文章
姚 鑫 · 十月 23, 2021 阅读大约需 12 分钟

第五十四章 SQL命令 INSERT（三）

第五十四章 SQL命令 INSERT（三）

SQLCODE错误

默认情况下，INSERT是要么全有要么全无的事件：要么完全插入行，要么根本不插入行。 IRIS返回一个状态变量SQLCODE，指示插入是成功还是失败。要将行插入到表中，插入操作必须满足所有表、字段名和字段值要求，如下所示。

表：
- 该表必须已经存在。尝试插入到不存在的表会导致SQLCODE-30错误。
- 不能将该表定义为READONLY。尝试编译引用ReadOnly表的插入会导致SQLCODE-115错误。请注意，此错误是在编译时发出的，而不是在执行时发出的。
- 如果通过视图更新表，则不能将该视图定义为只读。尝试这样做会导致SQLCODE-35错误。如果视图基于分割表，则不能通过使用CHECK OPTION定义的视图进行插入。尝试这样做会导致SQLCODE-35，其中不允许基于带有CHECK选项条件的切片表的视图(sample.myview)使用%msg INSERT/UPDATE/DELETE。
- 必须具有适当的权限才能插入表

字段名称：

- 该字段必须存在。尝试插入不存在的字段会导致SQLCODE-29错误。
- 插入必须指定所有必填字段。尝试插入行而不为必填字段指定值会导致SQLCODE-108错误。
- 插入不能包含重复的字段名称。尝试插入包含两个同名字段的行会导致SQLCODE-377错误。
- 插入不能包含定义为READONLY的字段。尝试编译引用READONLY字段的插入会导致SQLCODE-138错误。请注意，此错误现在在编译时发出，而不是仅在执行时发出。使用链接表向导链接表时，可以选择将字段定义为只读。源系统上的字段可能不是只读的，但如果IRIS将链接表的字段定义为只读，则尝试引用此字段的INSERT将导致SQLCODE-138错误。

字段值：

- 每个字段值都必须通过数据类型验证。尝试插入不适合该字段数据类型的字段值会导致SQLCODE-104错误。请注意，这仅适用于插入的数据值；如果采用字段的默认值，则不必通过数据类型验证或数据大小验证。
    - 数据类型不匹配：决定是否合适的是字段的数据类型，而不是插入数据的类型。例如，除非字符串通过当前模式的日期验证，否则尝试将字符串数据类型值插入日期字段会失败；但是，尝试将日期数据类型值插入字符串字段会成功，将日期作为文字字符串插入。可以使用CONVERT函数将数据转换为目标数据类型。
    - 数据大小不匹配：数据值必须在字段的MAXLEN、MAXVAL和MINVAL范围内。例如，试图将长度超过24个字符的字符串插入到定义为VARCHAR(24)的字段中，或试图将大于127个字符的数字插入到定义为TINYINT的字段中，将导致SQLCODE-104错误。
    - 数字类型不匹配：如果通过ODBC或JDBC提供了无效的双精度数，则会出现SQLCODE-104错误。
- 每个字段值必须将显示传递到逻辑模式转换。尝试以无法转换为逻辑存储值的格式插入字段值会导致SQLCODE-146错误（对于日期）或SQLCODE-147错误（对于时间）。
- 每个字段值都必须通过数据约束验证：

    - 必须为定义为NOT NULL的字段提供数据值。如果没有默认值，则不指定数据值将导致SQLCODE-108错误，表明没有指定必填字段。
    - 字段值必须符合唯一性约束。尝试在具有唯一性约束的字段(或字段组)中插入重复字段值会导致SQLCODE-119错误。如果字段具有唯一数据约束，或者如果已将唯一字段约束应用于一组字段，则返回此错误。如果为唯一字段或主键字段指定了重复的值，或者未指定值并且第二次使用该字段的默认值将提供重复的值，则可能会发生此错误。SQLCODE-119%msg字符串包括违反唯一性约束的字段和值。例如：<Table 'Sample.MyTable', Constraint 'MYTABLEUNIQUE3', Field(s) FullName="Molly Bloom"; failed unique check> or <Table 'Sample.MyTable', Constraint 'MYTABLEPKEY2', Field(s) FullName="Molly Bloom"; failed unique check>
    - 使用VALUELIST参数定义为永久类属性的字段只能接受VALUELIST中列出的值之一作为有效值，或者不提供任何值(NULL)。VALUELIST有效值区分大小写。指定与VALUELIST值不匹配的数据值会导致SQLCODE-104字段值未通过验证错误。
- 数字以规范形式插入，但可以使用前导和尾随零以及多个前导符号指定。但是，在SQL中，两个连续的减号被解析为单行注释指示符。因此，尝试使用两个连续的前导减号指定一个数字会导致SQLCODE-12错误。
- 默认情况下，INSERT不能为系统生成值的字段指定值，例如RowID、IDKey或Identity字段。默认情况下，尝试为这些字段中的任何一个插入非空字段值都会导致SQLCODE-111错误。尝试为其中一个字段插入NULL会导致IRIS使用系统生成的值覆盖NULL；插入成功完成，并且不会发出错误代码。

