

第 4 章 字符串、数值及其常用函数

表中的列都有其所属的数据类型，创建表时，需要根据实际需要，指定列的数据类型，查询或修改表中的数据时，要根据其数据类型采用合适的操作方式。

本章主要包括：

- 字符串类型
- 数值类型
- 常用字符串处理函数
- 常用数值处理函数
- 字符串及数值类型转换函数

4.1 字符串类型

常用数据类型包括字符串、数值及日期时间类型，本章主要说明前两种数据类型，日期时间类型请参考下一章内容。

下表是 SQL:2011 标准字符串数据类型及其对应的 Oracle 与 SQL Server 支持的字符串数据类型，对于 SQL:2011 标准数据类型，Oracle 与 SQL Server 都是支持的。

char(n)及 nchar(n)为定长字符串类型，即实际值的长度小于 n 时，用空格填满，占用空间总是 n。Oracle 的 varchar2(n)、nvarchar2(n)以及 SQL Server 的 varchar(n)、nvarchar(n)为可变长度字符串类型，当实际值的长度小于 n 时，不用空格填满，占用空间为实际值的长度。

表 4-1 字符串类型对比

SQL:2011	Oracle		SQL Server	
character(n)	char(n)	$1 \leq n \leq 2000$	char(n)	$1 \leq n \leq 8000$
char(n)				
character varying(n)	varchar2(n)	$1 \leq n \leq 4000$	varchar(n)	$1 \leq n \leq 8000$
char varying(n)				
national character(n)	nchar(n)	$1 \leq n \leq 1000$	nchar(n)	$1 \leq n \leq 4000$
national cahr(n)				
nchar(n)				
national character varying(n)	nvarchar2(n)	$1 \leq n \leq 2000$	nvarchar(n)	$1 \leq n \leq 4000$
national char varying(n)				
nchar varying(n)				

char(n)、varchar2(n)以及 varchar(n)使用数据库字符集(national character set)存储字符串，长度单位默认为字节。Oracle 12c 之前的版本中，n 的最大值为 4000 字节。Oracle 12c 中，若 max_string_size 设置为 extended，n 的最大值为 32767，若 max_string_size 设置为 standard，n 的最大值仍为 4000。另外，使用 varchar2(n)定义列的类型时，不能省略参数 n。

值得注意的是，在 Oracle 中的可变长度字符串类型是用 varchar2(n)及 nvarchar2(n)表示，这与 SQL 标准及 SQL Server 都不同，若使用 varchar(n)类型，则 Oracle 会自动存储为 varchar2(n)，在以后版本，Oracle 可能会改变这种情况，建议使用 varchar2(n)。

nchar(n)、nvarchar2(n)以及 nvarchar(n)使用国家字符集(national character set)存储字符串，国家字符集只能使用 Unicode 编码形式存储字符，在创建数据库时指定，用于存储数据库字

符集不包括的字符。对于 Oracle 数据库，可选的国家字符集为 UTF8 和 AL16UTF16，默认为 AL16UTF16。n 为其允许的最大长度，长度单位为字符，而不是字节，在存储范围方面，字符数最大值 n 不能超过此类型所能存储的最大字节数，不同国家字符集存储一个字符占用的字节数可能不同，因此对于不同的国家字符集，n 的最大取值也会不同。在 SQL Server 中，这两类数据的实际存储长度一般是 2n。

以下几个示例说明 Oracle 与 SQL Server 对 SQL:2011 标准字符串数据类型在形式上都是支持的。

在 Oracle 中执行下面建表命令，使用 SQL:2011 标准字符串数据类型：

```
SQL> create table t
2 (
3   a varchar(5),
4   b character varying(5),
5   c national character varying(5)
6 )
7 /

表已创建。

SQL> desc t
名称                                是否为空? 类型
-----
A                                VARCHAR2 (5)
B                                VARCHAR2 (5)
C                                NVARCHAR2 (5)
```

在 SQL Server 中执行上述相同命令：

```
1> create table t
2> (
3>   a varchar(5),
4>   b character varying(5),
5>   c national character varying(5)
6> )
7> go
1> describe t
2> go
column_name type          nullable
-----
a          varchar(5)
b          varchar(5)
c          nvarchar(1)
```

