11.4其它议题_SQLServer2005数据库开发详解Microsoft认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/562/2021_2022_114_E5_85_B6_E5_AE_83_c100_562951.htm 在本章中，我们约略解释了通过 .NET 程序语言编写 SQL Server 内五种对象的方式。然而，T-SQL 是 SQL Server 本机的语言，所以它有不可被替换性。我们一般用它进行数据维护时，可以将它广泛地分成两类行为：一是数据查询与维护，也就是直接引用 SELECT / INSERT / UPDATE / DELETE 等 T-SQL 语法；另一类为程序应用，也就是使用 WHILE、SET、创建游标(Cursor)等语法。而总的来说 .NET CLR 是提供程序应用的另一种选择，数据查询与维护依然是以 T-SQL 为主。数据方面的运行(Set Orient)依然以 T-SQL 为佳，在访问数据时，应当尽量发挥 T-SQL 的功能。尤其在 SQL Server 2005 大幅增加 T-SQL 语言的能力(请参考本书第四章)后，你应当先研究如何通过 T-SQL 完成需求，然后再用 .NET CLR 补其不足。.NET Framework 有数千个类、上万个属性、方法与事件，提供强大的程序编写与调试功能，不管是数据结构，如复杂的文本(例如通过 Regular Expression 访问与验证文本内容)与数值运算、数组、列表的访问，加解密运算、网络访问等等，这些都大幅延伸了执行在 SQL Server 内的对象的能力。且因为通过 .NET 语言编写的程序集是编译后的结果，比解析式的 T-SQL 对象更有效率。也由于 .NET 语言编写程序的特性，程序设计师很有可能习惯性地组织 T-SQL 语法后传给 SQL Server，虽然 .NET 编写的 SQL 对象是存在 SQL Server 2005 之内，但依然是一句句独立的 T-SQL 语法交由查询引擎(Query Engine)执行，查询引擎仍

旧要解析、确认、创建执行计划等等。相较之下，静态的 T-SQL 所编写的对象只有在第一次执行时需要创建执行计划，因而比较有效率。另外，若基础的数据维护，也就是单纯添加、修改和删除等操作，通过 .NET 写的对象再传进 SQL Server 查询引擎，等同多了一层调用，性能一定比直接调用执行 T-SQL 慢一点。也因此，单纯 T-SQL 写成的对象依然有存在的必要。综合以上所述我们可稍做归纳，适合以 T-SQL 编写的对象如下：l 通过游标更新记录，虽然更新记录用批处理更新更好，但若非要用游标更新特定的某些记录，因为 ADO.NET 不支持可更新数据的游标，因此只能用 T-SQL 的游标。l 单句或两三句的 T-SQL 语法，通过简单的参数传递来建立存储过程或用户自定义函数。l 通过T-SQL 建立的简单对象，用以防止前端程序直接访问基础数据表(Base table)。在 SQL Server 2000 以前的版本，我们也可通过 C/C 语言来编写系统扩展存储过程（extended stored procedures）。但 在SQL Server 2005 以后，通过 .NET CLR 来执行 .NET 编写的对象较上述的 C/C 的扩展存储过程更好。除了 .NET Framework 功能丰富，VB.NET/C# 等程序语言较好编写对象外，还有如下的原因：l 对对象执行的安全管理更好：在创建对象时，.NET CLR 的对象可以限制访问资源的权限，也就是通过 CREATE ASSEMBLY 语法，利用 WITH PERMISSION_SET = { SAFE | EXTERNAL_ACCESS | UNSAFE } 子句限制该组件的访问权限(Code Access Security)，默认是 Safe，代表该组件只能访问 SQL Server 内的资源。若设置成 External_Access，则代表该组件可以访问外部系统的资源，例如文件、网络、系统环境变量、注册表(registry)等等。若设为 Unsafe 则该组件可以访问

系统所有的资源，包含 unmanaged 程序代码(也就是 COM 对象或 Win 32 API)，由于没有任何限制，这是最不安全的设置，在 SQL Server 2005 的在线帮助中极力强调不要设置此种安全权限。而以往 C/C 所编写的扩展存储过程几乎没有可设的，自然也就不如 CLR 所提供的安全管理有效。l 稳定可信赖：CLR 执行环境对于错误处理、资源管理、程序管理、数据类型检查等都有加强，比纯粹用 C/C 编写的程序还要稳定。而 SQL Server 自身对于扩展存储过程所使用的资源并无法有效管理，这可能导致扩展存储过程编写得不好，却用完了 SQL Server 可用的资源。l 性能与弹性：SQL Server 2005 所集成的 .NET 执行环境比原有扩展存储过程调用 ODS(Open Data Services)更有效率与弹性。而且对于 SQL Server 2005 版本所新增的数据类型，如 XML、(n)varchar(max)、varbinary(max)等，ODS 并未支持，通过 .NET 提供的 System.Data.Sqlserver 程序界面才能有效访问这些数据类型。当然，通过 C/C 来编写扩展存储过程也并非一无是处，若不访问 SQL Server 内的数据，也不返回结果给用户端，单就执行一些商业逻辑程序代码，C/C 编译的本机程序代码性能一般来说应该较 .NET 的 managed 程序代码（中间语言）来得好，且 SQL Server 也不需要在 managed 环境与本机环境中切换。若只考虑就性能而不考虑可信赖度/稳定性/弹性/扩展性，以 C/C 编写扩展存储过程仍是可考虑的选项。SQL Server 2005 提供的 .NET 程序编写结构与中间层应用程序服务器的程序架构相同，这提升了应用系统架构设计上的弹性。T-SQL 可以调用 .NET 所编写的对象，反过来也成立，因此可以规划各自擅长的部分，彼此调用，组合使用。例如需要复杂的商业逻辑运算，集成各应用

系统，转译查询的数据结果成用户操作界面所需的中间数据等等，依然是应用程序服务器的工作。但数据正确性确认，或是在 SQL Server 执行商业逻辑可以减少大量数据在网络来回传递等工作，则可以考虑用 .NET 在 SQL Server 服务器端编写对象。整体来说，SQL Server 2005 融入 .NET 架构后，你有更大的空间考虑摆放商业逻辑的位置。基于性能、稳定、扩充性、安全、方便性等等，或许应该重新思考，定位应用系统整体架构的设计。