文章
姚 鑫 · 五月 8, 2021 阅读大约需 7 分钟

第三章 使用多维存储(全局变量)（四）

第三章 使用多维存储(全局变量)（四）

管理事务

InterSystems IRIS提供了使用全局变量实现完整事务处理所需的基本操作。
InterSystems IRIS对象和SQL自动利用这些特性。
如果直接将事务性数据写入全局变量，则可以使用这些操作。

事务命令是TSTART，它定义事务的开始;
TCOMMIT，它提交当前事务;
和TROLLBACK，它将中止当前事务，并撤消自事务开始以来对全局变量所做的任何更改。

例如，下面的ObjectScript代码定义了事务的开始，设置了一些全局变量节点，然后根据ok的值提交或回滚事务:

```
/// w ##class(PHA.TEST.Global).GlobalTro(0)
ClassMethod GlobalTro(ok)
{

    TSTART

    Set ^Data(1) = "Apple1"
    Set ^Data(2) = "Berry1"

    If (ok) {
        TCOMMIT
    }
    Else {
        TROLLBACK
    }
    zw ^Data
    q ""
}
```

TSTART在InterSystems IRIS日志文件中写入事务开始标记。
这定义了事务的起始边界。
在上面的示例中，如果变量ok为true(非零)，则TCOMMIT命令标记事务成功结束，并将事务完成标记写入日志文件。
如果ok为false(0)，那么TROLLBACK命令将撤消自事务开始以来进行的每一个set或kill操作。
在这种情况下，^Data(1)和^Data(2)被恢复到原来的值。

注意，在事务成功完成时，不会写入任何数据。
这是因为事务期间对数据库的所有修改都是在事务过程中正常执行的。
只有在回滚的情况下，数据库中的数据才会受到影响。
这意味着本例中的事务具有有限的隔离性;
也就是说，其他进程可以在事务提交之前看到修改后的全局值。

这通常被称为未提交的读取。
这是好是坏取决于应用程序的需求;
在许多情况下，这是完全合理的行为。
如果应用程序需要更高级别的隔离，则可以通过使用锁来实现。
这将在下一节中进行描述。

锁和事务

要创建隔离事务-也就是说，为了防止其他进程在提交事务之前看到修改的数据-需要使用锁。在ObjectScript中，可以通过lock命令直接获取和释放锁定。锁按照约定工作；对于给定的数据结构(如用于持久对象)，所有需要锁的代码都使用相同的逻辑锁引用(即，锁命令使用相同的地址)。

在事务中，锁有一个特殊的行为;
在事务过程中获取的任何锁在事务结束之前都不会被释放。
要了解为什么会这样，请考虑典型事务执行的操作:
1. 使用TSTART启动事务。
2. 获取要修改的一个或多个节点上的锁。这通常被称为“ 写 ”锁。
3. 修改一个或多个节点。
4. 释放锁(或多个锁)。因为我们处于事务中，所以这些锁在此时实际上不会被释放。
5. 使用TCOMMIT提交事务。此时，上一步中释放的所有锁实际上都已释放。

如果另一个进程想要查看此事务中涉及的节点，并且不想看到未提交的修改，则它只需在从节点读取数据之前测试锁(称为“ 读 ”锁)。因为写锁定一直保持到事务结束，所以在事务完成(提交或回滚)之前，读取进程看不到数据。

大多数数据库管理系统使用类似的机制来提供事务隔离。InterSystems IRIS的独特之处在于它让开发人员可以使用这种机制。这使得有可能为新的应用程序类型创建自定义数据库结构，同时仍然支持事务。当然，可以简单地使用InterSystems IRIS对象或SQL来管理数据，并让事务得到自动管理。

对TSTART的嵌套调用

InterSystems IRIS维护一个特殊的系统变量$TLEVEL，该变量跟踪TSTART命令被调用的次数。$TLEVEL从值0开始；每次调用TSTART时，$TLEVEL的值递增1，而每次调用TCOMMIT时，$TLEVEL的值递减1。如果调用TCOMMIT导致将$TLEVEL设置回0，则事务结束(以COMMIT结束)。

调用TROLLBACK命令总是终止当前事务，并将$TLEVEL设置回0，而不管$TLEVEL的值是多少。

此行为使应用程序能够将事务包装在本身包含事务的代码(如对象方法)周围。例如，持久对象提供的%Save方法始终将其操作作为事务执行。通过显式调用TSTART和TCOMMIT，可以创建包含几个对象保存操作的更大事务：

```
TSTART
Set sc = object1.%Save()
If ($$$ISOK(sc)) {
    // ??????????????
    Set sc = object2.%Save()
}

If ($$$ISERR(sc)) {
    // ????????????
    TROLLBACK
}
Else {
    // ??
    TCOMMIT
}
```