这里的 describe 存储过程的定义请参考第 2 章内容。

4.2 数值数据类型

下表是 SQL:2011 标准数值类型、Oracle 与 SQL Server 所支持数值类型的大致对比，SQL Server 部分给出了相关类型的表示范围，要注意各对应数据类型的关系并不是等同，只是大致相似：

表 4-2 数值类型对比

SQL:2011	Oracle	SQL Server
numeric[p,s]	number(p,s)	numeric[p,s]

decimal[(p,s)]			
	number(19, 4)	money	-922,337,203,685,477.5808~ 922,337,203,685,477.5807
	number(10, 4)	small money	-214,748.3648~ 214,748.3647
integer	number(38)	bigint	-9,223,372,036,854,775,808~ 9,223,372,036,854,775,807
		int	-2,147,483,648~ 2,147,483,647
samllint		smallint	-32,768~32,767
		tinyint	0~255
float	float(126)	float	默认为 float(53)
double precision		float(53)	
real	float(63)	real	等同于 float(24)
	number		
	binary_float		
	binary_double		

4.2.1 定点数值数据类型 number(p[,s])与 numeric[(p,s)]

定点数值数据类型在 Oracle 中使用 number(p[,s]), 在 SQL Server 中使用 numeric[(p,s)], numeric 也是 SQL:2011 标准类型。

关于 Oracle 的 number[(p,s)]中的 p 和 s 参数有以下特点:

- p 和 s 的范围: $1 \leq p \leq 38$, $-84 \leq s \leq 127$ 。
- 若省略 s, 则默认为 0, 如 number(4)即表示 number(4, 0)。
- 若 $s > 0$, 则精确到小数点右边 s 位, 并四舍五入, 然后检验有效数位是否不超过 p, 若超过, 则报错。若四舍五入后, 有效数位不超过 p, 且 $s > p$, 则小数点右边至少有 $s-p$ 个 0 填充, 如 0.01234 或 0.00012 都满足 number(4, 5)的要求。
- 若 $s < 0$, 则精确到小数点左边 s 位, 并四舍五入。然后检验有效数位是否不超过 $p+|s|$, 如果超过, 则报错。
- 若使用 number 时, 未附带 p 及 s, 则 number 与 float 是相同的, 即 float(126), 主要用于表示浮点数, 其含义请参考本节后面内容。

SQL Server 的 numeric[(p,s)]中的 p 和 s 有以下特点:

- p 和 s 的范围: $1 \leq p \leq 38$, $0 \leq s \leq p$ 。
- 若省略 s, 则默认为 0, 如 number(4)即表示 number(4, 0)。
- 若未附带 p 及 s, 则 numeric 表示 numeric(18), 只能表示整数, 与 Oracle 中不附带 p、s 参数的 number 类型显然不同, Oracle 的 number 类型相当于 float 或 float(126), 主要用于表示浮点数。

另外, SQL Server 还支持特殊的定点数值数据类型 money 与 smallmoney, 适合用于货币值运算, 大致相当于 numeric(19,4)与 numeric(10,4)。

4.2.2 整型

Oracle 支持 SQL 标准类型 numeric、decimal、integer、samllint, 都统一转换为 number(38)存储。除了这些标准类型外, SQL Server 还支持 bigint 及 tinyint, 可以使用户对整型数值的存储进行更精细的控制。

4.2.3 浮点型

Oracle 的浮点数据类型 float(n)中的 n 表示二进制精度，最大为 126，这也是默认值，number 类型与 float 等价。二进制精度的 126 大致相当于十进制精度的 38，二进制精度和十进制精度的换算关系为： $d=\text{round}(b*0.30103)$ 。

如 1234.5678 按照 float(10)存储，二进制精度 10 按照上面公式转换为十进制精度 3，数值 1234.5678 以科学计数法表示为 1.2345678E+3，按照十进制精度 3 处理 1.2345678 得到 1.235，这样，按照 float(10)，最后存储的数值为 1.235E+3，即 1235；如果按照 float(18)存储，则按照上述步骤，二进制精度 18 对应十进制精度 5，最后存储的是 1.23457*E+3，即 1234.57。

