30
    CONNECT_TIME 600;

CREATE USER jfee
    IDENTIFIED BY password
    DEFAULT TABLESPACE users
    TEMPORARY TABLESPACE temp_ts
    QUOTA 500K ON users
    PROFILE clerk;

CREATE USER dcranney
    IDENTIFIED BY password
    DEFAULT TABLESPACE users
    TEMPORARY TABLESPACE temp_ts
    QUOTA unlimited ON users;

CREATE USER userscott
     IDENTIFIED BY password;
```

Listing All Users and Associated Information

To find all users and their associated information as defined in the database, query the DBA_USERS view. For detailed information on the DBA_USERS view, see *Oracle Database Reference*.

For example:

```
SELECT USERNAME, PROFILE, ACCOUNT_STATUS, AUTHENTICATION_TYPE FROM DBA_USI

USERNAME        PROFILE         ACCOUNT_STATUS   AUTHENTICATION_TYPE
--------------- --------------- ---------------  -------------------
SYS             DEFAULT         OPEN             PASSWORD
SYSTEM          DEFAULT         OPEN             PASSWORD
USERSCOTT       DEFAULT         OPEN             PASSWORD
JFEE            CLERK           OPEN             GLOBAL
```

```
DCRANNEY        DEFAULT         OPEN            EXTERNAL
```

Listing All Tablespace Quotas

Use the DBA_TS_QUOTAS view to list all tablespace quotas specifically assigned to each user. (For detailed information on this view, see *Oracle Database Reference*.) For example:

```
SELECT * FROM DBA_TS_QUOTAS;

TABLESPACE    USERNAME    BYTES     MAX_BYTES    BLOCKS    MAX_BLOCKS
----------    ---------  --------   ----------   -------   ----------
USERS         JFEE              0       512000         0          250
USERS         DCRANNEY          0           -1         0           -1
```

When specific quotas are assigned, the exact number is indicated in the MAX_BYTES column. This number is always a multiple of the database block size, so if you specify a tablespace quota that is not a multiple of the database block size, then it is rounded up accordingly. Unlimited quotas are indicated by -1.

Listing All Profiles and Assigned Limits

The DBA_PROFILE view lists all profiles in the database and associated settings for each limit in each profile. (For detailed information on this view, see *Oracle Database Reference*.) For example:

```
SELECT * FROM DBA_PROFILES
   ORDER BY PROFILE;

PROFILE             RESOURCE_NAME              RESOURCE   LIMIT
-----------------   --------------             ---------- --------------
CLERK               COMPOSITE_LIMIT            KERNEL     DEFAULT
CLERK               FAILED_LOGIN_ATTEMPTS      PASSWORD   DEFAULT
CLERK               PASSWORD_LIFE_TIME         PASSWORD   DEFAULT
CLERK               PASSWORD_REUSE_TIME        PASSWORD   DEFAULT
CLERK               PASSWORD_REUSE_MAX         PASSWORD   DEFAULT
CLERK               PASSWORD_VERIFY_FUNCTION   PASSWORD   DEFAULT
CLERK               PASSWORD_LOCK_TIME         PASSWORD   DEFAULT
CLERK               PASSWORD_GRACE_TIME        PASSWORD   DEFAULT
CLERK               PRIVATE_SGA                KERNEL     DEFAULT
CLERK               CONNECT_TIME               KERNEL     600
CLERK               IDLE_TIME                  KERNEL     30
CLERK               LOGICAL_READS_PER_CALL     KERNEL     DEFAULT
CLERK               LOGICAL_READS_PER_SESSION  KERNEL     DEFAULT
CLERK               CPU_PER_CALL               KERNEL     DEFAULT
CLERK               CPU_PER_SESSION            KERNEL     DEFAULT
CLERK               SESSIONS_PER_USER          KERNEL     1
DEFAULT             COMPOSITE_LIMIT            KERNEL     UNLIMITED
DEFAULT             PRIVATE_SGA                KERNEL     UNLIMITED
DEFAULT             SESSIONS_PER_USER          KERNEL     UNLIMITED
DEFAULT             CPU_PER_CALL               KERNEL     UNLIMITED
DEFAULT             LOGICAL_READS_PER_CALL     KERNEL     UNLIMITED
DEFAULT             CONNECT_TIME               KERNEL     UNLIMITED
DEFAULT             IDLE_TIME                  KERNEL     UNLIMITED
DEFAULT             LOGICAL_READS_PER_SESSION  KERNEL     UNLIMITED
DEFAULT             CPU_PER_SESSION            KERNEL     UNLIMITED
DEFAULT             FAILED_LOGIN_ATTEMPTS      PASSWORD   10
DEFAULT             PASSWORD_LIFE_TIME         PASSWORD   180
DEFAULT             PASSWORD_REUSE_MAX         PASSWORD   UNLIMITED
DEFAULT             PASSWORD_LOCK_TIME         PASSWORD   1
DEFAULT             PASSWORD_GRACE_TIME        PASSWORD   7
DEFAULT             PASSWORD_VERIFY_FUNCTION   PASSWORD   UNLIMITED
DEFAULT             PASSWORD_REUSE_TIME        PASSWORD   UNLIMITED
32 rows selected.
```

Viewing Memory Use for Each User Session

To find the memory use for each user session, query the V$SESSION view. (For detailed information on this view, see *Oracle Database Reference*. The following query lists all current sessions, showing the Oracle Database user and current User Global Area (UGA)

memory use for each session:

```
SELECT USERNAME, VALUE || 'bytes' "Current UGA memory"
   FROM V$SESSION sess, V$SESSTAT stat, V$STATNAME name
WHERE sess.SID = stat.SID
   AND stat.STATISTIC# = name.STATISTIC#
   AND name.NAME = 'session uga memory';

USERNAME                       Current UGA memory
------------------------------ ----------------------------------------
                               18636bytes
                               17464bytes
                               19180bytes
                               18364bytes
                               39384bytes
                               35292bytes
                               17696bytes
                               15868bytes
USERSCOTT                      42244bytes
SYS                            98196bytes
SYSTEM                         30648bytes

11 rows selected.
```

To see the maximum UGA memory allocated to each session since the instance started, replace `'session uga memory'` in the preceding query with `'session uga memory max'`.
