文章
姚 鑫 · 八月 31, 2021 阅读大约需 10 分钟

第二章 SQL命令 ALTER TABLE（二）

第二章 SQL命令 ALTER TABLE（二）

删除列限制

DROP COLUMN可以删除指定为逗号分隔列表的多个列定义。每个列出的列名后面必须紧跟其RESTORY或CASCADE(如果未指定，则默认为RESTRICE)和%DELDATA或%NODELDATE(如果未指定，则默认为%NODELDATA)选项。

默认情况下，删除列定义不会从数据映射中删除存储在该列中的任何数据。
要同时删除列定义和数据，请指定%DELDATA选项。

删除列定义并不删除相应的列级特权。
例如，授予用户在该列上插入、更新或删除数据的权限。
这将产生以下后果:
- 如果删除了一个列，然后添加了另一个同名的列，那么用户和角色将在新列上拥有与旧列相同的特权。
- 删除列后，不可能撤销该列的对象特权。

由于这些原因，通常建议在删除列定义之前使用REVOKE命令从列中撤销列级特权。

RESTRICT关键字(或无关键字):如果列出现在索引中，或者定义在外键约束或其他唯一约束中，则不能删除该列。
为该列尝试DROP COLUMN失败，并出现SQLCODE -322错误。
默认为RESTRICT。

CASCADE关键字:如果该列出现在索引中，该索引将被删除。
可能有多个索引。
如果列出现在外键中，则将删除外键。
可能有多个外键。

如果列在COMPUTECODE或COMPUTEONCHANGE子句中使用，则不能删除该列。
尝试这样做会导致SQLCODE -400错误。

添加约束限制

可以向以逗号分隔的字段列表添加约束。
例如，可以添加UNIQUE
(FName,SurName)约束，它在两个字段FName和SurName的组合值上建立一个UNIQUE约束。
类似地，可以在以逗号分隔的字段列表上添加主键约束或外键约束。

约束可以命名，也可以不命名。
如果未命名，SQL将使用表名生成约束名称。
例如，MYTABLEUnique1或MYTABLEPKEY1。

下面的示例创建了两个未命名的约束，将惟一约束和主键约束添加到以逗号分隔的字段列表中:

```
ALTER TABLE SQLUser.MyStudents
```

```
    ADD UNIQUE (FName,SurName),PRIMARY KEY (Fname,Surname)
```

- 字段必须存在才能在约束中使用。
  指定不存在的字段将产生SQLCODE -31错误。
- 不能在约束中使用RowId字段。指定RowId(ID)字段会生成SQLCODE-31错误。
- 不能在约束中使用流字段。指定流字段会生成SQLCODE-400错误：“ invalid index attribute ”
- 一个约束只能应用于一个字段一次。对一个字段指定同一约束两次会生成SQLCODE-400错误：“ index name conflict ”。

通过使用可选的CONSTRAINT标识符关键字子句，可以创建一个命名约束。命名约束必须是有效的标识符；约束名称不区分大小写。这为约束提供了一个名称供将来使用。这在以下示例中显示:

```
ALTER TABLE SQLUser.MyStudents
    ADD CONSTRAINT UnqFullName UNIQUE (FName,SurName)
```

可以将多个约束指定为逗号分隔的列表；约束名称应用于第一个约束，其他约束接收默认名称。

约束名称对于表必须是唯一的。为字段指定两次相同的约束名称会生成SQLCODE -400错误: “ index name conflict ”。

添加主键限制

主键值是必需且唯一的。因此，向现有字段或字段组合添加主键约束会使这些字段中的每个字段都成为必填字段。如果向现有字段列表添加主键约束，则这些字段的组合值必须是唯一的。如果现有字段允许空值，则不能向该字段添加主键约束。如果字段(或字段列表)包含非唯一值，则不能向该字段(或字段列表)添加主键约束。

