

Getting Started With

为9.7更新

DB2

快速入门的捷径

适合程序开发人员和数据库管理员



Raul F. Chong 著

Ian Hakes
Rav Ahuja 合著

Dr. Arvind Krishna 序

周其力 译

DB2 ON CAMPUS BOOK SERIES

DB2 Express-C

快 速 入 门

源于社区 贡献社区

RAUL CHONG, IAN HAKES, RAVAHUJA 著

DR. ARVIND KRISHNA 序



第 二 版

第三版（2009年6月）

第三次印刷（2010年10月）

本书适用于 Linux®、UNIX®、Windows® 上的 IBM® DB2® Express-C Version 9.7

© Copyright IBM Corporation, 2007, 2010. 版权所有.

目录

关于本书	10
声明	10
本书的读者对象	10
本书的架构	10
一本属于社区的书	11
本书的贡献者	12
致谢	12
翻译者	12
序	13
PART I – 概览和安装	15
第 1 章 – DB2 Express-C 是什么?	17
1.1 免费开发、部署和分发... 无限制!	17
1.2 下载 DB2 Express-C	18
1.3 用户帮助和技术支持	18
1.4 DB2 服务器	18
1.5 DB2 客户端和驱动	19
1.6 应用程序开发的自由性	20
1.7 DB2 版本号与 DB2 版本分类	21
1.8 升级到其他的 DB2 版本	21
1.9 DB2 Express-C 的维护和更新	21
1.10 相关免费软件和 DB2 组件	22
1.10.1 IBM 数据工作室 (Data Studio)	22
1.10.4 DB2 文字搜索	22
1.10.5 WebSphere Application Server – Community Edition	22
1.11 小结	23
第 2 章 – DB2 相关特性产品	25
2.1 DB2 Express 订购许可证 (FTL) 中包含的功能	27
2.1.1 Fix packs 补丁包	27
2.1.2 高可用性灾难恢复 (HADR)	27
2.1.3 数据复制 (Data Replication)	28
2.2 DB2 Express-C 所不具备的功能	28
2.2.1 数据库分区	28
2.2.2 连接集中器 (Connection Concentrator)	29
2.2.3 Geodetic Extender	29
2.2.4 基于标签的访问控制 (Label-based Access Control LBAC)	29
2.2.5 工作负载管理 (Workload Manager WLM)	30
2.2.7 SQL 兼容性	30
2.3 DB2 相关收费产品	31
2.3.1 DB2 连接 (DB2 Connect)	32
2.3.2 InfoSphere Federation Server	32
2.3.3 InfoSphere Replication Server	33
2.3.4 Optim Development Studio (ODS)	33
2.3.5 Optim Database Administrator (ODA)	33
2.4 DB2 的 Amazon Elastic 的云端计算	33
2.5 小结	34

6 DB2 Express-C 快速入门

Chapter 3 – DB2 installation.....	35
3.1 安装前提条件	35
3.2 操作系统中的安装权限.....	35
3.3 安装向导	35
3.4 验证您的安装	41
3.5 自动安装	42
3.6 小结.....	43
3.7 实验.....	43
第 4 章 – DB2 的应用环境.....	47
4.1 DB2 配置.....	55
4.1.1 环境变量	55
4.1.2 数据库管理器配置文件 (dbm cfg)	56
4.1.3 数据库配置文件 (db cfg)	57
4.1.4 DB2 概要文件注册表	58
4.2 DB2 管理服务器 (不建议)	59
4.3 小结.....	60
4.4 实验.....	60
第一部分 – 使用创建数据库向导创建一个新的数据库	60
第二部分 – 实例、数据库和配置管理	61
第五章 DB2 工具.....	65
5.3 命令编辑器 (不建议)	70
5.4 SQL 帮助向导 (不建议)	72
5.5 显示 SQL 按钮 (不建议)	74
5.6 任务中心 Task Center (不建议)	74
5.6.1 工具目录数据库 (不建议)	75
5.7 日志 (不建议)	76
5.8 运行状况监视器 (不建议)	78
5.8.1 运行状况中心 (不建议)	78
图 5.20 运行状况中心	79
5.10 脚本.....	80
5.10.1 SQL 脚本	80
5.10.1.1 执行 SQL 脚本	81
5.10.2 操作系统 (shell) 脚本.....	81
5.11 基于 Windows Vista 的考虑	82
5.12 小结.....	82
5.13 实验.....	83
PART II – DB2 Express-C 数据库管理.....	87
第 6 章 – DB2 体系结构	89
6.1 DB2 进程模型	89
6.2 DB2 内存模型	91
6.3 DB2 存储模型	92
6.3.1 数据页和扩展数据块	92
6.3.2 缓冲池	92
6.3.3 表空间	94
6.3.3.2 表空间的管理	95
6.3.3.3 数据是怎么存储在表空间的	96
6.4 小结.....	98

6.5 实验.....	98
第 7 章 – DB2 客户端的连接.....	101
7.1 DB2 目录	101
7.2 配置助手（Configuration Assistant ）	102
7.2.1 服务器端的安装要求	102
7.2.2 Setup required at the client 客户端的安装要求	104
7.2.3 建立客户端与服务器端概要文件	107
7.3 小结.....	110
7.4 实验.....	110
第 8 章 – 数据库对象.....	113
8.1 模式.....	113
8.2 公用同义词（或别名）	114
8.3.2 标识列	118
8.3.3 序列对象	119
8.3.4 系统目录表	119
8.3.5 已声明全局临时表（DGTTs）	120
第 9 章 – 数据迁移工具.....	129
9.1 导出（EXPORT）工具	130
9.2 导入（IMPORT）工具	130
9.3 使用 LOAD 来导入	131
9.4 db2move 工具.....	132
9.5 db2look 工具.....	133
9.6 小结.....	134
9.7 实验.....	135
第 10 章 – 数据库安全.....	139
10.1 认证.....	140
10.2 授权.....	140
10.2.1 特权.....	141
10.2.2 权限	141
10.2.3 任务	145
10.3 关于组特权.....	146
10.4 PUBLIC 组	146
10.5 GRANT 和 REVOKE 语句.....	146
10.6 查看授权和特权	147
10.7 在 Windows 上的扩展安全.....	148
10.8 小结.....	148
10.9 实验.....	148
第 11 章 – 备份和恢复.....	153
11.1 数据库的日志记录	153
11.2 日志的类型.....	154
11.3 日志记录的类型	154
11.3.1 循环日志记录	154
11.3.2 归档日志	155
11.4 从控制中心配置数据库日志	155
11.5 日志的参数.....	156
11.6 数据库备份.....	157

11.7 数据库恢复.....	158
11.7.1 恢复类型	158
11.7.2 数据库恢复	159
11.8 其他关于备份和恢复的操作	159
11.9 小结.....	159
11.10 实验.....	160
第 12 章 – 维护任务	163
12.1 重组（REORG）、运行统计（RUNSTATS）、重绑定（REBIND）	163
12.1.1 重组（REORG）命令	163
12.1.2 运行统计（RUNSTATS）命令	164
12.1.3 绑定 / 重新绑定	164
12.1.4 在控制中心执行维护工作.....	165
12.2 维护方式	166
12.3 小结.....	168
12.4 实验.....	168
第 13 章 – 并行与锁定.....	171
13.1 事务（Transactions）	171
13.2 并行（Concurrency）	171
13.3 无并行控制导致的问题	172
13.3.1 丢失更新（Lost update）	173
13.3.2 未落实的读（Uncommitted read）	173
13.3.3 不可重复读（Non-repeatable read）	174
13.3.4 幻象（Phantom read）	174
13.4 隔离级别（Isolation Levels）	175
13.4.1 未落实的读.....	175
13.4.2 游标稳定性.....	175
13.4.2.1 当前提交	176
13.4.3 读稳定性	176
13.4.4 可重复读	177
13.4.5 隔离级别对比	177
13.4.6 设定隔离级别	178
13.5 锁定升级	179
13.6 锁定监视	179
13.7 锁定等待	180
13.8 死锁的引发与侦测	181
13.9 并行与锁定的最佳实践：	182
13.10 小结.....	183
13.11 实验.....	183
PART III – 学习 DB2: 应用程序开发	189
第 14 章 –DB2 应用程序开发介绍.....	191
14.1 DB2 应用程序的开发: 概述.....	191
14.3 客户端的开发	193
14.4 XML 与 DB2 pureXML.....	203
14.6 管理 API.....	205
14.7 其他发展	205
14.8 开发工具	206
14.9 示例程序	206

第 15 章 – DB2 pureXML	207
15.1 在数据库中使用 XML	208
15.2 XML 数据库.....	208
15.2.1 启用 XML 的数据库.....	208
15.2.2 原生 XML 数据库	209
15.3 DB2 中的 XML	209
15.3.1 pureXML 技术优势	210
15.3.2 XPath 基础.....	212
15.3.3 XQuery 的定义.....	214
15.3.4 插入 XML 文档	215
15.3.5 查询 XML 数据	218
15.3.6 使用 SQL/XML 执行联合操作	223
15.3.7 使用 XQuery 执行联合操作.....	223
15.3.8 更新与删除操作	224
15.3.9 XML 索引.....	225
附录 A — 排除故障	231
A.1 查找错误代码的更多信息.....	231
A.2 SQLCODE 与 SQLSTATE	232
A.3 DB2 管理通知日志.....	232
A.4 db2diag.log.....	233
A.5 CLI 追踪.....	233
A.6 DB2 缺陷与补丁	233
附录 B — 参考文献和其他资源	235
B.1 参考文献	235
B.2 网站	235
B.3 书籍	236
B.4 联系我们	237

关于本书

声明

© Copyright IBM Corporation 2007, 2010

All Rights Reserved.

IBM Canada
8200 Warden Avenue
Markham, ON
L6G 1C7
Canada

未经上面提到的所有版权所有人事先同意，禁止以任何形式或任何手段对本文档或其中的任何部分进行复印或复制，或者翻译成其他语言。

IBM 对此处的内容不作任何保证或陈述，并明确声明免除任何关于适销性和适用于某种特定用途的暗示保证。**IBM** 对本文档中可能出现的错误不承担任何责任，包括（但不限于）翻译错误。本文档中包含的信息可随时更改而不另行通知。**IBM** 保留进行任何更改的权利，而不负责通知任何人这些修订或更改。**IBM** 不作出使此处包含的信息保持最新的承诺。

本文档中涉及非 **IBM** 产品的信息是从这些产品的供应商处获取的。**IBM** 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 **IBM** 产品的声明。有关非 **IBM** 产品性能的问题应当向这些产品的供应商提出。

IBM, **IBM** 徽标, **ibm.com** 均为 **International Business Machines Corporation** 在美国和/或其他国家或地区的商标或注册商标。其他产品和服务名称可能为 **IBM** 或其他公司的商标。**IBM** 商标列表可以在 www.ibm.com/legal/copytrade.shtml 查询。

Java 和所有基于 **Java** 的商标和徽标是 **Sun Microsystems, Inc.** 在美国和/其他国家或地区的商标。

Microsoft®、**Windows®** 和 **Windows** 徽标是 **Microsoft Corporation** 在美国和/其他国家或地区的商标。

Linux® 是 **Linus Torvalds** 在美国和其他国家或地区的注册商标。

Unix® 是 **Open Group** 在美国和其他国家或地区的注册商标。

其他公司、产品或服务名称可能是其他公司的商标或服务标记。

本出版物中所提到的 **IBM** 产品或服务并不暗示 **IBM** 将在所有 **IBM** 开展业务的国家或地区中提供它们。

本书的读者对象

本书面向于那些从事或准备从事数据库相关工作的读者，比如数据库管理员（DBA）、数据库相关的开发人员、咨询人员、软件架构设计人员、产品管理人员、决策人员以及学生。

本书的架构

Part I, 概览和安装。这一部分解释什么是 **DB2 Express-C**，介绍 **DB2** 产品家族的不同产品和各自的特性，并介绍 **DB2 Express-C** 的安装和数据库的创建，最后介绍 **DB2** 提供的各种工具。

Part II, 阐述 **DB2** 的数据库管理，使您掌握关于 **DB2** 的环境、体系结构、远程连接、数据对象、数据迁移（import/export/load）、数据库安全、备份和恢复、并行和锁定以及其它常见的维护任务。

Part III, 阐述 **DB2** 的应用开发。介绍 **DB2** 的应该程序开发包括服务器端和客户端 **DB2** 开发。同时讨论 **SQL/XML**、**XQuery** 以及 **pureXML®**。

附录提供排除故障的有用信息。

大多数章节提供了实验，随书提供了实验所需要的输入文件并打包成压缩文件 **expressc_book_exercises_9.7.zip**。

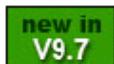
书中的内容也用作为 DB2 on Campus 项目的课程材料，并与www.channelDB2.com/oncampus上的学习视频相一致。您可以在 DB2 Express-C 的网站上获得更多关于 DB2 on Campus 的信息www.ibm.com/db2/express/students.html。

注意：

更多关于 DB2 on Campus 项目的信息，请参加下面视频：

<http://www.channeldb2.com/video/video/show?id=807741:Video:3902>

在此书最新的第三版中我们添加和改动了很多内容。为了方便读者已经阅读过第二版的读者们，我们使用下面的图标点明最新版 DB2 9.7 的更新。



一本属于社区的书

本书由 DB2 Express-C 团队创作，并免费向 DB2 Express-C 社区提供。此时此刻，本书下载次数已经超过了 85000 次并且由来自于全世界的自愿者翻译成九种不同的语言版本。如果您想提供反馈信息、贡献新的材料、改进现有材料、或者帮助翻译本书到其它语言，请将您的计划发送邮件到 db2x@ca.ibm.com 并署标题为 DB2 Express-C book changes。

此书的成功启发了超过二十五本关于 IBM 产品和非 IBM 科技的在线免费图书。此书将成为 DB2 校园图书系列的一部分，并于 2010 年 1 月发行。

更多关于本书或社区图书系列的信息，请访问 IBM DB2 Express-C 网站
www.ibm.com/db2/express

本书的贡献者

以下的人为本书做出了贡献。

Raul F. Chong – 领衔作者

Raul 是 IBM Toronto Lab 的 DB2 on Campus 项目经理。

Ian Hakes – 作者和编辑

Ian 曾任 DB2 Express-C 社区管理员，现任 IBM Toronto Lab 使用性能专家。

Rav S. Ahuja – 作者和出版

Rav 是 IBM Toronto Lab 的 DB2 产品资深经理。

Claire Hong – 审阅

Claire 是 IBM 中国软件开发中心的软件工程师。

Jing Jing Xiao – 审阅

Jing Jing 是 IBM 中国软件开发中心的软件工程师。

Guang Zhou Zhang – 审阅

Guang Zhou 是 IBM 中国软件开发中心的软件工程师。

Wei Wang – 审阅

Wei 是 IBM 中国软件开发中心的软件工程师。

致谢

我们真诚感谢下列的每个人，他们帮助和开发本书所引用的材料：

- 感谢来自 IBM 多伦多实验室的 Ted Wasserman, Clara Liu 和 Paul Yip 他们制作的材料成为本书的框架。
- 感谢 Don Chamberlin 和 Cindy Saracco 在 IBM developerWorks 关于 XQuery 的精彩文章，感谢 Matthias Nicola 关于 pureXML™的阐述。
- 感谢 Kevin Czap 和 Grant Hutchison 制作了 DB2 技术简报材料。
- 感谢 Katherine Boyachok 为本书设计了封面。
- 感谢 Susan Visser 对本书发布的帮助。

