第1章 Oracle概述

1-1 概述

* 用户——表空间
* 表——约束
* 查询操作

1-2 安装Oracle

* oracle 11g版本。
* 官网下载2个安装包（2GB+），解压到同一个文件夹。
* 安装：

1. 安装和配置数据库选项。
2. 普通PC安装选择“桌面类”。
3. 设置安装路径、数据库版本（企业版）、字符集、全局数据库名称以及登录口令。
4. 设置用户口令，sys和system两个用户的默认口令为上一步设置的登录口令。

1-3 卸载Oracle

》deinstall.bat

使用Oracle安装录下（\product\11.2.0\dbhome\_1\deinstall\）的deinstall.bat批处理文件进行卸载。

第2章 用户和表空间

2-1.2 使用系统用户登录Oracle

》SQL Plus工具

使用**SQL Plus**工具登录Oracle，操作数据库。

》系统用户

|  |  |  |  |
| --- | --- | --- | --- |
| 用户名 | 级别 | 权限 | 密码 |
| sys | 必须以操作员或者系统管理员的身份登录 | 最大 | 默认登录口令 |
| system | 属于管理员级别的用户，可以直接登录 |  | 默认登录口令 |
| sysman | 管理员级别用户，主要用于操作企业管理器 |  | 默认登录口令 |
| scott | 不是管理员 |  | tiger |
| … | … | … | … |
|  |  |  |  |

》登录命令

[connect] **用户名**/**密码** [@**服务器**] [as sysdba|sysoper]

若数据库不在本机上，要使用@指定数据库所在服务器；当时有sys等高权限的用户登录时，可能要使用as表明登录身份。

》查看登录用户

show user

直接查看当前登录的用户。

\*\* 在Oracle中，使用数据库命令可以不用以分号结尾，而SQL语句则必须以分号结尾。

》用户数据字典

**dba\_users**数据字典：存放了系统用户相关信息的数据表。

2-3 启用用户

scott用户默认情况下是被锁定的，即无法登录使用，因此需要进行解锁才能使用。

》解锁用户

alter user **用户名** account unlock；

》锁定用户

alter user **用户名** account lock；

2-5 表空间概述

* 表空间是Oracle数据库的逻辑存储空间。
* 一个表空间由一个或多个数据文件组成。
* Oracle数据库的数据实际上是存储在表空间的数据文件中。

》表空间分类

1. **永久表空间：**存放数据表、视图、存储过程等永久性数据的表空间。
2. **临时表空间：**存放临时数据的表空间。
3. **UNDO表空间：**存放数据被修改之前的值的表空间。

2-5 用户表空间

》表空间数据字典

**dba\_tablespaces**数据字典：存放管理员所拥有的表空间的相关信息，只有具有管理员身份的用户才能访问

**user\_tablespaces**数据字典：存放普通用户所用有的表空间的相关信息。

**tablespace\_name**字段：指代表空间的名称。

》用户数据字典

**dba\_users**数据字典：存放管理员信息的数据表，只有具有管理员身份的用户才能访问。

**user\_users**数据字典：存放普通用户相关信息数据表。

**username**字段：指代用户名称。

**default\_tablespace**字段：指代用户默认表空间。

**temporary\_tablespace**字段：指代用户的临时表空间。

一般用户的临时表空间都是temp表空间，默认表空间为system。

》更改用户表空间

alter user **用户名** default|temporary tablespace **表空间名称**；

2-8 创建表空间

》创建永久表空间

create teblespace **表空间名称**

datafile ' **[数据文件路径]+ 数据文件名称 .dbf** '

size **文件初始大小**；

》创建临时表空间

create temporary teblespace **临时表空间名称**

tempfile' **[数据文件路径]+ 数据文件名称 .dbf** '

size **文件初始大小**；

\*\* 省略数据文件路径时，数据文件会默认存放到Oracle的安装路径下。

》数据文件数据字典

**dba\_data\_files** 数据字典（数据表）存放了永久数据文件的相关信息。

**dba\_temp\_files** 数据字典（数据表）存放了临时数据文件的相关信息。

**file\_name** 是表空间数据文件存放路径的一个字段。

**tablespace\_name** 是表空间名称的字段，查询时要用大写字母。

默认数据文件存储路径：

D:\ORACLE\ORACLE\_DATABASE\ORACLEDB\_USING\PRODUCT\11.2.0\DBHOME\_1\DATABASE\TEST.DBF

2-9 修改表空间

》修改表空间的状态

alter teblespace 表空间名称 online|offline ;

