幻灯片 1

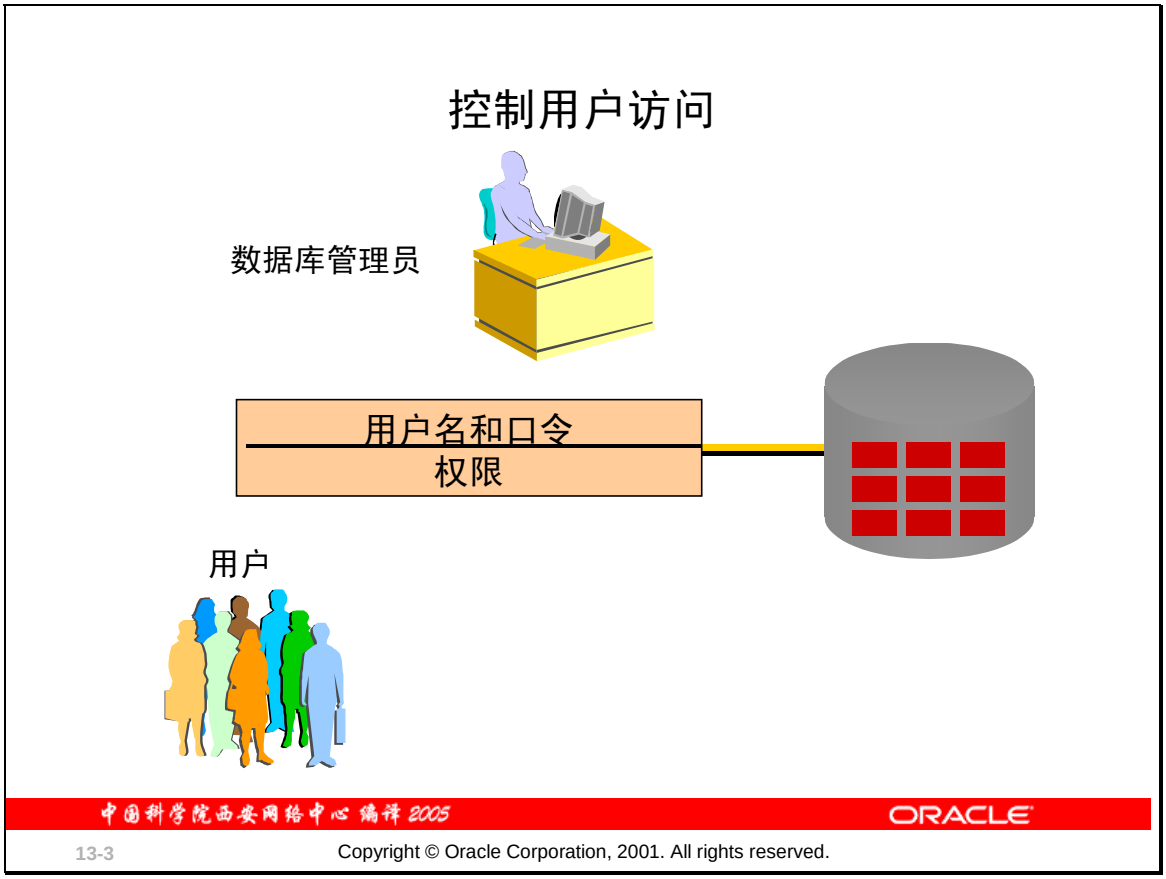
进度表：	时间	主题
	20 分钟	讲演
	20 分钟	练习
	40 分钟	总共

幻灯片 2

课程目标

在本课中，你将学习怎样控制对指定数据库对象的访问，并且用不同级别的访问权限添加新的用户。

幻灯片 3



控制用户访问

在多用户环境中，你想要控制数据库访问和使用的安全，可以用下面 Oracle 服务器提供的数据库安全措施：

- 控制数据库访问
- 在数据库中只允许访问指定的对象
- 用 Oracle 数据字典确认给予的和收回的权限
- 创建数据库对象的同义词

数据库安全可以被划分为两个范畴：系统安全和数据安全。系统安全在系统级别访问和使用数据库，例如，用户名和口令，分配给用户的磁盘空间和用户能够执行的系统操作；数据库安全包括访问和使用数据库对象和用户能够对数据库对象进行的操作。

幻灯片 4

权限

- 数据库安全：
  - 系统安全
  - 数据安全
- 系统权限：授权访问数据库
- 对象权限：操纵数据对象的内容
- 方案：对象的集合，例如表、视图和序列

权限

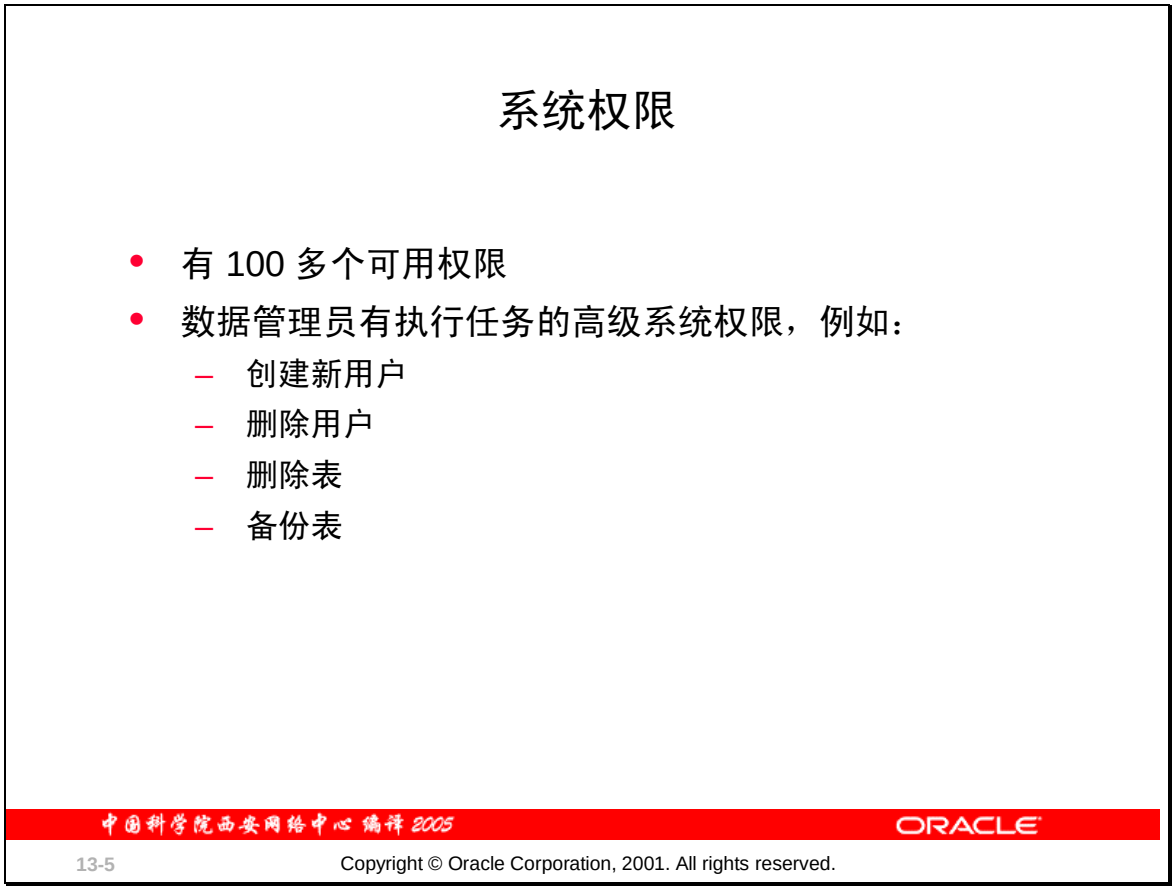
权限是执行特殊 SQL 语句的权利。数据库管理员 (DBA) 是一个具有授予用户访问数据库及其对象的能力的高级用户。用户需要*系统权限*来访问数据库，需要*对象权限*来操纵数据库中对象的内容。用户也可以将被授予的权限给其它用户或者角色，角色是相关权限的命名分组。

方案

方案是对象的集合，例如，视图和序列的集合。方案被数据库用户多拥有，并且与用户有相同的名字。

更多信息，见 *Oracle9i Application Developer's Guide – Fundamentals*，“建立安全策略” 部分；*Oracle9i Concepts*，“数据库安全”主题。

