第 6 章

数据类型和运算符

学习指引

数据库表由多列字段构成，每一个字段指定了不同的数据类型，不同的数据类型也决定了 Oracle 在存储时的使用方式，以及在使用时选择什么运算符号进行运算。本章介绍 Oracle 的数据类型和运算符，主要内容包括常见数据类型的概念与应用、数据类型的选择方法、常见运算符的应用等。

重点导读

- 熟悉常见数据类型的概念和区别。
- 掌握如何选择数据类型。
- 熟悉常见运算符的概念和区别。

6.1 Oracle 数据类型介绍

Oracle 支持多种数据类型，按照类型来分，可以分为字符串类型、数字类型、日期类型、LOB 类型、LONG RAW&RAW 类型、ROWID&UROWID 类型。其中最常用的数据类型包括数值类型、日期与时间类型和字符串类型等。

6.1.1 数值类型

数值型数据类型主要用来存储数字，Oracle 提供了多种数值数据类型，不同的数据类型提供不同的取值范围，可以存储的值范围越大，其所需要的存储空间也越大。表 6-1 为 Oracle 的常用数值类型。

表 6-1 Oracle 的常用数值类型

类 型 名 称	描 述
NUMBER(P,S)	数字类型，P 为整数位，S 为小数位
DECIMAL(P,S)	数字类型，P 为整数位，S 为小数位

续表

类 型 名 称	描　　述
INTEGER	整数类型，数值较小的整数
FLOAT	浮点数类型，NUMBER(38)，双精度
REAL	实数类型，NUMBER(63)，精度更高

Oracle 的数值类型主要通过 number(m,n)类型来实现，语法格式如下：

```
number(m,n)
```

其中，m 的取值范围为 1～38，n 的取值范围为-84～127。

number(m,n)是可变长的数值列，允许 0、正值及负值，m 是所有有效数字的位数，n 是小数点以后的位数。例如：

```
number(5,2)
```

这个字段的最大值是 99.999，如果数值超出了位数限制，就会被截取多余的位数。例如：

```
number(5,2)
```

但在一行数据中的这个字段输入 575.316，则真正保存到字段中的数值是 575.32。例如：

```
number(3,0)
```

输入 575.316，真正保存的数据是 575。对于整数，可以省略后面的 0，直接表示如下：

```
number(3)
```

【例 6-1】创建表 tb_emp1，其中 tel 字段的数值最大设定为 11，在 SQL Plus 窗口中输入的 SQL 语句如下：

```
CREATE TABLE tb emp1
(
   id         NUMBER(11),
   name       VARCHAR2(25),
   age        NUMBER(2),
   tel        NUMBER(11),
   address    VARCHAR2(25)
);
```

按 Enter 键，语句执行结果如图 6-1 所示，即可完成数据表的创建。

这里可以看到 age 字段的数据类型为 NUMBER(2)，注意到后面的数字 2，这表示的是该数据类型指定的最大长度，如果插入数值的位数大于 2，则会弹出错误信息。例如，这里插入一个大于 2 位的数值来表示年龄，可以在 SQL Plus 窗口中输入以下 SQL 语句：

```
INSERT INTO tb_emp1 ( age ) VALUES(100);
```

按 Enter 键，语句执行结果如图 6-2 所示，可以看到提示的错误信息。

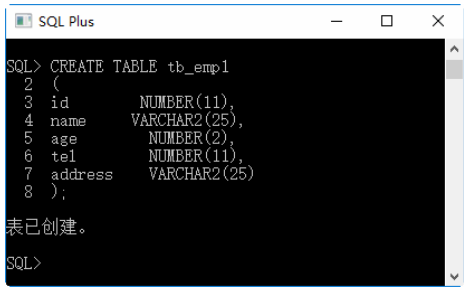

图 6-1　创建表 tb_emp1

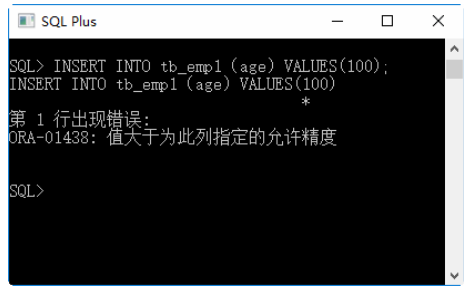

图 6-2　错误信息提示

在 SQL Plus 窗口中修改 SQL 语句：

```
INSERT INTO tb_emp1 (age) VALUES(50);
```

按 Enter 键，语句执行结果如图 6-3 所示，可以看到成功创建行。

在 SQL Plus 窗口中输入查看表结构的 SQL 语句：

```
SELECT * FROM tb_emp1;
```

按 Enter 键，语句执行结果如图 6-4 所示，可以看到成功创建行。

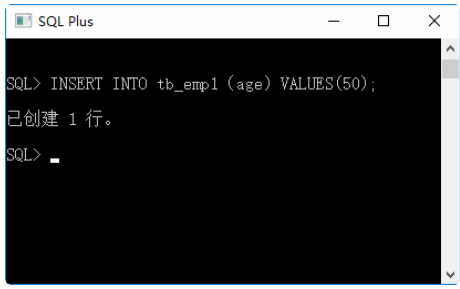

图 6-3　插入一行数据

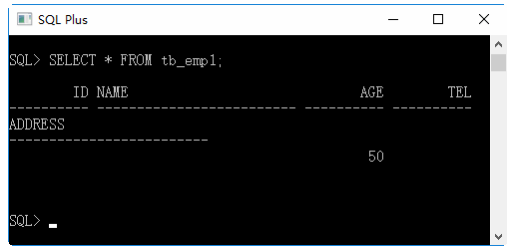

图 6-4　查看表结构

【例 6-2】创建表 tb_emp2，其中字段 a、b、c 数据类型依次为 NUMBER(2)、NUMBER(4)、NUMBER(6)，在 SQL Plus 窗口中输入的 SQL 语句如下：

```
CREATE TABLE tb_emp2
(
    a       NUMBER(2),
    b       NUMBER(4),
    c       NUMBER(6)
);
```

