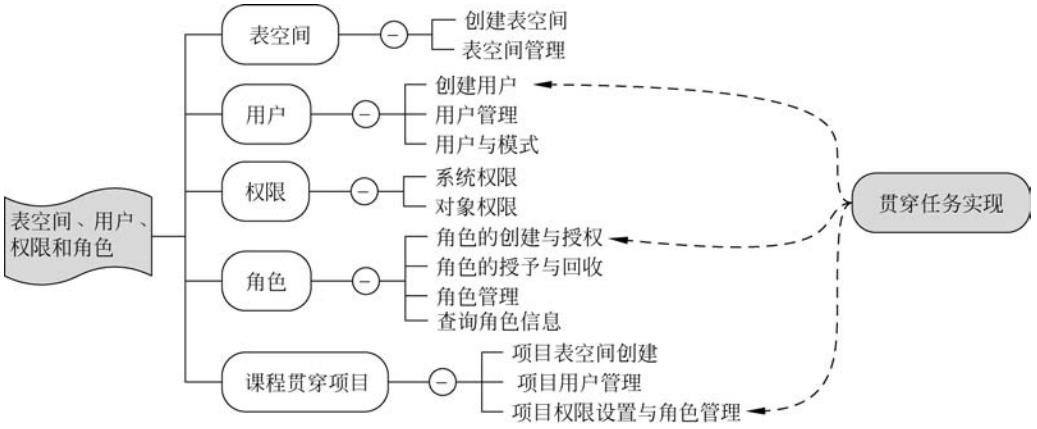
# 第3章 表空间、用户、权限和角色

## 任务驱动

本章任务完成 Q_MicroChat 微聊项目表空间创建、用户管理和权限设置及角色管理。具体任务分解如下：

- 【任务 3-1】 项目表空间创建
- 【任务 3-2】 项目用户管理
- 【任务 3-3】 项目权限设置及角色管理

## 学习导航/课程定位




## 本章目标

| 知　识　点 | Listen(听) | Know(懂) | Do(做) | Revise(复习) | Master(精通) |
|---|---|---|---|---|---|
| 表空间的作用 | ★ | ★ | | | |
| 表空间的创建及管理 | ★ | ★ | ★ | ★ | ★ |
| 用户的作用 | ★ | ★ | | | |
| 用户的创建及管理 | ★ | ★ | ★ | ★ | ★ |
| 系统权限的作用与使用 | ★ | ★ | ★ | ★ | ★ |
| 对象权限的作用与使用 | ★ | ★ | ★ | ★ | ★ |
| 角色的作用与使用 | ★ | ★ | ★ | ★ | ★ |

# 3.1 表空间

表空间是 Oracle 逻辑存储结构中数据的逻辑组织形式，Oracle 使用表空间将各种数据库对象组合在一起，当用户对数据库对象进行操作时，面向的是逻辑对象，而非直接操作物理文件，极大地方便了数据库操作。

在 Oracle 中，可以通过查询数据字典 DBA_TABLESPACES 了解系统已存在的表空间的详细信息，如下述示例所示。

**【示例】 查询系统中所有表空间的信息。**

```
SQL>  SELECT tablespace_name,status,extent_management,
  2  allocation_type,segment_space_management,contents
  3  FROM dba_tablespaces;
TABLESPACE_NAME   STATUS   EXTENT_MANAGEMENT  ALLOCATION_TYPE   SEGMENT_SPAC   CONTENTS
------------  -------  ---------------  --------------  -----------  --------
SYSTEM            ONLINE   LOCAL              SYSTEM            MANUAL         PERMANENT
SYSAUX            ONLINE   LOCAL              SYSTEM            AUTO           PERMANENT
UNDOTBS1          ONLINE   LOCAL              SYSTEM            MANUAL         UNDO
TEMP              ONLINE   LOCAL              UNIFORM           MANUAL         TEMPORARY
USERS             ONLINE   LOCAL              SYSTEM            AUTO           PERMANENT
```

其中：

- tablespace_name 表示表空间的名称。
- status 表示表空间的状态。
- extent_management 表示表空间的管理方式。
- allocation_type 表示表空间中区的分配方式。
- segment_space_management 表示表空间中段的管理方式。
- contents 表示表空间的类型。

在 Oracle 数据库创建时，系统会自动创建 SYSTEM(系统表空间)、SYSAUX(辅助系统表空间)、TEMP(临时表空间)、UNDOTBS1(撤销表空间)、USERS(用户表空间)，其中，SYSTEM 表空间主要用于存储数据字典信息、数据库对象定义等信息，用户对象尽量不要存放在此表空间，以免影响数据库的稳定性与执行效率；SYSAUX 表空间是 Oracle 10g 开始新增的辅助系统表空间，主要用于存储数据库组件信息、示例数据库对象信息等，以减少 SYSTEM 表空间的负荷；TEMP 是 Oracle 数据库专门进行临时数据管理的表空间，只存储临时数据，临时数据在会话结束时自动释放；UNDOTBS1 是 Oracle 数据库专门用于回退信息管理的表空间；USERS 是专用于存储用户所创建的数据库对象的表空间，也是整个数据库的默认表空间，即如果某个普通用户(非系统用户)创建时没有为其分配表空间，则默认使用 USERS 表空间。

## 3.1.1 创建表空间

在数据库应用中对于表空间的使用，Oracle 数据库建议每个应用分别对应一个独立的

表空间，所有用户对象和数据保存在非系统表空间中，这样不仅能够分离不同应用的数据，而且能够减少读取数据文件时产生的 I/O 冲突。

根据表空间的功能特性，一个表空间的创建，应该需要确定表空间的名称、表空间的管理方式、表空间的类型、表空间的数据文件、区的分配方式、段的管理方式、表空间数据块大小等参数。Oracle 数据库表空间创建脚本的基本语法如下。

**【语法】**

```
CREATE [ TEMPORARY | UNDO ] TABLESPACE tablespace_name
DATAFILE datafile_spacification
[ BLOCKSIZE number K ]
[ ONLINE | OFFLINE ]
[ LOGGING | NOLOGGING ]
[ FORCE LOGGING ]
[ COMPRESS | NOCOMPRESS ]
[ EXTENT MANAGEMENT DICTIONARY | LOCAL [AUTOALLOCATE | UNIFORM SIZE number K|M]]
[ SEGMENT SPACE MANAGEMENT AUTO | MANUAL ];
```

其中：

- TEMPORARY|UNDO 表示创建的表空间类型，分为永久表空间(PERMANENT)、临时表空间(TEMPORARY)和撤销表空间(UNDO)，默认为永久表空间。
- DATAFILE 用于指定表空间所对应的数据文件的信息，具体参数在下方单独介绍。
- BLOCKSIZE 表示表空间所基于的数据块大小，默认为标准块表空间。
- ONLINE|OFFLINE 表示新建表空间处于脱机状态还是联机状态。
- LOGGING|NOLOGGING 表示指定所有保存在该表空间中的默认日志选项，其中 LOGGING 表示数据库对象的创建以及数据的 DML 操作信息都写入重做日志文件，NOLOGGING 表示数据的加载操作不写入重做日志文件。
- FORCE LOGGING 表示表空间中所有对象发生的变化信息都将被写入重做日志文件，忽略 NOLOGGING 子句的作用。
- COMPRESS|NOCOMPRESS 表示是否将数据块中的数据进行压缩，COMPRESS 表示压缩，压缩的结果是消去列中的重复值，当检索数据时，Oracle 会自动对数据解压缩；NOCOMPRESS 表示不执行压缩。
- EXTENT MANAGEMENT 用于指定表空间的管理方式，包括字典管理方式(DICTIONARY)和本地管理方式(LOCAL)，默认为 LOCAL；本地管理表空间中区的分配方式默认为自动分配，用参数 AUTOALLOCATE 表示；参数 UNIFORM 表示定制分配。
- SEGMENT SPACE MANAGEMENT 表示本地管理表空间中段的管理方式，默认为自动管理方式 AUTO。

上述语法中，除数据文件配置外，各可选参数采用其默认值便可以完成一个本地管理的永久表空间的创建。其中，数据文件的详细配置语法如下。

**【语法】**

```
CREATE TABLESPACE tablespace_name
DATAFILE path/file_name SIZE number K|M [REUSE]
```

```
[ AUTOEXTEND OFF|ON
    [ NEXT number K|M MAXSIZE UNLIMITED|number K|M ]
];
```