上面结论，可以通过下面实验验证：

```
SQL> create table t(original_num varchar(20),f10 float(10), f18 float(18));
表已创建。
SQL> insert into t values('1234.5678', 1234.5678, 1234.5678);
已创建 1 行。
SQL> select * from t;
```

ORIGINAL_NUM	F10	F18
1234.5678	1235	1234.57

在 Oracle 中使用 SQL 标准数据类型 float 或 double precision 时，转换为 float(126)，使用标准类型 real 时，转换为 float(63)。

Oracle 的 binary_float 与 binary_double 数据类型是 Oracle 10g 引入的用于表示浮点数值的新类型，执行复杂的科学数值计算时，相比使用 number 或 float 数据类型存储的数据，运算速度会有显著提高。

Oracle 的 binary_float 是 32-bit 的单精度浮点型数值数据类型。每一个 binary_float 值需要 5 个字节，包括一个字节表示长度。

Oracle 的 binary_double 是 64-bit 的双精度浮点型数值数据类型。每一个 binary_double 值需要 9 个字节，包括 1 个字节表示长度。

两种数据类型的各个部分所占空间如下表所示。

表 4-3 binary_float 与 binary_double 类型

类型	符号(bits)	小数(bits)	指数(bits)
binary_float	1	23	8
binary_double	1	52	11

下面是使用两种数据类型的示例。

创建表 t:

```
SQL> create table t
2 (
3   num_origin varchar(30),
4   num_binary_float binary_float,
5   num_binray_double binary_double
6 )
7 /
```

添加如下记录:

```
SQL> insert into t values
2 (
3   '123456789123456789.123456789',
```

```

4      123456789123456789.123456789,
5      123456789123456789.123456789
6 )
7 /

```

添加上述数值后，在表中存储的数据为：

```

SQL> set numwidth 25
SQL> select * from t;
NUM_ORIGIN                                NUM_BINARY_FLOAT    NUM_BINRAY_DOUBLE
-----
123456789123456789.123456789              1.23456791E+017     1.2345678912345678E+017

```

SQL Server 的 float(n) 中的 n 表示二进制精度，n 的取值范围从 1 到 53，符合下面规定：

- 若 $1 \leq n \leq 24$ 时，则按 24 对待，大致相当于十进制精度 7。
- 若 $25 \leq n \leq 53$ 时，则按 53 对待，大致相当于十进制精度 15。

如数值 1234567812345678.1234，按 float(24) 存储，则转换为 1.2345678E+15，如按 float(53) 存储，则转换为 1.234567812345678E+15，即 1234567812345678.0。

上述结论，可验证如下：

```

1> create table t(f1 float(1), f24 float(24), f25 float(25), f53 float(53))
2> go
1> insert into t values
2> (
3>     1234567812345678.1234, 1234567812345678.1234,
4>     1234567812345678.1234, 1234567812345678.1234
5> )
6> go

(1 行受影响)
1> select * from t
2> go
2> go
f1              f24              f25              f53
-----
1.2345678E+15   1.2345678E+15      1234567812345678.0  1234567812345678.0

```

4.3 在 SQL Server 中查询数据类型对应关系

在 SQL Server 中提供了两个视图可以得到 SQL Server 的数据类型与 DB2、Oracle 以及 Sybase 数据类型之间的对应关系：

```

1> select * from msdb.dbo.sysdatatypemappings
2> go

```

或

```

1> select * from msdb.dbo.MSdatatype_mappings
2> go

```

4.4 常用字符串处理函数

下表是 Oracle 与 SQL Server 中的常用字符串处理函数的一个总体对比：

表 4-5 字符串函数对比

功能	Oracle	SQL Server
合并字符串		+或 concat
返回 ascii 码/字符	ascii/chr	ascii/char
返回字母的大/小写	lower/upper	lower/upper
返回字符串长度	length	len
裁剪字符串左侧/右侧指定字符	ltrim/rtrim	ltrim/rtrim
代替指定字符串	replace	replace
返回字符串子串	substr	substring

