User Management and Database Security
Oracle database security management

- Controlling access to data (authorization)
- Authenticating users
- Ensuring data integrity
- Auditing user’s actions
- Managing enterprise security
Managing Users

- Creating new database users
- Altering and dropping existing database users
- Monitoring information about existing users
Users and Security

- DBA defines users who can access db
- Security domain defines the settings that apply to users
Database Schema

- Schema: named collection of objects like tables, views, procedures, etc.
- When a user is created a schema with same name is created
- Hence username and schema name used interchangeably
- Some of the objects a user can own
Checklist for Creating Users (Developers not end users)

- Choose a username and authentication mechanism.
- Identify tablespaces in which the user needs to store objects.
- Decide on quotas for each tablespace.
- Assign default tablespace and temporary tablespace.
- Create a user.
- Grant privileges and roles to the user.
- If no default tablespace is assigned to the user the System tablespace becomes the default for that user.
Creating a New User: Server Authentication

- Set the initial password:
- Expires at login forcing user to change password

```
CREATE USER anil IDENTIFIED BY panil
DEFAULT TABLESPACE data01
TEMPORARY TABLESPACE temp
QUOTA 15m ON data01
PASSWORD EXPIRE;
```
Creating a New User: Operating System Authentication

- Use OS_AUTHENT_PREFIX (in parameter file)
- Example: O/S User = user15

<table>
<thead>
<tr>
<th>OS_AUTHENT_PREFIX</th>
<th>Database User</th>
<th>Remote Login Possible</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OS_</strong></td>
<td>OS_USER15</td>
<td>No</td>
</tr>
<tr>
<td>empty string</td>
<td>USER15</td>
<td>No</td>
</tr>
<tr>
<td>&quot; &quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPS$ (default)</td>
<td>OPS$USER15</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Creating a New User: Operating System Authentication (contd.)

- E.g., An OS user tikekarr;
- Use IDENTIFIED EXTERNALLY clause with create user
- Also exists as a database user
- Oracle will not validate
- To use sql*plus say
  - Sqlplus /
Creating a New User: Guidelines

• Choose a standard password initially; use O/S authentication sparingly.
• Use the EXPIRE keyword to force users to reset their passwords.
• Always assign temporary tablespace.
• Restrict quotas to few users; use QUOTA UNLIMITED with caution.
• Educate users:
  – To connect
  – To change password
Controlling Account Lock and Password

ALTER USER anil
IDENTIFIED BY hisgrandpa
PASSWORD EXPIRE;

ALTER USER anil
IDENTIFIED BY hisgrandpa
ACCOUNT LOCK | UNLOCK;
Changing User Quota on Tablespace

• To get a user out of system (fired/resigned):
  – Use password expiration
  – Lock account
  – Alter password
  – Change profile
  – Export/import user schema elsewhere

ALTER USER anil
QUOTA 0 ON data01;
Dropping a User

- Use the CASCADE clause if the schema contains objects.

```sql
DROP USER anil;
```

```sql
DROP USER anil CASCADE;
```

- User currently connected cannot be dropped
Monitoring Users

**DBA_USERS**
- USERNAME
- USER_ID
- CREATED
- ACCOUNT_STATUS
- LOCK_DATE
- EXPIRY_DATE
- DEFAULT_TABLESPACE
- TEMPORARY_TABLESPACE

**DBA_TS_QUOTAS**
- USERNAME
- TABLESPACE_NAME
- BYTES
- MAX_BYTES
- BLOCKS
- MAX_BLOCKS
Monitoring Users (contd.)

- Select `tablespace_name`, `blocks`, `max_blocks`, `bytes`, `max_bytes` From `dba_ts_quota` Where `username` = ‘SCOTT’;
- -1 in `MAX_BLOCKS` or `MAX_BYTES` indicates unlimited quota
- Select `username`, `account_status`, `temporary_tablespace` From `dba_users`;
- Lists all users, their account status and temp. ts
Exercise III Managing Users
Profiles

- Are named sets of resource and password limits
- Are assigned to users by the CREATE/ALTER USER command
- Can be enabled or disabled
- Can relate to the DEFAULT profile
- Can limit system resources on session or call level
Managing Resources with Profiles

1. Create profiles.

2. Assign profiles to the user.

3. Enable resource limits.
Creating a Profile: Resource Limit

CREATE PROFILE developer_prof LIMIT
SESSIONS_PER_USER 2
CPU_PER_SESSION 10000
IDLE_TIME 60
CONNECT_TIME 480;
## Setting Resource Limits at Session Level