按 Enter 键，语句执行结果如图 6-5 所示，可以看到成功创建表。

执行成功之后，使用 DESC 查看表结构，在 SQL Plus 窗口中输入的 SQL 语句如下：

```
SQL> DESC tb_emp2;
```

按 Enter 键，语句执行结果如图 6-6 所示，可以看到表的结构。

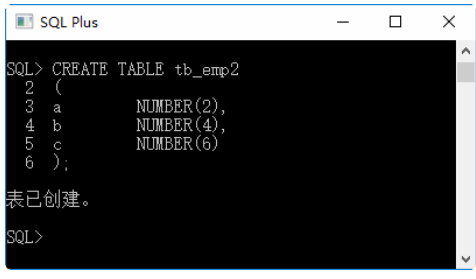

图 6-5　创建表 tb_emp2

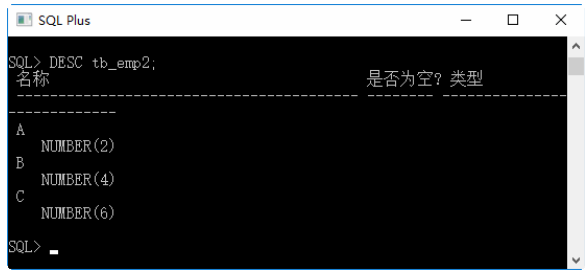

图 6-6　查询表结构

【例 6-3】创建表 tb_emp3，其中字段 a、b、c 的数据类型依次为 NUMBER (8,1)、NUMBER(8,3)和 NUMBER (8,2)，向表中插入数据 8.1、8.15 和 8.123，在 SQL Plus 窗口中输入的 SQL 语句如下：

```
CREATE TABLE tb emp3
(
    a  NUMBER (8,1),
    b  NUMBER (8,3),
    c  NUMBER (8,2)
```

```
);
```

按 Enter 键，语句执行结果如图 6-7 所示，可以看到创建的数据表。

向表中插入数据，在 SQL Plus 窗口中输入的 SQL 语句如下：

```
SQL>INSERT INTO tb_emp3 VALUES(8.12, 8.15, 8.123);
```

按 Enter 键，语句执行结果如图 6-8 所示，可以看到创建的行。

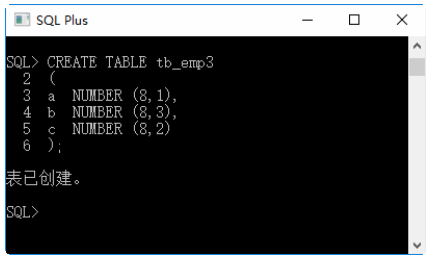

图 6-7 创建表 tb_emp3

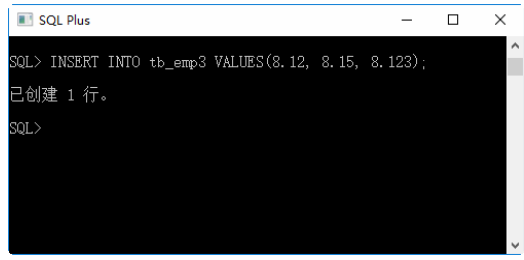

图 6-8　向表中插入数据

插入数据后，查看输入的数据信息。在 SQL Plus 窗口中输入的 SQL 语句如下：

```
SQL> SELECT * FROM tb_emp3;
```

按 Enter 键，语句执行结果如图 6-9 所示，从结果可以看出，8.12 和 8.123 分别被存储为 8.1 和 8.12。

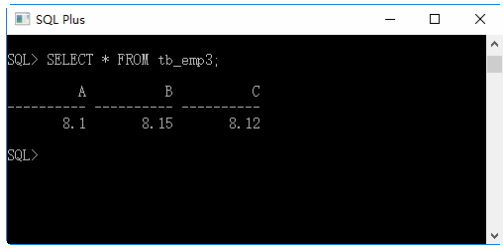

图 6-9　查看插入的数据

6.1.2　日期与时间类型

Oracle 中表示日期的数据类型主要包括 DATE 和 TIMESTAMP，具体含义和区别如表 6-2 所示。

表 6-2　Oracle 常用日期与时间类型

类 型 名 称	描　　述
DATE	日期（日-月-年），DD-MM-YY(HH-MI-SS)，用来存储日期和时间，取值范围是公元前 4712 年到公元 9999 年 12 月 31
TIMESTAMP	日期（日-月-年），DD-MM-YY(HH-MI-SS:FF3)，用来存储日期和时间，与 date 类型的区别就是显示日期和时间时更精确，date 类型的时间精确到秒，而 timestamp 的数据类型可以精确到小数秒，timestamp 存放日期和时间还能显示上午、下午和时区

【例 6-4】创建数据表 tb_emp4，定义数据类型为 date 的字段 d，向表中插入值'12-4 月-2018'，在 SQL Plus 窗口中输入创建表 tb_emp4 的 SQL 语句如下：

```
CREATE TABLE tb emp4
(
   id         NUMBER(10),
   name       VARCHAR2(25),
```

```
    birthday  date,
    tel       NUMBER(11),
    address   VARCHAR2(25)
);
```

按 Enter 键，语句执行结果如图 6-10 所示，即可看到创建好的表。

在插入数据之前，需要知道数据库默认的时间格式，在 SQL Plus 窗口中输入查询系统时间格式的 SQL 语句如下：

```
SQL> select sysdate from dual;
```

