

文章

姚鑫 · 六月 19, 2022 阅读大约需 3 分钟

第四章 锁定和并发控制（四）

第四章 锁定和并发控制（四）

避免死锁

增量锁定具有潜在危险，因为它可能导致称为死锁的情况。当两个进程各自对已被另一个进程锁定的变量断言增量锁定时，就会出现这种情况。因为尝试的锁是增量的，所以现有的锁不会被释放。结果，每个进程在等待另一个进程释放现有锁的同时挂起。

举个例子：

1. 进程 A 发出此命令：`lock + ^MyGlobal(15)`
2. 进程 B 发出此命令：`lock + ^MyOtherGlobal(15)`
3. 进程 A 发出此命令：`lock + ^MyOtherGlobal(15)`

此 LOCK 命令不返回；进程被阻塞，直到进程 B 释放这个锁。

4. 进程 B 发出此命令：`lock + ^MyGlobal(15)`

此 LOCK 命令不返回；进程被阻塞，直到进程 A 释放这个锁。但是，进程 A 被阻塞，无法释放锁。现在这些进程都在等待对方。

有几种方法可以防止死锁：

- 始终包含 `timeout` 参数。
- 对于发出增量 LOCK 命令的顺序，请遵循严格的协议。只要所有进程都遵循相同的锁名称顺序，就不会发生死锁。一个简单的协议是按排序顺序添加锁。
- 使用简单锁定而不是增量锁定；也就是说，不要使用 `+` 运算符。如前所述，对于简单锁定，LOCK 命令首先释放进程持有的所有先前锁定。（然而，在实践中，简单的锁定并不经常使用。）

如果发生死锁，可以使用管理门户或 `^LOCKTAB`

锁的实际用途

本节介绍在实践中使用锁的基本方法。

控制对应用程序数据的访问

锁经常用于控制对存储在全局变量中的应用程序数据的访问。应用程序可能需要读取或修改此数据的特定部分，并且应用程序将在执行此操作之前创建一个或多个锁，如下所示：

- 如果应用程序需要读取一个或多个全局节点，并且不希望其他进程在读取操作期间修改这些值，请为这些节

点创建共享锁。

- 如果应用程序需要修改一个或多个全局节点，并且不希望其他进程在修改期间读取这些节点，请为这些节点创建排他锁。

然后按计划阅读或进行修改。完成后，取下锁。

请记住，锁定机制纯粹按照约定工作。任何其他将读取或修改这些节点的代码也必须在执行这些操作之前尝试获取锁。

防止同步行为

锁也用于防止多个进程执行相同的活动行为。在这种情况下，还使用了GLOBAL，但GLOBAL包含用于应用程序内部目的的数据，而不是纯应用程序数据。作为一个简单的示例，假设有一个例程 (^NightlyBatch)，在任何给定时间都不应由多个进程运行。该例程可以在其处理的早期阶段执行以下操作：

1. 在特定全局节点上创建排他锁，例如 ^AppStateData("NightlyBatch")。为此操作指定超时。
2. 如果获得锁，则在全局中设置节点以记录例程已启动（以及任何其他相关信息）。例如：

```
set ^AppStateData("NightlyBatch")=1
set ^AppStateData("NightlyBatch","user")=$USERNAME
```

或者，如果在超时期限内未获得锁，则退出并显示错误消息，指示该例程已启动。

然后，在其处理结束时，同一例程将清除适用的全局节点并释放锁。

以下部分示例演示了这种技术，该技术改编自内部使用的代码：

```
/// w ##class(PHA.TEST.AdvancedConcepts).Lock()
ClassMethod Lock()
{
    lock ^AppStateData("NightlyBatch"):0
    if '$TEST {
        write "??????????"
        write !, "????????????: "_^AppStateData("NightlyBatch","user")
        quit
    }
    set ^AppStateData("NightlyBatch") = 1
    set ^AppStateData("NightlyBatch","user") = $USERNAME
    set ^AppStateData("NightlyBatch","starttime") = $h
    b
    kill ^AppStateData("NightlyBatch")
    lock -^AppStateData("NightlyBatch")
}
```

[#SQL #Caché](#)

源

URL:

<https://cn.community.intersystems.com/post/%E7%AC%AC%E5%9B%9B%E7%AB%A0-%E9%94%81%E5%AE%9A%E5%92%8C%E5%B9%B6%E5%8F%91%E6%8E%A7%E5%88%B6%EF%BC%88%E5%9B%9B%EF%BC%89>

