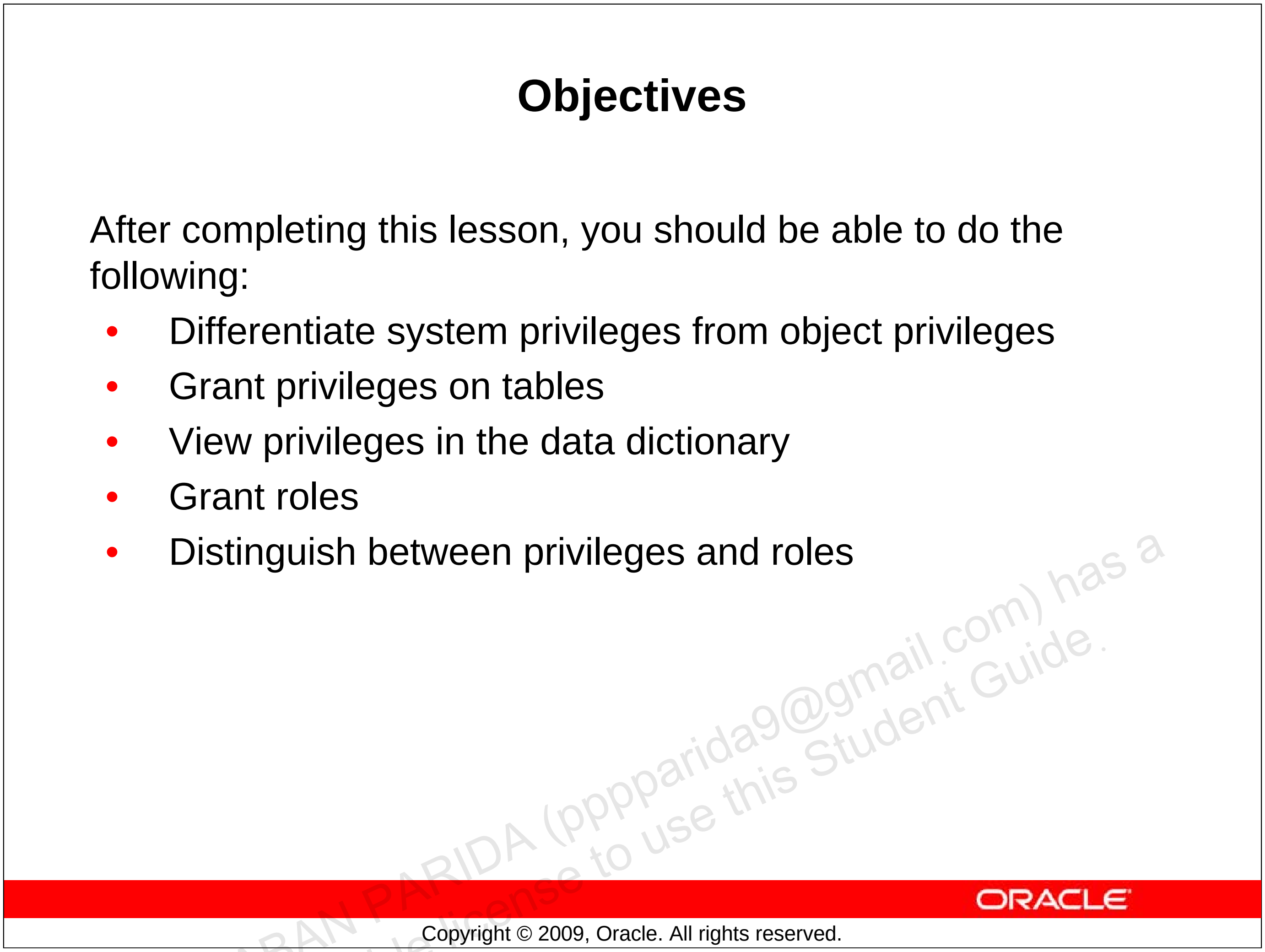
Controlling User Access



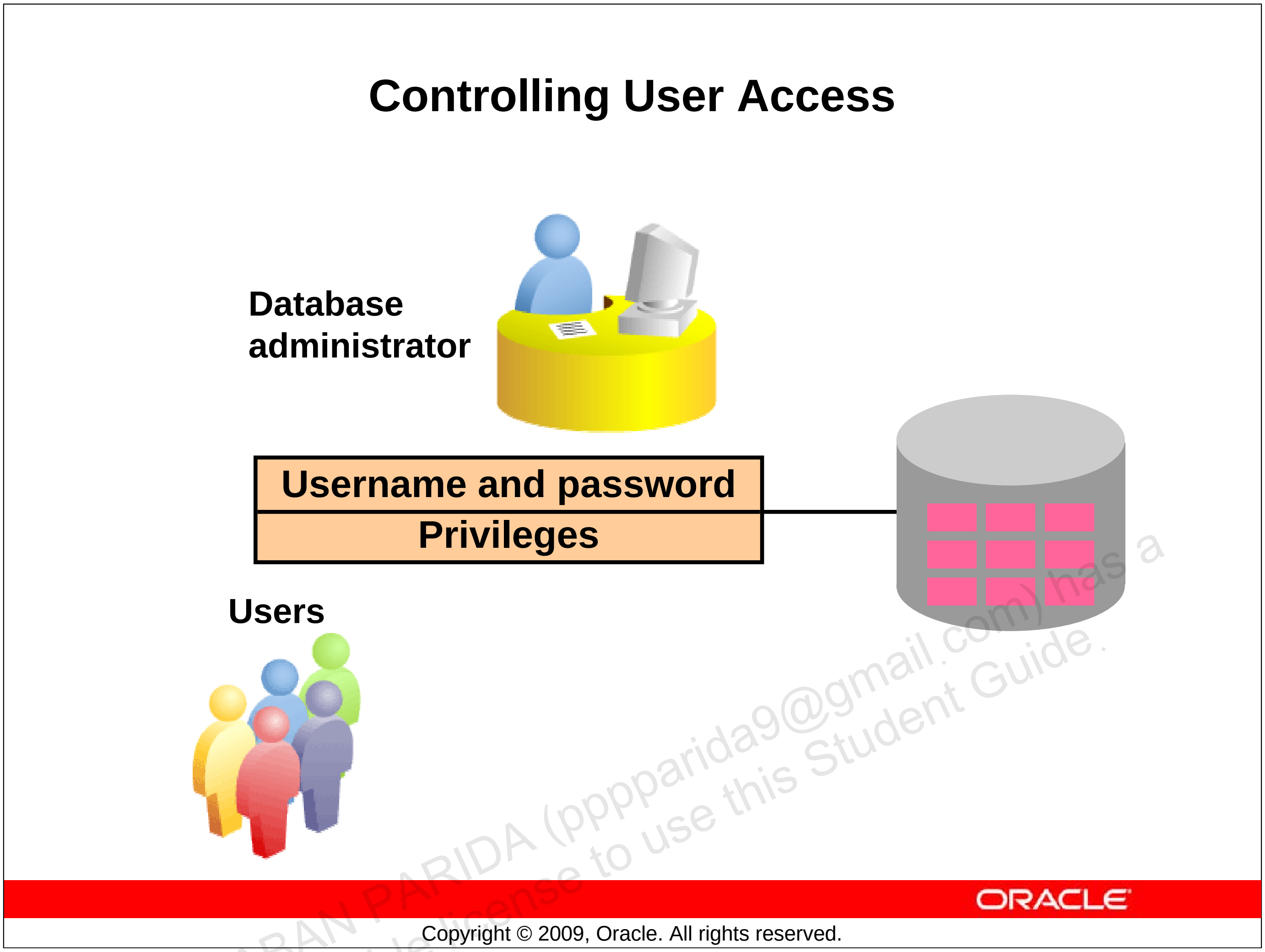
Objectives

After completing this lesson, you should be able to do the following:

- Differentiate system privileges from object privileges
- Grant privileges on tables
- View privileges in the data dictionary
- Grant roles
- Distinguish between privileges and roles

Objectives

In this lesson, you learn how to control database access to specific objects and add new users with different levels of access privileges.



Controlling User Access

Database administrator

Username and password

Privileges

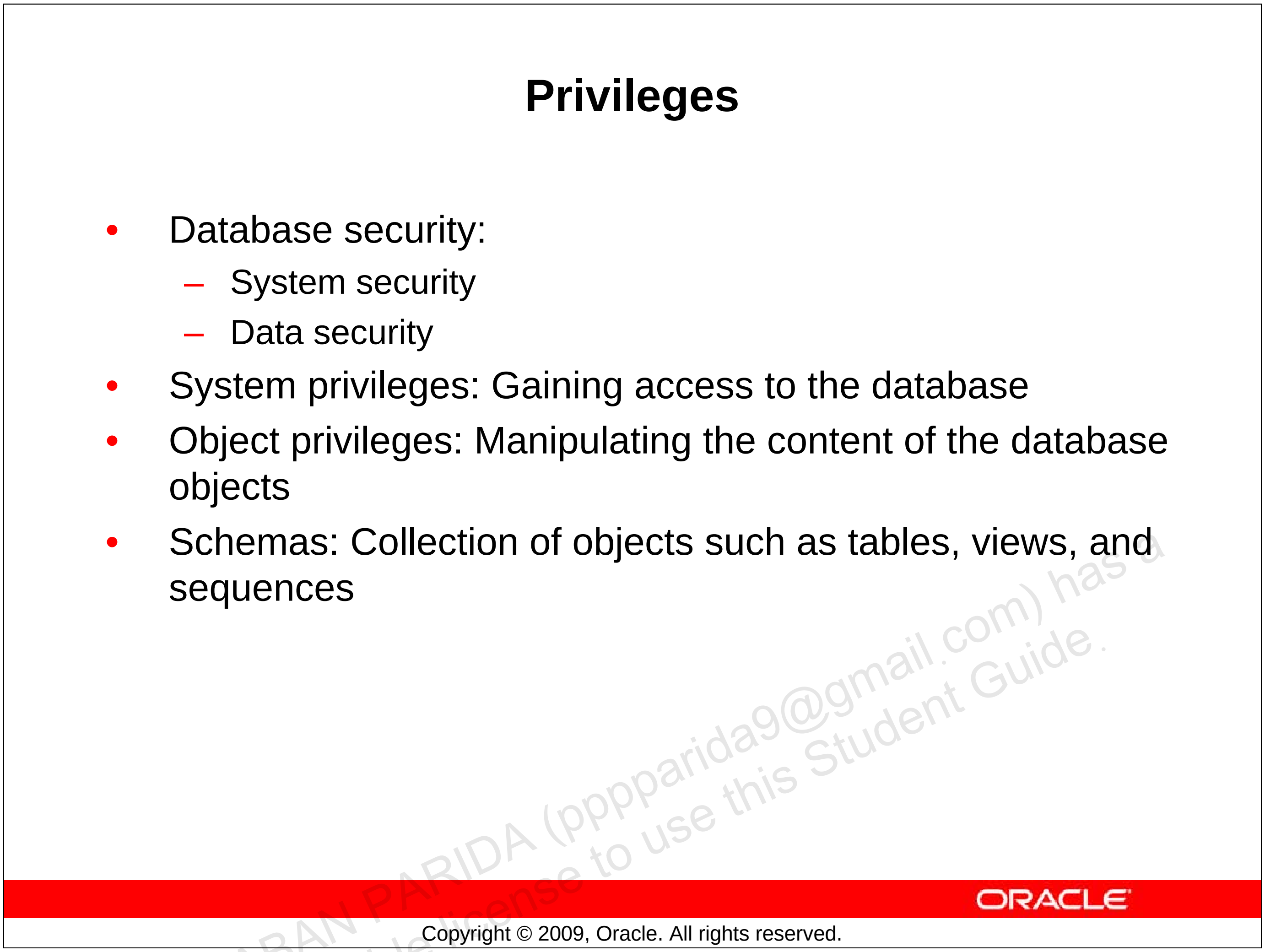
Users

Controlling User Access

In a multiple-user environment, you want to maintain security of the database access and use. With Oracle server database security, you can do the following:

- Control database access.
- Give access to specific objects in the database.
- Confirm given and received privileges with the Oracle data dictionary.
- Create synonyms for database objects.