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU_PER_SESSION</td>
<td>Total CPU time measured in hundredths of seconds</td>
</tr>
<tr>
<td>SESSIONS_PER_USER</td>
<td>Number of concurrent sessions allowed for each username</td>
</tr>
<tr>
<td>CONNECT_TIME</td>
<td>Elapsed connect time measured in minutes</td>
</tr>
<tr>
<td>IDLE_TIME</td>
<td>Periods of inactive time measured in minutes</td>
</tr>
<tr>
<td>LOGICAL_READS_PER_SESSION</td>
<td>Number of data blocks (physical and logical reads)</td>
</tr>
<tr>
<td>PRIVATE_SGA</td>
<td>Private space in the SGA measured in bytes (for MTS only)</td>
</tr>
</tbody>
</table>
## Setting Resources at Call Level

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU_PER_CALL</td>
<td>CPU time per call in hundredths of seconds</td>
</tr>
<tr>
<td>LOGICAL_READS_PER_CALL</td>
<td>Number of data blocks</td>
</tr>
</tbody>
</table>
Resource Parameter

• Composite_limit
  – A composite limit is a sum of several of the resource parameters, measured in service units.
  – These resources are weighted by their importance.
  – Oracle takes into account four parameters to compute a weighted composite_limit:
    • Cpu_per_session
    • Connect_time
    • Logical_reads_per_session
    • Private_sga.
  – You can set a weight for these four parameter by using the alter resource cost statement.
Assigning Profiles to a User

CREATE USER user3 IDENTIFIED BY user3
DEFAULT TABLESPACE data01
TEMPORARY TABLESPACE temp
QUOTA unlimited ON data01
PROFILE developer_prof;

ALTER USER scott
PROFILE developer_prof;
Enabling Resource Limits

- Set the initialization parameter RESOURCE_LIMIT to TRUE
  - or
- Enforce the resource limits by enabling the parameter with the ALTER SYSTEM command

ALTER SYSTEM SET RESOURCE_LIMIT=TRUE;
Altering a Profile

```
ALTER PROFILE default LIMIT
SESSIONS_PER_USER 5
CPU_PER_CALL 3600
IDLE_TIME 30;
```
Dropping a Profile

DROP PROFILE developer_prof;

DROP PROFILE developer_prof CASCADE;
Viewing Resource Limits

**DBA_USERS**
- profile
- username

**DBA_PROFILES**
- profile
- resource_name
- resource_type (KERNEL)
- limit
Password Management

- Password history
- Account locking
- Setting up profiles
- Password expiration and aging
- Password verification
- User

© CMS INSTITUTE, 2004. All rights reserved. No part of this material may be reproduced, stored or emailed without the prior permission of Programme Director, CMS Institute
Enabling Password Management

- Set up password management by using profiles and assigning them to users.
- Lock, unlock, and expire accounts using the CREATE USER or ALTER USER command.
- Password limits are always enforced, even if RESOURCE_LIMIT for an instance is set to FALSE.
Creating a Profile: Password Settings

CREATE PROFILE grace_5 LIMIT
   FAILED_LOGIN_ATTEMPTS 3
   PASSWORD_LIFE_TIME 30
   PASSWORD_REUSE_TIME 30
   PASSWORD_VERIFY_FUNCTION verify_function
   PASSWORD_GRACE_TIME 5;
## Password Settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FAILED_LOGIN_ATTEMPTS</td>
<td>Number of failed login attempts before lockout of the account</td>
</tr>
<tr>
<td>PASSWORD_LOCK_TIME</td>
<td>Number of days for which the account remains locked upon password expiration</td>
</tr>
<tr>
<td>PASSWORD_LIFE_TIME</td>
<td>Lifetime of the password in days after which the password expires</td>
</tr>
<tr>
<td>PASSWORD_GRACE_TIME</td>
<td>Grace period in days for changing the password after the first successful login after the password has expired</td>
</tr>
</tbody>
</table>
## Password Settings (contd.)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PASSWORD_REUSE_TIME</td>
<td>Number of days before a password can be reused</td>
</tr>
<tr>
<td>PASSWORD_REUSE_MAX</td>
<td>Maximum number of times a password can be reused</td>
</tr>
<tr>
<td>PASSWORD_VERIFY_FUNCTION</td>
<td>PL/SQL function that makes a password complexity check before a password is assigned</td>
</tr>
</tbody>
</table>
User- Provided Password Function

- Function must be created in the SYS schema and must have the following specification:

```sql
function_name(
    userid_parameter IN VARCHAR2(30),
    password_parameter IN VARCHAR2(30),
    old_password_parameter IN VARCHAR2(30))
RETURN BOOLEAN
```
Password Verification Function

VERIFY_FUNCTION

- Minimum length is four characters
- Password should not be equal to username
- Password should have at least one alpha, one numeric, and one special character
- Password should differ from the previous password by at least three letters
Viewing Password Information

- **DBA_USERS**
  - profile
  - username
  - account_status
  - lock_date
  - expiry_date