翻译者

我们要感谢来自社区的每位翻译志愿者，他们耗费了无数个日夜和周末，完成了不可思议的工作，使得本书成为现实。下表列出了每一位志愿贡献者。

贡献者的名字	所在公司/学校	职位名称	所作工作
周其力	杭州电子科技大学	教师	计算机专业

序

创新是技术进步的基石。在 **IBM**, 创新是数据库服务器发展中极之重要的组成部分。早在上世纪六七十年代, 我们作为这个领域的先锋, 开拓了先进的数据管理技术, 一直到现在, 我们还一如既往的保持这种创新的精神, 不断地在信息管理领域进行技术革新, 单是数千个属于 **IBM** 技术专家的数据管理专利就很好的印证了这一点。经过我们不懈的努力, 世界上的不少大型组织都依赖 **IBM** 的产品, 比如 **DB2**, 为他们提供满足最大需求的源动力和胜任严峻挑战的数据管理解决方案。

现在, **DB2** 不再只是针对大型企业。随着 **DB2 Express-C** 的发布, 中小企业不需支付任何费用就能够享用 **DB2** 技术带来的丰厚回报。尽管当前还存在其它的免费或者开源数据库服务器, 但 **DB2 Express-C** 所提供的独特优势使它成为最好的选择。

DB2 Express-C 为我们展现了很多技术进展, 这些创新赋予了 **DB2** 新的能力, 减轻了管理数据库的负担, 改善了数据库运行的性能, 减少了基础设施建设的费用。

DB2 Express-C 的混合科技有能力管理原始格式管理关系数据和 **XML** 数据的复合数据库。这使得 **DB2** 成为 **SOA** 和 **Web 2.0** 应用的理想选择, 因为这些应用往往存在大量的 **XML** 数据流动。**DB2 Express-C** 不像其它的一些免费数据库, 它不限制您存储在数据库中数据量的大小, 也不限制您创建数据库数量的多少。而且, 如果您需要 **IBM** 的帮助, 那也会像点一下鼠标那样简单。

本书作为使用 **DB2 Express-C** 的起步学习和使用指南, 能够很好地帮助您理解 **DB2** 的概念, 并提高您管理 **DB2** 以及使用 **DB2** 进行应用开发的能力。本书所介绍的技巧和知识与 **Linux**、**Unix**、**Windows** 中高级版本 **DB2** 所需的知识和技巧是非常相近或一致的。

尽管 **DB2 Express-C** 不是一个开源的产品, 在 **IBM**, 我们深信支持和推动社区的首创精神。我很高兴能看到一本由 **DB2 Express-C** 社区成员创作的书, 并且免费向社区所有成员提供。我非常感激您们, 您们用所具有的知识和经验丰富充实本书, 并将本书翻译成多种语言, 使得更多的人能够获益。



Arvind Krishna
Vice President, Data Servers
Information Management, IBM Software Group

PART I – 概览和安装

1

第 1 章 – DB2 Express-C 是什么？

DB2 Express-C 数据服务器软件（“DB2 Express-C”）是 IBM DB2 系列产品的成员之一，DB2 是管理关系数据和 XML 数据的强大的数据库软件。DB2 Express-C 是则 DB2 的一个免费的、无限制且易操作的一个版本。DB2 Express-C 中的字母 ‘C’ 代表社区（Community）。DB2 Express-C 社区中的每位成员以各种方式相互帮助。DB2 Express-C 社区由各种设计、开发、部署或利用数据库解决方案的个人或公司组成。社区成员包括：

- 需要一个以建立独立的、C/S 结构的、基于 Web 的企业级应用为目标的开放标准数据库软件的应用程序开发者。
- 想把一个功能齐全的数据库捆绑或嵌入其解决解决方案的独立软件开发商（ISVs）、硬件提供商、基础设施提供商以及其它类型的方案提供者。
- 需要一个强壮的数据库服务器用来培训、开发技能、评估和制作原型的顾问、数据库管理员和 IT 架构师。
- 需要一个日常应用和操作的可靠数据库服务器的起步型、小型和中型企业。
- 为构建 Web 2.0 和下一代应用程序而寻找易用数据服务的数据库爱好者和先进技术热衷者。
- 需要为教学、课程、工程和研究使用高度通用的数据库服务器的学者、教师和其它学术研究者。

DB2 Express-C 具有与其它用于 Linux, UNIX, Windows 操作系统中非免费的 DB2 版本相同的核心功能和基本代码。DB2 Express-C 可以在 32 位或 64 位的 Linux 或 Windows, Solaris(x64) 和测试版本的 Mac OS X(x64) 操作系统上运行。同时可以运行在拥有任何数量的处理器和存储器的系统上而没有任何特殊的存储器和系统安装需求。DB2 Express-C 也免费提供 pureXML。pureXML 是一项独一无二的为 DB2 原生地存储和处理 XML 文档而开发的独特技术。

1.1 免费开发、部署和分发... 无限制！

这句话总结了 DB2 Express-C 的主要思想：

- 免费开发：如果您是一个应用程序开发人员并且需要一个应用程序数据库，您可以使用 DB2 Express-C。
- 免费部署：如果您的工作在生产环境并且需要一个数据管理系统来存储您的重要记录，您可以使用 DB2 Express-C。
- 免费分发：如果您正在开发一个应用程序或一个需要嵌入数据库的工具，您可以使用 DB2 Express-C。即使 DB2 Express-C 嵌入到您的应用程序中，当您出售您的应用程序并发布时，它 DB2 Express-C 仍然是免费的。为了分发 DB2 Express-C，您必须在 IBM 注册，这个注册也是免费的。
- 无限制：众多数据库竞争者提供数据库时，在数据库的规模、数量和用户数上都设有限制，而 DB2 Express-C 没有任何相关数据限制。您的数据库可以无限增长而不会违反协议许可，并且，关于连接数和每个服务器上的连接数和用户数量也没有任何的限制。

注意:

更多关于 DB2 Express-C 以及它在信息需求世界和 Web2.0 中所扮演的角色, 请参见以下视频:

<http://www.channeldb2.com/video/video/show?id=807741:Video:3922>

1.2 下载 DB2 Express-C

所有的 DB2 Express-C 的映像可以从 ibm.com/db2/express 下载和使用。下列映像可以使用:

- DB2 Express-C 9.7.2 for Windows
- DB2 Express-C 9.7.2 for Windows 64-bit
- DB2 Express-C 9.7.2 for Linux
- DB2 Express-C 9.7.2 for Linux 64-bit
- DB2 Express-C 9.7.2 for Linux on Power
- DB2 Express-C 9.7.2 for Solaris x86-64
- DB2 Express-C 9.5.2 beta for Mac OS X

注意:

DB2 Express-C 还为 Windows 提供了一个较小的版本, 这个版本比普通的版本小 44%。它只有英文版本并且不包括 GUI 工具和文本搜索功能。

1.3 用户帮助和技术支持

有关 DB2 Express-C 的技术疑难, 您可以在 DB2 Express-C 论坛上提交您的问题。

这个免费论坛由来自 IBM 的 DB2 专家们来管理, 尽管这个论坛以自愿为基础向社区成员提供问题的答案。

IBM 也提供一个便宜的 DB2 Express 数据服务器软件 (“DB2 Express”) 的年度订购 (也叫作 12 个月的许可和订购或 FTL, Fixed Term License)。这个订购提供一天 24 小时, 全年无休的技术支持和软件更新。一年的低价订购 (例如在美国约是\$1,499 /服务/年, 价格会随国家而不同), 除了技术支持和软件维护, 同时也可以得其他的附加功能: HADR (高可用性灾难恢复) 和 SQL 复制 (用于从其它的 DB2 服务器复制数据)。关于 DB2 Express-C 订购的进一步信息可以在下面网址找到: www.ibm.com/db2/express/support.html

1.4 DB2 服务器

所有的 DB2 服务器版本都包含相同的核心组件; 用户可根据各自所需来选择不同价格的组件配置。图 1.1 图示了 DB2 产品的不同版本。

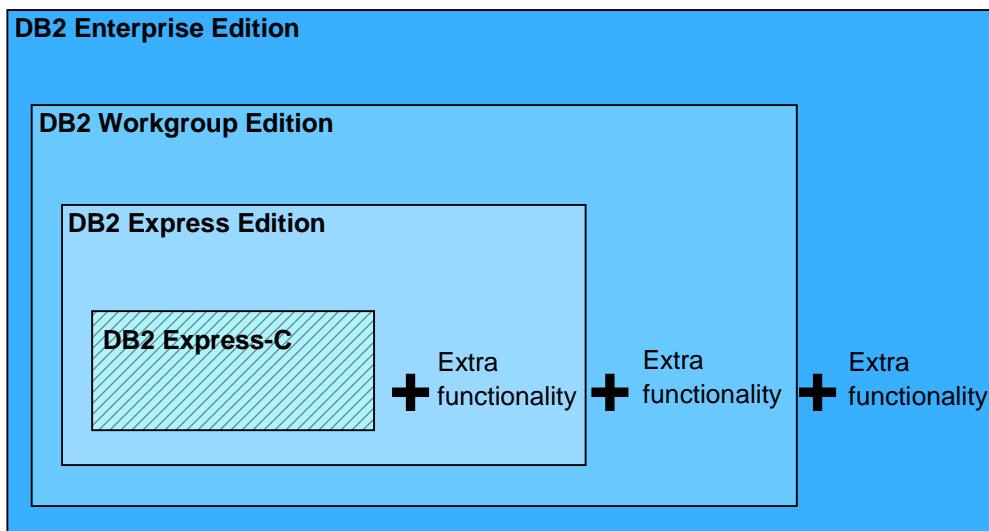


图 1.1 –DB2 服务器

如图 1.1 所示，DB2 Express-C 与去除某些组件的 DB2 Express 相同。DB2 Express-C 对社区是免费的。可以通过免费论坛获得技术帮助或者如果您购买了一年的订阅（Fixed Term License）就可以得到正式的 24 x 7 的 IBM DB2 技术支持。

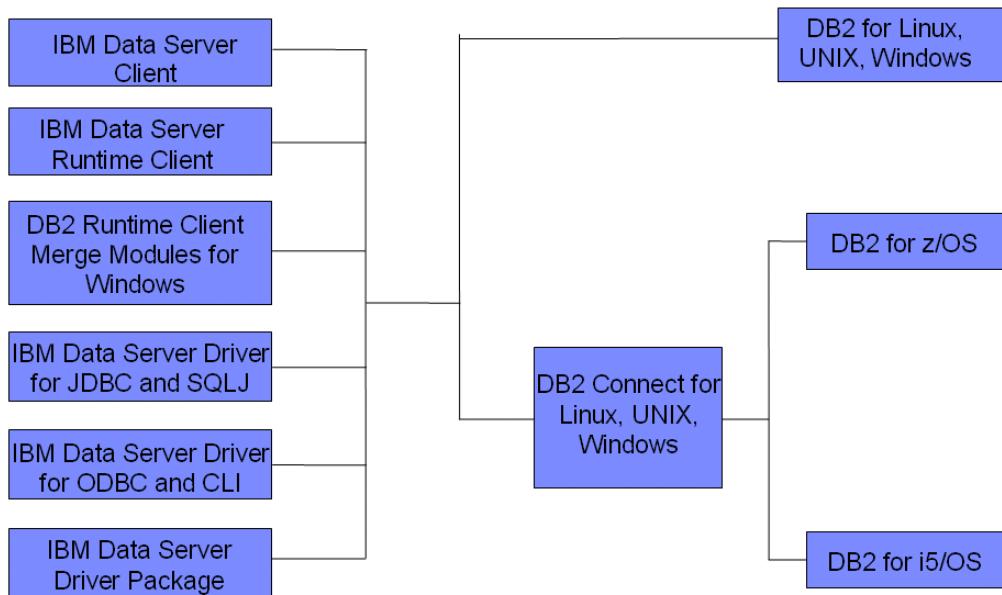
图 1.1 也解释了容易从 DB2 Express-C 升级的原因。因为所有 DB2 服务器拥有相同的核心组件，如果您愿意，您可以将 DB2 Express-C 升级到任何 DB2 服务器版本。这也意味着基于某一 DB2 版本的应用程序可以无修改地运行在其它的高级的 DB2 版本上。而且从一个版本中所学到的技能同样适用于其它版本。

1.5 DB2 客户端和驱动

一个 DB2 客户端包含了连接到 DB2 服务器端的所需的功能；但是 DB2 客户端并不总是需要安装的。例如，JDBC Type 4 应用程序为了可以直接连接到 DB2 服务器端，仅仅需要安装一个 JDBC 驱动就可以了。DB2 客户端和驱动是以多种形式存在的：

- IBM 数据库服务器客户端：十分完善，包括 GUI 工具集和驱动程序集。
- IBM 数据库服务器运行时客户端：只拥有基本功能和驱动程序集的轻量级客户端。
- DB2 Windows 的运行时合并模块客户端：主要用来嵌入一个 DB2 运行时客户端以用作 Windows 安装应用程序的一部分。
- JDBC 和 SQLJ 的 IBM 数据库服务器驱动：允许 Java 应用程序连接到 DB2 服务器。
- ODBC 和 CLI 的 IBM 数据库服务器驱动：允许 ODBC 和 CLI 应用程序不用安装大型的客户端而直接连接到 DB2 服务器端。
- IBM 数据库服务器驱动包括：作为 ODBCCLI 和开源的补充，用来支持.NET 应用环境的 Windows 驱动。这个驱动就是原来 ODCB、CLI 和.NET 的 IBM 数据服务器驱动。

图 1.2 展示了不同的 DB2 客户端和驱动

**图 1.2 DB2 客户端和驱动**

在图 1.2 的左边列出了所有 DB2 客户端和驱动。尽管所有的 DB2 客户端都包含了必需的驱动，但是从 DB2 数据服务器软件（“DB2”）v.9 开始我们也提供了单独的驱动。DB2 客户端和驱动可以在 DB2 Express-C 网站免费下载。客户端和驱动可以用来连接到 Linux, Unix 或 Windows 的 DB2 服务器上的。为了连接到 z/OS® 或 i5/OS® 的 DB2 服务器，则需要通过 DB2 连接服务器（DB2 Connect™ server，如图 1.2 中间部分所示）。我们将在第二章中讨论 DB2 的连接软件（“DB2 Connect”）。