管理并发性

设置或检索单个全局变量节点的操作是原子的；它可以保证始终成功并获得一致的结果。对于多个节点上的操作或控制事务隔离，InterSystems IRIS提供获取和释放锁的功能。

锁由IRIS锁管理器管理。在ObjectScript中，可以通过lock命令直接获取和释放锁定。(InterSystems IRIS对象和SQL根据需要自动获取和释放锁)。

检查最新的全局变量引用

最新的全局变量引用记录在ObjectScript $ZREFERENCE特殊变量中。$ZREFERENCE包含最新的全局引用，包括下标和扩展全局引用(如果指定)。请注意，$ZREFERENCE既不指示全局引用是否成功，也不指示指定的全局是否存在。InterSystems IRIS只记录最近指定的全局引用。

裸全球变量引用

在带下标的全局引用之后，InterSystems IRIS会将裸指示符设置为该全局名称和下标级别。然后，可以使用裸全局引用(省略全局名称和更高级别的下标)对相同的全局变量和下标级别进行后续引用。这简化了在相同(或更低)下标级别对相同全局变量的重复引用。

在裸引用中指定较低的下标级别会将裸指示符重置为该下标级别。因此，在使用裸全局变量引用时，始终使用由最新全局引用建立的下标级别。

裸指示符值记录在$ZREFERENCE特殊变量中。裸露指示符被初始化为空字符串。在未设置裸指示器的情况下尝试裸全局引用会导致<NAKED> 错误。更改命名空间会重新初始化裸体指示符。可以通过将$ZREFERENCE设置为空字符串(“ ”)来重新初始化裸指示符。

在下面的示例中，第一个引用中指定了带下标的GLOBAL ^Produce(“ fruit ” ,1)。InterSystems IRIS将此全局变量名称和下标保存在裸体指示符中，以便后续的裸体全局引用可以省略全局名称 “ Production ” 和更高下标级别的 “ Fruit ” 。当^(3，1)裸引用达到更低的下标级别时，此新的下标级别将成为任何后续裸全局变引用的假设。

```
/// w ##class(PHA.TEST.Global).GlobalNake()
ClassMethod GlobalNake()
{
    SET ^Produce("fruit",1)="Apples"  /* ????????  */
    SET ^(2)="Oranges"                /* ????????? */
    SET ^(3)="Pears"                  /* ????????2 */
    SET ^(3,1)="Bartlett pears"       /* ???????3  */
    SET ^(2)="Anjou pears"            /* ????????3 */
    WRITE "latest global reference is: ",$ZREFERENCE,!
    ZWRITE ^Produce
    KILL ^Produce
    q ""
}
```

```
DHC-APP>w ##class(PHA.TEST.Global).GlobalNake()
latest global reference is: ^Produce("fruit",3,2)
^Produce("fruit",1)="Apples"
^Produce("fruit",2)="Oranges"
^Produce("fruit",3)="Pears"
^Produce("fruit",3,1)="Bartlett pears"
^Produce("fruit",3,2)="Anjou pears"
```

除了极少数例外，每个全局变量变引用(全引用或裸引用)都会设置裸指示器。$ZREFERENCE特殊变量包含最新全局变引用的完整全局名称和下标，即使这是一个裸全局引用。ZWRITE命令还显示每个全局的完整全局名称和下标，无论它是否使用裸引用设置。

应谨慎使用裸全局变量引用，因为InterSystems IRIS在不总是明显的情况下设置裸指示器，包括以下情况：
- 完整全局变量引用最初设置裸露指示符，随后的完整全局引用或裸露全局引用会更改裸露指示符，即使全局引用不成功。例如，试图写入不存在的全局变量的值会设置裸指示符。
- 无论InterSystems IRIS如何计算后置条件，引用下标全局的后置条件命令都会设置裸指示符。
- 引用下标全局变量的可选函数参数可能设置或不设置裸指示符，具体取决于IRIS是否计算所有参数。例如，$get的第二个参数总是设置裸指示符，即使它包含的默认值没有使用。InterSystems
IRIS按从左到右的顺序计算参数，因此最后一个参数可能会重置由第一个参数设置的裸指示符。
- 回滚事务的TROLLBACK命令不会将裸指示符回滚到事务开始时的值。

如果完整全局变量引用包含扩展全局变量引用，则后续的裸全局变量引用将采用相同的扩展全局引用；不必将扩展引用指定为裸全局引用的一部分。

#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%E5%9B%9B%EF%BC%89