SQL Server 数据库设计

SQL Server 数据库设计

服务器整合能够给你的前期投资带来丰厚的回报，它需要花很多的计划来完成整合。但是在最后，你需要管理的服务器就更少了，许可证费用也更低。另外一份报酬就是告诉你的 CIO 你目前正向这绿色数据中心前进。本篇技术专题除了介绍 SQL Server 整合技术之外，还结合数据库架构师 Brian Walker 在 SQL Server Shop 里的一些启发，和大家一起分享 SQL Server 数据库设计灾难，即数据库设计和 SQL Server 性能的完善技巧。更重要的是，在本文中将列出 10 个常见的需要避免的查询设计错误。

SQL Server 整合

多年以来，SQL Server 整合已经成为一种广泛应用的策略。随着服务器发展越来越强大，它们能够在单个硬件上处理更多的服务。由于在多年前就引进了多核处理机，现在拥有同样数量物理 CPU 的服务器能够在许可要求不增长的情况下处理更多的工作。从整合的角度看来，微软 SQL Server 的最新版本并没有改变很多——在 SQL Server 2005 和 SQL Server 2008 中的技术仍然是相同的。

- ❖ 为什么 SQL Server 整合是一种优化技术

SQL Server 数据库设计灾难

如果一个外行很容易看懂你的 SQL Server 数据库设计，你是不是会感觉很尴尬？有没有可能在这些表上安装一个 foreign key constraints？你用正确的列的数据类型了吗？有没有对 naming conventions 进行限制？或者，数据库架构在其他人看来是不是很乱？数据库架构师 Brian Walke 最近在 SQL Server Shop 里得到了一些启发，在本文中和我们分享一下完善的数据库设计和 SQL Server 性能。

- ❖ SQL Server 数据库设计灾难：不该做什么

- SQL Server 数据库设计灾难：它是如何开始的

SQL Server 数据库查询设计

随着 SQL Server 数据库的填充和持续的数据增长，以及用户对于次秒级响应时间的期望，避免编写糟糕的查询就是至关重要的了。在这篇文章中，将会列出 10 个常见的需要避免的查询设计错误。阅读此文，确保你不会成为这些错误的受害者，考虑给出的建议，修改你的查询。

- SQL Server 查询设计：避免 10 个错误（一）
- SQL Server 查询设计：避免 10 个错误（二）

SQL Server 数据库同步、复制、报告显示

你可通过查询 sys.数据库视图以及或者数据库兼容性视图查找用户创建的 SQL Server 数据库名。在使两个单独服务器上的 SQL Server 数据库同步的问题上，如果仅仅是为了故障转移，你可以用 log shipping；如果你运行的是 SQL Server 2005，你可以使用数据库镜像进行数据库同步。如果你想拥有两个数据库，你需要使用 SQL Server 复制。若 SQL Server 数据库被标注了“suspect”，你首先做的第一步就是放松，因为恐慌不会使你解决任何问题……

- 有没有能查找用户创建 SQL Server 数据库名的查询？
- 使两个单独活动服务器上的 SQL Server 数据库同步
- 标注了“suspect”的 SQL Server 数据库
- 如何在升级到 SQL Server 2005 时复制数据库？
- 如何在 Excel 中显示 SQL Server 数据库报告？
- 接收 data notification：SQL Server 触发器 VS.存储程序

为什么 SQL Server 整合是一种优化技术

多年以来，SQL Server 整合已经成为一种广泛应用的策略。随着服务器发展越来越强大，它们能够在单个硬件上处理更多的服务。由于在多年前就引进了多核处理机，现在拥有同样数量物理 CPU 的服务器能够在许可要求不增长的情况下处理更多的工作。从整合的角度看来，微软 SQL Server 的最新版本并没有改变——在 SQL Server 2005 和 SQL Server 2008 中的技术仍然是相同的。

许可

服务器整合最大的益处就是许可。一旦完成迁移，由于现在的服务器数量少了，所以需要的许可证数量也就减少了。如果你使用的是基于 CPU 的许可证，那你可以节省一部份费用了。而对于基于 CAL 许可来说费用节省得就要少一些，因为服务器许可证比 CPU 许可证要便宜一些。