Database security can be classified into two categories: system security and data security. System security covers access and use of the database at the system level such as the username and password, the disk space allocated to users, and the system operations that users can perform. Database security covers access and use of the database objects and the actions that those users can have on the objects.



Privileges

- Database security:
  - System security
  - Data security
- System privileges: Gaining access to the database
- Object privileges: Manipulating the content of the database objects
- Schemas: Collection of objects such as tables, views, and sequences

Privileges

Privileges are the right to execute particular SQL statements. The database administrator (DBA) is a high-level user with the ability to create users and grant users access to the database and its objects. Users require *system privileges* to gain access to the database and *object privileges* to manipulate the content of the objects in the database. Users can also be given the privilege to grant additional privileges to other users or to *roles*, which are named groups of related privileges.

Schemas

A *schema* is a collection of objects such as tables, views, and sequences. The schema is owned by a database user and has the same name as that user.

For more information, see the *Oracle Database10g Application Developer's Guide–Fundamentals* reference manual.

System Privileges

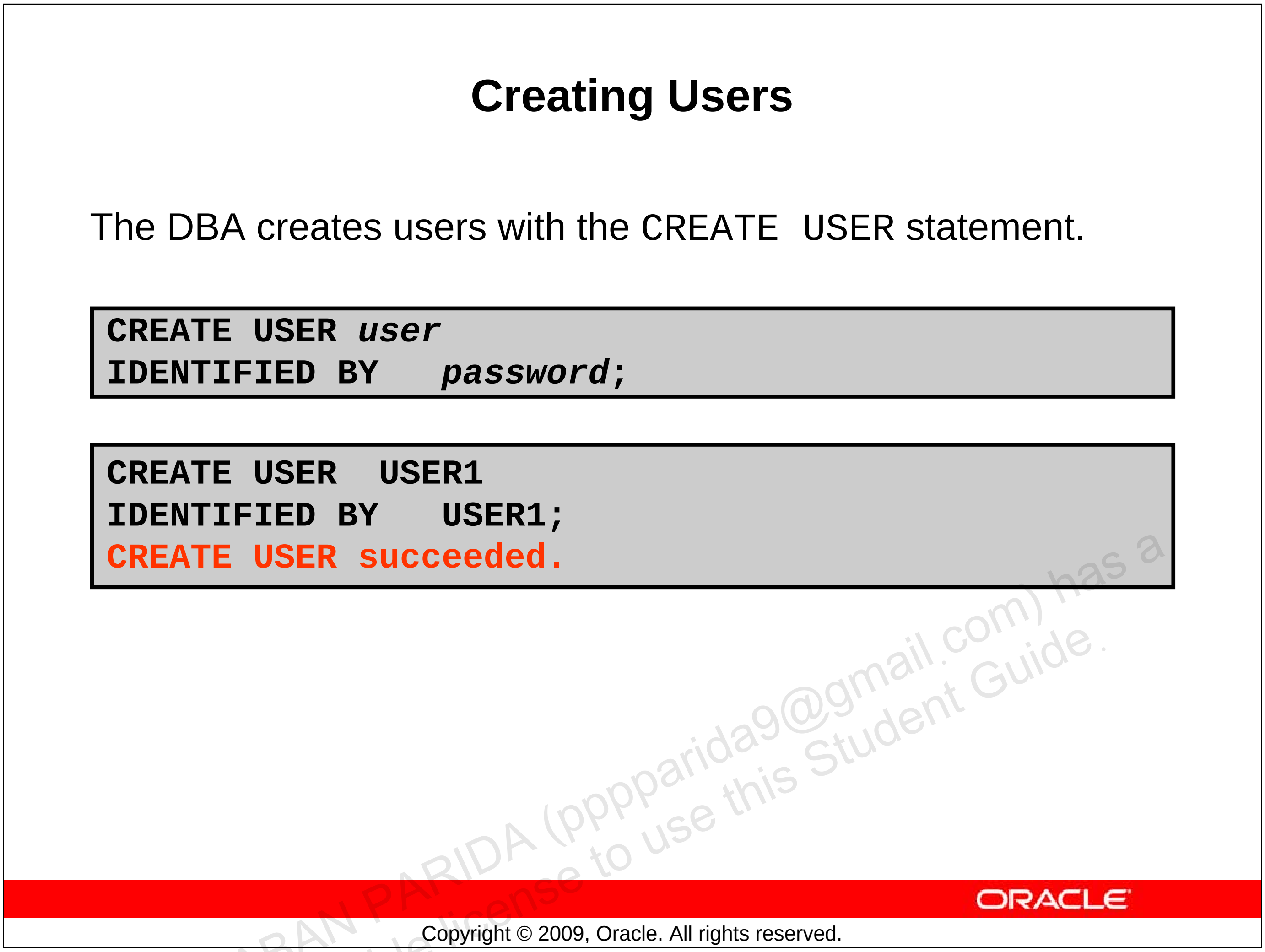
- More than 100 privileges are available.
- The database administrator has high-level system privileges for tasks such as:
  - Creating new users
  - Removing users
  - Removing tables
  - Backing up tables

System Privileges

More than 100 distinct system privileges are available for users and roles. System privileges typically are provided by the database administrator.

Typical DBA Privileges

System Privilege	Operations Authorized
`CREATE USER`	Grantee can create other Oracle users.
`DROP USER`	Grantee can drop another user.
`DROP ANY TABLE`	Grantee can drop a table in any schema.
`BACKUP ANY TABLE`	Grantee can back up any table in any schema with the export utility.
`SELECT ANY TABLE`	Grantee can query tables, views, or materialized views in any schema.
`CREATE ANY TABLE`	Grantee can create tables in any schema.



Creating Users

The DBA creates users with the `CREATE USER` statement.

```
CREATE USER user
IDENTIFIED BY   password;
```

```
CREATE USER  USER1
IDENTIFIED BY   USER1;
CREATE USER succeeded.
```

Creating a User

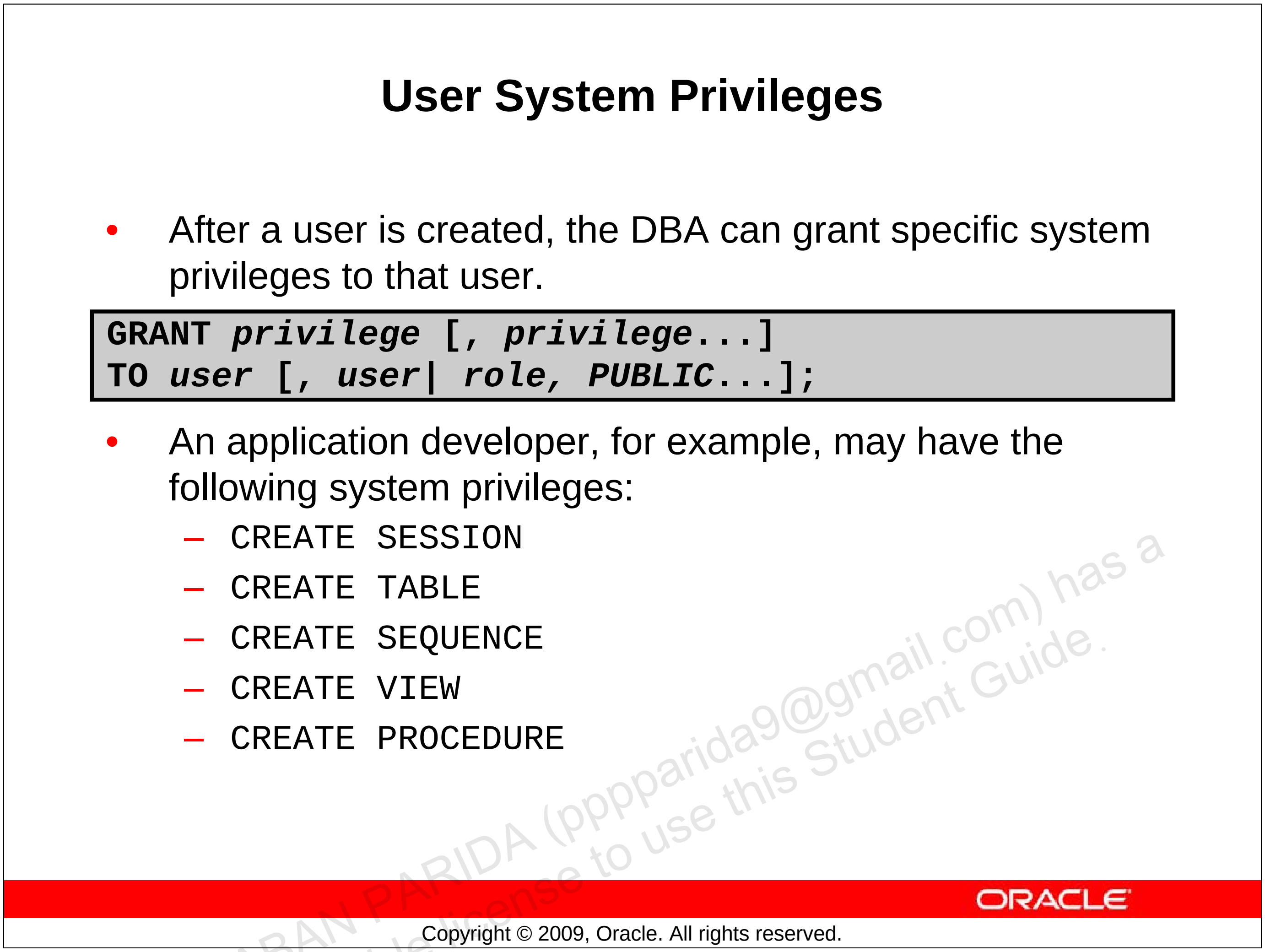
The DBA creates a user by executing the `CREATE USER` statement. The user does not have any privileges at this point. The DBA can then grant privileges to that user. These privileges determine what the user can do at the database level.

The slide gives the abridged syntax for creating a user.

In the syntax:

`user`	Is the name of the user to be created
`Password`	Specifies that the user must log in with this password

For more information, see *Oracle Database10g SQL Reference*.



User System Privileges

- After a user is created, the DBA can grant specific system privileges to that user.

```
GRANT privilege [, privilege...]
TO user [, user| role, PUBLIC...];
```

- An application developer, for example, may have the following system privileges:
  - `CREATE SESSION`
  - `CREATE TABLE`
  - `CREATE SEQUENCE`
  - `CREATE VIEW`
  - `CREATE PROCEDURE`

Typical User Privileges

After the DBA creates a user, the DBA can assign privileges to that user.