注意：

虽然这本书集中说明 DB2 数据服务器，但是，IBM 数据客户端也可以与 IBM 公司其他的数据服务器链接，如：Informix。因此，一般命名为“IBM 数据服务器客户端”，而不是更具体地称为“DB2 客户机”。

1.6 应用程序开发的自由性

DB2 提供了一个基于标准的和在 DB2 家族中是透明的应用程序开发环境。DB2 产品线中的 SQL 标准提供了通用的、用于数据存取的应用程序开发接口。

另外，每个 DB2 产品都提供了 SQL 预编译器和应用程序编程接口（APIs），它允许开发人员在移动应用程序开发中嵌入静态或动态 SQL。DB2 甚至有一个原生的、由.NET 管理的数据提供者（provider）并且与 Microsoft® Visual Studio 工具集成在一起。

在 DB2 开发中可以使用的语言和标准包括：

- SQL, XQuery, XPath
- C/C++ (CLI, ODBC and embedded SQL)
- Java (JDBC and SQLJ)
- COBOL
- PHP
- Perl
- Python
- Ruby on Rails
- .NET languages

- OLE-DB
- ADO
- MS Office: Excel, Access, Word
- Web services

1.7 DB2 版本号与 DB2 版本分类

如果您是一个 DB2 的初学者，您可能不太清楚 DB2 版本号与 DB2 版本分类之间的区别。

每几年，IBM 都会公开发布一个新的 DB2 版本号。新版本号包括新特性和对产品的重要改进。目前，DB2 Version 9 被 IBM 官方支持。一个版本号也许会有几个发布号，来发布一些虽新但不足以发布新版本号的新功能。例如：9.5 和 9.7 是 DB2 Version 9 的发布号。依过去来看，IBM 差不多每年都会发布一个新的 DB2，但是需要 3 年或者更长才有一个新的版本号。当前最新的发布号是 V9.7（先前以 DB2 ‘Viper 2’为代码编号）并在 2007 年十月普遍使用（Generally Available，GA）。每个发布号也会有一些修改号，但它们通常包含对缺陷的修复而不会有新的功能。当前最新的 DB2 Express-C 的版本号，发布号，修改号（V,R,M）是 9.7.0，表示代码级为 9.7，修复补丁为 0，也就是说这是一个普遍可以的版本（GA）。

另一方面，版本分类是每一个版本号中选取适当的组件进行打包以适应不同的功能需要。正如之前讨论，一个版本分类是一个依不同的价格和许可而打包成不同的功能集。DB2 V9.7（也叫做 DB2 9.7）有许多版本分类。例如，DB2 Express-C 9.7，DB2 Express 9.7，DB2 Workgroup 9.7，和 DB2 Enterprise 9.7（如图 1.1 所示）。

1.8 升级到其他的 DB2 版本

随着数据库需求的增加，您可能想升级到 DB2 的更高版本以支持更强大的硬件配置。这种情形下，它是很容易升级的：

- 如果您在同一台电脑上从 DB2 Express-C 升级到 DB2 Express（固定期限许可证），您需要做的就是把许可证用于 db2licm 命令。
- 如果您在相同的计算机系统中升级，首先卸载 DB2 Express-C 然后安装新的 DB2 版本。当您卸载 DB2 Express-C 的时候，数据库不会被删除（推荐先对数据库进行备份）。
- 如果升级时，要将新的版本安装到相同操作系统但硬件更强大的计算机时，请直接安装新的 DB2 版本在更强大的计算机上，然后从原来的计算机里备份您的数据库映像，并将其拷贝到新计算机上，再在新计算机中恢复数据备份映像。您也需要从原计算机中保存实例配置设置（dbm cfg）并将这些设置应用到新系统中。备份和恢复命令的更多细节将在第 11 章-备份和恢复中讨论。dbm cfg 的更多细节将在第 5 章-DB2 环境中讨论。
- 不管是以上那一种情形您的客户程序都不需要修改。

1.9 DB2 Express-C 的维护和更新

DB2 Express-C 的安装映像一直都更新，这些更新包括最新的发行或者更新的版本又或者有最新的漏洞修复包。DB2 Express-C 的更新，过去通常是一年一次。DB2 Express-C 是一个不收费和免费的产品，没有任何官方的维护更新或者定期的补丁（通常一年更新几次）。一旦 DB2 Express-C 有了最新更新或者升级，之前的版本的 DB2 Express-C 将不再维护。

我们在前面曾经讨论过，如果您需要访问安全补丁和定期软件的更新或者漏洞修复包，IBM 提供：DB2 Express 年度订阅的许可证。一旦您购买了订阅，您可以使用您的许可证进行对 DB2 Express-C 安装的更新，这让您在订阅期间得到 DB2 的技术支持和访问更新及漏洞修复包。订阅许可证让您得到免费的版本更新，或者您可以选择只使用某个特殊的版本或者更新，只要那个版本仍然被支持的，您订阅期间就可以要到相应的漏洞修复包和安全补丁。

1.10 相关免费软件和 DB2 组件

从 DB2 Express-C 下载页(www.ibm.com/db2/express/download.html)下载的软件都是免费的。除了 DB2 Express-C 的压缩包之外，这里还有其它可以免费下载和使用的有用软件：

- Visual Studio Add-ins
- DB2 Spatial Extender

也有面向 DB2 Express-C 初学者的其他一些工具集，可以从 IBM Alphaworks 网络站点 (www.alphaworks.ibm.com/datamgmt) 上下载：

- Starter Toolkit for DB2 on Rails (www.alphaworks.ibm.com/tech/db2onrails/)
- Web 2.0 Starter Toolkit for DB2 (www.alphaworks.ibm.com/tech/web2db2)

如果您正想要一个免费的轻量级的应用程序服务器，IBM 可提供：

- WebSphere® Application Server – Community Edition (WAS CE)

注意:

虽然 DB2 有“空间扩展”功能已接近 10 年，但仍有许多用户不知道。您可以在包括 DB2 Express-C 的所有 DB2 版本中免费利用 DB2 Spatial Extender。Spatial Extender 允许您通过 SQL 来处理空间和地域上的数据。例如：此功能可以帮助您解答类似“每一个住在多伦多去年到我们这里消费超过 3000 美元的客户距离哪一家零售店最近？”的问题。您甚至可以把 DB2 Spatial Extender 用到医疗上。例如：此功能可以帮助您回答类似“什么是恶性细胞的 MRI 脑扫描成像模式？”的问题。

欲了解更多信息，请参阅在为 Linux, UNIX 和 Windows Information Center 提供的 IBM DB2 数据库中关于 DB2 Spatial Extender 的话题。

1.10.1 IBM 数据工作室 (Data Studio)

IBM 数据工作室是一个基于 Eclipse 的工具，它允许您在整个数据管理生命周期里管理您的数据库帮助您开发 XQuery, SQL scripts, 用户定义函数和存储过程并包括一个集成的调试器。另外，IBM 数据工作室允许您用物理数据模型图 (PDM) 来理解表之间的实体关系。数据工作室也可以帮您仅使用简单的拖放功能，就能像 Web 服务一样开发和发布您的数据而不用编程。数据工作室 IBM Data Studio 目前以代替 DB2 一些已过时工具，如控制中心 Control Center 和命令编辑器 Command Editor（它们都包含在 DB2 中，但不再发展了）。我们将在第 5 章 DB2 工具中讨论数据工作室。

1.10.4 DB2 文字搜索

DB2 文字搜索 Text Search 是 DB2 一个可选的集成组件。这是一项由 IBM OmniFind™ 支撑的技术，这项技术支持对包括存储在 DB2 中的 XML 文档在内的文件进行强大，快速和细节性的全文搜索。这个组件使用语言学处理，在文本中发现不同形式的搜索术语。例如，如果您正在寻找一个词“study”，DB2 文本搜索也会同时搜索其他形式的词汇例如“studies”或者“studied”。

为了安装 DB2 文本搜索组件，选择自定义安装，然后在服务器支持类选择 DB2 文本搜索。

注意:

类似的功能在被叫做 Net Search Extender (NSE) 的 DB2 扩充器中也是有用的。有了 DB2 文字搜索，NSE 正在被淘汰。

1.10.5 WebSphere Application Server – Community Edition

IBM WebSphere Application Server Community Edition(WASCE)是一个轻量级的可免费获得的 Java EE 5 应用程序。基于 Apache Geronimo 技术。它利用开源社区的最新创新，来开发一个集成的、可用的和灵活的平台，用以开发和配置 Java 应用程序。可选的 WASCE 技术支持可以从年度订阅中获得。

1.11 小结

DB2 Express-C 版本提供了一个无成本的最好的产品种类。它提供了没有数据库尺寸限制的自由开发，设置和发布，而且它有着和 DB2 相同的内核功能及 pureXML 技术。DB2 Express-C 支持很大范围内的客户，驱动和开发语言，它同时提供了升级到其他 DB2 版本的简单路线。

2

第 2 章 – DB2 相关特性产品

本章描述 DB2 的所有特性，如 DB2 Express-C 年度的订阅许可证（Fixed Term License 或者 FTL），还有在其它 DB2 版本的其它特性，一般情况下，其它的这些特性需要支付额外费用。

Table 2.1 列出了免费（没有保证）DB2 Express-C 和年度订阅许可证选项之间的不同和亮点

特点	免费（无保险）	付费订阅许可证* (FTL)
DB2 核心性能	是	是
免费的管理员工具	是	是
免费的开发工具	是	是
自主能力	是	是
pureXML 特色	是	是
免费的社区支持***	是	是
官方 IBM24/7 支持	否	是
修复包	否	是
高可用性 (HADR)	否	是
SQL 数据复制	否	是
备份压缩	否	是
最大处理器利用	双核	四核（最大两插槽）
最大内存利用	2GB	4GB
有效更新	每年一次的完整更新	每年几次的安全补丁和修复包
访问之前版本的安装映像	否，仅有当前的版本和 beta 版本	是，通过 IBM 护照
每年每个服务器的价格**	0	US \$2,995

Table 2.1：比较免费 DB2 Express-C 和付费订阅许可证的区别

* 以上订阅的特性只有当订阅许可证有效才可以被使用

** 订阅价格以美金结算，如有修改恕不另行通知

*** 免费社区帮助是有在线论坛来实现

表 2.2 列出了产品特征和它们是否包含在 DB2 9.7 的不同版本中。那些您可以单独购买的功能按名字列出了相应的 DB2 版本，并且用浅灰色背景突出显示。

功能	DB2 Express 订阅 (FTL)	DB2 Express 版	DB2 Workgroup Server 版 (WSE)	DB2 Enterprise Server 版 (ESE)
同构 SQL 复制	有	有	有	有
同构联合	有	有	有	有
网络搜索扩展， DB2 文本搜索	有	有	有	有

空间扩展	有	有	有	有
备份压缩	有	有	有	有
pureXML 技术	有	有	有	有
高可用性灾难恢复	有	高可用性	有	有
Tivoli® 系统自动化	有		有	有
高级拷贝服务	无		有	有
在线重组	无		有	有
具体化查询表 (MQT)	无	无	无	有
多维群集 (MDC)	无	无	无	有
查询并行性	无	无	无	有
连接集中器	无	无	无	有
表分区	无	无	无	有
调控器	无	无	无	有
压缩: 行级, 索引, XML, 临时表	无	无	无	存储优化
基于标签的存取控制 (LBAC)	无	无	无	高级采取控制
地域扩展	无	无	无	地域数据管理特征
查询巡视器	无	无	无	性能优化
DB2 负载管理	无	无	无	
性能专家	无	无	无	
同构查询复制	无	无	无	ESE 的同构复制特征
数据库分区	无	无	无	无

表 2.2: DB2 9.7 版中, 不同版本分类的特性和功能对比

在其它的 DB2 版本中可用的功能有:

付费的 DB2 Express 版功能

- 高可用性 (High Availability)

DB2 工作组版本免费提供的功能:

- 高可用性 (High Availability) 和灾难恢复 (HADR), Tivoli 系统自动化, 在线重组, 高级拷贝服务。
- DB2 可用于 UNIX 平台, AIX®, Solaris 和 HP-UX。

DB2 企业版提供免费的功能组件, 如:

- 表 (范围) 分区
- 具体化查询表 (MQT, Materialized Query Tables)
- 多维群集 (MDC, Multi-dimensional Clustering)

- 查询并行性 (Query Parallelism)
- 连接集中器 (Connection Concentrator)
- 调控器 (Governor)

DB2 企业版付费提供的功能:

- 存储优化功能 (包括压缩)
- 高级存储控制 (微调和高级安全性)
- 性能优化 (工作负载管理 Workload Management, 性能专家 Performance Expert, 查询监视器 Query Patrol)
- 地域数据管理 (地理位置分析 geographical location analysis)

相关的 DB2 收费产品:

- DB2 Connect
- InfoSphere 数据仓库版
- InfoSphere 平衡仓库 (InfoSphere Balanced Warehouse)
- WebSphere Federation Server
- WebSphere Replication Server

2.1 DB2 Express 订购许可证 (FTL) 中包含的功能

这一节主要介绍 DB2 补丁包 (Fix pack), HADR 和 SQL 复制。

2.1.1 Fix packs 补丁包

DB2 补丁包 (Fix pack) 是一组附加到已安装 DB2 产品的代码集合, 用于修复产品发布之后所发现的代码问题。如果已经安装了订购许可, DB2 补丁包 (Fix pack) 可以免费下载和安装。通常每四个月或者合理的时间更新一次。

欲下载最新的 DB2 补丁包 (Fix pack), 请查看 DB2 技术支持站点:
http://www.ibm.com/software/data/db2/support/db2_9/

2.1.2 高可用性灾难恢复 (HADR)

高可用性灾难恢复 (HADR) 是一个提供数据库可靠性的功能, 它为系统的部分或整个站点的故障提供一个高可靠的灾难恢复解决方案。一个 HADR 环境通常由两个数据服务器组成, 一个主服务器和一个次服务器 (它们可以是地理位置物理分开的)。主服务器中存有源数据库并可被客户端应用程序访问。当事务在主服务器中被处理时, 数据库的日志记录就通过网络自动被运送到从服务器中。从服务器中有主数据库的完全克隆, 这个克隆是由备份主数据库并在从服务器系统中恢复而产生的。当从数据库接收到主服务器日志记录时, 从数据库会重播及处理该日志记录。通过这样不断重播日志记录对次服务器进行更新, 在次数据库中保存有主数据库的同步复制数据库, 这样, 当主数据库出现故障时就可以由次数据库接管任务。

完全的DB2支持HADR解决方案有:

- 对用户和客户端应用程序来说透明的极速故障恢复
- 保证完全的原子数据库事务的原子性以防止数据丢失
- 不需要中断服务的升级系统和应用程序的能力
- 远程系统故障恢复, 提供从破坏数据中心的本地灾难中完全恢复
- 用 DB2 图形工具方便管理

