文章
姚 鑫 · 九月 15, 2021 阅读大约需 10 分钟

第十七章 SQL命令 CREATE TABLE（四）

第十七章 SQL命令 CREATE TABLE（四）

唯一字段约束

唯一字段约束对多个字段的组合值施加唯一值约束。它具有以下语法：

```
CONSTRAINT uname UNIQUE (f1,f2)
```

此约束指定字段f1和f2的值组合必须始终是唯一的，即使这两个字段本身的值可能不是唯一的。可以为此约束指定一个、两个或多个字段。

此约束中指定的所有字段都必须在字段定义中定义。如果在此约束中指定的字段没有出现在字段定义中，则会生成SQLCODE-86错误。指定的字段应定义为非空。任何指定的字段都不应定义为唯一的，因为这会使指定此约束变得毫无意义。

字段可以按任何顺序指定。字段顺序指定相应索引定义的字段顺序。允许重复的字段名称。虽然可以在唯一字段约束中指定单个字段名称，但这与为该字段指定唯一数据约束在功能上是相同的。单字段约束确实提供了约束名称以供将来使用。

可以在表定义中指定多个唯一字段约束语句。约束语句可以在字段定义中的任何位置指定；按照惯例，它们通常放在已定义字段列表的末尾。

约束名称

Constraint关键字和唯一字段约束名称是可选的。以下各项在功能上等效：

```
CONSTRAINT myuniquefields UNIQUE (name,dateofbirth)
UNIQUE (name,dateofbirth)
```

约束名唯一地标识约束，并且还用于派生相应的索引名。
建议指定约束名称;
当使用ALTER TABLE命令从表定义中删除约束时，需要此约束名。
约束名称可以是任何有效的标识符;
如果指定为分隔符，则约束名可以包含".", "^", ",", "->"字符。

ALTER TABLE无法删除约束UNIQUE中列出的列。尝试这样做会生成SQLCODE-322错误。

RowID记录标识符

在 SQL 中，每条记录都由一个唯一的整数值标识，称为 RowID。在 SQL 中，不需要指定 RowID 字段。创建表并指定所需的数据字段时，会自动创建 RowID 字段。此 RowID

在内部使用，但未映射到类属性。默认情况下，它的存在仅在类投影到 SQL 表时可见。在这个投影的 SQL 表中，会出现一个额外的 RowID 字段。默认情况下，此字段命名为 “ ID ” 并分配给第 1 列。

%PUBLICROWID

默认情况下，RowID是隐藏和私有的。指定%PUBLICROWID关键字使RowID不会隐藏和公开。如果指定%PUBLICROWID关键字，则使用 “ not SqlRowIdPrivate ” 定义与表对应的类。此可选关键字可以在逗号分隔的表元素列表中的任何位置指定。ALTER TABLE不能用于指定%PUBLICROWID。

如果RowID是公共的:

- RowID值通过SELECT *显示。
- RowID可以用作外键引用。
- 如果没有定义主键，RowID将被视为具有约束名称RowIDFieldAsPKey的隐式主键约束。
- 如果没有指定要复制的字段名，则不能使用该表将数据复制到重复表中。

位图扩展索引

使用CREATE TABLE创建表时，缺省情况下， IRIS会自动定义相应类的位图范围索引。位图区索引的SQL MapName为%%DDLBEIndex：

```
Index DDLBEIndex [ Extent, SqlName = "%%DDLBEIndex", Type = bitmap ];
```

在以下任何情况下都不会创建该位图范围索引：

- 该表被定义为临时表。
- 该表定义了显式IDKEY索引。
- 该表包含定义的标识字段，该字段没有MINVAL=1。
- $SYSTEM.SQL.Util.SetOption()方法DDLDefineBitmapExtent选项设置为0以覆盖系统范围的默认设置。要确定当前设置，请调用$SYSTEM.SQL.CurrentSettings()方法，该方法显示a Do classes created by a DDL CREATE TABLE statement define a bitmap extent index。

如果在创建位图索引后，对自动定义位图范围索引的表调用CREATE BITMAPEXTENT INDEX，则先前定义的位图范围索引将重命名为CREATE BITMAPEXTENT INDEX语句指定的名称。

有关自动删除现有位图范围索引的DDL操作，请参阅ALTER TABLE。

IDENTITY标识字段

SQL自动为每个表创建一个RowID字段，其中包含一个系统生成的整数，作为唯一的记录id。
可选的IDENTITY关键字允许定义一个具有与RowID记录id字段相同属性的命名字段。
IDENTITY字段作为一个单字段IDKEY索引，其值是系统生成的唯一整数。

定义标识字段可防止将主键定义为IDKEY。

与任何系统生成的ID字段一样，IDENTITY字段具有以下特征:

- 每个表只能将一个字段定义为IDENTITY字段。
  试图为表定义多个IDENTITY字段会产生SQLCODE -308错误。
- IDENTITY字段的数据类型必须是整数数据类型。