其中：

- tablespace_name 为创建的表空间名称。
- path/file_name 为所创建的表空间实际存储在磁盘中的数据文件地址。
- SIZE 指定数据文件初始大小。
- REUSE 表示如果该数据文件已经存在，则清除该文件并重新创建；如果未使用该关键字，则当数据文件已经存在时将出错。
- AUTOEXTEND 指定数据文件是否可以自动扩展。如果可以自动扩展，需要设置 NEXT 值指明每次扩展的大小，设置 MAXSIZE 值指明文件的最终大小。默认值为 OFF，不可扩展。

根据 CREATE TABLESPACE 命令参数的默认值，创建一个本地管理的永久表空间 qstspace，区采用自动分配方式，段采用自动管理方式，如下述示例所示。

**【示例】 创建名为 qstspace 的本地管理的永久表空间。**

```
SQL> CREATE TABLESPACE qstspace
  2  DATAFILE 'e:\oracle12c\userspace\tbs.dbf' SIZE 30M;
表空间已创建。
```

创建一个本地管理的永久表空间 qstspace2，区采用定制配置方式，区的大小为 512KB，段采用自动管理方式。如下述示例所示。

**【示例】 创建名为 qstspace2 的本地管理的永久表空间。**

```
SQL> CREATE TABLESPACE qstspace2
  2  DATAFILE 'e:\oracle12c\userspace\tbs2.dbf' SIZE 30M
  3  EXTENT MANAGEMENT LOCAL UNIFORM SIZE 512K;
表空间已创建。
```

表空间设计理念灵活性的一个方面在于数据文件的可扩展性。当存储在某个数据文件中的数据量超过了初始大小时，数据文件可以进行自动扩展，同时为避免频繁扩展影响数据库性能，应该设定一个合理的增长幅度，并且为防止无限制的增长带来硬盘空间耗尽的风险，还应为每个表空间的数据文件设定最大尺寸。如下述示例所示。

**【示例】 指定数据文件的可扩展性。**

```
SQL> CREATE TABLESPACE qstspace3
  2  DATAFILE 'e:\oracle12c\userspace\tbs3.dbf' SIZE 30M
  3  AUTOEXTEND ON NEXT 5M  MAXSIZE 500M;
表空间已创建。
```

一个表空间也可以对应多个数据文件，为一个表空间创建多个数据文件需要指定多个数据文件的完整路径和详细的选项参数，各数据文件之间用逗号分隔，如下述示例所示。

【示例】 为一个表空间创建多个数据文件。

```
SQL > CREATE TABLESPACE qstspace4
  2   DATAFILE 'e:\oracle12c\userspace\data_1.dbf' size 20M,
  3   'e:\oracle12c\userspace\data_2.dbf' size 5M;
表空间已创建。
```

注意

表空间的创建还有很多其他选项，由于本书不过多涉及数据库管理知识，此处不再一一列举，读者可根据需求查看 Oracle 帮助文档进行详细了解。

### 3.1.2 表空间管理

表空间创建完成后，可以对表空间进行管理和维护，包括改变表空间大小、设置默认表空间、重命名表空间、删除表空间等。

#### 1. 改变表空间大小

由于表空间的大小是由其所拥有的数据文件的数量及大小决定的，因此通过为表空间添加数据文件、改变现有数据文件的大小、改变数据文件的扩展方式，都可以达到改变表空间大小的目的。

ALTER TABLESPACE...ADD DATAFILE 命令可以为永久表空间添加数据文件，如下述示例所示。

【示例】 为 qstspace 表空间添加一个大小为 10MB 的新的数据文件。

```
SQL > ALTER TABLESPACE qstspace
  2   ADD DATAFILE 'e:\oracle12c\userspace\tbs_1.dbf' SIZE 10M;
表空间已更改。
```

通过 ALTER DATABASE DATAFILE...RESIZE 命令可以改变表空间已有数据文件的大小，如下述示例所示。

【示例】 将 qstspace 表空间的数据文件 tbs_1.dbf 的大小增加到 20MB。

```
SQL > ALTER DATABASE DATAFILE 'e:\oracle12c\userspace\tbs_1.dbf' RESIZE 20M;
数据库已更改。
```

在创建表空间或为表空间添加数据文件时如果没有指定 AUTOEXTEND 选项，则该数据文件的大小默认是固定的，通过 ALTER DATABASE DATAFILE ...AUTOEXTEND 命令可以将数据文件设置为可以自动扩展。

【示例】 将 qstspace 表空间的数据文件 tbs_1.dbf 设置为自动扩展。

```
SQL > ALTER DATABASE DATAFILE 'e:\oracle12c\userspace\tbs_1.dbf'
  2   AUTOEXTEND ON NEXT 5M MAXSIZE 100M;
数据库已更改。
```

### 2. 设置默认表空间

默认表空间是相对用户来说的，每个登录 Oracle 数据库的用户都有一个默认表空间，如果在创建用户时未为该用户显示指定默认表空间，则将统一使用数据库的默认表空间 USERS。通过查询数据字典 DATABASE_PROPERTIES 可以获取当前数据库的默认永久表空间，如下述示例所示。

**【示例】 查询数据库默认表空间。**

```
SQL> SELECT property_name,property_value FROM database_properties
  2  WHERE property_name = 'DEFAULT_PERMANENT_TABLESPACE';
PROPERTY_NAME                          PROPERTY_VALUE
------------------------------    ---------------
DEFAULT_PERMANENT_TABLESPACE           USERS
```

如果需要将数据库的默认表空间指定为其他，则可以使用 ALTER DATABASE DEFAULT TABLESPACE 命令进行重设。如下述示例所示。

**【示例】 修改数据库默认表空间为 qstspace。**

```
SQL> ALTER DATABASE DEFAULT TABLESPACE qstspace;
数据库已更改。
```

重设之后，无论是新建的用户还是已有用户，创建时如果未显示指定默认表空间，则将使用重设后的新的数据库默认表空间。

### 3. 重命名表空间

对于用户创建的表空间，可以使用 ALTER TABLESPACE...RENAME TO 命令对表空间的名称进行修改。

**【示例】 修改数据库表空间 qstspace 的名称为 new_qstspace。**

```
SQL> ALTER TABLESPACE qstspace RENAME TO new_qstspace;
表空间已更改。
```

当重命名一个表空间时，数据库会自动更新数据字典、控制文件以及数据文件对该表空间的引用。重命名表空间时，该表空间 ID 号并没有修改，因此如果该表空间是数据库的默认表空间，那么重命名后仍然是数据库的默认表空间。另外，处于脱机状态、只读状态以及数据库的系统表空间 SYSTEM 和 SYSAUX 不能重命名。

### 4. 删除表空间

对于用户创建的某个表空间，如果不再有存在的必要，则可以使用 DROP TABLESPACE 命令将该表空间从数据库中删除，从而释放磁盘空间。

删除表空间有两种方式，一种是仅删除其在数据库中的记录(数据字典和控制文件中与该表空间相关的信息)；另一种是将记录和数据文件一起删除。

**【示例】 仅删除表空间在数据库中的记录。**

```
SQL> DROP TABLESPACE qstspace4;
表空间已删除。
```

上述示例在数据库中成功删除了表空间 qstspace4 的信息，但是查看其对应的数据文件，会发现仍然存在于磁盘中。若想在删除表空间时同时删除数据文件，需要添加 INCLUDING 参数，如下述示例所示。

**【示例】 删除表空间及其数据文件。**

```
SQL > DROP TABLESPACE qstspace3 INCLUDING CONTENTS AND DATAFILES;
表空间已删除。
```

表空间一旦被删除，该表空间中的所有数据也将永久性丢失，因此在删除前一定要确认该表空间的数据不再使用。Oracle 建议在删除之前和之后分别对数据库进行一次完全备份，这样即使是误删除了表空间或删除表空间后数据库无法正常运行，也可以通过备份恢复数据库。此外，需要注意的是，数据库的默认表空间不能删除，如果要删除，需要重新指定数据库的默认表空间，然后才可以删除原有的默认表空间。

## 3.2 用户

用户是数据库中最基本的对象之一，Oracle 中的用户可以分为两类：一类是创建数据库时系统预定义的用户，也称为系统用户；另一类是根据应用需要由系统用户创建的用户。在创建 Oracle 数据库时，系统用户根据作用不同又可分为以下 3 类。

