文章
姚 鑫 · 五月 5, 2021 阅读大约需 6 分钟

第三章 使用多维存储(全局变量)（一）

第三章 使用多维存储(全局变量)（一）

本章描述了使用多维存储(全局变量)可以执行的各种操作。

以全局变量存储数据

在全局节点中存储数据很简单:像对待任何其他变量一样对待全局变量。
区别在于对全局变量的操作是自动写入数据库的。

创建全局变量

创建新的全局变量不需要设置工作；只需将数据设置为全局变量即可隐式创建新的全局结构。可以创建全局变量(或全局变量下标)并通过单个操作将数据放入其中，也可以创建全局变量(或下标)并通过将其设置为空字符串将其保留为空。在ObjectScript中，这些操作是使用SET命令完成的。

下面的例子定义了一个名为Color(如果还不存在)的全局变量，并将值“ Red ”与之关联。
如果已经存在一个名为Color的全局变量，那么这些示例将其修改为包含新信息。

在ObjectScript中：

```
SET ^Color = "Red"
```

注意：在应用程序中使用直接全局访变量问时，应制定并遵守命名约定，以防止应用程序的不同部分相互“ 遍历 ”；这类似于为类、方法和其他变量开发命名约定。

在全局变量节点中存储数据

要在全局下标节点中存储值，只需像设置任何其他变量数组一样设置全局节点的值。如果指定的节点以前不存在，则会创建该节点。如果它确实存在，则其内容将替换为新值。

可以通过表达式(称为全局引用)指定全局内的节点。全局引用由脱字符(^)、全局名称和(如果需要)一个或多个下标值组成。下标(如果有)用括号“ () ”括起来，并用逗号分隔。每个下标值本身都是一个表达式：文字值、变量、逻辑表达式，甚至是全局引用。

设置全局节点的值是一个原子操作：它肯定会成功，不需要使用任何锁来确保并发性。

以下都是有效的全局引用：

在ObjectScript中：

```
SET ^Data = 2
SET ^Data("Color")="Red"
```

```
SET ^Data(1,1)=100          /*?????(1?1)????100??????(^DATA(1))??????? */
SET ^Data(^Data)=10         /*????^data????????? */
SET ^Data(a,b)=50           /*????a?b???????? */
SET ^Data(a+10)=50
```

此外，还可以在运行时使用间接方式构造全局引用。

在全局变量节点中存储结构化数据

每个全局节点可以包含最多32K个字符的单个字符串。

数据通常以以下方式之一存储在节点中：

- 作为最多32K个字符的单个字符串(具体地说，32K - 1)。
- 作为包含多条数据的字符分隔字符串。

要使用字符分隔符在节点中存储一组字段，只需使用连接操作符()将这些值连接在一起。下面的ObjectScript示例使用#字符作为分隔符：

```
SET ^Data(id)=field(1)_"#"_field(2)_"#"_field(3)
```

检索数据时，可以使用$PIECE函数将字段拆分：

```
SET data = $GET(^Data(id))
FOR i=1:1:3 {
    SET field(i) = $PIECE(data,"#",i)
}
QUIT
```

- 作为包含多条数据的$LIST编码字符串。

$LIST函数使用特殊的长度编码方案，不需要保留分隔符。(这是InterSystems IRIS对象和SQL使用的默认结构。)

要在节点中存储一组字段，请使用$LISTBUILD函数构造列表：

```
SET ^Data(id)=$LISTBUILD(field(1),field(2),field(3))
```

检索数据时，可以使用$LIST或$LISTGET函数将字段拆分：

```
SET data = $GET(^Data(id))
FOR i = 1:1:3 {
    SET field(i)=$LIST(data,i)
}
QUIT
```

- 作为较大数据集(例如流或 “ BLOB ”)的一部分。

由于单个节点的数据量限制在略低于32K，因此可以通过将数据存储在一组连续节点中来实现更大的结构(如流)：

```
SET ^Data("Stream1",1) = "First part of stream...."
SET ^Data("Stream1",2) = "Second part of stream...."
SET ^Data("Stream1",3) = "Third part of stream...."
```