如果不指定数据类型，则将其数据类型自动定义为BIGINT。
可以指定任何整数数据类型，如integer或SMALLINT;
建议使用BIGINT匹配RowID的数据类型。
接受任何指定的字段约束，如NOT NULL或UNIQUE，但忽略。

- 数据值由系统生成。
  它们由唯一的非零正整数组成。
- 默认情况下，IDENTITY字段数据值不能由用户指定。
  默认情况下，INSERT语句没有也不能指定IDENTITY字段值。
  尝试这样做会产生SQLCODE -111错误。
  要确定是否可以指定IDENTITY字段值，调用$SYSTEM.SQL.Util.GetOption("IdentityInsert")方法;
  默认值是0。要更改当前进程的此设置，请调用$SYSTEM.SQL.Util.SetOption()方法，如下所示：设置status =$SYSTEM.SQL.Util.SetOption(“ IdentityInsert ”，1，.oldval)。也可以在表定义中指定%CLASSPARAMETER ALLOWIDENTITYINSERT=1。指定ALLOWIDENTITYINSERT=1将覆盖使用SetOption(“ IdentityInsert ”)应用的任何设置。
- 不能在UPDATE语句中修改标识字段数据值。尝试这样做会生成SQLCODE-107错误。
- 系统会自动将标识字段上的主键投影到ODBC和JDBC。如果CREATE TABLE或ALTER TABLE语句在标识字段或包括标识字段的一组列上定义了主键约束或唯一约束，则会忽略约束定义，并且不会创建相应的主键或唯一索引定义。
- SELECT*语句确实返回表的标识字段。

在INSERT、UPDATE或DELETE操作之后，可以使用LASTIDENTITY函数返回最近修改的记录的标识字段的值。
如果未定义标识字段，LASTIDENTITY将返回最近修改的记录的RowID值。

以下嵌入式SQL程序创建一个具有标识字段的表，然后在表中插入一条记录，从而生成标识字段值：

```
/// d ##class(PHA.TEST.SQLCommand).CreateTable5()
ClassMethod CreateTable5()
{
    d $SYSTEM.Security.Login("_SYSTEM","SYS")
    &sql(
        CREATE TABLE Employee
        (
            EmpNum INT NOT NULL,
            MyID   IDENTITY NOT NULL,
            Name   CHAR(30) NOT NULL,
            CONSTRAINT EMPLOYEEPK PRIMARY KEY (EmpNum)
        )
    )
    if SQLCODE '= 0 {
        w !,"??????: ",SQLCODE
    } else {
        w !,"????"
    }
    &sql(
        INSERT INTO Employee
        (
            EmpNum,Name
        )
        SELECT ID,Name FROM SQLUser.Person WHERE Age >= '25'
    )
    if SQLCODE '= 0 {
        w !,"???? error is: ",SQLCODE
    } else {
        w !,"????????"
    }
}
```

在本例中，主键(EmpNum)取自另一个表的ID字段。因此，EmpNum值是唯一的整数，但是(因为WHERE子句)它们的序列中可能包含空格。标识字段myid为每条记录分配一个用户可见的唯一顺序整数。

ROWERSION、SERIAL和AUTOINCREMENT字段

SQL提供三种类型的系统生成的整数计数器字段。这三种数据类型都是扩展%Library.BigInt数据类型类的子类。

计数器类型	计数器范围	自动增加	当用户提供的值为	用户提供的值	重复的值	类型字段	计数器复位	分片表支持
AUTOINCREMENT	per-table	INSERT	NULL or 0	允许，不影响系统计数器	允许	每个表一个	截断表	Yes
SERIAL	per-serial计数器字段	INSERT	NULL or 0	允许，可增加系统计数器	Allowed	multiple per table	截断表	No
ROWVERSION	namespace-wide	INSERT and UPDATE		Not Allowed	Not Allowed	one per table	not reset	No

下面的CREATE TABLE示例定义了这些字段:

```
CREATE TABLE MyStudents (
   Name VARCHAR(16) NOT NULL,
   DOB DATE,
   AutoInc BIGINT AUTO_INCREMENT,
   Counter SERIAL,
   RowVer ROWVERSION
   )
```

指定ROWVERSION和SERIAL关键字，而不是显式的数据类型。
因此以下是有效的字段定义语法:MySerial SERIAL或MyRowVer ROWVERSION。

AUTOINCREMENT关键字在显式数据类型之后指定。也可以使用%Library.AutoIncrement数据类型定义AUTOINCREMENT字段。因此，以下是有效的字段定义语法：MyAutoInc %AutoIncrement, MyAutoInc %AutoIncrement AUTOINCREMENT, or MyAutoInc INTEGER AUTOINCREMENT。