- 管理员用户：包括 SYS、SYSTEM 等。其中 SYS 是数据库中具有最高权限的数据库管理员，拥有数据字典，可以创建、启动、修改和关闭数据库；SYSTEM 是一个辅助的数据库管理员，主要负责一些管理工作，如创建用户、删除用户等。
- 示例方案用户：在安装 Oracle 数据库软件时创建数据库或利用 DBCA 创建数据库时，如果选择了“示例方案”，则在数据库中会创建一些用户以及这些用户对应的数据库应用案例，如 HR、SCOTT、BI、SH 等。默认情况下，这些用户的状态为账户锁定、口令过期。
- 内置用户：有一些 Oracle 数据特性或 Oracle 组件需要自己单独的模式，因此为它们创建了一些内置用户，如 APEX_PUBLIC_USER、DIP 等非管理员内置用户。默认情况下，这些用户的状态为账户锁定、口令过期。

通过查询数据字典 DBA_USERS 可以了解当前数据库的所有用户信息，如下述示例所示。

**【示例】 查看当前数据库所有用户信息。**

```
SQL > SELECT username,account_status,default_tablespace FROM dba_users;
USERNAME                  ACCOUNT_STATUS                 DEFAULT_TABLESPACE
 ----------------    ------------------------    --------------------
SYSTEM                    OPEN                           SYSTEM
SYS                       OPEN                           SYSTEM
DBSNMP                    EXPIRED & LOCKED               SYSAUX
SCOTT                     EXPIRED & LOCKED               USERS
ORACLE_OCM                EXPIRED & LOCKED               USERS
OJVMSYS                   EXPIRED & LOCKED               USERS
SYSKM                     EXPIRED & LOCKED               USERS
…
```

其中：

- USERNAME 列标识用户的登录名。
- ACCOUNT_STATUS 列标识账号的当前状态，处于 OPEN 状态的账号为可用账号，处于 EXPIERD&LOCKED 状态的账号是过期和锁定账号，此状态的用户不能登录数据库。
- DEFAULT_TABLESPACE 列标识用户的默认表空间。

### 3.2.1 创建用户

Oracle 的管理员用户拥有数据库大多数对象的操作权限，然而在正式开发过程中，使用该用户是不安全的，一旦操作失误，将有可能对数据库造成严重损害。因此，在实际应用中，可以通过管理员用户创建一个新用户，然后通过设置该用户的权限控制用户对数据库的访问和操作。

在 Oracle 数据库中，使用 CREATE USER 命令创建新用户，执行该语句的用户必须具有 CREATE USER 权限。CREATE USER 命令语法如下：

**【语法】**

```
CREATE USER user_name
IDENTIFIED BY password
[DEFAULT TABLESPACE default_tablespace]
[TEMPORARY TABLESPACE temp_tablespace]
[PASSWORD EXPIRE]
[ACCOUNT LOCK|UNLOCK];
```

其中：

- user_name 指定要创建的数据库用户名。
- IDENTIFIED BY password 指定用户采用的数据库身份认证，口令为 password，口令由大小写字母、数字混合组成，总长度大于等于 8 个字符。
- DEFAULT TABLESPACE 指定用户的默认表空间，如果用户没有指定该表空间，则系统将使用数据库默认表空间存储。
- TEMPORARY TABLESPACE 指定用户的临时表空间，如果用户没有指定该表空间，则系统将使用数据库默认的临时表空间存储。
- PASSWORD EXPIRE 指定用户口令的初始状态为过期，用户在首次登录数据库时必须修改口令。
- ACCOUNT LOCK|UNLOCK 设置用户的初始状态为锁定或不锁定，默认为不锁定。

**【示例】 创建一个名为 test 的普通用户。**

```
SQL> CREATE USER test IDENTIFIED BY TESTtest123;
用户已创建。
SQL> -- 查询用户的属性信息
SQL> SELECT username,account_status,default_tablespace,temporary_tablespace
  2  FROM dba_users WHERE lower(username) = 'test';
USERNAME        ACCOUNT_STATUS        DEFAULT_TABLESPACE       TEMPORARY_TABLESPACE
--------------- --------------------- ------------------------ ------------------------
TEST            OPEN                  USERS                    TEMP
```

通过示例运行结果可知，用户 test 已经成功创建。对于在创建时未显示指定默认表空间和临时表空间的用户，其默认表空间为 USERS，临时表空间为 TEMP。

下述示例演示在创建用户的同时，重新为其指定默认表空间。

**【示例】 创建用户的同时为其指定默认表空间。**

```
SQL> CREATE USER test02 IDENTIFIED BY TESTtest02123 DEFAULT TABLESPACE qstspace;
用户已创建。
SQL> -- 查询用户的属性信息
SQL> SELECT username,account_status,default_tablespace,temporary_tablespace
FROM dba_users WHERE lower(username) = 'test02';
USERNAME       ACCOUNT_STATUS         DEFAULT_TABLESPACE         TEMPORARY_TABLESPACE
--------   ----------------    -------------------    --------------------
TEST02             OPEN                    QSTSPACE                   TEMP
```

## 3.2.2 用户管理

用户创建完成后，可以对用户设置进行修改，包括口令、默认表空间、用户状态等，对于永久不再使用的用户，也可以进行删除。

### 1. 修改用户密码

ALTER USER...IDENTIFIED BY 命令用于修改用户密码，使用示例如下。

**【示例】 修改用户口令。**

```
SQL> ALTER USER test02 IDENTIFIED BY TESTtest2015;
用户已更改。
```

其中，TESTtest2015 为用户新设置的密码。

### 2. 修改用户默认表空间

ALTER USER...DEFAULT TABLESPACE 命令用于修改用户默认表空间，使用示例如下。

**【示例】 修改用户默认表空间。**

```
SQL> ALTER USER test02 DEFAULT TABLESPACE users;
用户已更改。
```

其中，users 为数据库默认表空间。修改完成后，此表空间也将作为用户 test02 的默认表空间。

### 3. 用户的锁定与解锁

Oracle 允许在任何时候对用户账户进行锁定与解锁。用户账户被锁定后，用户就不能再登录数据库了，但不影响其所有数据库对象的正常使用。当用户账户被解锁后，用户重新恢复正常的数据库连接和登录。通常在下列情况下可以考虑锁定用户账户，而不是删除用户账户：

- 用户需要中断工作一段时间再回来工作，此时可以临时将该用户账户锁定。
- 用户永久性离开，但其拥有的数据库对象仍然被其他用户引用，此时为避免其他用户的数据库对象失效，可以将该用户永久锁定，而不是删除该用户。
- 在应用程序开发过程中使用的一些数据库账户，在系统开发完成后，这些账户不再使用，应该将其锁定。
- 在数据库中有一些 Oracle 的内置账户，其所拥有的数据库对象对数据库特定功能特性、特定组件提供支持，应该将其锁定。

ALTER USER...ACCOUNT LOCK|UNLOCK 命令用于对用户账户进行锁定或解锁，如下述示例所示。

**【示例】 用户账户的锁定与解锁。**

```
SQL> ALTER USER test ACCOUNT LOCK;
用户已更改。
SQL> ALTER USER test ACCOUNT UNLOCK;
用户已更改。
```

**【示例】 将示例用户 scott 解锁。**

```
SQL> ALTER USER scott ACCOUNT LOCK;
用户已更改。
```

#### 4. 删除用户

DROP USER 命令用于删除数据库用户，执行该命令的用户需要具有 DROP USER 系统权限，该命令的基本语法如下：

**【语法】**

```
DROP USER username [CASCADE];
```

删除用户时，不能删除当前正在连接数据库的用户，需要先终止该用户的会话然后再删除。如果该用户拥有数据库对象，则必须先删除该用户的所有数据库对象，然后再删除该用户，此时 DROP USER 命令需要使用 CASCADE 选项。

**【示例】 删除用户。**

```
SQL> DROP USER test02 CASCADE;
用户已删除。
```

### 3.2.3 用户与模式

模式(Schema)是指用户所拥有的所有对象的集合。模式与用户相对应，当在 Oracle 数据库中创建一个用户时，系统会自动在数据库中创建一个与用户同名的模式。模式作为数据库对象的容器而存在，用于数据库对象的管理，这些数据库对象包括：表、索引、视图、序列、同义词、PL/SQL 包、存储函数、存储过程等，而表空间、用户、角色等数据库对象不属于任何模式，称为非模式对象。