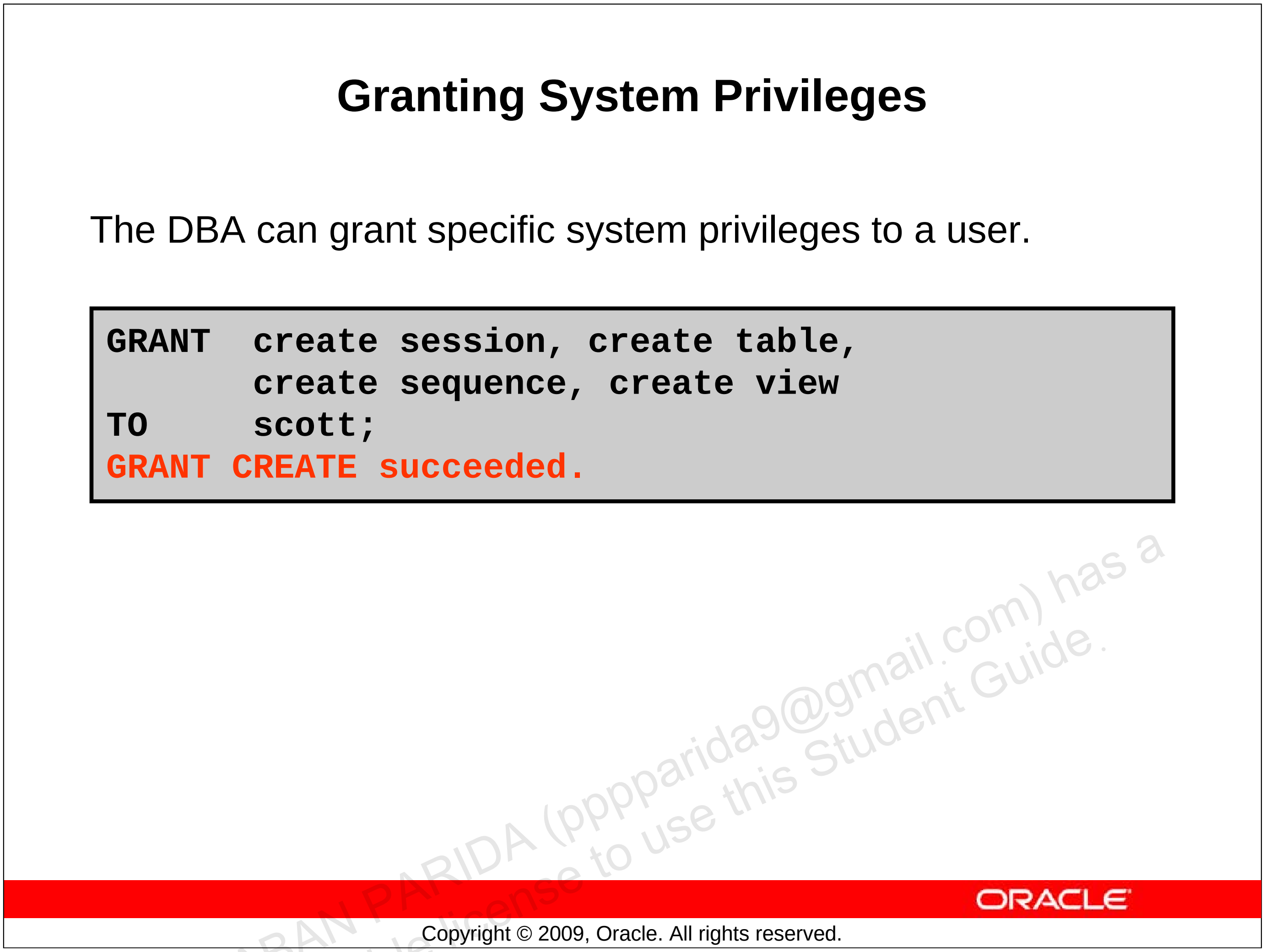
System Privilege	**Operations Authorized**
`CREATE SESSION`	Connect to the database.
`CREATE TABLE`	Create tables in the user's schema.
`CREATE SEQUENCE`	Create a sequence in the user's schema.
`CREATE VIEW`	Create a view in the user's schema.
`CREATE PROCEDURE`	Create a stored procedure, function, or package in the user's schema.

In the syntax:

`privilege` Is the system privilege to be granted

`user` `|role|PUBLIC` Is the name of the user, the name of the role, or `PUBLIC` designates that every user is granted the privilege

Note: Current system privileges can be found in the `SESSION_PRIVS` dictionary view.



Granting System Privileges

The DBA can grant specific system privileges to a user.

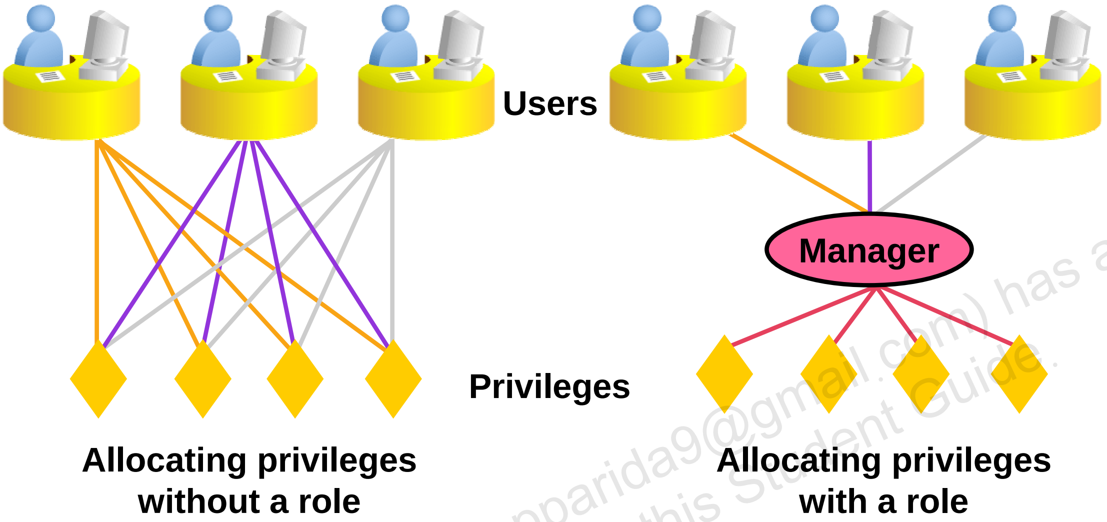
```
GRANT  create session, create table,
       create sequence, create view
TO     scott;
GRANT CREATE succeeded.
```

Granting System Privileges

The DBA uses the `GRANT` statement to allocate system privileges to the user. After the user has been granted the privileges, the user can immediately use those privileges.

In the example in the slide, user Scott has been assigned the privileges to create sessions, tables, sequences, and views.

What Is a Role?



Users

Manager

Privileges

Allocating privileges without a role

Allocating privileges with a role

What Is a Role?

A role is a named group of related privileges that can be granted to the user. This method makes it easier to revoke and maintain privileges.

A user can have access to several roles, and several users can be assigned the same role. Roles are typically created for a database application.

Creating and Assigning a Role

First, the DBA must create the role. Then the DBA can assign privileges to the role and assign the role to users.

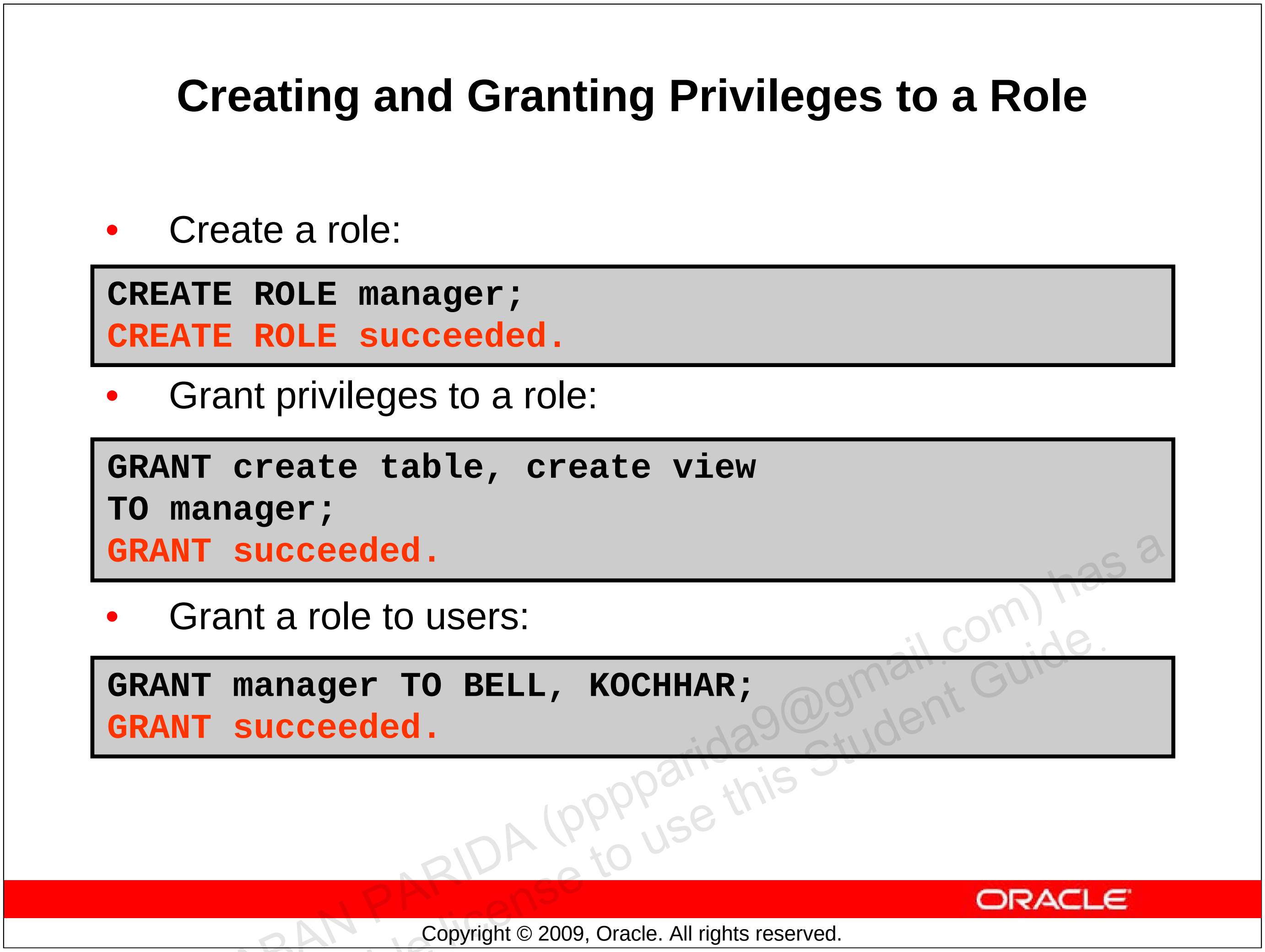
Syntax

```
CREATE   ROLE role;
```

In the syntax:

role   Is the name of the role to be created

After the role is created, the DBA can use the `GRANT` statement to assign the role to users as well as assign privileges to the role.



