文章
姚 鑫 · 一月 22阅读大约需 分钟

第三十一章 SQL函数 CONVERT

第三十一章 SQL函数 CONVERT

将给定表达式转换为指定数据类型的函数。

```
CONVERT(datatype,expression[,format-code])

{fn CONVERT(expression,datatype)}
```

参数

- expression - 要转换的表达式。
- datatype - 要将表达式转换为的数据类型。
- format - 可选-
  指定日期和时间格式的整数代码，用于在日期/时间/时间戳数据类型和字符数据类型之间进行转换。
  此参数仅用于通用标量语法形式。

描述

这里描述了CONVERT函数的两种不同实现。
两者都将一种数据类型中的表达式转换为另一种数据类型中的相应值。
两者都执行日期和时间转换。

注 意:这两个CONVERT实现中的参数以不同的顺序表示。
第一个是与MS SQL Server兼容的通用 IRIS标量函数，它接受三个参数。
第二个是带有两个参数的 ODBC标量函数。
下面的文本将分别处理这两种形式的CONVERT。

- CONVERT(datatype,expression)支持流数据的转换。
  例如，可以将字符流字段的内容转换为数据类型为VARCHAR的字符串。
- {fn CONVERT(expression,datatype)}不支持流数据的转换;
  指定要表达的流字段将导致SQLCODE -37错误。

为两个版本的CONVERT指定一个无效值将导致SQLCODE -141。

如果表达式没有定义的数据类型(例如ObjectScript提供的主机变量)，则其数据类型默认为字符串数据类型。

CONVERT(datatype,expression,format-code)

可以通过执行VARCHAR-to-VARCHAR转换来截断字符串，指定输出字符串长度小于表达式字符串长度。

在使用CONVERT(或CAST)时，如果字符数据类型(如CHAR或VARCHAR)没有指定长度，则默认的最大长度为30个

字符。
如果二进制数据类型(如binary或VARBINARY)没有指定长度，则默认的最大长度为30个字符。
否则，这些没有指定长度的数据类型将被映射到一个1个字符的MAXLEN，如data types表所示。

可以执行BIT数据类型转换。
允许的值为1、0或NULL。
如果指定任的其他值，IRIS将发出SQLCODE -141错误。
在下面的嵌入式SQL示例中，两者都是一个NULL的BIT转换:

```
ClassMethod Convert()
{
    s a=""
    &sql(
        SELECT CONVERT(BIT,:a),
            CONVERT(BIT,NULL)
        INTO :x,:y)
    w !,"SQLCODE=",SQLCODE
    w !,"the host variable is:",x
    w !,"the NULL keyword is:",y
}
```

可选的format-code参数指定日期、datetime或时间格式。
该格式既可用于定义从日期/时间/时间戳数据类型转换为字符串时的输出，也可用于定义从字符串转换为日期/时间/
时间戳数据类型时的输入。
支持以下格式代码;
输出两位数年份的格式代码列在第一列;
输出四位数年或不输出年的格式列在第二列:
Two-digit year codes

1

2

3

4

5

6

Two-digit year codes

7

10

11

12

以是日期和时间转换的特性
- 取值范围:允许的日期范围为0001-01-01 ~ 9999-12-31。
- 默认值:
  - 将时间值转换为TIMESTAMP、POSIXTIME、DATETIME或SMALLDATETIME时，日期默认为1900-01-01。
 注 意，对于{fn CONVERT()}，日期默认为1841-01-01。
  - 将日期值转换为TIMESTAMP、POSIXTIME、DATETIME或SMALLDATETIME时，时间默认为00:00:00。
- Default Format:如果没有指定Format -code，CONVERT将尝试从指定的值确定格式。
如果不能，则默认为格式代码100。
- 两位数年份:从00到49的两位数年份转换为21世纪的日期(2000到2049);
从50到99的两位数年份转换为20世纪的日期(1950到1999)。
- 分数秒:分数秒前可以加句号(.)或冒号(:)。
这些符号有不同的含义:
  - 句点是默认值，可用于所有格式代码。
句号表示标准分数;
因此，12:00:00.4表示十分之四秒，而12:00:00.004表示千分之四秒。
分数精度的位数没有限制。
  - 冒号只能用于以格式代码值:9/109、13/113、14/114、130和131。
冒号表示后面的数字是千分之一秒;
因此12:00:00:4表示四万分之一秒(12:00:00.004)。
冒号后面的数字限制为3位。

当指定表达式的格式无效或格式与格式代码不匹配时，将产生SQLCODE -141错误。
指定一个不存在的格式代码将返回1900-01-01 00:00:00。

{fn CONVERT(expression,datatype)}

这是ODBC标量函数。
它支持以下ODBC显式数据类型转换。
必须使用“SQL_”关键字指定这种形式的CONVERT的数据类型转换。
在下表中，有两组转换数据类型，第一组转换数据值和数据类型，第二组转换数据类型，但不转换数据值:

Source
Any numeric data type
%String
%Date
%Time
%PosixTime
%TimeStamp

Source	
Any non-stream data type	
Any non-stream data type	

SQL_VARCHAR是标准的ODBC表示。
在转换为SQL_VARCHAR时，日期和时间被转换为相应的ODBC表示;
数字数据类型值转换为字符串表示。
从SQL_VARCHAR转换时，该值必须是有效的ODBC Time、Timestamp或Date表示。

