DBA Commands

Users, Roles and Privileges

An ORACLE role is a set of privileges (or the type of access that each user needs depending on his or her status and responsibilities)

Database system privileges let you execute specific sets of commands. The CREATE TABLE privilege, for example, lets you create tables

Database object privileges give you the ability to perform some operation on various objects. The DELETE privilege, for example, lets you delete rows from tables and views.

Creating a User

SYSTEM and SYS users are created by default with passwords MANAGER and CHANGE_ON_INSTALL

create user user identified (by password | externally);

create user Dora identified by avocado;

To change a password, use the alter user command:

alter user Dora identified by psyche;

Password Management

Passwords can expire, and accounts may be locked due to repeated failed attempts to connect. When you change your password, a password history may be maintained in order to prevent reuse of previous passwords.

The expiration characteristics of your account’s password are determined by the profile assigned to your account. Profiles, which are created by the create profile command, are managed by the dba.

- The “lifetime” of your password, which determines how frequently you must change it
- The grace period following your password’s “expiration date” during which you can change the password
- The number of consecutive failed connect attempts allowed before the account is automatically “locked”
- The number of days the account will remain locked
• The number of days that must pass before you can reuse a password
• The number of password changes that must pass before you can reuse a password

Additional password management features allow the minimum length of passwords to be enforced.

Another way to change password: use **password** command

```
password
changing password for dora
old password:
new password:
Retype new password:
```

When the password has been successfully changed, you will receive the feedback:

Password changed

**Three Standard Roles**
CONNECT, RESOURCE and DBA

**The CONNECT Role**

Privileges to: CREATE SESSION, ALTER SESSION, CREATE CLUSTER, CREATE DATABASE LINK, CREATE SEQUENCE, CREATE SYNONYM, CREATE TABLE, and CREATE VIEW.

Users do not have the ability to create tables or clusters unless you grant them a quota on a tablespace.

You can create your own role and grant privileges:

```
Create role APPLICATION_USER;
Grant CREATE SESSION to APPLICATION_USER;
Grant APPLICATION_USER to username;
```

**The RESOURCE Role**

The RESOURCE role has the following system privileges:
CREATE CLUSTER, CREATE INDEX, CREATE PROCEDURE, CREATE SEQUENCE, CREATE TABLE, CREATE TRIGGER, CREATE TYPE.

Users who have the RESOURCE role do not have the ability to create tables, indexes and clusters unless you first grant them a space quota in a tablespace.
Grant RESOURCE role to developers who will be creating PL/SQL objects such as procedures and triggers.

The DBA Role

The DBA role has all system privileges – including unlimited space quotas – and the ability to grant all privileges to other users.

SYSTEM is for use by a DBA user.

In ORACLE, the DBA is granted the EXP_FULL_DATABASE and IMP_FULL_DATABASE roles, which in turn have privileges necessary for exporting and importing the full Oracle database.

Format for the grant command

grant {system privilege | role} 
    [, {system privilege | role}…] 
    to {user | role} [ , {user | role} ]… 
    [with admin option]

Revoking Privileges

Privileges granted can be taken away.

revoke {system privilege | role} 
   [ , {system privilege | role}…] 
   from {user | role} [ , {user | role} ]…

An individual with the DBA role can revoke CONNECT, RESOURCE, DBA, or any other privilege or role from anyone, including another DBA.

This, of course, is dangerous!!

To remove a user and all the resources owned by that user, use the drop user command like this:

drop user user [cascade];

The cascade option drops the user along with all the objects owned by the user, including referential integrity constraints.

What Users Can Grant

A user can grant privileges on any object he or she owns. The dba can grant any system privileges.
Let Dora own the COMFORT table and is a dba

create user Judy identified by sarah;
user created.

grant CONNECT to Judy;
role granted.

create user Bob identified by carolyn;
user created.

grant CONNECT, RESOURCE to Bob;
Role granted.

To give others access to your tables, use a second form of the grant command:

grant object privilege [(column [, column])] on object to {user | role} [with grant option];

The privileges a user can grant include these:

- Tables and views (user’s own only):
  - INSERT
  - UPDATE (all or specific columns)
  - DELETE
- On tables only
  - ALTER (Table – all or specific columns – or sequence)
  - REFERENCES
  - INDEX (columns in a table)
  - ALL (of the above)
- On procedures, functions, packages, abstract datatypes, and libraries:
  - EXECUTE
- On tables, views, sequences, and snapshots:
  - SELECT
- On directories (for BFILE LOB datatypes):
  - READ

Dora gives Bob SELECT access to the COMFORT table:

grant select on COMFORT to Bob;
grant succeeded.

