文章
Qiao Peng · 三月 29, 2021 阅读大约需 8 分钟

# CDC系列之三 ：建立InterSystems IRIS/Caché的Global数据变更与SQL表记录的对应关系

一些熟悉SQL的用户希望用SQL表的方式获取InterSystems IRIS/Caché的变更数据。知道了Global和SQL表的对应关系，就可以知道是哪一张SQL表数据变化了，甚至通过SQL查询获取变更的数据。
下面介绍如何实现这种方式，和注意事项。

## 获取Global和SQL表的对应关系

通常InterSystems IRIS/Caché的持久化的对象模型（类）和SQL表之间有一一对应的关系；而持久化的对象模型和Global之间也有一一对应关系。建立Global和SQL表的对应关系，通常可以使用以下的SQL查询特定SQL schema下所有表对应的Global：

```
SELECT CC.SqlQualifiedNameQ SQLTable, CS.parent Class, CS.DataLocation
FROM %Dictionary.CompiledStorage CS, %Dictionary.CompiledClass CC
WHERE CS.parent = CC.ID
AND CC.SqlSchemaName= <schemaname>
```

其中<schemaname>是SQL的Schema名称;
返回字段SQLTable是SQL表名、Class是对象类名、DataLocation是保存数据的Global名称。

## 多种建模方式Global和SQL表的对应关系的影响

InterSystems IRIS/Caché都是支持多种建模方式的数据平台，常见的建模方式有SQL、面向对象、多维数组。如果之前不了解InterSystems IRIS/Caché的多维数组，可以先简单理解为键值对。
无论使用何种建模方式，都可以得到3套模型：SQL模型、对象模型和多维数组存储模型（Global模型）。
上面提到通常InterSystems IRIS/Caché的持久化的对象模型（类）和SQL表之间有一一对应的关系。但由于SQL表达模型的局限性，InterSystems IRIS/Caché的对象模型和SQL表之间并不总是一一对应的关系。

下面就逐一分析各种建模方式下，如何分析和获取Global和SQL表的对应关系。

### 1. 基于SQL建模

如果InterSystems IRIS/Caché模型就是用SQL建模的，查找Global和SQL表的对应关系很简单：对象模型是基于SQL模型自动创建的，因此它们之间是一对一的关系。
这是，在Caché里，编译出的Storage 类型为%Library.CacheStorage；在InterSystems IRIS里，编译出的Storage类型为%Storage.Persistent。

可以执行SQL查询：

```
SELECT CC.SqlQualifiedNameQ as SQLTable, CS.parent as Class, CS.DataLocation
```

```
FROM %Dictionary.CompiledStorage CS, %Dictionary.CompiledClass CC
WHERE CS.parent = CC.ID
AND CC.SqlSchemaName= <schemaname> AND type='%Library.CacheStorage'
```

其中<schemaname>是SQL的Schema名称，如果Schema里有 “ _” ，应将其去掉。
Parent字段是SQL表对应的类名，DataLocation是保存数据的Global名称。
而Global的第一个下标就是行号/Id字段。

例如使用DDL创建一个table：

```
Create table Demo.Mytable(name varchar(20), notes varchar(100))
```

通过上面的SQL查询，DataLocation为Demo.MytableD。那么Journal中所有对于Global为Demo.MytableD的操作就是对表Demo.Mytable的记录操作。例如^Demo.MytableD(123),就是对行号/Id字段为123的Demo.Mytable记录到操作，从而可以通过SQL: SELECT name, notes from Demo.Mytable WHERE id = 123 获取这条变更的记录。

## 2. 基于对象建模

如果InterSystems IRIS/Caché模型是使用对象建模的，查找Global和SQL表的对应关系有时并不那么简单。
例如以下描述患者及地址的简单对象模型：

```
Class Demo.Address Extends %SerialObject
{
Property Type As %String;
Property City As %String;
Property Street As %String;
Property RoomNo As %String;
}
```

```
Class Demo.Patient Extends %Persistent
{
Property Name As %String;
Property Gender As %String;
Property DOB As %Date;
Property Addresses As list Of Demo.Address(SQLPROJECTION = "table", STORAGEDEFAULT =
"array");
}
```

因为患者可能有多个地址，因此Demo.Patient类用以列表类型（list）描述地址属性。
其中地址Demo.Address对象模型是通过被引用的持久化对象来序列化的，此处引用它的持久化对象类就是Demo.Patient。Demo.Address类并没有独立的Global保存其数据，它的数据是保存在Demo.Patient的Global中的。

这个对象模型很容易理解，以对象方式在InterSystems IRIS/Caché里也很容易操作数据。但SQL的二维表无法表达这样稍微复杂一点的模型，因此需要将患者的地址投射为一张地址表，并用主外键将地址表和患者表的记录关联起来。