按 Enter 键，语句执行结果如图 6-11 所示，可以看到系统默认的时间格式。

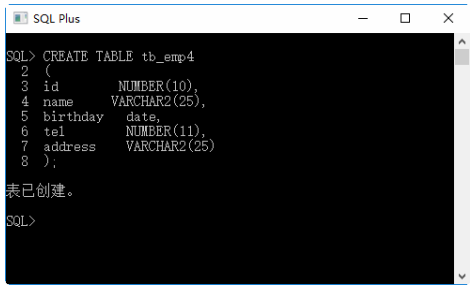

图 6-10　创建表 tb_emp4

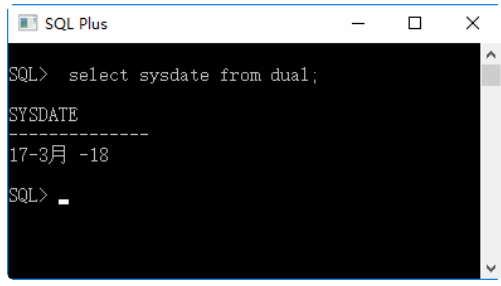

图 6-11　查询系统时间格式

向表中插入时间数据，在 SQL Plus 窗口中输入的 SQL 语句如下：

```
SQL> INSERT INTO tb_emp4(birthday) values('12-4 月-2018');
```

按 Enter 键，语句执行结果如图 6-12 所示，即可创建 1 行。

查看输入的时间数据，在 SQL Plus 窗口中输入的 SQL 语句如下：

```
SQL> SELECT * FROM tb_emp4;
```

按 Enter 键，语句执行结果如图 6-13 所示，即可看到创建的表内容。

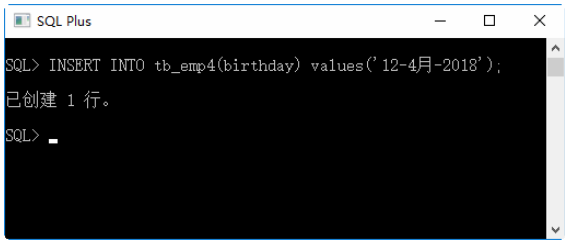

图 6-12　向表中插入时间数据

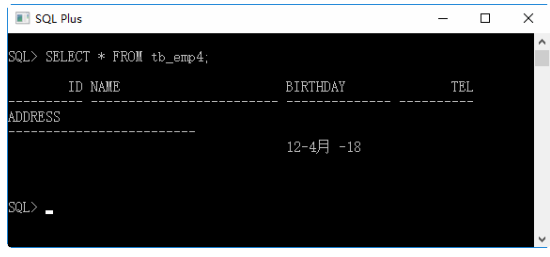

图 6-13　查看输入的时间数据

如果用户想按照指定的格式输入时间，需要修改时间的默认格式。例如，输入格式为年-月-日，修改的 SQL 语句如下：

```
SQL> alter session set nls_date_format='yyyy-mm-dd';
```

按 Enter 键，语句执行结果如图 6-14 所示，即可看到会话已更改的信息提示。

然后查看输入的时间数据，可以看到时间格式发生了改变，如图 6-15 所示。

【例 6-5】创建数据表 tb_emp5，定义数据类型为 DATE 的字段 d，向表中插入“YYYY-MM-DD”和“YYYYMMDD”字符串格式日期，在 SQL Plus 窗口中输入的 SQL 语句如下：

```
CREATE TABLE tb emp5
(
    name      VARCHAR2(25),
```

```
    birthday  date,
    tel       NUMBER(11)
);
```

按 Enter 键，语句执行结果如图 6-16 所示，即可看到成功创建表。

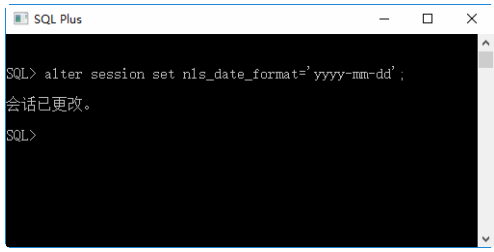

图 6-14　修改时间格式

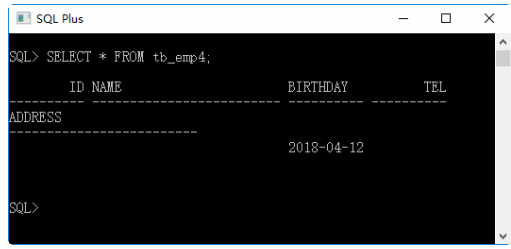

图 6-15　查看输入的时间数据

修改日期的默认格式，SQL 语句如下：

```
SQL> alter session set nls_date_format='yyyy-mm-dd';
```

按 Enter 键，语句执行结果如图 6-17 所示。

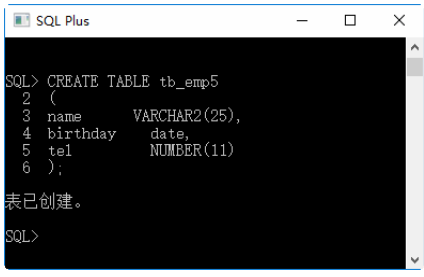

图 6-16　创建表 tb_emp5

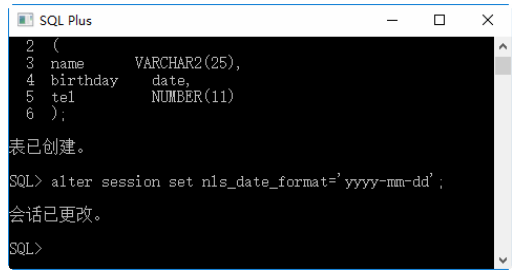

图 6-17　修改默认的日期格式

向表中插入“YYYY-MM-DD”格式日期：

```
SQL> INSERT INTO tb_emp5(birthday) values('2018-05-08');
```