4.4.1 字符串合并

字符串合并，Oracle 使用 “||” 运算符，SQL Server 使用 “+” 运算符或 concat() 函数。

如果参与字符串合并操作的既有字符串又有数值数据或日期时间数据，Oracle 会自动将数值数据或日期时间数据转换为字符串常量，SQL Server 却会把 “+” 符号作为数值运算或日期时间的 “加” 运算符，并尝试把字符串转换为数值数据或日期时间数据完成求和运算，而不会把数值数据或日期时间数据自动转换为字符串常量，这种情况下，需要使用 cast 或 convert 函数把数值数据或日期时间数据显式转换为字符串常量。SQL Server 的 concat() 函数与 Oracle 的 “||” 运算符功能相似，会把参与合并的数值或日期时间数据自动转换为字符串。

若参与合并操作的数据都是字符串，则 Oracle 的 “||” 和 SQL Server 的 “+” 运算符的用法相同，若数值数据或日期时间数据参与合并操作，则两者的用法还是有明显不同的。下面是简单示例。

Oracle 能够自动把数值数据转换为字符串常量，不需要使用转换函数：

```
SQL> select ename || ' ' || job || ' ' || hiredate || ' ' || sal as emp_info
2  from emp
3  where empno=7369
4  /
```

EMP_INFO

SMITH CLERK 17-12 月-80 900

同样的操作，SQL Server 却不支持：

```
1> select ename + ' ' + job + ' ' + hiredate + sal as emp_info
2> from emp
3> where empno=7369
4> go
```

消息 241，级别 16，状态 1，服务器 APPLE，第 1 行
从字符串转换日期和/或时间时，转换失败。

而必须使用数据类型转换函数显式转换：

```
1> select ename+' '+job+' '+cast(hiredate as varchar)+' '+
2>         cast(sal as varchar) as emp_info
3> from emp
4> where empno=7369
5> go
emp_info
```

SMITH CLERK 12 17 1980 12:00AM 800.00

如果 cast 函数的 varchar 未附加长度，则默认为 30，上面查询中的 cast(hiredate as varchar)

与 cast(hiredate as varchar(30))等价， cast(sal as varchar)与 cast(sal as varchar(30))等价。

SQL Server 的 concat()函数用法与 Oracle 的 “||” 运算符相似，使用时不需要把数值型或日期型数据转换为字符串，下面是简单示例：

```
1> select empno,
2> concat(ename, ' ', job, ' ', hiredate, ' ', sal) as emp_info
3> from emp
4> where empno=7369
5> go
empno emp_info
-----
7369 SMITH CLERK 12 17 1980 12:00AM 800.00
```

4.4.2 ascii 码与字符的转换

Oracle 使用 ascii 与 chr 分别求出一个字符的 ascii 码以及一个 ascii 码对应的字符，同样的功能，SQL Server 使用 ascii 与 char。

在 Oracle 中使用 ascii 及 chr：

```
SQL> select ascii('a'), chr(97) from dual;

          ASCII('A') C
          -----
                97 a
```

在 SQL Server 中使用 ascii 及 char：

```
1> select ascii('a'), char(97)
2> go

-----
          97 a
```

chr 及 char 的一个很有用的功能是在字符串中插入控制字符，如 ascii 码 9 表示制表符，10 表示换行符，13 表示回车符。

在 Oracle 中执行：

```
SQL> select ename||chr(10)||chr(13)||job||chr(10)||chr(13) as emp_info
2 from emp
3 where deptno=10
4 /

EMP_INFO
-----
CLARK
MANAGER

KING
PRESIDENT

MILLER
CLERK
```

在 SQL Server 中执行：

```
1> select ename+char(10)+char(13)+job+char(10)+char(13) as emp_info
```

```

2> from emp
3> where deptno=10
4> go
emp_info
-----
CLARK
MANAGER

KING
PRESIDENT

MILLER
CLERK

```

4.4.3 字符串大小写转换

字符串的大小写转换使用 upper 与 lower 函数, 在 Oracle 和 SQL Server 中这两个函数的用法相同。

在 Oracle 中执行:

```

SQL> select upper(ename), lower(ename)
  2  from emp
  3  where deptno=10
  4  /

UPPER(ENAM) LOWER(ENAM)
-----
CLARK      clark
KING       king
MILLER     miller