幻灯片 5



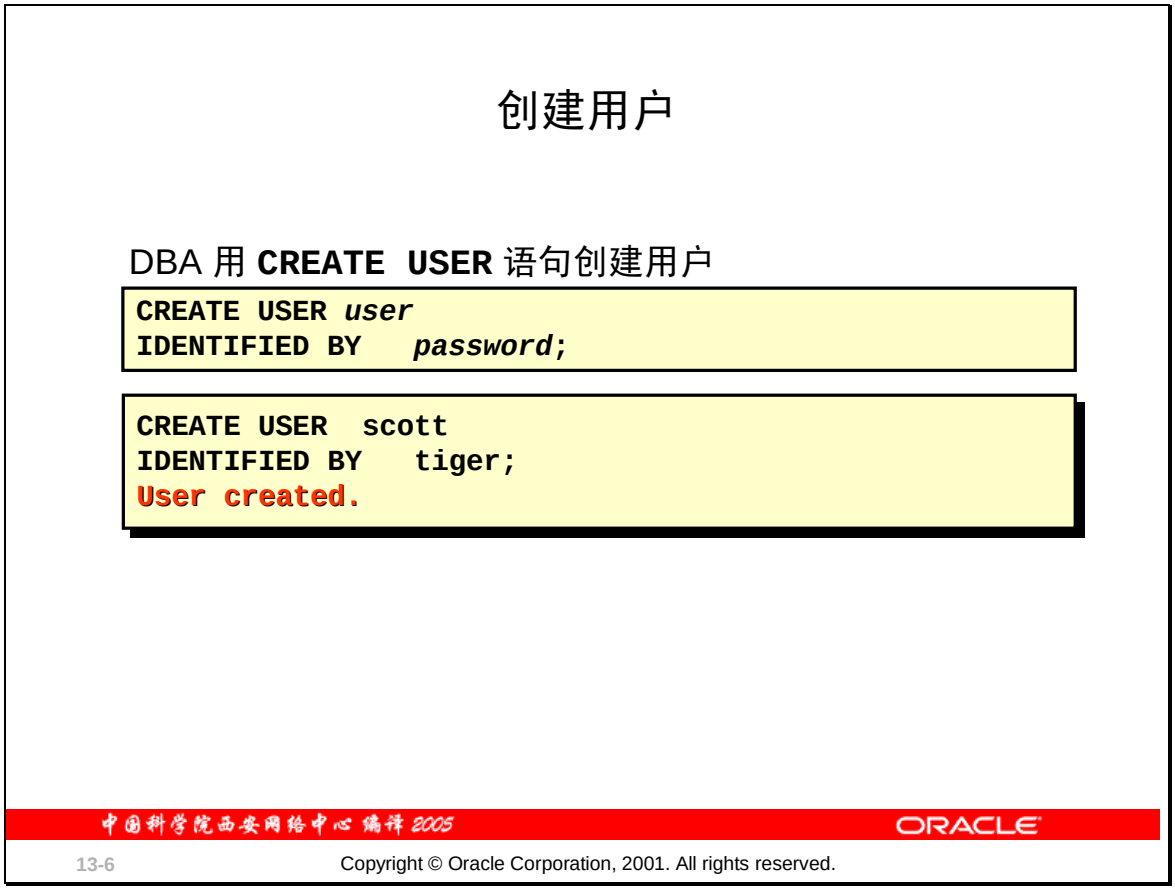
系统权限

对用户和角色有 100 多个不同的可用系统权限，系统权限有数据库管理员提供。

典型的 DBA 权限

系统权限	授权的操作
CREATE USER	受让人可以创建其他 Oracle 用户 (需要有 DBA 角色权限)。
DROP USER	受让人可以删除另一个用户。
DROP ANY TABLE	受让人可以删除在任意方案中的表。
BACKUP ANY TABLE	受让人用导出实用程序可以备份在任何方案中的任何表。
SELECT ANY TABLE	受让人可以查询在任何方案中的表、视图或快照。
CREATE ANY TABLE	受让人可以在任何方案中创建表。

幻灯片 6



创建用户

DBA 通过执行 CREATE USER 语句来创建用户，在这时用户没有任何权限。DBA 可以给用户授予权限，这些权限决定用户能够在数据库级别做什么。
幻灯片给出了用有删节的语法创建用户的例子。

在语法中：

user	是被创建的用户的名字
password	指定用户必须用该口令登录

更多信息，见 *Oracle9i SQL Reference*，“授权”和“创建用户”。

教师注释

关于删除 DROP USER 用户的信息，参考 *Oracle9i SQL Reference*，“删除用户”。

幻灯片 7

典型的用户权限

现在已经创建了一个用户，DBA 可以指定权限给该用户。

系统权限	授权的操作
CREATE SESSION	连接到数据库
CREATE TABLE	在用户的方案中创建表
CREATE SEQUENCE	在用户的方案中创建序列
CREATE VIEW	在用户的方案中创建视图
CREATE PROCEDURE	在用户的方案中创建存储过程、函数或包

在语法中：

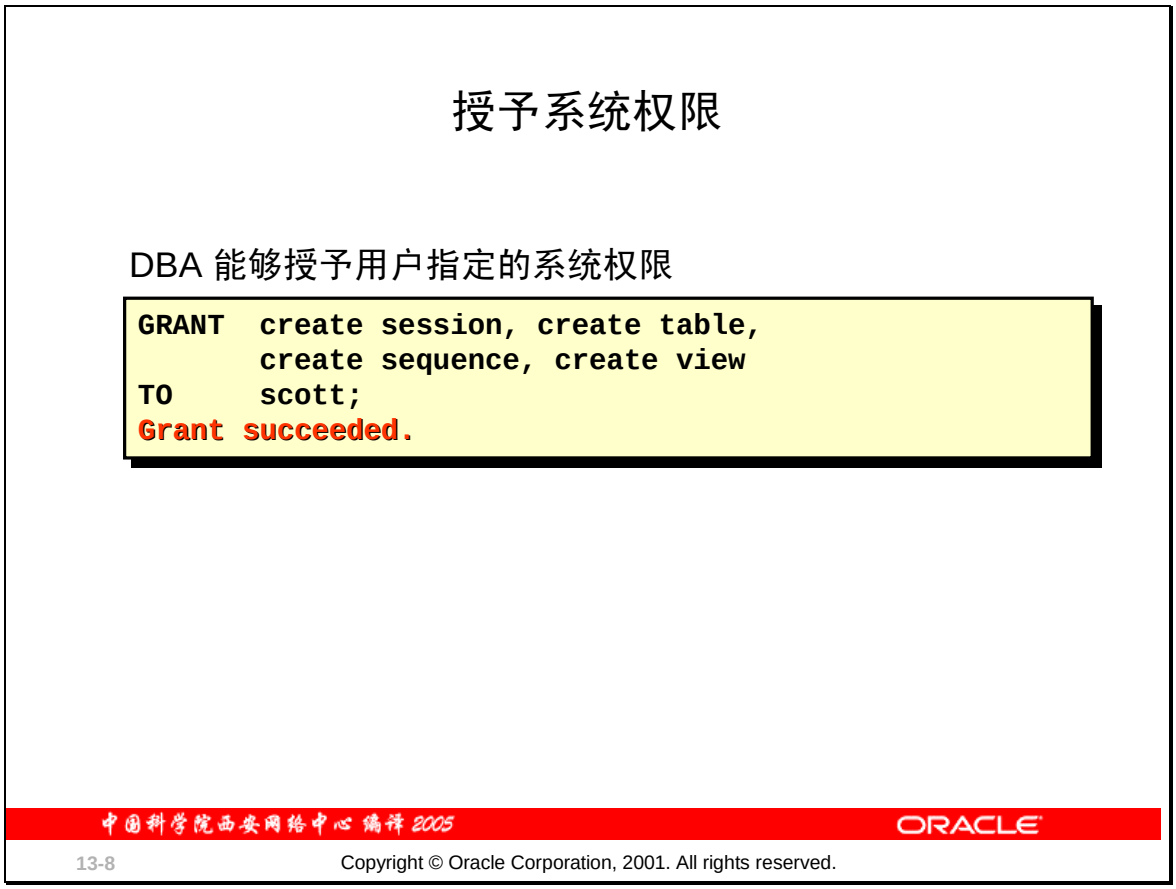
privilege	要被授予的系统权限
user|role|PUBLIC	是用户的名字，角色的名字或 PUBLIC，PUBLIC 指定每一个用户被授予权限

