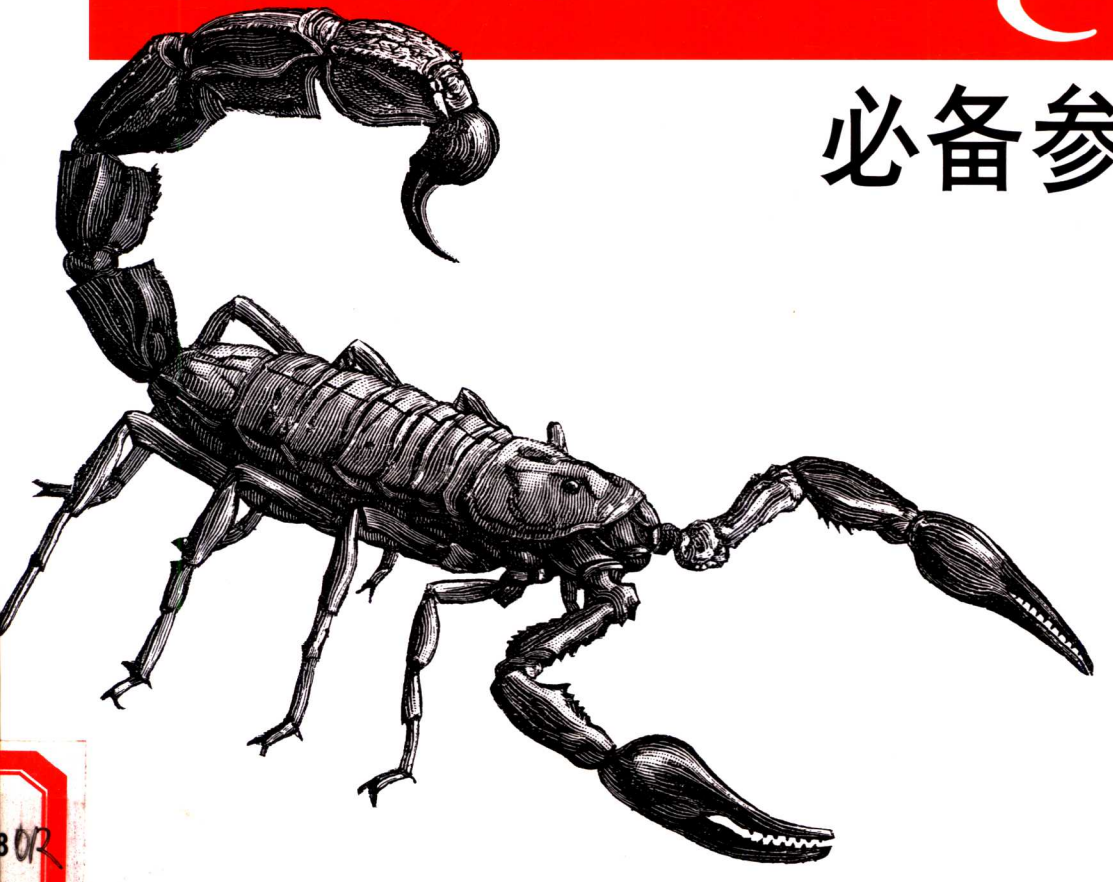
Oracle SQL: The Essential Reference

涵盖 Oracle8 & Oracle8i

Oracle SQL

必备参考



David C. Kreines 著
Ken Jacobs 序
吴安青 薛涛 卫红权 译

O'REILLY®
中国电力出版社

Oracle SQL必备参考

David C. Kreines 著
Ken Jacobs 序
吴安青 薛涛 卫红权 译

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Paris • Sebastopol • Taipei • Tokyo

