SQL 故障诊断与排错技术

Forrest Houlette 著

康 博 译

清华大学出版社

(京) 新登字 158 号
北京市版权局著作权合同登记号：01-2001-4313

内 容 简 介

SQL 语句实质上是说明性语言，所以很难排除它的故障。因为它没有交互式调试器，也无法使用已开发的错误对象来生成提示错误的信息，甚至无法准确获知哪一行代码导致了错误。因此，我们必须发挥自己的能动性才能建立排除 SQL 故障的环境。

本书主要介绍有效排除 SQL 查询语句故障所需要的全部知识，介绍如何建立 SQL 调试环境，并提供了最好的编程技能练习。本书还引导读者使用常用的查询类型，同时提供调试查询的具体内容。

本书适用于熟悉 SQL 知识并希望学习调试 SQL 语句的所有读者。

书　　名：SQL 故障诊断与排错技术
作　　者：Forrest Houlette 著　康博　译
出 版 者：清华大学出版社(北京清华大学学研大厦，邮编 100084)
http://www.tup.tsinghua.edu.cn
责任编辑：杨海儿
印 刷 者：北京昌平环球印刷厂
发 行 者：新华书店总店北京发行所
开　　本：787×1092 1/16　**印张**：15.75　**字数**：373 千字
版　　次：2002 年 4 月第 1 版　2002 年 4 月第 1 次印刷
书　　号：ISBN 7-302-05293-X/TP・3112
印　　数：0001～5000
定　　价：26.00 元

前　言

本书介绍如何排除 SQL 故障的知识。SQL 是一种特殊的编程语言，因为它是说明语言，而不是过程语言。SQL 不要求编写代码，以执行从数据库中检索记录并处理它们的任务，而是允许用户输入类似于 SELECT*FROM MyTable 这样的语句。查询处理器负责处理获取数据的过程。用户甚至不必关心这个过程。

因为 SQL 语句在本质上是说明性语言，所以很难排除它的故障。它没有交互式调试器，也无法使用开发好的错误对象来生成关于错误的信息，甚至无法准确获知哪一行代码导致了错误。

因此，我们必须发挥聪明才智，才能建立排除 SQL 故障的环境。这就是本书的主题。

本书内容

我的父亲喜欢说，我所使用的 Franklin ACE 1000 的内存比他过去使用的大型机的内存还要大。他的故事类似于“当我还是个孩子时”的故事，说在大雪纷飞的日子里，要步行十英里到学校去。这种故事很有教育意义。只要开始使用 SQL，您就拥有了调试 SQL 程序的技能，这种技能同我父亲步行十英里，冒着大雪去为他的大型机取回打孔卡那个编程时代所具有的调试技能相同。也就是说，必须将程序看作是黑箱，在黑箱中输入 SQL 语句，却几乎得不到什么输出结果。我们不得不从很少的输出信息中推断程序内部发生的问题。

SQL 语言可以定义数据库和操纵数据。它已经进行了标准化处理，因此可以在与 SQL 兼容的数据库上执行语句块。SQL 标准本身不提供调试工具。我们必须发挥创造性，构建调试环境。应该把这看作是一次机会，而不是一个挫折。

如果读者在编写大型机程序时使用过打孔卡，就具备排除 SQL 查询故障的调试技能。但有一些读者还不具备这种技能，他们希望得到这方面的指导，获得必要的技能。本书将指导读者学习这种调试技能。我说这些并不是在吹嘘自己曾经使用过打孔卡，而是提出自己的调试建议，不是空洞地谈论步行十英里，冒着大雪去上学这些老掉牙的故事。（我在加利福尼亚州南部长大）。

本书介绍了在有效排除 SQL 查询语句故障时所需要掌握的全部知识，介绍了如何建立 SQL 调试环境，并提供了最好的编程技能练习。本书还引导读者使用常用的查询类型，同时提供调试查询的内容。

本书的读者对象

本书假定读者已经学习过 SQL。读者应该熟悉如何使用 SELECT，INSERT，UPDATE 和 DELETE 编写查询的知识，能正确区分 FROM 短语和 WHERE 子句，还能区分 HAVING 子句和 WHERE 子句的作用。如果读者具备这些知识，就可以学习本书。如果不具备这

