文章
姚 鑫 · 三月 9, 2021 阅读大约需 11 分钟

第六章 SQL定义和使用视图

第六章 SQL定义和使用视图

视图是一种虚拟表，由执行时通过SELECT语句或几个SELECT语句的UNION从一个或多个物理表中检索到的数据组成。 SELECT可以通过指定表或其他视图的任意组合来访问数据。因此，存储了视图的视图提供了物理表的所有灵活性和安全性特权。

InterSystemsIRIS®数据平台上的InterSystems SQL支持在视图上定义和执行查询的功能。

注意：不能对以只读方式安装的数据库中存储的数据创建视图。
无法在通过ODBC或JDBC网关连接链接的Informix表中存储的数据上创建视图。这是因为InterSystems IRIS查询转换对这种类型的查询使用FROM子句中的子查询。 Informix不支持FROM子句子查询。

创建一个视图

可以通过几种方式定义视图：

- 使用SQL CREATE VIEW命令（在DDL脚本中或通过JDBC或ODBC）。
- 使用管理门户的 “ 创建视图 ” 界面。

视图名称:不合格的视图名称是一个简单的标识符：MyView。合格的视图名称由两个简单的标识符组成，即模式名称和视图名称，以句点分隔：MySchema.MyView。视图名称和表名称遵循相同的命名约定，并对不合格的名称执行相同的架构名称解析。同一模式中的视图和表不能具有相同的名称。

可以使用$SYSTEM.SQL.ViewExists（）方法确定视图名称是否已存在。此方法还返回投影视图的类名称。可以使用$SYSTEM.SQL.TableExists（）方法确定表名是否已存在。

视图可用于创建表的受限子集。以下嵌入式SQL示例创建一个视图，该视图限制了可以通过该视图访问的原始表的行（通过WHERE子句）和列（假设Sample.Person包含两个以上的列）：

```
/// d ##class(PHA.TEST.SQL).View()
ClassMethod View()
{
    &sql(CREATE VIEW Sample.VSrStaff
        AS SELECT Name AS Vname,Age AS Vage
        FROM Sample.Person WHERE Age>75)
    IF SQLCODE=0 {
        WRITE "??????",!
    } ELSEIF SQLCODE=-201 {
        WRITE "??????",!
    } ELSE {
        WRITE "SQL??: ",SQLCODE," ",%msg,!
    }
}
```

```
DHC-APP>d ##class(PHA.TEST.SQL).View()
??????
```

以下嵌入式SQL示例基于SalesPeople表创建一个视图，并创建一个新的计算值列TotalPay：

```
/// d ##class(PHA.TEST.SQL).View1()
ClassMethod View1()
{
    &sql(CREATE VIEW Sample.VSalesPay AS
        SELECT Name,(Salary + Commission) AS TotalPay
        FROM Sample.SalesPeople)
    IF SQLCODE=0 {
        WRITE "??????",!
    } ELSEIF SQLCODE=-201 {
        WRITE "??????",!
    } ELSE {
        WRITE "SQL??: ",SQLCODE," ",%msg,!
    }
}
```

管理门户创建视图界面

可以从管理门户创建视图。转到InterSystems IRIS管理门户。在系统资源管理器中，选择SQL。使用页面顶部的Switch选项选择一个名称空间；这将显示可用名称空间的列表。选择名称空间后，单击“ 操作 ”下拉列表，然后选择“ 创建视图 ”。

