数据库的安全性主要通过用户、权限和角色进行管理。

1 用户管理

1.1 数据库用户账户

要访问数据库，用户必须指定有效的数据库用户账户，而且还要根据该用户账户的要求成功通过验证，每个数据库用户都有一个唯一的数据库账户。Oracle建议这样做是为了避免潜在的安全漏洞以及为特定的审计活动提供有意义的数据。但是，有时候若干用户会共享一个公用数据库账户，每个用户账户都包括以下项：

- 唯一的用户名：用户名不能超过30个字节，不能包含特殊字符，而且必须以字母开头；
- 验证方法：最常见的验证方法是口令，但是Oracle 12c支持口令、全局和外部验证方法；
- 默认表空间：如果用户未指定其他表空间，则可在默认表空间创建对象。请注意，具有默认表空间并不意味着用户在该表空间具有创建对象的权限，也不意味这用户在该表空间中具有用于创建对象的空间限额，这两项需要另外授权；
- 临时表空间：这是实例代表用户创建临时对象（如排序和临时表）的位置，临时表空间没有限额；
- 用户概要文件：分配给用户的一组资源和口令限制；
- 初始使用者组：有资源管理器使用；
- 账户状态：用户只可访问“打开”的账户。

方案：方案是数据库用户拥有的数据库对象的集合。方案对象是直接引用数据库数据的逻辑结构，方案对象包括表、视图、序列、存储过程、同义词，索引和数据库链接等结构。通常，方案对象包括应用程序在数据库中创建的所有内容。

1.2 预定义用户账号

安装Oracle数据库时，安装进程会在数据库中自动创建一系列预定义的管理、非管理以及示例方案用户账号，分别为：

- 预定义的管理员账号；
- 预定义的非管理员用户账号；

- 预定义的示例模式用户账号；

1.2.1 管理员账号

管理员账号有SYS，SYSTEM，SYSBACKUP，SYSDG，SYSKM，SYSMAN以及DBSNMP等。

1.2.1.1 SYS

据库数据字典的所有基表和视图都存储在SYS用户中，这些基表和视图对于数据库的操作非常重要。为了维护数据字典的完整性，SYS用户下的表只由数据库操作，任何用户或数据库管理员都不应该修改它们，而且任何人都不应该在SYS用户中创建任何表（但是，如果需要，可以更改数据字典设置的存储参数）。

1.2.1.2 SYSTEM

SYSTEM用户用于创建展示管理信息的附加表和视图，以及各种Oracle数据库选项和工具使用的内部表和视图，同样，不要在SYSTEM用户下创建存放普通用户的数据表。

1.2.1.3 SYSBACKUP

主要用于使用RMAN或SQL*Plus进行数据库的备份和恢复操作。

1.2.1.4 SYSDG

主要用于进行Data Guard操作。

1.2.1.5 SYSKM

主要用于透明的数据加密密钥存储操作。

1.2.1.6 SYSMAN

主要用于执行Oracle Enterprise Manager Cloud Control（OEMCC）管理任务。

1.2.1.7 DBSNMP

Oracle Enterprise Manager Cloud Control（OEMCC）的管理代理使用该账户进行监视和管理数据库。

1.2.2 非管理员用户账号

非管理员的用户账号有DIP，ORACLE_OCM等。

1.2.3 示例模式用户账号

示例模式的用户账号有HR，OE，PM，SH，SCOTT等。

1.3 管理权限授权的操作

默认情况下，sys和system账户被授予DBA角色，另外，sys账户还具有带admin option的所有权限并拥有数据字典。要连接到sys账户，对于数据库实例，必须使用as sysdba子句，对于ASM实例，必须使用as sysasm子句。授予了sysdba权限的任何用户都可以通过使用as sysdba子句连接到sys账户。仅允许授予了sysdba、sysoper或sysasm权限的特权用户启动和关闭实例。system账户不具有sysdba权限，system还被授予aq_administrator_role和mgmt_user角色。sys和system账户是数据库所必须的账户，不能将其删除。

Oracle Enterprise Manager的管理代理使用dbsnmp账户来监视和管理数据库，sysman账户用于执行OEM管理任务，dbsnmp和sysman都没有sysdba权限。