模式必须依赖于用户的存在而存在，即不存在不属于任何用户的模式对象。通常情况下，用户所创建的数据库对象都保存在与自己同名的模式中。“模式.对象名”的组合可以标识某个对象的所有者，默认情况下，用户在命令中引用的对象是同名模式中的对象。例如，以 SYSTEM 用户登录后编写的 SQL 语句“SELECT * FROM dual”，在执行时会被翻译为“SELECT * FROM system.dual”。如果要引用其他模式中的对象，则必须在该对象名之前指明对象所属模式。例如，在 SYSTEM 模式下访问 SCOTT 模式中的 EMP 表，SQL 语句为“SELECT * FROM scott.emp”。

## 3.3 权限

Oracle 数据库使用权限控制用户对数据库的访问和用户在数据库中所能执行的操作。所谓权限就是执行特定类型 SQL 命令或访问其他数据对象的权利。用户在数据库中可以执行什么样的操作，以及可以对哪些对象进行操作，完全取决于该用户所拥有的权限。

在 Oracle 数据库中，用户权限分为系统权限和对象权限两类。

- 系统权限：是指在数据库级别执行某种操作的权限，或针对某一类对象执行某种操作的权限，如 CREATE SESSION 权限、CREATE TABLE 权限。
- 对象权限：是指对某个特定的数据库对象执行某种操作的权限，如对某个表的 DML 操作。

### 3.3.1 系统权限

在 Oracle 数据库中，有 200 多种系统权限，每种系统权限都为用户提供了执行某一种或某一类数据库操作的能力。由于系统权限有较大的数据库操作能力，因此，应该只将系统权限授予值得信赖的用户。可以通过数据字典视图 DBA_SYS_PRIVS 获得所有系统权限的信息。

**【示例】 获取所有系统权限名称。**

```
SQL> SELECT DISTINCT privilege FROM dba_sys_privs;
PRIVILEGE
---------------------------------
CREATE JOB
INHERIT ANY PRIVILEGES
DROP ANY PROCEDURE
DROP ANY MATERIALIZED VIEW
ALTER ANY RULE SET
ALTER ANY TABLE
CREATE ANY CUBE DIMENSION
DROP ANY DIRECTORY
ALTER ANY OPERATOR
DROP ANY DIMENSION
EM EXPRESS CONNECT
…
```

系统权限可以分为以下两大类：

- 一类是对数据库某一类对象的操作能力，与具体的数据库对象无关，通常带有 ANY

关键字。例如,CREATE ANY TABLE 系统权限允许用户在任何模式中创建表;SELECT ANY TABLE 系统权限允许用户查询数据库中任何模式中的表和视图。

- 另一类系统权限是数据库级别的某种操作能力。例如,CREATE SESSION 系统权限允许用户登录数据库。

常用的数据库系统权限及其功能说明如表 3-1 所示。

**表 3-1 常用的数据库系统权限及其功能说明**

| 系统权限 | 功能 | 系统权限 | 功能 |
|---|---|---|---|
| CREATE TABLE | 在当前用户模式中创建、修改、删除表 | CREATE USER | 创建用户 |
| CREATE ANY TABLE | 在任何模式中创建表 | ALTER USER | 修改用户 |
| ALTER ANY TABLE | 修改任何模式中的表或视图 | DROP USER | 删除用户 |
| DROP ANY TABLE | 删除任何模式中的表 | CREATE SESSION | 连接登录数据库 |
| CREATE ROLE | 创建角色 | CREATE ANY INDEX | 在任何模式中创建索引 |
| ALTER ANY ROLE | 修改任何角色 | ALTER ANY INDEX | 修改任何模式中的索引 |
| DROP ANY ROLE | 删除任何角色 | DROP ANY INDEX | 删除任何模式中的索引 |

### 1. 系统权限的授予

在给用户授予系统权限时,应该根据用户的身份进行。例如,数据库管理员用户应该具有创建表空间、修改数据库结构、修改用户权限、可以对数据库中任何模式中的对象进行管理的权限;数据库开发人员应该具有在自己的模式中创建表、视图、索引等数据库对象的权限;普通用户可以只具有连接登录数据库的系统权限。

系统权限授予的语法如下。

**【语法】**

```
GRANT system_privilege_list | [ALL PRIVILEGES] TO user_list [WITH ADMIN OPTION];
```

其中:

- system_privilege_list 表示系统权限列表,多个系统权限以逗号分隔。
- ALL PRIVILEGES 表示所有系统权限。
- user_list 表示用户列表,多个用户以逗号分隔。
- WITH ADMIN OPTION 表示允许系统权限接收者再把此权限授予其他用户。

**【示例】 为用户授予连接登录数据库的系统权限。**

```
SQL> CONN SYSTEM/QSTqst2015;
已连接。
SQL> GRANT CREATE SESSION TO test;
授权成功。
```

**【示例】 为用户授予创建表、创建序列的系统权限。**

```
SQL> CONN SYSTEM/QSTqst2015;
已连接。
SQL> GRANT CREATE TABLE,CREATE SEQUENCE TO test;
授权成功。
```

在 Oracle 数据库中，特定条件下，用户可以将其获得的权限全部或部分再授予其他用户，这称为权限的传递性，使用 WITH ADMIN OPTION 选项实现。如下述示例所示。

**【示例】 为用户授予一定的系统权限，且具有传递性。**

```
SQL > CONN SYSTEM/QSTqst2015;
已连接。
SQL > GRANT CREATE TABLE,CREATE SEQUENCE TO test WITH ADMIN OPTION;
授权成功。
SQL > CONN test/QSTqst2015;
已连接。
SQL > GRANT CREATE TABLE TO test02;
授权成功。
```

上述示例中，用户 test 获得了在 test 模式中创建表和创建序列的权限，用户 test02 也获得了在 test02 模式中创建表的系统权限。

### 2. 系统权限的回收

数据库管理员可以使用 REVOKE 命令回收用户获得的系统权限。语法如下。

**【语法】**

```
REVOKE sys_privilege_list | [ALL PRIVILEGES] FROM user_list;
```

**【示例】**

```
SQL > CONN SYSTEM/QSTqst2015;
已连接。
SQL > REVOKE CREATE TABLE,CREATE SEQUENCE FROM test;
撤销成功。
```

如果一个用户获得的系统权限具有传递性，并且给其他用户进行了授权，那么当该用户的系统权限被回收后，其他用户的系统权限并不受影响。例如上述示例中，用户 test 的系统权限被回收后，并不会影响用户 test02 从其获得的创建表的系统权限。

## 3.3.2 对象权限

对象权限是指对某个特定模式对象的操作权限。数据库模式对象所有者拥有该对象的所有对象权限，对象权限的管理实际上是对象所有者对其他用户操作该对象的权限管理。

在 Oracle 数据库中，不同类型的模式对象具有不同的对象权限，还有一些对象没有对象权限，只能通过系统权限进行控制，如索引、触发器、数据库链接等对象。Oracle 数据库中常用的数据库对象权限及功能说明如表 3-2 所示。

**表 3-2 常用的数据库对象权限及其功能说明**

| 对 象 权 限 | 功 能 |
|---|---|
| SELECT | 用于查询表、视图和序列 |
| UPDATE | 更新表、视图中的数据 |
| INSERT | 向表、视图中插入新的记录 |

续表

| 对象权限 | 功　能 |
| --- | --- |
| DELETE | 删除表、视图中的数据 |
| ALTER | 修改表、序列的属性 |
| INDEX | 在表上创建索引 |
| REFERENCE | 为表创建外键 |
| EXECUTE | 函数、存储过程、程序包等的调用或执行 |

### 1. 对象权限的授予

在 Oracle 数据库中，用户可以直接访问同名模式中的数据库对象，如果要访问其他模式中的数据库对象，就要具有相应模式对象的对象权限。为用户赋予对象权限使用 GRANT 命令，语法如下。

**【语法】**

```
GRANT object_privilege_list | ALL [PRIVILEGES]
ON [schema.]object
TO user_list [WITH GRANT OPTION];
```

其中：

- object_privilege_list 表示对象权限列表，多个对象权限以逗号分隔。
- ALL [PRIVILEGES]表示某对象上的所有对象权限。
- [schema.]object 表示模式对象，默认为当前模式中的对象。
- user_list 表示用户列表，多个用户以逗号分隔。
- WITH GRANT OPTION 表示允许对象权限接收者把此对象权限授予其他用户。