些知识，可以阅读《SQL 入门指南》（Osborne/McGraw-Hill，2000），因为在阅读如何使用具体的 SQL 语句的章节之前，您还需要复习 SQL 编程的基础知识。

本书适用于希望提升使用 SQL 技能的 SQL 程序员。SQL 程序员往往在调试 SQL 语句上花费大量的时间。最近，我调试了一个给出了错误数据的报表。我运行了产品报表副本，并在产品数据库上对此报表运行了一个查询语句。比较其结果，我发现一个问题：报表未能输出全部数据。我又对测试服务器运行了查询。在此环境中报表输出了全部数据。这花费了我大半天的时间确认问题所在，如果您能迅速找出问题，就不必再阅读本书了。

但是，您可能和大多数人一样，无法迅速找出问题。要知道我花了大半天时间比较在报表模板中的设置和在测试和产品环境中的查询输出结果，没有发现产品报表输出的、从产品查询返回的数据有什么改变。在快要结束时我才找出了问题。产品报表输出的是测试数据。本书介绍的就是这类经验，因此读者就不必像我一样花费一天的时间来调试同样的问题。而且，到目前为止，我还没有遇到一位不需要这种经验的 SQL 程序员。

本书的学习方法

为了从本书受益，您需要从第 1 章开始完整地学习每章内容。由于本书介绍的是个人调试经验，所以不应把本书当作百科全书式的参考资料。本书的主要目的不是提供基本的 SQL 语法，虽然本书也有这些内容。本书介绍了许多 SELECT 语句的内容，比如编写 SELECT 语句的经验。如果您从头开始阅读本书，就可以更好地受益。

在学习本书的过程中，需要标记一些章节。在如何编写 SELECT 语句方面，本书提供的编程方式可以解决频繁出现的编程问题。在这些页上作出记号，以便再次找到这些范例。读者应创建自己的索引，为此，要标记包含最常遇到的问题页。当然，也可以标记已经解决的问题。这样本书才能成为 SQL 编程的指南教材。

章节的划分

本书分为三部分。第 I 部分介绍如何构建功能性的开发环境，使用此环境可以有效地排除 SQL 查询的故障。第 II 部分介绍基本的 SQL 语句，复习其语法和用法，还讨论了每一种语句的组合用法。第III部分介绍更复杂的问题，还给出了解决用于主要 SQL 数据库的、针对具体 SQL 开发平台的材料。

各章的具体内容如下：

- 第 1 章“选择排除故障的环境”，学习可以使用的调试环境，以及如何选择合适的调试环境。
- 第 2 章“使用良好规范”，详细介绍 SQL 编程范例以及如何使用它们。
- 第 3 章“准备排除故障的环境”，指导如何创建有效的调试环境。
- 第 4 章“创建数据库”，接着前面三章的内容，介绍创建数据库的最佳进程，因为需要解决的许多问题实际上都源自设计，而不是编程过程。
- 第 5 章“表的规范化”，指导设计用于 SQL 查询的表。本章主要关注 SQL 程序员如何选择表的设计方式，以避免问题的产生。

- 第 6 章“使用数据类型”，学习与数据类型和在 SQL 数据库中实现数据类型时相关的问题。
- 第 7 章“选择数据”，这是使用 SELECT 语句的基础知识，本章还介绍了复合 SELECT 语句的一般形式。
- 第 8 章“插入数据”，介绍 INSERT 语句，解决在使用 INSERT 过程中出现的常见问题。
- 第 9 章“更新数据”，介绍 UPDATE 语句的知识，并介绍了 UPDATE 语句可能导致的常见问题。
- 第 10 章“删除数据”，介绍使用 DELETE 语句的知识，DELETE 可能是 SQL 中最危险的语句。本章介绍如何避免与删除数据操作相关的问题。
- 第 11 章“分组和聚集数据”，介绍聚集函数和 HAVING 语句的用法。
- 第 12 章“使用连接”，详细介绍创建连接和连接的常见用法。
- 第 13 章“使用子查询”，指导如何在查询中编写查询，以及如何解决创建子查询所带来的性能问题。
- 第 14 章“使用视图”，深入介绍虚拟表的用法，虚拟表也常称为视图。本章讨论最适合于使用此工具的常见情况。
- 第 15 章“触发器、存储过程和参数”，讨论如何在数据库服务器上创建过程，这可能是 SQL 程序员最常见的任务。本章介绍如何用 SQL 使数据库程序具备最佳性能。
- 第 16 章“事务处理”，分析事务，事务是维护数据一致性的一种手段。在本章将学习如何在优化的查询中创建和使用事务。
- 第 17 章“使用游标和异常”，介绍在任何数据库程序中与速度响应最相关的主题。游标很慢，但是本章将介绍如何有效地使用游标，如何激活异常，监控游标的工作。
- 第 18 章“树形结构”，介绍如何使用数据库中给出的树数据结构。本章介绍开发高级 SQL 的知识，并学习如何有效地使用常见的数据结构。