Creating and Granting Privileges to a Role

- Create a role:

```
CREATE ROLE manager;
CREATE ROLE succeeded.
```

- Grant privileges to a role:

```
GRANT create table, create view
TO manager;
GRANT succeeded.
```

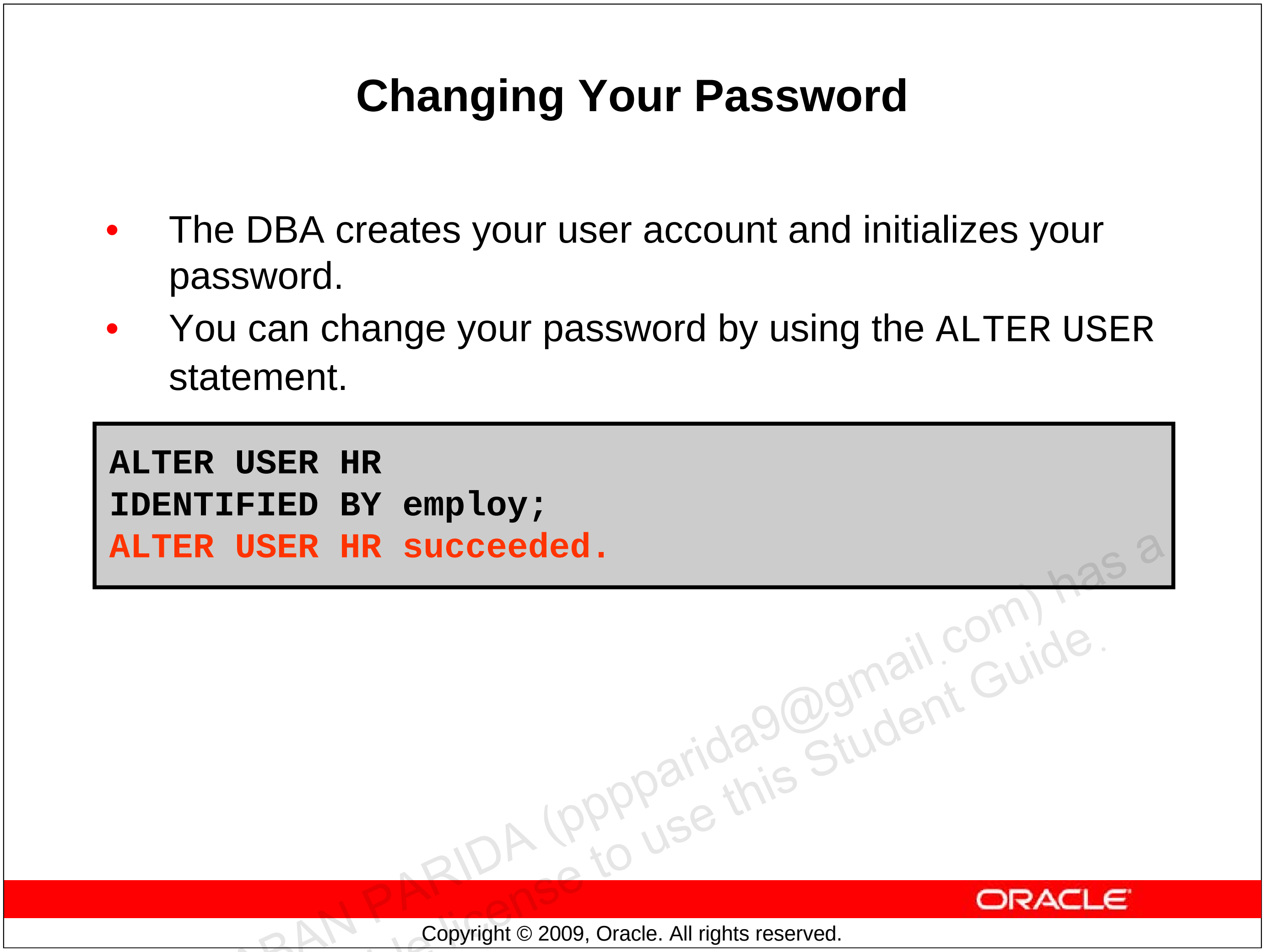
- Grant a role to users:

```
GRANT manager TO BELL, KOCHHAR;
GRANT succeeded.
```

Creating a Role

The example in the slide creates a manager role and then enables managers to create tables and views. It then grants Bell and Kochhar the role of managers. Now Bell and Kochhar can create tables and views.

If users have multiple roles granted to them, they receive all the privileges associated with all the roles.



Changing Your Password

- The DBA creates your user account and initializes your password.
- You can change your password by using the ALTER USER statement.

```
ALTER USER HR
IDENTIFIED BY employ;
ALTER USER HR succeeded.
```

Changing Your Password

The DBA creates an account and initializes a password for every user. You can change your password by using the ALTER USER statement.

Syntax

```
ALTER USER user IDENTIFIED BY password;
```

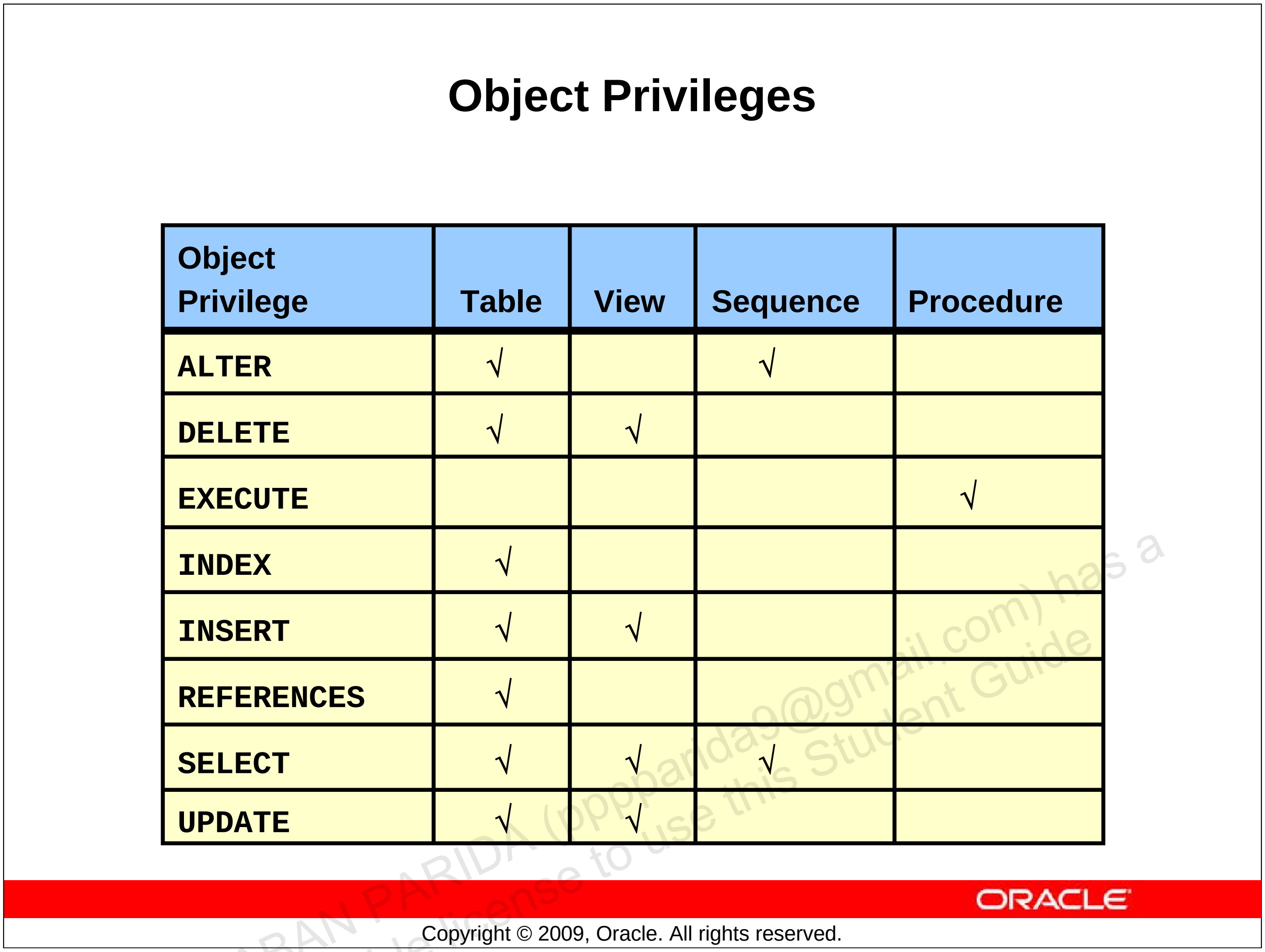
In the syntax:

user	Is the name of the user
password	Specifies the new password

Although this statement can be used to change your password, there are many other options. You must have the ALTER USER privilege to change any other option.

For more information, see the *Oracle Database10g SQL Reference* manual.

Note: SQL*Plus has a PASSWORD command (PASSW) that can be used to change the password of a user when the user is logged in. This command is not available in SQL Developer.