**【示例】 将 scott 模式中的 emp 表的部分对象权限授予用户 test。**

```
SQL> CONN SYSTEM/QSTqst2015;
已连接。
SQL> GRANT SELECT,INSERT,UPDATE ON scott.emp TO test;
授权成功。
SQL> CONN test/QSTqst2015;
已连接。
SQL> SELECT * FROM scott.emp WHERE empno = 7782;
EMPNO     ENAME      JOB        MGR        HIREDATE         SAL        COMM       DEPTNO
-----     -------    -------    -------    -----------      -------    --------   --------
7782      CLARK      MANAGER    7839       09-6月 -81       2450                  10
```

上述示例中，通过管理员用户将 scott 模式中的 emp 表的 SELECT、INSERT、UPDATE 对象权限授予用户 test，然后在 test 模式下，便可以对表 scott.emp 进行查询、插入和更新的操作了。

下述示例演示管理员将 scott 模式中的 emp 表的 SELECT、INSERT、UPDATE 对象权限授予用户 test，用户 test 再将 scott.emp 表的 SELECT、INSERT 权限传递授予 test02。

【示例】 对象权限的传递。

```
SQL> CONN SYSTEM/QSTqst2015;
已连接。
SQL> GRANT SELECT,INSERT,UPDATE ON scott.emp TO test WITH GRANT OPTION;
授权成功。
SQL> CONN test/QSTqst2015;
已连接。
SQL> GRANT SELECT,INSERT ON scott.emp TO test02;
授权成功。
SQL> CONN test02/QSTqst2015;
已连接。
SQL> SELECT * FROM scott.emp WHERE empno = 7782;
EMPNO     ENAME      JOB        MGR        HIREDATE        SAL        COMM       DEPTNO
------    -------    -------    -------    -----------    -------    --------   --------
7782      CLARK      MANAGER    7839       09-6月 -81     2450                   10
```

如果一个用户需要具有某模式对象的所有对象权限,则可以使用 ALL[PRIVILEGES]选项一次性分配。

【示例】 为用户赋予 scott.emp 表的所有对象权限。

```
SQL> CONN SYSTEM/QSTqst2015;
已连接。
SQL> GRANT ALL ON scott.emp TO test;
授权成功。
```

### 2. 对象权限的回收

数据库管理员或模式对象所有者可以使用 REVOKE 命令回收用户获得的对象权限,命令语法如下。

【语法】

```
REVOKE object_privilege_list | ALL [PRIVILEGE]
ON [schema.]object
FROM user_list [CASCADE CONSTRAINTS];
```

其中:

- CASCADE CONSTRAINTS 表示当回收 REFERENCE 对象权限或回收 ALL PRIVILEGES 对象权限时,删除利用 REFERENCE 对象权限创建的外键约束。

【示例】 回收用户 test02 在 scott.emp 表上的某些权限。

```
SQL> CONN SYSTEM/QSTqst2015;
已连接。
SQL> REVOKE SELECT,INSERT ON scott.emp FROM test02
撤销成功。
```

【示例】 回收用户 test 在 scott.emp 表上的所有权限。

```
SQL> CONN SYSTEM/QSTqst2015;
已连接。
```

```
SQL> REVOKE ALL ON scott.emp FROM test
撤销成功。
```

如果一个用户获得的对象权限具有传递性，并且已给其他用户授权，那么当该用户的对象权限被回收后，其他用户的对象权限也将被回收。如下述示例将 scott.emp 表的所有对象权限授予用户 test，且具有传递性，然后用户 test 再将 scott.emp 表的 SELECT 对象权限授予用户 test02。当用户 test 在 scott.emp 表上的 SELECT 对象权限被回收后，用户 test02 在 scott.emp 表上的 SELECT 对象权限也被回收。

**【示例】 权限操作示例。**

```
SQL> CONN SYSTEM/QSTqst2015;
已连接。
SQL> GRANT ALL ON scott.emp TO test WITH GRANT OPTION;
授权成功。
SQL> CONN test/QSTqst2015;
已连接。
SQL> GRANT SELECT ON scott.emp TO test02;
授权成功。
SQL> CONN SYSTEM/QSTqst2015;
已连接。
SQL> REVOKE SELECT ON scott.emp FROM test;
撤销成功。
SQL> CONN test02/QSTqst2015;
已连接。
SQL> SELECT * FROM scott.emp;
SELECT * FROM scott.emp
                    *
第 1 行出现错误:
ORA-00942: 表或视图不存在
```

## 3.4 角色

虽然可以利用 GRANT 命令为所有用户分配权限，但是如果数据库的用户众多，而且权限关系复杂的话，则为用户分配权限的工作量将变得十分庞大。因此 Oracle 提出了角色的概念。

角色是指系统权限或者对象权限的集合。Oracle 允许首先创建一个角色，然后将角色信息赋予用户，从而间接地将权限分配给用户。因为角色的可复用性，因此可以将角色再次分配给其他用户，从而减少了重复工作。用户直接使用权限和使用角色管理权限的方式比较如图 3-1 所示。

在 Oracle 数据库中，角色分为系统预定义角色和用户自定义角色两类。系统预定义角色由系统创建，并由系统授权了相应的权限；用户自定义角色由用户定义，并由用户为其授权。

Oracle 数据库中有 50 多个系统预定义角色，表 3-3 列举了几个比较常用的系统预定义角色的名称及其具有的权限。

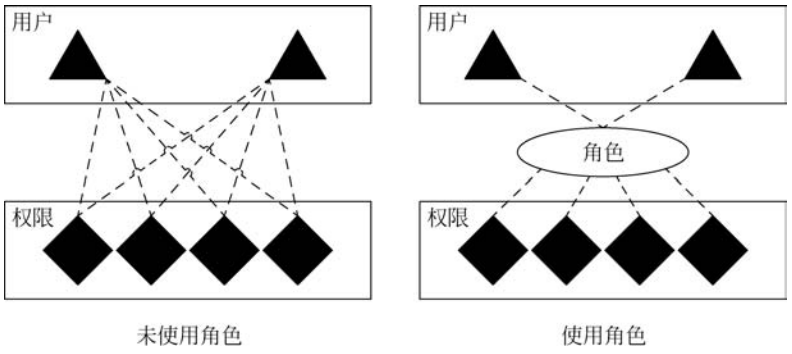



图 3-1　角色和权限的使用

**表 3-3　常用的系统预定义角色**

| 角 色 名 称 | 权　　限 |
|---|---|
| DBA | 包含所有系统权限，且带有 WITH ADMIN OPTION 选项，即可以将系统权限授予其他用户 |
| CONNECT | CREATE SESSION、ALTER SESSION、CREATE SEQUENCE、CREATE SYNONYM、CREATE VIEW、CREATE CLUSTER、CREATE DATABASE LINK |
| RESOURCE | CREATE SEQUENCE、CREATE TABLE、CREATE TRIGGER、CREATE TYPE、CREATE CLUSTER、CREATE PROCEDURE |

### 3.4.1　角色的创建与授权

Oracle 数据库允许用户自定义角色，并对自定义角色进行权限的授予与回收。用户自定义角色的创建语法如下。

**【语法】**

```
CREATE ROLE role_name [NOT IDENTIFIED] | [IDENTIFIED BY password];
```

其中：

- role_name 表示自定义角色的名称，该名称不能与任何用户名或其他角色名相同。
- NOT IDENTIFIED 表示角色采用数据库认证，激活角色时不需要口令。
- IDENTIFIED BY password 表示角色采用数据库认证，激活角色时需要输入口令。

**【示例】　创建一个不需要口令的数据库认证的角色。**

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> CREATE ROLE role_emp;
角色已创建。
```

**【示例】　创建一个需要口令的数据库认证的角色。**

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> CREATE ROLE role_manager IDENTIFIED BY rolemanager;
角色已创建。
```

创建一个角色后，如果不给角色授权，那么角色是没有用处的。给角色授权实际上是给角色授予适当的系统权限、对象权限或已有角色。角色权限的授予和回收的过程与用户权限类似，同时还需注意以下事项：

- 为角色授权的用户本身必须具有要授予的权限，并且在其获得权限时具有传递性，即指定了 WITH ADMIN OPTION 或 WITH GRANT OPTION 选项。
- 给角色授权时不能指定传递性，即不能带有 WITH ADMIN OPTION 或 WITH GRANT OPTION 选项。