如果你没有从微软 SQL Server 的一个版本升级到另一个版本，你可能还不会意识到许可证的节省费用。如果你还停留在 SQL Server 的同一个版本上，你也不会看到许可证的节省费用——你也许不需要购买新的许可证。因为许可证版本相同，你可能将许可证从一台旧服务器迁移到新服务器上。然而，你可以免费在新 SQL Server 上使用另外的许可证。

一些应用软件可能由于某些原因需要与其他的软件隔离区分。例如，它们可能有大量的随机存储内存 RAM 或者设涉及到全实例（instance-wide）安全情况。在安装其他实例时，你必须正确许可这一实例。在你使用 CPU 许可证时，你就不需要为实例购买另外的许可证——一套 CPU 许可证就包括了所有安装在服务器上的实例。

硬件

对 SQL Server 进行整合会使你的其余的服务器不能再使用。你可以将这些服务器作其他用途，还可以将它们用作应用软件或者是开发服务器。

由于新服务器上数据库数量在增加，在你设计新服务器硬件时需要另外的硬件。这些硬件资源包括更多的 RAM，在磁盘子系统中需要更多的 spindle。

在你设计新的数据库服务器时要考虑设置群集解决方案。由于被动节点的花费，这样做就会增加你最初的系统花费，但是从长远来看，减少的系统停机时间就会满足它自身的需要。

设计和建立集群式数据库解决方案应该通过有经验的数据库管理员和 Windows 管理员来完成。建立集群式解决方案比单独的 SQL Server 要难得多。

在一个集群上由多个集群式的实例时，你仍然只需要许可你的活动节点。如果所有的实例在一个单独的节点上运行，那么只有这个节点必须得到许可。如果你的这些实例在一个以上的节点上运行，那么所有在 SQL Server 上运行的节点就必须得到许可。如果你用的是 CPU 许可证并且有两个活动节点，每个节点上有四个 CPU，那么你就需要许可八个 CPU。

系统监视

监视一个新整合的 SQL Server 就意味着在购买代理程序的时候会减少你的开支。在你监视系统检查性能问题时，很难将一个数据库的系统负载和另一个数据库分开。这是因为性能监视器对 CPU 有不同的计算器，并且内存使用是建立在每个数据库的基础之上。查看着故障信息的唯一方法就是将数据库隔离到单独的实例之上。

随着 SQL Server 可见性的增强，更多的应用软件和更多用户连接到了同一个 SQL Server 上，你就会更加关注系统性能。

整合 gotchas

由于连接到数据库服务器上的应用软件的数量逐渐增加，单个数据库就已经出现了性能问题，其他在数据库服务器上的应用软件也出现了性能问题。在对系统上的数据库进行调优时你必须更加细心。

在 SQLServer 上安装服务包、进行热修理时你需要更多的时间测试所有应对新版本的应用软件。你能够对他们进行安装时，服务包和热修理之间会出现其他多余时间。所以你的服务器在更长的时间内很容易受到安全问题的影响。

可能现在一个服务器上有几个部门的应用软件，这就意味着各部门之间的政策可能成为了一个或多个问题——更是一个部门应用软件第一次引起的性能问题影响另一个不部门的应用软件。

在 SQL Server2000、2005 和 2008 上整合

在进行时，你需要考虑同时升级 SQL Server 版本。从以前旧版的微软 SQL Server 到更新的版本，你会发现另外的性能改善使你将更多的数据库迁移到服务器上。虽然其中一个版本的升级在您整理项目大大增加了合并的复杂性，最终的结果会减少整个服务器。

服务器整合能够给你的前期投资带来丰厚的回报，它需要花很多的计划来完成整合。但是在最后，你需要管理的服务器就更少了，许可证费用也更低。另外一份报酬就是告诉你的 CIO 你目前正向这绿色数据中心前进。这对上层管理和投资者来说是很好的消息——他们也在做自己该做的那部分，在改善环境的同时还能盈利。