- DB2 9.7 新的特性将可以让客户在正在待机的服务器上进行阅读。“read-on-standby”的特性可以在 DB2 9.7 的修复包 1 中找到。

注意:

请连接访问以下链接来察看关于HADR如何工作:

<http://www.ibm.com/software/data/db2/express/demo.html>

2.1.3 数据复制 (Data Replication)

这个功能允许在源服务器和目标服务器之间复制数据。源数据库的数据变化会被捕获，然后应用到目标服务器上。图 2.1 演示了复制工作的整个过程。

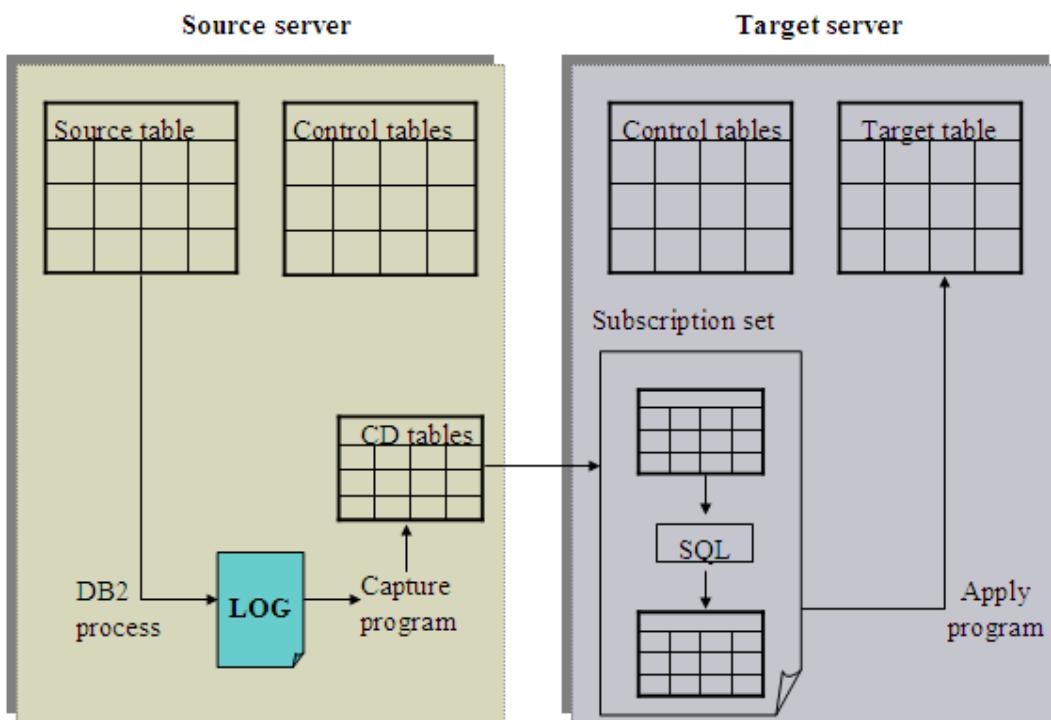


图 2.1 – SQL 复制

在图 2.1 中有两个服务器，一个源服务器一个目标服务器。在源服务器中，有一个捕捉程序可以捕捉到数据库中发生的数据改变。在目标服务器中有一个应用程序，它可以把数据改变应用到目标服务器中。不论是缓和压力、充填数据仓库或是数据集市还是审计数据历史，在需要复制数据的地方，这个功能都是非常有用的。使用 SQL 复制功能可以在 DB2 Express-C 和包括运行于 Linux, UNIX, z/OS, 和 i5/OS 系统的其它 IBM 数据服务之间复制数据。

2.2 DB2 Express-C 所不具备的功能

这一节描述在 DB2 的其它版本有的而在 DB2 Express-C 中或年度订阅的许可证不具有的功能。

2.2.1 数据库分区

数据库分区功能 (DPF) 提供了架构于数据库集群的分布式查询处理功能。目前只有 DB2 InfoSphere Warehouse Editions 才具有数据库分区功能。它允许数据库扩展到多个分区或节点上，这些节点是可以分布在不同计算服务器上的。DPF 是基于无共享体系结构的，每一个数据库分区 (partition) 通常会在它所管理的存储上拥有数据库数据的一个部分。

每个连接到数据库集群中的计算机需要用自己的 CPU, 内存和磁盘来进行数据处理, 并允许把大型任务和复杂的查询分割成更小的部分然后分到不同的数据库节点进行平行处理。相对于数据存放在单一的服务器中, 这样做的结果是提高了并发性和反应时间。DPF 在数据仓库的环境下和商业智能类的工作中尤其有用, 不管是在处理几百 GB 还是几百 TB 的数据量。

2.2.2 连接集中器 (Connection Concentrator)

连接集中器支持同一时间的大量用户连接。以前, 每一个数据库连接都需要一个数据库代理 (database agent)。数据库连接集中器引入“逻辑代理”的概念允许一个代理处理多个连接。代理的更详细内容将在第 6 章-DB2 体系结构中进行讨论。

2.2.3 Geodetic Extender

DB2 Geodetic Extender 是 DB2 企业版中可选的收费功能。这个扩展为需要更容易分析地理位置的商务智能和电子政务的应用提供了便利。DB2 Geodetic Extender 可以构建任何规模的虚拟地球。大部分的位置信息是由 GPS 等全球系统提供的, 并可以表现为经度/纬度的形式 (geocode)。业务数据, 如地址等, 可以由 DB2 Geodetic Extender 转换为 geocode, 企业在这些未规格化的数据上工作得更好, 而不用在表现层规格化为地图数据 (平面地图) 以将它们显示和打印出来。

2.2.4 基于标签的访问控制 (Label-based Access Control LBAC)

LBAC 在行和列层面提供了更细化的保密控制。它使用标签 (label) 去同时和用户会话 (user session) 及数据行或列相关联, 以此来授权表格中的数据访问。图解 2.2 说明了 LBAC 如何工作。

No LBAC	SEC=254	SEC=100	SEC=50	ID	SALARY
SELECT * FROM EMP WHERE SALARY >= 50000				255	60000
				100	50000
				50	70000
				50	45000
				60	30000
				250	56000
				102	82000
				100	54000
				75	33000
				253	46000
				90	83000
				200	78000

图解 2.2 – 一个 LBAC 如何工作的示范

在这个图解中, 表 EMP 有一个列 **SALARY** 和一个内部列 **ID**, **ID** 实际上包含了对应行的标签。在表中其他列仅仅是作为示范目的。如果此表中的查询被执行, 凭靠用户的标签, 他将能够看到不同的行。有标题为"No LBAC"代表了如果 LBAC 没有被实行的, 那么行将被选择。您可以看到所有的工资大于等于 50000 的行都被选中。

现在我们来说一下用户发行一个拥有 100 的安全标签的查询。在这个案例中您能看到从左边数第三纵行被选择的行中, DB2 找到了工资大于或者等于 50,000 的行, 然后查看该行的安全标签。例如, 第一行的工资有 60000, 它的标签 ID 是 255, 而该用户所有的安全标签仅有 100, 所以该用户并不能看到这行, 因此在查询的结果中这一行不能够被返回。

LBAC 安全需要被有 SECADM 权限的安全管理员开发。

2.2.5 工作负载管理 (Workload Manager WLM)

Workload Manager 基于用户和应用优先度，并结合资源可用性和工作量上限来进行数据库范围内的工作负载的管理。它允许您调节自己的数据库工作量和查询，以使重要的和高优先级的查询可以得到及时的运行，并可以防止‘流氓’查询独占系统资源，确保系统有效运行。WLM 是 DB2 9.7 中提供的一个功能，比 DB2 先前版本中 Query Patroller 和 DB2 Governor tools 更强大。

2.2.6 深度压缩

DB2 支持几种形式的压缩：

- **NULL 和默认值压缩 (NULL 和 Default Value Compression)**
这种类型的压缩应用在一些通常为 NULL 或者系统默认值譬如 0 的纵列上，通常不会占用磁盘储存空间。
- **多维集群 (Multidimensional Clustering)**
多维集群 (MDC) 表格，物理数据页被集群在多维里。他们使用块索引 (Block Index)，在某种意义上讲是一种索引压缩，因为他们指向了一些记录的分段而非只有一个记录。
- **数据库备份压缩**
这个是指备份映像，索引和 LOB 数据空间被压缩。
- **数据行压缩**
行压缩是在一行数据中用更小的符号覆盖重复的字符。这个更小字符和符号的映射被保存在一个字典中。行压缩可以急剧的提高 I/O bound workloads 的性能，更多的行可以从磁盘到内存中被发来发去。因为行更小，所以您可以从存储中收益，这也是 IT 预算中最为昂贵的一项开支。对于 CPU-bound workload，这里可以有一些多余的开销，在处理数据之前需要被解压缩。注意：从被压缩的记录中写的日志数据也是同样以压缩格式存在。

当访问 XML 和 LOB 纵列的时候，一般 DB2 不会使用缓冲池 (bufferpoolMemory)，但是对在磁盘中直接执行的 I/O。这样是可以被完成的，这样做的原因是因为 XML 和 LOBs 通常尺寸比较大，因此如果把它带到缓冲池内存中很有可能将会导致需要移动页面到缓冲池内存中现有的数据页将被回写到存储上或清洗掉。使用 DB2 9.7，XML 允许内联 (inline) 线一些小的 XML 文档（小于 32K）。这意味着小的 XML 文档可以在再基本的表格行中被储存，而且不会被存在一个单独分开的内部存储对象：XDA 中。这种方法接近的两种优势是：第一，可以通过缓冲池 (bufferpool) 访问 XML 文档，第二，XML 文档也可以从数据行压缩中获益。

DB2 9.7 中更进一步增强了压缩功能：

- XDA 内部对象 (XML 存储的地方点) 现在可以被压缩。
- 索引和临时性的表格 (系统和用户) 可以被压缩。
- LOBs 可以被内联成跟相类似于 XML 的内联，现在 LOBs 可以同样以相似的方法被内联。

2.2.7 SQL 兼容性

new in
V9.7

当许多厂商跟随者 SQL 92 和 SQL/PSM 的标准，并不是所有标准的特性都是可以被支持，在标准中没有被支持的特性有。随着 DB2 9.7 SQL 兼容性特性，除了 DB2 自己的 SQL PL 以外，DB2 现在可以支持大部分被其他的 RDBMS 厂商所支持 PL/SQL 语法。图解 2.3 总结这种兼容性是如何做到的。

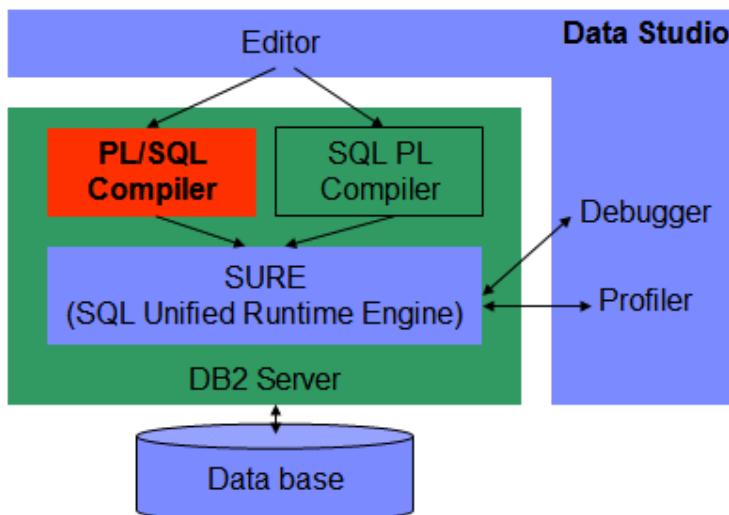


图 2.3 –DB2 中支持的 PL/SQL

从上图中，我们可以注意到，一个 PL/SQL 编译器已经被开发和修建到 DB2 的引擎中。

SQL 的兼容性特性也同样包含对 CLPPlus 工具的支持。CLPPlus 是一个命令行工具，允许运行 SQL 和其他命令。它跟 DB2 的命令行处理器（CLP）很相似。图解 2.4 展示了 CLPPlus 工具。

```

c:\ Command Prompt - clpplus
c:\>>clpplus
CLP Plus: Release 1.0
Copyright <c> 2008, IBM CORPORATION. All rights reserved.

SQL> connect srielau@localhost:50000/stmtconc
Enter Password:
Connected to DB2/NT SQL09070 <localhost:50000/stmtconc> AS srielau

SQL> select sysdate from dual;
1
-----
2008-08-02 19:38:34.0
SQL>

```

图 2.4 – CLPPlus

现在 DB2 支持大多数的 RPL/SQL 数据类型，例如 BINARY_INTEGER, RAW 等等。其他的 Oracle 数据类型例如 VARCHAR2 也被支持，而无需 SQL 兼容性特性。但是您必须用 DB2_COMPATIBILITY_VECTOR 注册表变量来打开这个选项。书中对于 Oracle 数据类型和注册表变量有更多相关的解释。

SQL 兼容性特征的概况现在可以在 DB2 9.7 工作组和企业版本中找到。而且也期待在 DB2 Express 中出现（包括年度订阅选项或者 FTL）。

目前 DB2 Express-C 9.7 还不支持 PL/SQL 支持和 CLPPlus 的特性，但是现在包括的一些特性简化启动 Oracle 应用程序到 DB2。这些特性包括新的数据类型，新的标量函数，模块支持和 currently committed(CC) semantics 对于游标稳定性（cursor stability）的隔离级。这些特征将在后面介绍。

2.3 DB2 相关收费产品

这个部分提供对于 DB2 中收费产品和服务的简要叙述。

2.3.1 DB2 连接 (DB2 Connect)

DB2 连接是允许 Linux, UNIX 或 Windows 客户端连接连接到 z/OS 或 i5/OS 服务器端的收费产品，如图 2.5 所示。DB2 连接在相反的连接时是不需要的，即从 z/OS 或 i5/OS DB2 连接到 Linux, UNIX 或 Windows 的服务器端不需要 DB2 连接。DB2 连接有两个可选的主要版本：DB2 连接个人版和 DB2 连接企业版。