下述示例演示使用管理员用户为角色 role_emp 和 role_manager 进行授权和回收。

**【示例】 角色的授权和回收。**

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> GRANT ALL ON scott.emp TO role_emp;
授权成功。
SQL> GRANT CREATE SESSION,CREATE TABLE,CREATE VIEW TO role_manager;
授权成功。
SQL> REVOKE UPDATE,DELETE ON scott.emp FROM role_emp;
撤销成功。
SQL> REVOKE CREATE TABLE,CREATE VIEW FROM role_manager;
撤销成功。
```

## 3.4.2 角色的授予与回收

创建完角色并给角色授权后，可以将角色授予用户或其他角色，实现权限的间接管理。将角色授予用户或其他角色，用户需要具有 GRANT ANY ROLE 系统权限或者用户是角色的创建者，或者用户在被授予该角色时使用了 WITH ADMIN OPTION 选项。

角色授予的语法如下。

**【语法】**

```
GRANT role_list TO user_list | role_list [WITH ADMIN OPTION];
```

其中：WITH ADMIN OPTION 表示被授予用户可以将此角色再授予其他用户，或从任何具有该角色的用户那里回收该角色。

**【示例】 将系统预定义角色授予一个角色。**

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> GRANT CONNECT,RESOURCE TO role_emp;
授权成功。
```

**【示例】 将用户自定义角色授予一个用户。**

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> GRANT role_emp TO test WITH ADMIN OPTION;
授权成功。
```

在角色授予时，可以在一个 GRANT 命令中同时为用户或角色授予系统权限和角色，但不能同时授予对象权限和角色。如下述示例所示。

**【示例】 系统权限、对象权限和角色的组合授予。**

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> GRANT CONNECT,role_emp TO test;
授权成功。
SQL> GRANT role_emp,SELECT ON scott.emp TO test;
GRANT role_emp,SELECT ON scott.emp TO test
        *
第 1 行出现错误:
ORA-00990: 权限缺失或无效
```

角色的回收与权限的回收类似，使用 REVOKE 命令从用户或其他角色回收角色，如下述示例所示。

**【示例】 从其他角色或用户回收角色。**

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> REVOKE RESOURCE FROM role_emp;
撤销成功。
SQL> REVOKE role_emp FROM test;
撤销成功。
```

### 3.4.3 角色管理

角色创建完成后，可以对自定义的角色进行维护管理，如修改角色的认证方式、禁用与激活角色以及删除角色等。

#### 1. 修改角色

角色的修改是指修改角色的认证方式，角色的名称不能修改，如果要修改角色名称，需要先删除角色，然后重建新名称的角色，再对新角色进行重新授权。

角色认证方式的修改语法如下。

**【语法】**

```
ALTER ROLE role [NOT IDENTIFIED] | [IDENTIFIED BY password];
```

**【示例】 为角色 role_emp 添加认证口令，取消 role_manager 的认证口令。**

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> ALTER ROLE role_emp IDENTIFIED BY roleemp;
角色已丢弃。
SQL> ALTER ROLE role_manager NOT IDENTIFIED;
角色已丢弃。
```

### 2. 禁用与激活角色

当一个角色授予某一个用户后，该角色即成为该用户的默认角色。当用户登录数据库时，用户所有默认角色都处于激活状态，而非默认角色处于禁用状态，因此可以通过设置用户的默认角色禁止或激活用户拥有的角色，使用的命令为 ALTER USER，语法如下。

**【语法】**

```
ALTER USER user
DEFAULT ROLE role_list | [ALL [EXCEPT role_list]] | NONE;
```

其中：

- user 表示设置默认角色的用户名称。
- role_list 表示指定的默认角色，多个角色名称以逗号分隔。
- ALL 表示将用户的所有角色都设置为默认角色，处于激活状态。
- EXCEPT role_list 表示除了指定的角色被禁用外，其余所有角色都为默认角色，处于激活状态。
- NONE 表示禁用用户的所有角色，即用户登录时所有角色都处于禁用状态。

**【示例】 禁用用户 test 的所有角色。**

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> ALTER USER test DEFAULT ROLE NONE;
用户已更改。
```

**【示例】 将用户 test 的所有角色设置为默认角色。**

```
SQL> ALTER USER test DEFAULT ROLE ALL;
用户已更改。
```

**【示例】 将用户 test 的部分角色设置为默认角色。**

```
SQL> ALTER USER test DEFAULT ROLE CONNECT,role_emp;
用户已更改。
```

**【示例】 将用户 test 除某个角色外其他所有角色设置为默认角色。**

```
SQL> ALTER USER test DEFAULT ROLE ALL EXCEPT role_emp;
用户已更改。
```

### 3. 删除角色

如果不再需要某个角色，可以使用 DROP ROLE 命令删除该角色。角色被删除后，系统将回收所有用户或其他角色从该角色中获得的权限，同时从数据字典中删除该角色的定义信息。

【示例】 删除用户自定义角色。

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> DROP ROLE role_emp;
角色已删除。
```

### 3.4.4 查询角色信息

在 Oracle 数据库中，可以通过查询包含角色信息的数据字典 DBA_ROLE_PRIVS 了解用户所具有的角色信息以及角色所具有的权限或角色信息。

【示例】 查询示例用户 scott 所具有的角色信息。

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> SELECT grantee,granted_role,admin_option,default_role
  2   FROM dba_role_privs
  3   WHERE grantee = 'SCOTT';
GRANTEE         GRANTED_ROLE           ADMIN_           DEFAUL
----------      -----------------      ------------     ---------------
SCOTT           RESOURCE               NO               YES
SCOTT           CONNECT                NO               YES
```

【示例】 查询用户 test 所具有的角色信息。

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> ALTER USER test DEFAULT ROLE CONNECT,role_emp;
用户已更改。
SQL> SELECT grantee,granted_role,admin_option,default_role
  2   FROM dba_role_privs
  3   WHERE grantee = 'TEST';
GRANTEE           GRANTED_ROLE             ADMIN_             DEFAUL
-----------       -------------------      -------------      ---------------
TEST              ROLE_EMP                 NO                 YES
TEST              CONNECT                  NO                 YES
```

【示例】 查询角色 role_emp 所具有的对象权限。

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> GRANT SELECT ON scott.emp TO role_emp;
授权成功。
SQL> SELECT role,owner,table_name,privilege
  2  FROM role_tab_privs
  3  WHERE role = 'ROLE_EMP';
ROLE            OWNER              TABLE_NAME                 PRIVILEGE
----------      --------------     --------------------       ------------------
ROLE_EMP        SCOTT              EMP                        SELECT
```

【示例】 查询角色 role_emp 所具有的系统权限。

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> GRANT CREATE TABLE,CREATE VIEW TO role_emp;
授权成功。
SQL> SELECT role,privilege,admin_option
  2  FROM role_sys_privs
  3  WHERE role = 'ROLE_EMP';
ROLE                PRIVILEGE                ADMIN_
-------------   ------------------   ----------
ROLE_EMP            CREATE VIEW              NO
ROLE_EMP            CREATE TABLE             NO
```

【示例】 查询角色 role_emp 所具有的角色。

```
SQL> CONN system/QSTqst2015;
已连接。
SQL> GRANT CONNECT,RESOURCE TO role_emp;
授权成功。
SQL> SELECT role,granted_role,admin_option
  2  FROM role_role_privs
  3  WHERE role = 'ROLE_EMP';
ROLE                GRANTED_ROLE             ADMIN_
-------------   ------------------   ---------------
ROLE_EMP            RESOURCE                 NO
ROLE_EMP            CONNECT                  NO
```

## 3.5 课程贯穿项目

### 3.5.1 【任务 3-1】 项目表空间创建

在数据库应用中,Oracle 数据库建议每个应用分别对应一个独立的表空间,并且将所有用户对象和数据保存在非系统表空间中,通过此种方式分离不同应用的数据,从而减少读取数据文件时产生的 I/O 冲突。

本节任务完成 Q_MicroChat 微聊项目表空间的创建,以及将该表空间设置为默认表空间。根据表空间的功能特性,一个表空间的创建,应该需要确定表空间的名称、表空间的类型、表空间的管理方式、表空间的数据文件、区的分配方式、段的管理方式、表空间数据块大小等参数。本项目创建的表空间名称为 ts_qmicrochat,表空间类型为永久表空间,表空间数据文件的位置和名称为 e:\oracle12c\userspace\ts_qmicrochat.dbf,数据文件大小为 100MB,数据文件自动扩展且每次扩展大小为 100MB。开启表空间的日志记录功能,设置表空间状态为可用,表空间采取本地管理方式自动分配区,段的管理方式为自动管理。具体 SQL 脚本如下所示。