作者的联系方式

如果您对本书有什么意见和建议，请与作者联系。作者的电子邮箱是：forrestw@bellsouth.net。如果希望了解作者的其他著作，可以访问作者的网址：http://www.writeenvironment.com。该网站上有作者的个人照片。

您会发现作者非常健谈。他很乐意倾听您的意见。

目　录

第Ⅰ部分　基　础

第Ⅱ部分　分 析 问 题

第III部分　解决复杂的问题

第Ⅰ部分　基　　础

第1章　选择排除故障的环境

本章主要内容：

- Query Analyzers
- 图形工具
- 开发环境
- 实际应用

要排除 Structured Query Language(SQL，结构化查询语言)的故障，必须承认自己还处于很低的起点。SQL 是数据库设计人员使用的语言。IBM 发明了 SQL，但是第一个 SQL 商业版本却是在 Oracle 的数据库中产生的。Oracle 通过第一次实现 SQL 在数据库市场上击败了 IBM。

关系数据库理论提出，必须有一种定义数据特征的方式，这种方式现在通称为数据定义语言(DDL)。DDL 可以描述数据类型，并将数据组织到表中命名的列里。还必须有一种操纵数据的方式，这种方式现在通称为数据操纵语言(DML)。DML 可以检索数据，对数据排序，并归纳数据。IBM 创建 SQL 时，数据库运行在大型机上，并由精通数据库管理的专业人士维护。

这些专业数据库维护人员并不要求 SQL 语言的功能超过当时推出的其他编程语言的功能。要知道在当时，数据库维护人员习惯于使用诸如 FORTRAN 和 COBOL 等编程语言。几乎所有这些程序员都使用过打孔卡来提交程序语句，在提交程序并以批处理方式运行之后，将会看到打印到绿条纸上的错误。这个开发小组所使用的错误报告信息由一个行号和一个错误提示组成。通常这种描述符仅仅是“语法错误”。程序员通常被认为了解语法，因此可以找出错误。

在某种意义上，SQL 的起源导致 SQL 不具备广泛的调试支持功能。而且，SQL 在本质上也不支持广泛的调试功能。在创建查询并提交查询以进行处理的机器上，通常不编译 SQL。在大多数情况下，查询必须提交给数据库服务器进行编译，即由安装了数据库软件的物理计算机进行编译。最好在创建查询的客户机上安装语法检查器。这个工具可以给我们提供调试信息，比如把字符串值包含在引号中，但是语法检查器无法告诉我们是否可以编译查询。为了对 SQL 语句进行编译，必须执行 SQL Prepare 操作，该操作将 SQL 语句本身作为参数传递。在编写语句时，数据库客户机软件将语句输出到数据库服

务器上，在数据库服务器上的编译器就编译语句，使之可以运行。

编译器优化查询，这是整个进程的一部分。优化器还考虑了实际数据库服务器上处理器的个数，优化器准备查询执行计划，将查询分解为可以调度的单元，并将这些单元正确地分配给处理器。编译器可以执行大型任务，但不能提供广泛的交互调试支持功能，只能将编译结果返还给数据库客户机软件。