按 Enter 键，语句执行结果如图 6-18 所示。

向表中插入“YYYYMMDD”格式日期：

```
SQL> INSERT INTO tb_emp5 (birthday) values('20180408');
```

按 Enter 键，语句执行结果如图 6-19 所示。

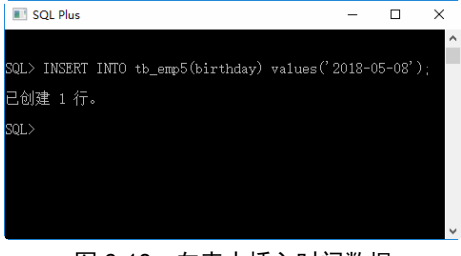

图 6-18　向表中插入时间数据

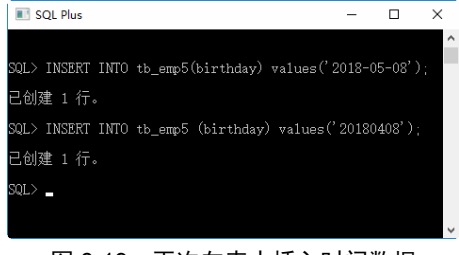

图 6-19　再次向表中插入时间数据

查看插入日期数据结果：

```
SQL> SELECT * FROM tb_emp5;
```

按 Enter 键，语句执行结果如图 6-20 所示，从运算结果中可以看出，各个不同类型的日期值都正确地

插入到了数据表中。

【例 6-6】创建表 tb_emp6 并向表 tb_emp6 中插入系统当前日期。首先创建表，SQL 语句如下：

```
CREATE TABLE tb emp6
(
   day    date
);
```

按 Enter 键，语句执行结果如图 6-21 所示。

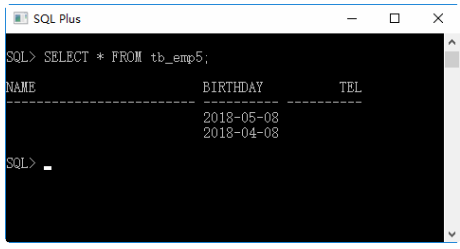

图 6-20　查看插入的日期数据

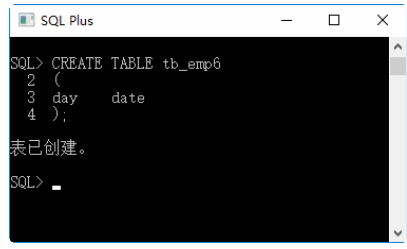

图 6-21　创建表 tb_emp6

向表中插入系统当前日期，SQL 语句如下：

```
SQL> INSERT INTO tb_emp6 values(SYSDATE);
```

按 Enter 键，语句执行结果如图 6-22 所示。

查看插入结果，SQL 语句如下：

```
SQL> SELECT * FROM tb_emp6;
```

按 Enter 键，语句执行结果如图 6-23 所示。

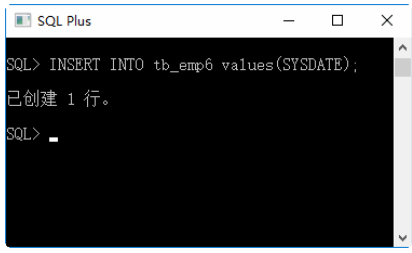

图 6-22　向表中插入系统当前日期

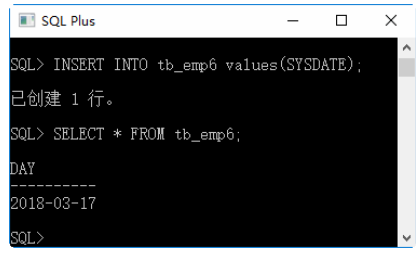

图 6-23　查询插入的结果

【例 6-7】向 tb_emp6 表中插入系统日期和时间并指定格式，首先删除表中的数据，SQL 语句如下：

```
DELETE FROM tb_emp6;
```

按 Enter 键，语句执行结果如图 6-24 所示。

向表中插入系统当前日期，SQL 语句如下：

```
SQL> INSERT INTO tb_emp6 values(to_date('2018-03-17 13:14:20','yyyy-MM-dd HH24:mi:ss') );
```

按 Enter 键，语句执行结果如图 6-25 所示。

查看插入结果，SQL 语句如下：

```
SQL> SELECT * FROM tb_emp6;
```

按 Enter 键，语句执行结果如图 6-26 所示，从运算结果中可以看出，只显示日期，时间被省略掉了。

【例 6-8】创建数据表 tb_emp7，定义数据类型为 TIMESTAMP 的字段 ts，向表中插入值'2018-9-16 17:03:00.9999'，创建数据表 tb_emp7，SQL 语句如下：

```
CREATE TABLE tb_emp7
```

```
    ( ts   TIMESTAMP
    );
```