注：当前会话的系统权限可以在字典视图 SESSION_PRIVS 中找到。

教师注释

上面对 GRANT 命令的语法显示不是语句的全语法。

幻灯片 8



授予系统权限

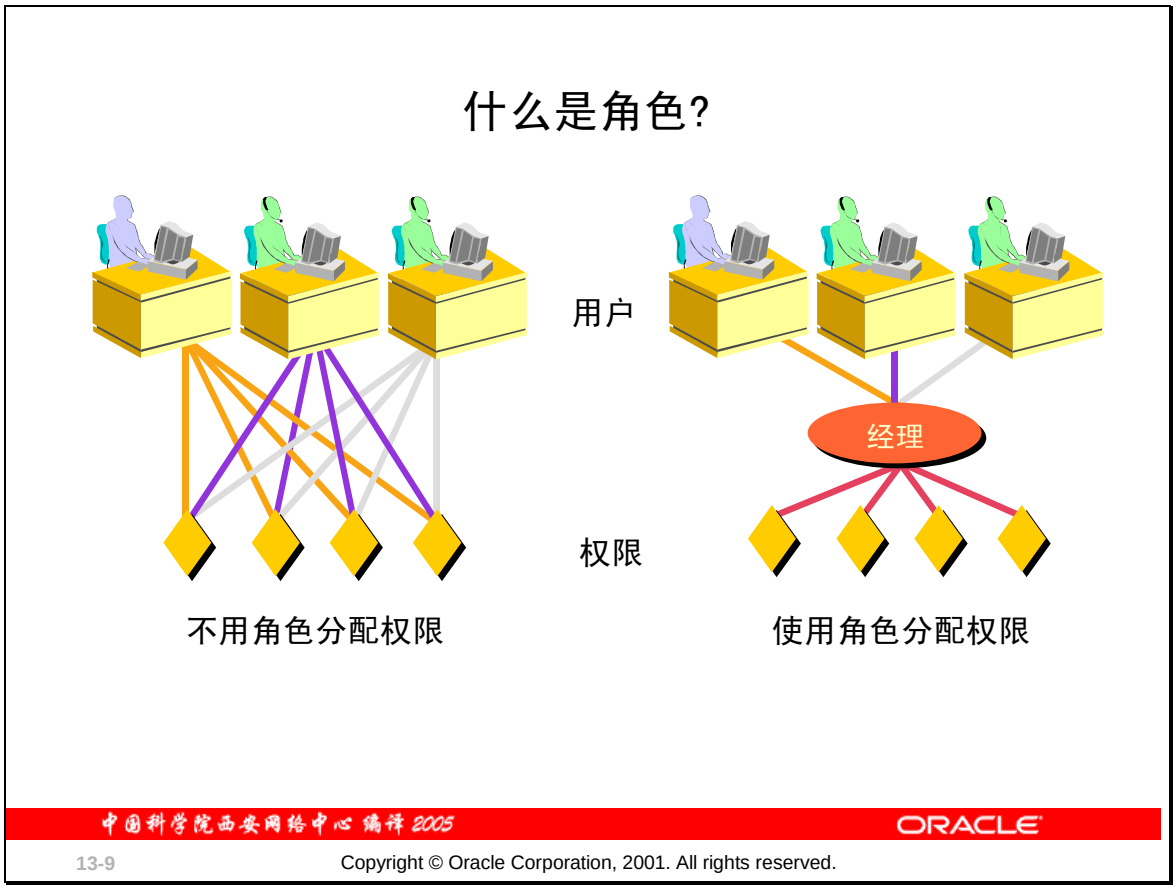
DBA 使用 GRANT 语句给用户分配系统权限，一旦用户被授予权限，用户就可以立即使用它们。

在幻灯片的例子中，用户 Scott 已经被指定了创建会话、表、序列和视图的权限。

教师注释

用户必须有足够的空间来创建表。

幻灯片 9



什么是角色?

角色是命名的可以授予用户的相关权限的组，该方法使得授予、撤回和维护权限容易的多。

一个用户可以使用几个角色，并且几个用户也可以被指定相同的角色。角色典型地为数据库应用程序创建。

创建和分配角色

首先，DBA 必须创建角色，然后，DBA 可以分配角色给角色和用户。

语法

```
CREATE  ROLE  role;
```

在语法中：

role 要被创建的角色的名字

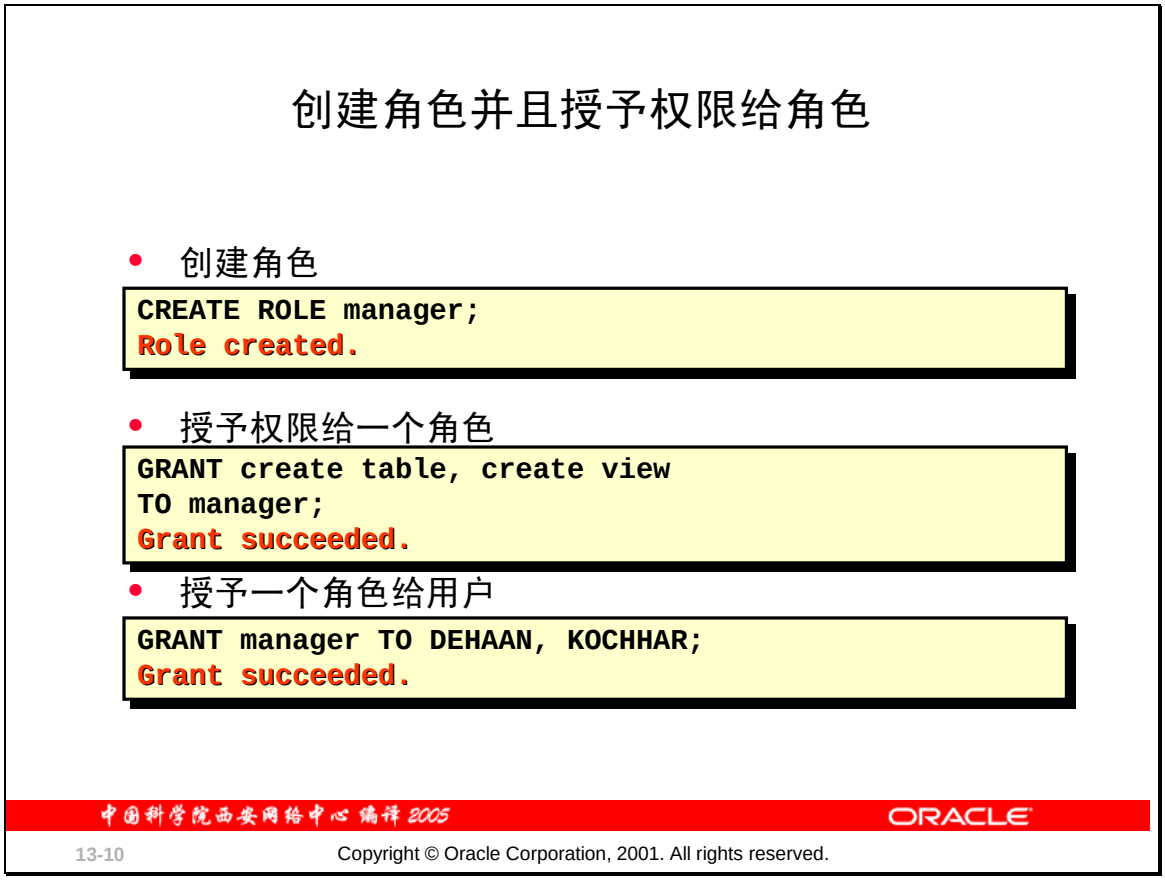
现在角色已被创建，DBA 可以用 GRANT 语句给用户指定角色，也可以指定权限给角色。