online（联机）：即可用状态。

offline（脱机）：即不可用状态。

alter teblespace 表空间名称 read only|read write ;

read only（只读）：即只读状态。

read write（读写）：即可读写状态。

\*\* 设置读写状态只有在表空间为online状态写才可以设置，online状态下默认为read write 可读写状态.

》查看表空间的状态

**dba\_tablespaces** 数据字典，存放了表空间的相关信息。

**status** 字段代表表空间的当前状态，有3种取值：online 、offline 、read only。

**tablespace\_name** 字段代表表空间名称，查询时要用大写字母。

2-10 修改数据文件

》增加数据文件

alter teblespace **表空间名称**

add datafile ' **[数据文件路径]+ 数据文件名称 .dbf** '

size **文件初始大小**；

\*\* 增加数据文件的文件存放规则与创建文件时的规则一样。

》删除数据文件

alter teblespace **表空间名称**

drop datafile ' **数据文件名称 .dbf** ' ；

\*\* 此语句无法删除创建表空间时创建的第一个数据文件。

2-12 删除表空间

drop tablespace **表空间名称** [**including contents**];

\*\* 不带 including contents 时，只是删除表空间，而保留数据文件，要彻底删除表空间和相关文件则要加上 including contents。

第3章 管理表

3-1 认识表

* 数据的基本存储单位
* 二维结构
* 行和列

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **列**（域或字段） | | | |
| **行**（记录） |  |  |  |  |
|  |  |  |  |
|  |  |  |  |

》一般约束

* 每一列必须具有相同的数据类型
* 列名唯一
* 每一行的数据唯一

3-2 数据类型

》字符型

**固定长度类型：**每个数据的长度固定为设置的数值n。

char（n）：n的最大值为2000。

nchar（n）：n的最大值为1000，以unicode的格式编码，存储汉字多用nchar类型。

**可变长度类型：**数据的长度有内容的大小决定，n为数据的最大长度。

varchar2（n）：n的最大值为4000。

nvarchar2（n）：n的最大值为2000，以unicode的格式编码。

》数值型

number（p，s）：p表示有效位数，s表示小数点位数，p-s为整数位数。

float（n）：浮点类型，存储126位的二进制数据。

》日期型

date：精确到秒的日期类型。

timestamp：时间戳，可精确到小数秒。

》大对象型

BLOB：以二进制形式存储数据，最大容量为4GB。

CLOB：以字符串形式存储数据，最大容量为4GB。

3-4 创建表

create table **表名称**（

**列名** **数据类型**，

……

**列名** **数据类型**

）；

\*\* 每一列以逗号（,）隔开，列名与数据类型以空格隔开，最后一列末尾无需加逗号；最后右括号需加分号（;）结束。

**desc关键字：**查看表结构，desc 表名。

3-5 修改表

》添加字段

alter table **表名称** add **列名** **数据类型**；

》修改字段数据类型

alter table **表名称** modify **列名** **新数据类型**；

包括修改数据的长度和修改数据的类型。

\*\* 通常在表中没有数据的情况下才能修改数据类型。

》删除字段

alter table **表名称** drop column **列名**；

》修改字段名称

alter table **表名称** rename column **列名** to **新列名**；

》重命名表

rename **表名称** to **新表名**；

\*\* 字段相关的内容都是属于表内部的结构，所以用alter table语句指定表进行修改；而表名称是属于表外部的特征，因而直接用rename直接重命名。

3-7 删除表

》truncate关键字

truncate table **表名称**；

截断表，即只清除表中的数据，而保留表结构。

》drop关键字

drop table **表名称**；

将表中数据和表结构全部删除。

第4章 操作表中数据

4-1 添加数据

》insert语句

insert into **表名**(**列名1**，**列名2**，…) values(**值1**，**值2**，…)；

表名后的字段列表省略时，默认对表中所有字段都插入数据。

插入的值有字符串时，字符串要加单引号。

\*\* Oracle的insert语句每次只能插入一行记录，不支持多行同时插入。

**sysdate变量：**全局变量，指代当前时间.

》在创建表时添设置默认值

在字段数据类型后加上“default 默认值”即可:

create table **表名称**（

**列名** **数据类型** default 默认值，

……

）；

》修改表为字段添加默认值

alter table **表名称** modify **列名** default 默认值；

\*\* 此处是为字段添加默认值，不属于修改字段属性的内容，因而modify后面没有column关键字.

4-2 添加数据

》创建表时复制数据

create table **表名** as **select语句**；

使用select从另一个表中复制到新表，同时复制所选字段的结构（同时复制表结构和数据）。

》添加数据时复制

insert into **表名**(**列名1**，**列名2**，…) **select语句**；

将select语句的查询结果插入表中，实现多行数据的插入。

\*\* select查询的字段的类型必须与插入表对应顺序字段的类型相同，字段名可以不同。

4-3修改数据

》update语句

update **表名** set **字段1=值1**，**字段2=值2**，… [**where条件语句**]；

使用set关键字，为指定字段赋值；当不加where语句时，表示对表中所有记录的指定属性修改值。

4-4 删除数据

》delete语句

delete from **表名** [**where条件语句**]；

当不加where语句时，会删除表中的所有数据。

\*\* delete与truncate的功能相似，都是删除表中的数据，但truncate是数据定义语言（DDL）层次的命令，因而操作速度更快。

第5章 约束

5-1 概述

* 确定字段之间、表之间的一些规则。
* 确保数据的完整性。

5-2 非空约束

确保记录的某些字段值不为空，限制了插入操作。

》创建表时添设置非空约束

在字段数据类型后加上“not null”即可：

create table **表名称**（

**列名** **数据类型** not null，

……

）；

\*\* 非空约束比较特殊，它没有约束名称，且只能在表级上创建。

》修改表时为字段添加非空约束

alter table **表名称** modify **列名** not null；

\*\* 此处是为字段添加约束，不属于修改字段属性的内容，因而modify后面没有column关键字；添加非空约束是，应确保指定字段不存在值为空的记录。

》删除非空约束

alter table **表名称** modify **列名** null；

\*\* 与添加非空约束相似，只是把not null改为null，表示运行为空值。

5-3.4 主键约束

* 确保表中每一行数据的唯一性。
* 条件：非空、唯一。
* 一张表只能设置一个主键约束。
* 主键约束可由多个字段共同构成，有多个字段构成的主键称为联合主键或复合主键。

》创建表时添设置主键约束

**①** 在字段数据类型后加上“primary key”即可：

create table **表名称**（

**列名** **数据类型** primary key，

……

）；

**②** 在表创建语句末尾创建约束：

constraint **约束名称** primary key(**字段1，字段2，…**)

create table **表名称**（

**列名** **数据类型**，

……，

constraint 约束名称 primary key(字段1，字段2，…)

）；

添加主键约束的字段会自动添加非空约束和唯一约束。

\*\* 方法①以列级方式创建约束，方法②创建的约束为表级约束。

》修改表时为字段添加主键约束

alter table **表名称** add constraint **约束名称** primary key(**字段1，字段2，…**)；

》更改约束名称

alter table **表名称** rename constraint **约束名称** to **新约束名称**；

该方式可以修改所有类型的约束的名称。

》删除约束

**①** 禁用约束

disable：约束不可用

enable：启用约束

alter table **表名称** disable|enable constraint **约束名称**；

**②** 删除约束

alter table **表名称** drop constraint **约束名称**；

\*\* 此外，因为主键在一张表中是唯一的，所以可以用drop primary key直接删除主键约束（cascade表示删除层叠链接到该主键约束上的其他约束）。