按 Enter 键，语句执行结果如图 6-27 所示。

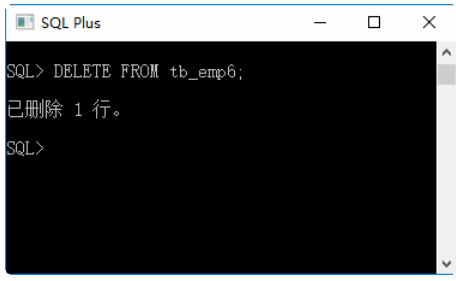

图 6-24　删除表中数据

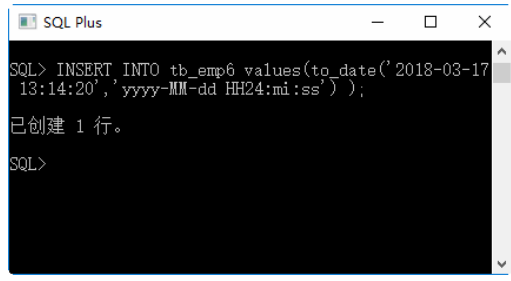

图 6-25　向表中插入系统当前日期

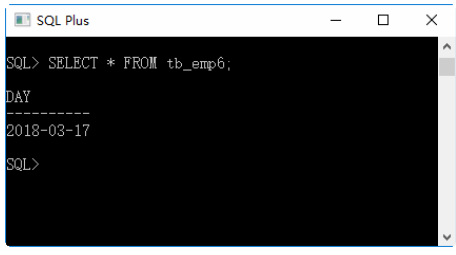

图 6-26　查询插入的日期数据

图 6-27　创建表 tb_emp7

向表中插入数据，SQL 语句如下：

```
INSERT INTO tb_emp7 values (to_timestamp('2018-9-16 17:03:00.9999', 'yyyy-mm-dd hh24:mi:ss:ff'));
```

按 Enter 键，语句执行结果如图 6-28 所示。

查看插入结果，SQL 语句如下：

```
SQL>SELECT * FROM tb_emp7;
TS
---------------------------------
16-9 月 -18 05.03.00.999900 下午
```

按 Enter 键，语句执行结果如图 6-29 所示。

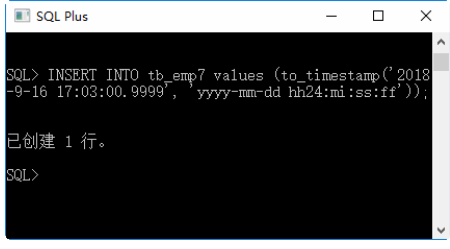

图 6-28　向表中插入数据

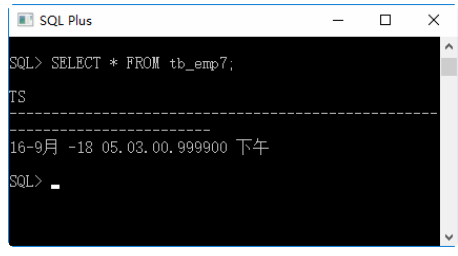

图 6-29　查询插入的数据

6.1.3　字符串类型

字符串类型用来存储字符串数据，包括 CHAR、NCHAR、VARCHAR2、NVARCHAR2 和 LONG 5 种，如表 6-3 所示。

表 6-3　Oracle 中字符串数据类型

类 型 名 称	说　　明	取值范围/B
CHAR	固定长度字符串	0～2000
NCHAR	根据字符集而定的固定长度字符串	0～1000
VARCHAR2	可变长度的字符串	0～4000
NVARCHAR2	根据字符集而定的可变长度字符串	0～1000
LONG	超长字符串	0～2G

VARCHAR2、NVARCHAR2 和 LONG 类型是变长类型，对于其存储需求取决于列值的实际长度，而不是取决于类型的最大可能尺寸。例如，一个 VARCHAR2(10)列能保存最大长度为 10 个字符的一个字符串，实际的存储需要是字符串的长度。

【例 6-9】创建数据表 tb_emp8，定义字段 ch 和 vch 的数据类型依次为 CHAR(4)、VARCHAR2(4)，向表中插入数据“ab”，创建表 tb_emp8，SQL 语句如下：

```
CREATE TABLE tb_emp8(
    ch    CHAR(4),
    vch   VARCHAR2(4)
);
```

