文章
姚 鑫 · 四月 26, 2022 阅读大约需 4 分钟

第124章 SQL函数 SECOND

第124章 SQL函数 SECOND

返回日期时间表达式的秒数的时间函数。

大纲

```
{fn SECOND(time-expression)}
```

参数

- time-expression -
  作为列名、另一个标量函数的结果或字符串或数字文字的表达式。它必须解析为时间戳字符串或 $HOROLOG 字符串，其中基础数据类型可以表示为 %Time、%TimeStamp 或 %PosixTime。

描述

SECOND 返回一个从 0 到 59 的整数，也可能返回小数秒。秒数是针对 $HOROLOG 或 $ZTIMESTAMP 值、ODBC 格式日期字符串（没有时间值）或时间戳计算的。

时间表达式时间戳可以是数据类型 %Library.PosixTime（编码的 64 位有符号整数），也可以是数据类型 %Library.TimeStamp (yyyy-mm-dd hh:mm:ss.fff)。

要更改默认时间格式，请使用 SET OPTION 命令。

必须提供时间戳字符串 (yyyy-mm-dd hh:mm:ss) 或 $HOROLOG 字符串。 $HOROLOG 字符串可以是完整的日期时间字符串 (63274,37279) 或只是 $HOROLOG (37279) 的时间整数部分。不能提供时间字符串 (hh:mm:ss)；无论实际秒数如何，这始终返回 0。

日期时间字符串的时间部分必须是有效时间。否则，将生成 SQLCODE -400错误 <ILLEGAL VALUE>。秒 (ss) 部分必须是 0 到 59 范围内的整数。前导零在输入时是可选的；前导零在输出上被抑制。

日期时间字符串的日期部分未经过验证。

当秒部分为 “ 0 ” 或 “ 00 ” 时，SECOND 返回 0 秒。如果提供了没有时间表达式的 ODBC 日期，或者完全省略了时间表达式的秒部分('hh', 'hh:mm', 'hh:mm:', or 'hh::')。

可以使用 DATEPART 或 DATENAME 返回相同的时间信息。

也可以使用 SECOND() 方法调用从 ObjectScript 调用此函数：

```
$SYSTEM.SQL.Functions.SECOND(time-expression)
```

小数秒

如果在 time-expression 中提供
SECOND，则返回秒的小数部分。尾随零被截断。如果未指定小数秒（例如：38.00），则小数分隔符也会被截断。

时间值的标准内部表示 ($HOROLOG) 不支持小数秒。时间戳确实支持小数秒。

以下 SQL 函数支持小数秒：SECOND、CURRENTTIMESTAMP、DATENAME、DATEPART 和 GETDATE。
CURTIME、CURRENTTIME 和 NOW 不支持小数秒。

SQL SET OPTION 语句允许设置小数秒的默认精度（小数位数）。

ObjectScript $ZTIMESTAMP 特殊变量可用于表示小数秒。 ObjectScript 函数
$ZDATETIME、$ZDATETIMEH、$ZTIME 和 $ZTIMEH 支持小数秒。

示例

以下示例都返回数字 38，因为它是时间表达式的 38 秒：

```
SELECT {fn SECOND('2018-02-16 18:45:38')} AS ODBCSeconds
```

```
38
```

```
SELECT {fn SECOND(67538)} AS HorologSeconds
```

```
38
```

以下示例返回 0.9 秒。前导零和尾随零被截断：

```
SELECT {fn SECOND('2018-02-16 18:45:00.9000')} AS Seconds_Given
```

```
0
```

以下示例返回 0 秒，因为省略了日期时间字符串的秒部分：

```
SELECT {fn SECOND('2018-02-16 18:45')} AS Seconds_Given
```

```
0
```

以下示例返回 0 秒，因为日期时间字符串中省略了时间表达式：

```
SELECT {fn SECOND('2018-02-16')} AS Seconds_Given
```

```
0
```

以下示例均返回当前时间的秒部分，以整秒为单位：

```
SELECT {fn SECOND(CURRENT_TIME)} AS Sec_CurrentT,
       {fn SECOND({fn CURTIME()})} AS Sec_CurT,
       {fn SECOND({fn NOW()})} AS Sec_Now,
       {fn SECOND($HOROLOG)} AS Sec_Horolog,
       {fn SECOND($ZTIMESTAMP)} AS Sec_ZTS

40  40  40  40  40
```

以下示例显示前导零被抑制。第一个 SECOND 函数返回长度为 2，其他函数返回长度为 1。省略的时间被认为是 0 秒，其长度为 1：

```
SELECT LENGTH({fn SECOND('2018-02-15 11:45:22')}),
       LENGTH({fn SECOND('2018-02-15 03:05:06')}),
       LENGTH({fn SECOND('2018-02-15 3:5:6')}),
       LENGTH({fn SECOND('2018-02-15')})

2   1   1   1
```

以下嵌入式 SQL 示例显示 SECOND 函数识别为区域设置指定的 TimeSeparator 字符：

```
/// d ##class(PHA.TEST.SQLFunction).Second()
ClassMethod Second()
{
    d ##class(%SYS.NLS.Format).SetFormatItem("TimeSeparator",".")
    &sql(
        SELECT {fn SECOND('2018-02-16 18.45.38')} INTO :a
    )
    w "seconds=",a
}
```

```
DHC-APP>d ##class(PHA.TEST.SQLFunction).Second()
seconds=38
```

#SQL #Caché

源URL: https://cn.community.intersystems.com/post/%E7%AC%AC124%E7%AB%A0-sql%E5%87%BD%E6%95%B0-second