文章
姚 鑫 · 五月 2, 2021 阅读大约需 5 分钟

第一章 简介global

第一章 简介global

InterSystems IRIS®的核心功能之一是其多维存储引擎。此功能允许应用程序以紧凑、高效的多维稀疏数组存储数据。这些数组称为全局数组。

本章介绍：
- 什么是全局变量(globals)，以及可以对其执行的操作。
- 全局变量的逻辑和物理结构，包括在分布式数据库体系结构中使用全局变量。
- 如何使用全局变量在应用程序中存储和检索数据。
- 如何使用全局变量。

特点

全局变量提供了一种在持久的多维数组中存储数据的易于使用的方法。

例如，可以使用名为^Settings的全局变量将值 “ Red ” 与键 “ Color ” 相关联：

```
SET ^Settings("Color")="Red"
```

可以利用全局变量的多维特性来定义更复杂的结构：

```
 SET ^Settings("Auto1","Properties","Color") = "Red"
 SET ^Settings("Auto1","Properties","Model") = "SUV"
 SET ^Settings("Auto2","Owner") = "Mo"
 SET ^Settings("Auto2","Properties","Color") = "Green"
```

全局变量具有以下功能：
- 简单易用-全局变量和其他编程语言变量一样易于使用。
- 多维-可以使用任意数量的下标指定全局内节点的地址。
例如，在 ^Settings("Auto2","Properties","Color")中，下标Color是全局设置中的第三级节点。
下标可以是整数、数字或字符串值，并且不需要是连续的。
- 稀疏-用于寻址全局节点的下标高度压缩，不需要连续的值。
- 高效-全局变量上的操作(插入、更新、删除、遍历和检索)都经过高度优化，可实现最高性能和并发性。还有用于特殊操作(如批量插入数据)的其他命令。有一组特殊的全局变量是为临时数据结构设计的(例如，用于对记录进行排序)。
- 可靠-InterSystems IRIS数据库提供了许多机制来确保存储在全局数据库中的数据的可靠性，包括逻辑级和物理级日志记录。执行数据库备份操作时，将备份存储在全局数据库中的数据。
- 分布式IRIS提供了多种方法来控制存储在全局数据库中的数据的物理位置。可以定义用于存储全局的物理数据库，或将全局的部分分布到多个数据库中。使用InterSystems IRIS的分布式数据库功能，可以在数据库和应用程序服务器系统网络中共享全局数据。此外，通过镜像技术，存储在一个系统上的全局数据库中的数据可以自动复制到另一个系统上。
-
并发-全局支持多个进程之间的并发访问。在单个节点(数组元素)中设置和检索值始终是原子的：不需要锁定即可保证

可靠的并发访问。此外，InterSystems IRIS支持一组强大的锁定操作，可用于为涉及多个节点的更复杂情况提供并发性。使用对象或SQL访问时，会自动处理此并发。
- 事务性-InterSystems IRIS提供定义事务边界的命令；可以启动、提交或回滚事务。在回滚的情况下，事务内对全局变量所做的所有修改都将被撤消；数据库的内容将恢复到事务前的状态。通过将各种InterSystems IRIS锁定操作与事务结合使用，可以使用全局变量执行传统的ACID事务。(ACID事务提供原子性、一致性、隔离性和持久性。)。使用对象或SQL访问时，事务会自动处理。

注意：本文档中描述的全局变量不应与另一种类型的InterSystems IRIS数组变量混淆：进程私有全局变量。进程私有全局变量不是持久的；它们仅在创建它们程序期间持续。进程私有全局变量也不是并发的；它们只能由创建它们的进程访问。进程专用全局可以通过其多字符名称前缀：^||或^|"^"|轻松地与全局区分开来。

例如

一个简单的例子就可以展示全局变量的易用性和性能。下面的程序示例创建一个10,000个节点的数组(如果存在，则首先将其删除)并将其存储在数据库中。可以尝试这样做，以了解全局变量的性能：

```
/// w ##class(PHA.TEST.Global).GlobalSimple()
ClassMethod GlobalSimple()
{
    Set start = $ZH  // get current time

    Kill ^Test.Global
    For i = 1:1:10000 {
    Set ^Test.Global(i) = i
    }

    Set elap = $ZH - start  // get elapsed time
    Write "Time (seconds): ",elap

    q ""
}
```

```
DHC-APP> w ##class(PHA.TEST.Global).GlobalSimple()
Time (seconds): .00307
```

我们还可以看到迭代和读取数组中的值需要多长时间(确保首先运行上面的示例来构建数组)：

-读取持久数组-

```
/// w ##class(PHA.TEST.Global).ReadGlobalSimple()
ClassMethod ReadGlobalSimple()
{
    Set start = $ZH  // get current time
    Set total = 0
    Set count = 0

    // get key and value for first node
    Set i = $Order(^Test.Global(""),1,data)

    While (i '= "") {
        Set count = count + 1
        Set total = total + data
```

```
        // get key and value for next node
        Set i = $Order(^Test.Global(i),1,data)
    }

    Set elap = $ZH - start  // get elapsed time

    Write "Nodes:           ",count,!
    Write "Total:           ",total,!
    Write "Time (seconds): ",elap,!

    q ""
}


DHC-APP>w ##class(PHA.TEST.Global).ReadGlobalSimple()
Nodes:          10000
Total:          50005000
Time (seconds): .001879
```

在应用程序中使用

在InterSystems IRIS应用程序中，全局变量有多种使用方式，包括：
- 作为对象和SQL引擎共享的底层存储机制。
- 作为用于为对象和SQL数据提供各种索引(包括位图索引)的机制。
- 作为用于执行不适合进程存储器的某些操作的工作空间。例如，当没有预先存在的索引可用于排序数据时，SQL引擎使用临时全局变量对数据进行排序。
- 用于在对象或SQL访问方面难以表达或效率低下的持久性对象或SQL表上执行专用操作。例如，可以定义一个方法(或存储过程或Web方法)来对表中保存的数据执行专门的分析。通过使用方法，这样的操作是完全封装的；调用者只需调用该方法。
- 实现特定于应用程序的自定义存储结构。许多应用程序需要存储难以用关系表示的数据。使用全局变量，可以定义自定义结构，并通过对象方法将其提供给外部客户端。
- 用于InterSystems IRIS系统使用的各种特殊用途的数据结构，例如配置数据、类定义、错误消息和可执行代码。

全局变量不受关系模型的限制。
它们提供了开发针对特定应用程序优化的定制结构的自由。
对于许多应用程序来说，明智地使用全局变量可能是提供性能的秘密武器，而这种性能是关系应用程序开发人员梦寐以求的。

无论应用程序是否直接使用全局变量，了解它们的操作都是有用的。
理解全局及其功能将帮助设计更高效的应用程序，并为确定应用程序的最佳部署配置提供帮助。

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

源URL: https://cn.community.intersystems.com/post/%E7%AC%AC%E4%B8%80%E7%AB%A0-%E7%AE%80%E4%BB%8Bglobal