这将显示“ 创建视图 ”窗口，其中包含以下字段：

- 模式：可以决定将视图包含在现有模式中，也可以创建一个新模式。如果选择选择现有模式，则会提供一个现有模式的下拉列表。如果选择创建新架构，请输入架构名称。在这两种情况下，如果省略模式，则InterSystems IRIS都会使用系统范围内的默认模式名称。
- 视图名称：有效的视图名称。不能对同一模式中的表和视图使用相同的名称。
- 使用Check Option：选项为READONLY，LOCAL，CASCADED。
- 将视图的所有特权授予PUBLIC：如果选中，则此选项为该视图授予所有用户执行特权。默认设置是不授予所有用户访问该视图的权限。
- 查看文字：可以通过以下三种方式中的任意一种来指定查看文字：
    - 在“ 查看文本 ”区域中键入SELECT语句。
    - 使用查询生成器创建SELECT语句，然后按OK将此查询提供给“ 查看文本 ”区域。
    - 如果在Management Portal SQL界面的左侧选择了一个缓存查询名称（例如%sqlcq.USER.cls4），然后调用Create View，则该缓存查询将提供给“ 视图文本 ”区域。请注意，在保存视图文本之前，必须在“ 视图文本 ”区域中用实际值替换主机变量引用。

视图和相应的类

定义视图时，InterSystems IRIS会生成一个相应的类。按照名称转换规则，SQL视图名称用于生成相应的唯一类名称。 Management Portal SQL界面显示现有视图的“ 目录详细信息 ”，包括此类名称。

修改视图

在Management Portal SQL界面中，可以选择一个现有视图以显示该视图的“ 目录详细信息 ”。 “ 目录详细信息视图信息 ”选项显示“ 编辑视图 ”链接，该链接提供了用于编辑视图文本（视图的SELECT语句）的界面。它还提供了一个下拉列表，以将“ 带检查选项 ”选择为无，READONLY，LOCAL或CASCADED。

可更新的视图

可更新的视图是可以在其上执行INSERT，UPDATE和DELETE操作的视图。仅当满足以下条件时，才认为视图是可更新的：

- 视图查询的FROM子句仅包含一个表引用。该表引用必须标识可更新的基表或可更新的视图。
- 视图查询的SELECT列表中的值表达式必须全部是列引用。
- 视图的查询中不得指定GROUP BY，HAVING或SELECT DISTINCT。
- 该视图不是投影为视图的类查询。
- 视图的类不包含类参数READONLY = 1（如果视图定义包含WITH READ ONLY子句，则为true）。

WITH CHECK选项

为了防止在视图上执行INSERT或UPDATE操作，而该操作会导致基础基表中的行不属于派生视图表的一部分，InterSystems SQL在视图定义中支持WITH CHECK OPTION子句。此子句只能与可更新视图一起使用。

WITH CHECK OPTION子句指定可更新视图上的任何INSERT或UPDATE操作必须对照视图定义的WHERE子句验证结果行，以确保插入或修改的行将成为派生视图表的一部分。

例如，以下DDL语句定义了一个可更新的GoodStudent视图，其中包含所有具有高GPA（平均绩点）的学生：

```
CREATE VIEW GoodStudent AS
    SELECT Name, GPA
      FROM Student
        WHERE GPA > 3.0
    WITH CHECK OPTION
```

由于视图包含WITH CHECK OPTION，因此任何尝试在GPA值小于或等于3.0的GoodStudent视图中插入或更新行都将失败（此类行将不表示“ 好学生 ”）。

有两种类型的WITH CHECK选项:
- WITH LOCAL CHECK选项意味着只检查INSERT或UPDATE语句中指定的视图的WHERE子句。
- 与级联检查选项(和级联检查选项)意味着视图的WHERE子句中指定的INSERT或UPDATE语句以及所有视图检查基于这一观点,无论外表或与当地检查没有其他选项在这些视图定义条款。

如果指定了just WITH CHECK选项，默认值是级联的。

在更新或插入期间，在为基础表的字段计算了所有默认值和触发的计算字段之后，并在常规表验证(必需字段、数据类型验证、约束等)之前，检查WITH CHECK选项条件。

在WITH CHECK选项验证通过后，插入或更新操作继续进行，就像在基表本身上执行插入或更新一样。
检查所有约束，拉出触发器，等等。

如果在INSERT或UPDATE语句中指定了%NOCHECK选项，则不检查WITH CHECK选项的有效性。

有两个与WITH CHECK选项验证相关的SQLCODE值(插入/更新会导致派生视图表中不存在一行):