O'Reilly & Associates, Inc. 授权中国电力出版社出版

中国电力出版社

图书在版编目（CIP）数据

Oracle SQL 必备参考 /（美）克雷尼斯（Kreines, D. C.）著；吴安青，薛涛，卫红权译．－北京：中国电力出版社，2002.11

书名原文：Oracle SQL: The Essential Reference

ISBN 7-5083-1102-7

I. O... II. ①克 ... ②吴 ... ③薛 ... ④卫 ... III. 关系数据库－数据库管理系统，SQL IV. TP311.138

中国版本图书馆 CIP 数据核字（2002）第 084654 号

北京市版权局著作权合同登记

图字：01-2002-1198 号

书　　名 / Oracle SQL 必备参考

书　　号 / ISBN 7-5083-1102-7

责任编辑 / 陈维宁

封面设计 / Ellie Volckhausen，张健

出版发行 / 中国电力出版社（www.infopower.com.cn）

地　　址 / 北京三里河路 6 号（邮政编码 100044）

经　　销 / 全国新华书店

印　　刷 / 北京市地矿印刷厂

开　　本 / 787 毫米 × 1092 毫米　16 开本　27 印张　395 千字

版　　次 / 2003 年 4 月第一版　2003 年 4 月第一次印刷

印　　数 / 0001-5000 册

定　　价 / 49.00 元（册）

O'Reilly & Associates 公司介绍

为了满足读者对网络和软件技术知识的迫切需求，世界著名计算机图书出版机构O'Reilly & Associates公司授权中国电力出版社，翻译出版一批该公司久负盛名的英文经典技术专著。

O'Reilly & Associates 公司是世界上在 UNIX、X、Internet 和其他开放系统图书领域具有领导地位的出版公司，同时是联机出版的先锋。

从最畅销的《The Whole Internet User's Guide & Catalog》（被纽约公共图书馆评为二十世纪最重要的50本书之一）到GNN（最早的Internet门户和商业网站），再到 WebSite（第一个桌面PC的Web服务器软件），O'Reilly & Associates 一直处于Internet发展的最前沿。

许多书店的反馈表明，O'Reilly & Associates是最稳定的计算机图书出版商——每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly & Associates公司具有深厚的计算机专业背景，这使得O'Reilly & Associates形成了一个非常不同于其他出版商的出版方针。O'Reilly & Associates所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly & Associates还有许多固定的作者群体——他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly & Associates依靠他们及时地推出图书。因为O'Reilly & Associates紧密地与计算机业界联系着，所以O'Reilly & Associates知道市场上真正需要什么图书。

作者简介

David C. Kreines 是 Rhodia 公司（Rhone-Poulenc S.A 的辅助机构）的数据库业务主管，而且是《Oracle Database Administration:The Essential Reference》（O'Reilly & Associates 于 1999 年出版）一书和《Oracle Scripts》一书（O'Reilly & Associates 于 1998 年出版）的作者之一。从 1985 年开始，Dave 已经开始以开发者和数据库管理员的身份与 Oracle 打交道，从 PC 到大型机的大量不同类型的平台都有所涉及。他通过了 Oracle 的 OCP DBA 认证，同时还经常参与美国和欧洲的 Oracle 会议、用户组织和刊物出版。Dave 已经担任了两届国际 Oracle 用户组织美国组（International Oracle Users Group-Americas，IOUG-A）的主席，并在委员会中呆了十年。

封面介绍

本书封面上的动物是一只蝎子。化石记录显示，蝎子是第一代节肢动物之一，而且在一些化石标本中的有腮动物可以显示蝎子是从海洋中的生物祖先进化而来。当今的蝎子是陆地居住生物，生活在岩石或沙地下，在夜间出动捕食昆虫。科学家掌握的大约有 1300 多种蝎子，大小从 1 英寸到 8 英寸不等，而且在颜色上也各不相同，有棕黄色，也有绿色和黑色。

蝎子通过脚上的感应触发器检测空气中的变动来定位猎物，利用尾部毒针散发出的麻醉毒液注射到昆虫身上，从而进行捕食。蝎子也使用毒针对敌人进行防御，包括黄蜂、蜈蚣和蜘蛛，也包括蜥蜴、鸟类和小的哺乳动物。

蝎子交配的礼节包括精细的求爱舞蹈，此时雌蝎子与雄蝎子通过脚进行交互，雄蝎子将精囊分泌在岩石或树枝上，雌蝎子在经过时吸取精子。对一些种类的蝎子而言，怀孕期可以持续一年半载，然后雌蝎子孵化出很多年轻的生命，这些小蝎子将在母体的背上度过一到两周的时间。

