Managing Users

· Creating new database users
· Altering and dropping existing users
· Monitoring information about existing users

Creating New Database Users

One of the primary tasks early on in the creation of a new database is adding new users. However, user creation is an ongoing task. As users enter and leave the organization, so too must the DBA keep track of access to the database granted to those users. When using Oracle’s own database authentication method, new users are created with the create user statement:

CREATE USER Krishnamoorthy
IDENTIFIED BY first01
DEFAULT TABLESPACE users_01
TEMPORARY TABLESPACE temp_01
QUOTA 10M ON users_01
PROFILE app_developer
PASSWORD EXPIRE
ACCOUNT UNLOCK;

This statement highlights several items of information that comprise the syntax and semantics of user creation:

**CREATE USER**

The user’s name in Oracle. The name should also start with a letter. On single-byte character sets, the name can be from 1 to 30 characters long. In addition, the name should contain one single-byte character. The username is not case sensitive and cannot be a reserved word.

**IDENTIFIED BY**
The user’s password in Oracle. This item should contain at least three characters, and preferably six or more. Generally, it is recommended that users change their password once they know their username is created.

**DEFAULT TABLESPACE**

Tablespace management is a crucial task in Oracle. The default tablespace names the location where the user’s database objects are created by default.

**TEMPORARY TABLESPACE**

If temporary tablespace is not explicitly specified by the DBA when the username is created, the location for all temporary segments for that user will be the SYSTEM tablespace. SYSTEM, as you already know, is a valuable resource that should not be used for user object storage.

**QUOTA**

A quota is a limit on the amount of space the user’s database objects can occupy within the tablespace. If a user attempts to create a database object that exceeds that user’s quota for that tablespace, then the object creation script will fail. Quotas can be specified either in kilobytes (K) or megabytes (M).

**PROFILE**

Profiles are a bundled set of resource-usage parameters that the DBA can set in order to limit the user’s overall host machine utilization. To reduce the chance that one user could affect the overall database performance with, say, a poorly formulated ad hoc report that drags the database to its knees, you may assign profiles for each user that limit the amount of time they can spend on the system.

**PASSWORD EXPIRE**

This clause enforces the requirement that a user change his or her password on first logging into Oracle. This extra level of password security guarantees that, the DBA, will know a user’s password.

**ACCOUNT UNLOCK**

This is the default for user accounts created. It means that the user’s account is available for use immediately. The DBA can prevent users from using their accounts by specifying account lock instead.

**Guidelines for User-Account Management**

The following list identifies several new guidelines to follow when managing user accounts.
➢ Use a standard password for user creation, such as 123abc or first1, and use password expire to force users to change this password to something else the first time they log into Oracle.
➢ Avoid OS authentication unless all your users will access Oracle while connected directly to the machine hosting your database.
➢ Be sure to always assign temporary tablespace and default tablespace to users.
➢ Give few users quota unlimited. Although it’s annoying to have users asking for more space, it’s even more annoying to reorganize tablespaces carelessly filled with database objects.
➢ Become familiar with the user-account management and other host machine limits that can be set via profiles. These new features take Oracle user-account management to new levels of security.

************
**Altering and Dropping Existing Users**

Once a user is created, there are a few reasons you’ll need to modify that user. One is to expire the password if a user forgets it, so that the next time the user logs in, the password can be changed by the user. The alter user identified by statement is used to change the user’s password:

```
ALTER USER Krishnamoorthy
IDENTIFIED BY forgotpassword
PASSWORD EXPIRE;
```

In certain situations, as the result of user profiles, a user’s account may become locked. This may occur if the user forgot his or her password and tried to log in using a bad password too many times.