获取流的代码(如%GlobalCharacterStream类提供的流)循环遍历结构中的连续节点，该结构将数据作为连续字符串提供。
- 作为一个位串。

如果正在实现位图索引(位字符串中的位对应表中的行的索引)，应该将全局索引的节点值设置为位字符串。
请注意IRIS使用压缩算法来编码位串;
因此，位串只能使用IRIS $BIT函数来处理。

- 作为一个空节点。

如果感兴趣的数据是由节点本身提供的，那么通常将实际下标设置为空字符串("")。
例如，将名称与ID值相关联的索引通常是这样的:

```
SET ^Data("APPLE",1) = ""
SET ^Data("ORANGE",2) = ""
SET ^Data("BANANA",3) = ""
```

删除全局节点

要从数据库中删除一个全局节点、一组子节点或整个全局节点，请使用ObjectScript kill或ZKILL命令。

Kill命令删除特定全局引用处的所有节点(数据及其在数组中的相应条目)，包括任何子代节点。也就是说，所有以指定下标开头的节点都将被删除。

例如，ObjectScript语句：

```
KILL ^Data
```

删除整个^Data全局变量。对此全局变量的后续引用将返回<UNDEFINED>错误。

ObjectScript语句：

```
KILL ^Data(100)
```

删除^Data全局变量中节点100的内容。如果有子代节点，如^data(100，1)、^data(100，2)和^data(100，1，2，3)，这些子节点也会被删除。

ObjectScript ZKILL命令用于删除指定的全局或全局下标节点。它不会删除子代子节点。

注意：在杀死一个大型全局变量之后，该全局变量曾经占用的空间可能没有完全释放，因为垃圾收集器守护进程在后台将这些块标记为空闲。因此，在终止大型全局变量之后立即调用SYS.Database类的ReturnUnusedSpace方法可能不会返回预期大小的空间，因为该全局占用的块可能尚未释放。

不能对全局变量使用new命令。

测试全变量局节点的存在

要测试特定全局变量(或其后代)是否包含数据，请使用$DATA函数。

$DATA返回一个值，该值指示指定的全局变量引用是否存在。可能的返回值包括：

状态值	含义
0	全局变量未定义。
1	全局变量存在并包含数据，但没有子代。请注意，空字符串(“ ”)可用作数据。
10	全局变量有后代(包含指向子节点的向下指针)，但本身不包含数据。对此类变量的任何直接引用都将导<UNDEFINED>错误。例如，如果$data(^y)返回10，则SET x=^y将产生<UNDEFINED>错误。
11	全局变量既包含数据，又有后代(包含指向子节点的向下指针)。

检索全局变量节点的值

要获取存储在特定全局变量节点中的值，只需使用全局引用作为表达式：

```
SET color = ^Data("Color")    ; assign to a local variable
WRITE ^Data("Color")          ; use as a command argument
SET x=$LENGTH(^Data("Color")) ; use as a function parameter
```

$GET函数

还可以使用$GET函数获取全局节点的值：

```
SET mydata = $GET(^Data("Color"))
```

这将检索指定节点的值(如果存在)，如果该节点没有值，则返回空字符串(“ ”)。如果节点没有值，可以使用可选的第二个参数$get返回指定的默认值。

WRITE、ZWRITE和ZZDUMP命令

可以使用各种ObjectScript显示命令显示全局变量或全局变量子节点的内容。WRITE命令以字符串形式返回指定全局或子节点的值。ZWRITE命令返回全局变量的名称及其值，以及它的每个子代节点及其值。ZZDUMP命令以十六进制转储格式返回指定全局或子节点的值。

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

源URL: https://cn.community.intersystems.com/post/%E7%AC%AC%E4%B8%89%E7%AB%A0-%E4%BD%BF%E7%94%A8%E5%A4%9A%E7%BB%B4%E5%AD%98%E5%82%A8%E5%85%A8%E5%B1%80%E5%8F%98%E9%87%8F%EF%BC%88%E4%B8%80%EF%BC%89