蝎子产生的毒液对人类来说是致命的。不要在夜间在荒凉的地方赤脚行走，或者，你需要在前面点一盏夜灯 —— 蝎子会在紫外线下发出荧光。

目录

序 1

前言 17

第一章 SQL 元素 25

词汇习惯 26

SQL 命名 28

架构对象 28

数据类型 31

数据转换 38

关系操作符 39

SQL 语句的结构 45

SQL 语句 49

第二章 数据定义语句 57

按任务列出的 SQL DLL 语句 57

SQL 语句的语法 64

第三章 数据操纵和控制语句 139
根据任务说明SQL DML和控制语句 139
SQL语句语法 140

第四章 公共SQL元素 167

第五章 SQL函数 179
聚集函数 180
数字函数 187
字符函数 197
日期函数 211
转换函数 220
其他函数 229

第六章 SQL*Plus 244
命令行语法 244
SQL*Plus编辑命令 248
规定SQL*Plus输出的格式 252
各种各样的SQL*Plus命令 263
SQL*Plus变量和相关命令 281
SQL*Plus系统变量 285

第七章 PL/SQL 307
PL/SQL结构 308
块头 310
声明部分 310
执行部分 326
异常部分 346
过程和包 351
触发器 367

第八章 SQL 语句调试 372
使用 EXPLAIN PLAN 373
使用 Oracle 的 SQL 跟踪工具 383
SQL*Plus 调试助手 394
改进查询性能 400

附录 SQL 资源 411

词汇表 415

序

SQL：庄严的历史和重要的将来

SQL语言是数据库管理的事实标准。对开发者或数据库管理员来说，熟练掌握SQL、编程语言知识及应用程序业务需要的知识一样重要。本书可以成为成功发挥Oracle8*i* 中 SQL 实现的必不可少的指南。

SQL具有漫长而古老的历史，它不仅是当今电子商务IT系统中的重要角色，而且具有美好的未来。本序中描述了SQL的起源、演变和未来，在你阅读这本优秀语言参考时，希望它能帮助你加深对 SQL 优越性的认识。

编程和数据访问语言

第二次世界大战时期，为了军事应用，人们开发了多用途的可编程计算机。1951年，产生了第一台商用多用途计算机 UNIVAC I。从那时到现在，人们已经开发了许多代编程语言。每一代编程语言都通过自动机械任务提高了程序员的生产力，允许程序员集中精力研究与应用程序相关的高层概念。

最早的程序通常都是采用机器码编写的，这些机器码数字对应于程序员希望存储在机器内存中的指令。允许程序员使用名字代替指令数字和内存位置的汇编语言开发于20世纪50年代早期。高级编程语言的开发标志着程序员工作所在的语义层有了

重大突破。这样的编程语言相继发明出来，从Fortran（1957）到C（1972）以及到Java（1995），都先后经历了成长期和衰退期。除Fortran、C和Java之外，Algol、COBOL、Ada、C++和Basic也都是我们已用于开发应用程序和系统程序的一些重要编程语言。

与如此众多的过程化编程语言相反，目前只有一种应用十分广泛的数据访问语言：SQL。SQL是非过程化数据访问语言，因为它让数据库管理系统负责决定怎样处理数据的查询问题。应用程序的程序员不必关心产生理想结果的数据访问路径和处理步骤。与底层机器代码和汇编语言相比，因为采用高级过程化编程语言更容易编写应用程序，所以SQL语言使得访问应用程序数据更加容易，尤其是还可以采用交互方式进行。SQL允许应用程序开发者全神贯注于业务逻辑，而不必关心使用索引或指针链提取或更新数据的问题。

虽然SQL在20世纪70年代中后期开发，并仍然在不断进化发展，但似乎不可能被其他语言取代。与过程化编程语言不同（对于过程化编程语言，主要的语言几乎不会被完全废止），当今大部分数据库管理系统都实现了SQL的某种方言。Rosetta石头（译注1）上包含的数据揭示了古埃及语言和象形文字的奥秘，因而可以引导大家更好地理解古埃及的历史和文化。当今，许多人则把SQL当作揭示企业数据库中数据值和信息的语言。

不同厂家的实现产品中有多种方言，但大多数SQL语言在当今市场上大部分商业数据库管理系统中是相同的。可以确信，存在其他数据访问语言，其中一些在大学内得到进一步开发，而另一些则在商业产品中得以实现，但是没有一种数据访问语言有SQL如此成功或者实现如此广泛。加利福尼亚大学伯克莱分校的Michael Stonebreaker教授曾经将SQL称为“银河系的数据语言”。

SQL的起源

那么，SQL是怎样产生、进化并发展的呢？20世纪70年代，SQL的故事开始于美国加利福尼亚州圣何塞旁边的IBM研究实验室。在IBM工作的数学家和研究员Ted Codd创造了数据管理的正式理论，并写了有重大影响的论文，名为“A Relational Model of Data for Large Shared Data Banks”，1970年6月发表在《Communications

