文章
姚 鑫 · 三月 21, 2021 阅读大约需 9 分钟

第十二章 使用嵌入式SQL（五）

第十二章 使用嵌入式SQL（五）

嵌入式SQL变量

以下局部变量在嵌入式SQL中具有特殊用途。这些局部变量名称区分大小写。在过程启动时，这些变量是不确定的。它们由嵌入式SQL操作设置。也可以使用SET命令直接设置它们，或使用NEW命令将其重置为未定义。像任何局部变量一样，值将在过程持续期间或直到设置为另一个值或使用NEW进行定义之前一直存在。例如，某些成功的嵌入式SQL操作未设置%ROWID。执行这些操作后，%ROWID是未定义的或保持设置为其先前值。

- %msg
- %ROWCOUNT
- %ROWID
- SQLCODE

这些局部变量不是由Dynamic SQL设置的。 （请注意，SQL Shell和Management Portal SQL接口执行Dynamic SQL。）相反，Dynamic SQL设置相应的对象属性。

在嵌入式SQL中使用以下ObjectScript特殊变量。这些特殊的变量名称不区分大小写。在过程启动时，这些变量将初始化为一个值。它们由嵌入式SQL操作设置。不能使用SET或NEW命令直接设置它们。
- $TLEVEL
- $USERNAME

作为已定义的InterSystems IRIS嵌入式SQL接口的一部分，InterSystems IRIS可以在嵌入式SQL处理期间设置任何这些变量。

如果嵌入式SQL在类方法中（procedureBlock = ON），则系统会自动将所有这些变量放在PublicList中，并自动将SQLCODE，%ROWID，%ROWCOUNT，%msg以及SQL语句。可以通过引用方法来传递这些变量；通过引用传递的变量将不会在类方法过程块中自动更新。

如果嵌入式SQL在例程中，则程序员有责任在调用嵌入式SQL之前新建%msg，%ROWCOUNT，%ROWID和SQLCODE变量。更新这些变量可防止干扰这些变量的先前设置。为避免<FRAMESTACK>错误，不应在迭代周期内执行此NEW操作。

%msg

包含系统提供的错误消息字符串的变量。如果InterSystems SQL将SQLCODE设置为负整数（表示错误），则仅设置%msg。如果SQLCODE设置为0或100，则%msg变量与其先前值保持不变。

此行为不同于相应的Dynamic SQL %Message属性，当没有当前错误时，该属性将设置为空字符串。

在某些情况下，特定的SQLCODE错误代码可能与一个以上的%msg字符串相关联，描述了生成SQLCODE的不同条件。 %msg还可以接受用户定义的消息字符串。当触发器代码显式设置%ok = 0来中止触发器时，这最常用于从触发器发出用户定义的消息。

当执行SQL代码时，将使用有效的NLS语言生成错误消息字符串。可以在不同的NLS语言环境中编译SQL代码。该消息将根据运行时NLS环境生成。请参见$ SYS.NLS.Locale.Language。

%ROWCOUNT

一个整数计数器，指示受特定语句影响的行数。

- INSERT，UPDATE，INSERT OR UPDATE和DELETE将%ROWCOUNT设置为受影响的行数。带有显式值的INSERT命令只能影响一行，因此将%ROWCOUNT设置为0或1。INSERT查询结果，UPDATE或DELETE可以影响多行，因此可以将%ROWCOUNT设置为0或正数。整数。
- 无论删除多少行还是删除任何行，TRUNCATE TABLE始终将%ROWCOUNT设置为-1。因此，要确定实际删除的行数，请在TRUNCATE TABLE之前对表执行COUNT（*），或者使用DELETE而不是TRUNCATE TABLE删除表中的所有行。
- 没有声明游标的SELECT只能作用于一行，因此执行简单的SELECT总是会将%ROWCOUNT设置为1（与检索到的选择标准匹配的单行）或0（没有与选择标准匹配的行）。
- DECLARE游标名CURSOR FOR SELECT不会初始化%ROWCOUNT； SELECT之后，%ROWCOUNT不变，而OPEN游标名之后，%ROWCOUNT不变。第一个成功的FETCH设置%ROWCOUNT。如果没有行符合查询选择条件，则FETCH设置%ROWCOUNT = 0；否则，设置%ROWCOUNT = 0。如果FETCH检索与查询选择条件匹配的行，则它将设置%ROWCOUNT = 1。随后的每个获取行的FETCH都将递增%ROWCOUNT。 CLOSE时或FETCH发出SQLCODE 100（无数据或无更多数据）时，%ROWCOUNT包含已检索的总行数。