(作者： Denny Cherry 译者：April 来源：TT 中国)

SQL Server 数据库设计灾难：不该做什么

如果一个外行很容易看懂你的 SQL Server 数据库设计，你是不是会感觉很尴尬？有没有可能在这些表上安装一个 foreign key constraints？你用正确的列的数据类型了吗？有没有对 naming conventions 进行限制？或者，数据库架构在其他人看来是不是很乱？数据库架构师 Brian Walke 最近在 SQL Server Shop 里得到了一些启发，在本文中和我们分享一下完善的数据库设计和 SQL Server 性能。

一些事物数据库的环境十分糟糕。

在五周内我有检查了三个业务数据库，我仍然难以相信我的发现。有一些 SQL Server 数据库公司至关重要的业务基础。它们每年能产生数百万的收入。每天成百上千的员工在这些数据库上进行计算，而这些数据库也充分发挥了它们的精确性、稳定性以及性能。就个人来说，我并不期望这些数据库能够在我的 iPod.里存储包含有 2600 首歌的一个简单的目录。

这些数据库都支持它们各自的事物，但是它们却遭遇到了性能问题。它们有许多问题，但是业务、服务器硬件以及产品数据库管理员都围绕在这些问题周围。这些问题处理起来是非常困难的，因为一些应用程序已经进行了编码以弥补数据库设计的缺陷。由于多余的花费以及丧失的生产力，业务也收到了损害。数据库设计还使 SQL Server 看起来非常脆弱。

过去的不幸

那么，我检查过的业务数据库哪里出了问题？所有的地方都出了问题，而且很严重。它们的规格化程度很差。有一些表没有 primary key constraints。表与表之间的关系并不是通过 foreign key constraint 来约定的，也只是偶然使用一下索引（indexing）。

基本的业务逻辑被隐藏在一些触发器里。此外，一些列的数据类型也不合适。那么一致性怎么样呢？那就不得而知了。

规范化的一处很容易理解，我也将不会对他进行详细阐述。我很有把握说，这些数据库中的许多表都没有在 1NF 里。没有带 primary key constraints 的表也显然没有达到 INF 的标准。此外，一些表里包含也含有多个限定值的列。大部分表里的大多数列都是无效的，也违反了 INF 标准。所以个人说来，我认为无效列对于开发人员来说也很痛苦。

安装 foreign key constraints 应该是第二个习性，好处就是能确保数据库的完整性。这些数据库中的一些表中包含许多的孤立的行，它们缺少 Declarative Referential Integrity（DRI）的原因就是像 COBOL 程序员一样的失落。一个数据库中的一些表的处境让我不颤而栗。详细情况如下：

如果你有一个 Book 表，这个表带有 AuthorID 外键列。在大多数行里面，AuthorID 列带有一个值向 Editor 表的外键。另一列表示 AuthorID 列里含有哪一类指向。因此，安装一个 foreign key constraint 就很有可能。太棒了！

接下来就是完成安装并让它能编写出一些很有用的 SQL 代码。例如，可以编写所有外键的索引，这些索引能够极大地提高包含 join 子句的语句性能，它们还能够从一开始就作为一个巨大的基地，而不是像性能调优一样一个接一个地增加索引。又例如，它能够产生审计和登陆的 SQL 触发器。总之，虽然业务逻辑属于存储程序，我还是认为触发器应该用来执行这些任务。

我检查的数据库有很多不适当的列。它们有一系列的单字节（标准的）字符串以及双字节（国际的）字符串。它们有一些定宽字符串如 20, 30, 40 和 50+字符串的宽度，在小的 datetime 数据类型超过所需的量时，它们也有一些含有 datetime 数据类型的列。

在你对 naming convention 进行限定时，一定要覆盖所有的数据库。选择单数或者复数的表明、选择使用缩写、大写、特殊字符还是前缀或者后缀。给所有的对象限定一个

naming convention。包括表、列（主键、外键或者是其他的对象），检查 constraints 和缺省值。任何对象都可以。

表架构里的一致性和可预见性具有很大的益处。例如，表架构对归档和清除旧数据的影响很大。利用关系型数据库是很难处理的一件事情。选择一个 parent row（或者一系列 parent row）并且和在每个派生的表格和每个相关的行一起进行复制或迁移。如果所有的表都有一致的架构，那么你就可以利用普通的存储程序进行归档和清除了。