**【任务 3-1】（1）创建表空间。**

```
SQL> CREATE TABLESPACE ts_qmicrochat
  2     DATAFILE 'e:\oracle12c\userspace\ts_qmicrochat.dbf' SIZE 100M
  3     AUTOEXTEND ON NEXT 100M
  4     LOGGING
  5     ONLINE
  6     EXTENT MANAGEMENT LOCAL AUTOALLOCATE
  7     SEGMENT SPACE MANAGEMENT AUTO;
表空间已创建。
```

下述任务完成将 ts_qmicrochat 表空间设置为默认表空间。

**【任务 3-1】（2）设置默认表空间。**

```
SQL> ALTER DATABASE DEFAULT TABLESPACE ts_qmicrochat;
数据库已更改。
```

## 3.5.2 【任务 3-2】 项目用户管理

Q_MicroChat 微聊项目对于用户管理有以下要求：

- 设置项目管理员用户，负责该项目所需要的所有数据库对象的创建及维护。
- 设置项目基础数据维护用户，负责对用户数据、个人动态数据（包括相册信息、文章信息）、群组数据的管理。
- 设置项目终端用户，负责对账户数据、好友数据、好友聊天数据、动态及评论数据、群组及群聊数据的部分管理。

根据用户分类需求，分别创建以下 3 个用户作为各类型用户代表，脚本实现如下。

**【任务 3-2】（1）创建管理员用户 qmicrochat_admin。**

```
SQL> CONN system/Qmicrochat2015@qmicrochat;
已连接。
SQL> CREATE USER qmicrochat_admin IDENTIFIED BY admin2015
  2  DEFAULT TABLESPACE ts_qmicrochat
  3  TEMPORARY TABLESPACE temp;
用户已创建。
```

**【任务 3-2】（2）创建基础数据维护用户 qmicrochat_operator。**

```
SQL> CREATE USER qmicrochat_operator IDENTIFIED BY operator2015;
用户已创建。
```

**【任务 3-2】（3）创建终端用户 qmicrochat_guest。**

```
SQL> CREATE USER qmicrochat_guest IDENTIFIED BY guest2015;
用户已创建。
```

创建完成后，通过下述脚本可以查询当前数据库所有用户的信息。

**【任务 3-2】（4）查询所有用户信息。**

```
SQL> SELECT username,default_tablespace,temporary_tablespace
  2  FROM dba_users
  3  WHERE account_status = 'OPEN';

USERNAME              DEFAULT_TABLESPACE          TEMPORARY_TABLESPACE
--------------------  --------------------------  ----------------------------
QMICROCHAT_ADMIN      TS_QMICROCHAT               TEMP
QMICROCHAT_GUEST      TS_QMICROCHAT               TEMP
QMICROCHAT_OPERATOR   TS_QMICROCHAT               TEMP
SYSTEM                SYSTEM                      TEMP
SYS                   SYSTEM                      TEMP
```

## 3.5.3 【任务 3-3】 项目权限设置及角色管理

Q_MicroChat 微聊项目中三类用户的权限要求如下：

- 项目管理员用户：需要具有创建及维护所有项目所需数据库对象的系统权限。
- 项目基础数据维护用户：需要具有对用户数据、个人动态数据（包括相册信息、文章信息）、群组数据进行管理的对象权限。
- 项目终端用户：需要具有对账户数据进行添加、修改、查询的对象权限；对好友关系数据、好友聊天数据、动态及评论数据、群组及群聊数据进行查询、添加、修改、删除的对象权限。

对于项目管理员用户，首先需要为其授予 UNLIMITED TABLESPACE 系统权限指定表空间限额，使其能够在表空间中创建数据库对象，同时通过授予 CONNECT 和 RESOURCE 角色的方式为其赋予对项目所需数据库对象的创建及维护的系统权限。

**【任务 3-3】（1）项目管理员用户角色分配。**

```
SQL> CONN system/Qmicrochat2015;
已连接。
SQL> GRANT UNLIMITED TABLESPACE TO qmicrochat_admin;
授权成功。
SQL> GRANT CONNECT,RESOURCE TO qmicrochat_admin;
授权成功。
```

对于项目基础数据维护用户，首先其需要具有连接登录数据库的系统权限 CREATE SESSION，其次需要具有对用户数据、个人动态数据（包括相册信息、文章信息）、群组数据进行管理的所有对象权限，此时可以使用 ALL PRIVILEGES 选项进行一次性分配。

**【任务 3-3】（2）项目基础数据维护用户权限分配。**

```
SQL> CONN system/Qmicrochat2015;
已连接。
SQL> GRANT create session TO qmicrochat_operator;
授权成功。
SQL> -- 以下的对象权限因还未创建相应的表,读者可以在第 4 章贯穿任务完成后再进行测试
SQL> GRANT ALL ON qmicrochat_admin.tb_users TO qmicrochat_operator;
授权成功。
```

```
SQL> GRANT ALL ON qmicrochat_admin.tb_personal_dynamics TO qmicrochat_operator;
授权成功。
SQL> GRANT ALL ON qmicrochat_admin.tb_photos_dynamics TO qmicrochat_operator;
授权成功。
SQL> GRANT ALL ON qmicrochat_admin.tb_artics_dynamics TO qmicrochat_operator;
授权成功。
SQL> GRANT ALL ON qmicrochat_admin.tb_groups TO qmicrochat_operator;
授权成功。
```

对于项目终端用户，首先需要具有连接登录数据库的系统权限 CREATE SESSION，其次需要具有对账户数据进行添加(INSERT)、修改(UPDATE)、查询(SELECT)的对象权限以及对好友关系数据、好友聊天数据、动态及评论数据、群组及群聊数据进行查询、添加、修改、删除的对象权限。

**【任务 3-3】 (3) 项目终端用户权限分配。**

```
SQL> CONN system/Qmicrochat2015;
已连接。
SQL> GRANT create session TO qmicrochat_guest;
授权成功。
SQL> CONN qmicrochat_admin/admin2015;
已连接。
SQL> -- 以下的对象权限因还未创建相应的表,读者可以在第 4 章贯穿任务完成后再进行测试
SQL> GRANT INSERT,UPDATE,SELECT ON tb_users TO qmicrochat_guest;
授权成功。
SQL> GRANT DELETE,INSERT,UPDATE,SELECT ON tb_friends TO qmicrochat_guest;
授权成功。
SQL> GRANT DELETE,INSERT,UPDATE,SELECT ON tb_personal_dynamics TO qmicrochat_guest;
授权成功。
SQL> GRANT DELETE,INSERT,UPDATE,SELECT ON tb_artics_dynamics TO qmicrochat_guest;
授权成功。
SQL> GRANT DELETE,INSERT,UPDATE,SELECT ON tb_photos_dynamics TO qmicrochat_guest;
授权成功。
SQL> GRANT DELETE,INSERT,UPDATE,SELECT ON tb_comment TO qmicrochat_guest;
授权成功。
SQL> GRANT DELETE,INSERT,UPDATE,SELECT ON tb_comment_reply TO qmicrochat_guest;
授权成功。
SQL> GRANT DELETE,INSERT,UPDATE,SELECT ON tb_user_chat TO qmicrochat_guest;
授权成功。
SQL> GRANT DELETE,INSERT,UPDATE,SELECT ON tb_groups TO qmicrochat_guest;
授权成功。
SQL> GRANT DELETE,INSERT,UPDATE,SELECT ON tb_group_chat TO qmicrochat_guest;
授权成功。
```