To unlock a user’s account while also making it possible for the user to change the password, the following alter user statement can be used:

```
ALTER USER Krishnamoorthy
IDENTIFIED BY forgotpassword
ACCOUNT UNLOCK
PASSWORD EXPIRE;
```

In an attempt to prevent misuse, you may want to lock an account that has been used many times unsuccessfully to gain access to Oracle, with the following statement:

```
ALTER USER athena
ACCOUNT LOCK;
```

**Changing User Tablespace Allocation**

You may want to reorganize tablespaces to distribute I/O load and make more effective use of the hardware running Oracle.

```
ALTER USER Krishnamoorthy
DEFAULT TABLESPACE overflow_tabspc01;
```

```
ALTER USER Krishnamoorthy
TEMPORARY TABLESPACE temp_overflow_01;
```
A quota can be altered by the DBA with the alter user quota statement. For example, the DBA may want to reduce the quota on the USERS_01 tablespace from 10MB to 5MB for user Krishnamoorthy.

```
ALTER USER Krishnamoorthy
QUOTA 5M ON users_01;
```

**Dropping User Accounts**

As users come and go, their access should be modified to reflect their departure. To drop a user from the database, you execute the drop user statement. If a user has created database objects, the user cannot be dropped until the objects are dropped, as well. In order to drop the user and all related database objects in one fell swoop, Oracle provides the cascade option.

```
DROP USER Krishnamoorthy CASCADE;
```

*****
Monitoring Information About Existing Users

The DBA may periodically want to monitor information about users. Several data dictionary views may be used for the purpose of obtaining information about users. Some information a DBA may want to collect includes default and temporary tablespace information, objects created by that user, and what the current account status for that user account is. The following data dictionary views can be used to determine this information.

1. **DBA_USERS** Contains username, Oracle-generated ID number, encrypted password, default and temporary tablespace information, and the user profile that was specified in the ID creation statements or any alteration that may have followed.

2. **DBA_OBJECTS** Contains the specific information about every object in the database. The DBA can determine which objects belong to which users by using the OWNER column of this view.

3. **DBA_TS_QUOTAS** Names all users and any tablespace quotas that have been created for them.

**********
Managing Privileges

· Identifying system and object privileges
· Granting and revoking privileges
· Controlling OS or password authentication

All access in an Oracle database requires database privileges. Access to connect to the database, the objects the user is permitted to see, and the objects the user is allowed to create are all controlled by privileges. Use of every database object and system resource is governed by privileges. There are privileges required to create objects, to access objects, to change data within tables, to execute stored procedures, to create users, and so on.

Identifying System Privileges

There are two categories of privileges, and the first is system privileges.

System privileges control the creation and maintenance of many database objects, such as rollback segments, synonyms, tables, and triggers. Additionally, the ability to use the analyze command and the Oracle database audit capability is governed by system privileges.

ADMIN FUNCTIONS

These privileges relate to activities typically reserved for and performed by the DBA. Privileges include alter system, audit system, audit any, alter database, analyze any, sysdba, sysoper, and grant any privilege.

DATABASE ACCESS

These privileges control who accesses the database, when they can access it, and what they can do regarding management of their own session. Privileges include create session, alter session, and restricted session.

TABLESPACES

Tablespaces are disk resources used to store database objects. These privileges determine who can maintain these disk resources. These privileges are typically reserved for DBAs. Privileges include create tablespace, alter tablespace, manage tablespace, drop tablespace, and unlimited tablespace.
** USERS **
These privileges are used to manage users on the Oracle database. Typically, these privileges are reserved for DBAs or security administrators. Privileges include create user, become user, alter user, and drop user.

** ROLLBACK SEGMENTS **
Rollback segments are disk resources that make aspects of transaction processing possible. The privileges include create rollback segment, alter rollback segment, and drop rollback segment.

** TABLES **
Tables store data in the Oracle database. The privileges include create table, create any table, alter any table, backup any table, drop any table, lock any table, comment any table, select any table, insert any table, update any table, and delete any table. The create table or create any table privilege also allows you to drop the table.

** CLUSTERS **
Clusters are used to store tables commonly used together in close physical proximity on disk. The privileges include create cluster, create any cluster, alter any cluster, and drop any cluster. The create cluster and create any cluster privileges also allow you to alter and drop those clusters.

** INDEXES **
Indexes are used to improve SQL statement performance on tables containing lots of row data. The privileges include create any index, alter any index, and drop any index. The create table privilege also allows you to alter and drop indexes.

** SYNONYMS **
A synonym is a database object that allows you to reference another object by a different name. A public synonym means that the synonym is available to every user in the database for the same purpose. The privileges include create synonym, create any synonym, drop any synonym, create public synonym, and drop public synonym. The create synonym privilege also allows you to alter and drop synonyms that you own.