我检查的几个数据库同时使用了不同的表架构。主键列，外键列以及审计列都不能预测到。IDENTITY 权限的使用也不能预测到，因此，归档和清除文件需要很小心编写自定义 SQL 代码来处理每个表，可结果还是导致了不能很好的维护。我认为表架构可能是一些事务数据库中的最需要认真对待的。

在第二部分里，Walker 将会介绍事务数据库在恶劣的条件下会带来什么后果，你又能做些什么来避免这些麻烦。

(作者：Brian Walker 译者：April 来源：TT 中国)

SQL Server 数据库设计灾难：它是如何开始的

这是数据库架构师 Brian Walker 关于数据库设计方面系列文章的第二部分。他写这一系列的文章是受到最近一些 SQL Server 销售店情况的启发。最终，他提出了一些关于改善数据库设计以及 SQL Server 性能的建议。

发生了什么：滚雪球效应

那么，事物数据库是如何处在一个糟糕的环境中的呢？一些数据库一开始就处在这样的状态中。仔细考虑一下，他们有的是会计绘制的 Excel 表格，或者是一些销售商组装电脑的 Access 数据库。增加大量的数据后，他们就有可能以原来的格式被迁移到了 SQL Server。这种解决方案在真正的服务器上能够广泛使用时，更多的人就开始使用这些数据库了。最后，业务就依靠这些混合型的数据库进行。

一旦业务依赖其中的一个数据库，集合里的数据就不断上涨，更多的应用程序都会利用这一数据库。拥有二流的数据库技术或 SQL 的应用开发人员会根据他们的主观意愿在数据库中增加一些表和其他的东西。没有人会调整数据库的扩展速度，他们只在乎通过数据库能够完成什么任务。在数据库性能出现问题时，就雇佣一名 DBA。DBA 自然会推荐一台力量更强大的服务器。DBA 随后设置了一些存储程序并且增加许多索引。性能改善之后，业务自然就继续开展。

当数据库开始扩展并增加一系列应用程序的时候，DBA 正忙于尝试解决数据库的性能问题。他/她建议用更多的服务器和数据库的副本将数据库负载分开。另一名 DBA 监控服务器的操作。数据库扩展持续了好几年的时间，现在业务就完全依赖数据库了。有了好几台强大的服务器就能尽可能地分开负载，DBA 团队总是不停地在进行战斗，但是还是会出现一些数据库性能问题。

最后，DBA 团队承认一些性能问题可能是由数据库设计自身引起的。他们接着就提出了“最佳实践”。但是最佳实践太少也提出得太迟了！DBA 团队和应用程序开发人员对数据库设计所负的责任并不一样。修补设计漏洞也需要对应用程序密码修改。否则业务就不授予他们权限。这样混合数据库就不会再改变了。

避免出现问题

你的公司是不是也处在上面几点中的某一个过程中？我希望没有。我希望你的公司能够以正确的方式做事情。有很多方式可以避免膨胀循环以及性能问题，即避免混合数据库让业务无法进行下去的问题。

从一开始就建立牢固的数据库基础。从开始的 Excel/Access 工具中找出业务逻辑、让一名有经验的数据库开发员在合适的关系数据库框架里安装。

有经验的数据库开发员能够规范表、建立一个标准的表结构，安装完整的 DRI、创建基本索引、选择合适的数据类型以及定义有用的自定环境。如果这些事情在前面已经做完了，那么之后的业务就会有所回报。

应用程序开发人员能更有效地使数据库更合理地扩展，如能够简单快速存档和清除。合理设计的数据库很难受到性能问题的影响。他们可能会更加有效地支持数据数量的增长或者应用程序和用户的增加（或者两者皆可）。