- **DBA_PROFILES**
  - profile
  - resource_name
  - resource_type (PASSWORD)
  - limit
Resource Management Problem for Production Database

- Batch job’s are taking up most of the available resources, which is hurting other, more critical jobs that need to run at the same time.
- Excessive loads at peak times are causing critical processes to run for an unacceptably long period of time.
- You schedule large jobs and really can’t predict when they might be launched.
- Some users are using an excessive amount of CPU, causing you to kill their session abruptly.
- Some users are using a very high degree of parallelism in their operations, which is hurting the performance of the system as a whole.
- You want to prioritize jobs according to some scheme, but you can’t do so using operating system resources.
With Oracle's Database Resource Manager, a database administrator can:

• Guarantee certain users a minimum amount of processing resources regardless of the load on the system and the number of users

• Distribute available processing resources by allocating percentages of CPU time to different users and applications.
  – In a data warehouse, a higher percentage may be given to ROLAP (relational on-line analytical processing) applications than to batch jobs.

• Limit the degree of parallelism of any operation performed by members of a group of users

• Create an active session pool.
  – This pool consists of a specified maximum number of user sessions allowed to be concurrently active within a group of users.
With Oracle's Database Resource Manager, a database administrator can: (contd.)

- Allow automatic switching of users from one group to another group based on administrator-defined criteria.
  - If a member of a particular group of users creates a session that runs for longer than a specified amount of time, that session can be automatically switched to another group of users with different resource requirements.

- Prevent the execution of operations that are estimated to run for a longer time than a predefined limit.

- Create an undo pool.
  - This pool consists of the amount of undo space that can be consumed in by a group of users.

- Configure an instance to use a particular method of allocating resources.
  - You can dynamically change the method, for example, from a daytime setup to a nighttime setup, without having to shut down and restart the instance.
Database Resource Manager Overview

- Resources are allocated to users according to a resource plan specified by the database administrator.
- The following terms are used in specifying a resource plan:
  - A resource plan
    - specifies how the resources are to be distributed among various users (resource consumer groups).
  - Resource consumer groups
    - allow the administrator to group user sessions together by resource requirements. Resource consumer groups are different from user roles; one database user can have different sessions assigned to different resource consumer groups.
  - Resource allocation methods
    - determine what policy to use when allocating for any particular resource. Resource allocation methods are used by resource plans and resource consumer groups.
Database Resource Manager Overview (contd.)

- Resource plan directives
  - are a means of assigning consumer groups to particular plans and partitioning resources among consumer groups by specifying parameters for each resource allocation method.

- The Database Resource Manager also allows for creation of plans within plans, called subplans.

- Subplans allow further subdivision of resources among different users of an application.

- Levels provide a mechanism to specify distribution of unused resources among available users. Up to eight levels of resource allocation can be specified.
Using the Database Resource Manager

• A DBA can manage the Database Resource Manager through executing procedures in the Oracle-supplied DBMSRESOURCE_MANAGER package.

• Sequence of actions need to take to start using the Database Resource Manager
  – Create a pending area.
  – Create a consumer group.
  – Create a resource plan.
  – Create a plan directive.
  – Validate the pending area.
  – Submit the pending area.
Creating a Pending Area

- Before you can modify an old plan or create a new plan, you need to activate or create a pending area using the Database Resource Manager package.
- All resource plans created will be stored in the data dictionary.

```sql
SQL> execute dbms_resource_manager.create_pending_area;
PL/SQL procedure successfully completed.

SQL> execute dbms_resource_manager.create_pending_area;
PL/SQL procedure successfully completed.

SQL> execute dbms_resource_manager.clear_pending_area;
PL/SQL procedure successfully completed.

SQL> execute dbms_resource_manager.clear_pending_area;
PL/SQL procedure successfully completed.
```
Creating Consumer Groups

- Once the pending area is active, create the consumer groups to which users are allocated.
- You can assign users initially to one group, and you can later switch them to other groups if necessary.

```sql
SQL> execute dbms_resource_manager.create_pending_area;
PL/SQL procedure successfully completed.
SQL> execute dbms_resource_manager.create_consumer_group
    (consumer_group => 'local', comment => 'local councils');
PL/SQL procedure successfully completed.
SQL> execute dbms_resource_manager.create_consumer_group
    (consumer_group => 'regional', comment => 'regional');
PL/SQL procedure successfully completed.
SQL> execute dbms_resource_manager.create_consumer_group
    (consumer_group => 'national', comment => 'national office');
PL/SQL procedure successfully completed.
```
Checking what groups exist in your database

- Use `DBA_RSRC_CONSUMER_GROUPS` view for information relating to what groups currently exist in your database.

```
SQL> SELECT consumer_group, status FROM dba_rscr_consumer_groups;
```
Default groups in Oracle Database

- **Other_groups**
  - This isn’t really a group, because you can’t assign users to it.
  - When a resource plan is active, other_groups is the catchall term for all sessions that don’t belong to this active resource plan.