alter table **表名称** drop primary key [cascade]；

》查看表的约束

**user\_constraints** 数据字典：存放表和表约束等信息的数据表。

**constraint\_name** 字段：约束名称。

**constraint\_type** 字段：约束的类型，P-主键，U-唯一约束，R-外键约束，C-检查约束。

**table\_name** 字段：表名称。

**status** 字段：约束的状态，disable或enable。

\*\* 数据字典中的数据貌似是以大写的形式存储的，所以在where条件中要使用大写。

\*\* 主键约束的名称通常以“pk”开头，以方法①设置主键时，数据库系统会自动为主键约束设置名称（看不懂的名称）。

5-5.6.7.8外键约束

* 包含外键的表为从表，被外键参照的表为主表。
* 外键字段的值必须全部来源于主表相应字段，或者为null。
* 主表被参照的字段必须为主键。

》 创建表时添设置外键约束

**①** 列级创建外键约束，在字段数据类型后加上“references”并指定主表和参照字段：

create table **从表名称**（

**列名** **数据类型** references **主表名称**(**字段名称**)，

……

）；

**②** 表级创建外键约束，在表创建语句末尾创建约束：

constraint **约束名称** foreign key(**字段名称**)

references **主表名称**(**字段名称**) [on delete cascade]

create table **从表名称**（

**列名** **数据类型**，

……，

constraint 约束名称 foreign key(字段名称)

references 主表名称(字段名称)

）；

\*\* 外键约束的名称通常以“fk”开头。

》修改表时为字段添加外键约束

alter table **从表名称** add constraint **约束名称** foreign key(**字段名称**)

references **主表名称**(**字段名称**) [on delete cascade]；

\*\* on delete cascade 为连级删除，当主表的被参照字段的值被删除是，从表的参照字段的相应值自动被删除。

\*\* 约束的名称必须是唯一的，不同的表也不能创建同名的约束。

》删除外键约束

与删除主键约束的方法相同，包括禁用约束和直接删除约束两种情形。

5-9.10.11 唯一约束

作用：保证字段值的唯一性。

》主键约束与唯一约束的比较

|  |  |
| --- | --- |
| **主键约束** | **唯一约束** |
| 字段值必须非空 | 允许存在一个空值 |
| 每张表只能有一个主键约束 | 一张表可以有多个唯一约束 |

》 创建表时添设置唯一约束

**①** 列级创建唯一约束，在字段数据类型后加上“unique”：

create table **从表名称**（

**列名** **数据类型** unique，

……

）；

**②** 表级创建外键约束，在表创建语句末尾创建约束：

constraint **约束名称** unque(**字段名称**)

create table **从表名称**（

**列名** **数据类型**，

……，

constraint 约束名称 unque(字段名称)

）；

\*\* 唯一约束的名称通常以“un”开头；当需要为多列添加唯一约束时，必须单独为每一列设置一个约束，而不能用“在unique关键字后添加多个字段”的形式。

》修改表时为字段添加唯一约束

alter table **从表名称** add constraint **约束名称** unique (**字段名称**)；

》删除唯一约束

与删除主键约束的方法相同，包括禁用约束和直接删除约束两种情形。

5-12.13.14 检查约束

作用：规范字段的值，使其具有实际的意义。

》创建表时添设置检查约束

**①** 列级创建唯一约束，在字段数据类型后加上“check”和相应的约束条件：

create table **从表名称**（

**列名** **数据类型** check**(条件表达式)**，

……

）；

**②** 表级创建外键约束，在表创建语句末尾创建约束：

constraint **约束名称** check(**条件表达式**)

create table **从表名称**（

**列名** **数据类型**，

……，

constraint 约束名称 check(条件表达式)

）；

\*\* 唯一约束的名称通常以“ck”开头 。

》修改表时为字段添加检查约束

alter table **从表名称** add constraint **约束名称** check(**条件表达式**)；

》删除检查约束

与删除主键约束的方法相同，包括禁用约束和直接删除约束两种情形。

第6章 查询语句

6-2 基本查询语句

》select关键字

select [distinct] **字段1，字段2，…** from **表名** [**where条件语句**];