```

在 SQL Server 中执行:

```

1> select upper(ename), lower(ename)
2> from emp
3> where deptno=10
4> go

-----
CLARK      clark
KING       king
MILLER     miller

```

4.4.4 求字符串长度

Oracle 使用 length 函数求字符串长度, SQL Server 使用 len 函数。

在 Oracle 中:

```

SQL> select ename, length(ename)
  2  from emp
  3  where deptno=10
  4  /

ENAME      LENGTH(ENAME)
-----
CLARK      5
KING       4
MILLER     6

```


在 SQL Server 中:

```
1> select ename, len(ename)
2> from emp
3> where deptno=10
4> go
ename
-----
CLARK          5
KING           4
MILLER         6
```

4.4.5 裁剪字符串，trim 系列函数

trim 系统函数包括 trim、ltrim 及 rtrim。

Oracle 的 ltrim、rtrim 及 trim 分别用于裁剪一个字符串左侧、右侧以及两侧的指定子串。

SQL Server 不支持 trim 函数,其 ltrim 及 rtrim 的功能也与 Oracle 的对应函数用法不同,只是裁剪左侧、右侧的空格,不能指定其他形式的子串,如果要裁剪一个字符串两侧的空格,可以结合使用这两个函数。

如果 Oracle 的 ltrim 及 rtrim 用于裁剪空格,则与 SQL Server 的对应函数用法相同。

SQL Server 的 ltrim 及 rtrim 函数只需以操作的字符串为参数。

Oracle 的 ltrim 与 rtrim 用法相同: ltrim(str, substr)及 rtrim(str, substr)

- str 表示操作的字符串。
- substr 表示要裁剪的子串,若裁剪空格,则可以省略。

Oracle 的 trim 函数用法为: trim(leading | trailing | both trim_str from str)

- leading 指定裁剪字符串左侧的子串。
- trailing 指定裁剪字符串右侧的子串。
- both 指定裁剪字符串两侧的子串, both 可以省略。
- trim_str 指定要裁剪的字符,只能指定一个字符。
- str 指定要操作的字符串。

下面示例在 Oracle 中使用 ltrim 函数:

```
SQL> select ltrim('  abc'), ltrim('abcd','abc'), ltrim('xxxabcd','x')
2  from dual
3  /

LTR L LTRI
--- - ----
abc d abcd
```

下面示例在 Oracle 中使用 trim 函数:

```
SQL> select trim('  abcd  '), trim(both 'x' from 'xxabcxx') from dual;

TRIM TRI
-----
abcd abc
```

在 SQL Server 中使用这 ltrim 及 rtrim 函数:

```
1> select ltrim('  abc'), rtrim('abc '), ltrim(rtrim('  abc '))
2> go

-----
abc   abc   abc
```

4.4.6 求子字符串，substr 与 substring

Oracle 使用 substr 函数，SQL Server 使用 substring 函数求一个字符串中的子串。

Oracle 的 substr 的用法为：substr(str, position, length)

- str 表示要操作的字符串。
- position 表示开始子串开始位置，默认由左侧开始，如果 position<0，则由右侧开始。
- length 表示子串的长度。

SQL Server 的 substring 的用法与 Oracle 的 substr 类似，只是开始位置不能为负。

在 Oracle 中执行：

```
SQL> select substr('abcdefgh', 3, 2), substr('abcdefgh', -3, 2)
2   from dual
3   /

SU SU
-- --
cd fg
```

在 SQL Server 中执行：

```
1> select substring('abcdefgh', 3, 2), substring('abcdefgh', -3, 2)
2> go

-----
cd
```

4.4.7 替换指定子串

Oracle 与 SQL Server 都使用 replace 替换一个字符串中的子串，用法相同：

replace(str, search_str, rep_str)

- str 表示操作的字符串。
- search_str 表示要搜索的子字符串。
- rep_str 表示要替换的目标字符串。

在 Oracle 中执行：

```
SQL> select replace('abcdefgh', 'de', 'xxx')
2   from dual
3   /

REPLACE('A
-----
abcxxxfg
```