许多 SQL 工具隐藏了细节

许多 SQL 工具隐藏编译 SQL 语句的实现过程。通常我们仅仅是执行查询。Open Database Connectivity API 则相反，它提供 3 个函数：SQLPrepare，SQLExecute 和 SQLPrepareAndExecute。SQLPrepare 编译查询，SQLExecute 执行编译后的查询，SQLPrepareAndExecute 同时负责这两项任务。Gupta 的 SQLWindows 是一种用于 Windows 的 SQL 编程环境，提供所有这些选项。Microsoft 的 ActiveX Data Objects(ADO)没有提供这些选项。专用的数据库工具，比如 Microsoft 的 Query Analyzer 没有给出运行查询所需要执行的步骤，但可以使用各种连接 API 来连接数据库。记住数据库引擎不管采用什么步骤都可以编译 SQL 语句并执行它。一些 SQL 环境，比如 Microsoft Access，允许在客户端进行编译，而其他环境要求在服务器端进行编译。我们并不知道工具在后台所执行的步骤。编译之后再执行是 SQL 环境的规范，即使环境隐藏了实现的细节也是如此。

为了执行 SQL 语句，必须首先执行 SQL Prepare，然后把处理好的语句传递给 SQL Execute 操作。SQL Execute 等价于执行一个可执行文件。要在 MS-DOS 中调用这种服务，只需在命令提示中输入可执行文件的名称并按下 ENTER 键。在其他操作系统下，可以在命令提示中输入 Run，并按下 ENTER 键。SQL Execute 操作与此类似。

这种操作可以根据编译器制订的计划去执行查询。在执行期间也会遇到错误。致命错误会返回某种退出代码。但是逻辑错误不会提供任何错误信息。逻辑错误返回数据，但不是所期望的数据。当执行语句时，可以得到三种结果：期望的数据，不期望的数据或错误代码。在后两种情况下，调试器没有什么用处。要么知道出现一个致命错误，这种错误可能会使调试器崩溃，要么知道从数据库中请求了错误数据，这是一个老话题：期望数据库能识别你的意图，而不是在 SQL 语句中所编写的实际代码。

如果希望运行一个 SQL 语句，此语句具有符号调试器的全部优点，就需要在数据库服务器上运行语句，或者在一个联网的环境中运行，这个环境支持将调试信息实时传递回客户机的功能。这两种情况都不太可能发生。为了在数据库服务器上运行代码，必须突破各种安全限制。头脑正常的人不会允许以这种方式在产品数据库服务器上运行代码。首先，这会使服务器崩溃，这在排除故障时是可能发生的，对于需要全天候提供数据的人来说，服务器瘫痪是不允许的。第二，赋予某人在某个服务器(产品服务器或开发服务器)上具有该级别的控制权会带来安全隐患。要调试某个数据库，需要有管理权限，且允许拥有该控制级别的程序员请求排除安全故障。最后，开发工作可能会使服务器性能略有下降。在安全限制比较高的情况下，随着管理特权的提高，病毒会在网络上传播。

设计提示：

无论在什么情况下，都应该使用开发服务器开发、测试查询和过程，而不是在产品服务器上。常见的错误是对错误的数据库执行查询。作者当前的客户机在产品服务器上存储了 80 个不同的数据库，它们还可以在相同的服务器上对第 81 个数据库进行开发。虽然最好将开发数据库与产品数据库分离，但是在此客户机配置中，开发人员很容易错误地对产品数据库执行查询。最近，作者被要求清理测试数据库中的数据，方法是改变社会安全号码和地址。在最后期限作者完成了这个任务。在编写和执行查询之前，很容易忘记设置活动的数据库。想像一下对错误的数据库执行 UPDATE CriticalTable SET SSN='123456789'语句的结果！

创建一个支持 SQL 调试功能的网络的成本总是太高。对于开创者来说，在开始创建 SQL 时，实用的联网技术还不成熟。但是有了 Xerox 开发的技术(Xerox 并不希望把这些技术泄露给 Palo Alto Research Center)，公众可以使用联网技术了，但是网络却总是被通信信息所堵塞，需要排除故障。因此，开发 SQL 时并没有考虑这种调试支持功能。有了 GB 容量的 Ethernet(以太网)和最近快速发展的硬件设备，这种调试器在理论上是可行的。但 SQL 调试技术的模式却是在以前的技术限制下形成的。