- SQLCODE -136-INSERT中视图的WITH CHECK OPTION验证失败。
- SQLCODE -137-视图的WITH CHECK OPTION验证在UPDATE中失败。

只读视图

只读视图是不能在其上执行INSERT，UPDATE和DELETE操作的视图。任何不符合可更新视图标准的视图都是只读视图。

视图定义可以指定WITH READ ONLY子句，以强制其成为只读视图。

如果尝试针对只读视图编译/准备INSERT，UPDATE或DELETE语句，则会生成SQLCODE -35错误。

查看ID：%VID

InterSystems IRIS为视图或FROM子句子查询返回的每一行分配一个整数视图ID（%VID）。与表行ID号一样，这些视图行ID号是系统分配的，唯一的，非空的，非零的和不可修改的。该%VID通常对用户不可见，并且仅在明确指定时返回。它以数据类型INTEGER返回。因为%VID值是顺序整数，所以如果视图返回有序数据，它们将更有意义。视图与TOP子句配对时，只能使用ORDER BY子句。以下嵌入式SQL示例创建一个名为VSrStaff的视图：

```
/// d ##class(PHA.TEST.SQL).View()
ClassMethod View()
{
    &sql(CREATE VIEW Sample.VSrStaff
        AS SELECT Name AS Vname,Age AS Vage
        FROM Sample.Person WHERE Age>75)
    IF SQLCODE=0 {
        WRITE "??????",!
    } ELSEIF SQLCODE=-201 {
        WRITE "??????",!
    } ELSE {
        WRITE "SQL??: ",SQLCODE," ",%msg,!
    }
}
```

下面的示例返回VSrStaff视图定义的所有数据（使用SELECT *），并且还指定应返回每一行的视图ID。与表行ID不同，使用星号语法时不显示视图行ID。仅当在SELECT中明确指定时才显示：

```
SELECT *,%VID AS ViewID FROM Sample.VSrStaff
```

%VID可用于进一步限制SELECT从视图返回的行数，如以下示例所示：

```
SELECT *,%VID AS ViewID FROM Sample.VSrStaff WHERE %VID BETWEEN 5 AND 10
```

因此，可以使用%VID代替TOP（或除TOP之外）来限制查询返回的行数。通常，TOP子句用于返回数据记录的一小部分。 %VID用于返回大多数或所有数据记录，以小的子集返回记录。此功能可能很有用，尤其是对于移植Oracle查询（%VID轻松映射到Oracle ROWNUM）而言。但是，与TOP相比，用户应了解使用%VID时的一些性能限制：

- %VID不执行第一行时间优化。 TOP优化为尽快返回第一行数据。 %VID优化以尽快返回完整的数据集。
- 如果查询指定排序的结果，则%VID不会执行有限的排序（这是TOP进行的特殊优化）。该查询首先对完整的数据集进行排序，然后使用%VID限制返回数据集。
  TOP是在排序之前应用的，因此SELECT只能执行有限的排序，仅涉及有限的行子集。

为了节省第一行优化和有限排序优化的时间，可以将FROM子句子查询与TOP和%VID结合使用。在FROM子查询中指定上限（在本例中为10）作为TOP的值，而不是使用TOP
ALL。使用%VID在WHERE子句中指定下限（在这种情况下，>

4）。以下示例使用此策略返回与上一个视图查询相同的结果：

```
SELECT *,%VID AS SubQueryID
   FROM (SELECT TOP 10 Name,Age
         FROM Sample.Person
         WHERE Age > 75
         ORDER BY Name)
   WHERE %VID > 4
```

即使显式指定了%PARALLEL关键字，也无法对指定%VID的查询执行并行执行。

List视图属性

INFORMATION.SCHEMA.VIEWS持久类显示有关当前名称空间中所有视图的信息。它提供了许多属性，包括视图定义，视图的所有者以及创建和最后修改视图时的时间戳。这些属性还包括视图是否可更新，如果可更新，是否使用检查选项定义。