在 SQL Server 中执行：

```
C:\>sqlcmd -d law -Y 20
1> select replace('abcdefgh', 'de', 'xxx')
2> go

-----
abcxxxfg
```

4.5 常用数值处理函数列表

Oracle 与 SQL Server 中的常用数值处理函数如下表所示，多数函数的用法比较简单，

我们不再举例说明。

表 4-6 数值函数对比

功能	Oracle	SQL Server
返回绝对值	abs	abs
返回大于或等于指定数值表达式的最小整数	ceil	ceiling
exp(n)返回 e 的次幂	exp	exp
返回小于或等于指定数值表达式的最大整数	floor	floor
返回自然对数	ln	log
返回常用对数	log	log10
返回 m 除以 n 的余数, mod(m, n)等价于 m%n	mod	%
返回 π 值, Oracle 可以使用 acos(-1)实现		pi
power(m,n)返回 m 的 n 次幂	power	power
产生 0 到 1 之间的随机数, Oracle 使用 dbms_random 包		rand
舍入到指定长度或精度	round	round
符号函数, 对于正、负数、0 分别返回-1、1、0	sign	sign
求平方根	sqrt	sqrt
将数字截断到指定的位数	trunc	trunc

4.6 字符串及数值类型转换函数

把数值型数据转换为字符串, 在 Oracle 中可以使用 cast 函数与 to_char 函数, 在 SQL Server 中可以使用 cast 函数及 convert 函数。cast 函数在 Oracle 及 SQL Server 中的用法类似。SQL Server 中未提供与 Oracle 的 to_char 功能类似的函数。Oracle 虽然也提供了 convert 函数, 但其功能是转换字符串所属的字符集, 与 SQL Server 的 convert 函数用法截然不同。

Oracle 提供了 cast 函数与 to_number 函数把字符串转换为数值型数据。SQL Server 提供了 cast 函数与 convert 函数把字符串转换为数值型数据。

4.6.1 在 Oracle 中把数值转换为字符串

Oracle 的 cast 函数的用法: `cast(expression as data_type[(length)])`

第一个参数是要转换的表达式, 一般是表的列名或常量, data_type 为转换成的目标数据类型, length 是精度。

如果要把数值型数据转换为字符串, 则 data_type 一般为 char 或 varchar2, 如果未附加 length 参数, 则默认为 1。

下面示例把 emp 表中的 sal 列用 cast 函数转换为字符串, 并与 ename 列合并:

```
SQL> conn scott/tiger
已连接。
SQL> set pagesize 50
SQL> select ename||' '||'s sal is '||cast(sal as char(7)) as "EMPLOYEE'S SAL"
   2  from emp
   3  /

EMPLOYEE'S SAL
-----
SMITH's sal is 800
```

```
ALLEN's sal is 1600
WARD's sal is 1250
.....
```

to_char 函数的用法为: to_char(*n*, *fmt*)

第一个参数 *n* 为要转换为字符串的数值常量或数值类型的字段, 第二个参数 *fmt* 是可选的, 用于指定转换后的字符串要满足的格式。

to_char 函数返回的结果为 varchar2 类型, 其长度恰为能够容纳转换后的字符串结果。

下面我们给出几个示例。

```
SQL> select to_char(999999, '999,999') from dual;
```

```
to_char(
-----
999,999
```

上面代码把整数 999999 转换为字符串, 并在中间添加逗号分隔符。

4.6.2 在 SQL Server 中把数值转换为字符串

SQL Server 的 cast 函数与 Oracle 相同: cast(*expression* as *data_type*[(*length*)])

第一个参数是要转换的表达式, 一般是表的列名或常量, *data_type* 为转换的目标数据类型, *length* 是精度。

把数值型数据转换为字符串, *data_type* 一般为 char 或 varchar, 若未附加 *length* 参数, 则默认为 30, 这与 Oracle 默认为 1 不同。