译注1：古埃及文物。

of the ACM》上。他定义了关系数据模型，该模型由数据结构（行和列组成的表）、对数据的操作（如选择、投影和连接）、确保数据连续性的完整性规则（例如主键和参照完整性）等组成。

Codd对关系模型的严格数学定义可以定义保持数据完整性和最小化冗余的数据库设计过程。所谓的规范化理论定义了第三范式，这里，数据库中的每张表都有可以惟一定义表中每一行记录的主键，而且行中的每一列都依赖于主键。特别是，用第三范式设计的数据库能支持在数据库设计时不能预期的应用程序和查询。

Codd还定义了数学数据操纵语言——DSL/Alpha。该语言是基于数学集合论建立的，而且可以用于快速查询并操纵组成关系数据库的数据表。Codd证明了可以使用自己定义的操作在任何查询方式下操纵关系数据库，所以可以得到任何组成数据库的结果。他称这种性质为关系完备性（relational completeness）。

与传统上非关系数据库中编写用复杂指针链定位的程序相比，Codd定义的语言功能非常强大。与占用几页的程序相比，Codd的语言可以在少数几行内查询如“找到比经理干得更多的职员”这样的问题。

20世纪70年代早期，基于Codd的思想，以IBM研究分队的形式形成的小组开发了关系数据库管理系统原型。由Frank King领导的名为System R的工程启动了。该工程的目标是开发一个完全支持SQL的关系数据库原型，但仍然传送现存非关系数据库的关键属性，包括多用户支持、事务、安全性和良好的性能。

System R小组认识到，Codd的数学DSL/Alpha语言对没有数学基础的人来说太难理解，所以，他们创建了名为SQUARE的语言，代表以关系表达式形式指定的查询。尽管改进了DSL/Alpha，SQUARE还不适合于键盘输入，因为它需要当时还很难表示的下标。

该小组然后决定将SQUARE的思想改为基于英语关键字的方法，因为这样比较容易录入。他们扩展并改进了这一新语言，并称之为SEQUEL，代表结构化英语查询语言（Structured English Query Language）。因为商标问题，该名字随后被改为SQL。SQL的一般发音为“sequel”，有时也被发音为“ess-queell”，它们都很常见。1974年，Don Chamberlin和Ray Boyce合写了题目为“SEQUEL:A Structured English Query Language”的论文，并发表在1974年5月的《the Proceedings of the May

1974 ACM SIGMOD Workshop on Data Description,Access,and Control》学报上。这是有关这门即将成为SQL的语言的最先广泛流传的论文。

1995年，包括早期开发System R和SQL在内的小组成员为了25周年的纪念而重新团聚。他们回忆起这些人和该工程，而且提供了有关SQL开发的有价值的观点。他们的讨论在WWW上可以找到，URL为：*http://www.mcjones.org/System_R/SQL_Reunion_95/index.html*。

SQL语言

SQL（以及Codd最初的数据操纵语言）的主要功能是，它以非过程化形式加快了数据集合的操作，而不需要程序逐个提取记录并指定处理每条记录的步骤顺序。它不像大部分语言编写的程序那样指定了执行的特定步骤顺序，SQL语句表达了用户理想的结果，而且使数据库管理系统负责产生尽可能高效的结果。SQL语句指定了要在行集合上执行操作（如过滤、分组和排序），而且数据库系统决定了访问数据的精确方式，以及产生理想结果需要的各种处理步骤的顺序。SQL一个非常有用的方面是“闭包（closure）”性质——查询结果以表的形式产生。因此，查询返回的结果行集可以插入到另一张表中，或者用作SQL中查询表达式的一部分，如“子查询”或部分视图定义。

SQL原始定义的另一个重要元素是它包含了定义数据库内容的语法。数据库管理员定义了它的架构（schema）——其他地方叫做表名、列名及数据类型，这是使用所谓的数据定义语言（DDL，Data Definition Language）实现的，实际上它并不是一种独立语言，而是像CREATE、DROP和ALTER一类的SQL命令（动词）集合。SQL在这一方面与大部分SQL语言一样，是数据操纵语言（DML，Data Manipulation Language），该部分的SQL用于查询和更新数据库。DDL由动词SELECT、INSERT、UPDATE和DELETE以及其他SQL动词组成，如GRANT和REVOKE，这些语句都用于指定用户用以访问数据的权利。