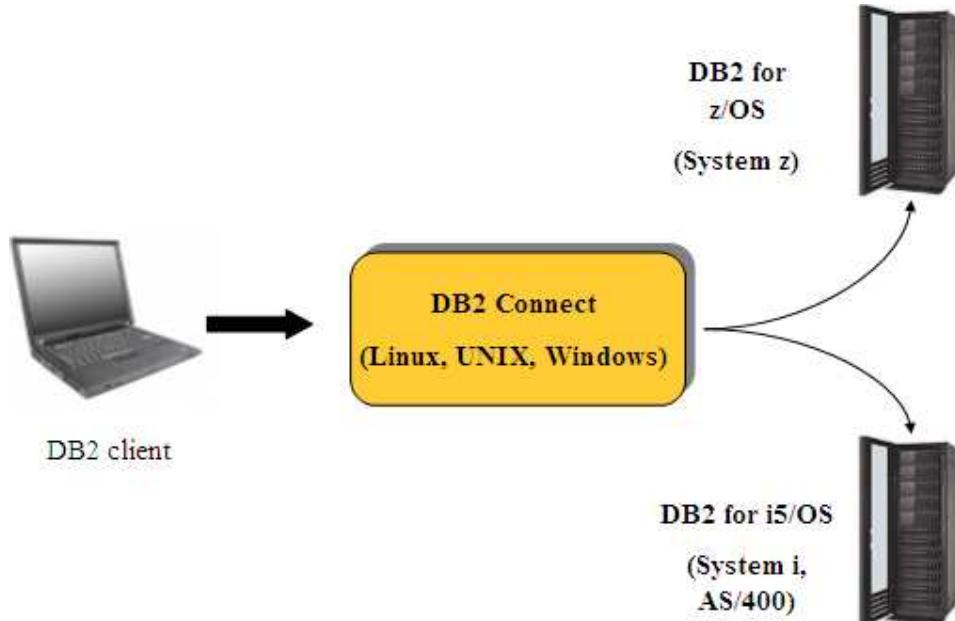


图 2.5 –DB2 连接

2.3.2 InfoSphere Federation Server

以前称为 WebSphere Information Integrator（提供联合支持），现在称为 InfoSphere Federation Server，它允许数据联合，意味着可以对来自不同的关系数据库系统的进行数据库查询。例如，如果您购买了 InfoSphere Federation Server，您可以运行查询语句：

显示在下面的列表 2.1 中。

```

SELECT *
FROM      Oracle.Table1  A
          DB2.Table2  B
          SQLServer.Table3 C
WHERE
          A.col1 < 100
          and B.col5 = 1000
          and C.col2 = 'Test'
  
```

列表 2.1 一个联合查询 (Federated Query)

图 2.6 是一个 InfoSphere Federation Server 的演示。

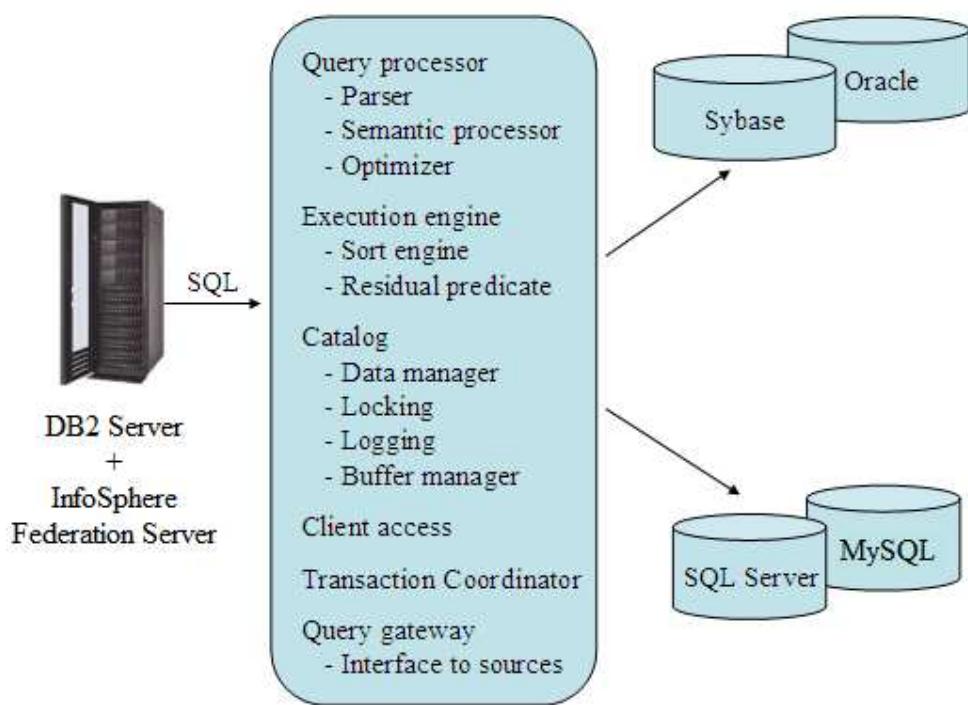


图 2.6 –InfoSphere Federation Server

对于属于 IBM 的数据库管理系统，已经内建了联合支持，这意味着不需要安装 InfoSphere Federation Server 就能够在不同的 IBM 数据库中进行联合操作。例如，欲在两个不同的 DB2 数据库间或一个 DB2 数据库和一个 Informix (Informix 是 IBM 家族的成员) 数据库间执行查询时不需要单独安装 InfoSphere Federation Server。

2.3.3 InfoSphere Replication Server

以前称为 WebSphere Information Integrator (提供复制支持)，现在称为 InfoSphere Replication Server，它允许在涉及非 IBM 数据服务时复制数据库记录。它也包括著名 Q-Replication，Q-Replication 使用信息队列来复制数据。

2.3.4 Optim Development Studio (ODS)

以前称为 Data Studio Developer，ODS 是一个基于 Eclipse 的工具，可以很便捷地与 Data Studio 整合及共享相同的 Eclipse。ODS 可以帮助从已有的 Oracle 或者 DB2 数据库中使用简单粘贴来创建研发数据库。

2.3.5 Optim Database Administrator (ODA)

以前称为 Data Studio Administrator，ODA 是一个以 Eclipse 为基础的工具，可以很便捷地与 Data Studio 整合及共享相同的 Eclipse。ODA 提供了一个改变管理 (change management) 的能力，以及使得对模式改变 (schema change) 做自动化更加容易。

2.4 DB2 的 Amazon Elastic 的云端计算

值得一提的是，IBM 已经同 Amazon Web Services(AWS)就在 Amazon's Elastic Compute Cloud(EC2)上运行 DB2 达成了合作伙伴的关系。AWS 提供了一系列整合服务用来形成一个计算平台 “in the cloud”，一个现购现付的模式。AWS 让您租赁计算熔炼 (虚拟服务器和储存空间)，然后您仅仅需要支付您需要使用的容量。打个比方，您准备用 one EC2 虚拟服务做来运行一班的数据

库操作和对于高峰时段或者繁忙时节需要几个小时额外的数据库服务器。在这个例子中，您可以仅仅支付给 AWS 几个小时数据库使用的时间。

IBM 提供了对于 DB2 在 Amazon's cloud platform 上的 3 种不同的调配方案：

- DB2 Express-C AMIs 对于评估和发展
- DB2 Express 和 DB2 Workgroup 的现购现付，生产就绪的 AMIs
- 使用您自己的 DB2 许可证可以去创建您自己的 AMIs

关于如何开始 DB2 在 Amazon EC2 的信息，请访问www.ibm.com/db2/cloud。

2.5 小结

DB2 Express-C 为开发数据库应用程序，配置它们到产品中，甚至为第三方解决方案内置和发布应用程序提供了免费，简单实用和坚实基础。理想状况下，您是对于以社区为基础的帮助和不需要任何最新的修补和高级的特性是没有任何问题的。无论如何，如果您需要从 IBM 得到正式的技术，常规的软件升级（修补包），或者额外的资源利用和高可用性的聚类支持，IBM 提供了一个便宜的年度 DB2 Express 订阅许可证（FTL）。如果您需要更多关于关键任务工作量和大规模数据库应用程序，IBM 提供了可以变更规模的 DB2 版本和相关产品。这允许您可以从 DB2 Express-C 开始，然后随着您业务的发展来进一步扩展。

3

Chapter 3 – DB2 installation

安装 DB2 相当简单，在一个典型安装中，只需要选择默认选项就可以在短时间内建立并运行一个 DB2 服务器。

首先从 DB2 Express-C 网站(www.ibm.com/db2/express)为您的平台下载一个适合的 DB2 Express-C 映像。

3.1 安装前提条件

DB2 Express-C 对应 Linux、Sun Solaris (x64)、Microsoft Windows 2003/XP/Vista 的版本，它也可作为对 Mac OS X 的测试版。而且不同的版本细分为 32bit、64bit、PowerPC (Linux) 以对应不同的 CPU 架构。如果您想在其它的平台上（如 Unix）运行 DB2，您必须购买前面提到的其它版本的数据库服务器。下面链接的文档中详细描述了 DB2 不同版本对操作系统的要求。

<http://www.ibm.com/software/data/db2/udb/sysreqs.html>

在硬件方面，DB2 Express-C 能够安装在任意 CPU 核心和内存的系统上。但必须注意，免费版本的 DB2 Express-C 只能够利用最大 2 核的 CPU 和 2G 的内存，付费版本则能够使用最大 4 核的 CPU 和 4G 的内存。DB2 能够安装在实际的系统中，或者是虚拟机上。当然，您也可以在更少资源的系统上运行它，比如在单 CPU 和 1G 内存的机器上运行。

关于硬件要求的最新消息，请查阅 DB2 Express-C 的网站：
<http://www-306.ibm.com/software/data/db2/express/getstarted.html>

3.2 操作系统中的安装权限

要将 DB2 Express-C 安装到 Linux 或者 Windows 上，您必须拥有一个足够权限的帐户。

在 Linux 系统中，您必须是 root (超级用户) 用户来安装 DB2 Express-C。当然，您可以使用其它的帐户来安装 DB2，但是在使用上会受到限制，比如，如果使用非 root 帐户安装的 DB2 Express-C，除了安装时创建的默认实例 (instances) 以外，不能再创建新的实例。

在 Windows 系统中，安装所用的帐户必须属于 Administrators 组。另外，在 Windows2008, Windows Vista 或更高版本中，非管理员可以执行安装，但是 DB2 安装向导会向安装者要求输入管理凭据。

如果安装需要创建或验证一个域账户，安装的用户 ID 必须属于域的域管理员组。

您还可以使用内置的本地系统帐户运行安装，虽然不推荐这样做。本地系统帐户不要求密码，但不能访问网络资源。

安装所用的帐户必须有“从网络访问此计算机”的权限。

注意：

下面链接是一个关于 DB2 Express-C 安装的介绍。虽然这个演示展示的是 DB2 9.5，但是除了安装版面的色彩，与 DB2 9.7 没有区别：

<http://www.channeldb2.com/video/video/show?id=807741:Video:4442>

3.3 安装向导

尽管安装 DB2 Express-C 有不同的方法，然而最简单的方法就是使用一个有界面的安装向导。在下载和解压 DB2 Express-C 后，您可以运行里面的安装向导：

- Windows: 运行在 EXP/image 文件夹中的 setup.exe
- Linux: 运行在 exp/disk1 文件夹中的 db2setup 命令

跟着 DB2 安装向导，整个安装过程是非常简单的。很多时候，默认的设置已经足够了，所以您只需要接受协议，然后点击“Next”几次，直到“Finish”按钮被激活，然后点击“Finish”按键。几分钟后，安装完成，DB2 就安装好了，并且开始正常运行。

图 3.1 展示了 DB2 安装启动面板（DB2 Setup Launchpad）。点击“安装一个产品（Install a Product）”，然后选择“全新安装（Install New）”，就可以将一个新的 DB2 Express-C 安装到您的系统中。如果您以前安装的 DB2 Express - C 或其他 DB2 版本，您可能会看到名为“Work with Existing”的按钮。DB2 里面您可以多次安装产品，安装可以在不同的版本或版本级别上进行。

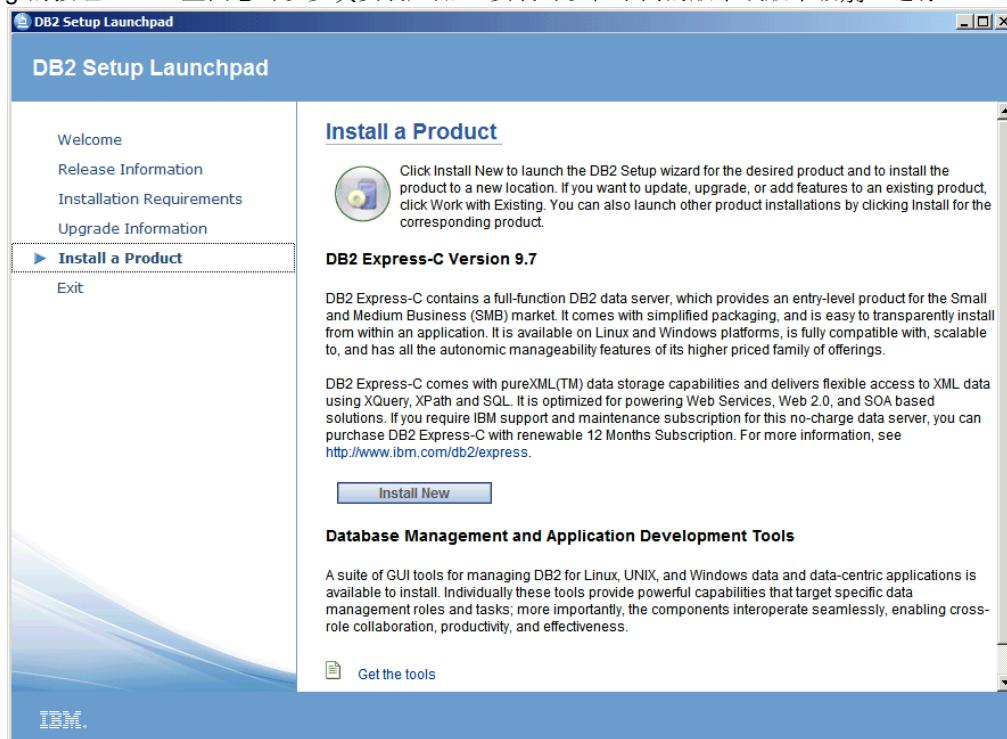


图 3.1 – DB2 安装启动面板（DB2 Setup Launchpad）

接受协议之后，出现安装类型，通常选择“典型（Typical）”安装（默认）就足够了，如图 3.2 所示。如果您想包括 DB2 全文检索组件，选择“自定义”（Custom）。