如果向现有字段添加主键约束，则该字段也可能自动定义为IDKey索引。这取决于数据是否存在，以及通过以下方式之一建立的配置设置：

- SQL SET OPTION PKEYISIDKEY语句。
- 系统范围的$SYSTEM.SQL.Util.SetOption()方法配置选项DDLPKeyNotIDKey。要确定当前设置，调用$SYSTEM.SQL.CurrentSettings()，它显示通过DDL创建的是主键而不是ID键;默认值是1。
- 进入管理门户，选择系统管理，配置，SQL和对象设置，SQL。
  查看通过DDL创建的表的将主键定义为ID键的当前设置。
  - 如果未选中该复选框(默认设置)，则主键不会成为类定义中的IDKey索引。使用不是IDKEY的主键访问记录的效率要低得多；但是，这种类型的主键值是可以修改的。
  - 如果选中该复选框，则当通过DDL指定主键约束并且该字段不包含数据时，主键索引也将定义为IDKey索引。如果该字段包含数据，则未定义IDKey索引。如果将主键定义为IDKey索引，则数据访问效率更高，但主键值一旦设置，就永远不能修改。

如果CREATE TABLE定义了位图区索引，然后使用ALTER
TABLE添加同时也是IDKey的主键，则系统会自动删除位图区索引。

已存在时添加主键

只能定义一个主键。默认情况下，当主键已经存在时， IRIS拒绝定义主键，或者拒绝定义同一主键两次，并发出SQLCODE-307错误。即使主键的第二个定义与第一个定义相同，也会发出SQLCODE-307错误。要确定当前配置，请调用$SYSTEM.SQL.CurrentSettings()，该函数显示当键存在时允许通过DDL创建主键设置。默认值为0(否)，这是建议的配置设置。如果此选项设置为1(是)，ALTER TABLE ADD PRIMARY KEY将导致
IRIS从类定义中删除主键索引，然后使用指定的主键字段重新创建此索引。

在管理门户、系统管理、配置、SQL和对象设置中，通过选中忽略冗余DDL语句复选框，可以在系统范围内设置此选项(以及其他类似的创建、更改和删除选项)。

但是，即使将此选项设置为允许在主键已存在的情况下创建主键，如果主键索引也是IDKEY索引并且表包含数据，则不能重新创建主键索引。尝试这样做会生成SQLCODE-307错误。

添加外键限制

默认情况下，不能有两个同名的外键。这样做会生成SQLCODE-311错误。要确定当前设置，请调用$SYSTEM.SQL.CurrentSettings()，它将显示 “ 当存在外键时允许DDL添加外键约束 ” 设置。默认值为0(否)，这是此选项的推荐设置。为1(是)时，即使已存在同名外键，也可以通过DDL添加外键。

在管理门户、系统管理、配置、SQL和对象设置中，通过选中忽略冗余DDL语句复选框，可以在系统范围内设置此选项(以及其他类似的创建、更改和删除选项)。

表定义不应该有两个名称不同的外键，这两个外键引用相同的字段-公共字段并执行相互矛盾的引用操作。根据ANSI标准，如果定义了对同一字段执行相互矛盾的引用操作的两个外键(例如，ON DELETE CASCADE和ON DELETE SET NULL)，
SQL不会发出错误。相反，当DELETE或UPDATE操作遇到这些相互矛盾的外键定义时，SQL会发出错误。

指定不存在的外键字段的添加外键会生成SQLCODE-31错误。

引用不存在的父键表的添加外键会生成SQLCODE-310错误。引用现有父键表中不存在的字段的添加外键会生成SQLCODE-316错误。如果未指定父键字段，则默认为ID字段。

在发出ADD外键之前，用户必须对被引用的表或被引用的表的列具有REFERENCES特权。如果通过动态SQL或xDBC执行ALTER TABLE，则需要REFERENCES权限。

引用可以采用非唯一值的字段(或字段组合)的添加外键会生成SQLCODE-314错误，并通过%msg提供更多详细信息。

NO ACTION是分片表唯一支持的引用操作。

当表中已经存在数据时，ADD外键将受到约束。
要更改此默认约束行为，请参考SET option命令的COMPILEMODE=NOCHECK选项。

当为单个字段定义ADD FOREIGN KEY约束且外键引用引用表的idkey时， IRIS将外键中的属性转换为引用属性。
此转换受以下限制:

- 该表不能包含任何数据。
- 外键上的属性不能是持久类(也就是说，它不能已经是引用属性)。
- 外键字段与引用的idkey字段的数据类型和数据类型参数必须相同。
- 外键字段不能是IDENTITY字段。