不要假设一般的 DBA 就是数据库开发员。他们不是同一个概念。他们的工作有一些地方相同，但是需要不同的技巧。

一般可以理解为有两种数据库管理员——产品数据库管理员和研发数据库管理员。但是对每种数据库管理员的定义都不是一样的。如果不同之处仅仅是他们的环境（产品或者研发），那就没有必要把他们区分开来了。

我有一个建议可以作出很清楚的区别：产品 DBA 就是关心数据库以外所有使用问题。而研发 DBA（数据库开发人员）关心的是数据库里面的所有事情。

一名产品 DBA 认为数据库是他所关注的最小的项目。研发 DBA 处理数据模式、规范化、表结构、DRI、索引、数据类型、命名约定、触发器、存储程序、视图、函数、密码生成、数据导入/导出、归档、清除以及审计。他们主要关注的对象就是一个数据库或者是一系列的数据库。研发 DBA（数据库开发员）必须懂得业务是如何运作的。

正确方法：产品 DBA 和研发 DBA

业务世界里这两种 DBA 的存在很有必要。然而，如果在数据库项目的早期阶段就牵涉到开发 DBA，那数据库就会更加强大，服务器硬件就更加省钱，产品 DBA 要解决的问题就更少了。

工作记录并没有显示业务会以正确的方式进行运作。他们仍然在寻找只有二流数据库技术或 SQL 技术的应用程序开发人员，也仍然在寻找产品 DBA 来解决这些问题。而对于研发 DBA 来说机会很少。

我担心这意味着业务数据库一开始就是混合数据库。这样他们创造的这种糟糕的条件会危害到以后的业务。

我的工作经常就是产品 DBA 或者应用程序开发人员，但是我喜欢成为开发 DBA（数据库开发人员）。

(作者：Brian Walker 译者：April 来源：TT 中国)

SQL Server 查询设计：避免 10 个错误（一）

随着 SQL Server 数据库的填充和持续的数据增长，以及用户对于次秒级响应时间的期望，避免编写糟糕的查询就是至关重要的了。在这篇文章中，将会列出 10 个常见的需要避免的查询设计错误。阅读此文，确保你不会成为这些错误的受害者，考虑给出的建议，修改你的查询。

前十个列表

1、数据模型和并发查询

如果你在构建数据模型的时候没有考虑到数据的访问方式，将会导致难以处理的查询。你可能会用到根本不必要的 join 增加代码，损害性能。

要纠正这个问题，考虑一下需要访问数据的查询。如果查询在这个处理阶段不是很清晰，那么将来在写代码的时候就会更困难。很有可能是数据库设计过于复杂，可以通过简化来改善查询的性能。

与此相关，如果你是个喜欢直观的人，那么就打印出数据模型，或者在你选择数据建模工具的时候查看一下在线的模型。这可以改善你的代码时间和精确性。

2、什么是最好的技巧?

这就是声名狼藉的指针与基于集合的逻辑讨论。传统的至理名言说，对所有的数据库访问都使用基于集合的逻辑。一般来说，我同意这是最好的经验之一。当基于集合的逻辑是正确的选择的时候，却使用了指针，可能会对性能产生很大的损害。SQL Server 的设计是使用基于集合的逻辑，并且在大多数处理中应该使用它。

事分两面，另一面就是指针的例子。在这种情况下，指针逻辑胜过基于集合的逻辑。从这个信息引申出来的结论就是，判断你要执行的处理的类型，选择最适合需要的技巧。

3、以原有的方式……

SQL Server 2005 为你的查询提供了一整套新的机会。所以使用老的办法可能仍然会起作用，但是也是时候去考虑一下最新的选择了。TRY…CATCH 错误处理方法是你最先应该使用在代码中的技巧之一。此外还要考虑的是对层次进行处理的时候，可以用到通用表压缩;最后一项考虑是扩展关系型数据库引擎的功能:通用语言运行时(CLR)。这三项技术都在极大程度上改变了你使用 SQL Server 工作的方式，而它们只是冰山一角。

4、你还让一只笨鹅在那里吗?