Administrative Privilege	Operations Authorized
SYSDBA	• Perform STARTUP and SHUTDOWN operations • ALTER DATABASE: open, mount, back up, or change character set • CREATE DATABASE • DROP DATABASE • CREATE SPFILE • ALTER DATABASE ARCHIVELOG • ALTER DATABASE RECOVER • Includes the RESTRICTED SESSION privilege This administrative privilege allows most operations, including the ability to view user data. It is the most powerful administrative privilege.
SYSOPER	• Perform STARTUP and SHUTDOWN operations • CREATE SPFILE • ALTER DATABASE: open, mount, or back up • ALTER DATABASE ARCHIVELOG • ALTER DATABASE RECOVER (Complete recovery only. Any form of incomplete recovery, such as UNTIL TIME | CHANGE | CANCEL | CONTROLFILE requires connecting as SYSDBA.) • Includes the RESTRICTED SESSION privilege This privilege allows a user to perform basic operational tasks, but without the ability to view user data.

1.4 概要文件

概要文件是用于限制数据库使用和实例资源的一组资源限制条件。通过概要文件还可以管理账户状态并对用户的口令进行限制（长度、到期时间等）。每个用户都分配有一个概要文件，而且该用户在指定时间只属于一个概要文件。如果在更改用户概要文件时用户已经登录，则所做的更改在用户下一次登录时才生效。

Default概要文件用作其他所有概要文件的基础。只有当resource_limit初始化参数设置为true，概要文件才能对用户强制实行资源限制，如果该值为false，则忽略概要文件的资源限制，概要文件的口令设置始终会强制执行。

管理员使用概要文件可控制系统资源和密码安全功能：

- CPU：可按会话或调用限制CPU资源。将cpu/session（CPU/会话）限制为1000表示，如果使用此概要文件的任一会话占用10秒以上的CPU时间（CPU时间限制以百分之一秒为单位），该会话就会收到错误消息并被注销：ORA-02392：exceeded session limit on CPU usage,you are being logged off，对每个调用所做的限制也起相同作用，但它不是限制用户的整个会话，而是防止任一命令占用过多cpu。如果cpu/call（CPU调用）受到限制，并且用户超出了该限制，则命令会终止，用户将收到如下错误消息：ORA-02393：exceeded call limit on CPU usage .
- 网络/内存：每个数据库会话都会占用系统内存资源和网络资源（如果会话不是来自服务器的本地用户），可以指定以下参数：

- 连接时间：指示用户在自动注销前可以保持连接的分钟数；
- 空闲时间：指示用户会话在自动注销前可以保持空闲的分钟数。只会计算服务器进程的空闲时间，空闲时间不考虑应用程序活动，Idle_time限制不受长时间进行的查询和其他操作的影响；
- 并行会话：指示使用数据库用户账户可以创建多少并行会话；
- 专用SGA：限制在系统全局区SGA中执行排序、合并位图等操作所占用的空间量，此限制尽在会话使用共享服务器时才生效；

- 磁盘I/O：限制用户在每个会话级或每个调用级可读取的数据量。“读取/会话”和“读取/调用”可限制内存和磁盘的总读取次数，这样做可确保执行大量I/O操作的语句不会过度使用内存和磁盘；
- 账户锁定：如果用户在指定的次数内尝试登录系统失败，系统会在设置的持续时间内自动锁定账户：
  - failed_login_attempts：指定在锁定账户前尝试登录的失败次数；
  - password_lock_time：指定尝试登录失败达到了指定的次数后锁定账户的天数；
- 口令失效和到期：使用口令具有生存期，口令在此生存期后会到期，必须对其进行更改；
  - password_life_time：确定口令生存期（天），之后该口令就会到期；
  - password_grace_time：指定首次成功登录后更改口令的宽限期（天），之后该口令就会到期；
- 口令历史记录：通过核对新口令可确保在指定的时间内或在指定的口令更改次数内不重复使用口令；
  - password_reuse_time：指定用户不能在指定天数内重复使用口令；
  - password_reuse_max：指定在可重复使用当前口令之前口令更改需达到的次数；
- 口令复杂性验证：通过对口令进行复杂性检查可验证口令是否符合特定的规则，这种检查一定要确保口令足够复杂，才能防止入侵者通过猜测口令尝试闯入系统。password_verify_function函数指定一个PL/SQL函数，以便在分配口令之前指定口