The **with grant option** clause of the **grant** command allows the recipient of that grant to pass along the privileges he or she has received to other users.

**Moving to Another User with connect**

close Bob/carolyn

close.

```sql
select * from Dora.COMFORT;
```

<table>
<thead>
<tr>
<th>City</th>
<th>SampleDat</th>
<th>Noon</th>
<th>Midnight</th>
<th>Precipitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>San Francisco</td>
<td>21-MAR-93</td>
<td>62.5</td>
<td>42.3</td>
<td>.5</td>
</tr>
<tr>
<td>San Francisco</td>
<td>22-JUN-93</td>
<td>51.1</td>
<td>71.9</td>
<td>.1</td>
</tr>
<tr>
<td>San Francisco</td>
<td>23-SEP-93</td>
<td>61.5</td>
<td></td>
<td>.1</td>
</tr>
<tr>
<td>San Francisco</td>
<td>22-DEC-93</td>
<td>52.6</td>
<td>39.8</td>
<td>2.3</td>
</tr>
<tr>
<td>Keene</td>
<td>21-MAR-93</td>
<td>39.9</td>
<td>-1.2</td>
<td>4.4</td>
</tr>
<tr>
<td>Keene</td>
<td>22-JUN-93</td>
<td>85.1</td>
<td>66.7</td>
<td>1.3</td>
</tr>
<tr>
<td>Keene</td>
<td>23-SEP-93</td>
<td>99.8</td>
<td>82.6</td>
<td></td>
</tr>
<tr>
<td>Keene</td>
<td>22-DEC-93</td>
<td>-7.2</td>
<td>-1.2</td>
<td>3.9</td>
</tr>
</tbody>
</table>

create view COMFORT as select * from Dora.COMFORT;

view created.

selecting from this view will produce exactly the same results as selecting from Dora.COMFORT:

```sql
select * from COMFORT;
```

<table>
<thead>
<tr>
<th>City</th>
<th>SampleDat</th>
<th>Noon</th>
<th>Midnight</th>
<th>Precipitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>San Francisco</td>
<td>21-MAR-93</td>
<td>62.5</td>
<td>42.3</td>
<td>.5</td>
</tr>
<tr>
<td>San Francisco</td>
<td>22-JUN-93</td>
<td>51.1</td>
<td>71.9</td>
<td>.1</td>
</tr>
<tr>
<td>San Francisco</td>
<td>23-SEP-93</td>
<td>61.5</td>
<td></td>
<td>.1</td>
</tr>
<tr>
<td>San Francisco</td>
<td>22-DEC-93</td>
<td>52.6</td>
<td>39.8</td>
<td>2.3</td>
</tr>
<tr>
<td>Keene</td>
<td>21-MAR-93</td>
<td>39.9</td>
<td>-1.2</td>
<td>4.4</td>
</tr>
<tr>
<td>Keene</td>
<td>22-JUN-93</td>
<td>85.1</td>
<td>66.7</td>
<td>1.3</td>
</tr>
<tr>
<td>Keene</td>
<td>23-SEP-93</td>
<td>99.8</td>
<td>82.6</td>
<td></td>
</tr>
<tr>
<td>Keene</td>
<td>22-DEC-93</td>
<td>-7.2</td>
<td>-1.2</td>
<td>3.9</td>
</tr>
</tbody>
</table>
Now Dora returns to her own username and creates a view that selects only a part of the COMFORT table:

```
create view SOME COMFORT as
  select * from COMFORT
  where City = 'Keene';
view created.
```

```
grant select, update on SOME COMFORT to Bob;
grant succeeded.
```

```
revoke all on COMFORT from Bob;
Revoke succeeded.
```

Dora then reconnects to Bob’s username to test the effects of this change:

```
connect Bob/carolyn
connected.
```

```
select * from COMFORT;
Error at line 1: ORA-0942: table or view does not exist.
```

```
select * from Dora.SOME COMFORT;
```

<table>
<thead>
<tr>
<th>City</th>
<th>SampleDat</th>
<th>Noon</th>
<th>Midnight</th>
<th>Precipitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keene</td>
<td>21-MAR-93</td>
<td>39.9</td>
<td>-1.2</td>
<td>4.4</td>
</tr>
<tr>
<td>Keene</td>
<td>22-JUN-93</td>
<td>85.1</td>
<td>66.7</td>
<td>1.3</td>
</tr>
<tr>
<td>Keene</td>
<td>23-SEP-93</td>
<td>99.8</td>
<td>82.6</td>
<td></td>
</tr>
<tr>
<td>Keene</td>
<td>22-DEC-93</td>
<td>-7.2</td>
<td>-1.2</td>
<td>3.9</td>
</tr>
</tbody>
</table>