检查你的代码，然后安排一个时间进行同样的查看，这是在部署代码之前必须要做的事情。检查你的代码，明确查询计划，是确保使用了合适的索引，并且查询会像你期望的那样运行的重要保障。

5、这是经典错误

输入 select *语句，想着表永远不会改变，这是一个经典的查询设计错误。即使在最简单的解决方案中，表的改变也是不可避免的，你需要查看代码确保没有包含一个额外的字段。或者，更糟糕的是，你必须等待应用程序崩溃，然后修正这些问题。最好的实践方案只是在你的查询中包含进来你需要的那些字段，然后必要的话就修改它们。不要把你的时间浪费在四处冒烟的模式中彻查代码。

(作者：Jeremy Kadlec 来源：TT 中国)

SQL Server 查询设计：避免 10 个错误（二）

6、我没有注释

不幸的是，我见过的大多数代码都很少或者根本没有注释。所以进行更改是一件令人畏惧的任务，即使是对那些最初开发了这个应用程序的开发人员和/或数据库管理员。注释你的代码真的是一个快速并且不痛苦的过程，对于未来的开发人员以安全和省时的方式理解和修改代码来说，这是至关重要的。

7、当然，我会测试的……

很少开发人员和数据库管理员会喜欢简单的测试，他们也不喜欢在发布代码到产品环境之前进行严格的测试。并且，开发环境通常在硬件和数据量上都达不到产品环境的规模。就是说，简单的查询在几百个或者甚至是几千个记录上都可以工作良好，但是在产品环境中就不是这样了。对于你的查询没有别的更好的准备办法了，只有在测试环境中对含有碎片的表中几百万条数据进行测试，以此来确保查询会按照你的期望运行。

8、让我用这个吧，就是这个！

输入 select 语句，没有包含 where 子句，期望中间层或者前端以比 SQL Server 更加有效的方式来处理得到的数据，这是个很糟糕的主意。SQL Server 就是设计用来处理查询，并且将其执行得非常高效的。将大量的数据移动只会让被洪水包围的系统和网络陷入困境。一定要尽可能地过滤你的数据，避免对性能产生影响。

9、请让我用视图来查询吧

视图可以满足你简化复杂查询中的代码的需求。它们通常用来帮助有权利的用户查询数据库。不幸的是，太多的好事情也会严重影响性能。视图就是一个简单的 select 语句，视图的 select 语句必须在每次你输入 select 语句的时候再次输入。限制视图的使

用，防止它们查询其他视图。或者，构建一个存储过程来查询数据，并且传递给它需要的参数来满足应用程序或者用户的需求。

10、不，这不是我的代码……

我们都犯错误，我们最后工作的那个系统应该是从我们在当前系统中获得的知识中获益。所以，记录你学到的东西，并且把它与你的团队共享，让集体受益。当你有机会的时候，回到先前的系统中，用你从那个项目中学到的知识改善它们。

结论

如果你在查询中犯了以上或者其他的错误，承认错误，努力去纠正它。说起来容易做起来难，但是纠正这些问题会让企业获益，并且对应用程序的名声有好处。看完了这篇文章，开始为你正在做或者以后要做的项目构建一个私人的代码指南吧。

（作者：Jeremy Kadlec 来源：TT中国）

有没有能查找用户创建 SQL Server 数据库名的查询？

问：你能给我介绍一个能获取所有 SQL Server 数据库名（用户创建的）的查询吗？

答：你可以查询 sys.数据库视图以及或者数据库兼容性视图。

（作者：Greg Low 译者：April 来源：TT 中国）

使两个单独活动服务器上的 SQL Server 数据库同步

问：我想用类似镜像的东西，而不是两个服务器上活动的数据库。服务器基本上就是用来将信息置入同一数据库、但是需要负载平衡排列。并且如果其中一个出现了故障，另外一个就会承担负载和流通数据。如果不用 SAN 解决方案，有没有什么方法能够让两个单独的服务器同步呢？

答：如果仅仅是为了故障转移，你可以用 log shipping；如果你运行的是 SQL Server 2005，你可以使用数据库镜像。如果你想拥有两个数据库，你需要使用 SQL Server 复制。

(作者：Greg Robidoux 译者：April 来源：TT 中国)