减少约束限制

默认情况下，如果外键约束引用唯一键约束或主键约束，则不能删除该约束。这样做会导致SQLCODE-317错误。要更改此默认外键约束行为，请参考SEToption命令的COMPILEMODE=NOCHECK选项。

删除主键约束的效果取决于主键也是ID键设置的设置(如上所述)：
- 如果PrimaryKey索引不是IDKey索引，则删除PRIMARY KEY约束将删除索引定义。
- 如果PrimaryKey索引也是IDKey索引，并且表中没有数据，则删除PRIMARY KEY约束将删除整个索引定义。
- 如果PrimaryKey索引也是IDKey索引，并且表中有数据，则删除PRIMARYKEY约束只会从IDKey索引定义中删除PRIMARYKEY限定符。

不存在时删除约束

默认情况下，IRIS拒绝在没有该约束的字段上删除字段约束的尝试，并发出SQLCODE-315错误。要确定当前设置，请调用$SYSTEM.SQL.CurrentSettings()，它显示允许DDL丢弃不存在的约束设置。默认值为0(否)，这是推荐设置。如果此选项设置为1(是)，ALTER TABLE DROP CONSTRAINT将导致
IRIS不执行任何操作，也不发出错误消息。

在管理门户、系统管理、配置、SQL和对象设置中，通过选中忽略冗余DDL语句复选框，可以在系统范围内设置此选项(以及其他类似的创建、更改和删除选项)。

示例

以下示例使用嵌入式SQL程序创建表，填充两行，然后更改表定义。

为了演示这一点，请按显示的顺序运行前两个嵌入式SQL程序。(这里有必要使用两个嵌入式SQL程序，因为除非引用的表已经存在，否则嵌入式SQL无法编译INSERT语句。)

```
ClassMethod AlterTable()
{
    DO $SYSTEM.Security.Login("_SYSTEM","SYS")
    &sql(
        DROP TABLE SQLUser.MyStudents
    )
    IF SQLCODE = 0 {
        WRITE !,"????"
    } ELSE {
        WRITE "DROP TABLE??SQLCODE=",SQLCODE
    }
    &sql(
        CREATE TABLE SQLUser.MyStudents
        (
            FirstName VARCHAR(35) NOT NULL,
            LastName VARCHAR(35) NOT NULL
        )
    )
    IF SQLCODE = 0 {
        WRITE !,"????"
    } ELSE {
        WRITE "CREATE TABLE??SQLCODE=",SQLCODE
    }
}


ClassMethod AlterTable1()
{
    DO $SYSTEM.Security.Login("_SYSTEM","SYS")
    NEW SQLCODE, %msg
    &sql(
        INSERT INTO SQLUser.MyStudents
        (
            FirstName, LastName
        )
        VALUES
        (
            'Yao','Vanderbilt'
        )
    )
    IF SQLCODE = 0 {
```

```
        WRITE !,"Inserted data in table"
    } ELSE {
        WRITE !,"SQLCODE=",SQLCODE,": ",%msg
    }
    &sql(
        INSERT INTO SQLUser.MyStudents
        (
            FirstName, LastName
        )
        VALUES
        (
            'Xin','Smith'
        )
    )
    IF SQLCODE = 0 {
        WRITE !,"Inserted data in table"
    } ELSE {
        WRITE !,"SQLCODE=",SQLCODE,": ",%msg
    }
}
```

下面的示例使用ALTER TABLE添加ColorPreference列。
因为列定义指定了默认值，所以系统会为表中已有的两行填充ColorPreference的值'Blue':

```
ClassMethod AlterTable2()
{
    NEW SQLCODE,%msg
    &sql(
        ALTER TABLE SQLUser.MyStudents
        ADD COLUMN ColorPreference VARCHAR(16) NOT NULL DEFAULT 'Blue'
    )
    IF SQLCODE = 0 {
        WRITE !,"????",!
    } ELSEIF SQLCODE = -306 {
        WRITE !,"SQLCODE=",SQLCODE,": ",%msg
    } ELSE {
        WRITE "SQLCODE error=",SQLCODE
    }
}
```