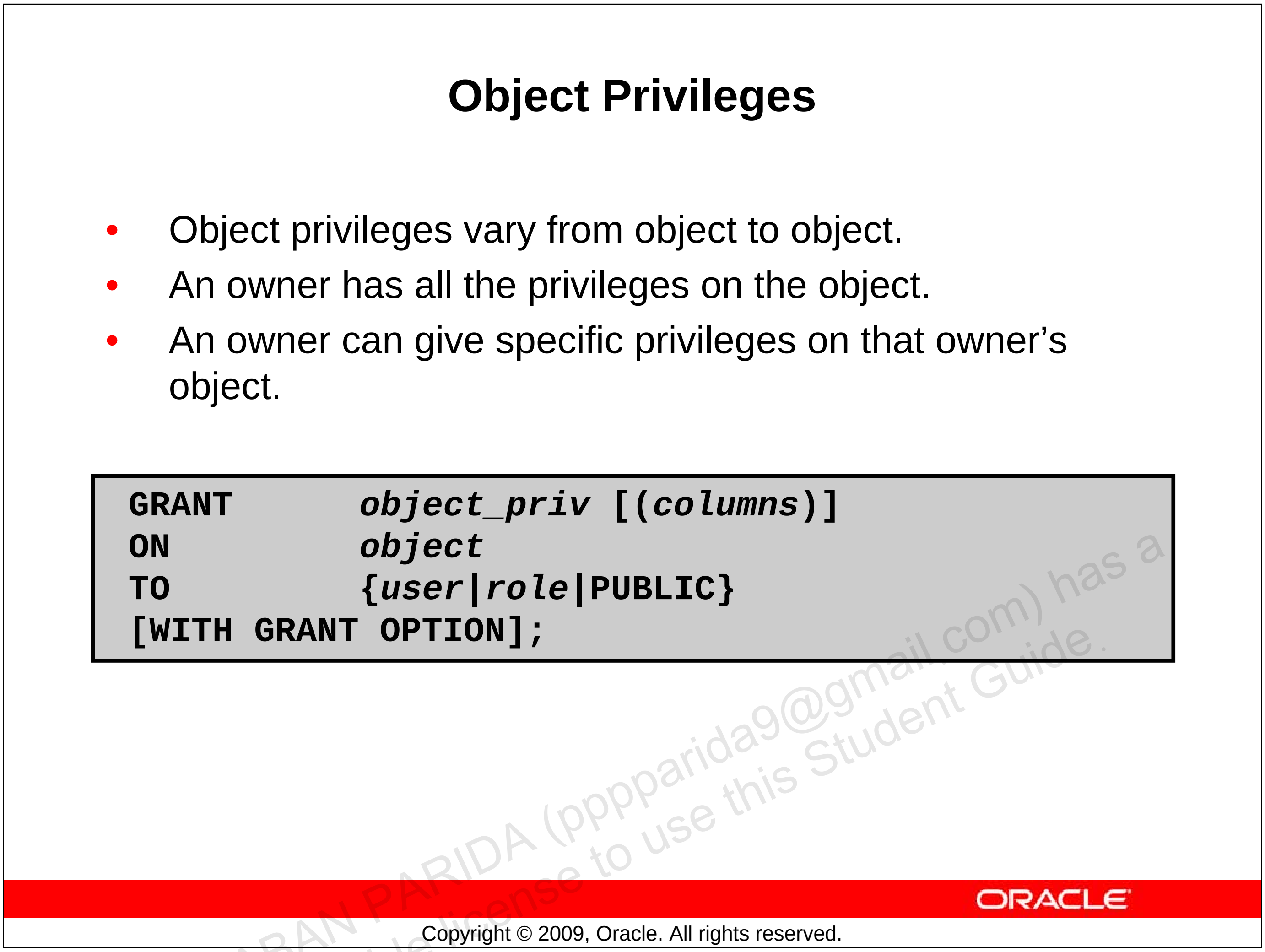


Object Privileges

Object Privilege	Table	View	Sequence	Procedure
ALTER	√		√	
DELETE	√	√		
EXECUTE				√
INDEX	√			
INSERT	√	√		
REFERENCES	√			
SELECT	√	√	√	
UPDATE	√	√		

Object Privileges

An *object privilege* is a privilege or right to perform a particular action on a specific table, view, sequence, or procedure. Each object has a particular set of grantable privileges. The table in the slide lists the privileges for various objects. Note that the only privileges that apply to a sequence are SELECT and ALTER. UPDATE, REFERENCES, and INSERT can be restricted by specifying a subset of updatable columns. A SELECT privilege can be restricted by creating a view with a subset of columns and granting the SELECT privilege only on the view. A privilege granted on a synonym is converted to a privilege on the base table referenced by the synonym.



Object Privileges

- Object privileges vary from object to object.
- An owner has all the privileges on the object.
- An owner can give specific privileges on that owner's object.

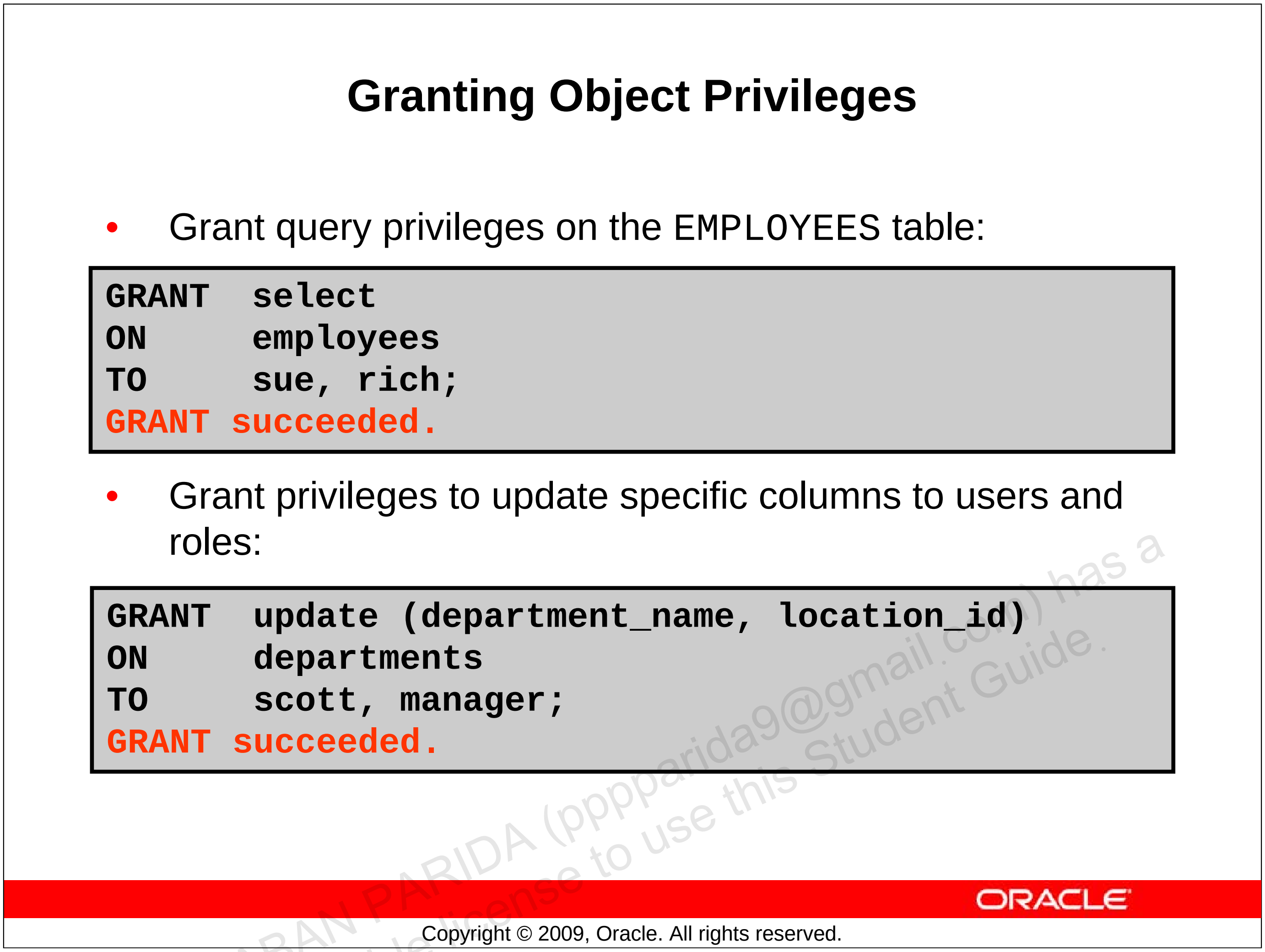
```
GRANT       object_priv [(columns)]
ON          object
TO          {user|role|PUBLIC}
[WITH GRANT OPTION];
```

Granting Object Privileges

Different object privileges are available for different types of schema objects. A user automatically has all object privileges for schema objects contained in the user's schema. A user can grant any object privilege on any schema object that the user owns to any other user or role. If the grant includes `WITH GRANT OPTION`, then the grantee can further grant the object privilege to other users; otherwise, the grantee can use the privilege but cannot grant it to other users.

In the syntax:

object_priv	Is an object privilege to be granted
ALL	Specifies all object privileges
columns	Specifies the column from a table or view on which privileges are granted
ON *object*	Is the object on which the privileges are granted
TO	Identifies to whom the privilege is granted
PUBLIC	Grants object privileges to all users
WITH GRANT OPTION	Enables the grantee to grant the object privileges to other users and roles



Granting Object Privileges

- Grant query privileges on the EMPLOYEES table:

```
GRANT  select
ON     employees
TO     sue, rich;
GRANT succeeded.
```

- Grant privileges to update specific columns to users and roles:

```
GRANT  update (department_name, location_id)
ON     departments
TO     scott, manager;
GRANT succeeded.
```

Guidelines

- To grant privileges on an object, the object must be in your own schema, or you must have been granted the object privileges WITH GRANT OPTION.
- An object owner can grant any object privilege on the object to any other user or role of the database.
- The owner of an object automatically acquires all object privileges on that object.

The first example in the slide grants users Sue and Rich the privilege to query your EMPLOYEES table. The second example grants UPDATE privileges on specific columns in the DEPARTMENTS table to Scott and to the manager role.

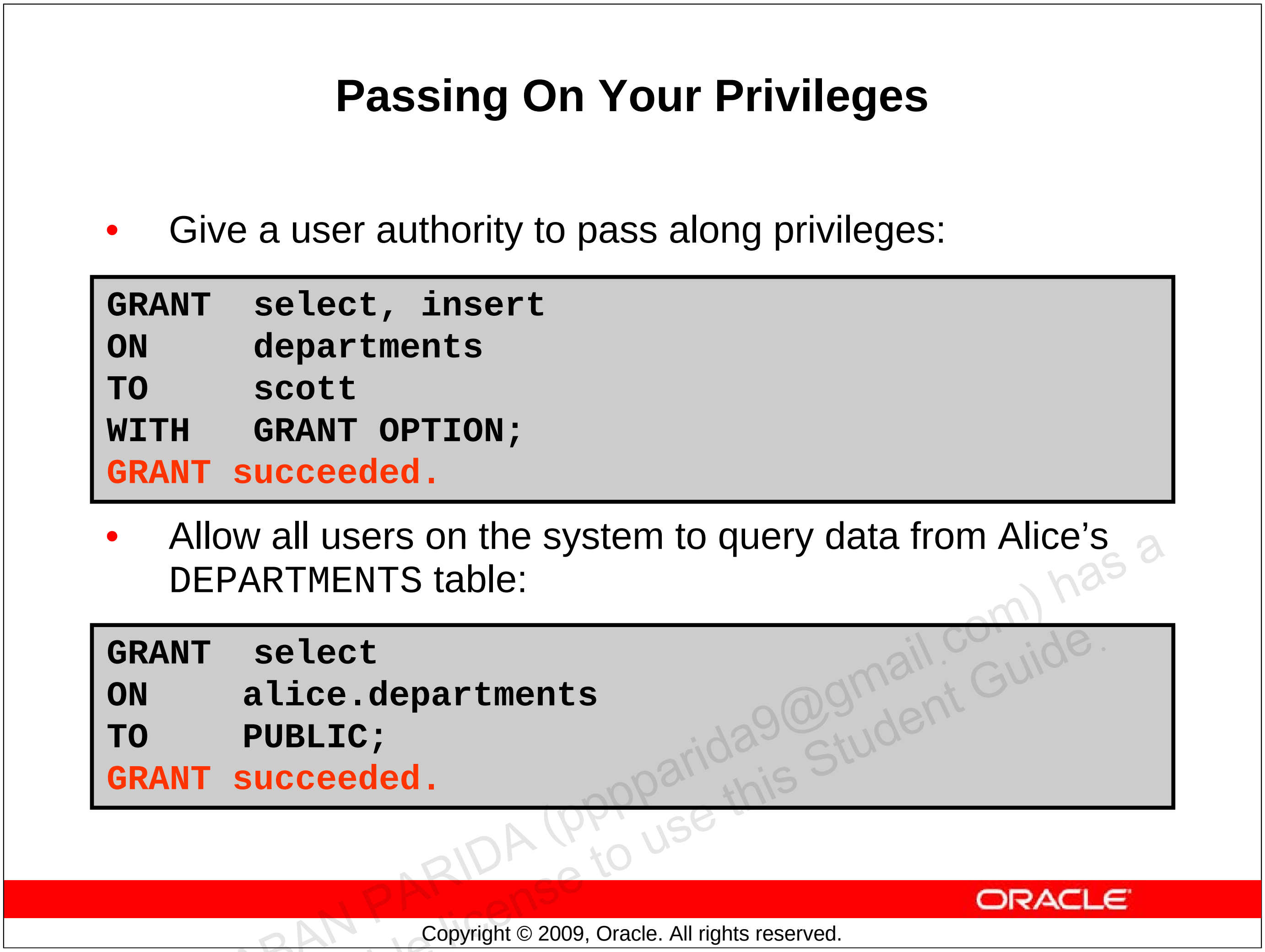
If Sue or Rich now want to use a SELECT statement to obtain data from the EMPLOYEES table, the syntax they must use is:

```
        SELECT  * FROM HR.employees;
```

Alternatively, they can create a synonym for the table and issue a SELECT statement from the synonym:

```
        CREATE SYNONYM emp FOR HR.employees;
        SELECT * FROM emp;
```

Note: DBAs generally allocate system privileges; any user who owns an object can grant object privileges.