教师注释

讨论下面关于角色的 4 点：

- 一个命名的相关权限组
- 可以授予用户
- 简化授予和撤消权限的过程
- 由 DBA 创建

幻灯片 10

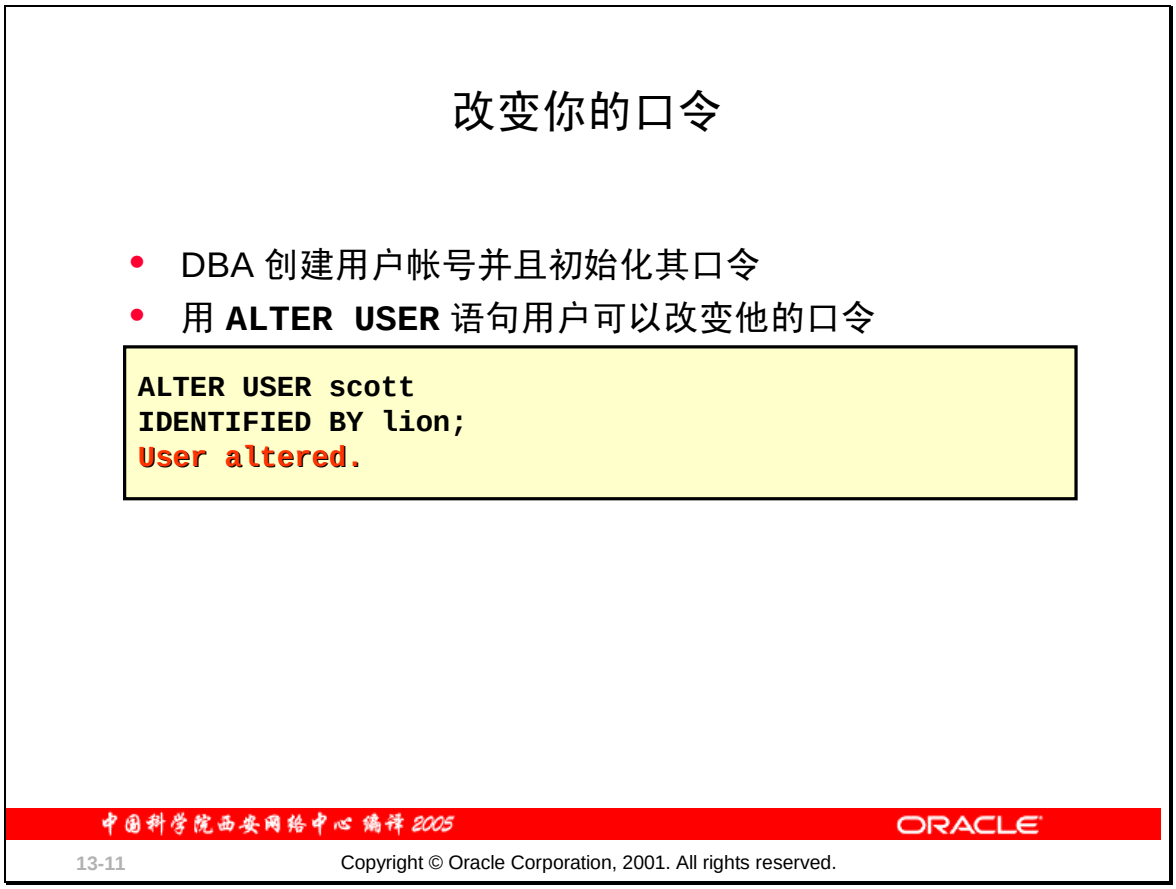


创建角色

幻灯片的例子创建一个经理角色，然后允许经理创建表和视图，然后授予该 DeHaan 和 Kochhar 经理角色，现在 DeHaan 和 Kochhar 可以创建表和视图。

如果用户被授予多个角色，他们收到所有角色的联合权限。

幻灯片 11



改变你的口令

DBA 创建一个帐号并为每个用户初始化一个口令，你可以用 ALTER USER 语句改变你的口令。

语法

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

在语法中：

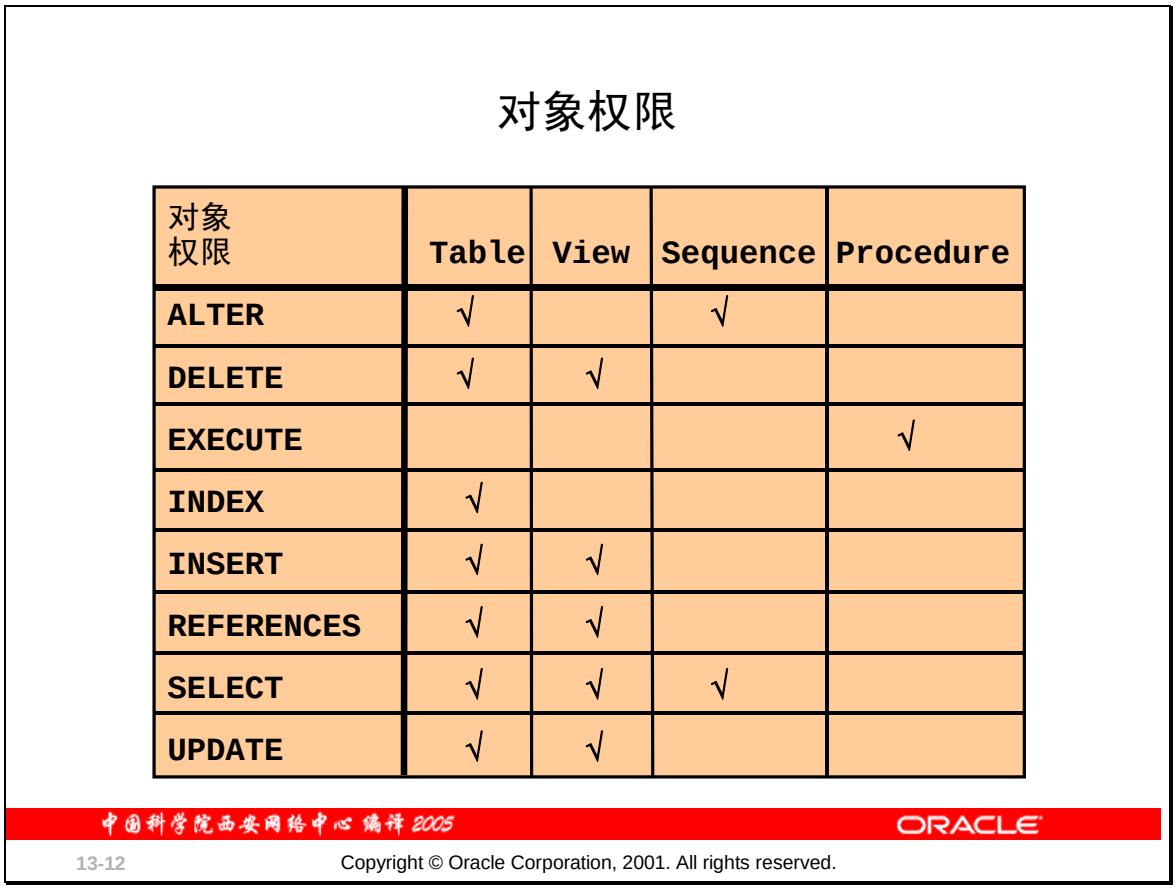
user 是用户的名字

password 指定新的口令

尽管该语句可以用于改变你的口令，还有许多其它的选择，为了改变任何其它的选项你必须有 ALTER USER 权限。

更多信息，见 *Oracle9i SQL Reference*，“修改用户”。

幻灯片 12



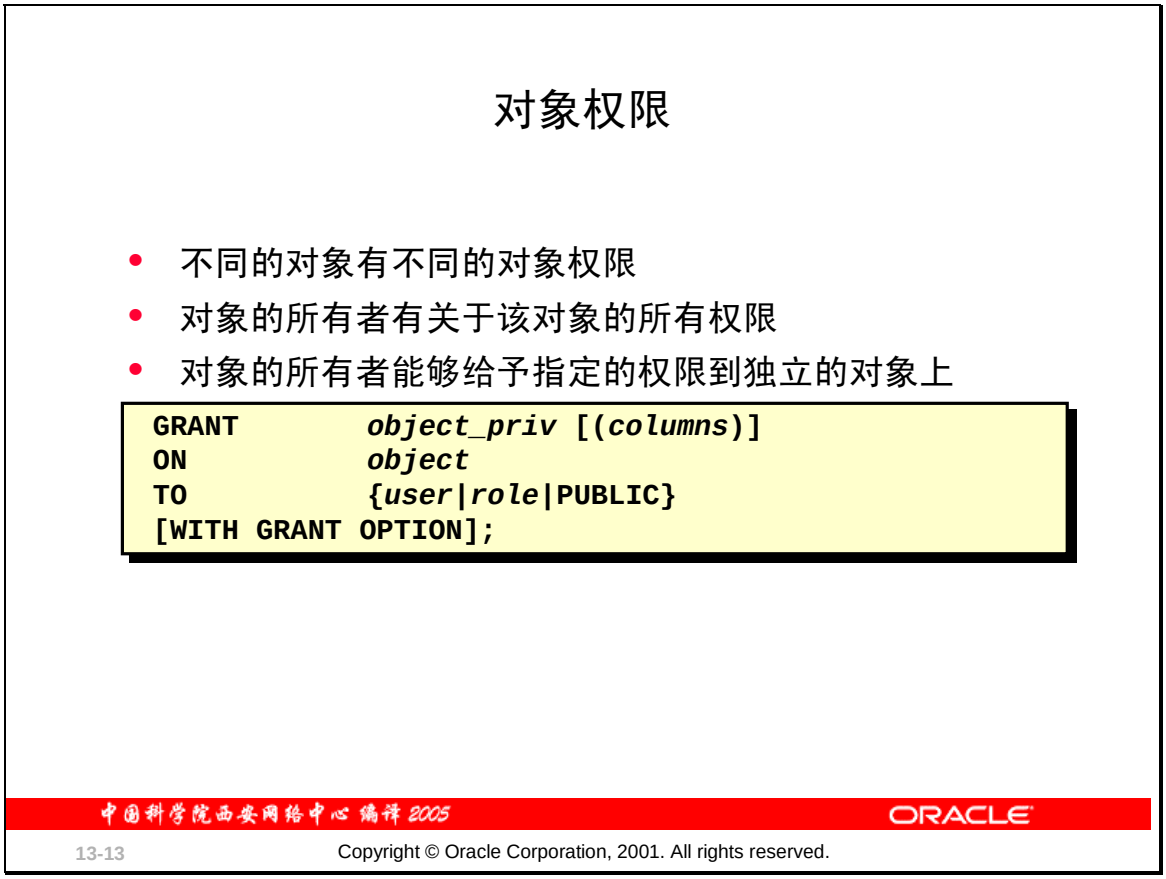
对象权限

对象权限 是在指定的表、视图、序列或过程上执行指定动作的权限或权利。每个对象都有一个特殊的可授予的权限集。幻灯片上的表列出了各种对象的权限注意，可用于序列的权限只有 SELECT 和 ALTER。UPDATE、REFERENCES 和 INSERT 权限可以通过指定一个可更新列的子集被限制，SELECT 的权限可以通过创建带一个列子集的视图并且只授予 SELECT 权限来限制。一个在同义词上被授予的权限将转换为由同义词引用的基表上的权限。