定义主键

定义主键是可选的。定义表格时，IRIS会自动创建一个生成的字段，即RowID Field(默认名称“ ID ”)，它的作用是唯一的行标识符。在将每条记录添加到表中时， IRIS会为该记录的RowID字段分配一个唯一的不可修改的正整数。可以有选择地定义一个主键，该主键还用作唯一的行标识符。主键允许用户定义对应用程序有意义的行标识符。例如，主键可以是员工ID字段、社会保险号、患者记录ID字段或库存库存编号。

可以使用PRIMARY KEY子句将一个字段(或一组字段)显式定义为主记录标识符。定义主键有三种语法形式：

```
CREATE TABLE MyTable (Field1 INT PRIMARY KEY, Field2 INT)

CREATE TABLE MyTable (Field1 INT, Field2 INT, PRIMARY KEY (Field1))

CREATE TABLE MyTable (Field1 INT, Field2 INT, CONSTRAINT MyTablePK PRIMARY KEY (Field
1))
```

第一种语法将一个字段定义为主键；通过将其指定为主键，根据定义，该字段是唯一的，并且不为空。第二和第三种语法可用于单个字段主键，但允许包含多个字段的主键。例如，主键(Field1、Field2)。如果指定单个字段，则根据定义，此字段是唯一的，并且不为空。如果指定逗号分隔的字段列表，则每个字段都被定义为非NULL，但只要字段

值的组合是唯一值，就可以包含重复值。第三种语法允许显式地命名主键；前两种语法形式生成一个主键名称，如下所示：表名 “ PKEY ” 约束COUNT INTEGER。

主键只接受唯一值，不接受NULL。(主键索引属性不会根据需要自动定义；但是，它实际上是必需的，因为不能为主键字段归档或保存空值。)。主键的排序规则类型在字段本身的定义中指定。

作为IDKEY的主键

默认情况下，主键不是唯一的IDKEY索引。在许多情况下，这样做更可取，因为它使能够更新主键值、设置主键的排序规则类型等。在某些情况下，最好将主键定义为IDKEY索引。请注意，这会对主键的未来使用施加IDKEY限制。

如果向现有字段添加主键约束，则该字段还可能自动定义为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$SYSTEM.SQL.Util.SetOption()方法配置选项DDLPKeyNotIDKey。要确定当前设置，请调用$SYSTEM.SQL.CurrentSettings()，它显示的是通过DDL而不是ID键创建的主键；默认值为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。
  - 如果没有选中复选框(默认情况下)，则Primary Key不会成为类定义中的IDKEY索引。
    使用非IDKEY的主键访问记录的效率显著降低;
    但是，这种类型的主键值是可以修改的。
  - 如果选中了复选框，当通过DDL指定Primary Key约束时，它将自动成为类定义中的IDKEY索引。
    选择了这个选项后，数据访问更加有效，但是主键值一旦设置，就永远不能修改。

但是，如果在表中定义了IDENTITY字段，则不能将主键定义为IDKEY，即使使用了这些配置设置之一来建立将主键定义为IDKEY。

IRIS支持作为IDKEY索引的一部分的属性(字段)成为SqlComputed。
例如，父引用字段。
属性必须是一个触发的计算字段。
定义为SqlComputed的IDKEY属性仅在首次保存新Object或INSERT操作时计算。
不支持UPDATE计算，因为作为IDKEY索引一部分的字段不能被更新。

没有主键

在大多数情况下，应该显式定义主键。但是，如果未指定主键，
IRIS将根据以下规则尝试使用另一个字段作为ODBC/JDBC投影的主键：

1. 如果单个字段上有IDKEY索引，则将IDKEY字段报告为SQLPrimaryKey字段。
2. 否则，如果使用SqlRowIdPrivate=0(默认值)定义类，则将RowID字段报告为SQLPrimaryKey字段。
3. 否则，如果有IDKEY索引，则将IDKEY字段报告为SQLPrimaryKey字段。
4. 否则，不报告SQLPrimaryKey。

多个主键

只能定义一个主键。默认情况下，当主键已经存在时， IRIS拒绝定义主键的尝试，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)，
IRIS将删除现有的主键约束，并将最后指定的主键建立为表的主键。

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)。

例如，下面的CREATE TABLE语句：

```
CREATE TABLE MyTable (f1 VARCHAR(16),
CONSTRAINT MyTablePK PRIMARY KEY (f1))
```

创建主键(如果不存在)。随后的ALTER TABLE语句：

```
ALTER TABLE MyTable ADD CONSTRAINT MyTablePK PRIMARY KEY (f1)
```

生成SQLCODE-307错误。

#SQL #Caché

源URL: https://cn.community.intersystems.com/post/%E7%AC%AC%E5%8D%81%E4%B8%83%E7%AB%A0-sql%E5%91%BD%E4%BB%A4-create-table%EF%BC%88%E5%9B%9B%EF%BC%89