当需要为多个用户赋予类似上述多个表的对象权限时，为了方便，可以先创建一个角色，给角色授予这些表的对象权限，然后将此角色授予多个用户。

用户权限、角色分配完成后，可以通过下述数据字典进行查询验证。

**【任务 3-3】（4）查询用户具有的对象权限。**

```
SQL> CONN system/Qmicrochat2015;
已连接。
SQL> SELECT grantee,owner,table_name,grantor,privilege,grantable
    FROM dba_tab_privs WHERE LOWER(grantee) = 'qmicrochat_operator';

GRANTEE              OWNER              TABLE_NAME  GRANTOR          PRIVILEGE   GRANTABLE
---------------- -------------- --------- ------------- --------- ---------
QMICROCHAT_OPERATOR  QMICROCHAT_ADMIN  TB_USERS    QMICROCHAT_ADMIN FLASHBACK   NO
QMICROCHAT_OPERATOR  QMICROCHAT_ADMIN  TB_GROUPS   QMICROCHAT_ADMIN INSERT      NO
QMICROCHAT_OPERATOR  QMICROCHAT_ADMIN  TB_USERS    QMICROCHAT_ADMIN INDEX       NO
QMICROCHAT_OPERATOR  QMICROCHAT_ADMIN  TB_GROUPS   QMICROCHAT_ADMIN SELECT      NO
…

SQL> SELECT grantee,owner,table_name,grantor,privilege,grantable
    FROM dba_tab_privs WHERE LOWER(grantee) = 'qmicrochat_guest';

GRANTEE            OWNER              TABLE_NAME  GRANTOR          PRIVILEGE   GRANTABLE
-------------- -------------- --------- ------------- --------- ---------
QMICROCHAT_GUEST   QMICROCHAT_ADMIN  TB_USERS    QMICROCHAT_ADMIN INSERT      NO
QMICROCHAT_GUEST   QMICROCHAT_ADMIN  TB_USERS    QMICROCHAT_ADMIN SELECT      NO
QMICROCHAT_GUEST   QMICROCHAT_ADMIN  TB_USERS    QMICROCHAT_ADMIN UPDATE      NO
QMICROCHAT_GUEST   QMICROCHAT_ADMIN  TB_USERS    QMICROCHAT_ADMIN DELETE      NO
QMICROCHAT_GUEST   QMICROCHAT_ADMIN  TB_PERSONAL_DYNAMICS  QMICROCHAT_ADMIN  INSERT  NO
…
```

**【任务 3-3】（5）查询用户具有的系统权限。**

```
SQL> SELECT grantee,privilege,admin_option
    FROM dba_sys_privs WHERE LOWER(grantee) = 'qmicrochat_operator';

GRANTEE                PRIVILEGE           ADMIN_OPTION
------------------ ---------------- ----------------------------
QMICROCHAT_OPERATOR    CREATE SESSION      NO
```

**【任务 3-3】（6）查询用户具有的角色。**

```
SQL> SELECT grantee,granted_role FROM dba_role_privs
  2     WHERE LOWER(grantee) = 'qmicrochat_admin';

GRANTEE                GRANTED_ROLE
------------------ --------------------------------------------------
QMICROCHAT_ADMIN       CONNECT
QMICROCHAT_ADMIN       RESOURCE
```

# 本章小结

## 小结

- 在 Oracle 数据库创建时，系统会自动创建 SYSTEM（系统表空间）、SYSAUX（辅助系统表空间）、TEMP（临时表空间）、UNDOTBS1（撤销表空间）、USERS（用户表空间）。其中，SYSTEM 和 SYSAUX 称为系统表空间；USERS 为整个数据库的默认表空间。
- 表空间的创建需要确定表空间的名称、表空间的管理方式、表空间的类型、表空间的数据文件、区的分配方式、段的管理方式、表空间数据块大小等参数。
- 每个应用应该分别对应一个独立的表空间，所有用户对象和数据保存在非系统表空间中，这样不仅能够分离不同应用的数据，而且能够减少读取数据文件时产生的 I/O 冲突。
- Oracle 中的用户可以分为两类：一类是创建数据库时系统预定义的用户，也称为系统用户；另一类是根据应用需要由系统用户创建的用户。在 Oracle 数据库中通过创建新用户，然后设置该用户的权限控制用户对数据库的访问和操作。
- Oracle 数据库使用权限控制用户对数据库的访问和用户在数据库中所能执行的操作。所谓权限就是执行特定类型 SQL 命令或访问其他数据对象的权利。
- 在 Oracle 数据库中，用户权限分为系统权限和对象权限两类。系统权限是指在数据库级别执行某种操作的权限，或针对某一类对象执行某种操作的权限；对象权限是指对某个特定的数据库对象执行某种操作的权限。
- 角色是指系统权限或者对象权限的集合。角色分为系统预定义角色和用户自定义角色两类。系统预定义角色由系统创建，并由系统授权了相应的权限；用户自定义角色由用户定义，并由用户为其授权。

## Q&A

1. 问：在数据库创建时，会自动创建哪些表空间？哪个是系统默认表空间？有何作用？如何更改系统默认表空间？

答：在 Oracle 数据库创建时，系统会自动创建 SYSTEM（系统表空间）、SYSAUX（辅助系统表空间）、TEMP（临时表空间）、UNDOTBS1（撤销表空间）、USERS（用户表空间）。数据库的默认表空间为 USERS，用于存储用户所创建的数据库对象。可以通过 ALTER DATABASE DEFAULT TABLESPACE 命令重设数据库默认表空间。

2. 问：Oracle 数据库中有哪些用户？如何分类？哪些用户可以创建新用户？

答：Oracle 中的用户可以分为两类：一类是创建数据库时系统预定义的用户，也称为系统用户，常用的如 SYS、SYSTEM、HR、SCOTT、SH 等；另一类是根据应用需要由系统用户创建的用户。管理员用户（SYS、SYSTEM）及具有 CREATE USER 权限的用户可以

创建新的用户。

3. 问：如何理解权限、角色及它们之间的关系？

答：Oracle 数据库中的权限包括系统权限和对象权限两类。系统权限是数据库级别的权限，而对象权限是特定数据库对象所具有的权限。对用户的授权有两种方式，一种方式是直接给用户授予系统权限或对象权限；另一种方式是通过角色间接给用户授权。角色是一系列权限的集合，包括系统预定义角色和用户自定义角色两种。DBA 可以根据需要创建角色，然后给角色授权，最后将角色授予用户。通过角色可以方便地管理不同身份用户的权限。

# 章节练习

## 习题

1. 下面(　　)不是对象权限。

A. Insert　　B. Update　　C. Delete　　D. Add

2. 下列(　　)属于模式对象。

A. 数据段　　B. 盘区　　C. 表　　D. 表空间

3. 收回用户权限的关键字为(　　)。

A. Grant　　B. Revoke　　C. Drop　　D. Create

4. 为用户分配权限的关键字为(　　)。

A. Comment　　B. Lock　　C. Select　　D. Grant

5. 删除用户的关键字为(　　)。

A. Drop　　B. Delete　　C. Create　　D. Alter

6. 一个模式只能被一个_____所拥有，其创建的所有模式对象都保存在自己的_____中。

7. 把用户名为 User01 的密码修改为 password 的语句为_____。

8. 将用户 User 解锁的语句为_____。

9. 简述用户和角色的区别。

## 上机

1. 训练目标：表空间的创建。

| 培养能力 | 掌握表空间的创建 | | |
|---|---|---|---|
| 掌握程度 | ★★★★★ | 难度 | 容易 |
| 代码行数 | 1 | 实施方式 | 重复编码 |
| 结束条件 | 独立编写，运行不出错 | | |
| 参考训练内容：创建一个表空间 testsize，其数据文件大小为 2MB，并设置自动增长尺寸为 1MB | | | |

2. 训练目标：表空间、用户、权限的创建和维护。

| 培养能力 | 熟练掌握表空间、用户和权限的创建和维护 | | |
|---|---|---|---|
| 掌握程度 | ★★★★★ | 难度 | 容易 |
| 代码行数 | 10 | 实施方式 | 重复编码 |
| 结束条件 | 独立编写，运行不出错 | | |

参考训练内容：

创建表空间 team，在此表空间下创建数据表 player，表的创建语句如下所示。

```
CREATE TABLE player{
    playid  NUMBER(6)  PRIMARY KEY,
    playname  VARCHAR2(30)  NOT NULL,
    teamnum  NUMBER(6)  NOT NULL  UNIQUE,
    info  VARCHAR2(50)
} TABLESPACE team;
```

在此条件下执行以下操作。

(1) 创建一个新账户，用户名为 account1，密码为 oldpwd1。

(2) 授权该用户对数据库中 player 表的 SELECT 和 INSERT 权限，并且授权该用户对 player 表的 info 字段的 UPDATE 权限。

(3) 用 SYSTEM 账号登录数据库，为用户 account1 授予对表 player 的 SELECT 和 INSERT 权限；授予更新 player 表 info 字段的 UPDATE 权限。

(4) 更改 account1 用户的密码为 newpwd2。

(5) 收回 account1 用户的权限。

(6) 将 account1 用户的账号信息从系统中删除