令复杂性检查。口令验证函数必须由sys用户拥有，而且必须返回布尔值（TRUE或FALSE），位于以下目录中的utlpwdmg.sql脚本提供了模型口令验证函数：

- Unix和Linux平台：$ORACLE_HOME/rdbms/admin；
- Windows平台：%ORACLE_HOME%\rdbms\admin。

1.5 空间限额

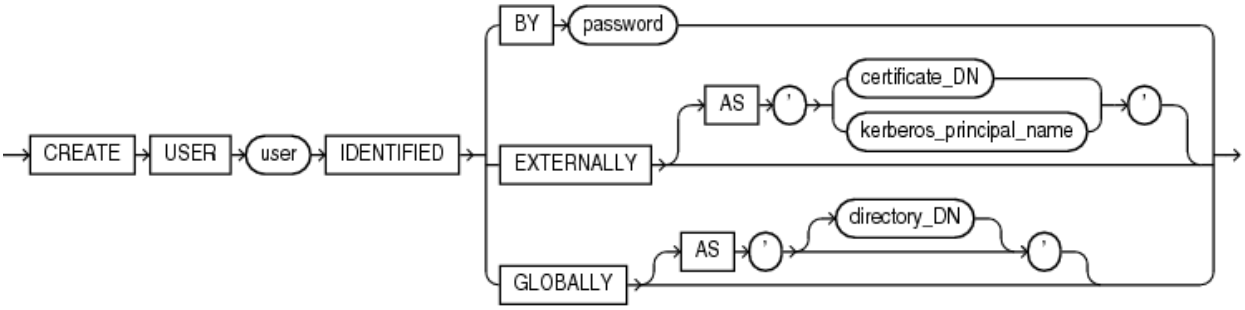
限额是允许给特定表空间具有的空间，默认情况下，对于任何表空间用户都没有限额，可使用以下三个选项为用户提供表空间限额：

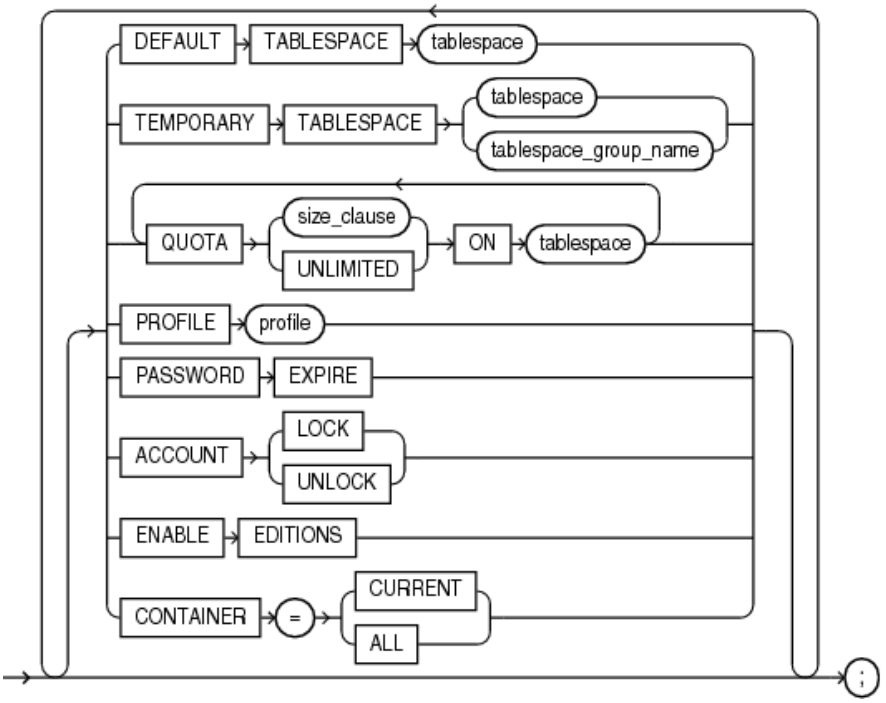
- 无：允许用户最大限度地使用表空间中的可用空间；
- 值：用户可以使用的空间（以KB或MB为单位），这并不保证一定会为用户保留该空间，因为，此值可能大于或小于表空间中的当前可用空间；
- unlimited tablespace系统权限：覆盖所有单独的表空间配额，对于所有表空间（包括system和sysaux），为用户提供无限制的限额，授予此权限时必须谨慎。