** VIEWS **

A view is an object containing a SQL statement that behaves like a table in Oracle, except that it stores no data. The privileges include create view, create any view, and drop any view. The create view privilege also allows you to alter and drop views that you own.

SEQUENCES

A sequence is an object in Oracle that generates numbers according to rules you can define. Privileges include create sequence, create any sequence, alter any sequence, drop any sequence, and select any sequence. The create sequence privilege also allows you to drop sequences that you own.

DATABASE LINKS

Database links are objects in Oracle that, within your session connected to one database, allow you to reference tables in another Oracle database without making a separate connection. A public database link is one available to all users in Oracle, while a private database link is one that only the owner can use. The create database link privilege also allows you to drop private database links that you own.

ROLES

Roles are objects that can be used for simplified privilege management. You create a role, grant privileges to it, and then grant the role to users. Privileges include create role, drop any role, grant any role, and alter any role.

TRANSACTIONS

These privileges are for resolving in-doubt transactions being processed on the Oracle database. Privileges include force transaction and force any transaction.

PL/SQL

You have already been introduced to the different PL/SQL blocks available in Oracle. These privileges allow you to create, run, and manage those different types of blocks. Privileges include create procedure, create any procedure, alter any procedure, drop any procedure, and execute any procedure. The create procedure privilege also allows you to alter and drop PL/SQL blocks that you own.

TRIGGERS

Triggers are PL/SQL blocks in Oracle that execute when a specified DML activity occurs on the table to which the trigger is associated. The create trigger privilege also allows you to alter and drop triggers that you own.
**PROFILES**
Profiles are objects in Oracle that allow you to impose limits on resources for users in the machine hosting Oracle. Privileges include create profile, alter profile, drop profile, and alter resource cost.

**SNAPSHOTS**
Snapshots are objects in Oracle that allow you to replicate data from a table in one database to a copy of the table in another. Privileges include create snapshot, create any snapshot, alter any snapshot, and drop any snapshot.

**DIRECTORIES**
Directories in Oracle are objects that refer to directories on the machine hosting the Oracle database. Privileges include create any directory and drop any directory.

**TYPES**
Types in Oracle correspond to user-defined types you can create using Oracle8’s Objects option. Privileges include create type, create any type, alter any type, drop any type, and execute any type. The create type privilege also allows you to alter and drop types that you own.

**LIBRARIES**
A library is an object that allows you to reference a set of procedures external to Oracle. Currently, only C procedures are supported. Privileges include create library, create any library, alter any library, drop any library, and execute any library.

**********
Identifying Object Privileges

The other category of privileges granted on the Oracle database is the set of object privileges. Object privileges permit the owner of database objects, such as tables, to administer access to those objects according to the following types of access. The eight types of object privileges are as follows:

- **Select**  Permits the grantee of this object privilege to access the data in a table, sequence, view, or snapshot.
- **Insert**  Permits the grantee of this object privilege to insert data into a table or, in some cases, a view.
- **Update**  Permits the grantee of this object privilege to update data into a table or view.
- **Delete**  Permits the grantee of this object privilege to delete data from a table or view.
- **Alter**  Permits the grantee of this object privilege to alter the definition of a table or sequence only. The alter privileges on all other database objects are considered system privileges.
- **Index**  Permits the grantee of this object privilege to create an index on a table already defined.
- **References**  Permits the grantee to create or alter a table in order to create a FOREIGN KEY constraint against data in the referenced table.
- **Execute**  Permits the grantee to run a stored procedure or function.

**********
Granting and Revoking Privileges

Giving privileges to users is done with the grant command. System privileges are first given to the SYS and SYSTEM users, and to any other user with the grant any privilege permission.

For example, executing the following grant statements gives access to create a table to user Krishnamoorthy, and object privileges on another table in the database:

```
GRANT CREATE TABLE TO Krishnamoorthy;          -- system
GRANT SELECT, UPDATE ON SRM.emp TO Krishnamoorthy; -- object
```

**Giving Administrative Ability Along with Privileges**

At the end of execution for the preceding two statements, Krishnamoorthy will have the ability to execute the create table command in her user schema and to select and update row data on the SRM.EMP table.