- **Default_consumer_groups**
  - If you don’t assign users to any group, they will, by default, become members of the default group.

- **Sys_group and low_group**
  - These are part of the default system_plan that exists in every database.
  - Oracle supplies three plans for each database
    - **SYSTEM_PLAN** : Plan to give system sessions priority.
    - **INTERNAL_QUIESCE** : Plan to internally quiesce system.
    - **INTERNAL_PLAN** : Plan to give system sessions priority
Validate & submit pending area

- Once you create the groups, you can then validate your pending area.
- Once the changes are accepted as being correct, you can submit the changes through the Database Resource Manager.

```
SQL> execute dbms_resource_manager.validate_pending_area;
PL/SQL procedure successfully completed.
SQL> execute dbms_resource_manager.submit_pending_area;
PL/SQL procedure successfully completed.
SQL> select consumer_group, status from dba_rsrc_consumer_groups;
```
Assigning users to consumer groups

- First grant the users privileges to switch their groups using `dbms_resource_manager_privs.grant_switch_consumer_group` procedure.
- Use `dbms_resource_manager_privs.set_initial_consumer_group` procedure to switch.

```sql
SQL> execute dbms_resource_manager_privs.grant_switch_consumer_group ('anil','local',TRUE);
PL/SQL procedure successfully completed.
SQL> execute dbms_resource_manager.set_initial_consumer_group('anil','local');
PL/SQL procedure successfully completed.
SQL> execute dbms_resource_manager_privs.grant_switch_consumer_group ('anil','local',TRUE);
PL/SQL procedure successfully completed.
SQL> execute dbms_resource_manager.set_initial_consumer_group('anil','local');
PL/SQL procedure successfully completed.
```

Verify Consumer Group Membership of Users

```sql
SQL> select username, initial_rsrc_consumer_group from dba_users;
```

© CMS INSTITUTE, 2004. All rights reserved. No part of this material may be reproduced, stored or emailed without the prior permission of Programme Director, CMS Institute
Create Resource Plans and Plan Directives

- Creating Resource Plans
  - Resource plans enable you to set limits on resource use by specifying limits on four variables: CPU, active session pool, degree of parallelism, and the order in which queued sessions will execute.
  - Currently, for all four parameters, only the default levels and methods provided by Oracle can be used.

SQL> execute dbms_resource_manager.create_pending_area;
PL/SQL procedure successfully completed.
SQL> execute dbms_resource_manager.create_plan (plan => 'membership_plan', comment => 'New Membership Recruitment');
PL/SQL procedure successfully completed.
Create Resource Plans and Plan Directives (contd.)

• Creating a Plan Directive
  – The plan directive assigns 70 percent of the available CPU at the first level to the local group and the rest, 30 percent, to the regional group.
  – It allocates 100 percent of the CPU at the second level to the national group.
  – In addition to the preceding three groups, you need to add a plan directive for the default other_groups for the Database Resource Manager to accept your plan directives.

• If you don’t include a resource directive for other_groups, Oracle won’t let you use your directives for the other groups if the plan directives is for a primary or top plan.
SQL> execute dbms_resource_manager.create_plan_directive
     (plan => 'membership_plan', GROUP_OR_SUBPLAN =>
     'local', COMMENT => 'LOCAL GROUP', CPU_P1 => 70);

SQL> execute dbms_resource_manager.create_plan_directive
     (plan => 'membership_plan', GROUP_OR_SUBPLAN =>
     'REGIONAL', COMMENT => 'regional group', CPU_P1 => 30);

SQL> execute dbms_resource_manager.create_plan_directive
     (plan => 'membership_plan', GROUP_OR_SUBPLAN =>
     'national', COMMENT => 'NATIONAL GROUP', CPU_P2 => 100);

SQL> execute dbms_resource_manager.create_plan_directive
     (plan => 'membership_plan', GROUP_OR_SUBPLAN =>
     'OTHER_GROUPS', comment => 'Default plan', CPU_P3 => 100);
Create Resource Plans and Plan Directives (contd.)