很明显，在数据字典（data dictionary）表的行和列中，SQL指定了存储在数据库中用于描述数据库本身内容的元数据（meta-data）。数据字典（或者目录）表也可以使用SQL进行查询，所以可以动态书写应用程序并调整为适合于操作的数据库的外形和内容。

SQL的最初设计意图并非作为一个完整的编程语言。SQL的非过程化面向集合的能力是访问和操纵数据的理想途径，但应用程序的业务逻辑需要更传统的过程化语言。System R开发者创建了嵌入式SQL（Embedded SQL），这是一种允许应用程序员在主编程语言（如COBOL、Fortran和C）中使用SQL语句的“子语言”。SQL语句采用的前缀为“EXEC SQL”，可以嵌入在程序的源代码中，而且可以引用主程序语言中的变量（主变量）。预编译程序代替了嵌入式SQL语句，该预编译程序叫做DBMS特定编程库。

虽然SQL的许多方面遵从Codd关系理论的原始定义，但是为了性能、易用性或易于实现，也做了许多定义上的让步。例如，在Codd语言中，查询结果通常直接由行组成，因为在定义中“投影”操作消除了重复值。在SQL中，除非在查询的SELECT清单中出现了DISTINCT关键字，否则在查询返回的行集合中就可以出现重复值。

此外，作为一种计算机语言，SQL具有自身的怪异之处和缺点。理想的语言应该更直观、更有规则，而且语言元素上的一些限制可以出现在上下文中。SQL的一些批评家发现了SQL在处理过程中存在丢失信息的缺点（即丢失null值），并发现了SQL可以采用多种途径编写不同的语句返回相同查询结果。

Chris Date是致力于普及管理技术和SQL的作家和演说家，他是SQL语言最有声誉的批评家之一。事实上，Date和Codd在处理丢失数据的方法上意见迥然不同。但是，所有的批评和缺陷以及那些已经在用的数据库管理系统，都已证实了SQL无限的价值，而且已经大大成功了，这远远超出了其发明者的期待范围。

SQL在20世纪80年代的商业发展

1977年，Larry Ellison和另外两人建立了关系软件公司（Relational Software Incorporated，RSI），其明确目的是推动世界上第一个商业关系数据库管理系统的市场发展。他们受1970年Codd有关描述关系模型的论文和1974年描述SQL的论文的启发，决定从头开发尽可能与当时IBM研究中心开发的原型相兼容的临时商业产品。Ellison的想法是要在小型计算机上实现SQL系统，而且他正确预见到，除了关系数据库的新颖性之外，与IBM兼容性也会对市场有吸引力。实际上，他们在限制与System R兼容性的承诺是如此彻底，以致于Larry Ellison本人给IBM的Don Chamberlin打电话询问系统使用的错误号。早期ORACLE的演示版通常包括用于

演示IBM System R原型功能的“所得报酬过少的管理者”查询。ORACLE占用空间很小，而且与System R比较，所需资源较少，而System R不仅需要的空间很大，而且需要运行在水冷的大型计算机上。

1979年，RSI推出了第一个商用关系数据库——ORACLE。ORACLE的名字来源于Ellison和他的同事为美国政府工作的一个项目。ORACLE版本1是内部原型，所以商业发行的版本是ORACLE 2。在ORACLE 2中应用SQL是很完善的，因为它包括连接、子查询和视图，也包括处理层的惟一语言扩展，即CONNECT BY从句，在后一个主要的版本中添加了外部连接、日期/时间数据类型和大量嵌入式函数等创新。

系统的首批用户成功地展开了在简单部门中的应用，此时的系统主要是为了决策支持而不是为了满足严格代表事务处理的需求。ORACLE的许多早期用户也对关系模型的功能和使用SQL提供的简易性印象深刻，所以他们通常都忽略了早期ORACLE版本在可靠性上的缺点。RSI在1982年改名为Oracle公司，并开始快速增长，在随后的10年中每年以双倍的速度增长。1989年，Oracle在加利福尼亚的Redwood Shores建立了其总部园区。数据库领域的一个小趣闻是：离Oracle总部最近的机场位于San Carlos，而它竟然也使用了三字母代码——SQL。