Now the view LITTLECOMFORT is created under Bob’s username, on top of the view SOME COMFORT:

```
create view LITTLE COMFORT as select * from Dora.SOME COMFORT;
```
and the row for September 23, 1993, is updated:

update LITTLECOMFORT set Noon = 88
  where SampleDate = To_Date('23-SEP-1993','DD-MON-YYYY');

1 row updated.

select * from LITTLECOMFORT;

<table>
<thead>
<tr>
<th>City</th>
<th>SampleDate</th>
<th>Noon</th>
<th>Midnight</th>
<th>Precipitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keene</td>
<td>21-MAR-93</td>
<td>39.9</td>
<td>-1.2</td>
<td>4.4</td>
</tr>
<tr>
<td>Keene</td>
<td>22-JUN-93</td>
<td>85.1</td>
<td>66.7</td>
<td>1.3</td>
</tr>
<tr>
<td>Keene</td>
<td>23-SEP-93</td>
<td>88</td>
<td>82.6</td>
<td></td>
</tr>
<tr>
<td>Keene</td>
<td>22-DEC-93</td>
<td>-7.2</td>
<td>-1.2</td>
<td>3.9</td>
</tr>
</tbody>
</table>

Note: You need to grant users SELECT access to any table in which they can update or delete records.

create synonym

An alternative method to creating a view that includes an entire table or view from another user is to create a synonym:

create synonym LITTLECOMFORT for Dora.SOMECOMFORT;

Using Ungranted Privileges

Let’s say an attempt is made to delete the row you just updated:

delete from LITTLECOMFORT where SampleDate = ‘23-SEP-93’;

Error at line 1: ORA-1031: insufficient privileges

Passing on Privileges

grant insert on Dora.SOMECOMFORT to Judy;

Error at line 1: ORA-01031: insufficient privileges
grant select on Dora.SOMECONFORT to Judy;

Error at line 1: ORA-01031: insufficient privileges

grant select on LITTLECONFORT to Judy;

Error at line 1: ORA-01720: grant option does not exist for ‘Dora.SOMECONFORT’

Since the LITTLECONFORT view relies on one of Dora’s views, and Bob was not granted SELECT with grant option on that view, Bob’s grant fails.

create table NOCONFORT as
select * from LITTLECONFORT;

Table created.

grant select on NOCONFORT to Judy;

Grant succeeded.

To test this grant, Judy’s username is connected, like this:

connect Judy/sarah
connected

select * from Bob.NOCOMFORT;

<table>
<thead>
<tr>
<th>City</th>
<th>SampleDat</th>
<th>Noon</th>
<th>Midnight</th>
<th>Precipitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keene</td>
<td>21-MAR-93</td>
<td>39.9</td>
<td>-1.2</td>
<td>4.4</td>
</tr>
<tr>
<td>Keene</td>
<td>22-JUN-93</td>
<td>85.1</td>
<td>66.7</td>
<td>1.3</td>
</tr>
<tr>
<td>Keene</td>
<td>23-SEP-93</td>
<td>88</td>
<td>82.6</td>
<td></td>
</tr>
<tr>
<td>Keene</td>
<td>22-DEC-93</td>
<td>-7.2</td>
<td>-1.2</td>
<td>3.9</td>
</tr>
</tbody>
</table>

If Dora wishes Bob to be able to pass on his privileges to others, she can add another clause to the grant statement:

grant select, update on SOMECONFORT to Bob with grant option;

Grant successful.
Creating a Role

In order to create a role, you need to have CREATE ROLE system privilege

create role role_name
[not identified | identified [by password | externally]];

When a role is first created, it has no privileges associated with it.

create role CLERK;
create role MANAGER;

Granting Privileges to a Role

grant select on COMFORT to CLERK;

The ability to log into the database is given via the CREATE SESSION system privilege.

grant CREATE SESSION to CLERK;
grant CREATE SESSION, CREATE DATABASE LINK to MANAGER;

Granting a Role to Another Role

grant CLERK to MANAGER;

grant CLERK to MANAGER with admin option;

If the with admin option clause is used, then the grantee has the authority to grant the role to other users or roles. The grantee can also alter or drop the role.