按 Enter 键，语句执行结果如图 6-30 所示，即可完成表的创建。

输入表数据，SQL 语句如下：

```
INSERT INTO tb_emp8 VALUES('ab', 'ab');
```

按 Enter 键，语句执行结果如图 6-31 所示，即可完成行的创建。

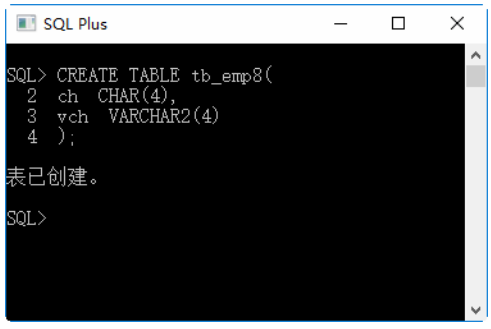

图 6-30　创建表 tb_emp8

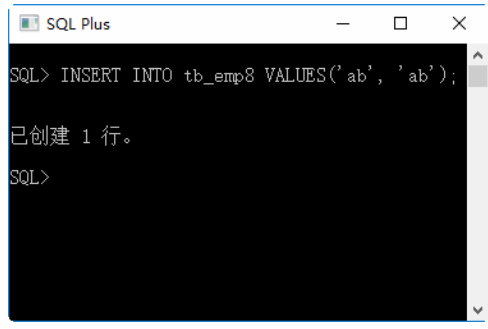

图 6-31　插入表数据

查询 ch 字段的存储长度，执行 SQL 语句如下：

```
SQL> Select length(ch) from tb_emp8;
```

按 Enter 键，语句执行结果如图 6-32 所示，即可查看 ch 字段的存储长度。

查询 vch 字段的存储长度，执行 SQL 语句如下：

```
SQL> Select length(vch) from tb_emp8;
```

按 Enter 键，语句执行结果如图 6-33 所示，即可查看 vch 字段的存储长度。

提示：从上述两个实例可以看出，固定长度字符串在存储时长度是固定的，而变长字符串的存储长度根据实际插入的数据长度而定。

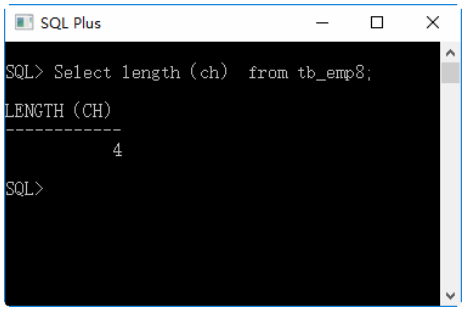

图 6-32　查询字段 ch 的存储长度

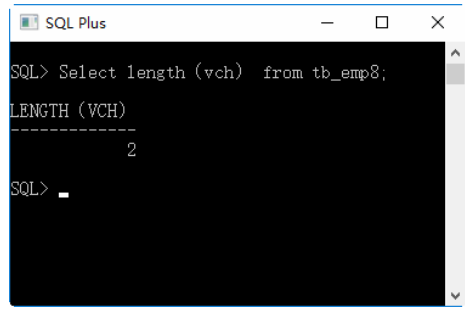

图 6-33　查询字段 vch 的存储长度

6.1.4　其他数据类型

除上面介绍的数值类型、日期与时间类型和字符串类型外，Oracle 还支持其他数据类型，如表 6-4 所示。

表 6-4　Oracle 支持的其他数据类型

类　型	含　义	存 储 描 述
RAW	固定长度的二进制数据	最大长度 2000B
LONG RAW	可变长度的二进制数据	最大长度 2GB
BLOB	二进制数据	最大长度 4GB
CLOB	字符数据	最大长度 4GB
NCLOB	根据字符集而定的字符数据	最大长度 4GB
BFILE	存放在数据库外的二进制数据	最大长度 4GB
ROWID	数据表中记录的唯一行号	10B
NROWID	二进制数据表中记录的唯一行号	最大长度 4000B

【例 6-10】创建数据表 tb_emp9，并插入一个固定长度的二进制数据，创建数据表，SQL 语句如下：

```
CREATE TABLE tb_emp9(
    ra  RAW(4)
);
```

按 Enter 键，语句执行结果如图 6-34 所示，即可完成表的创建。

输入表数据，SQL 语句如下：

```
INSERT INTO tb_emp9 VALUES('101010');
```

按 Enter 键，语句执行结果如图 6-35 所示，即可完成表数据的输入。

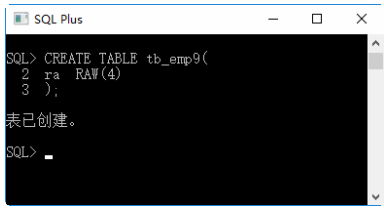

图 6-34　创建表 tb_emp9

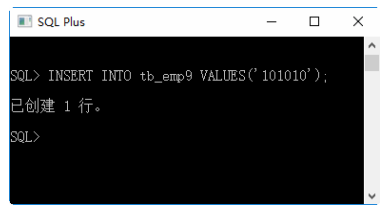

图 6-35　向表中插入数据

查询 ra 字段的存储长度，执行 SQL 语句如下：

```
Select length(ra) from tb_emp9;
```

按 Enter 键，语句执行结果如图 6-36 所示，即可查询 ra 字段的存储长度。

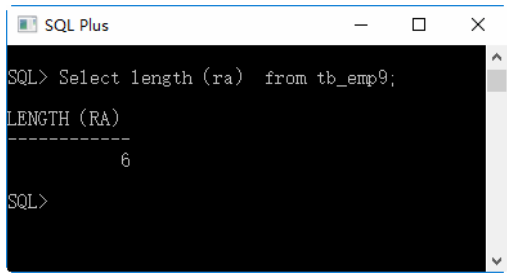

图 6-36　查询字段 ra 字段的存储长度

6.2　数据类型的选择

Oracle 提供了大量的数据类型，为了优化存储，提高数据库性能，在任何情况下均应使用最精确的类型。即在所有可以表示该列值的类型中，该类型使用的存储最少。

1. 整数和小数

数值数据类型只有 NUMBER 型，但是 NUMBER 功能不小，它可以存储正数、负数、零、定点数和精度为 30 位的浮点数。其格式为 number（m，n），其中 m 为精度，表示数字的总位数，范围为 1～38；n 为范围，表示小数点右边的数字的位数，范围为-84～127。

如果不需要小数部分，则使用整数来保存数据，可以定义为 number（m，0）或者 number（m）；如果需要表示小数部分，则使用 number（m，n）。

2. 日期与时间类型

如果只需要记录日期，则可以使用 DATE 类型。如果需要记录日期和时间，可以使用 IMESTAMP 类型。特别是需要显示上午、下午或者时区时，必须使用 IMESTAMP 类型。

3. 字符类型之间选择

CHAR 是固定长度字符，VARCHAR 是可变长度字符；CHAR 会自动补齐插入数据的尾部空格，VARCHAR 不会补齐尾部空格。

CHAR 是固定长度，所以，它的处理速度比 VARCHAR2 要快，它的缺点是浪费存储空间。所以，对存储不大，但在速度上有要求的可以使用 CHAR 类型；反之，可以使用 VARCHAR2 类型来实现。