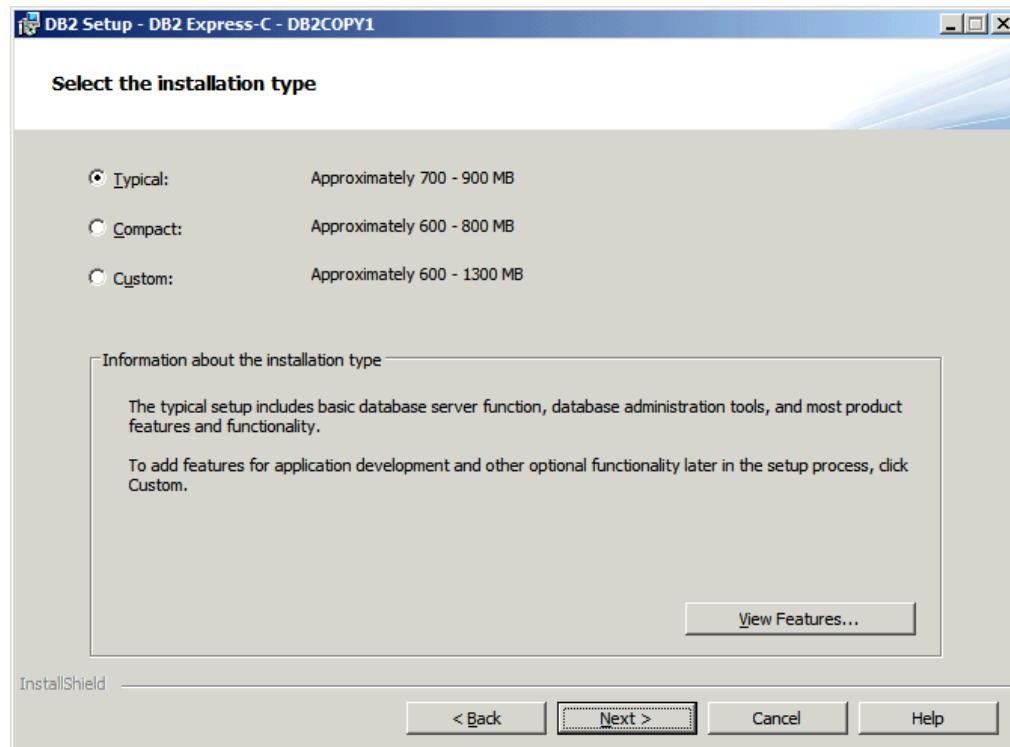


图 3.2 – 安装类型

接下来，在图 3.3 的画面中，您可以选择安装、建立响应文件、或安装并建立响应文件。响应文件在 3.4 节讨论。在这里，默认选项（Install IBM DB2 Express Edition on this computer and save my settings in a response file）已经足够。

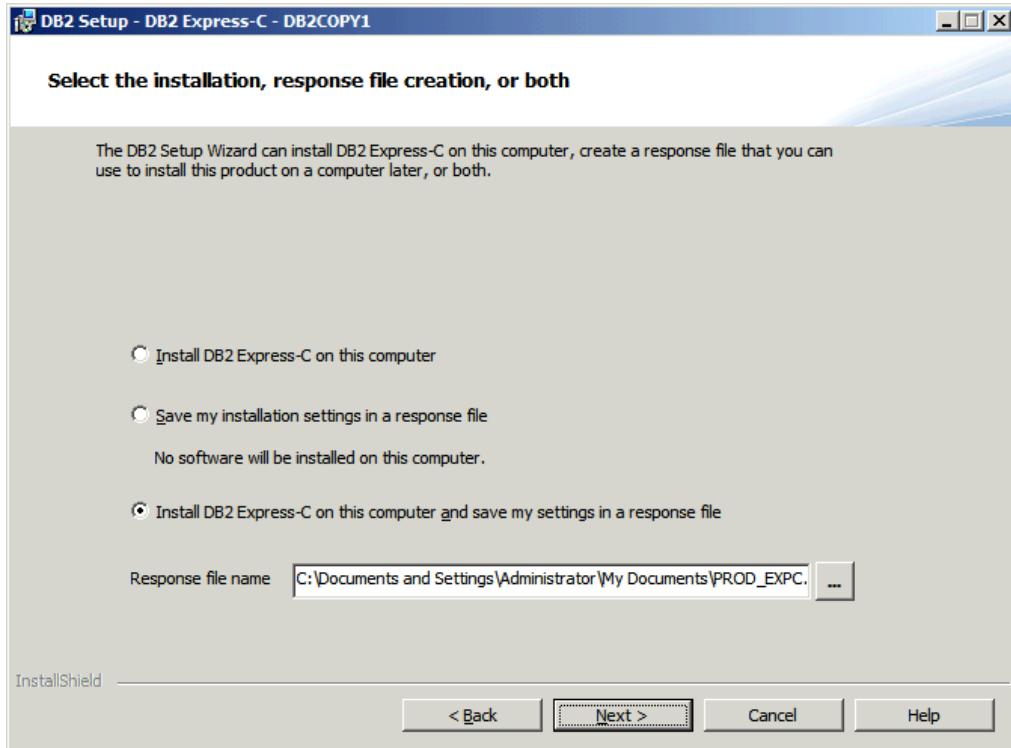
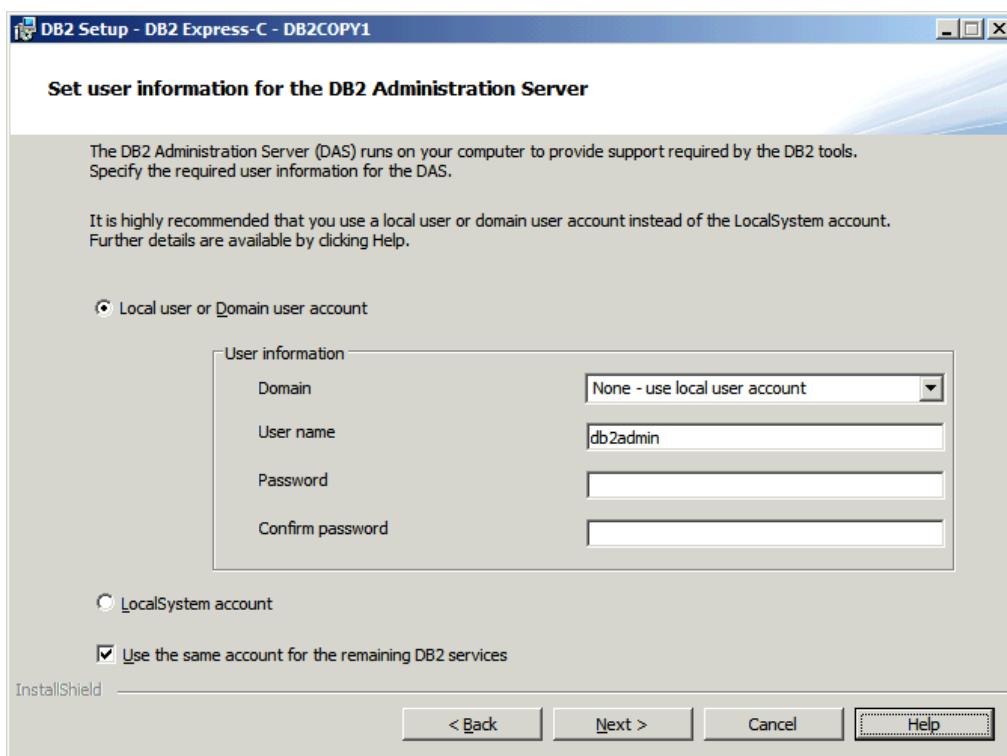


图 3.3 – 选择安装还是创建响应文件

在接下的几步都选择默认设置，当您到了如图 3.4 的窗口时，您可以输入一个用户 ID，这个用户将会使用 DB2 的实例和其它服务。这个用户必须是 windows 中本地管理员（Local Administrator）组的一员。如果您输入的用户 ID 不存在，这个用户 ID 就会被创建，并成为一个本地管理员。如果用户不属于任何一个域，请将域（domain）留空。Windows 中，默认创建的用户名是 db2admin，在 Linux 中，默认创建的用户名是 db2inst1。

**图 3.4 – 为默认的 DB2 实例配置用户信息**

最后，如图 3.5，安装向导显示安装摘要，摘要中显示了所要安装的内容以及每一个阶段的设置值。单击“Finish”执行安装，程序文件将会安装到您的系统中。

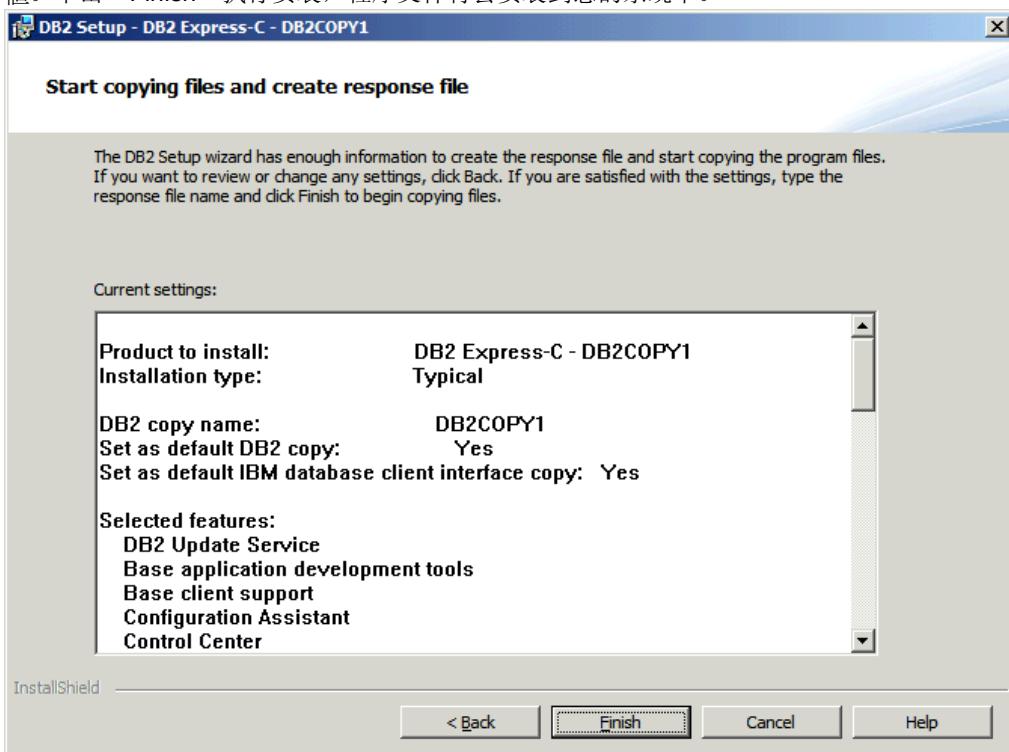


图 3.5 – 安装摘要

当安装完成，类似图 3.6 1 的窗口中会出现。通知您安装向导过程的结果以及任何所需的安装完成进一步的步骤。

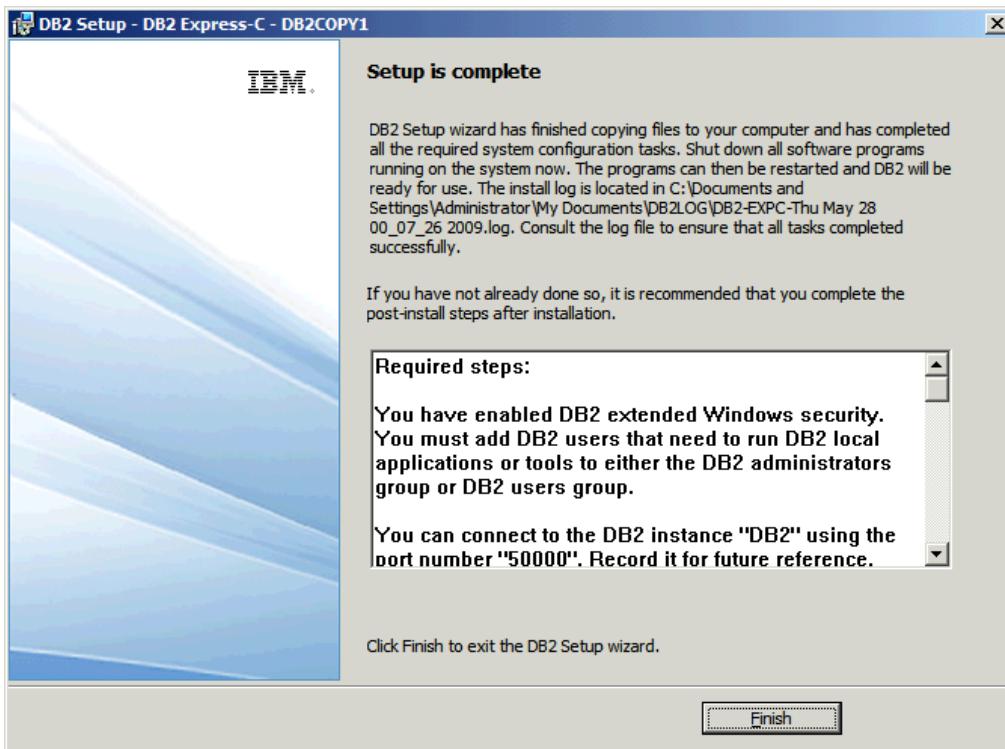


图 3.6 – 安装完成

在单击“完成”，在安装结果窗口，如图 3.6 所示，第一步骤的 DB2 应用程序将启动，如图 3.7 所示。

这个小应用程序概述了几种不同的选择让您开始使用 DB2，如创建默认的示例数据库（适当命名的 SAMPLE）或创建您自己的新的数据库。如果您不想要探索通过第一步的 DB2 在这个时候，您可以关闭该窗口，并在稍后时间调用。

要手动启动 Windows DB2 的第一步，选择“开始”->“程序”->“IBM DB2”->“DB2COPY1”（默认）->“设置了工具”->“第一步骤”或运行命令，从命令提示符 db2fs。