In order to give user Krishnamoorthy some additional power to administer to other users the privileges granted to her, the DBA can execute the following queries:

```
GRANT CREATE TABLE TO Krishnamoorthy WITH ADMIN OPTION;
GRANT SELECT, UPDATE ON SRM.emp TO Krishnamoorthy WITH GRANT OPTION;
```

The with admin option clause gives Krishnamoorthy the ability to give or take away the system privilege to others. Additionally, it gives Krishnamoorthy the ability to make other users administrators of that same privilege.

No additional syntax is necessary for revoking either a system privilege granted with admin option or an object privilege granted with grant option.

```
REVOKE CREATE TABLE FROM Krishnamoorthy;
REVOKE SELECT, UPDATE ON SRM.emp FROM Krishnamoorthy;
```

In the same way, roles can be revoked from users, even if the user created the role and thus has the admin option. The ability to revoke any role comes from the grant any role privilege, while the ability to
grant or revoke certain system privileges comes from being granted the privilege with the admin option. When a system privilege is revoked, there are no cascading events that take place along with it.

```
GRANT privilege ON object TO user;
```

To grant object privileges to others, you must own the database object, you must have been given the object privilege with grant option, or you must have the grant any privilege ability given to you. In addition to granting object privileges on database objects, privileges can also be granted on columns within the database object.

**OPEN TO THE PUBLIC**

Another aspect of privileges and access to the database involves a special user on the database. This user is called PUBLIC. If a system privilege, object privilege, or role is granted to the PUBLIC user, then every user in the database has that privilege.

**********
Managing Resource Use

1. Creating and assigning profiles to control resource use
2. Altering and dropping profiles
3. Administering passwords using profiles
4. Obtaining profile information from the data dictionary

Oracle’s use of the host machine on behalf of certain users can be managed by creating specific user profiles to correspond to the amount of activity anticipated by average transactions generated by those different types of users. The principle of user profiles is not to force the user off the system every time an artificially low resource-usage threshold is exceeded. Allow the users to do everything they need to on the Oracle database, while also limiting unwanted or unacceptable use.

Creating and Assigning Profiles to Control Resource Use

A special user profile exists in Oracle at database creation called DEFAULT. If no profile is assigned with the profile clause of the create user statement, the DEFAULT profile is assigned to that user. DEFAULT gives users unlimited use of all resources definable in the database. You might create a user profile like the one in the following code block:

CREATE PROFILE developer LIMIT
SESSIONS_PER_USER 1
CPU_PER_SESSION 10000
CPU_PER_CALL 20
CONNECT_TIME 240
IDLE_TIME 20
LOGICAL_READS_PER_SESSION 50000
LOGICAL_READS_PER_CALL 400
PRIVATE_SGA 1024;

This code block is a good example of using profiles to set individual resource limits. All other resources that are not explicitly assigned limits when you create a profile will be assigned the default values specified in the DEFAULT profile.

Once profiles are created, they are assigned to users with the profile clause in either the create user or alter user statement. The following code block contains examples:

CREATE USER GITANJAN
IDENTIFIED BY orange#tabby
TEMPORARY TABLESPACE temp_01
QUOTA 5M ON temp_01
PROFILE developer;

ALTER USER GITANJAN
PROFILE developer;

Setting Individual Resource Limits: Session Level

The following resource-usage areas can have limits assigned for them within the profiles you create. If a session-level resource limit is exceeded, the user gets an error and the session is terminated automatically. At the session level, the resource limits are as follows:

- **sessions_per_user** The number of sessions a user can open concurrently with the Oracle database.
- **cpu_per_session** The maximum allowed CPU time in 1/100 seconds that a user can utilize in one session.
- **logical_reads_per_session**  The maximum number of disk I/O block reads that can be executed in support of the user processing in one session.

- **idle_time**  The time in minutes that a user can issue no commands before Oracle times out their session.

- **connect_time**  The total amount of time in minutes that a user can be connected to the database.

- **private_sga**  The amount of private memory in kilobytes or megabytes that can be allocated to a user for private storage.

### Individual Resource Limits: Call Level

At the call level, the resource-usage areas can have limits assigned for them within the profiles you create. Call-level usage limits are identified as follows:

- **logical_reads_per_call**  The maximum number of disk I/O block reads that can be executed in support of the user’s processing in one session.