Granting a Role to Users

Note: Privileges that are granted to users via roles cannot be used as the basis for views, procedures, functions, packages, or foreign keys. When creating these types of database objects, you must rely on direct grants of the necessary privileges.

grant CLERK to Bob;

grant MANAGER to Dora with admin option;

Dora now has the authority to grant the MANAGER role to other users or roles, or to alter or drop the role.
**Adding a Password to a Role**

To enable security for a role, use the **identified** keyword in the *alter role* command.

```sql
alter role MANAGER identified by cygnusxi;
```

If that role is set up as a default role for the user, then no password will be required for that role when the user logs in.

```sql
alter role MANAGER identified externally;
```

**Removing a Password from a Role**

```sql
alter user username
default role [{role1, role2}
[all|all except role1, role2][NONE]];
```

```sql
alter user Bob
default role CLERK;
```

To enable a nondefault role, use the *set role* command, as shown in this example:

```sql
set role CLERK;
```

```sql
set role all;
set role all except CLERK;
```

If a role has a password associated with it, then the password must be specified via an **identified by** clause:

```sql
set role MANAGER identified by cygnusxi;
```

To disable a role in your session, use the *set role none* command

```sql
set role none;
```

**Revoking Privileges from a Role**

```sql
revoke SELECT on COMFORT from CLERK;
```
Dropping a Role

drop role MANAGER;
drop role CLERK;

Granting update to Specific Columns

grant update (Noon, Midnight) on COMFORT to Judy;

Revoking Privileges

If object privileges can be granted, they can also be taken away.

revoke object privilege [, object privilege…] on object from {user | role} [, {user | role}] [cascade constraints];

revoke all removes any of the privileges listed previously,

The with grant option is revoked along with the privilege to which it was attached.

If a user defines referential integrity constraints on the object, ORACLE drops these constraints if you revoke privileges on the object using the cascade constraints option.

Security by User

Access to tables can be granted specifically, table by table, and view by view, to each user.
```
select * from WORKER;

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Lodging</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adah Talbot</td>
<td>23</td>
<td>Papa king</td>
</tr>
<tr>
<td>Andrew Dye</td>
<td>29</td>
<td>Rose Hill</td>
</tr>
<tr>
<td>Bart Sarjeant</td>
<td>22</td>
<td>Cranmer</td>
</tr>
<tr>
<td>Dick Jones</td>
<td>18</td>
<td>Rose Hill</td>
</tr>
<tr>
<td>Donald Rollo</td>
<td>16</td>
<td>Matts</td>
</tr>
<tr>
<td>Elbert Talbot</td>
<td>43</td>
<td>Weitbrocht</td>
</tr>
<tr>
<td>George Oscar</td>
<td>41</td>
<td>Rose Hill</td>
</tr>
<tr>
<td>Gerhardt Kentgen</td>
<td>55</td>
<td>Papa King</td>
</tr>
<tr>
<td>Helen Brandt</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Jed Hopkins</td>
<td>33</td>
<td>Matts</td>
</tr>
<tr>
<td>John Pearson</td>
<td>27</td>
<td>Rose Hill</td>
</tr>
<tr>
<td>Kay and Palmer Walbom</td>
<td></td>
<td>Rose Hill</td>
</tr>
<tr>
<td>Pat Lavay</td>
<td>21</td>
<td>Rose Hill</td>
</tr>
<tr>
<td>Peter lawson</td>
<td>25</td>
<td>Cranmer</td>
</tr>
<tr>
<td>Richard Koch and brothers</td>
<td></td>
<td>Weitbrocht</td>
</tr>
<tr>
<td>Roland Brandt</td>
<td>35</td>
<td>Matts</td>
</tr>
<tr>
<td>Victoria Lynn</td>
<td>32</td>
<td>Mullers</td>
</tr>
<tr>
<td>Wilfred Lowell</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>William Swing</td>
<td>15</td>
<td>Cranmer</td>
</tr>
</tbody>
</table>
```

create view YOURAGE as
select * from WORKER
    where SUBSTR(name, 1, INSTR(Name, ' ')-1) = User;
view created.

```
select * from YOURAGE;

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Lodging</th>
</tr>
</thead>
<tbody>
<tr>
<td>George Oscar</td>
<td>41</td>
<td>Rose Hill</td>
</tr>
</tbody>
</table>
```

Now George **grants select** on this view to Bart Sarjeant:

grant SELECT on YOURAGE to Bart;

He then **connects** to Bart to check the effect:

connect Bart/stjohn
connected.
select * from George.YOURAGE;

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Lodging</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bart Sarjeant</td>
<td>22</td>
<td>Cranmer</td>
</tr>
</tbody>
</table>

**Granting Access to the Public**

grant select on YOURAGE to public;

create public synonym YOURAGE for George.YOURAGE;

from this point forward, anyone can access YOURAGE without prefixing it with George.

**Granting Limited Resources**

alter user Bob
quota 100M on USERS;

A user’s space quota may be set when the user is created, via the `create user` command.

If there is no limit on the user’s space quota, then you can grant that user the UNLIMITED TABLESPACE system privilege.