下面示例把 emp 表中的 sal 列用 cast 函数转换为字符串, 并与 ename 列合并:

```
1> select ename+''''+s sal is '+cast(sal as char(7)) as "EMPLOYEE'S SAL"
2> from emp
3> go
EMPLOYEE'S SAL
-----
SMITH's sal is 800.00
ALLEN's sal is 1600.00
WARD's sal is 1250.00
.....
```

convert 函数主要用于把数值型或日期型数据转换为字符串, 虽然使用 convert 函数也可以把只包含数字的字符串转换为数值型数据, 但这种转换, SQL Server 一般可以隐式完成。本节我们主要说明数值型数据转换为字符串的用法。

convert 函数的用法为: convert(*data_type*[(*length*)], *expression*[,*style*])

其中, *data_type* 是要转换的目标数据类型, *expression* 是要转换的表达式, 一般为表的列名或常量, *style* 参数一般用于把日期型数据转换为字符串时指定格式。

如把整数 2008 转换为字符串, 可以使用: convert(char(4), 2008)

下面示例把 emp 表中的 sal 列用 convert 函数转换为字符串, 并与 ename 列合并:

```
1> select ename+''''+s sal is '+convert(char(7),sal) as "EMPLOYEE'S SAL"
2> from emp
3> go
EMPLOYEE'S SAL
-----
SMITH's sal is 800.00
ALLEN's sal is 1600.00
WARD's sal is 1250.00
.....
```

上面示例中的“+”号是合并字符串的运算符，`ename` 后面的 4 个单引号，表示在显示结果中要打印出一个单引号，之所以 4 个单引号，是因为在 SQL 语言中，单引号用于括住字符串常量。

4.6.3 Oracle 中把字符串转换为数值

依照 4.6.1 所提到的 `cast` 函数的用法，把字符串转换为数值，只要第一个参数替换为字符串常量或字符串类型的字段，第二个参数替换为合适的数值类型，即可以把字符串转换为数值型数据。

如把字符串 '100' 转换为整数 100 后与整数 300 相加：

```
SQL> select cast('100' as number)+300 from dual;

cast('100' ASNUMBER)+300
-----
400
```

类似的，下面示例把 100.21 转换为 `number(5,2)` 类型的数值：

```
SQL> select cast('100.21' as number(5,2))+300 from dual;

cast('100.21' ASNUMBER(5,2))+300
-----
400.21
```

上述示例，如果不使用 `cast` 函数，Oracle 默认会进行隐式转换：

```
SQL> select '100'+300 from dual;

'100'+300
-----
400
```

`to_number` 函数的用法为：`to_number(expression, fmt)`

第一个参数为要转换的字符串或字符串类型字段，第二个参数 `fmt` 可选(一般省略)，用于指定第一个参数中各个部分的格式。

以上示例可以使用 `to_number` 函数完成：

```
SQL> select to_number('100')+300 from dual;

TO_NUMBER('100')+300
-----
400
```

转换时，也可以包含小数部分：

```
SQL> select to_number('100.201')+300 from dual;

to_number('100.201')+300
-----
400.201
```

4.6.4 SQL Server 中把字符串转换为数值

与 Oracle 的 `cast` 函数用法类似，把字符串转换为数值，只要第一个参数替换为字符串常量或字符串类型的字段，第二个参数指定为合适的数值类型，即可以把字符串转换为数值型数据。

另外要注意对应于 Oracle 中的 number 数值类型，在 SQL Server 中称为 numeric。
下面示例把字符串'100'转换为 number 类型的数值：

```
1> select cast('100' as numeric)
2> go
```

```
-----
100
```

类似的，下面示例把 100.21 转换为 numeric(5,2)类型的数值：

```
1> select cast('100.21' as numeric(5,2))
2> go
```

```
-----
100.21
```

与 Oracle 类似，如果未作显式转换，SQL Server 默认也完成隐式转换：

```
1> select '100'+300
2> go
```

```
-----
400
```

依照 4.6.2 提到的 convert 函数的用法，只要把第一个参数设置为转换的目标数值类型，第二个参数设置为要转换的字符串常量或字符串类型字段，即可把字符串转换为数值数据。

下面示例使用 convert 函数完成与 4.6.3 同样的功能：

```
1> select convert(numeric(5,2), '100.21')
2> go
```

```
-----
100.21
```