Passing On Your Privileges

- Give a user authority to pass along privileges:

```
GRANT  select, insert
ON     departments
TO     scott
WITH   GRANT OPTION;
GRANT succeeded.
```

- Allow all users on the system to query data from Alice's DEPARTMENTS table:

```
GRANT  select
ON    alice.departments
TO    PUBLIC;
GRANT succeeded.
```

Passing On Your Privileges

WITH GRANT OPTION Keyword

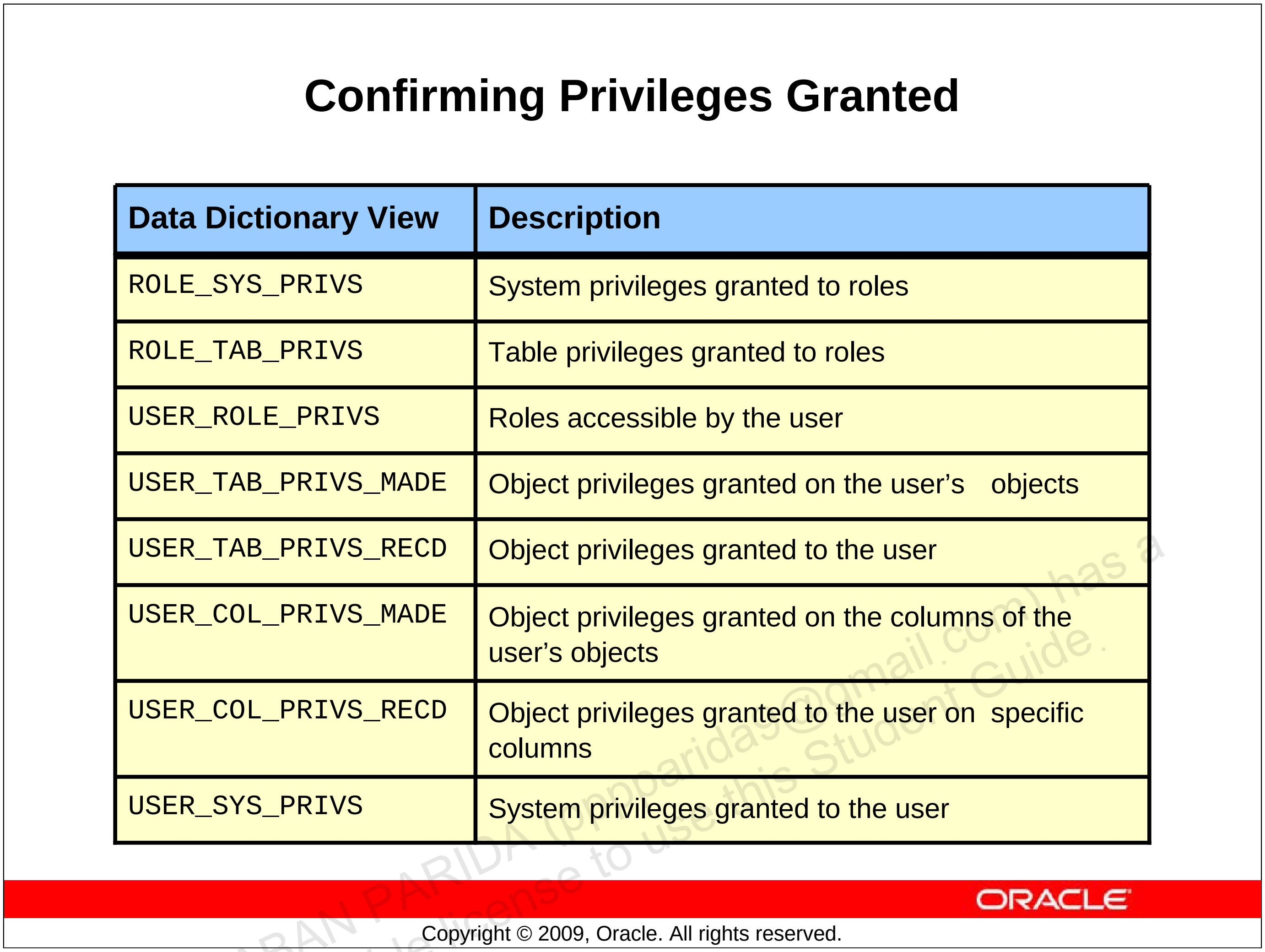
A privilege that is granted with the WITH GRANT OPTION clause can be passed on to other users and roles by the grantee. Object privileges granted with the WITH GRANT OPTION clause are revoked when the grantor's privilege is revoked.

The example in the slide gives user Scott access to your DEPARTMENTS table with the privileges to query the table and add rows to the table. The example also shows that Scott can give others these privileges.

PUBLIC Keyword

An owner of a table can grant access to all users by using the PUBLIC keyword.

The second example allows all users on the system to query data from Alice's DEPARTMENTS table.



Confirming Privileges Granted

Data Dictionary View	Description
`ROLE_SYS_PRIVS`	System privileges granted to roles
`ROLE_TAB_PRIVS`	Table privileges granted to roles
`USER_ROLE_PRIVS`	Roles accessible by the user
`USER_TAB_PRIVS_MADE`	Object privileges granted on the user’s objects
`USER_TAB_PRIVS_RECD`	Object privileges granted to the user
`USER_COL_PRIVS_MADE`	Object privileges granted on the columns of the user’s objects
`USER_COL_PRIVS_RECD`	Object privileges granted to the user on specific columns
`USER_SYS_PRIVS`	System privileges granted to the user

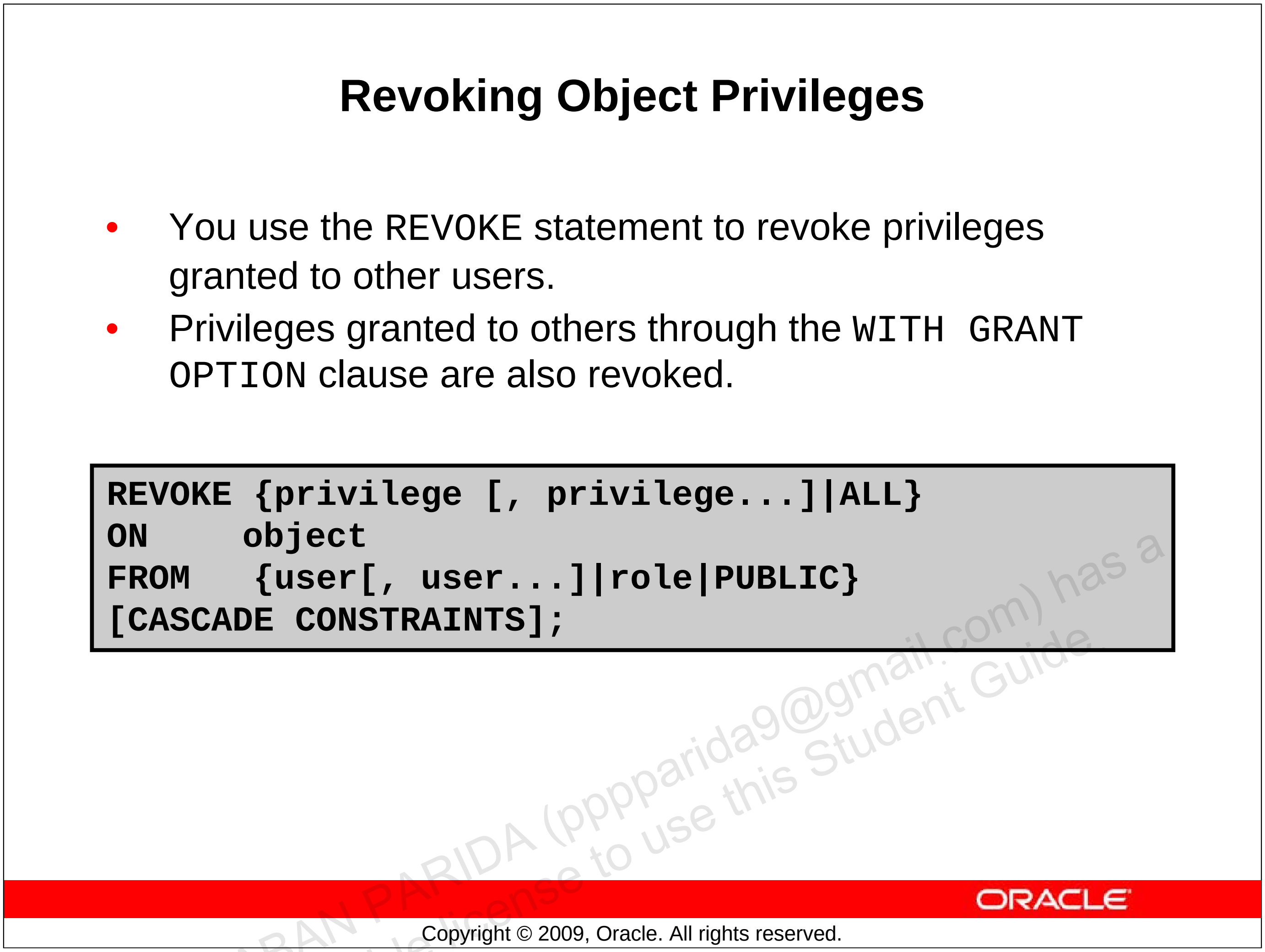
Confirming Granted Privileges

If you attempt to perform an unauthorized operation, such as deleting a row from a table for which you do not have the `DELETE` privilege, the Oracle server does not permit the operation to take place.

If you receive the Oracle server error message “Table or view does not exist,” then you have done either of the following:

- Named a table or view that does not exist
- Attempted to perform an operation on a table or view for which you do not have the appropriate privilege

You can access the data dictionary to view the privileges that you have. The chart in the slide describes various data dictionary views.