如果定义了数据类型为ROWVERSION的字段，则在插入行时会自动为其分配系统生成的计数器值。尝试将值插入ROWVERSION字段会导致SQLCODE-138错误。

可以使IDENTITY字段接受用户指定的值。
通过设置SetOption(“ IdentityInsert ”)方法，您可以覆盖IDENTITY字段的默认约束，并允许将唯一整数值插入IDENTITY字段。
(可以通过调用GetOption(“ IdentityInsert ”)方法返回该约束的当前设置。)
插入IDENTITY字段值将更改IDENTITY计数器，以便后续系统生成的值从这个用户指定的值递增。
试图为IDENTITY字段插入NULL将产生SQLCODE -108错误。

IDKey数据有以下限制:
因为索引中的多个IDKey字段是用 “ ¦¦ ” (双竖条)字符分隔的，所以不能在IDKey字段数据中包含这个字符串。

插入不能包含值违反外键引用完整性的字段，除非指定了%NOCHECK关键字，或者外键是用NOCHECK关键字定义的。
否则，尝试违反外键引用完整性的插入将导致SQLCODE -121错误，并带有%msg，如下所示:<Table 'Sample.MyTable', Foreign Key Constraint 'MYTABLEFKey2', Field(s) FULLNAME failed referential integrity check>

- 字段值不能是子查询。
  试图将子查询指定为字段值将导致SQLCODE -144错误。

插入操作

Privileges

要将一行或多行数据插入到表中，您必须拥有该表的表级特权或列级特权。

表级权限

- 用户必须对指定的表具有INSERT权限。
- 如果使用SELECT查询从另一个表插入数据，用户必须对该表具有SELECT权限。

如果用户是该表的Owner(创建者)，则自动授予该用户对该表的所有特权。
否则，必须向用户授予该表的权限。
如果不这样做，将导致一个带有%msg的SQLCODE -99错误。
可以通过调用%CHECKPRIV命令来确定当前用户是否具有适当的特权。
可以使用GRANT命令分配用户表权限。

要插入到分片表，您必须对目标表具有insert权限。
如果没有这些权限会出现SQLCODE -253错误:Sharded INSERT/UPDATE/DELETE run-time error。

表级特权相当于(但不完全相同)在表的所有列上拥有列级特权。

列级权限

如果没有表级的INSERT权限，则必须对表中的至少一列具有列级的INSERT权限。
要将指定的值插入到列中，必须对该列具有列级insert权限。
只有具有INSERT权限的列才能接收INSERT命令中指定的值。

如果对指定的列没有列级INSERT权限， SQL将插入列的默认值(如果定义了)或NULL(如果没有定义默认值)。
如果对没有默认值且定义为not NULL的列没有INSERT权限， IRIS会在Prepare时间发出SQLCODE -99 (privilege Violation)错误。

如果INSERT命令指定结果集SELECT的WHERE子句中的字段，则如果这些字段不是数据插入字段，则必须具有这些字段的SELECT权限，如果这些字段包含在结果集中，则必须具有这些字段的SELECT和INSERT权限。

当属性被定义为ReadOnly时，相应的表字段也被定义为ReadOnly。
只读字段只能使用InitialExpression或SqlComputed赋值。
尝试为具有列级ReadOnly (SELECT或REFERENCES)权限的字段插入值将导致SQLCODE -138错误:无法为只读字段插入/更新值。

可以使用%CHECKPRIV来确定是否具有适当的列级特权。

快速插入

当使用JDBC在表中插入行时 IRIS默认情况下会自动执行高效的Fast Insert操作。
Fast Insert将插入的数据的规范化和格式化从服务器转移到客户机。
然后，服务器可以直接将表的整行数据设置为全局数据，而无需对服务器进行操作。
这将这些任务从服务器转移到客户机上，可以显著提高INSERT性能。
由于客户端承担了格式化数据的任务，因此在客户端环境中可能会出现不可预见的使用量增加。
如果有问题，可以使用FeatureOption属性禁用快速插入。