教师注释

你可以用 ALTER VIEW 和 ALTER PROCEDURE 命令重新编译视图和 PL/SQL 过程、函数和包。

幻灯片 13



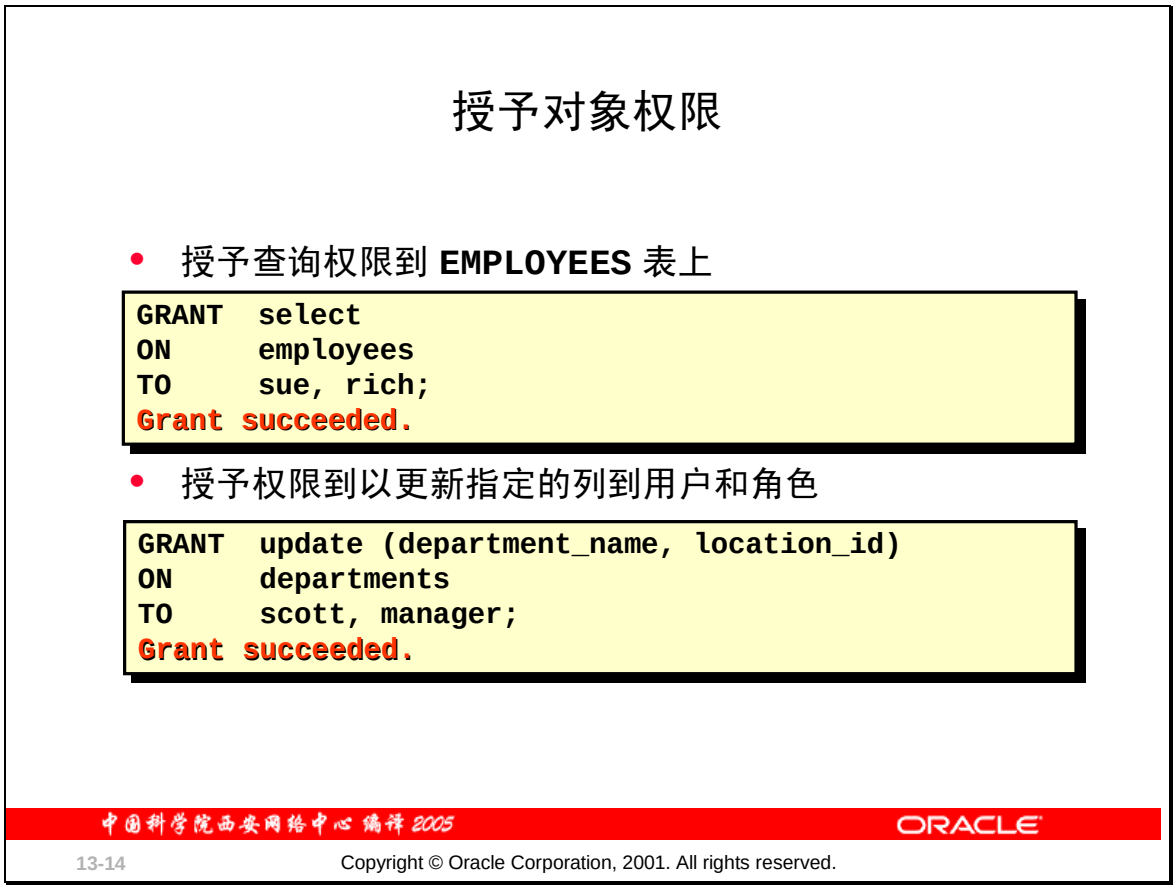
授予对象权限

不同的对象权限对于不同类型的方案对象的是有用的，一个用户自动拥有包含在该用户的方案中的所有对象权限，一个用户可以授予该用户所拥有的任何方案对象上任何对象权限给另一个用户或角色。如果授权包括 WITH GRANT OPTION 选项，那么，得到权限的用户可以再将权限授予其他的用户；否则，受让人可以使用权限，但不能授予它给其他用户。

在语法中：

object_priv	是将被授予的对象权限
ALL	指定所有对象权限
columns	从一个表或视图中指定被授予权限的列
ON *object*	是权限被授予的对象
TO	指定权限被授予谁
PUBLIC	授予权限给所有用户
WITH GRANT OPTION	允许被授予权限的人再授予对象权限给其他用户和角色

幻灯片 14



原则

- 为了授予权限到一个对象上，对象必须在你自己拥有的方案中，或者你必须被用 WITH GRANT OPTION 选项授予了对象权限。
- 一个对象所有者可以授予任何该对象上的对象权限给任何其他的用户或者数据库的角色。
- 任何对象的所有者自动地获得该对象所有对象权限。

幻灯片中第一个例子授予用户 Sue 和 Rich 查询当前用户 EMPLOYEES 表的权限。第二个例子授予在 DEPARTMENTS 表中指定列上的 UPDATE 权限给 Scott 和经理角色。

如果 Sue 或 Rich 现在想要从雇员表中 SELECT 数据，他们必须用的语法是：

```
SELECT  *
FROM    scott.employees;
```

作为选择。他们可以为该表创建一个同义词并且从同义词中选择：

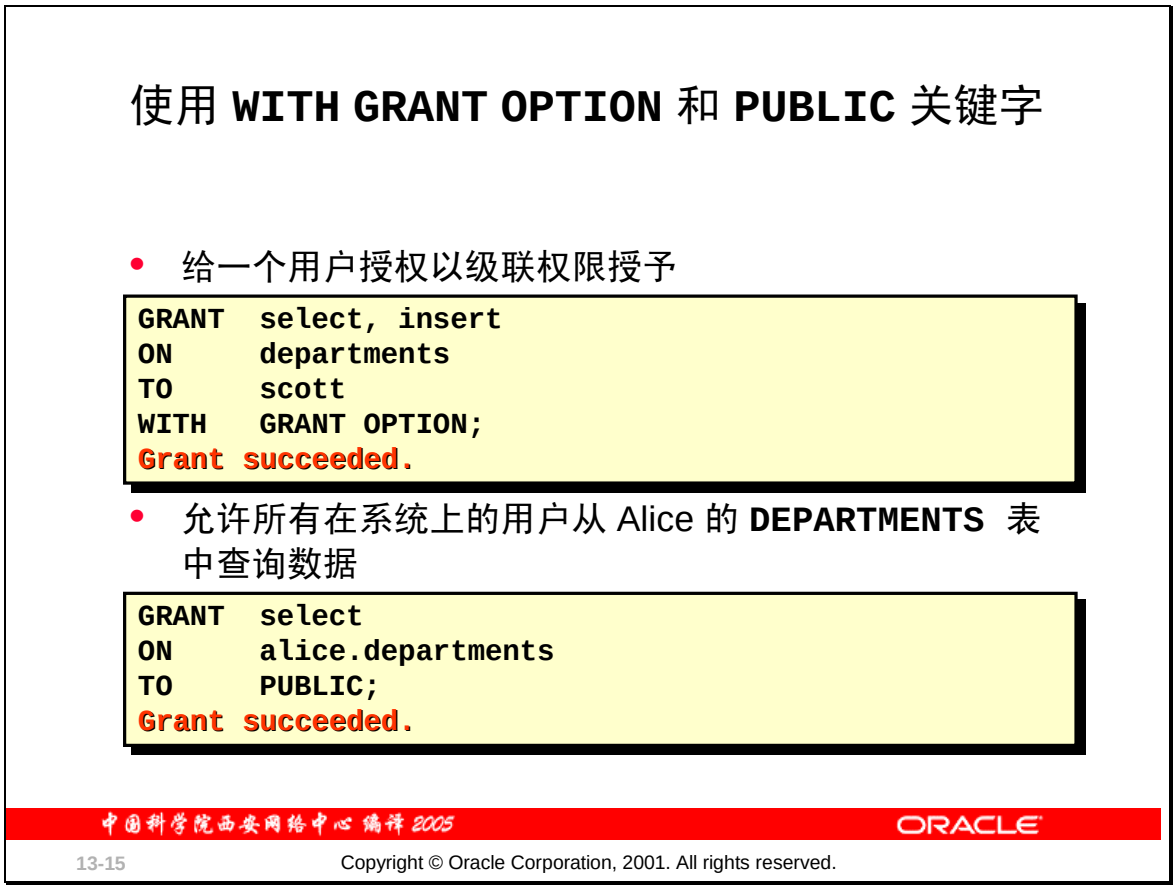
```
CREATE SYNONYM emp FOR scott.employees;
SELECT * FROM emp;
```