若要查询表中的所有字段，可用“\*”代替字段列表。

select默认是保留重复的记录的，使用select distinct可以去除结果中重复的记录。

6-3 在SQL Plus中设置结果显示格式

》设置字段显示的别名

column **字段名** heading **别名**；

该语句设置了select语句显示结果是表头显示的字段名称，而不会改变表中真实的字段名称。

\*\* column关键字可以简写为col。

》设置数据显示的格式

column **字段名** format**格式**；

**①** 设置字符型数据的显示宽度：

**a字符个数**：a表示设置的是字符类型的数据，指定一列的显示宽度。

**②** 设置数字格式：

**9**：一个“9”代表以为数字，如999.9，表示最大显示3位整数和1位小数，若实际的数字大小超过了设置的格式，则整个数字会用一串“#”表示；同时数字格式前还可以加“$”等符号，如$999.9，表示在显示的数字前加一个“$”符号。

》清除显示格式

column **字段名** clear；

6-5 为查询结果的字段设置别名

》as关键字

select **字段** as **别名**, …… ;

在查询的字段后使用as关键字为字段设置别名，同时as关键字还可以省略不写。

6-6.7 运算符和表达式

* 算数运算符（+，–，\*，/）
* 比较运算符（>，<，>=，<=，=，<>不等于）
* 逻辑运算符（and，or，not）

表达式=操作数+运算符=一个值。

\*\* 在select语句的字段列表中可以使用算术运算符对查询结果进行算数运算，而比较运算符和逻辑运算符多用在where条件语句中。

6-8 带条件的查询语句

》where关键字

where **布尔表达式**

使用where语句设置查询条件，将每一行记录各个字段的值代入where条件语句中，若结果为真（true），则返回该记录。

**单一条件：**仅使用比较运算符。

**多条件：**使用逻辑运算符连接表达式。

**优先级：**比较运算符 > 逻辑运算符

not > and > or

6-9 模糊查询

》like关键字

将字段的值与匹配字符串进行匹配，一致的返回真（true）。

通配符：

下划线（**\_**）：只匹配一个字符。

百分号（**%**）：匹配0到多个字符。

使用方式：

where **字段** like ‘**匹配字符串**’

其中，通配符也是要放在单引号中，表示字符串的一部分。

6-10 范围查询

》between…and…

**字段** between **值1** and **值2**

表示查询的值的范围，通常只用于数值型的数据，所表示的范围是一个闭区间，只要指定字段的值落在该区间内，则返回真。

》in关键字

**字段** in(**值1，值2，…**)

只要指定字段与取值列表中的其中一个值相同，则返回真。

\*\* 在between和in关键字前面，可加上not运算符，表示“不在指定区间内”或“不属于指定的集合”。

6-11 查询结果排序

》order by关键字

order by **字段1** desc|asc，**字段2** desc|asc，…

select **…** from **…** [**where条件语句**]

order by 字段1 desc|asc，字段2 desc|asc，…;

desc表示降序排列，asc为升序排列；当需要按多个字段排序时，在order by后面接多个“字段 排序方式”的组合即可，组合间用逗号隔开。

6-12 case…when语句

》case…when关键字

**①** case后的字段与when的值进行匹配

case **字段**

when **值1** then **结果1**

when **值2** then **结果2**

……

[else **结果**]

end

将case后指定字段的值与when后的值依次比较，若相同，则返回相应then后面的结果，然后结束语句；若都不相同，则返回else后的结果。

**②** when后使用比较表达式

case

when **字段 ?比较运算符 值1** then **结果1**

when **字段 ?比较运算符 值2** then **结果2**

……

[else **结果**]

end

when后面的表达式依次执行，若为真，则返回相应then的结果，然后结束语句；若都不为真，则执行else的结果。

**使用：**case…when语句可以使用在select语句的字段列表中，作为对查询结果的判断分类，返回分类的结果。当所有when都不为真，且没有eles子句时，则什么都不返回。

\*\* 每个when子句后面是没有逗号分隔的。

6-15 decode函数

》decode函数

与case…when相似，用于匹配字段的值进行分类。

decode(**字段**，**值1**，**结果1**，**值2**，**结果2**，**…**，[**默认结果**])

将字段值与值比较，相同则返回相应的结果，并结束函数；若都不相同，则返回默认结果。

**使用：**decode函数可以在select语句的字段列表中使用，为对查询结果的判断分类。当字段值与所有值都不相等时，返回null。