服务器和客户端都必须支持快速插入。
要在客户端中启用或禁用Fast Insert，请在类实例的定义中使用FeatureOption属性，如下所示:

```
Properties p = new Properties();
p.setProperty("FeatureOption","3");   / 2 is fast Insert, 1 is fast Select, 3 is both
```

如果Fast Insert是活动的，则使用缓存查询执行的Insert将使用Fast Insert执行。
生成缓存查询的初始INSERT不是使用Fast INSERT执行的。
这使能够比较初始插入与使用缓存查询执行的后续Fast Inserts的性能。
如果不支持快速插入(出于以下原因)，则执行普通插入。

快速插入必须在表上执行。
不能在可更新视图上执行。
当表具有以下任何特征时，不执行快速插入:
- 该表使用嵌入式(嵌套)存储结构(%SerialObject)。
- 该表是一个链接的表。
- 该表是子表。
- 该表有一个显式定义的多字段IDKEY索引。
- 该表有一个SERIAL (%Counter)、AUTOINCREMENT或%RowVersion字段。
- 该表有一个属性(字段)，带有定义的VALUELIST参数。

- 表有一个已定义的插入触发器。
- 该表执行字段值的LogicalToStorage转换。
- 这个表是一个Shard Master表。

如果Insert语句具有以下特征之一，则不能执行快速插入:
- 它指定一个流字段((数据类型%stream.GlobalCharacter或%Stream.GlobalBinary)、集合字段(列表或数组)或只读字段。
这些类型的字段可以存在于表中，但不能在INSERT中指定。
- 它指定一个用双括号括起来的字面值，禁止字面值替换。
例如,((A))。
- 它指定一个省略日期值的{ts}时间戳值。
- 它包括一个DEFAULT VALUES子句。

对于SQL xDBC语句审计事件，使用Fast INSERT接口的INSERT语句具有SQL fastINSERT语句的描述。
如果使用Fast Insert接口，则Audit事件不包括任何参数数据，但包括消息参数值对于fastInsert语句不可用。

参照完整性

如果没有指定%NOCHECK关键字， IRIS将使用系统范围的配置设置来确定是否执行外键引用完整性检查;
默认值是执行外键引用完整性检查。
您可以在系统范围内设置此默认值，如外键引用完整性检查中所述。
要确定当前系统范围的设置，调用$SYSTEM.SQL.CurrentSettings()。

此设置不适用于用NOCHECK关键字定义的外键。

在INSERT操作期间，对于每个外键引用，都会在引用表中相应的行上获得一个共享锁。
在执行引用完整性检查和插入该行时，此行被锁定。
然后释放锁(直到事务结束才持有锁)。
这确保了引用的行不会在引用完整性检查和插入操作完成之间发生更改。

但是，如果指定了%NOLOCK关键字，则不会对指定的表或引用表中相应的外键行执行锁操作。

子表插入

在对子表执行INSERT操作期间，父表中相应行的共享锁将被获取。
在插入子表行时，此行被锁定。
然后释放锁(直到事务结束才持有锁)。
这确保在插入操作期间不会更改引用的父行。

原子性

默认情况下，INSERT、UPDATE、DELETE和TRUNCATE TABLE是原子操作。
INSERT要么成功完成，要么回滚整个操作。
如果不能插入任何指定的行，则不插入任何指定的行，并且数据库恢复到发出INSERT之前的状态。

可以通过调用SET TRANSACTION %COMMITMODE来修改SQL中当前进程的这个默认值。
可以通过调用SetOption()方法在ObjectScript中修改当前进程的这个默认值，如下SET
status=$SYSTEM.SQL.Util.SetOption("AutoCommit"，intval，.oldval)。
以下intval整数选项是可用的:
- 1或IMPLICIT (autocommit on)——默认行为，如上所述。
每个INSERT构成一个单独的事务。
- 2或EXPLICIT (autocommit
off)——如果没有事务在进行，INSERT会自动启动一个事务，但必须显式地COMMIT或ROLLBACK来结束事务。
在EXPLICIT模式下，每个事务的数据库操作数是用户定义的。
- 0或NONE(没有自动事务)——调用INSERT时不会启动任何事务。
INSERT操作失败可能会使数据库处于不一致的状态，一些指定的行被插入，而一些未插入。