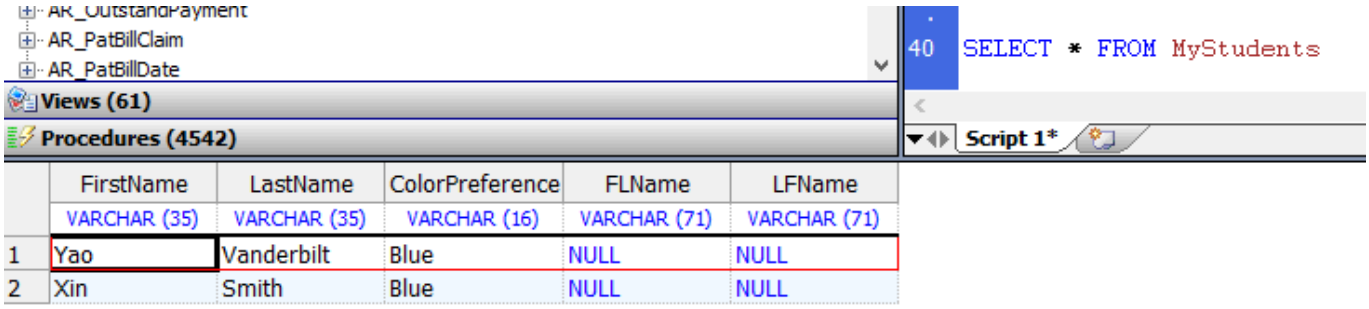
下面的示例使用ALTER TABLE添加两个计算列:FLName和LFName。
对于已存在的行，这些列没有值。
对于任何随后插入的行，将为每一列计算一个值:

```
ClassMethod AlterTable3()
{
    NEW SQLCODE,%msg
    &sql(
        ALTER TABLE SQLUser.MyStudents
            ADD COLUMN FLName VARCHAR(71) COMPUTECODE
            {
                SET {FLName}={FirstName}_" "_{LastName}
            }
```

```
            COMPUTEONCHANGE
            (
                FirstName,LastName
            ),
            COLUMN LFName VARCHAR(71) COMPUTECODE
            {
                SET {LFName}={LastName}_ "," _{FirstName}
            }
            COMPUTEONCHANGE
            (
                FirstName,LastName
            )
        )
    IF SQLCODE=0 {
        WRITE !,"???????",!
    } ELSE {
        WRITE "SQLCODE error=",SQLCODE
    }
}
```

DDL创建的 User.MyStudents 表:

```
Class User.MyStudents Extends %Persistent [ ClassType = persistent, DdlAllowed, Final
, Owner = {yx}, ProcedureBlock, SqlRowIdPrivate, SqlTableName = MyStudents ]
{

Property FirstName As %Library.String(MAXLEN = 35) [ Required, SqlColumnNumber = 2 ];

Property LastName As %Library.String(MAXLEN = 35) [ Required, SqlColumnNumber = 3 ];

Property ColorPreference As %Library.String(MAXLEN = 16) [ InitialExpression = "Blue"
, Required, SqlColumnNumber = 4 ];

Property FLName As %Library.String(MAXLEN = 71) [ SqlColumnNumber = 5, SqlComputeCode
 = {   SET {FLName}={FirstName}_" "_{LastName}
}, SqlComputed ];

Property LFName As %Library.String(MAXLEN = 71) [ SqlColumnNumber = 6, SqlComputeCode
 = {   SET {LFName}={LastName}_ "," _{FirstName}
}, SqlComputed ];

/// Bitmap Extent Index auto-
generated by DDL CREATE TABLE statement.  Do not edit the SqlName of this index.
Index DDLBEIndex [ Extent, SqlName = "%%DDLBEIndex", Type = bitmap ];

}
```



	FirstName	LastName	ColorPreference	FLName	LFName
	VARCHAR (35)	VARCHAR (35)	VARCHAR (16)	VARCHAR (71)	VARCHAR (71)
1	Yao	Vanderbilt	Blue	NULL	NULL
2	Xin	Smith	Blue	NULL	NULL

#SQL #Caché

源
URL:
https://cn.community.intersystems.com/post/%E7%AC%AC%E4%BA%8C%E7%AB%A0-sql%E5%91%BD%E4%BB%A4-alter-table%EF%BC%88%E4%BA%8C%EF%BC%89