- **cpu_per_call**  The maximum allowed CPU time in 1/100 seconds that any individual operation in a user session can use.

### Enabling Resource Limits

To use resource limits, you must first change the RESOURCE_LIMIT initsid.ora parameter to TRUE on your Oracle database. To enable resource restriction, the DBA should issue the following statement:

```sql
ALTER SYSTEM
SET RESOURCE_LIMIT = TRUE;
```

******
Administering Passwords Using Profiles

Four new features exist in Oracle8 to handle password management more effectively. These features are

- account locking,
- password aging and expiration,
- password history, and
- password complexity requirements.

These new features are designed to make it harder than ever to hack the Oracle8 database as an authorized user without knowing the user’s password. This protects the integrity of assigned usernames, as well as the overall data integrity of the Oracle database.

Though not required to enable password management in Oracle8, the DBA can run the utlpwdmg.sql script as SYS to support the functionality of password management. When the password management script is run, all default password management settings placed in the DEFAULT profile are enforced at all times on the Oracle8 database.

Account Management

Account locking allows Oracle8 to lock out an account when users attempt to log into the database unsuccessfully on several attempts. The maximum allowed number of failed attempts is defined per user or by group. The number of failed attempts is specified by the DBA or security officer in ways that will be defined shortly, and tracked by Oracle such that if the user fails to log into the database in the specified number of tries, Oracle locks out the user automatically. In addition, a time period for automatic user lockout can be defined such that the failed login attempt counter will reset after that time period, and the user may try to log into the database again.

Password Aging and Rotation

A password is also aged in the Oracle8 database. The DBA or security administrator can set a password to have a maximum lifetime in the Oracle database. Once a threshold time period passes, the user must
change his or her password or be unable to access the Oracle database. A grace period can be defined, during which the user must change the password. If the time of the grace period passes and the user doesn’t change the password, the account is then locked and only the security administrator can unlock it. A useful technique for creating new users is to create them with expired passwords, such that the user enters the grace period on first login and must change the password during that time.

A potential problem arises when users are forced to change their passwords. Sometimes users try to “fool” the system by changing the expired password to something else, and then immediately changing the password back. To prevent this, Oracle8 supports a password history feature that keeps track of recently used passwords and disallows their use for a specified amount of time or number of changes.

**Password Complexity Verification**

The most important to the integrity of an Oracle user’s account, is the feature of password complexity verification. There are many commonly accepted practices in creating a password, such as making sure it has a certain character length, that it is not a proper name or word in the dictionary, that it is not all numbers or all characters, and so on.

To prevent users from unwittingly subverting the security of the database, Oracle8 supports the automatic verification of password complexity with the use of a PL/SQL function that can be applied during user or group profile creation to prevent users from creating passwords of insufficient complexity. The checks provided by the default function include making sure the minimum password length is four characters and is not the same as the username. Also, the password must contain at least one letter, number, and punctuation character, and the password must be different from the previous password defined by at least three characters.

The overall call syntax must conform to the details in the following code listing. In addition, the new routine must be assigned as the password verification routine in the user’s profile or the DEFAULT profile. In the create profile statement, the following must be present: password_verify_function user_pwcmlpx_fname, where user_pwcmlpx_fname is the name of the user-defined password complexity function.
The call to the PL/SQL complexity verification function must conform to the following parameter-passing and return-value requirements:

\[
USER_PWCMPLX_FNAME \\
(\text{user\_id\_parm} \quad \text{IN VARCHAR2}, \\
\text{new\_passwd\_parm} \quad \text{IN VARCHAR2}, \\
\text{old\_passwd\_parm} \quad \text{IN VARCHAR2}) \quad \text{RETURN BOOLEAN};
\]

**Password Management Resource Limits in the DEFAULT Profile**

After the utlpwdmg.sql script is run, default values will specified for several password-management resource limits. An explanation of each option is listed below, along with its default value:

- **failed_login_attempts** Number of unsuccessful attempts at login a user can make before account locks. Default is 3.
- **password_life_time** Number of days a password will remain active. Default is 60.
- **password_reuse_time** Number of days before the password can be reused. Default is 1,800 (approximately 5 years).
- **password_reuse_max** Number of times the password must be changed before one can be reused. Default is unlimited.
- **password_lock_time** Number of days after which Oracle will unlock a user account locked automatically when the user exceeds failed_login_attempts. Default is 1/1,440 (1 minute).
- **password_grace_time** Number of days during which an expired password must be changed by the user or else Oracle permanently locks the account. Default is 10.
- **password_verify_function** Function used for password complexity verification. The default function is called verify_function().
Auditing the Database

- Differentiating between database and value-based auditing
- Using database auditing
- Viewing enabled auditing options
- Retrieving and maintaining auditing information

Several things about your database are always audited. They include privileged operations that DBAs typically perform, such as starting and stopping the instance and logins as sysdba or as sysoper. You can find information about these activities in the ALERT log on your database, along with information about log switches, checkpoints, and tablespaces taken offline or put online. Audit information is stored in a few different places in Oracle, depending on whether you specify your audit trail to be maintained within Oracle or in an operating system file.

Differentiating Between Database and Value-Based Auditing

There is a difference between database auditing and value-based auditing. Database auditing pertains to audits on database object access, user session activity, startup, shutdown, and other database activity. The information about these database events is stored in the audit trail, and the information can then be used to monitor potentially damaging activities, such as rows being removed from tables.

Value-based auditing pertains to audits on actual column/row values that are changed as the result of database activity. The Oracle audit trail does not track value-based audit information, so instead you must develop triggers, tables, PL/SQL code, or client applications that handle this level of auditing in the database.

Using Database Auditing

A database audit is most effective when the DBA or security administrator knows what he or she is looking for. The best way to conduct a database audit is to start the audit with a general idea about what
may be occurring on the database. Once the goals are established, set the audit to monitor those aspects
of database use and review the results to either confirm or disprove the hypothesis.

**Using the Audit Command for Privilege or Statement Audits**

After deciding what to audit, you must begin auditing by setting the AUDIT_TRAIL init.ora
parameter appropriately.

The general syntax for setting up auditing on statements or system privileges is as follows. The
following code block shows an example of an audit statement:

```
AUDIT CREATE TABLE, ALTER TABLE, DROP TABLE
BY Krishnamoorthy
WHENEVER SUCCESSFUL;
```

The following code block demonstrates use of the default keyword:

```
AUDIT INSERT
ON DEFAULT
WHENEVER SUCCESSFUL;
```

**Using the Audit Command for Object Audits**

Any privilege that can be granted can also be audited. However, since there are nearly 100 system and
object privileges that can be granted on the Oracle database, the creation of an audit statement can be an
excessively long task.

Oracle allows the administrator to specify the name of an object to audit, and Oracle will audit all
privileged operations.

```
AUDIT TABLE
BY Mareeswaran
WHENEVER SUCCESSFUL;
```

Finally, the person setting up auditing can also specify that audit records are to be compiled by session.
This means that audit will record data for audited activities in every session, as opposed to by access.
Eliminating the when successful clause tells audit to record every table creation, alteration, or drop
activity for every session that connects to the database, regardless of whether or not they were successful.

AUDIT TABLE
BY SESSION;

**Using Audit Definition Shortcuts**

There are other options available to consolidate the specification of database activities into one easy command for auditing. These commands are listed here:

- **Connect** Audits the user connections to the database. Can be substituted with session for the same effect. Audits the login and logout activities of every database user.
- **Resource** Audits detailed information related to the activities typically performed by an application developer or a development DBA, such as creating tables, views, clusters, links, stored procedures, and rollback segments.
- **dba** Audits activities related to “true” database administration, including the creation of users and roles, and granting system privileges and system audits.
- **All** Is the equivalent of an “on/off” switch, where all database activities are monitored and recorded.

**Disabling Audit Configuration**

There are two methods used to disable auditing. The first method is to change the initialization parameter AUDIT_TRAIL to NONE. On database shutdown and restart, this option will disable the audit functionality on the Oracle database. The other option used for changing the activities audit will record is called noaudit. This option can be executed in two ways. The first is used to turn off selective areas that are currently being audited.

NOAUDIT INSERT ON application.products;

**Retrieving and Maintaining Auditing Information**

The following data dictionary views are used to find results from audits currently taking place in the Oracle database.