要在此模式中提供事务支持，必须使用START transaction来启动事务，并使用COMMIT或ROLLBACK来结束事务。

分片表始终没有自动事务模式，这意味着对分片表的所有插入、更新和删除都是在事务范围之外执行的。

可以使用GetOption(“ AutoCommit ”)方法确定当前进程的原子性设置，如下面的ObjectScript示例所示:

```
ClassMethod Insert1()
{
    s stat = $SYSTEM.SQL.SetOption("AutoCommit",$RANDOM(3),.oldval)
    if stat '= 1 {
        w "SetOption ??:"
        d $System.Status.DisplayError(stat)
        q
    }
    s x = $SYSTEM.SQL.GetOption("AutoCommit")
    if x = 1 {
        w "????????r",!
        w "???????"
    } elseif x = 0 {
        w "????????????:",!
        w "DELETE???????????",!
        w "?????"
    } else {
        w "??????????"
    }
}
```

事务锁

如果没有指定%NOLOCK关键字，系统将自动对INSERT、UPDATE和DELETE操作执行标准的记录锁定。
在当前事务期间，每个受影响的记录(行)都被锁定。

默认的锁阈值是每个表1000个锁。
这意味着，如果在事务期间从表中插入超过1000条记录，就会达到锁阈值
IRIS会自动将锁级别从记录锁升级为表锁。
这允许在事务期间进行大规模插入，而不会溢出锁表。

IRIS应用以下两种锁升级策略之一:

- “ E ”类型的锁升级: IRIS使用这种类型的锁升级，如果以下条件为真:
  持久性(可以从Management Portal SQL模式显示的Catalog Details中确定这一点)。
  (2)类要么不指定IDKey索引，要么指定单一属性的IDKey索引。
  “ E ”类型的锁升级在ObjectScript Reference中的lock命令中进行了描述。
- 传统SQL锁升级:类不使用 “ E ”类型锁升级的最可能的原因是存在一个多属性IDKey索引。
  在本例中，每个%Save都会增加锁计数器的值。
  这意味着如果在事务中保存单个对象1001次， IRIS将尝试升级锁。

对于这两种锁升级策略，您可以使用$SYSTEM.SQL.Util.GetOption(“ LockThreshold ”)方法确定当前系统范围的锁阈值。
默认值是1000。
这个系统范围的锁阈值是可配置的:
- 使用$SYSTEM.SQL.Util.SetOption("LockThreshold")方法。
- 使用管理门户，选择系统管理、配置、SQL和对象设置、SQL。
显示和编辑锁定升级阈值的当前设置。
默认值是1000个锁。
如果更改此设置，则更改后启动的任何新进程都将具有新设置。

需要在“ %Admin Manage Resource ”中具有“ USE ”权限才能修改锁定阈值。
IRIS会立即将对锁阈值的任何更改应用到所有当前进程。

自动锁升级的潜在后果是，当试图升级到表锁的进程与持有该表中记录锁的另一个进程冲突时，可能发生死锁情况。有几种可能的策略可以避免这种情况:(1)增加锁升级阈值，以便锁升级不太可能在事务中发生。
(2)大幅降低锁升级阈值，以便锁升级几乎立即发生，从而减少其他进程锁定同一表中的记录的机会。
(3)在事务期间应用表锁，不执行记录锁。
这可以在事务开始时指定LOCK TABLE，然后指定UNLOCK TABLE(不带IMMEDIATE关键字，以便表锁一直持续到事务结束)，然后使用%NOLOCK选项执行插入操作。

自动锁升级旨在防止锁表溢出。
但是，如果执行大量插入操作导致出现<LOCKTABLEFULL>错误，INSERT将发出SQLCODE -110错误。

行级安全性

IRIS行级安全性允许INSERT添加行，即使定义了行安全性，也不允许随后访问该行。为确保INSERT不会阻止随后对该行进行SELECT访问，建议通过具有WITH CHECK选项的视图执行INSERT。

Microsoft Access

要使用INSERT通过Microsoft Access将数据添加到
IRIS表格中，请将表格RowID字段标记为专用，或者在一个或多个附加字段上定义唯一索引。

#SQL #Caché

源URL:
https://cn.community.intersystems.com/post/%E7%AC%AC%E4%BA%94%E5%8D%81%E5%9B%9B%E7%AB%A0-sql%E5%91%BD%E4%BB%A4-insert%EF%BC%88%E4%B8%89%EF%BC%89