自1979年Oracle中引入SQL以来，已经有了许多成功的SQL实现，而且在关系技术上的成功非常明显。令人惊奇的是，IBM从研究关系数据库管理及SQL开发中获益颇费了一些时日。尽管Codd的论文在1970年发表，并且1974年第一次描述了SQL语言，但IBM仍然用了多年时间才开发出其第一个投放市场的SQL产品。直到1981年，IBM才推出了应用于DOS/VSE和VM操作系统上的SQL/DS（使用了原始System R的许多原型代码）。1985年，IBM推出了运行在MVS大型机上的DB2，因为它非常小心地将其定位在需要主要决策支持的部门应用程序上，所以不会与其标志性层次系统IMS竞争。但因为IBM当时在IT行业居于统治地位，而且SQL也日益成为事实上的工业标准，所以这些消息的发布大大加速了SQL和关系系统的发展。

预见到关系数据库巨大潜能的并不只是IBM的研究者，而将该技术引向市场所能获得的巨大商机也并非只有Larry Ellison才知道。从20世纪70年代早期开始，加利福尼亚大学伯克利分校的Michael Stonebreaker教授和他的计算机科学专业的学生就开发了用于当时的Unix操作系统的INGRES关系数据库原型。伯克利小组是根据Codd的思想建立的，但是具有明确的竞争性，至少在INGRES小组和IBM研究所

之间存在学术竞争性。1980年，Stonebreaker建立了一个名为RTI（Relational Technology Incorporated）的公司，将INGRES推向市场。最后，RTI改名为Ingres有限公司。该公司最后被Ask公司收购，随后被Computer Associates收购，该公司目前专攻OpenIngres产品市场。

INGRES实现了称为QUEL的数据访问语言，该语言与SEQUEL相似。一些人认为QUEL是比SQL“更好”的一种语言，因为其武断性限制更少（更具“正交性”），而且具有SQL缺乏的一些功能。但无论其技术优点怎样，QUEL都没有SQL所具有的市场里程碑的意义，因为它被视为一种专有语言。直觉上，SQL可能会成为一种事实上的工业标准，可以可靠地应用于许多产品。结果，为了保留竞争性，大约在1986年，Ingres公司应用了SQL的一个语言子集，处于现存的QUEL接口层之上，但缺少一些主要功能，如空值和子查询等。INGRES的后继版本支持本地的SQL实现。

在关系数据库市场的早期，非关系数据库的坚定维护者视SQL和关系数据库如玩具，从来不会用于重要商业应用。SQL和关系系统的倡导者则赞扬他们的系统产品，而且声称可以克服理论上的性能障碍。

一些人认为，SQL的高层关系接口不能与被应用程序程序员调用的低层操纵接口相竞争。其他人则认为，关系表的物理存储组织和通过索引需要访问的数据值，不会像记录结构中嵌入的指针一样可以直接进行访问。System R开发者宣称SQL语句的自动编译和查询优化可以克服这些问题。当然，经过许多年，SQL系统在关系技术上的改进（随着硬件性能的显著改进）使其甚至适用于对事务处理要求最高的系统。关系数据库系统也能够利用SQL设置的面向对象特性，支持多CPU并行执行SQL语句，提供大数据仓库数据库上复杂查询的高度可升级性能。

在20世纪80年代，许多其他厂家提出了SQL系统。Relational Data Systems，就是后来的Informix公司，于1984年推出了带有SQL接口的同名数据库管理系统。在其他硬件厂家中，数字设备公司（Digital Equipment Corporation，DEC）于1985年发布了Rdb。Rdb应用的不是SQL，而是竞争的关系语言，叫做RDML。RDML在DEC客户中相当流行，但DEC并没有试图让其变得更流行，所以没有使其标准化。1988年，DEC推出了Rdb 5，承认需要遵守工业标准，具有完全内在的SQL应用特性。1994年，数字设备公司将Rdb卖给了Oracle公司，后者目前仍然在销售和支持该产品。

1985年，Teradata并行查询机的引入是SQL演变上的一个重要里程碑。Teradata系统使用了由Intel 8086处理器与私有树网络连接组成的特殊用途硬件平台，这是可以自动并行执行SQL语句的第一个商业数据库产品。但是，Teradata的SQL方言有很大局限，最初缺乏对视图和参照完整性的支持。Teradata系统被定位为面向数据仓库应用程序的查询处理需要，而且通常认为不能应用于事务处理系统。