Revoking Object Privileges

- You use the REVOKE statement to revoke privileges granted to other users.
- Privileges granted to others through the WITH GRANT OPTION clause are also revoked.

```
REVOKE {privilege [, privilege...]|ALL}
ON     object
FROM   {user[, user...]|role|PUBLIC}
[CASCADE CONSTRAINTS];
```

Revoking Object Privileges

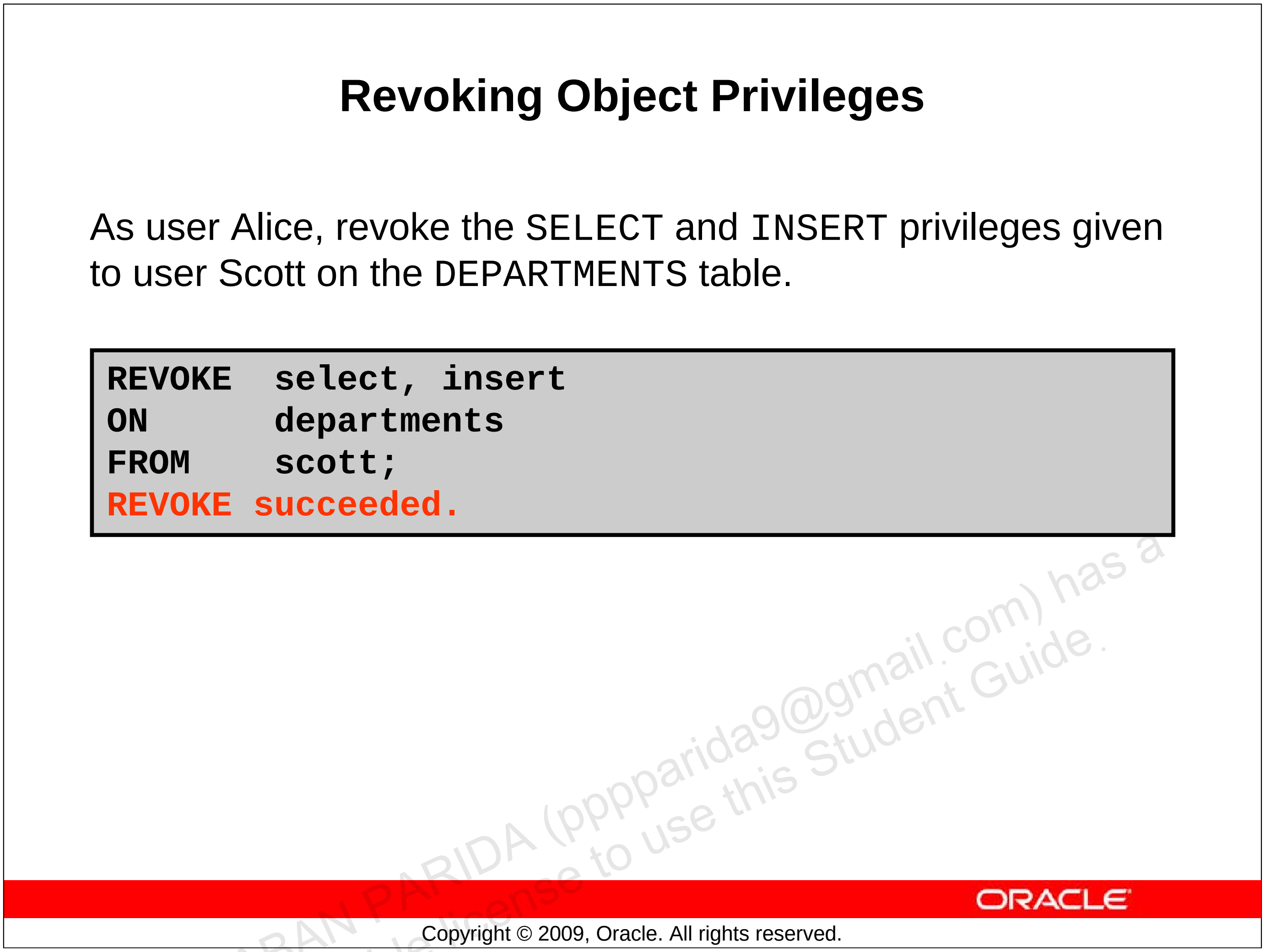
You can remove privileges granted to other users by using the REVOKE statement. When you use the REVOKE statement, the privileges that you specify are revoked from the users you name and from any other users to whom those privileges were granted by the revoked user.

In the syntax:

CASCADE is required to remove any referential integrity constraints made to the CONSTRAINTS object by means of the REFERENCES privilege

For more information, see *Oracle Database10g SQL Reference*.

Note: If a user were to leave the company and you revoke his privileges, you must regrant any privileges that this user may have granted to other users. If you drop the user account without revoking privileges from it, then the system privileges granted by this user to other users are not affected by this action.



Revoking Object Privileges

As user Alice, revoke the SELECT and INSERT privileges given to user Scott on the DEPARTMENTS table.

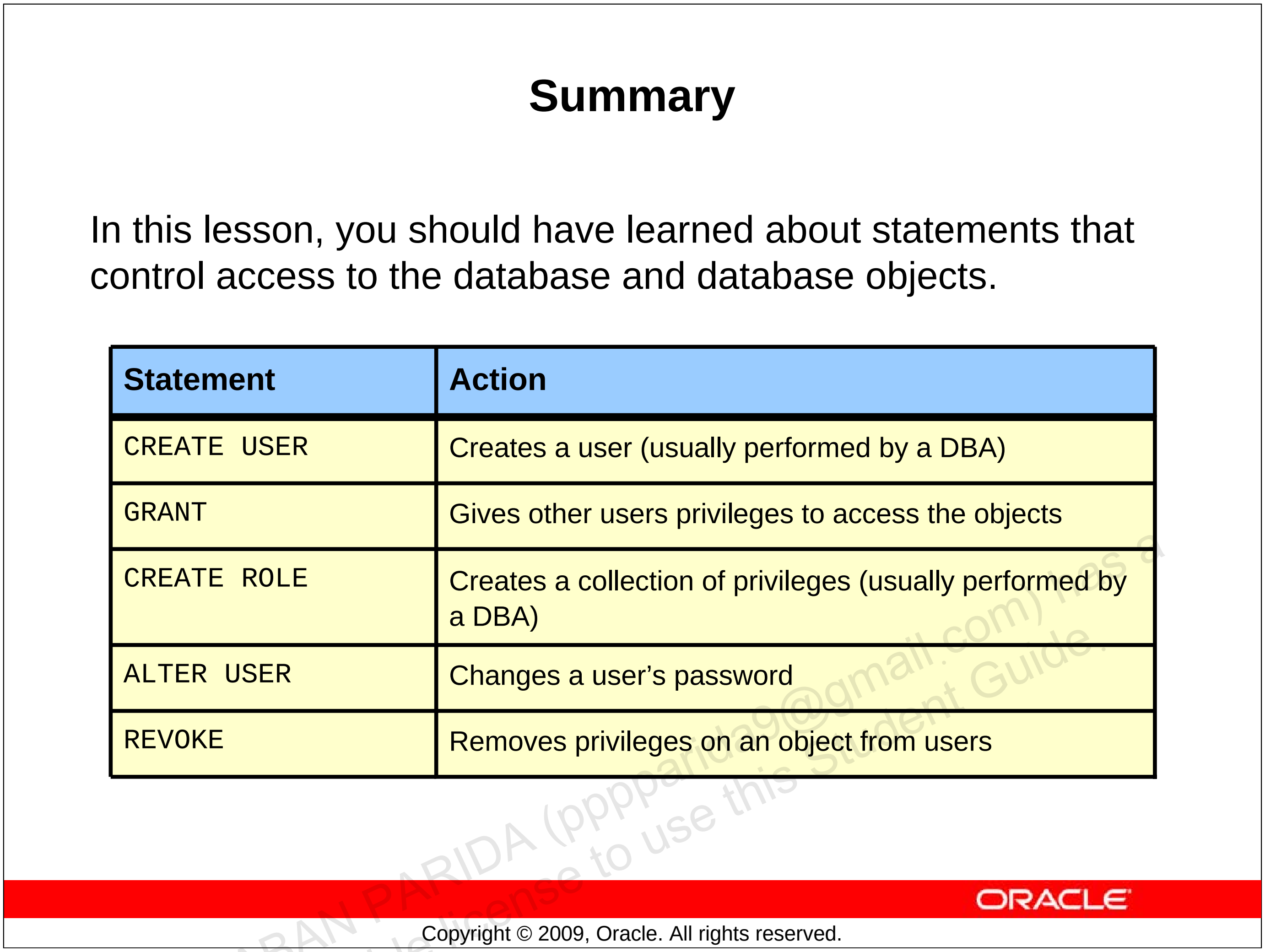
```
REVOKE  select, insert
ON      departments
FROM    scott;
REVOKE succeeded.
```

Revoking Object Privileges (continued)

The example in the slide revokes SELECT and INSERT privileges given to user Scott on the DEPARTMENTS table.

Note: If a user is granted a privilege with the WITH GRANT OPTION clause, that user can also grant the privilege with the WITH GRANT OPTION clause, so that a long chain of grantees is possible, but no circular grants (granting to a grant ancestor) are permitted. If the owner revokes a privilege from a user who granted the privilege to other users, then the revoking cascades to all the privileges granted.

For example, if user A grants a SELECT privilege on a table to user B including the WITH GRANT OPTION clause, user B can grant to user C the SELECT privilege with the WITH GRANT OPTION clause as well, and user C can then grant to user D the SELECT privilege. If user A revokes privileges from user B, then the privileges granted to users C and D are also revoked.



Summary

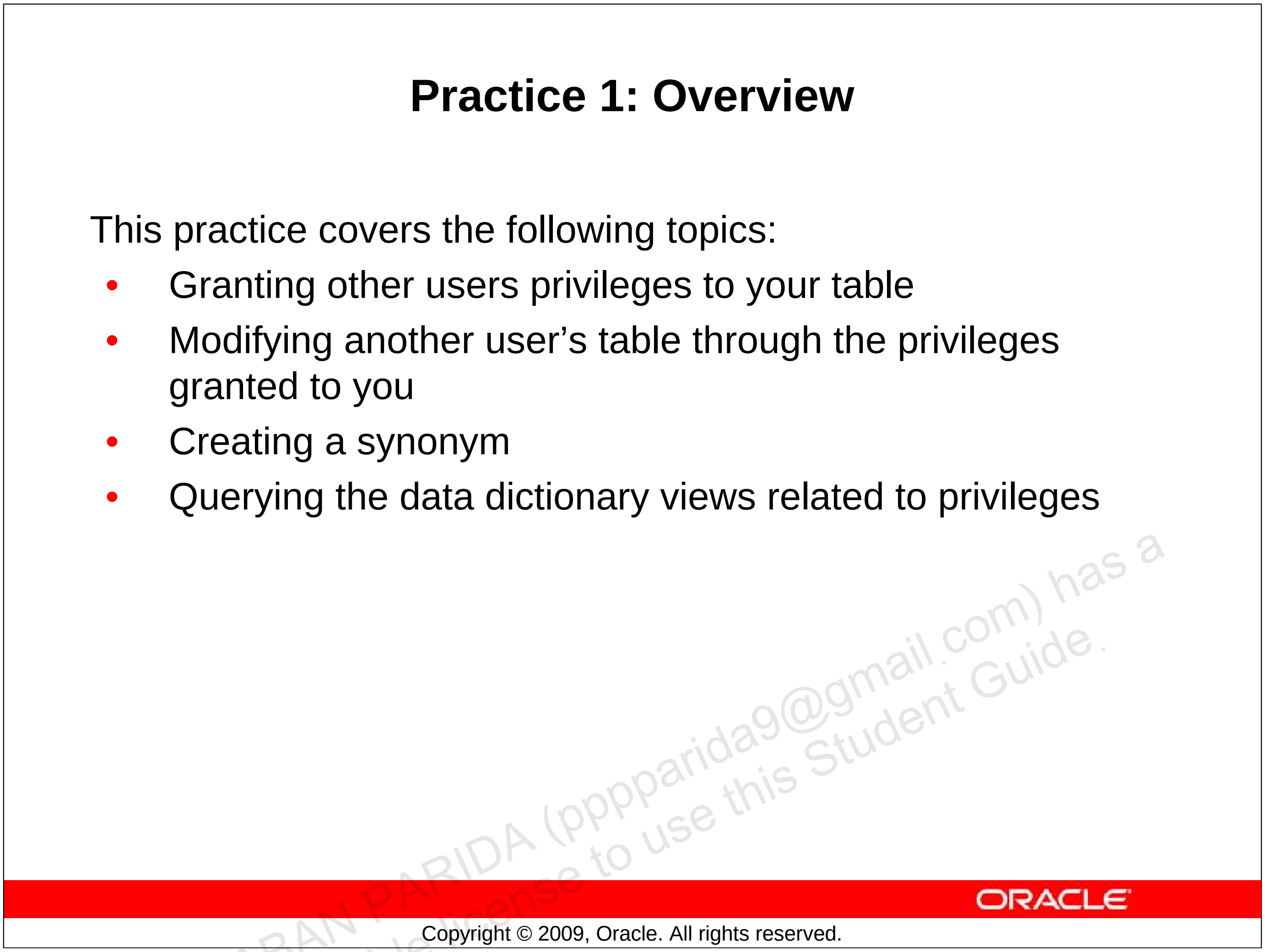
In this lesson, you should have learned about statements that control access to the database and database objects.