- 当将时间值转换为SQL_TIMESTAMP或SQL_POSIXTIME时，未指定的日期默认为1841-01-01。
  注 意，对于CONVERT()，日期默认为1900-01-01。
- 将date值转换为SQL_TIMESTAMP或SQL_POSIXTIME时，时间默认为00:00:00。

在这种语法形式中，小数秒前面可以加句号(.)或冒号(:)。
这些符号有不同的含义。
句号表示标准分数;
因此，12:00:00.4表示十分之四秒，而12:00:00.004表示千分之四秒。
冒号表示接下来的是千分之一秒;
因此12:00:00:4表示千分之四秒。
冒号后面的数字限制为3位。

在转换为整数数据类型或SQL_DOUBLE数据类型时，数据值(包括日期和时间)将转换为数字表示。
对于SQL_DATE，这是自1841年1月1日以来的天数。
对于SQL_TIME，这是自午夜以来的秒数。
当遇到非数字字符时，输入字符串将被截断。
整数数据类型还截断十进制数字，返回数字的整数部分。

{fn CONVERT(expression,datatype)}不支持流数据的转换;
指定要表达的流字段将导致SQLCODE -37错误。

转换成的数据类型的NULL仍然是NULL。

空字符串(")，或任何非数字字符串值转换如下:
- SQL_VARCHAR和SQL_TIMESTAMP返回提供的值。
- 数字数据类型转换为0(零)。
- SQL_DATE和SQL_TIME转换为NULL。

CONVERT 类方法

还可以使用CONVERT()方法调用执行数据类型转换，使用" SQL_"关键字指定数据类型:

```
$SYSTEM.SQL.Functions.CONVERT(expression,convert-to-type,convert-from-type)
```

如下示例所示:

```
 WRITE $SYSTEM.SQL.CONVERT(60945,"SQL_VARCHAR","SQL_DATE")
2007-11-11
```

示例

CONVERT() 示例

下面的示例使用标量语法形式的CONVERT。

下面的示例比较了使用DECIMAL和DOUBLE数据类型对小数的转换:

```
SELECT CONVERT(DECIMAL,-123456789.0000123456789) AS DecimalVal,
       CONVERT(DOUBLE,-123456789.0000123456789) AS DoubleVal
```

下面的示例将字符流字段转换为VARCHAR文本字符串。
它还使用CHAR_LENGTH显示字符流字段的长度:

```
SELECT Notes,CONVERT(VARCHAR(80),Notes) AS NoteText,CHAR_LENGTH(Notes) AS TextLen
FROM Sample.Employee WHERE Notes IS NOT NULL
```

下面的例子展示了几种将出生日期字段(DOB)转换为格式化字符串的方法:

```
SELECT DOB,
       CONVERT(VARCHAR(20),DOB) AS DOBDefault,
       CONVERT(VARCHAR(20),DOB,100) AS DOB100,
       CONVERT(VARCHAR(20),DOB,107) AS DOB107,
       CONVERT(VARCHAR(20),DOB,114) AS DOB114,
       CONVERT(VARCHAR(20),DOB,126) AS DOB126
FROM Sample.Person
```

默认格式和代码100格式是相同的。
因为DOB字段不包含时间值，所以显示时间的格式(这里包括默认值100、114和126)提供一个零值，它表示12:00AM (午夜)。
代码126格式提供了一个不包含空格的日期和时间字符串。

{fn CONVERT()} 示例

下面的示例使用了ODBC语法形式的CONVERT。

下面的嵌入式SQL示例将混合字符串转换为整数。
IRIS在第一个非数字字符处截断字符串，然后将结果数字转换为规范形式:

```
ClassMethod Convert1()
{
    s a="007 James Bond"
    &sql(SELECT {fn CONVERT(:a, SQL_INTEGER)} INTO :x)
    w !,"SQLCODE=",SQLCODE
```

```
    w !,"the host variable is:",x
}



DHC-APP>d ##class(PHA.TEST.SQLCommand).Convert1()

SQLCODE=0
the host variable is:7
```

下面的示例将“DOB”(出生日期)列中的日期转换为SQL_TIMESTAMP数据类型。

```
SELECT DOB,{fn CONVERT(DOB,SQL_TIMESTAMP)} AS DOBtoTstamp
     FROM Sample.Person
```

生成时间戳格式为“yyyy-mm-dd hh:mm:ss”。

下面的示例将“DOB”(出生日期)列中的日期转换为SQL_INTEGER数据类型。

```
SELECT DOB,{fn CONVERT(DOB,SQL_INTEGER)} AS DOBtoInt
     FROM Sample.Person
```

下面的示例将“DOB”(出生日期)列中的日期转换为SQL_VARCHAR数据类型。

```
SELECT DOB,{fn CONVERT(DOB,SQL_VARCHAR)} AS DOBtoVChar
     FROM Sample.Person
```

生成字符串格式为:yyyy-mm-dd。

#SQL #Caché

源 URL: https://cn.community.intersystems.com/post/%E7%AC%AC%E4%B8%89%E5%8D%81%E4%B8%80%E7%AB%A0-sql%E5%87%BD%E6%95%B0-convert