1.6 创建用户

给每一个用户分配默认表空间和临时表空间，如果用户在创建对象时未指定表空间，则将在分配给对象所有者的默认表空间中创建对象，这样，可以控制用户对象的创建位置，如果为选择默认表空间，则使用系统定义的默认永久表空间，对于临时表空间也是如此，如果未指定表空间，则使用系统定义的临时表空间。

1.6.1 创建用户语法





1.6.2 创建用户

创建和配置数据库用户使用CREATE USER语法，要创建数据库用户，必须具有CREATE USER系统权限，创建完数据库用户后，该用户权限域为空，欲登录数据库，用户必须具有CREATE SESSION权限。

1.6.2.1 创建用户演示一

1）管理员身份登录

[oracle@strong ~]$ sqlplus /nolog

SQL*Plus: Release 12.1.0.2.0 Production on Tue Jul 23 21:34:24 2019

Copyright (c) 1982, 2014, Oracle. All rights reserved.

SQL> conn / as sysdba

Connected.
SQL>
2）创建stu用户
SQL> create user stu identified by stu;

User created.
3）验证新创建用户stu
SQL> conn stu/stu@192.168.56.102:1521/orcl
ERROR:
ORA-01045: user STU lacks CREATE SESSION privilege; logon denied
SQL> grant create session to stu;

Grant succeeded.

SQL> conn stu/stu@192.168.56.102:1521/orcl
Connected.

1.6.2.2 创建用户演示二

1）管理员身份登录
SQL> conn / as sysdba
Connected.
SQL>
2）创建新用户u01，并授权
SQL> create user u01 identified by u01 default tablespace users temporary tablespace temp quota 5M on users password expire;

User created.

SQL> grant create session to u01;

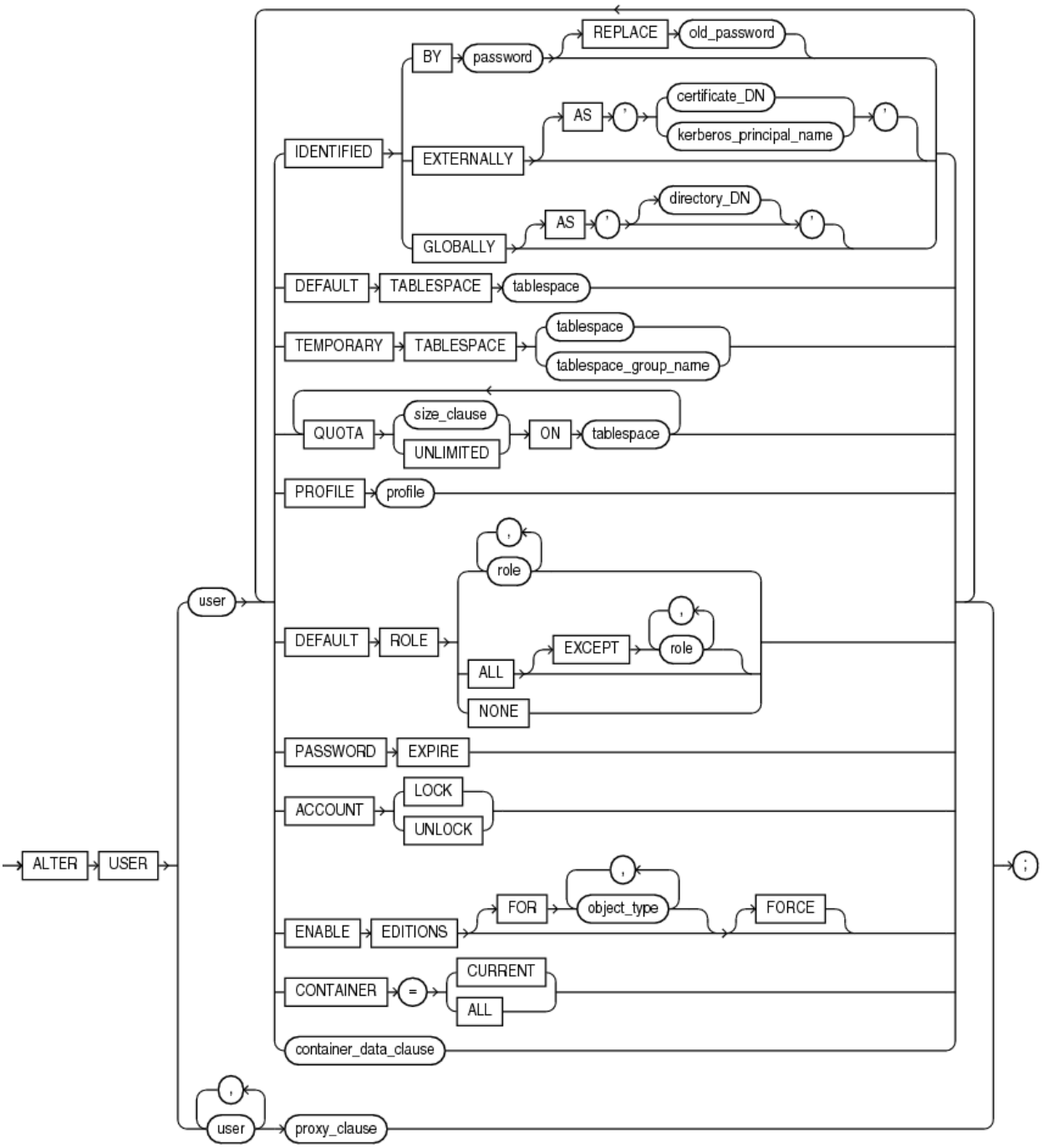
Grant succeeded.
3）验证新创建用户u01
SQL> conn u01/u01@192.168.56.102:1521/orcl
ERROR:

ORA-28001: the password has expired

Changing password for u01
New password:
Retype new password:
Password changed
Connected.

1.7 修改用户

1.7.1 修改用户语法



1.7.2 修改用户演示

1）修改用户密码

SQL> alter user stu identified by stu;

User altered.

2）修改用户默认表空间

SQL> alter user stu default tablespace test;

User altered.

3）强制用户下次登录时修改密码

SQL> alter user stu password expire;

User altered.

4）锁定用户

SQL> alter user stu account lock;

User altered.

5）解锁用户

SQL> alter user stu account unlock;

User altered.

1.8 监视用户

1.8.1 查询用户的默认表空间

```
SQL> col username for a30
SQL> col account_status for a20
SQL> col default_tablespace for a30
SQL> col temporary_tablespace for a30
SQL> col profile for a30
SQL> SELECT
t.username,t.account_status,t.default_tablespace,t.temporary_tablespace,t.profile
FROM Dba_Users t WHERE t.username='STU';
```

```
USERNAME                       ACCOUNT_STATUS      DEFAULT_TABLESPACE
TEMPORARY_TABLESPACE       PROFILE
------------------------------ ------------------- ------------------------------ ----------
------------------- ------------------------------
STU                            OPEN                USERS                          TEMP
DEFAULT

SQL>
```

1.8.2 查询用户的空间配额

```
SQL> SELECT t.tablespace_name,t.username,t.bytes,t.max_bytes FROM
dba_ts_quotas t WHERE t.username='U01';

TABLESPACE_NAME                USERNAME                       BYTES  MAX_BYTES
------------------------------ ------------------------------ ---------- ----------
USERS                          U01                                 0    5242880
```

1.8.3 查询用户会话信息

```
SQL> col machine for a30
SQL> col program for a50
SQL> SELECT t.SID,t.SERIAL#,t.MACHINE,t.PROGRAM FROM v$session t WHERE
t.USERNAME='STU';

       SID    SERIAL# MACHINE                        PROGRAM
---------- ---------- ------------------------------ --------------------------------------------------
----------
        61      30704 strong.oracle.com              sqlplus@strong.oracle.com (TNS V1-V3)
```

1.8.4 删除用户会话信息

```
SQL> alter system kill session '61,30704';
```

System altered.

SQL> SELECT t.SID,t.SERIAL#,t.MACHINE,t.PROGRAM,t.status FROM v$session t WHERE t.USERNAME='STU';

```
       SID    SERIAL# MACHINE                        PROGRAM
STATUS
---------- ---------- ------------------------------ ----------------------------------------
---------- --------
        61     30704 strong.oracle.com              sqlplus@strong.oracle.com (TNS V1-V3)
KILLED
```

1.9 删除用户

1.9.1 用户包含对象

```
SQL> drop user stu;
drop user stu
*
ERROR at line 1:
ORA-01922: CASCADE must be specified to drop 'STU'
```

> 用户包含对象，删除用户须加CASCADE。

1.9.2 用户不包含对象

SQL> drop user u01;

User dropped.