上面的患者对象模型中Addresses属性的SQLPROJECTION和STORAGEDEFAULT参数就是将属性Addresses投射为一张SQL表。

编译后，这个患者对象模型，在SQL上会投射出两张表：
患者表：Demo.Patient
地址表：Demo.PatientAddresses
注意：SQL表Demo.PatientAddresses并不是由对象类Demo.Address投射而来，它是由对象类Patient的列表类型的属性Addresses投射而来。对象类Demo.Address是序列化类，它并不会投射SQL表。

执行SQL查询

```
SELECT CC.SqlQualifiedNameQ as SQLTable, CS.parent as Class, CS.DataLocation
FROM %Dictionary.CompiledStorage CS, %Dictionary.CompiledClass CC
WHERE CS.parent = CC.ID
AND CC.SqlSchemaName= 'Demo'
```

将返回类似如下结果：

| SQLTable | Class | DataLocation |
|---|---|---|
| Demo.Patient | Demo.Patient | ^Demo.PatientD |
| Demo.PatientAddresses | Demo.Patient | |

注意：SQL表Demo.PatientAddresses并没有对应的Global，因为它的数据是保存在Patient的Global里的。

这时可以使用以下SQL查询获取SQL表Demo.PatientAddresses对应的Global和下标：

```
SELECT CC.ID||'_'||CSD.Attribute as SQLTable, CS.parent as Class, CS.DataLocation, CS
D.Structure, CSD.Subscript
FROM %Dictionary.CompiledStorage CS, %Dictionary.CompiledClass CC, %Dictionary.Compil
edStorageData CSD
WHERE CS.parent = CC.Name
  AND CS.ID1 = CSD.parent
  AND CC.SqlSchemaName= 'Demo'
  AND CC.ID||CSD.Attribute in (select parent from %Dictionary.CompiledStorage where D
ataLocation is null)
```

它返回类似如下结果：

| SQLTable | Class | DataLocation | Structure | Subscript |
|---|---|---|---|---|
| Demo.PatientAddresses | Demo.Patient | ^Demo.PatientD | subnode | “ Addresses ” |

这说明SQL表Demo.PatientAddresses的数据放在Global
^Demo.PatientD的下标为"Addresses"的子节点下。所以对Global节点
^Demo.PatientD( “ Addresses ” )的数据变更就是对SQL表Demo.PatientAddresses的数据变更。

类似的，当创建对象模型的父子关系时，父子关系子方的数据可以保存在父方的Global中。如以下模型：
患者模型

```
Class Demo.Patient Extends %Persistent
{
Property Name As %String;
Property Gender As %String;
Property DOB As %Date;
Property Addresses As list Of Demo.Address(SQLPROJECTION = "table", STORAGEDEFAULT =
"array");
Relationship Encounters As Demo.Encounter [ Cardinality = children, Inverse = Patient
 ];
}
```

就诊模型，它和患者模型是父子关系

```
Class Demo.Encounter Extends %Persistent
{
Property EncounterNo As %String;
Property VisitDate As %Date;
Relationship Patient As Demo.Patient [ Cardinality = parent, Inverse = Encounters ];
}
```

执行SQL查询

```
SELECT CC.SqlQualifiedNameQ as SQLTable, CS.parent as Class, CS.DataLocation
FROM %Dictionary.CompiledStorage CS, %Dictionary.CompiledClass CC
WHERE CS.parent = CC.ID
AND CC.SqlSchemaName= 'Demo'
```

将返回类似如下结果：

| SQLTable | Class | DataLocation |
|---|---|---|
| Demo.Encounter | Demo.Encounter | **{%%PARENT}( “ Encounters ” )** |
| Demo.Patient | Demo.Patient | ^Demo.PatientD |
| Demo.PatientAddresses | Demo.PatientAddresses | |

这说明SQL表Demo.Encounter的数据放在Global ^Demo.PatientD的下标为" Encounters"的子节点下。所以对Global ^Demo.PatientD( “ Encounters ” )的数据变更就是对SQL表Demo.Encounter的数据变更。

## 3. 基于Global建模

直接基于Global建模并不常见。如果是直接基于Global建模的，可以在Global模型的基础上再建立对象模型，这样数据不仅可以使用多维数组方式操作，也可以通过对象和SQL方式操作。

这种情况下的对象模型，使用的Storage 类型在Caché里为%CacheSQLStorage，在InterSystems IRIS里为%Storage.SQL。

