

## 文章

[Jingwei Wang](#) · 七月 28, 2022 阅读大约需 4 分钟

## InterSystems SQL 的使用 - 第七部分 - Stored Procedures

### 定义 Stored Procedures

可以使用以下方式定义stored procedures

0. 使用DDL定义存储过程
1. 使用类方法定义存储过程

#### 使用DDL定义存储过程

CREATE PROCEDURE

可以创建一个查询，它总是作为一个存储过程被预测。一个查询可以返回一个单一的结果集。

```
CREATE PROCEDURE AgeQuerySP(IN topnum INT 10, IN minage INT 20)
BEGIN
SELECT TOP ?topnum Name, Age FROM Sample.Person
WHERE Age > :minage;
END
```

列表中的每个参数声明包括（按顺序）。指定参数模式是IN（输入值），OUT（输出值），还是INOUT（修改值）。如果省略，默认的参数模式是IN，参数名称是区分大小写的。

CREATE QUERY 可以创建一个可以选择作为存储过程投射的查询。一个查询可以返回一个单一的结果集，这个查询可能是也可能不是作为存储过程公开的。要创建一个作为存储过程公开的查询，你必须指定PROCEDURE关键字作为其特征之一。你也可以使用CREATE PROCEDURE语句来创建一个作为存储过程公开的查询。

为了创建一个查询，你必须拥有%CREATEQUERY管理权限，正如GRANT命令所指定的那样。如果类的定义是一个已部署的类，你不能在该类中创建一个查询。

```
CREATE QUERY AgeQuery(IN topnum INT DEFAULT 10,IN minage INT 20)
PROCEDURE
BEGIN
SELECT TOP :topnum Name, Age FROM Sample.Person
WHERE Age > :minage ;
END
```

#### 使用类定义存储过程

要定义一个方法存储过程，只需定义一个类方法并设置其SqlProc关键字。

```
Class MyApp.Person Extends %Persistent [DdlAllowed]
{
?
```

```

/// This procedure finds total sales for a territory
ClassMethod FindTotal(territory As %String) As %Integer [SqlProc]
{
    // use embedded sql to find total sales
    &sql(SELECT SUM(SalesAmount) INTO :total
        FROM Sales
        WHERE Territory = :territory
    )
    ?
    Quit total
}
}

```

这个类被编译后，FindTotal()方法将作为存储过程MyApp.PersonFindTotal()投射到SQL。你可以使用该方法的SqlName关键字改变SQL对存储过程的命名。

该方法使用一个存储过程上下文处理程序，在存储过程和其调用者（例如，ODBC服务器）之间来回传递存储过程上下文。这个过程上下文处理程序是由InterSystems

IRIS使用%sqlcontext对象自动生成的（作为%qHandle:%SQLProcContext）。

%sqlcontext由SQLCODE错误状态、SQL行数、错误信息等属性组成，使用相应的SQL变量进行设置，如下所示。

```

SET %sqlcontext.%SQLCODE=SQLCODE
SET %sqlcontext.%ROWCOUNT=%ROWCOUNT
SET %sqlcontext.%Message=%msg

```

## 使用 Stored Procedures

当执行一个以SQL函数为参数的存储过程时，使用CALL来调用该存储过程，如下面的例子。

```
CALL sp.MyProc(CURRENT_DATE)
```

SELECT查询不支持执行带有SQL函数参数的存储过程。SELECT支持用一个SQL函数参数执行一个存储函数。

xDBC不支持使用SELECT或CALL来执行一个带有SQL函数参数的存储过程。

你可以在一个SQL查询中使用一个存储函数，就像它是一个内置的SQL函数一样。函数的名称是存储函数的SQL名称（在这里是"Square"），由定义它的schema（包）名称限定（在这里是"MyApp"）。

下面的查询使用了Square函数。

```
SELECT Cost, MyApp.Utils_Square(Cost) As SquareCost FROM Products
```

## 查询Stored Procedures 信息

INFORMATION.SCHEMA.ROUTINES持久化类显示关于当前命名空间中所有routine和程序的信息。

下面的例子返回routine名称，方法或查询名称，routine类型（PROCEDURE或FUNCTION），routine主体（SQL=带SQL的类查询，EXTERNAL=不带SQL的类查询），返回数据类型，以及当前名称空间中模式"Sample"中所有routine的routine定义。

```
SELECT ROUTINE_NAME, METHOD_OR_QUERY_NAME, ROUTINE_TYPE, ROUTINE_BODY, SQL_DATA_ACCESS, IS_
USER_DEFINED_CAST,
DATA_TYPE || ' ' || CHARACTER_MAXIMUM_LENGTH AS Returns, NUMERIC_PRECISION || ':' || NUMERIC_S
CALE AS PrecisionScale,
ROUTINE_DEFINITION
FROM INFORMATION_SCHEMA.ROUTINES WHERE ROUTINE_SCHEMA='Sample '
```

INFORMATION\_SCHEMA.PARAMETERS

持久化类显示当前命名空间中所有routine和程序的输入和输出参数的信息。

下面的例子返回了当前命名空间中模式 "Sample

"中所有routine的routine名称、参数名称、是输入参数还是输出参数，以及参数数据类型信息。

```
SELECT SPECIFIC_NAME, PARAMETER_NAME, PARAMETER_MODE, ORDINAL_POSITION,
DATA_TYPE, CHARACTER_MAXIMUM_LENGTH AS MaxLen, NUMERIC_PRECISION || ':' || NUMERIC_SCALE AS
PrecisionScale
FROM INFORMATION_SCHEMA.PARAMETERS WHERE SPECIFIC_SCHEMA='Sample '
```

[#SQL #InterSystems IRIS for Health](#)

---

#### 源

URL: <https://cn.community.intersystems.com/post/intersystems-sql-%E7%9A%84%E4%BD%BF%E7%94%A8-%E7%AC%AC%E4%B8%83%E9%83%A8%E5%88%86-stored-procedures>