因此，在创建 SQL 语句时，就可以使用所有这些调试支持功能，早在 1978 年编写第一个 FORTRAN 程序时就有这些支持功能。对于那些从事计算机工作时间没有这么长的人来说，可能使用过共享大型机。那时，Woz 还没有创建 Apple I。所编写的程序语句都驻留在打孔卡上，语句的长度不能超过 80 个字符。当测试程序时，需要将卡带放在计算机中心，由技术人员把卡带放到读卡器上。授予用户有限的访问权限，在大约三小时内得到输出结果。然后就需要仔细研究输出的结果，寻找错误根源。通常需要添加打印语句，以便随时在程序中显示变量值或希望得到的输出结果。然后重新提交程序，再等待三小时得到新的结果，这一次就可以得到显示的错误信息。

为什么在本书的开始描述这种可怕的情况呢？原因非常简单：大多数 SQL 数据库所提供的调试功能都不会多于这种情况下的调试功能。编写 SQL 语句时，需要开发我在 1978 年上高中时编写 FORTRAN 程序时所具备的所有技能。如果习惯于使用带有调试器的集成开发环境，其调试器可以遍历子例程和函数，并同时查看表达式的值，那么您在试图调试 SQL 时就会感到非常失望，因为此时没有这种调试支持功能。

很不幸这不是个好消息。但是我在本书却可以提供好消息。本书将介绍如何成功排除 SQL 故障的技能。无论是执行对数据库的单一查询，还是执行包含几个语句的扩展 SQL 过程，这种技能都可以胜任。为了有效排除 SQL 的故障，必须注意一些限制条件。这些限制条件决定了所使用的操作环境。如果无法实现监视功能，就不能在一个循环中监视变量值的变化。但可以避开 SQL 强加在操作环境上的所有限制条件。我们必须像以往的程序员那样思考，才能智胜所要处理的系统。

那么如何智胜 SQL？最简单的答案是“撒谎、欺骗和偷盗”。本书的主要内容就是解释这句话的重要性。现实一点的说法是仔细选择工具，并深刻理解该工具。

1.1 查询分析器

要创建 SQL 语句，需要一个编辑器。可以使用任何文本编辑器，如图 1-1 所示。可以使用 Notepad，Brief，Multi-Edit 或其他可用的文本编辑器，来创建在数据库上执行的语句。许多人都采用了这种方法。但是，使用纯文本编辑器将无法利用许多功能。

例如，如果使用 Notepad 或类似的编辑器来创建 SQL 语句，就无法在数据库上执行该编辑器中的语句。因此，必须编写某种程序来打开 Notepad 创建的文本文件，并在数据库上运行该文件。在这种情况下几乎没有什么调试功能。如果语句返回一个错误，顶多只能猜测其原因。比如可能有这样的错误，ActiveX Data Objects 记录集返回的记录数是-1。这个值表示“现在无法确定记录集中记录的准确个数”。如果花费大量时间研究 ActiveX Data Objects，很快就会发现记录集对象上的记录数属性通常都返回-1。此时我们就会怀疑，此对象是否可以计算其记录的个数。