标注了“suspect”的 SQL Server 数据库

问：我的 SQL Server 数据库被标注了“suspect”。我该怎么做？

答：第一步就是放松。恐慌不会使你解决任何问题，碰到这一类 SQL Server 的问题要格外小心！所以首先请你深呼吸一口并记住一切都会好起来的。

其次，阅读由 Tibor Karaszi’撰写的 Recommended actions for corrupt or suspect databases 一文。如果你已经执行到了第六部但是还是没有对你的数据库进行备份并让它运行，那就该从你过去的备份中恢复。但是你可能仍有机会重新获取数据库中已经改变的数据。SQL Server 2005 介绍了一种新的数据库状态，叫做“EMERGENCY”。这种状态将数据库置于只读、单个用户模式中，也自己允许拥有 sysadmin 权限的人访问。这样一来，你就要在全部恢复前获取数据。

为了将数据库置入 EMERGENCY 状态中，可以使用下面的 T-SQL 语句代替你的数据库名。

```
ALTER DATABASE YourDatabase SET EMERGENCY;
```

(作者：Adam Machanic 译者：April 来源：TT 中国)

如何在升级到 SQL Server 2005 时复制数据库？

问：我打算将大概 300 个数据库从 SQL Server 2000 32-bit 迁移到 SQL Server 2005 64-bi。我已经尝试了使用“复制数据库”的功能但是失败了。此外我还需要移开所有登录。希望你能给我提一些建议。

答：我发现数据库复制向导很慢，在效率方面要比迁移数据的其他方法差一些。如果你需要在一系列服务器之间进行少量数据的快速迁移，那它就是最佳解决方法，你也不必花时间对数据库进行备份或者卸载数据库。

如果你需要迁移 300 个数据库，我觉得你会发现最快的方法就是在 SQL Server 2000 实例上使用 sp_detach_db 卸载每个数据库并且在 SQL Server 2005 实例上重建数据库。这样做还有利于保护你的数据库用户。

但是即使用这种方法你也仍然要登录 SQL Server 2005 实例。你可以查看 Microsoft Knowledge 里标题为《如何在 SQL Server 实例之间迁移注册程序和密码》这篇文章。这篇文章包括你所需要的脚本。

注意：如果你采用这种方法，你就需要重测数据库注册程序。你可以 sp_change_users_login 存储程序来完成。

(作者：Adam Machanic 译者：April 来源：TT 中国)

如何在 Excel 中显示 SQL Server 数据库报告？

问：我们现在使用的是 SQL Server 2000 企业版。我们想发掘一份 SQL Server 数据库报告并将它展示在 Excel 中。使用查询就会计算出所有数据，并将这些数据置入具体的 Excel 表格中便于计算和绘图。我们能不能用 VB 呢？这是不是还涉及到 SQL 2000 提供的其他服务？

答：你可以用 DTS 和 SQL Server 2000 直接将数据迁移到 Excel 表中，因为 Excel 表在你的 data pump 中就是目的地。你应该去查看 Microsoft knowledge 里的这篇文章：《如何用 SQL Server 数据转换服务将数据迁移到 Excel 表》。在这篇文章里你能找到更加详细的答案。

(作者：Joe Toscano 译者：April 来源：TT 中国)

接收 data notification：SQL Server 触发器 VS.存储程序

问：我现在使用的是 SQL Server2000，我有一个产品有效期数据库域。我想提前 30 天知道每一种产品什么时候失效。那么我如何创建一个能够提醒我的触发器？

答：触发器在这种情况下中不会正常工作。你在行或者数据行上执行 INSERT/UPDATE/SELECT 命令时，一般只会用到更新了的数据。如果数据还没有更新，那所有的触发器就不会工作。你的这种情况中，需要 notification 代替。

我会开发一种存储程序来计算产品最近的有效期限。在该查询中，我会用到 DATEDIFF 函数查看是否有匹配的行。然后，我会创建一个 scheduled job，如果数值大于零，我会发送电子邮件给自己。

(作者：Roman Rehak 译者：April 来源：TT 中国)