2 权限管理

权限是用户执行特定类型的SQL语句或访问其他用户的对象的一组权限，Oracle DB允许控制用户在数据库中能够（或无法）执行的操作。权限可分为系统权限和对象权限。

> 参考： https://docs.oracle.com/database/121/SQLRF/statements_9014.htm#SQLRF01603

2.1 系统权限

每种系统权限都允许用户执行一个特定的数据库操作或一类数据库操作。例如，创建表空间的权限就是一种系统权限。系统权限可由管理员授予，或者由被显式授权管理权限的用户授予，有170多种不同的系统权限，很多系统权限都包含any子句；

请在授予系统权限之前仔细考虑安全要求，因为某些系统权限通常只授予给管理员：

- restricted session：使用这个权限可以登录，即使数据库是在受限模式打开的；
- sysdba和sysoper：使用这两个权限可以在数据库中执行关闭、启动、恢复及其他管理任务。用户使用sysoper可执行基本操作任务，但不能查看用户数据。这个权限包括以下系统权限：
    - startup和shutdown；
    - create spfile；
    - alter database open/mount/backup；
    - alter database archivelog；
    - alter database recover（仅限完全恢复。任何形式的不完全恢复，如until time|change|cancel|controlfile，都需要以sysdba身份建立连接）；
    - restricted session；
    - 除此之外，sysdba系统权限还允许执行不完全恢复和删除数据库，用户使用sysdba系统权限可以sys用户身份有效地建立连接；
- sysasm：使用此权限可以启动、关闭和管理asm实例；
- drop any object：用户使用drop any权限可删除其他用户拥有的对象；
- create、manage、drop和alter tablespace：这些权限允许进行表空间管理，包括创建、删除和更改表空间的属性；
- create library：Oracle DB允许开发人员在PL/SQL内创建和调用外部代码（例如C库），词库必须由数据库中的library对象指定。create library权限允许用户创建可以从PL/SQL执行的任意代码库；
- create any directory：作为一种安全措施，代码所在的操作系统目录必须链接到一个虚拟Oracle目录对象，使用create any directory权限时，有可能会调用不安全的代码对象。用户使用create any directory权限可以在Oracle软件所有者能够访

问的任何目录中创建目录对象（具有读写访问权限），这意味着用户可以访问这些目录中的外部过程。用户可以尝试直接读写任何数据库文件，如数据文件、重做日志和审计日志。确保您的组织采用了安全策略，以防止此类强大的权限被误用；

- grant any object privilege：使用此权限可以对其他人拥有的对象授予对象权限；
- alter database和alter system：这些权限的功能很强，可用于修改数据库和Oracle实例，例如，重命名数据文件或刷新缓冲区高速缓存。

2.2 对象权限

用户可以使用对象权限对特定对象（如表、视图、序列、过程、函数或程序包）执行特定的操作。在没有特定权限的情况下，用户只能访问他们自己拥有的对象，对象权限可以由对象的所有者或管理员授予，也可以由被显式授予了权限，可以为其他人员分配对某个对象的权限的人员授予；

2.3 权限授予

授予系统权限的SQL语法：grant <system_privilege> to <grantee clause> [with admin option];
授予对象权限的SQL语法：grant <object_privilege> on <object> to <grantee clause> [with grant option]。

2.4 权限撤销

撤销系统权限的SQL语法：revoke <system_privilege> from <grantee clause>
撤销对象权限的SQL语法：revoke <object_privilege> on <object> from <grantee clause>

> 撤销系统权限不会产生级联影响，撤销与数据操纵语言（DDL）或带有grant option的对象权限时会产生级联影响。
> 参考：https://docs.oracle.com/database/121/SQLRF/statements_9022.htm#SQLRF01609