从Ingres出来的核心人物Britton-Lee也设计并销售了“关系数据库机”，但该产品只取得了有限的市场成功，而且很快被Teradata收购，并最终从市场上消失了。继而NCR又收购了Teradata（随后NCR也被AT&T收购）。目前，NCR/Teradata已经放弃了采用特殊硬件的方法，现在都运行在使用Windows NT和Unix操作系统的多用途平台上。Teradata在数据仓库市场上已经非常成功，尤其在它们的数据仓库上已经具有大量追随者和大量数据。Teradata和Britton-Lee都发现很难与硬件和软件设计变革并驾齐驱，而且很难用需要特殊硬件的私有方法实现常用硬件系统。

另一个著名的里程碑是，1987年Tandem引入了不停止的SQL（NonStop SQL）。为了得到优秀的事务处理性能和高度实用性，NonStop SQL进行了优化。Tandem通过模拟简单银行事务的工作量支持其运行性能。这个基准的派生最终变成了事务处理性能理事会（Transaction Processing Performance Council，TPC）开发的第一个工业标准基准。NonStop SQL的引入打破了关系系统不能提供高端事务处理应用程序所需性能的神话。

Sybase是SQL市场上一个非常重要但相对姗姗来迟的迟到者，Sybase公司于1987年才推出了第一个版本的产品。微软购买了Sybase产品的源代码并随后于1993年推出了Windows NT下的SQL Server。

Sybase被设计为客户/服务器结构，应用程序运行在PC或工作站上，并通过网络访问数据库服务器。在并行操作下，我们看到了SQL语言和接口高层特性出人意料的好处。例如，仅通过调用一些网络消息，单条SQL语句就能迭代大量的行集，或者把表连接在一起。通常，采用低层导航式接口进行这样的操作会带来过多的网络流通量。

Sybase是第一个可编程的SQL数据库系统，而且也具有显著的市场影响。采用Sybase，DBA或应用程序开发者可以实现业务逻辑，并用触发器和Transact-SQL（Sybase公司专有的过程化语言）写的存储过程加强数据完整性规则。DBA和应用

程序开发者可以书写包含嵌入式SQL语句的程序提取，或更新数据库数据来执行完整的业务事务。触发器可以与数据库表相关，并可以在INSERT、UPDATE或DELETE操作之后执行，使事务生效、审核并执行其他转换。这一方法减少了网络流通量，因为整个业务事务可以用存储过程执行，并进行有效的网络调用。利用存储在数据库中并在数据库服务器上执行的存储过程，应用程序不需要与服务器上被访问的每一条记录进行通信，也不需要与业务事务中需要完成的每一条SQL语句进行通信。

可编程特性的另一个重要好处是，数据库服务器可以保护数据库的完整性，从而防止恶意的个别用户和应用程序通过网络对数据库进行攻击。虽然如参照完整性这样的基本关系型完整性的规则通常定义得很好，但是作为数据库架构的一部分，数据库触发器积极加强了需要过程定义的任意业务规则。通过数据库中的中心化业务逻辑，就不需要在访问数据库的每一个应用程序上编码，从而有效地避免了冗余和错误，并给终端用户直接访问数据提供了灵活性。

Oracle公司对自己的一部分产品使用了Ada编程语言作为PL/SQL模型，这是其专有的过程语言。像Ada一样，PL/SQL中包含了如异常处理和可以应用于开发的可用的、大型复杂系统的参数类型定义等语言特性。将过程语言添加到SQL标准中，从而最终组成了PL/SQL。PL/SQL最早出现在1988年的客户端中（在Oracle的SQL*Forms中），随后在1992年的Oracle7中使用了触发器和数据库中执行的存储过程。

SQL的进化：20世纪90年代和以后

如果20世纪70年代是SQL发明创造的十年，而20世纪80年代是SQL商业化的十年，那么20世纪90年代就是SQL进化的十年。在这一时期，SQL产品的各个厂家为了支持新的、要求高的应用程序而竞相在市场上推出新的特性。近年来，商业SQL产品和SQL标准都已被扩展，新特性支持面向对象的编程语言和多媒体数据、集成了Java和XML，而且满足数据仓库应用的需要。SQL已经成为一种充满活力的语言，并随市场的发展而不断发展。

20世纪90年代早期，面向对象的编程变成了商业应用程序开发的流行模式，因为程序员发现他们可以使用对象书写更快、更可靠的复杂应用程序。面向对象的语言