在 Linux 上，执行从终端窗口 db2fs 命令。

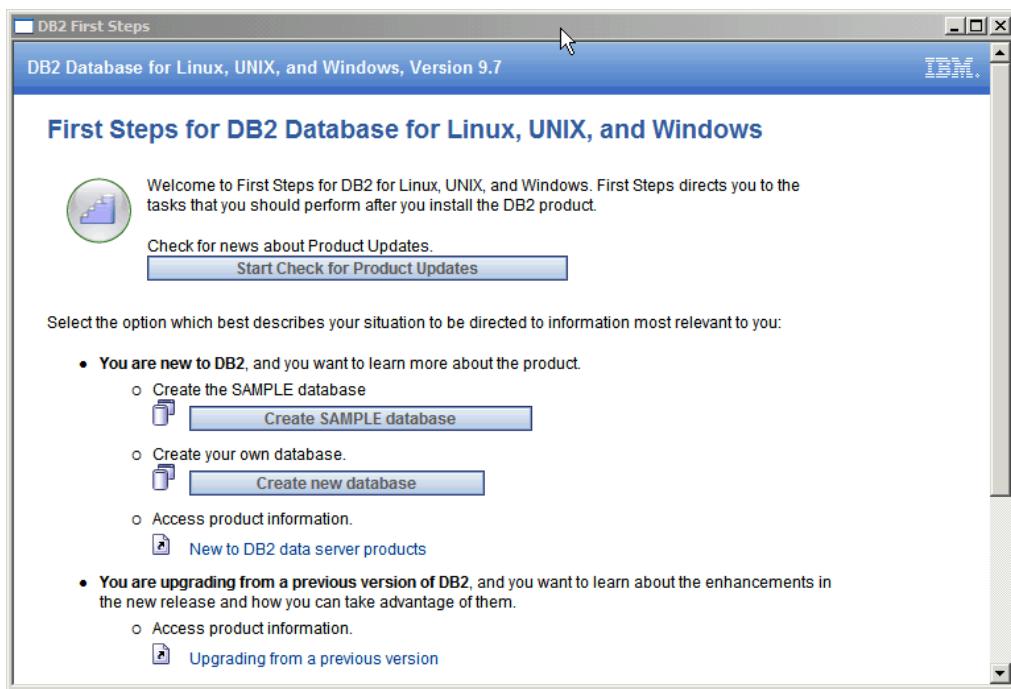


图 3.7 起始步骤

3.4 验证您的安装

安装 DB2 之后，您可以从 DB2 命令窗口（DB2 Command Window）（适用于 Windows）或从终端（适用于 Linux）上，运行三个命令来验证您的安装状态良好：

db2level: 此命令显示有关的 DB2 安装的产品，修订包的水平，和其他详细信息。

db2licm -l: 此命令会列出您所安装的 DB2 信息。

db2val: 这是一个 DB2 9.7 中的新命令。它会验证您所安装的拷贝的核心功能。它会验证您所创建的实例是一致的，并验证数据库的创建及数据库连接。

图 3.8 下面提供了这三个命令的输出示例。

new in
V9.7

```

C:\Program Files\IBM\SQLLIB\BIN>db2level
DB21085I Instance "DB2" uses "32" bits and DB2 code release "SQL09070" with
level identifier "08010107".
Informational tokens are "DB2 v9.7.0.441", "s090521", "NT3297", and Fix Pack
"0".
Product is installed at "C:\PROGRA~1\IBM\SQLLIB" with DB2 Copy Name "DB2COPY1"

C:\Program Files\IBM\SQLLIB\BIN>db2licm -l
Product name: "DB2 Express-C"
License type: "Unwarranted"
Expiry date: "Permanent"
Product identifier: "db2expc"
Version information: "9.7"
Max number of CPUs: "2"
Max amount of memory <GB>: "2"

C:\Program Files\IBM\SQLLIB\BIN>db2val

DBI1379I The db2val command is running. This can take several minutes.
DBI1333I Installation file validation for the DB2 copy DB2COPY1
was successful.

DBI1339I The instance validation for the instance DB2 was
successful.

DBI1343I The db2val command completed successfully. For details, see
the log file C:\DOCUME~1\ADMINI~1\MYDOCU~1\DB2LOG\db2val-Fri May 29 04_1
40 2009.log.
C:\Program Files\IBM\SQLLIB\BIN>

```

图 3.8 - db2level, db2licm, 并 db2val 命令来验证您的安装

在图中, db2level 命令的输出表明您正在运行的 DB2 9.7 (DB2 v9.7.0.441) 在修订包 0, 这意味着您的 DB2 代码, 在 GA 版本上, 并没有任何的修订 (fixes)。在 db2licm -l 命令显示已安装的 DB2 Express - C 版本拥有永久的但未获保证的许可, 允许使用最多二个核心, 以及高达 2GB 的内存。db2val 输出的结果本身已经自我解释。

注意：

如果您想在任何时间验证数据库的一致性, 使用检查 (INSPECT) 工具。

3.5 自动安装

有时, 您需要将 DB2 客户端安装到多台机器上, 又或者您需要将 DB2 数据库服务器嵌入到您的应用程序, 并在安装这个应用程序同时安装 DB2 数据库服务器。这些情况下, DB2 的自动安装是一个理想的方法。

DB2 利用响应文件来进行自动安装, 它以简单的文字选项保存了安装所需的信息。列表 3.1 下面展示了响应文件的一小部分。

```

PROD=UDB_EXPRESS_EDITION
LIC AGREEMENT=ACCEPT
FILE=C:\Program Files\IBM\SQLLIB\
INSTALL_TYPE=TYPICAL
LANG=EN
INSTANCE=DB2
DB2.NAME=DB2
DEFAULT_INSTANCE=DB2
DB2.SVCENAME=db2c_DB2
DB2.DB2COMM=TCPIP
...

```

列表 3.1 – 示例响应文件

有许多方法创建响应文件：

- 在安装 **DB2 Express-C** 过程中，第一步允许您选择将安装信息保存到响应文件中，如图 3.3 所示，您可以设定响应文件的保存路径和文件名。在安装结束时，该向导会在生成到一个指定的目录下用指定的和文件名生成一个响应文件。这是一个文本文件，所以您可以手动编辑后。
- 修改 **DB2 Express-C** 中提供的响应文件样例。响应文件样例（.rsp 扩展名）在 db2/platform/samples/directory。
- 在 Windows 中，您还可以使用相应文件生成器来生成响应文件：

```
db2rspgn -d <output directory>
```

有了响应文件后，可以用响应文件来自动安装 DB2，在 Windows 中，您运行下面的命令来执行安装：

```
setup -u <response filename>
```

在 Linux 中，您运行下面的命令进行自动安装：

```
db2setup -r <response filename>
```

3.6 小结

本章涵盖了安装 **DB2 Express - C** 中的细节。**DB2** 的这个版本可运行于 **Linux**, **Solaris** 和大部分的 **Windows** 系列，可以运行在 32 位, 64 位和 **Power PC** 体系上。讨论了在系统上安装 **DB2** 所需的用户授权之后，我们使用 **DB2** 安装向导的 **GUI** 做了一个简单的安装演练。然后讨论了安装之后的活动，运行 **DB2 First Steps** 和验证安装。最后，我们介绍了如何使用 **DB2** 响应文件创建和执行无提示安装。

3.7 实验

在这个实验中，您将安装 **DB2 Express- C** 和创建 SAMPLE 数据库。

实验目标:

在您开始探索 **DB2 Express-C** 的各项特性，使用各种工具之前，您必须将 **DB2 Express-C** 安装到系统中。在本实验中，您将会在 **Windows** 系统中进行基本的 **DB2 Express-C** 安装，每一步都是非常简单的。

实验过程

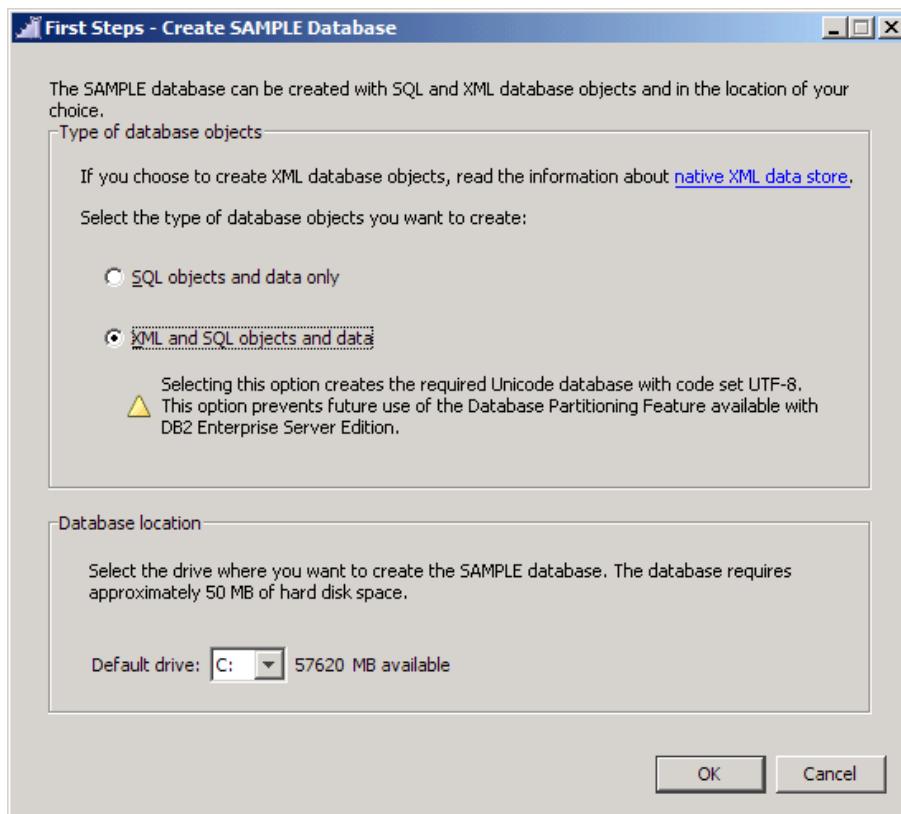
1. 获得 **DB2 Express-C** 安装压缩包。可以在 **DB2 Express-C** 网站（www.ibm.com/db2/express）上下载合适的 **DB2 Express-C** 安装压缩包或者订购包含安装程序压缩包的 **Discovery Kit DVD**。然后将获得的安装压缩包解压缩到自定义的路径。
2. 定位文件。浏览到第一步解压的路径，找到 **DB2** 的安装文件。
3. 运行 **Launchpad**。双击 **setup.exe** 运行 **DB2 Launchpad**。如果是 **Linux** 系统，请以 **root** 身份运行 **db2setup** 命令。在 **Launchpad** 窗口左边的面板中，单击 **Install Product** 选项。
4. 运行 **DB2** 安装向导。**DB2** 安装向导会先检查系统是否满足安装条件，并检查系统是否已经安装过 **DB2**。单击 **Next** 按钮进入下一步。点击 **Install New** 来开始安装，然后点击 **Next**。
5. 查看许可证。查阅并接受许可协议（选择“我同意（I Accept）”），然后单击 **Next**。
6. 选择安装类型。本实验中，选择典型（Typical）选项（默认的选项）。精简 Compact 选项会执行一个最基本的安装，而自定义 Custom 选项则允许您自己定义安装的细节。点击 **Next** 进入下一步。
7. 选择安装，响应文件的创建，或两者都选择：保留默认，让 **DB2** 安装，并创建一个响应文件。单击下一步按钮继续。
8. 选择安装文件夹：在此屏幕上，您可以使用自定义的驱动器和目录来安装 **DB2** 代码。

您需要确保有足够的空间用于安装。在这个例子中，请使用默认的驱动器及目录设置（如下所示）：

```
Drive: C:  
Directory: C:\Program Files\IBM\SQLLIB
```

点击下一步（NEXT）按钮继续。

9. 设置用户信息。当 DB2 Express-C 安装完成后，一些 DB2 进程会作为系统服务运行。因为这些服务需要一个操作系统帐户来运行，所以必须设置必要的用户信息。在 Windows 系统中，推荐使用默认的 db2admin。如果指定的用户不存在，DB2 则会在系统中新建一个用户。您可以指定一个已经存在的系统帐户，不过这个系统帐户必须有本地管理员权限。我们推荐您使用默认的用户名，并确保输入了密码。在 Linux 中推荐使用默认的 db2inst1 作为实例所有者，db2fenc1 作为执行存储过程的隔离用户，dasusr1 作为 DB2 管理服务器 DAS 用户。单击 Next 进入下一步。
10. 配置 DB2 实例。一个 DB2 实例可以看成是数据库的容器。实例必须先于数据库存在，数据库是在实例中创建的。Windows 版本的 DB2 Express-C 安装时，会自动创建名为 DB2 的实例。在 Linux 中，默认的实例名字为 db2inst1。我们会在后面的本书章节详细阐述 DB2 的实例。默认的情况下，DB2 实例去监听的 TCP/IP 连接端口是 50000。默认的协议和端口号可以通过单击协议（Protocols）和开始（Startup）按键来修改。我们推荐使用默认的设置。单击 Next 进入下一步。
11. 开始安装。在安装摘要中先检查一下之前的每一步的选项，单击 Install 按钮开始将文件复制到安装路径中。DB2 同时会执行一些初始化的配置。
12. First Steps。当安装完成后，会显示一个叫 First Steps 的工具。First Steps 工具也可通过 db2fs 来启动。
13. SAMPLE 数据库是本书用于测试和实验目的的数据库。它可以通过点击“Create SAMPLE”按钮，从 FIRST STEPS 中创建。点击此按钮，下图的窗口就会显示。选择第二个选项（XML and SQL objects and data）。SAMPLE 数据库也可以使用命令 db2sampl - XML 的 SQL 语句创建。



14. 几分钟后，您可以验证该数据库已创建。打开 DB2 控制中心，从 Windows 的开始菜单依次点击：*Start -> Programs -> IBM DB2 -> DB2COPY1 (Default) -> General Administration Tools -> Control Center*。您也可以直接用命令 db2cc 来启动控制中心。当您第一次启动控制中心，会有一个弹出窗口要求您选择您想只用的控制中心视图。保留默认（ADVANCED），然后单击确定。在左边的面板，打开 All Database 文件夹。如果您无法看到该文件夹中 SAMPLE 数据库，请务必通过选择 View -> Refresh 刷新您的视图。
15. 重启系统，这一步是可选的。尽管这一步并没有在 DB2 官方安装文档中提及，我们还是推荐您重启系统（在可能的情况下）来确保所有的进程都成功启动，并且清除安装过程中可能没有正确释放的内存占用。
16. 通过运行 db2level, db2licm 和 db2val 命令来验证您的 DB2 安装。从 Windows 开始菜单，打开 DB2 命令窗口如下：*Start -> Programs -> IBM DB2 -> DB2COPY1 (Default) -> Command Line Tools -> Command Window* 在命令窗口（或 Linux Shell）中键入 db2level 和检查输出。对指令 db2licm-1 做同样工作。最后执行 db2val 命令。如果 db2val 成功完成，安装的程序是在良好的状态。如果有错误，请检查日志文件中的错误消息，了解更多细节。这三个命令的输出应该类似于前面图 3.8 所示。

4

第 4 章 – DB2 的应用环境

本章讨论 DB2 的应用环境。图 4.1 是对 DB2 的综合性概述，红色椭圆区域标注了本章重点关注的内容。图的左侧部分介绍了不同的 DB2 命令：SQL，SQL/XML，和能与 DB2 数据服务器相交互的 XQuery 语句。图的中间部分显示了与 DB2 数据服务器相交互所应用的不同工具。图的右侧部分显示了基本的 DB2 环境所涵盖的内容：实例，数据库和相关的配置文件。