2.5 权限授权和撤销演示

2.5.1 对象权限授权和撤销演示

场景：

1）DBA将带有admin option的create table系统权限授予Joe；

2）Joe创建表；

3）Joe将create table系统权限授予Emily；

4）Emily创建表；

5）DBA撤销Joe的create table系统权限。

结果：

Joe的表仍旧存在，但Joe不能再创建新表，Emily的表仍旧存在，而且她仍旧具有create table系统权限。

2.5.2 对象权限授权和撤销演示

场景：

1）Joe被授予了对employees的SELECT对象权限（带grant option）；

2）Joe将对employees的SELECT权限授予Emily；

3）撤销对Joe的SELECT权限；

结果：

撤销级联到Emily。

3 角色管理

3.1 角色概述

角色是一组权限的集合，将角色赋给一个用户或其他角色，这个用户和其他角色就拥有了这个角色中的所有权限。

3.2 角色的优点

使用角色有以下优点：

- 简化权限管理：使用角色可简化权限管理，可以将一组权限授予给某个角色，然后将该角色授予给每个用户，而不是将同一组权限授予给多个用户；
- 进行动态权限管理：如果修改了与某个角色关联的权限，则所有被授予该角色的用户都会立即自动获得修改后的权限；

- 有选择地提供权限：通过启动或禁用角色可以暂时打开或关闭权限，这样便可以在指定情形下控制用户的权限。

3.3 角色特性

在大多数系统中，将必要的权限分别授予给每个用户是一项很耗时的工作，而且很容易出错，Oracle软件通过角色提供了简单且受控的权限管理方式，角色是可授予给用户或其他角色的、由相关权限组成的指定组。设计角色是为了简化数据库中的权限管理，从而增强数据库的安全性。角色有以下特性：

- 角色就像用户，可以授予角色权限或撤销角色权限；
- 角色就像系统权限一样，可以将其授予给授予用户或其它角色，也可以从用户或其它角色撤销；
- 角色可以由系统权限和对象权限组成；
- 可以对授予了某一角色的每个用户启用或禁用该角色；
- 可能需要口令才能启用角色；
- 角色不归由任何用户拥有，也不属于任何方案。