- You can now validate and submit your new top-level plan, membership_plan.
- Determining the status of the Resource Plans from dba_rsrrc_plan_directives

```sql
SQL> execute dbms_resource_manager.validate_pending_area;
PL/SQL procedure successfully completed.
SQL> execute dbms_resource_manager.submit_pending_area;
PL/SQL procedure successfully completed.
SQL> select plan, group_or_subplan, cpu_p1, cpu_p2, cpu_p3, status
    from dba_rsrc_plan_directives;
```
Enabling the Database Resource Manager

- Oracle will not automatically enforce the resource plans.
- Explicitly activate the Database Resource Manager, either by specifying the initialization parameter resource_manager_plan in the init.ora file or by using the alter system command.
- Query V$RSRC_CONSUMER_GROUP to see what resource usage among the consumers groups looks like.

```
SQL> alter system set resource_manager_plan=MEMBERSHIP_PLAN;
System altered.

SQL select * from v$rsrc_plan;

SQL> select name, active_sessions, cpu_wait_time, consumed_cpu_time, current_undo_consumption from v$rsrc_consumer_group;
```
Managing Privileges

• Two types of privileges:
  – SYSTEM: enables users to perform particular actions in the database
    • create, alter, drop, etc.
  – OBJECT: enables users to access and manipulate a specific object
    • select, update, insert, exec, etc.
System Privileges

- There are about 126 system privileges.
- The ANY-keyword in the privileges signifies that users have the privilege in every schema.
- The GRANT command adds a privilege to a user or a group of users.
- The REVOKE command deletes the privileges.
- Users with ANY privilege can access data dictionary tables.
### System Privileges: Examples

<table>
<thead>
<tr>
<th>Category</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX</td>
<td>CREATE ANY INDEX, ALTER ANY INDEX</td>
</tr>
<tr>
<td></td>
<td>DROP ANY INDEX</td>
</tr>
<tr>
<td>TABLE</td>
<td>CREATE TABLE (includes dropping privilege, create index)</td>
</tr>
<tr>
<td></td>
<td>CREATE ANY TABLE, ALTER ANY TABLE</td>
</tr>
<tr>
<td></td>
<td>DROP ANY TABLE (need this for truncating)</td>
</tr>
<tr>
<td></td>
<td>SELECT ANY TABLE, UPDATE ANY TABLE</td>
</tr>
<tr>
<td></td>
<td>DELETE ANY TABLE</td>
</tr>
<tr>
<td>SESSION</td>
<td>CREATE SESSION (need this to do anything)</td>
</tr>
<tr>
<td></td>
<td>ALTER SESSION</td>
</tr>
<tr>
<td></td>
<td>RESTRICTED SESSION (when db in restricted mode)</td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>CREATE TABLESPACE</td>
</tr>
<tr>
<td></td>
<td>ALTER TABLESPACE</td>
</tr>
<tr>
<td></td>
<td>DROP TABLESPACE</td>
</tr>
<tr>
<td></td>
<td>UNLIMITED TABLESPACE</td>
</tr>
</tbody>
</table>
Granting System Privileges

GRANT CREATE SESSION, CREATE TABLE TO user1;

GRANT CREATE SESSION TO scott
WITH ADMIN OPTION; (enables scott to grant the privilege
or role to other users or roles)
## SYSDBA and SYSOPER Privileges

<table>
<thead>
<tr>
<th>Category</th>
<th>Examples</th>
</tr>
</thead>
</table>
| SYSOPER    | STARTUP  
SHUTDOWN  
ALTER DATABASE OPEN | MOUNT  
ALTER DATABASE BACKUP CONTROLFILE  
ALTER TABLESPACE BEGIN/END BACKUP  
RECOVER DATABASE,  
ALTER DATABASE ARCHIVELOG  
RESTRICTED SESSION |
| SYSDBA     | SYSOPER privileges WITH ADMIN OPTION  
CREATE DATABASE  
RECOVER DATABASE UNTIL  
(any operation on db or objects in db)  
user SYSTEM not as powerful as SYS |
SYSDBA and SYSOPER Privileges (contd.)

- **User SYS:**
  - Owner of data dictionary, can make changes
  - Granted SYSOPER and SYSDBA roles
  - Can start and shutdown database

- **User STSTEM:**
  - Not granted SYSOPER and SYSDBA roles
  - Cannot start/shutdown database
  - Cannot modify data dictionary
  - Safer to be SYSTEM than SYS
Password File Authentication

- Create the password file and set the REMOTE_LOGIN_PASSWORDFILE parameter.
- Set REMOTE_LOGIN_PASSWORD_FILE=EXCLUSIVE.
- Grant SYSOPER and SYSDBA privileges to users.
- Query V$PWFILE_USERS to verify the password file members.
Displaying System Privileges

- Select * from dba_sys_privs;
- Select * from session_privs; (current session)
System Privilege Restrictions

- O7_DICTIONARY_ACCESSIBILITY = TRUE
  - Reverts to Oracle7 behavior
  - Removes the restrictions on system privileges with the ANY keyword
  - Defaults to TRUE
Revoking System Privileges

REVOKE CREATE TABLE FROM user1;
(can REVOKE privileges granted with GRANT command)

REVOKE CREATE SESSION FROM scott;
Revoking System Privileges Using WITH ADMIN OPTION

**GRANT**

- DBA
- USER 1
- SCOTT

**REVOKE** (doesn’t cascade)

- DBA
- USER 1
- SCOTT

© CMS INSTITUTE, 2004. All rights reserved. No part of this material may be reproduced, stored or emailed without the prior permission of Programme Director, CMS Institute
Revoking System Privileges Using WITH ADMIN OPTION (contd.)