此SELECT行为与相应的Dynamic SQL%ROWCOUNT属性不同，该属性在查询执行完成时设置为0，并且仅在程序迭代查询返回的结果集时才递增。

如果SELECT查询仅返回聚合函数，则每个FETCH都将设置%ROWCOUNT = 1。即使表中没有数据，第一个FETCH始终以SQLCODE = 0来完成；任何后续的FETCH均以SQLCODE = 100完成，并设置%ROWCOUNT = 1。

以下嵌入式SQL示例声明一个游标，并使用FETCH来获取表中的每一行。到达数据结尾（SQLCODE = 100）时，%ROWCOUNT包含已检索的行数：

```
/// d ##class(PHA.TEST.SQL).ROWCOUNT()
ClassMethod ROWCOUNT()
{
    SET name="LastName,FirstName",state="##"
    &sql(DECLARE EmpCursor CURSOR FOR
        SELECT Name, Home_State
        INTO :name,:state FROM Sample.Person
        WHERE Home_State %STARTSWITH 'M')
    WRITE !,"BEFORE: Name=",name," State=",state
    &sql(OPEN EmpCursor)
    QUIT:(SQLCODE'=0)
    FOR {
        &sql(FETCH EmpCursor)
        QUIT:SQLCODE
        WRITE !,"Row fetch count: ",%ROWCOUNT
        WRITE " Name=",name," State=",state
    }
    WRITE !,"????SQLCODE: ",SQLCODE
    &sql(CLOSE EmpCursor)
    WRITE !,"AFTER: Name=",name," State=",state
    WRITE !,"??????: ",%ROWCOUNT
}


DHC-APP>d ##class(PHA.TEST.SQL).ROWCOUNT()
```

```
BEFORE: Name=LastName,FirstName State=##
Row fetch count: 1 Name=O'Rielly,Chris H. State=MS
Row fetch count: 2 Name=Orwell,John V. State=MT
Row fetch count: 3 Name=Zevon,Heloisa O. State=MI
...
Row fetch count: 37 Name=Joyce,Elmo R. State=MO
Row fetch count: 38 Name=Jafari,Christine Z. State=MI
????SQLCODE: 100
AFTER: Name=Jafari,Christine Z. State=OH
??????: 38
```

以下嵌入式SQL示例执行UPDATE并设置受更改影响的行数：

```
/// d ##class(PHA.TEST.SQL).ROWCOUNT1()
ClassMethod ROWCOUNT1()
{
    &sql(UPDATE Sample.Employee
        SET Salary = (Salary * 1.1)
        WHERE Salary < 50000)
    IF SQLCODE<0 {
        WRITE "SQLCODE error ",SQLCODE," ",%msg  QUIT
        }
    WRITE "Employees: ", %ROWCOUNT,!
}
```

```
DHC-APP>d ##class(PHA.TEST.SQL).ROWCOUNT1()
Employees: 48
```

请记住，所有嵌入式SQL语句（在给定进程内）都会修改%ROWCOUNT变量。如需要%ROWCOUNT提供的值，请确保在执行其他Embedded SQL语句之前获取其值。根据嵌入式SQL的调用方式，可能必须在输入嵌入式SQL之前新建%ROWCOUNT变量。

另请注意，显式回滚事务不会影响%ROWCOUNT的值。例如，以下内容将报告已进行了更改，即使它们已经滚动了。

```
/// d ##class(PHA.TEST.SQL).ROWCOUNT2()
ClassMethod ROWCOUNT2()
{
    TSTART // ????
    NEW SQLCODE,%ROWCOUNT,%ROWID
    &sql(UPDATE Sample.Employee
        SET Salary = (Salary * 1.1)
        WHERE Salary < 50000)
    IF SQLCODE<0 {
        WRITE "SQLCODE error ",SQLCODE," ",%msg  QUIT
    }
    TROLLBACK // ????;?????rowcount
    Write "Employees: ", %ROWCOUNT,!
}
```

```
DHC-APP>d ##class(PHA.TEST.SQL).ROWCOUNT2()
Employees: 37
```

隐式事务（例如，如果UPDATE未通过约束检查）由%ROWCOUNT反映。

%ROWID

初始化进程时，未定义%ROWID。当发出NEW %ROWID命令时，%ROWID将重置为未定义。 %ROWID由下面描述的嵌入式SQL操作设置。如果该操作不成功或成功完成，但未获取或修改任何行，则%ROWID值与其先前值保持不变：未定义，或由先前的嵌入式SQL操作设置为某个值。因此，在每个嵌入式SQL操作之前，请务必新建%ROWID。