在嵌入式SQL中指定时，INFORMATION.SCHEMA.VIEWS需要#include%occInclude宏预处理程序指令。 Dynamic SQL不需要此伪指令。

VIEWDEFINITION属性（SqlFieldName = VIEWDEFINITION）以字符串形式返回当前名称空间中所有视图的视图字段名称和视图查询表达式。例如，

```
SELECT View_Definition FROM INFORMATION_SCHEMA.VIEWS
```

返回诸如“（vName，vAge）SELECT Name，Age FROM Sample.Person WHERE Age> 21”的字符串。当从Management Portal SQL执行查询界面发出时，此字符串的显示仅限于前100个字符，其中不包括空格和换行符，并且（如有必要）附加表示省略号的省略号（...）。否则，发出此查询将为每个视图返回最多1048576个字符的字符串，在视图字段列表和查询文本之间有一个换行符，并保留了视图查询表达式中指定的空格，并（如有必要）附加了省略号（...）表示内容被截断。

以下示例返回当前名称空间中所有视图的视图名称（TableName字段）和所有者名称：

```
SELECT Table_Name,Owner FROM INFORMATION_SCHEMA.VIEWS
```

以下示例返回当前名称空间中所有非系统视图的所有信息：

```
SELECT * FROM INFORMATION_SCHEMA.VIEWS WHERE Owner != '_SYSTEM'
```

INFORMATION.SCHEMA.VIEWCOLUMNUSAGE持久性类显示当前名称空间中每个视图的源表字段的名称：

```
SELECT * FROM INFORMATION_SCHEMA.VIEW_COLUMN_USAGE WHERE View_Name='VSrStaff'
```

可以使用管理门户网站SQL界面中的“目录详细信息”选项卡为单个视图显示与INFORMATION.SCHEMA.VIEWS相同的信息。视图的“目录详细信息”包括每个视图字段的定义（数据类型，最大长度，最小值/最大值等），以及INFORMATION.SCHEMA视图类未提供的详细信息。“目录详细信息”视图信息显示还提供了用于编辑视图定义的选项。

列出视图依赖

INFORMATION.SCHEMA.VIEWTABLEUSAGE持久类显示当前名称空间中的所有视图及其依赖的表。在下面的示例中显示：

```
SELECT View_Schema,View_Name,Table_Schema,Table_Name FROM INFORMATION_SCHEMA.VIEW_TAB
LE_USAGE
```

可以调用%Library.SQLCatalog.SQLViewDependsOn类查询以列出指定视图所依赖的表。可以为此类查询指定schema.viewname。如果仅指定视图名称，则它将使用系统范围的默认架构名称。调用者必须具有指定视图的特权才能执行此类查询。在下面的示例中显示：

```
/// d ##class(PHA.TEST.SQL).View3()
ClassMethod View3()
{
    SET statemt=##class(%SQL.Statement).%New()
    SET cqStatus=statemt.%PrepareClassQuery("%Library.SQLCatalog","SQLViewDependsOn")
    IF cqStatus'=1 {
        WRITE "%PrepareClassQuery failed:" DO $System.Status.DisplayError(cqStatus) Q
UIT
    }
    SET rset=statemt.%Execute("vschema.vname")
    DO rset.%Display()
}
```

```
DHC-APP>d ##class(PHA.TEST.SQL).View3()
```

```
Dumping result #1
SCHEMA  TABLE_NAME

0 Rows(s) Affected
```

此SQLViewDependsOn查询列出了视图所依赖的表，并列出了表架构和表名。如果调用者没有该视图所依赖的表的特权，则该表及其模式将列为<NOT PRIVILEGED>。这允许没有表特权的调用者确定视图所依赖的表数量，而不是表的名称。

#SQL #Caché #InterSystems IRIS #InterSystems IRIS for Health

源URL: https://cn.community.intersystems.com/post/%E7%AC%AC%E5%85%AD%E7%AB%A0-sql%E5%AE%9A%E4%B9%89%E5%92%8C%E4%BD%BF%E7%94%A8%E8%A7%86%E5%9B%BE