RESULT

DBA  USER 1  SCOTT
## Object Privileges

<table>
<thead>
<tr>
<th>Object priv.</th>
<th>Table</th>
<th>View</th>
<th>Sequence</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTER</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>DELETE</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXECUTE</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>INDEX</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REFERENCES</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SELECT</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>UPDATE</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Granting Object Privileges

GRANT EXECUTE ON dbms_pipe TO public;

GRANT UPDATE(ename,sal) ON emp TO user1 WITH GRANT OPTION;

Column (field) level grants
Displaying Object Privileges

<table>
<thead>
<tr>
<th>DBA_TAB_PRIVS</th>
<th>DBA_COL_PRIVS</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRANTEE</td>
<td>GRANTEE</td>
</tr>
<tr>
<td>OWNER</td>
<td>OWNER</td>
</tr>
<tr>
<td>TABLE_NAME</td>
<td>TABLE_NAME</td>
</tr>
<tr>
<td>GRANTOR</td>
<td>COLUMN_NAME</td>
</tr>
<tr>
<td>PRIVILEGE</td>
<td>GRANTOR</td>
</tr>
<tr>
<td>GRANTABLE</td>
<td>PRIVILEGE</td>
</tr>
<tr>
<td></td>
<td>GRANTABLE</td>
</tr>
</tbody>
</table>

Object privileges       Col specific privileges
Revoking Object Privileges

- Select * from dba_tab_privs where grantee = ‘SCOTT’;
- Select * from dba_col_privs;

REVOKE execute ON dbms_pipe FROM scott;
Revoking Object Privileges
Using WITH GRANT OPTION

GRANT

SCOTT

USER 1

USER 2

REVOKE

SCOTT

USER 1

USER 2

© CMS INSTITUTE, 2004. All rights reserved. No part of this material may be reproduced, stored or emailed without the prior permission of Programme Director, CMS Institute
Revoking Object Privileges
Using WITH GRANT OPTION  (contd.)

RESULT

SCOTT  USER 1  USER 2

Summary: revoking object privileges will cascade
Roles

• Role: named groups of related privileges
  – Granted/revoked with same commands as for privileges
  – Maybe granted to user or role (except itself)
  – Can consist of object and system privileges
  – May be enabled/disabled
  – Can require password to enable
  – Not owned by anyone
Roles (contd.)

**Users**

**Roles**

**Privileges**

- CREATE TABLE
- SELECT ON EMP
- INSERT ON EMP
- UPDATE ON EMP
Benefits of Roles

- Reduced granting of privileges
- Dynamic privilege management
- Selective availability of privileges
- Granted through the OS
- No cascading revokes
- Improved performance
Creating Roles

CREATE ROLE sales_clerk;

CREATE ROLE hr_clerk IDENTIFIED BY bonus;

CREATE ROLE hr_manager IDENTIFIED EXTERNALLY;
# Using Predefined Roles

<table>
<thead>
<tr>
<th>Role Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONNECT</td>
<td>These two roles are provided for backward compatibility.</td>
</tr>
<tr>
<td>RESOURCE</td>
<td></td>
</tr>
<tr>
<td>DBA</td>
<td>All system privileges WITH ADMIN OPTION</td>
</tr>
<tr>
<td>EXP_FULL_DATABASE</td>
<td>Privileges to export the DB</td>
</tr>
<tr>
<td>IMP_FULL_DATABASE</td>
<td>Privileges to import the DB</td>
</tr>
<tr>
<td>DELETE_CATALOG_ROLE</td>
<td>DELETE privileges on DD tables</td>
</tr>
<tr>
<td>EXECUTE_CATALOG_ROLE</td>
<td>EXECUTE privilege on DD packages</td>
</tr>
<tr>
<td>SELECT_CATALOG_ROLE</td>
<td>SELECT privilege on DD tables</td>
</tr>
</tbody>
</table>
Modifying Roles

ALTER ROLE sales_clerk
IDENTIFIED BY commission;

ALTER ROLE hr_clerk
IDENTIFIED EXTERNALLY;

ALTER ROLE hr_manager
NOT IDENTIFIED;
Assigning Roles

GRANT sales_clerk TO scott;

GRANT hr_clerk, TO hr_manager;

GRANT hr_manager TO scott WITH ADMIN OPTION;
Assigning Privileges to Roles

GRANT create table, create any index TO hr_clerk;