注：DBAs 通常分配系统权限；任何拥有对象的用户都可以授予对象权限。

教师注释

请读本课结尾的教师注释。

幻灯片 15



WITH GRANT OPTION 关键字

带 WITH GRANT OPTION 子句被授予的权限可以被受让人传递到其他的用户和角色。当授予者的权限被撤消时，用 WITH GRANT OPTION 子句授予的对象权限也被撤消。

幻灯片中例子给予用户 Scott 访问你的 DEPARTMENTS 表的权限，包括查询表和添加表中行，该例子允许 Scott 再给予其他用户这些权限。

PUBLIC 关键字

表的所有者可以用 PUBLIC 关键字给所有用户授权。

第二个例子允许所有用户从 Alice 的 DEPARTMENTS 表中查询数据。

教师注释

如果一个语句不用对象的全名，Oracle 服务器隐式地用当前的用户名 (或方案) 作为对象名的前缀。例如，如果用户 Scott 查询 DEPARTMENTS 表，系统将从 SCOTT.DEPARTMENTS 表中做选择。

如果一个语句不用对象的全名，并且当前用户不拥有该名字的对象，系统将用 PUBLIC.作为对象名的前缀。例如，如果用户 Scott 查询 USER_OBJECTS 表，但 Scott 不拥有该表，系统将用 PUBLIC.USER_OBJECTS 公共同义词的方法从数据字典视图中进行选择。

幻灯片 16

确认已授予的权限

数据字典视图	说明
ROLE_SYS_PRIVS	授予角色的系统权限
ROLE_TAB_PRIVS	授予角色的表权限
USER_ROLE_PRIVS	可由用户访问的角色
USER_TAB_PRIVS_MADE	授予用户的对象上的对象权限
USER_TAB_PRIVS_RECD	授予用户的对象权限
USER_COL_PRIVS_MADE	授予用户对象的列上的对象权限
USER_COL_PRIVS_RECD	授予用户在指定列上的对象权限
USER_SYS_PRIVS	授予用户的系统权限

确认已授予的权限

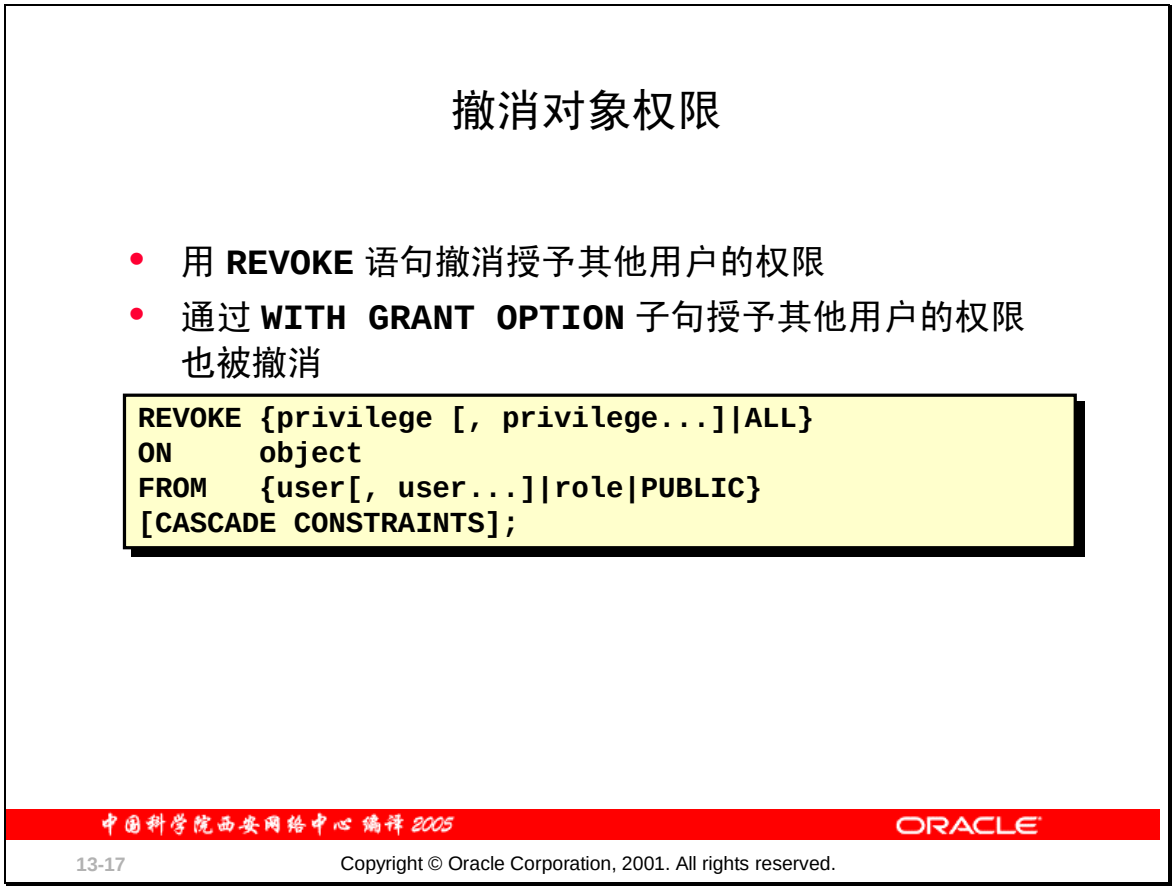
如果你试图执行一个为授权的操作，例如从你没有删除权限的表中删除行，Oracle 服务器将不允许该操作发生。

如果你收到 Oracle 服务器错误信息 “table or view does not exist,”说明发生了下面的错误：

- 指定的表或视图不存在
- 试图在一个你没有适当权限的表或视图上执行一个操作。

你可以通过访问数据字典来查看你所有的权限。在幻灯片上的表中描述了各种数据字典视图。

幻灯片 17



撤消对象权限

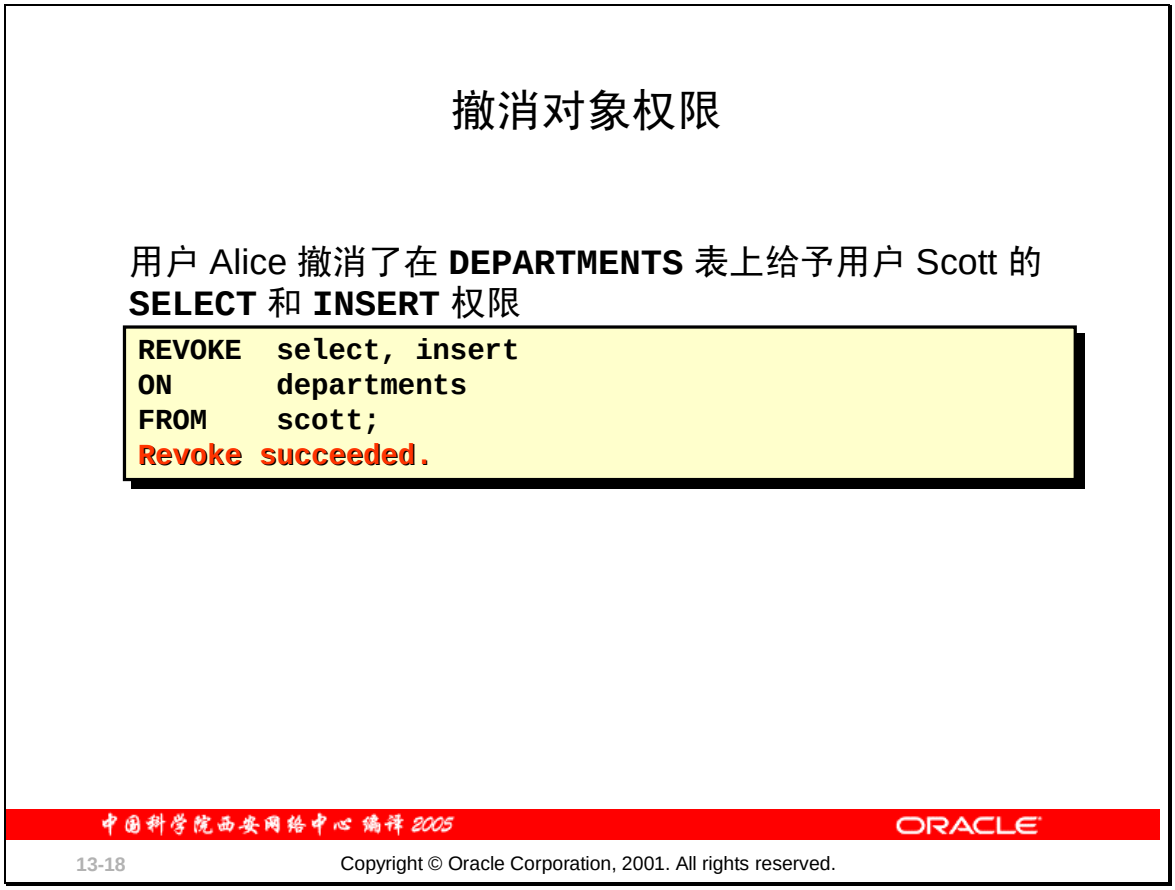
你可以用 REVOKE 语句撤消授予其他用户的对象权限。当你用 REVOKE 语句时，你指定要从用户那里撤消的权限，并且通过 WITH GRANT OPTION 子句被级联授权的那些用户的权限也将被撤消。