例如如下使用%CacheSQLStorage的Caché对象类Demo.Department：

```
Class Demo.Department Extends %Persistent [ StorageStrategy = SQLStorage ]
{
Property Id As %Integer;
Property Name As %String;
Property Parent As Demo.Department;
Index MyId On Id [ IdKey ];
Storage SQLStorage
{
<SQLMap name="DataMap">
<Data name="Id">
<Delimiter>"^"</Delimiter>
<Node>"id"</Node>
<Piece>1</Piece>
</Data>
<Data name="Name">
<Delimiter>"^"</Delimiter>
<Node>"Name"</Node>
<Piece>1</Piece>
</Data>
```

```
<Data name="Parent">
<Delimiter>"^"</Delimiter>
<Node>"Parent"</Node>
<Piece>1</Piece>
</Data>
<Global>^MyDepartment</Global>
<RowIdSpec name="1">
<Field>Id</Field>
</RowIdSpec>
<Subscript name="1">
<Expression>{Id}</Expression>
</Subscript>
<Subscript name="2">
<Expression>"Dep"</Expression>
</Subscript>
<Type>data</Type>
</SQLMap>
<StreamLocation>^Demo.DepartmentS</StreamLocation>
<Type>%CacheSQLStorage</Type>
}
}
```

对于这种使用%CacheSQLStorage或%Storage.SQL的对象类所投射出的SQL表，可以使用以下SQL查询获取SQL表对应的Global和下标：

```
SELECT CC.SqlQualifiedNameQ as sqltable,
CC.id as class, CSM._Global as DataLocation, CSM.Structure, CSMS.Name as subscript,CS
MS.Expression
FROM %Dictionary.CompiledStorage CS,
%Dictionary.CompiledClass CC,
%Dictionary.CompiledStorageSQLMap CSM,
%Dictionary.CompiledStorageSQLMapSub CSMS
WHERE CS.parent = CC.ID
AND CS.ID1 = CSM.parent
AND CSM.ID = CSMS.parent
AND CSM.Type='data'
AND CC.SqlSchemaName= <schemaname>
```

其中<schemaname>是SQL的Schema名称。
它返回类似如下结果：

| SQLTable | Class | DataLocation | Structure | Subscript | Expression |
|---|---|---|---|---|---|
| Demo.Department | Demo.Department | ^MyDepartment | | 1 | {Id} |
| Demo.Department | Demo.Department | ^MyDepartment | | 2 | “ Dep ” |

说明SQL表Demo.Department的数据保存在^MyDepartment
中，并且放在2个下标下，第一维下标为Id字段，第二维下标为字符串常量"Dep"。

这时，可以通过^MyDepartment的第一维下标（字段Id）值，执行SQL语句获取变更的整条记录：

```
SELECT * FROM Demo.Department WHERE id=?
```

使用%CacheSQLStorage/%Storage.SQL为Storage类型的InterSystems IRIS/Caché对象类，其Global模型可以任意灵活，而且不影响对象操作和SQL操作。但这也提高了将Global变更对应到SQL表的难度。

## 总结

通过上面的分析，如果要从InterSystems
IRIS/Caché中通过SQL方式分析数据变更，需要先了解其建模方式，分析Global对应SQL表的关系。
可以建立一张SQL表，存储分析得到的Global及下标和SQL表对应关系。当通过Dejournal filter发现global数据变更时，查询该SQL表，将数据变更表达为对应SQL表和对应记录（RowID），从而使用SQL获取完整的变化记录。

**其它注意事项：**
流类型的属性/字段，通常保存在名为S的Global内，因此这些Global的数据更新也应该捕获并转换为对应SQL表的数据变更记录。

## CDC系列

更多的CDC选项实现，请参考：

1. CDC系列之一 ：使用Dejournal Filter在InterSystems IRIS/Caché上通过Mirroring实现CDC功能

2. CDC系列之二 ：使用Dejournaling filter routine在Caché上通过Shadow实现CDC

3. CDC系列之三 ：建立InterSystems IRIS/Caché的Global数据变更与SQL表记录的对应关系

4. CDC系列之四：使用DSTIME特性在InterSystems IRIS/Caché上实现CDC功能

#Caché #Ensemble #InterSystems IRIS #InterSystems IRIS for Health

---

**源**
**URL:**
https://cn.community.intersystems.com/post/cdc%E7%B3%BB%E5%88%97%E4%B9%8B%E4%B8%89-%EF%BC%9A%E5%BB%BA%E7%AB%8Bintersystems-iriscach%C3%A9%E7%9A%84global%E6%95%B0%E6%8D%AE%E5%8F%98%E6%9B%B4%E4%B8%8Esql%E8%A1%A8%E8%AE%B0%E5%BD%95%E7%9A%84%E5%AF%B9%E5%BA%94%E5%85%B3%E7%B3%BB