- **DBA_AUDIT_EXISTS** A list of audit entries generated by the exists option of the audit command.
- **DBA_AUDIT_OBJECT**  A list of audit entries generated for object audits.
- **DBA_AUDIT_SESSION**  A list of audit entries generated by session connects and disconnects.
- **DBA_AUDIT_STATEMENT**  A list of audit entries generated by statement options of the audit command.
- **DBA_AUDIT_TRAIL**  A list of all entries in the AUD$ table collected by the audit command.

**Unit IV Part-A Questions**

1. **Write a short note on ‘quota’**.

   A quota is a limit on the amount of space the user’s database objects can occupy within the tablespace. If a user attempts to create a database object that exceeds that user’s quota for that tablespace, then the object creation script will fail. Quotas can be specified either in kilobytes (K) or megabytes (M).

2. **Explain the call-level usage limits**.

   At the call level, the resource-usage areas can have limits assigned for them within the profiles you create. If the user exceeds the call-level usage limits they have been assigned, the SQL statement that produced the error is terminated, any transaction changes made by the offending statement only are rolled back, previous statements remain intact, and the user remains connected to Oracle. Call-level usage limits are identified as follows:

   - **logical_reads_per_call**  The maximum number of disk I/O block reads that can be executed in support of the user’s processing in one session.
   - **cpu_per_call**  The maximum allowed CPU time in 1/100 seconds that any individual operation in a user session can use.

3. **Explain any two object privileges**

   - **Select**  Permits the grantee of this object privilege to access the data in a table, sequence, view, or snapshot.
   - **Insert**  Permits the grantee of this object privilege to insert data into a table or, in some cases, a view.
   - **Update**  Permits the grantee of this object privilege to update data into a table or view.
Delete  Permits the grantee of this object privilege to delete data from a table or view.

Alter  Permits the grantee of this object privilege to alter the definition of a table or sequence only. The alter privileges on all other database objects are considered system privileges.

Index  Permits the grantee of this object privilege to create an index on a table already defined.

References  Permits the grantee to create or alter a table in order to create a FOREIGN KEY constraint against data in the referenced table.

Execute  Permits the grantee to run a stored procedure or function.

4. How do you use AUDIT command for statement audits?

After deciding what to audit, you must begin auditing by setting the AUDIT_TRAIL init.ora parameter appropriately.

The general syntax for setting up auditing on statements or system privileges is as follows. The following code block shows an example of an audit statement:

AUDIT CREATE TABLE, ALTER TABLE, DROP TABLE
BY Krishnamoorthy
WHENEVER SUCCESSFUL;

5. What are the guidelines available for user account management?

➢ Use a standard password for user creation, such as 123abc or first1, and use password expire to force users to change this password to something else the first time they log into Oracle.

➢ Avoid OS authentication unless all your users will access Oracle while connected directly to the machine hosting your database.

➢ Be sure to always assign temporary tablespace and default tablespace to users.

➢ Give few users quota unlimited. Although it’s annoying to have users asking for more space, it’s even more annoying to reorganize tablespaces carelessly filled with database objects.

➢ Become familiar with the user-account management and other host machine limits that can be set via profiles. These new features take Oracle user-account management to new levels of security.

6. Write a note on dropping user accounts.

As users come and go, their access should be modified to reflect their departure. To drop a user from the database, you execute the drop user statement. If a user has created database objects, the
user cannot be dropped until the objects are dropped, as well. In order to drop the user and all related database objects in one fell swoop, Oracle provides the cascade option.

DROP USER Krishnamoorthy CASCADE;

Unit IV Part-B

1. Discuss about creating new database users
2. Explain (i) monitoring information about existing users (ii) altering and dropping existing users
3. Discuss the methods of Administering passwords using Profiles in Oracle
4. Explain the various system privileges
5. Discuss about Granting and revoking privileges
6. Elucidate the importance of (i) Administering profiles (ii) controlling resource use with profiles
7. A Database administrator is involved with maintaining the database of a company with thousands of users. Illustrate how the DBA can perform: Creating new database users, altering and dropping existing users.

8. Explain the concepts of various system privileges, granting and revoking user privileges