Statement	Action
`CREATE USER`	Creates a user (usually performed by a DBA)
`GRANT`	Gives other users privileges to access the objects
`CREATE ROLE`	Creates a collection of privileges (usually performed by a DBA)
`ALTER USER`	Changes a user's password
`REVOKE`	Removes privileges on an object from users

Summary

DBAs establish initial database security for users by assigning privileges to the users.

- The DBA creates users who must have a password. The DBA is also responsible for establishing the initial system privileges for a user.
- After the user has created an object, the user can pass along any of the available object privileges to other users or to all users by using the `GRANT` statement.
- A DBA can create roles by using the `CREATE ROLE` statement to pass along a collection of system or object privileges to multiple users. Roles make granting and revoking privileges easier to maintain.
- Users can change their password by using the `ALTER USER` statement.
- You can remove privileges from users by using the `REVOKE` statement.
- With data dictionary views, users can view the privileges granted to them and those that are granted on their objects.
- With database links, you can access data on remote databases. Privileges cannot be granted on remote objects.



Practice 1: Overview

This practice covers the following topics:

- Granting other users privileges to your table
- Modifying another user's table through the privileges granted to you
- Creating a synonym
- Querying the data dictionary views related to privileges

Practice 1: Overview

In this exercise, you practice controlling access to the database objects. You use two accounts: `ora21` and `ora22`.

Practice 1

To complete question 6 and the subsequent ones, you need to connect to the database using SQL Developer. To do so, double-click the SQL Developer icon on the desktop.

To create a new database connection in the Connections Navigator, right-click Connections. Select New Connection from the shortcut menu. The New/Select Database Connection dialog box appears.

Create a database connection using the following information:

a. Connection Name: `ora21`
b. Username: `ora21`
c. Password: `ora21`
d. Hostname: localhost
e. Port: 1521
f. SID: ORCL
g. Ensure that you select the Save Password check box.

Create another database connection using the following information:

a. Connection Name: `ora22`
b. Username: `ora22`
c. Password: `ora22`
d. Hostname: localhost
e. Port: 1521
f. SID: ORCL
g. Ensure that you select the Save Password check box.

1. Which privilege should a user be given to log on to the Oracle server? Is this a system or an object privilege?

   ____

2. Which privilege should a user be given to create tables?

   ____

3. If you create a table, who can pass along privileges to other users on your table?

   ____

4. You are the DBA. You are creating many users who require the same system privileges. What should you use to make your job easier?

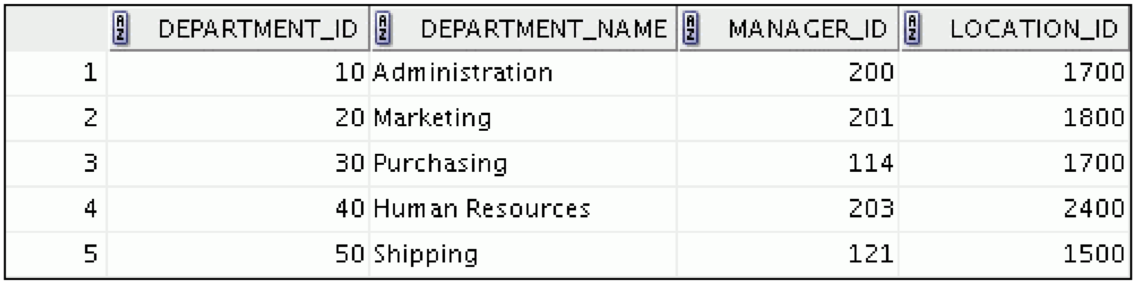
   ____

5. Which command would you use to change your password?

   ____

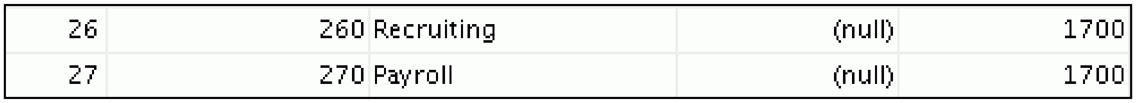
Practice 1 (continued)

6. Connect as user `ora21`. Query all the rows in your `DEPARTMENTS` table.



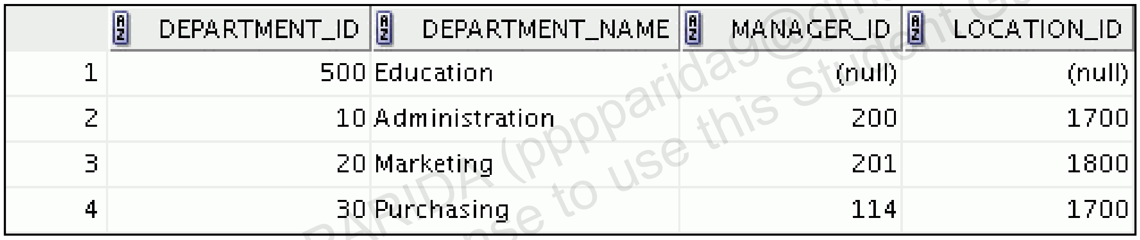
	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	30	Purchasing	114	1700
4	40	Human Resources	203	2400
5	50	Shipping	121	1500

...



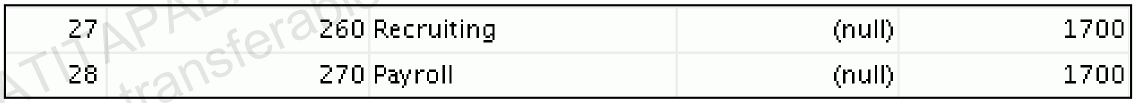
26	260	Recruiting	(null)	1700
27	270	Payroll	(null)	1700

7. Add a new row to your `DEPARTMENTS` table. Add Education as department number 500.
8. Grant user `ora22` access to your `DEPARTMENTS` table.
9. Connect as user `ora22`. Query user `ora21`'s `DEPARTMENTS` table.



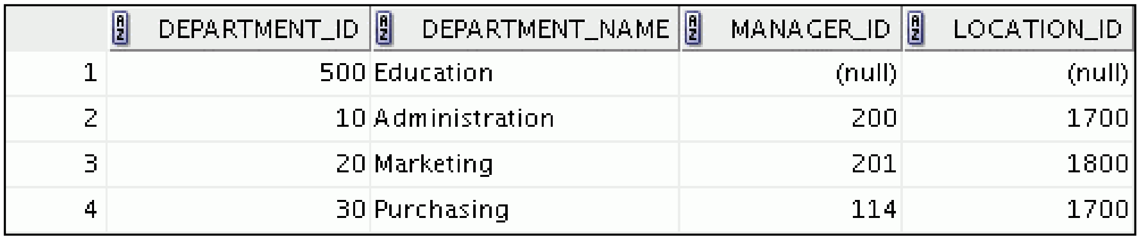
	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	500	Education	(null)	(null)
2	10	Administration	200	1700
3	20	Marketing	201	1800
4	30	Purchasing	114	1700

...



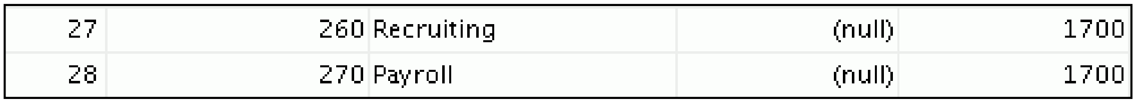
27	260	Recruiting	(null)	1700
28	270	Payroll	(null)	1700

10. Create a synonym for user `ora21`'s `DEPARTMENTS` table. Query all the rows in user `ora21`'s `DEPARTMENTS` table by using your synonym.



	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	500	Education	(null)	(null)
2	10	Administration	200	1700
3	20	Marketing	201	1800
4	30	Purchasing	114	1700

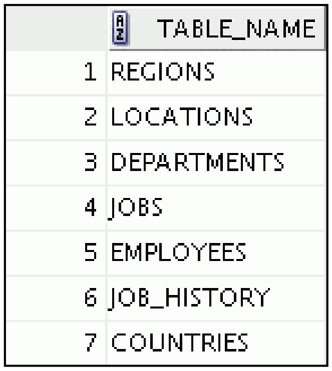
...



27	260	Recruiting	(null)	1700
28	270	Payroll	(null)	1700

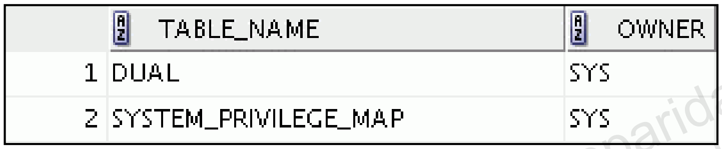
Practice 1 (continued)

11. As user `ora22`, query the `USER_TABLES` data dictionary to see information about the tables that you own.



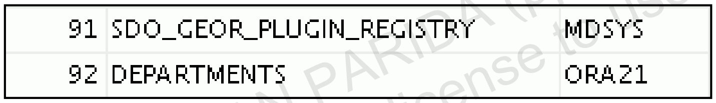
	TABLE_NAME
1	REGIONS
2	LOCATIONS
3	DEPARTMENTS
4	JOBS
5	EMPLOYEES
6	JOB_HISTORY
7	COUNTRIES

12. As user `ora22`, query the `ALL_TABLES` data dictionary view to see information about all the tables that you can access. Exclude the tables that you own.
Note: Your list may not exactly match the following list:



	TABLE_NAME	OWNER
1	DUAL	SYS
2	SYSTEM_PRIVILEGE_MAP	SYS

…



91	SDO_GEOR_PLUGIN_REGISTRY	MDSYS
92	DEPARTMENTS	ORA21

13. Connect as user `ora21` and revoke the `SELECT` privilege from user `ora22`.

14. Remove the row that you inserted into the `DEPARTMENTS` table in step 7 and save the changes.