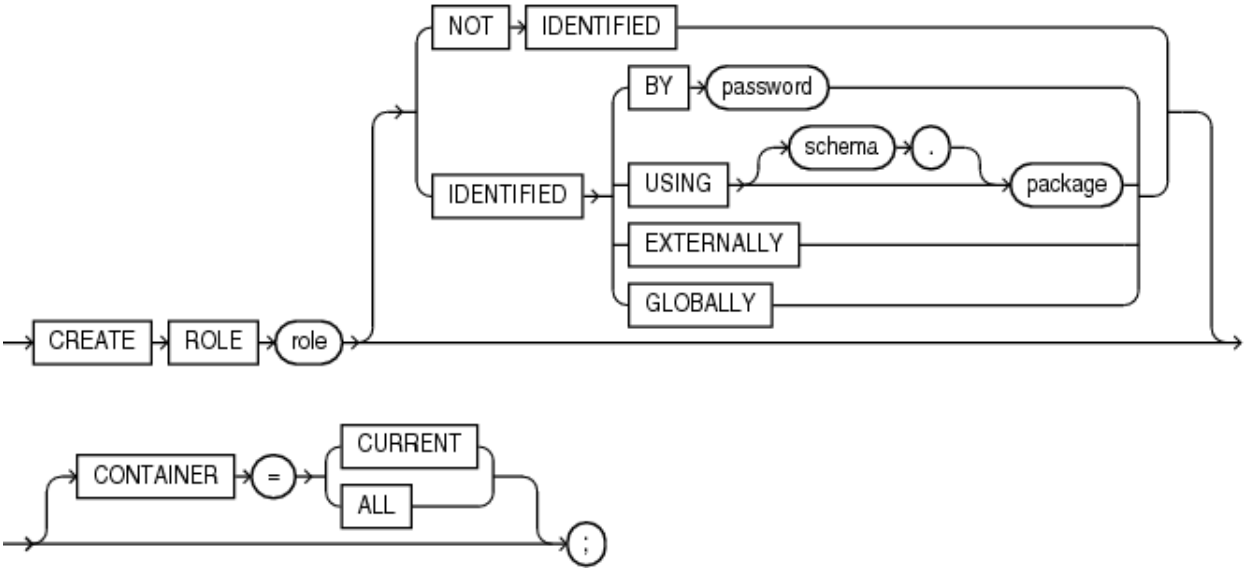
3.4 预定义角色

预定义角色是在数据库安装后，系统自动创建的一些常用的角色，主要看以下预定义角色及其包括的权限：

角色	权限
CONNECT	CREATE SESSION
RESOURCE	CREATE CLUSTER、CREATE PROCEDURE、CREATE SEQUENCE、CREATE OPERATOR、CREATE TRIGGER、CREATE TABLE、CREATE INDEXTYPE、CREATE TYPE
SCHEDULER_ADMIN	CREATE EXTERNAL JOB、EXECUTE ANY CLASS、EXECUTE ANY PROGRAM、CREATE ANY JOB、CREATE JOB、MANAGE SCHEDULER
DBA	大多数系统权限：几个其他角色，不要授予非管理员
SELECT_CATALOG_ROLE	无系统权限：HS_ADMIN_ROLE以及对数据字典的1700多个对象权限；

3.5 角色管理

3.5.1 创建角色语法



3.5.2 角色管理

1）建一个角色

SQL> create role clerk;

Role created.

2）授权给角色

SQL> grant connect,create table,create view,create type to clerk;

Grant succeeded.

3）授予角色给用户

SQL> grant clerk to stu;

Grant succeeded.

4）查看角色所包含的权限

```
SQL> col role for a30
SQL> select * from role_sys_privs t WHERE t.role='CLERK';

ROLE                           PRIVILEGE                                ADM COM INH
------------------------------ ---------------------------------------- --- --- ---
CLERK                          CREATE TABLE                             NO  NO  NO
CLERK                          CREATE TYPE                              NO  NO  NO
CLERK                          CREATE VIEW                              NO  NO  NO
```

5）创建带有口令以角色(在生效带有口令的角色时必须提供口令：在对应的账户下生效)

```
SQL> create role role1 identified by role1;

Role created.
```

6）修改角色：是否需要口令

```
sql>alter role role1 not identified;
sql>alter role role1 identified by password1;
```

7）设置当前用户要生效的角色

角色的生效是一个什么概念呢？假设用户a有b1,b2,b3三个角色，那么如果b1未生效，则b1所包含的权限对于a来讲是不拥有的，只有角色生效了，角色内的权限才作用于用户，最大可生效角色数由参数MAX_ENABLED_ROLES设定；在用户登录后，oracle将所有直接赋给用户的权限和用户默认角色中的权限赋给用户。

```
sql>set role role1;//使role1生效
sql>set role role,role2;//使role1,role2生效
sql>set role role1 identified by password1;//使用带有口令的role1生效
sql>set role all;//使用该用户的所有角色生效
sql>set role none;//设置所有角色失效
sql>set role all except role1;//除role1外的该用户的所有其它角色生效。
sql>select * from SESSION_ROLES;//查看当前用户的生效的角色。
```

8）修改指定用户，设置其默认角色

```
sql>alter user user1 default role role1;
sql>alter user user1 default role all except role1;
```

9）删除角色

```
SQL> drop role role1;

Role dropped.
```

角色删除后，原来拥用该角色的用户就不再拥有该角色了，相应的权限也就没有了。

4 数据字典视图

4.1 用户相关的数据字典视图

- dba_users：用户信息；
- dba_ts_quotas：表空间配额信息；
- v$session：会话信息；
- v$open_cursor：用户SQL语句；

4.2 权限相关的数据字典视图

- dba_role_privs：授予的角色权限；
- dba_sys_privs：授予的系统权限；
- dba_tab_privs：授予的对象权限；
- dba_col_privs：授予的列权限；
- role_sys_privs：角色对应的权限；

4.3 角色相关的数据字典视图

- role_sys_privs：角色对应的权限信息（预定义角色）；
- role_tab_privs：角色对应的对象信息；
- dba_roles：定义的角色信息；
- dba_role_privs：授予的角色权限；

4.4 查看某个用户所有的权限和角色

```
select privilege
  from dba_sys_privs
 where grantee = 'SCOTT'
union
select privilege
  from dba_tab_privs
 where grantee = 'SCOTT'
union
select privilege
  from dba_sys_privs
 where grantee in
       (select granted_role from dba_role_privs where grantee = 'SCOTT')
union
select privilege
  from dba_tab_privs
 where grantee in
       (select granted_role from dba_role_privs where grantee = 'SCOTT');
```