6.3　常见运算符介绍

运用运算符可以更加灵活地使用表中的数据，常见的运算符类型有算术运算符、比较运算符、逻辑运算符、位运算符等。

6.3.1　算术运算符

算术运算符是 SQL 中最基本的运算符，用于各类数值运算，包括加（+）、减（-）、乘（*）、除（/），如表 6-5 所示。

表 6-5　Oracle 中的算术运算符

运　算　符	作　　用
+	加法运算
-	减法运算
*	乘法运算
/	除法运算，返回商

下面分别讨论不同算术运算符的使用方法。

【例 6-11】创建表 tb_emp10，定义数据类型为 NUMBER 的字段 num，插入值 64，对 num 值进行算术运算。

首先创建表 tb_emp10，输入 SQL 语句如下：

```
CREATE TABLE tb_emp10
( num    NUMBER
);
```

按 Enter 键，语句执行结果如图 6-37 所示，即可完成表的创建。

向字段 num 插入数据 50，SQL 语句如下：

```
INSERT INTO tb_emp10 values(50);
```

按 Enter 键，语句执行结果如图 6-38 所示，即可完成数据的插入。

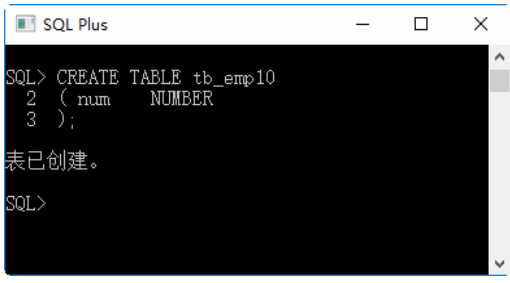

图 6-37　创建表 tb_emp10

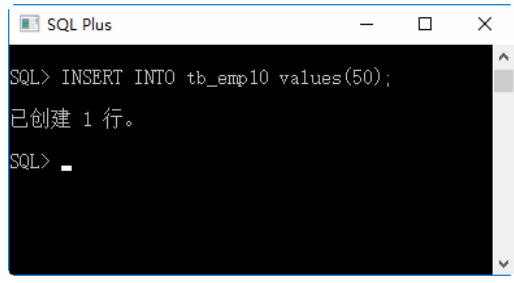

图 6-38　向表中插入数据

接下来，对 num 值进行加法和减法运算，SQL 语句如下：

```
SQL> SELECT num, num+10, num-3+5, num+5-3, num+36.5 FROM tb_emp10;
```

按 Enter 键，语句执行结果如图 6-39 所示，即可完成数据的加法和减法运算。

由计算结果可以看到，可以对 num 字段的值进行加法和减法运算，而且由于“+”和“-”的优先级相同，因此，先加后减和先减后加的结果是相同的。

【例 6-12】对 tb_emp10 表中的 num 进行乘法、除法运算。

```
SQL> SELECT num, num *2, num /2, num/3 FROM tb_emp10;
```

按 Enter 键，语句执行结果如图 6-40 所示。从运算结果中可以看出，对 num 进行除法运算时，由于 50

无法被 3 整除，因此，Oracle 对 num/3 求商的结果保存到了小数点后面 7 位，结果为 16.6666667。

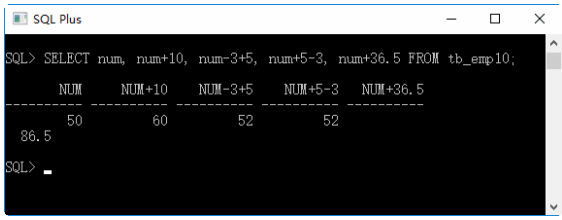

图 6-39　完成数据的加减运算

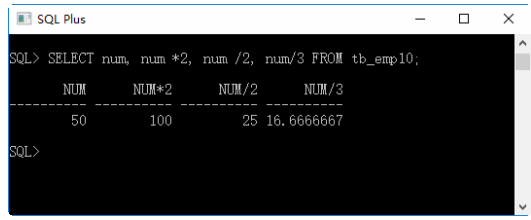

图 6-40　对数据进行乘法与除法运算

在数学运算时，除数为 0 的除法是没有意义的，因此，除法运算中的除数不能为 0，如果被 0 除，则返回错误提示信息。

【例 6-13】用 0 除 num。

```
SQL> SELECT num/0 FROM tb_emp10;
```

按 Enter 键，语句执行结果如图 6-41 所示。

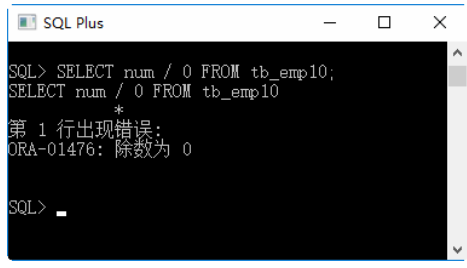

图 6-41　用 0 处于数值的错误提示

6.3.2　比较运算符

比较运算符用于比较运算，包括大于（>）、小于（<）、等于（=）、大于或等于（>=）、小于或等于（<=）、不等于（!=），以及 IN、BETWEEN…AND、IS NULL、LIKE 等。

比较运算符经常在 SELECT 的查询条件子句中使用，用来查询满足指定条件的记录。Oracle 中的比较运算符如表 6-6 所示。

表 6-6　Oracle 中的比较运算符

运　算　符	作　　用
=	等于
<=>	安全的等于
< > (!=)	不等于
<=	小于或等于
>=	大于或等于
>	大于
IS NULL	判断一个值是否为 NULL
IS NOT NULL	判断一个值是否不为 NULL
BETWEEN AND	判断一个值是否落在两个值之间

