文章
姚 鑫 · 十一月 6, 2021 阅读大约需 4 分钟

第六十八章 SQL命令 SAVEPOINT

第六十八章 SQL命令 SAVEPOINT

在事务中标记一个点。

大纲

```
SAVEPOINT pointname
```

参数

- pointname - 保存点的名称，指定为标识符。

描述

SAVEPOINT语句标记事务中的一个点。建立保存点使能够执行事务回滚到保存点，撤消在此期间完成的所有工作并释放在此期间获得的所有锁。在长期运行的事务或具有内部控制结构的事务中，通常希望能够回滚事务的一部分，而不撤消在事务期间提交的所有工作。

保存点的建立会递增$TLEVEL事务级别计数器。回滚到保存点会将$TLEVEL事务级别计数器递减到紧接在保存点之前的值。可以在一个事务内建立最多255个保存点。超过这个保存点数量会导致SQLCODE-400致命错误，这是在SQL执行期间捕获的<TRANSACTION LEVEL> 异常。终端提示符将当前事务级别显示为提示符的TLn：前缀，其中n是介于1和255之间的整数，表示当前$TLEVEL计数。

每个保存点都与一个保存点名称相关联，这是一个唯一的标识符。保存点名称不区分大小写。保存点名称可以是分隔的标识符。

- 如果指定的保存点没有点名，或者指定的点名不是有效的标识符或SQL保留字，则会发出运行时SQLCODE-301错误。
- 如果指定点名称以“ SYS ”开头的保存点，则会发出运行时SQLCODE-302错误。这些保存点名称是保留的。

保存点名称不区分大小写；因此resetpt,ResetPt和“ RESETPT ”是相同的点名。此重复项是在回滚到保存点期间检测到的，而不是在保存点期间检测到的。当指定具有重复点名的SAVEPOINT语句时， IRIS会递增事务级别计数器，就像点名是唯一的一样。但是，最近的点名称会覆盖保存点名称表中所有先前重复的值。因此，当指定回滚到保存点点名时， IRIS会回滚到具有该点名称的最近建立的保存点，并相应地递减事务级别计数器。但是，如果再次指定回滚到同名的保存点点名，则会生成SQLCODE-375错误，并显示%msg：Cannot Rollback to Unestabled SavePoint‘ name ’，整个事务将回滚，$TLEVEL计数恢复为0。

使用保存点

嵌入式SQL、动态SQL、ODBC和JDBC支持SAVEPOINT语句。在JDBC中，connection.setSavepoint(Pointname)设置一个保存点，connection.roll back(Pointname)回滚到指定的保存点。

如果已建立保存点，请执行以下操作：

- 回滚到保存点点名将回滚自指定保存点以来所做的工作，删除该保存点和所有中间保存点，并将$TLEVEL事务级别计数器递减删除的保存点数量。如果pointname不存在或已经回滚，此命令将回滚整个事务，将$TLEVEL重置为0，并释放所有锁。
- 回滚回滚当前事务期间完成的所有工作，回滚自START TRANSACTION以来完成的工作。它将$TLEVEL事务级别计数器重置为零，并释放所有锁。请注意，常规回滚会忽略保存点。
- COMMIT提交在当前事务期间完成的所有工作。它将$TLEVEL事务级别计数器重置为零，并释放所有锁。请注意，提交操作会忽略保存点。

在事务内发出第二个START TRANSACTION对保存点或$TLEVEL事务级别计数器没有影响。

如果事务操作未能成功完成，则会发出SQLCODE-400错误。

示例

以下嵌入式SQL示例创建具有两个保存点的事务：

```
ClassMethod Savepoint()
{
    n SQLCODE,%ROWCOUNT,%ROWID
    &sql(
        START TRANSACTION
    )
    &sql(
        DELETE FROM Sample.Person WHERE Name = NULL
    )
    if SQLCODE = 100 {
        w !,"??????????"
    } elseif SQLCODE '= 0 {
        &sql(ROLLBACK)
    } else {
        w !,%ROWCOUNT," ???Null Name??"
    }
    &sql(
        SAVEPOINT svpt_age1
    )
    &sql(
        DELETE FROM Sample.Person WHERE Age = NULL
    )
    if SQLCODE = 100 {
        w !,"??????????"
    } elseif SQLCODE '= 0 {
        &sql(ROLLBACK TO SAVEPOINT svpt_age1)
    } else {
        w !,%ROWCOUNT," ???????"
    }
    &sql(
        SAVEPOINT svpt_age2
    )
    &sql(
        DELETE FROM Sample.Person WHERE Age > 65
    )
    if SQLCODE = 0 {
        &sql(COMMIT)
    } elseif SQLCODE = 100 {
        &sql(COMMIT)
    } else {
```

```
        &sql(
            ROLLBACK TO SAVEPOINT svpt_age2
        )
        w !,"????????"
    }
    &sql(COMMIT)
    &sql(COMMIT)
}
```

ObjectScript和SQL事务

使用TSTART和TCOMMIT的ObjectScript事务处理与使用SQL语句START transaction、SAVEPOINT和COMMIT的SQL事务处理不同，也不兼容。
ObjectScript和InterSystems SQL都提供了对嵌套事务的有限支持。
ObjectScript事务处理不与SQL锁控制变量交互;
特别需要关注的是SQL锁升级变量。
应用程序不应该尝试混合这两种事务处理类型。

如果事务涉及SQL更新语句，则事务应该由SQL START transaction语句启动，并使用SQL COMMIT语句提交。
使用TSTART/TCOMMIT嵌套的方法可以包含在事务中，只要它们不初始化事务。
方法和存储过程通常不应该使用SQL事务控制语句，除非按照设计，它们是事务的主控制器。

#SQL #Caché

源URL:
https://cn.community.intersystems.com/post/%E7%AC%AC%E5%85%AD%E5%8D%81%E5%85%AB%E7%AB%A0-sql%E5%91%BD%E4%BB%A4-savepoint