%ROWID设置为受以下操作影响的最后一行的RowID：
- INSERT，UPDATE，INSERT OR UPDATE或DELETE：单行操作后，%ROWID变量包含系统分配的RowID（对象ID）值，该值分配给插入，更新或删除的记录。经过多行操作之后，%ROWID变量包含系统分配的最后一条插入，更新或删除的记录的RowID（对象ID）的值。如果未插入，更新或删除任何记录，则%ROWID变量值将保持不变。 TRUNCATE TABLE没有设置%ROWID。
- 基于游标的SELECT：DECLARE游标名称CURSOR和OPEN游标名称语句未初始化%ROWID； %ROWID值与其先前值保持不变。第一个成功的FETCH设置%ROWID。随后的每个获取行的FETCH都会将%ROWID重置为当前RowID值。如果FETCH检索一行可更新游标，则会设置%ROWID。可更新游标是其中顶部FROM子句仅包含一个元素（单个表名或可更新视图名）的游标。如果游标不可更新，则%ROWID保持不变。如果没有行符合查询选择条件，则FETCH不会更改先前的%ROWID值（如果有）。 CLOSE时或FETCH发出SQLCODE 100（无数据或无更多数据）时，%ROWID包含检索到的最后一行的RowID。

具有DISTINCT关键字或GROUP BY子句的基于游标的SELECT不会设置%ROWID。
%ROWID值与其先前的值（如果有）保持不变。

如果基于游标的SELECT仅返回聚合函数值，则不会设置%ROWID。如果它同时返回字段值和聚合函数值，则将每个FETCH的%ROWID值设置为查询返回的最后一行的RowID。

- 没有声明游标的SELECT不会设置%ROWID。完成简单的SELECT语句后，%ROWID值将保持不变。

在Dynamic SQL中，相应的%ROWID属性返回插入，更新或删除的最后一条记录的RowID值。执行SELECT查询时，Dynamic SQL不会返回%ROWID属性值。

可以使用以下方法调用从ObjectScript中检索当前的%ROWID：

```
DHC-APP>  WRITE $SYSTEM.SQL.GetROWID()
213
```

在执行INSERT，UPDATE，DELETE，TRUNCATE TABLE或基于游标的SELECT操作之后，LASTIDENTITY SQL函数将为最近修改的记录返回IDENTITY字段的值。如果表没有IDENTITY字段，则此函数返回最近修改记录的RowID。

SQLCODE

运行嵌入式SQL查询后，必须在处理输出主机变量之前检查SQLCODE。

如果SQLCODE = 0，则查询成功完成并返回数据。输出主机变量包含字段值。
如果SQLCODE = 100，则查询成功完成，但是输出主机变量值可能不同。任何一个：
- 查询返回一个或多个数据行（SQLCODE = 0），然后到达数据的末尾（SQLCODE = 100），在这种情况下，输出主机变量设置为返回的最后一行的字段值。 %ROWCOUNT> 0。
- 查询未返回任何数据，在这种情况下，输出主机变量未定义。 %ROWCOUNT = 0。

如果查询仅返回聚合函数，则即使表中没有数据，第一个FETCH也会始终以SQLCODE = 0和%ROWCOUNT = 1来完成。第二个FETCH以SQLCODE = 100和%ROWCOUNT = 1结束。如果表中没有数据或没有数据与查询条件匹配，查询将根据需要将输出主机变量设置为0或空字符串。

如果SQLCODE为负数，则查询失败，并显示错误条件。

根据嵌入式SQL的调用方式，可能必须在输入嵌入式SQL之前新建SQLCODE变量。在触发代码中，将SQLCODE设置为非零值会自动将%ok = 0设置为中止并回滚触发操作。

在动态SQL中，相应的%SQLCODE属性返回SQL错误代码值。

$TLEVEL

事务级计数器。
InterSystems SQL将$TLEVEL初始化为0。
如果没有当前事务，$TLEVEL为0。
- 初始START TRANSACTION将$LEVEL设置为1。其他START TRANSACTION语句对$TLEVEL无效。
- 每个SAVEPOINT语句将$TLEVEL加1。
- ROLLBACK TO SAVEPOINT点名语句减少$TLEVEL。递减量取决于指定的保存点。
- COMMIT将$LEVEL重置为0。
- ROLLBACK将$LEVEL重置为0。

还可以使用%INTRANSACTION语句来确定事务是否在进行中。

$TLEVEL也由ObjectScript事务命令设置。

$USERNAME

SQL用户名与InterSystems IRIS用户名相同，存储在ObjectScript $USERNAME特殊变量中。用户名可以用作系统范围的默认架构，也可以用作架构搜索路径中的元素。

#SQL #Caché #InterSystems IRIS #InterSystems IRIS for Health

源URL:
https://cn.community.intersystems.com/post/%E7%AC%AC%E5%8D%81%E4%BA%8C%E7%AB%A0-%E4%BD%BF%E7%94%A8%E5%B5%8C%E5%85%A5%E5%BC%8Fsql%EF%BC%88%E4%BA%94%EF%BC%89