在语法中：

CASCADE CONSTRAINTS	用于删除任何与该对象相关的约束和对象，例如索引、触发器、权限、完整性约束等。

更多信息，见 *Oracle9i SQL Reference*，“撤消”。

幻灯片 18



撤消对象权限 (续)

幻灯片中的例子撤消给予用户 Scott 在 DEPARTMENTS 表上的 SELECT 和 INSERT 权限。

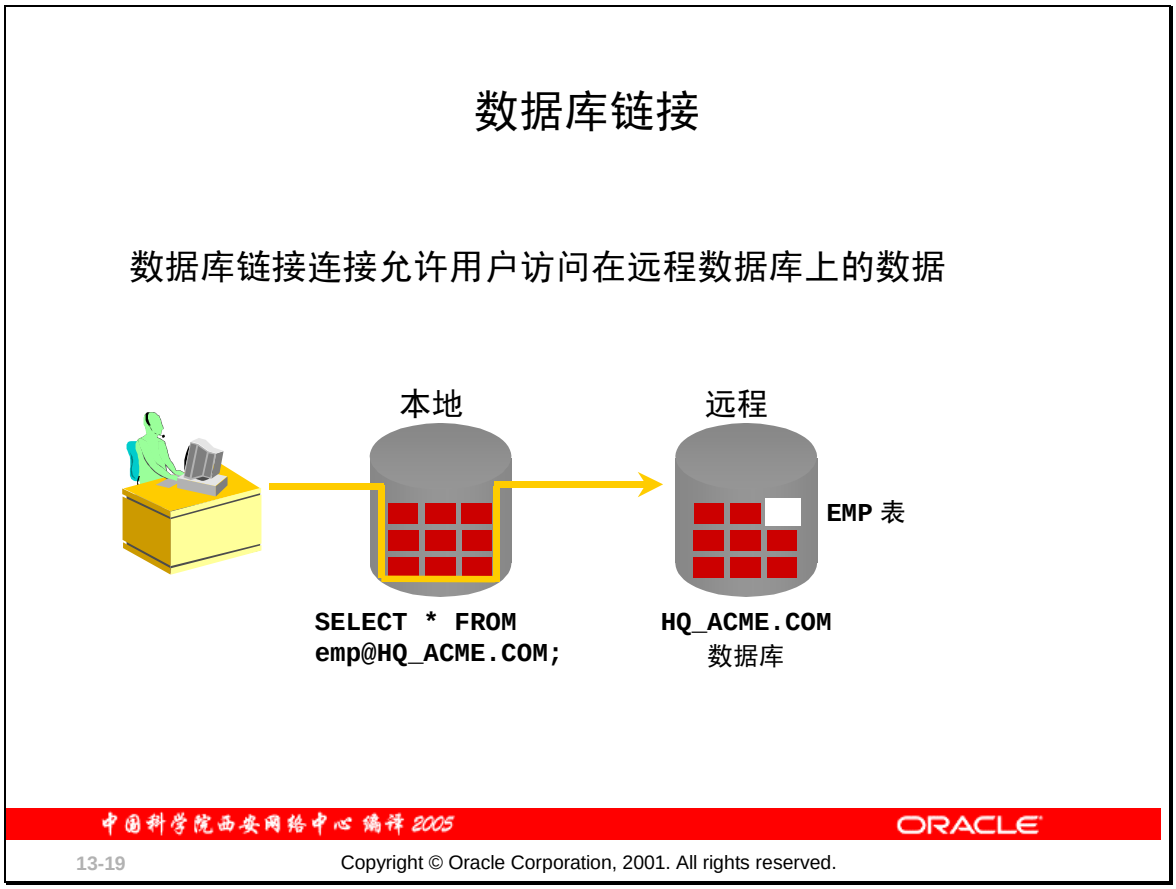
注：如果一个用户被用 WITH GRANT OPTION 子句授予权限，那么，该用户用 WITH GRANT OPTION 子句授予权限给其他用户，所以可能产生一个很长的受让人的链，但该链不允许循环。如果所有者从撤消了一个用户的权限，那么，所有授予的权限将级联地被撤消。

例如，如果用户 A 用 WITH GRANT OPTION 子句授予表上的 SELECT 权限给用户 B，用户 B 也可以再用 WITH GRANT OPTION 子句授予用户 C SELECT 权限，然后用户 C 还可以授予用户 D SELECT 权限，如果用户 A 撤消了用户 B 的权限，那么，被授予用户 C 和 D 的权限也被撤消。

教师注释

撤消系统权限的内容不在本课程的范围内，关于该主题的信息，请参考：*Oracle9i SQL Reference*，“撤消 `system_privileges_and_roles`”。

幻灯片 19



数据链接

一个数据库链接是一个指针，该指针定义一条从 Oracle 数据库服务器到另一个数据库服务器的通信路径。链接指针实际上在一个数据字典表中被定义为一个条目，为了访问该链接，你必须被连接到包含数据字典条目的本地数据库。

一个数据库链接是一条有感知的路径，一个客户可以通过该路径连接到本地数据库 A，再用存储在数据库 A 中的链接访问数据库 B 中的信息，但连接到数据库 B 的用户不能使用同一个链接来访问数据库 A 中的数据，他们必须定义一个链接，并存储在数据库 B 的数据字典中。

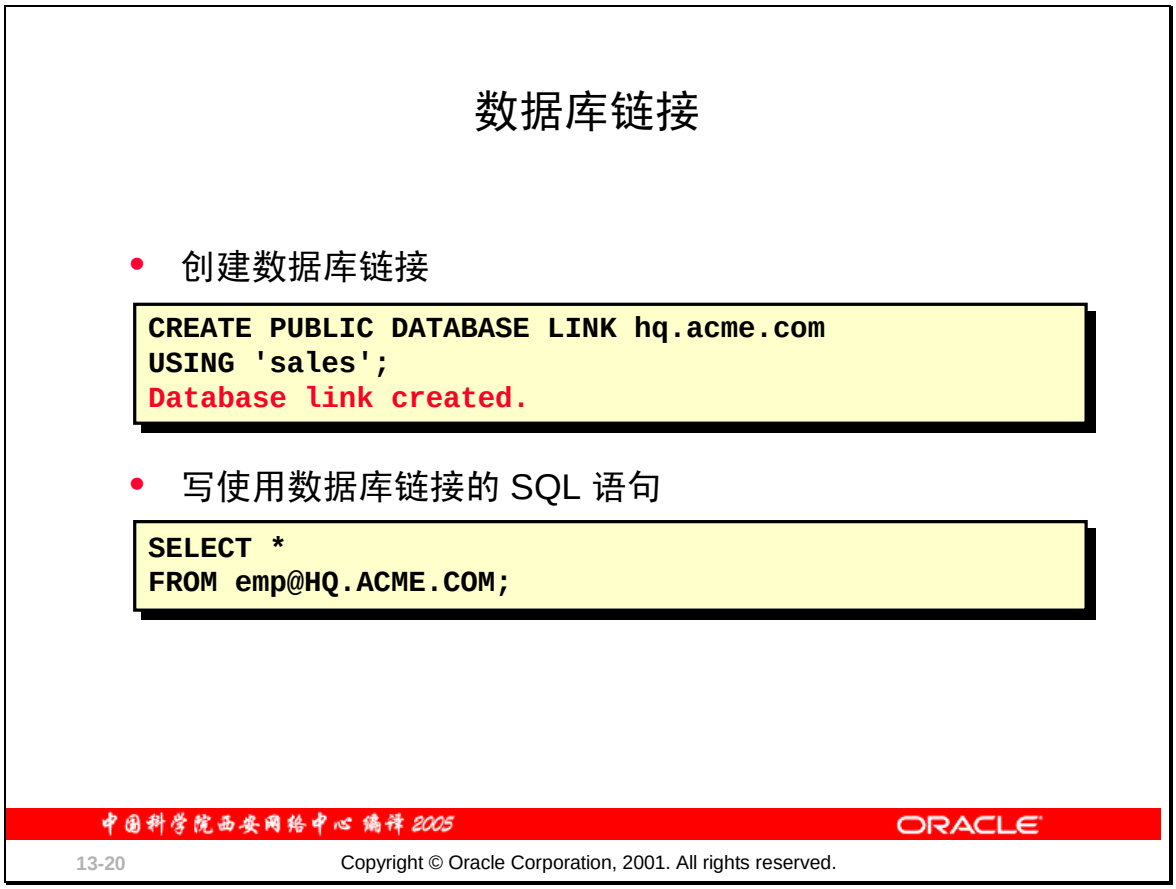
一个数据库链接的连接提供本地用户访问远程数据库上数据的能力，为了产生这种连接，在分布式计算机系统中的每个数据库必须有一个唯一的全局数据库名，全局数据库名唯一地标识一个在分布式系统中的数据库服务器。