GRANT create_session TO hr_manager;
Enabling and Disabling Roles

- Disable a role to temporarily revoke the role from a user.
- Enable a role to temporarily grant it.
- The SET ROLE command enables and disables roles.
- Default roles are enabled for a user at login.
- A password may be required to enable a role.
Enabling and Disabling Roles: Examples

SET ROLE sales_clerk IDENTIFIED BY commission;

Enable: this is how users would activate their role

SET ROLE ALL EXCEPT sales_clerk;

SET ROLE NONE;

Disable all roles for current session
Removing Roles from Users

```sql
REVOKE hr_manager FROM PUBLIC;
REVOKE hr_manager FROM PUBLIC;
REVOKE sales_clerk FROM scott;
REVOKE hr_manager FROM PUBLIC;
```
Removing Roles

DROP ROLE hr_manager;
Guidelines for Creating Roles

- **Users**
  - HR_CLERK
  - HR_MANAGER
  - PAY_CLERK

- **Application roles**
  - BENEFITS
  - PAYROLL

- **Application privileges**
  - Benefits privileges
  - Payroll privileges

© CMS INSTITUTE, 2004. All rights reserved. No part of this material may be reproduced, stored or emailed without the prior permission of Programme Director, CMS Institute
Guidelines for using Passwords and Default Roles

Password protected non-default

PAY_CLERK

Insert, update, delete and select privileges

Default role

PAY_CLERK_RO

Select privileges
PUBLIC

• The PUBLIC User Group and Roles
  – To give a certain privilege or role to all the users in the database, simple grant this privilege/role to the user group PUBLIC, which exists in every database by default.

• Using Secure Application Roles
  – Secure application roles in Oracle9i are roles that are implemented through a package.
# Displaying Role Information

<table>
<thead>
<tr>
<th>Role View</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBA_ROLES</td>
<td>All roles which exist in the database</td>
</tr>
<tr>
<td>DBA_ROLE_PRIVS</td>
<td>Roles granted to users and roles</td>
</tr>
<tr>
<td>ROLE_ROLE_PRIVS</td>
<td>Roles which are granted to roles</td>
</tr>
<tr>
<td>DBA_SYS_PRIVS</td>
<td>System privileges granted to users and roles</td>
</tr>
<tr>
<td>ROLE_SYS_PRIVS</td>
<td>System privileges granted to roles</td>
</tr>
<tr>
<td>ROLE_TAB_PRIVS</td>
<td>Table privileges granted to roles</td>
</tr>
<tr>
<td>SESSION_ROLES</td>
<td>Roles which the user currently has enabled.</td>
</tr>
</tbody>
</table>

```sql
Select role, password_required from dba_roles;
```
Exercise IV Managing user roles & privileges
Fine-grained Data Security

- Oracle9i database provides a lower level security of data using fine-grained data security techniques.
- You can allow all users to access a central table such as payroll table, but transparent to the users you can institute security policies that limit access of an individual user to only those rows in a table.
- Oracle uses two main concepts to enforce fine-grained security within database:
  - An application context.
  - A fine-grained access control policy.
- Oracle uses the term Virtual Private Database to refer to the implementation of the fine-grained access control policies through application contexts.
Auditing Categories

• Auditing privileged operations
  – Always audited
  – Startup, shutdown, and SYSDBA or SYSOPER connections

• Database auditing
  – Enabled by DBA
  – Cannot record column values

• Value-based or application auditing
  – Implemented through code
  – Can record column values
  – Used to track changes to tables
System-level Triggers

- **Database start-up triggers:**
  - These triggers are used to execute code that you want to execute immediately after database start-up.

- **Logon triggers:**
  - Provide you with information regarding the logon times of a user, along with details about the user’s session.

- **Logoff triggers:**
  - Similar to the logon triggers, but they execute right before the user’s session logs off.

- **DDL triggers:**
  - To capture all database object changes with these triggers.

- **Server error triggers:**
  - Capture all major PL/SQL code errors into a special table.
Value-Based Auditing: An Example

CREATE TRIGGER scott.auditemployee
AFTER INSERT OR DELETE OR UPDATE
ON scott.emp
FOR EACH ROW
BEGIN
INSERT INTO scott.audit_employee
VALUES ( :OLD.empno, :OLD.name,…,
:NEW.empno, :NEW.name,…,
USER, SYSDATE);
END;
Database Auditing

Enable DB auditing

Parameter file

Specify audit options

Server process

Generate audit trail

Audit options

Audit trail

Other tables

Database

Review audit information

Execute command

User

DBA

Specify audit options

Execute command

Server process

Generate audit trail

Audit options

Audit trail

Other tables

Database

Review audit information

Enable DB auditing
Enabling Database Auditing

AUDIT_TRAIL Parameter
- file
  - OS audit trail
  - DB
  - Instance
    - DBA
    - AUDIT_TABLE