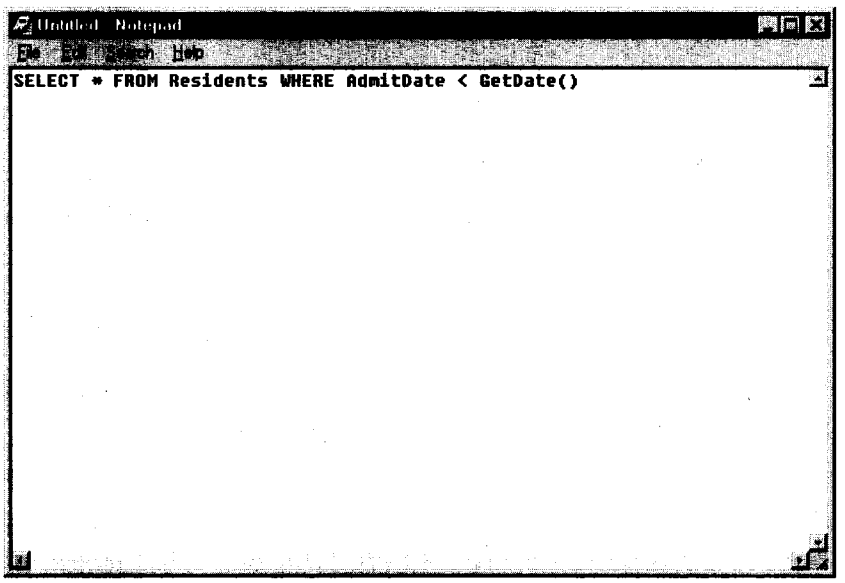

图 1-1　用于创建 SQL 语句的文本编辑器

如果曾经使用文本编辑器和程序开发 SQL 查询，就会知道使用这种开发方式很难调试 SQL。我们中意的编辑器应该可以在同一个窗口界面中输入查询、执行查询并返回结果。大多数数据库供应商都会提供这种产品。Microsoft 的产品是 Query Analyzer，其实所有类似产品都可以使用这个名称。这种编辑器通常会以彩色编码的方式来表示 SQL 语法。关键字是蓝色的，字符串是粉红色的，函数是灰色的，等等。彩色编码方法的主要优点是，可以知道何时使用了某种关键字，何时未使用关键字，如图 1-2 所示。在某些情况下，避免使用关键字非常重要。

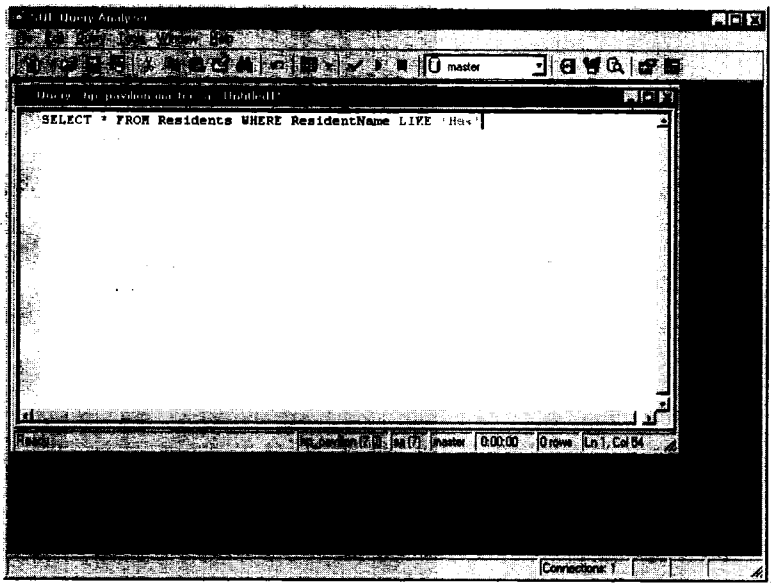

图 1-2　查询分析器中彩色编码方式的范例

设计提示：

SQL 和执行 SQL 的数据库常常可以使用各种不受推荐的方式。当看到警告时，要相信其他人已经试过了这种不可行的做法。仅仅因为可以在数据库中对 SQL 进行某种处理，并不表示就应该那样做。

例如，表的列名不能与 SQL 关键字相同。我们希望将记录的描述字段缩写为 DESC，但是这样做会导致更大的问题，因为 DESC 是表示降序排列的 SQL 关键字。为了引用列名称，必须重新对工作区排序。在 SQL 的各种版本中，其解决办法也各不相同。一些版本要求将列名称放在方括号中，比如[DESC]。另外一些版本要求使用两个双引号界定列名称，比如“"DESC"”。许多数据库出于兼容性的考虑，会接受几种语法变体，但是查询只能在程序中执行。如果在数据库供应商的查询分析器中执行查询，就必须使用正确的语法。一种非常有用的作法是，通过工具将语法问题用彩色代码表示出来。

设计提示：

有时 SQL 语法的具体实现在给定的数据库中也有变化。在 Query Analyzer 或存储过程构建实用工具中执行查询时，Microsoft 的 SQL Server 7.0 把分号作为 SQL 语句的结束符。但是，视图构建实用工具不把分号作为结束符。

查询分析器是非常有用的工具。这个工具融合了文本编辑器上的彩色编码功能和向数据库提交查询以便处理的功能。单击 Run 按钮就可以执行 SQL 语句。这种工具可以运行成组或成批的语句，甚至可以按顺序运行多批的语句。由于这些工具具有很大的灵活性，因此可以使用它们构建非常复杂的 SQL 程序。

除了可以构建和执行查询之外，还可以获取查询状态的信息。例如，这些工具提供