

文章

[姚鑫](#) · 五月 14, 2022 阅读大约需 4 分钟

第142章 SQL函数 TO_CHAR (三)

第142章 SQL函数 TO_CHAR (三)

数字到字符串示例

以下嵌入式 SQL 示例显示了基本的数字到字符串的转换：

```
/// d ##class(PHA.TEST.SQLFunction).ToChar3()
ClassMethod ToChar3()
{
    &sql(
        SELECT
            TO_CHAR(1000, '9999'),
            TO_CHAR(10, '9999')
        INTO
            :numfull, :numshort
    )
    if SQLCODE '= 0 {
        w !, "Error code ", SQLCODE
    } else {
        w !, "Formatted number:", numfull
        w !, "Formatted number:", numshort
        w !, "Note leading blanks"
    }
}
```

```
DHC-APP>d ##class(PHA.TEST.SQLFunction).ToChar3()
```

```
Formatted number: 1000
```

```
Formatted number: 10
```

```
Note leading blank
```

返回具有适当数量的前导空格的指定数字。无符号正数前面总是有一个空白字符。如果指定数字的位数少于格式参数，则提供额外的前导空格。

以下嵌入式 SQL 示例显示了分隔符的使用：

```
/// d ##class(PHA.TEST.SQLFunction).ToChar4()
ClassMethod ToChar4()
{
    &sql(
        SELECT
            TO_CHAR(1000, '9,999.99'),
            TO_CHAR(1000, '9G999D99')
        INTO
```

```

        :comma,:groupsep
    )
    if SQLCODE != 0 {
        w !,"Error code ",SQLCODE
    } else {
        w !,"Formatted number:",comma
        w !,"Formatted number:",groupsep
        w !,"Note leading blank"
    }
}

```

```
DHC-APP>d ##class(PHA.TEST.SQLFunction).ToChar4()
```

```

Formatted number: 1,000.00
Formatted number: 1,000.00
Note leading blank

```

第一个 TO_CHAR 返回字符串'1,000.00'。第二个 TO_CHAR 也可能返回此值，但显示的分隔符取决于区域设置。

以下嵌入式 SQL 示例显示了正号和负号的使用：

```

/// d ##class(PHA.TEST.SQLFunction).ToChar5()
ClassMethod ToChar5()
{
    &sql(
        SELECT
            TO_CHAR(10, '99.99'),
            TO_CHAR(-10, '99.99'),
            TO_CHAR(10, 'S99.99'),
            TO_CHAR(-10, 'S99.99'),
            TO_CHAR(10, '99.99S'),
            TO_CHAR(-10, '99.99S')
        INTO
            :pos,:neg,:poslead,:neglead,:postrail,:negtrail
    )
    if SQLCODE != 0 {
        w !,"Error code ",SQLCODE
    } else {
        w !,"Formatted number:",pos
        w !,"Formatted number:",neg
        w !,"Formatted number:",poslead
        w !,"Formatted number:",neglead
        w !,"Formatted number:",postrail
        w !,"Formatted number:",negtrail
        w !,"Note use of leading blank"
    }
}

```

```
DHC-APP>d ##class(PHA.TEST.SQLFunction).ToChar5()
```

```

Formatted number: 10.00
Formatted number:-10.00
Formatted number:+10.00
Formatted number:-10.00

```

Formatted number:10.00+
 Formatted number:10.00-
 Note use of leading blank

请注意，前导空格仅出现在没有符号格式的正数之前。无论符号的位置如何，负数或任何带符号的数字之前都不会出现前导空格。

以下嵌入式 SQL 示例显示了使用“FM”格式覆盖无符号正数的默认前导空白：

```

/// d ##class(PHA.TEST.SQLFunction).ToChar6()
ClassMethod ToChar6()
{
    &sql(
        SELECT
            TO_CHAR(12345678.90, '99,999,999.99'),
            TO_CHAR(12345678.90, 'FM99,999,999.99')
        INTO
            :num, :fmnum
    )
    if SQLCODE '= 0 {
        w !, "Error code ", SQLCODE
    } else {
        w !, "Formatted number:", num
        w !, "Formatted number:", fmnum
        w !, "Note leading blank"
    }
}

```

```
DHC-APP>d ##class(PHA.TEST.SQLFunction).ToChar6()
```

Formatted number: 12,345,678.90
 Formatted number:12,345,678.90
 Note leading blank

以下嵌入式 SQL 示例显示了前导美元符号的使用：

```

/// d ##class(PHA.TEST.SQLFunction).ToChar7()
ClassMethod ToChar7()
{
    &sql(
        SELECT
            TO_CHAR(1234567890, '$9G999G999G999'),
            TO_CHAR(1234567890, 'S$9G999G999G999'),
            TO_CHAR(12345678.90, '$99G999G999D99')
        INTO
            :d, :sd, :dD
    )
    if SQLCODE '= 0 {
        w !, "Error code ", SQLCODE
    } else {
        w !, "Formatted number:", d
        w !, "Formatted number:", sd
        w !, "Formatted number:", dD
        w !, "Note leading blanks"
    }
}

```

```

    }
}

```

```
DHC-APP> d ##class(PHA.TEST.SQLFunction).ToChar7()
```

```

Formatted number: $1,234,567,890
Formatted number:+$1,234,567,890
Formatted number: $12,345,678.90
Note leading blanks

```

美元符号前面总是有一个符号或一个空白字符。

以下嵌入式 SQL 示例显示了当 format 参数包含的整数位数少于输入数值时会发生什么：

```

/// d ##class(PHA.TEST.SQLFunction).ToChar8()
ClassMethod ToChar8()
{
    &sql(
        SELECT
            TO_CHAR(1234567.89, '9'),
            TO_CHAR(1234567.89, '99'),
            TO_CHAR(1234567.89, '99D99')
        INTO :a,:b,:c
    )
    if SQLCODE '= 0 {
        w !,"Error code ",SQLCODE
    } else {
        w !,"Formatted number:",a
        w !,"Formatted number:",b
        w !,"Formatted number:",c
    }
}

```

每个 TOCHAR 分别返回一串井号：“##”、“###”和“#####”。

```
DHC-APP> d ##class(PHA.TEST.SQLFunction).ToChar8()
```

```

Formatted number:##
Formatted number:###
Formatted number:#####

```

以下嵌入式 SQL 示例显示了当 format 参数包含的十进制（小数）位数少于输入数值表达式时会发生什么：

```

/// d ##class(PHA.TEST.SQLFunction).ToChar9()
ClassMethod ToChar9()
{
    &sql(
        SELECT
            TO_CHAR(1234567.4999, '9999999.9'),
            TO_CHAR(1234567.91, '9999999')
        INTO :a,:b
    )
}

```

返回的数字分别四舍五入为 “1234567.5” 和 “1234568”。

Formatted number: 1234568

源

<https://cn.community.intersystems.com/post/%E7%AC%AC142%E7%AB%A0-sql%E5%87%BD%E6%95%B0-toch-ar%E7%BC%88%E4%B8%89%E7%BC%89>