AUDS$ table
- DB
- OS
- OS
- NONE

No trail
Enabling Auditing Options

- **Statement auditing**
  
  ```sql
  AUDIT user;
  ```

- **Privilege auditing**
  
  ```sql
  AUDIT select any table
  BY scott BY ACCESS;
  ```

- **Schema object auditing**
  
  ```sql
  AUDIT LOCK ON scott.emp
  BY ACCESS WHENEVER SUCCESSFUL;
  ```
# Auditing Schema Objects

<table>
<thead>
<tr>
<th>Option</th>
<th>Table</th>
<th>View</th>
<th>Sequence</th>
<th>Stored Program</th>
<th>Snapshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTER</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>AUDIT</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>COMMENT</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DELETE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>EXECUTE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>GRANT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>INDEX</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>INSERT</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>LOCK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>READ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>RENAME</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>SELECT</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>UPDATE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
## Viewing Auditing Options

<table>
<thead>
<tr>
<th>Data Dictionary View</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL_DEF_AUDIT_OPTS</td>
<td>Default audit options</td>
</tr>
<tr>
<td>DBA_STMT_AUDIT_OPTS</td>
<td>Statement auditing options</td>
</tr>
<tr>
<td>DBA_PRIV_AUDIT_OPTS</td>
<td>Privilege auditing options</td>
</tr>
<tr>
<td>DBA_OBJ_AUDIT_OPTS</td>
<td>Schema object auditing options</td>
</tr>
</tbody>
</table>
Disabling Auditing Options

NOAUDIT user WHENEVER SUCCESSFUL;

NOAUDIT create table BY scott;

NOAUDIT LOCK ON emp;
The Audit Trail

- Stores the records generated by statement, privilege, and object auditing
- The audit records are stored in the SYS.AUD$ data dictionary table or in the OS audit trail
- Each record in the audit trail includes:
  - The user who executed the statement
  - The command issued (action code)
  - Any system or object privilege used
  - The objects referenced in the statement
  - The date and time the statement was issued
Oracle Default Auditing

- When you don’t specify any type of logging, by default Oracle will log three types of database actions under all circumstances.

- The auditing actions and the audit records are written to the default $ORACLE_HOME/rdbms/audit directory.
  - Connections as SYSOPER or SYSDBA.
  - Database start-up
  - Database shutdown
# Viewing Auditing Results

<table>
<thead>
<tr>
<th>Audit Trail View</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBA_AUDIT_TRAIL</td>
<td>All audit trail entries</td>
</tr>
<tr>
<td>DBA_AUDIT_EXISTS</td>
<td>Records for AUDIT EXISTS/NOT EXISTS</td>
</tr>
<tr>
<td>DBA_AUDIT_OBJECT</td>
<td>Records concerning schema objects</td>
</tr>
<tr>
<td>DBA_AUDIT_SESSION</td>
<td>All connect and disconnect entries</td>
</tr>
<tr>
<td>DBA_AUDIT_STATEMENT</td>
<td>Statement auditing records</td>
</tr>
</tbody>
</table>
Auditing Guidelines

• Focus auditing
  – Object auditing, where possible
  – Only specific users
  – By session
  – Successful or unsuccessful

• Maintain the audit trail
  – Monitor the growth of the audit trail
  – Protect the audit trail from unauthorized access
  – Cleaning OS audit files
The Password File

- Use orapwd to create the password file.
- The remote_login_passwordfile initialization parameter.
- None
  - No password file is used.
  - This is the default, and it permits only operating system-authenticated users to perform DBA task.
- Shared
  - Creates a shared password file with a single user: SYS.
  - Any user who wants to perform privileged tasks has to log in as SYS.
- Exclusive
  - Uses a password file.
  - Any user can be granted the SYSDBA and SYSOPER privileges, and when the user SYS does so, the user is automatically added to the password file.
Encrypted Passwords

- By default, Oracle user passwords aren’t encrypted, which leaves them vulnerable to unauthorized usage.
  - `ora_encrypt_login=true` (client)
  - `dblink_encrypt_login=true` (server)
- Oracle will always encrypt a password when it’s sending it across a network.
Authentication Methods

- External Authentication
- Proxy Authentication
- Centralized User Authorization using LDAP
Database Security Do’s Don’ts

• User Accounts default lock except SYS and SYSTEM.
• Passwords, don’t hard-code
• Operating System Authentication
• Audit your Database
• Grant Privileges Appropriately.
• Set appropriate Permissions.
• Safeguard the Network and the Listener.
• Keep Up-to-Date for latest news about new security vulnerabilities and the patches to overcome them.
• Use Oracle’s Advanced Security Feature.
• Take Care of Application Security.
Exercise V Auditing the database Usage