数据库链接最大的好处是，允许用户访问在远程数据库中的另一个用户的对象，但是他们被该对象所拥有的权限集合所限制，换句话说，一个本地用户可以访问远程数据库而不需要是远程数据库上的用户。

上面的例子显示了一个用户 SCOTT 用全局名 HQ.ACME.COM 访问在远程数据库上的 EMP。

注：典型地，DBA 负责创建数据库链接。字典视图 USER_DB_LINKS 包含有关用户可以访问的链接信息。

幻灯片 20



使用数据库链接

上面的例子显示了怎样创建数据库链接。USING 子句指出了一个远程数据库的服务名。

一旦数据库连接被创建，你就可以写一个 SQL 语句访问远程站点中的数据。如果一个同义词被设置，你可以用同义词来写 SQL 语句。

例如：

```
CREATE PUBLIC SYNONYM HQ_EMP FOR emp@HQ.ACME.COM;
```

然后用该同义词写 SQL 语句：

```
SELECT * FROM HQ_EMP;
```

你不能在远程对象上授予权限。

教师注释

让学生知道使用分布式数据库要包括比这儿显示的更多的内容。如果学生想要更多信息，参考 *Oracle9i Concepts*，“分布式数据库概念”。

幻灯片 21

小结

在本课中，您应该已经学会使用 DCL 语句，DCL 语句控制访问数据库和数据对象：

语句	作用
CREATE USER	创建用户 (通常由 DBA 执行)
GRANT	给予其他用户权限来访问本用户的对象
CREATE ROLE	创建一个权限的集合 (通常由 DBA 执行)
ALTER USER	改变用户口令
REVOKE	从删除在用户对象上的权限

小结

- DBAs 用指定权限给用户来为用户建立初始数据库安全。
- DBA 创建的用户必须有一个口令。DBA 也负责为用户建立初始系统权限。
- 一旦用户已经创建了一个对象，用户可以用 GRANT 语句传递任何可用的对象权限给另一个用户或所有用户。
- DBA 可以用 CREATE ROLE 语句创建角色来传递系统权限或对象权限的集合给多个用户。角色使得授予或撤消权限更容易。
- 用户可以 ALTER USER 语句改变他们的口令。
- 你可以用 REVOKE 语句删除用户的权限。
- 用数据字典视图，用户可以查看已授给他们的权限，并且那些权限被授予在他们的哪些对象上。
- 用数据库链接，你可以访问在远程数据库上的数据。权限不能被授予远程对象。

幻灯片 22

练习 13 概览

本章练习包括下面的主题：

- 授予其他用户权限以访问你的表
- 授予你权限，修改另一个用户的表
- 创建同义词
- 查询有关权限的数据字典视图

练习 13 概览

与其他的学生合作做这些练习，这些练习是控制数据库对象访问的。

教师注释

为了做这些练习，将学生划分为不同的组，然后将再成对划分这些组，所以一半是第一组，另一半是第二组。

幻灯片 23

练习 13

1. 用户应该被授予什么权限才能登录 Oracle 服务器？是系统权限还是对象权限？

The `CREATE SESSION` system privilege

2. 用户应该被授予什么权限才能够创建表?

The `CREATE TABLE` privilege

3. 如果你创建表，谁可以传递在你的表上的权限给其他用户?

You can, or anyone you have given those privileges to by using the `WITH GRANT OPTION`.

4. 如果你是 DBA，你正在创建许多用户，他们需要相同的系统权限，你什么办法才能使你的工作更容易一些?

Create a role containing the system privileges and grant the role to the users

5. 你用什么命令改变你的口令?

The `ALTER USER` statement

6. 授权另一个用户访问你的 DEPARTMENTS 表，有用户授权你访问他的 DEPARTMENTS 表。

第二组执行 GRANT 语句。

```
GRANT select
ON departments
TO <user1>;
```

第一组执行 GRANT 语句。

```
GRANT select
ON departments
TO <user2>;
```

在这里 *user1 是第一组的名字，user2 是第二组的名字。*

7. 查询你的 DEPARTMENTS 表中所有的行。

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting		1700

8 rows selected.

```
SELECT *
FROM departments;
```

8. 添加一个新行到你的 DEPARTMENTS 表中。第一组将添加 Education 作为部门，部门号为 200。第二组将添加 Human Resources 部门，部门号为 210。查询其它组的表。

第一组执行下面的 INSERT 语句。

```
INSERT INTO departments(department_id, department_name)
VALUES (200, 'Education');
COMMIT;
```

第二组执行下面的 INSERT 语句。

```
INSERT INTO departments(department_id, department_name)
VALUES (210, 'Human Resources');
COMMIT;
```

9. 创建一个同义词，用于其他组的 DEPARTMENTS 表。

第一组创建一个同义词，并命名为 *team2*。

```
CREATE SYNONYM team2
FOR <user2>.DEPARTMENTS;
```

第二组创建一个同义词，并命名为 team1。

```
CREATE SYNONYM team1
FOR <user1>. DEPARTMENTS;
```

10. 用你的同义词查询在其他组中的 DEPARTMENTS 表的所有行。

第一组 SELECT 语句的结果：

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
510	Human Resources		
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting		1700

9 rows selected.

第二组 SELECT 语句的结果：

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
500	Education		
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting		1700

9 rows selected.

第一组执行下面的 SELECT 语句。

```
SELECT *
FROM team2;
```

第二组执行下面的 SELECT 语句。

```
SELECT *
FROM team1;
```

11. 查询 USER_TABLES 数据字典，查看你所拥有的表的有关信息。

TABLE_NAME
COUNTRIES
DEPARTMENTS
EMPLOYEES
JOBS
JOB_GRADES
JOB_HISTORY
LOCATIONS
REGIONS

8 rows selected.

```
SELECT table_name
FROM user_tables;
```

12. 查询 ALL_TABLES 数据字典视图。查看你能够访问，但你不是所有者的所有的表的有关信息。

注：你的列表可能不能精确匹配下面显示的列表。

TABLE_NAME	OWNER
DEPARTMENTS	*owner*

```
SELECT table_name, owner
FROM all_tables
WHERE owner <> <your account>;
```

13. 从其他组撤消你的表上的 SELECT 权限。

第一组撤消权限。

```
REVOKE select
ON departments
FROM user2;
```

第二组撤消权限。

```
REVOKE select
ON departments
FROM user1;
```

14. 删除你在第 8 题中插入到 DEPARTMENTS 表中的行，保存改变。

幻灯片 26

教师注释 (对 13-14 页)

让学生知道 *privilege(col,col)*语法只能被用于 UPDATE，大多数学生用带 SELECT 的该语法，如下面所显示：

```
GRANT SELECT(salary,last_name)
ON employees TO scott;
```

上面的语法将返回错误：`ERROR at line 1:ORA-00969: missing ON keyword.`

教师注释

让学生知道有关细粒度访问控制，使用细粒度访问控制，你可以用函数实现安全策略，然后将他们的安全策略与表或视图结合起来。无论怎样访问数据 (例如，特别查询)，数据库服务器都自动强制那些安全策略。

你可以：

- 用对 ELECT、INSERT、UPDATE 和 DELETE 命令用不同的策略。
- 只在需要他们的地方用安全策略 (例如，关于销售信息)
- 可以对每个表使用多个策略，包括在打包的应用程序中建立于基础策略之上的策略

至于细粒度访问控制的实现，你可能需要使用在 PL/SQL 中的函数或包，PL/SQL DBMS_RLS 包使你能够管理你的安全策略，用这个包，你可以添加、删除启用、禁用和刷新你所创建的策略。有关实现细粒度访问控制的更多信息，参考：*Oracle9i Concepts*，“细粒度访问控制”。