续表

运　算　符	作　　用
IN	判断一个值是 IN 列表中的任意一个值
NOT IN	判断一个值不是 IN 列表中的任意一个值
LIKE	通配符匹配

下面分别讨论不同比较运算符的含义。

1．等于运算符=

等号“=”用来判断数字、字符串和表达式是否相等。

2．不等于运算符!=

“!=”用于判断数字、字符串、表达式不相等的判断。

3．小于或等于运算符<=

“<=”用来判断左边的操作数是否小于或者等于右边的操作数。

4．小于运算符<

“<”运算符用来判断左边的操作数是否小于右边的操作数。

5．大于或等于运算符 >=

“>=”运算符用来判断左边的操作数是否大于或者等于右边的操作数。

6．大于运算符>

“>”运算符用来判断左边的操作数是否大于右边的操作数。

7．BETWEEN…AND 运算符

BETWEEN…AND 运算符用于测试是否在指定的范围内，通常和 WHERE 字句一起使用，BETWEEN…AND 条件返回一个介于指定上限和下限之间的范围值。

例如下面的例子，选出出生在 1980—1990 年的教师姓名：

```
SELECT name FROM teacher
WHERE  birth  BETWEEN  '1980' AND '1990';
```

上述语句包含上限值和下限值，与下面的语句效果一样。

```
SELECT name FROM teacher
WHERE birth>= '1980' AND birth<= '1990';
```

8．IN 运算符

IN 运算符用来判断操作数是否为 IN 列表中的其中一个值。NOT IN 运算符用来判断操作数是否不是 IN 列表中的其中一个值。

例如选出年龄是 35 岁和 45 岁的教师：

```
SELECT name FROM teacher
WHERE  age  IN ( 35, 45 );
```

9．LIKE

LIKE 运算符用来匹配字符串。在一个学校中，教师有多位，如果想要查找符合某个条件的教师，就可以使用 LIKE 运算符进行查询。

LIKE 运算符在进行匹配时，可以使用下面两种通配符：

（1）“%”，用来代表有零个或者多个字符组成的任意顺序的字符串。

（2）“_”，只能匹配一个字符。

例如选出张姓的所有教师：

```
SELECT name FROM teacher
WHERE  name LIKE '张%';
```

6.3.3 逻辑运算符

在 Oracle 中逻辑运算符的求值所得结果均为 1（TRUE）、0（FALSE），这类运算符有逻辑非（NOT 或者!）、逻辑与（AND 或者&&）、逻辑或（OR 或者||）、逻辑异或（XOR），如表 6-7 所示。

表 6-7　Oracle 中的逻辑运算符

运　算　符	作　　用
NOT	逻辑非
AND	逻辑与
OR	逻辑或

这 3 个运算符的作用如下。

（1）NOT 运算符：又称取反运算符，NOT 通常是单目运算符，即 NOT 右侧才能包含表达式，是对结果取反，如果表达式结果为 True，那么 NOT 的结果就为 False；否则，如果表达式的结果为 False，那么 NOT 的结果就为 True。

NOT 运算符后面常常和 IN、LIKE、BETWEEN…AND 和 NULL 等关键字一起使用。

例如，选择学生年龄不是 25 或者 26 的学生姓名：

```
SELECT name FROM student
WHERE  age  IN(25, 26);
```

（2）AND 运算符：对于 AND 运算符来说，要求两边的表达式结果都为 True，因此，通常称为全运算符，如果任何一方的返回结果为 NULL 或 False，那么逻辑运算的结果就为 False，也就是说记录不匹配 WHERE 子句的要求。

例如，选择学生年龄是 25 而且是姓张的学生姓名：

```
SELECT name FROM student
WHERE  age=25 AND  name LIKE '张%';
```

（3）OR 运算符：OR 运算符又称或运算符，也就是说，只要左右两侧的布尔表达式任何一方为 True，结果就为 True。

例如，选择学生年龄是 25 或者姓张的学生姓名：

```
SELECT name FROM student
WHERE  age=25 OR  name LIKE '张%';
```

这样，无论年龄为 25 的学生还是姓张的学生，都会被选择出来。

6.3.4 位运算符

位操作运算符是参与运算的操作数，按二进制位进行运算，包括位与（&）、位或（|）、位非（～）、位异或（^）、左移（<<）、右移（>>）6 种，如表 6-8 所示。

表 6-8　Oracle 中的位运算符

运　算　符	作　　用
位与（&）	位于运算
位或（|）	位或运算
位非（～）	位非运算
位异或（^）	位异或运行
左移（<<）	左移运算
右移（>>）	右移运算

6.3.5　运算符的优先级

运算符的优先级决定了不同的运算符在表达式中计算的先后顺序，表 6-9 列出了 Oracle 中的各类运算符及其优先级。

表 6-9　运算符按优先级由低到高排列

优　先　级	运　算　符
最低	=（赋值运算），:=
	OR
	AND
	NOT
	=（比较运算），<=>，>=，>，<=，<，<>，!= ，IS，LIKE， REGEXP，IN
	&
	<<，>>
	–，+
	*，/
	–（负号）
最高	!

可以看到，不同运算符的优先级是不同的。一般情况下，级别高的运算符先进行计算，如果级别相同，Oracle 按表达式的顺序从左到右依次计算。当然，在无法确定优先级的情况下，可以使用圆括号（）来改变优先级，并且这样会使计算过程更加清晰。

6.4　就业面试技巧与解析

6.4.1　面试技巧与解析（一）

面试官：何时可以到职？

应聘者：如果被录用的话，随时都可以任职。

6.4.2 面试技巧与解析（二）

面试官：如何适应办公室工作的新环境？

应聘者：我想我应该从以下三个方面来适应办公室新环境：首先办公室里每个人有各自的岗位与职责，不得擅离岗位；其次，根据领导指示和工作安排，制订工作计划，提前预备，并按计划完成；再次，多请示并及时汇报，遇到不明白的要虚心请教；最后，抓间隙时间，多学习，努力提高自己的政治素质和业务水平。