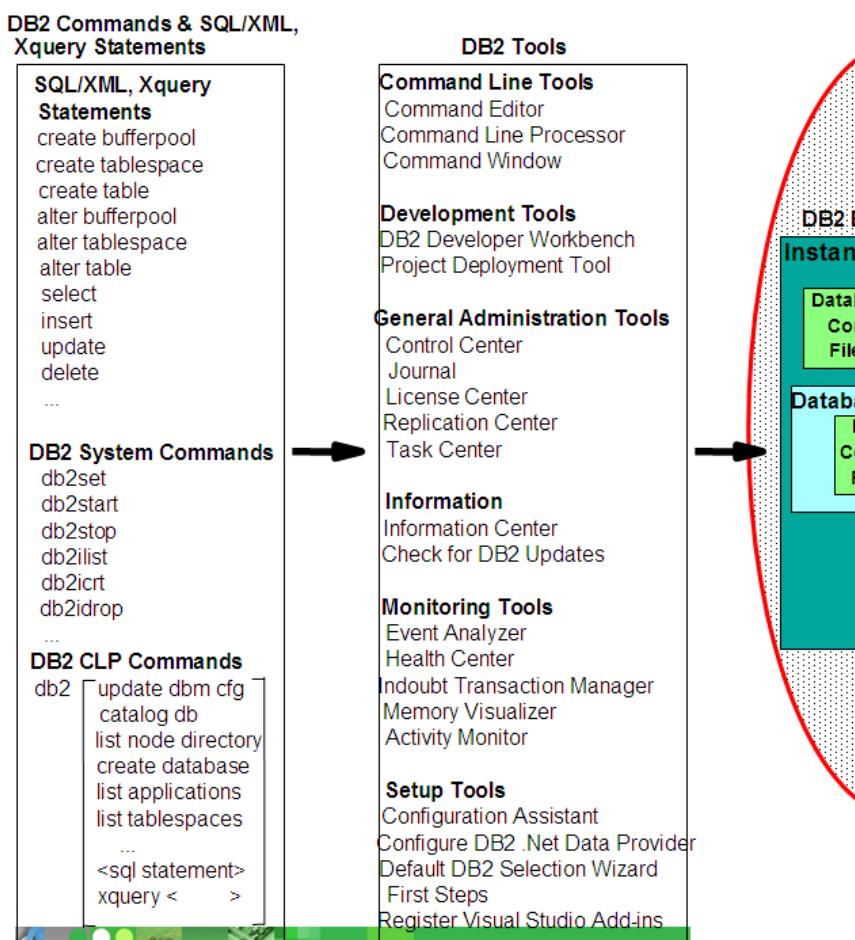


图 4.1– DB2 概览图：DB2 应用环境

注意：

更多关于 DB2 应用环境的信息，请参见以下视频

<http://www.channeldb2.com/video/video/show?id=807741:Video:4029>

<http://www.channeldb2.com/video/video/show?id=807741:Video:4042>

为了讲述 DB2 应用环境，我们可以将会逐步介绍 DB2 的每个组件。

图 4.2 的灰色方框表示安装完 DB2 Express-C 9.7 之后的 DB2 数据服务器。

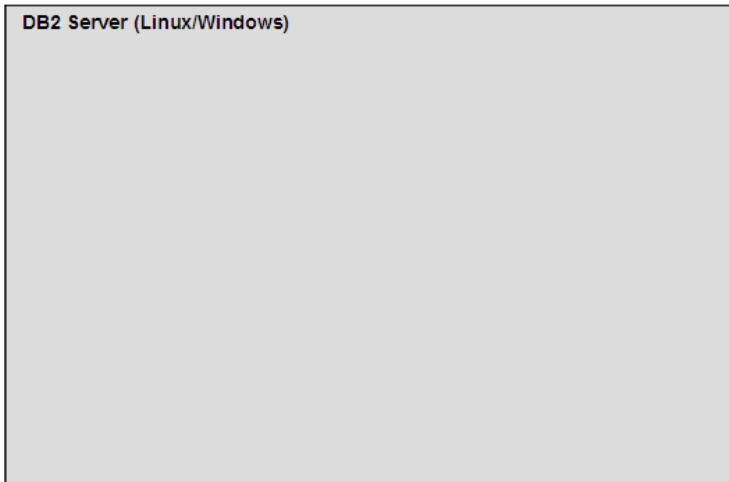


图 4.2 - 安装完 DB2 Express-C 9.7 之后的 DB2 数据服务器

在 Windows 系统中，安装时默认创建一个实例名称为“DB2”（Linux 系统下创建的实例名称为“db2inst1”）。如图 4.3 的绿色部分。实例是应用程序运行和创建数据库所必须的独立环境。一个数据库服务器上可以创建多个用于不同目的实例。例如，一个实例用于针对产品数据库的保存，再有一个实例用于测试环境数据库，还可以有一个实例用于开发环境。所有的这些实例都是相互独立，也就是说在一个实例上实现的操作不会影响到其它实例。

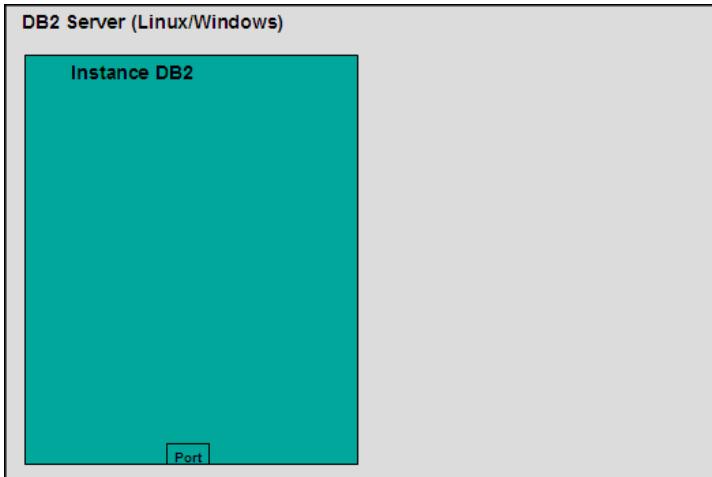


图 4.3 – 默认创建的 DB2 实例

可以用如下命令新建数据库实例，`db2icrt <实例名称>`，这里的<实例名称>可以用任意 8 个字符代替。例如，创建一个名为 *myinst* 的实例其命令为：`db2icrt myinst`。

图 4.4 分隔的绿色区域显示了一个名为 *myinst* 的新数据库实例。

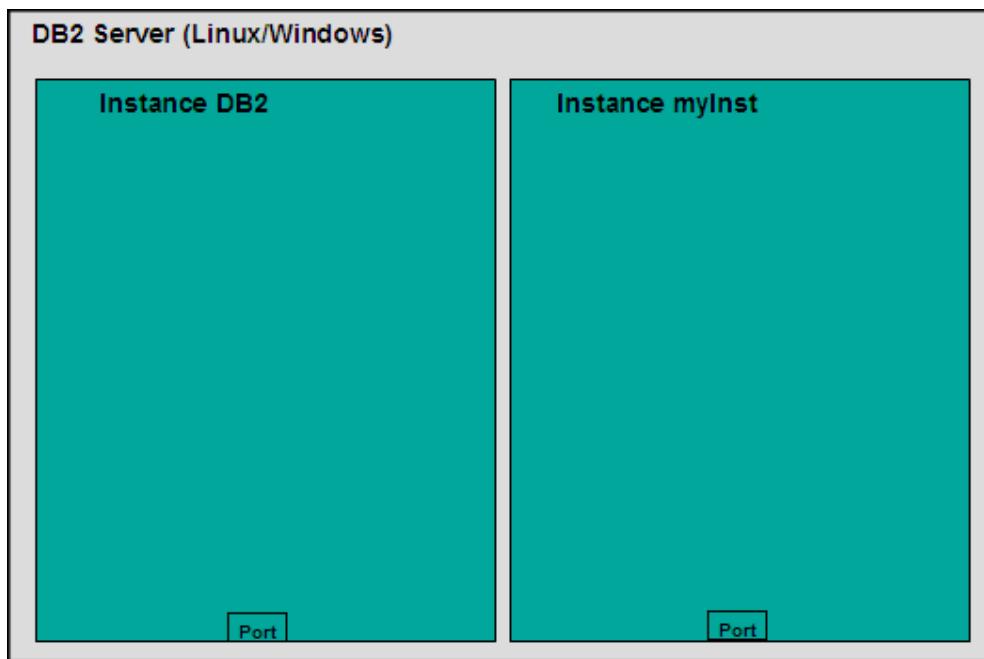


图 4.4 – 一个包含两个数据库实例的 DB2 服务器

必须注意的是每一个实例的端口号必须是唯一的。这样在您远程连接数据库时，此端口号可以保证您能够正确的连接实例。在 Windows 中可以使用 DB2 命令窗口，输入如下命令来激活任意 DB2 实例：

```
set db2instance=myinst
```

这样，如果您现在从命令窗口创建数据库，数据库会在 myinst 实例中被创建。运行如下命令显示所有实例：

```
db2ilist
```

在 Linux 操作系统上，每一个实例必须对应一个 Linux 操作系统用户，因此，两个实例的转换可以通过转换用户得以简单的实现（用 su 命令）。

表 4.1 显示了一些常用的实例层命令。

命令	描述
db2start	启动当前实例
db2stop	停止当前实例
db2icrt	创建一个新的实例
db2idrop	删除一个实例
db2ilist	显示系统您当前的所有实例清单
db2 get instance	显示当前运行的实例

表 4.1 – 实例层上常用的 DB2 命令

以上一些命令也可以通过控制中心（Control Center）来执行。例如，在控制中心，展开实例文件夹（*Instances*），右键单击所要操作的实例，接着选择 *Start* 来启动该实例，这一操作就等同于在 DB2 命令窗口中执行 db2start 命令；或者选择 *Stop* 来停止该实例，这一操作则等同于在 DB2 命令窗口中执行 db2stop 命令，如图 4.5 所示。

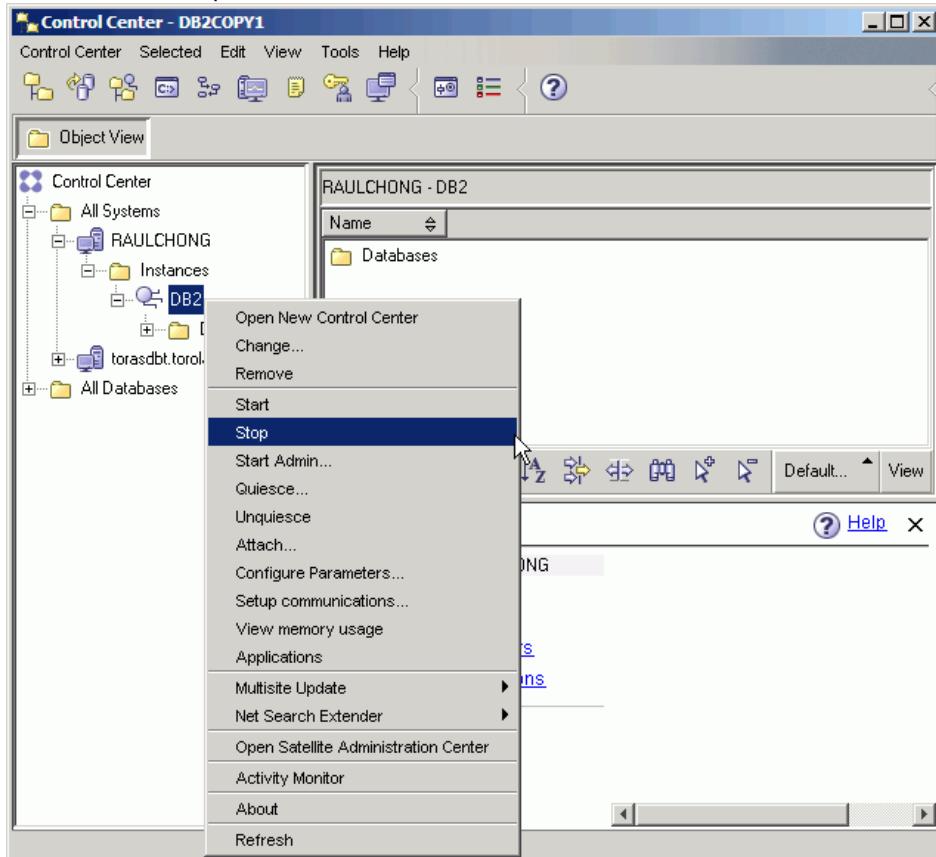


图 4.5 – 控制中心的实例命令

在当前运行实例中创建一个数据库，可以在 DB2 命令窗口执行如下命令：

```
db2 create database mydb1
```

若要显示所有已创建的数据库可以执行如下命令：

```
db2 list db directory
```

在任一实例中都可以创建多个数据库。数据库是诸如表，视图，索引等对象的集合。数据库之间是相互独立的单元，因此一个数据库并不与其他数据库共享内部对象。图 4.6 显示了在“DB2”实例中创建的数据库“MYDB1”。

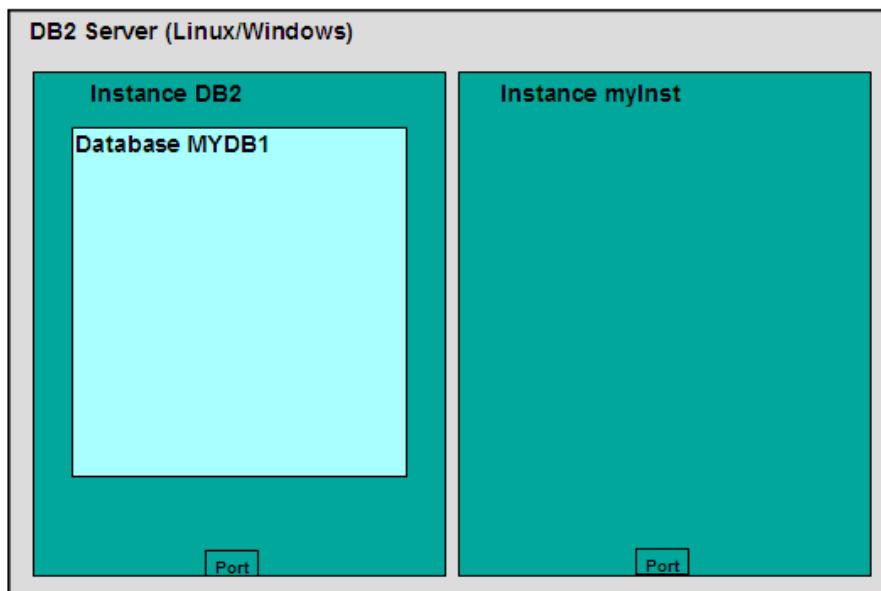


图 4.6 – 在实例 “DB2” 中创建的数据库 “MYDB1”。

表 4.2 显示了一些应用于数据库层的命令。

命令/SQL 语句	描述
db2 create database	创建一个新的数据库
db2 drop database	删除一个数据库
db2 connect to <database_name>	连接数据库
db2 create table/create view/create index	分别创建表, 视图, 和索引

表 4.2 – 数据库层的命令/SQL 语句。

如果想要在 “myinst” 实例中创建相同名称 (MYDB1) 的数据库, 可以在 DB2 命令窗口执行如下的命令:

```
db2 list db directory
set db2instance=myinst
db2 create database mydb1
set db2instance=db2
```

图 4.7 展示了在 “myinst” 实例中创建的新数据库 “MYDB1”。

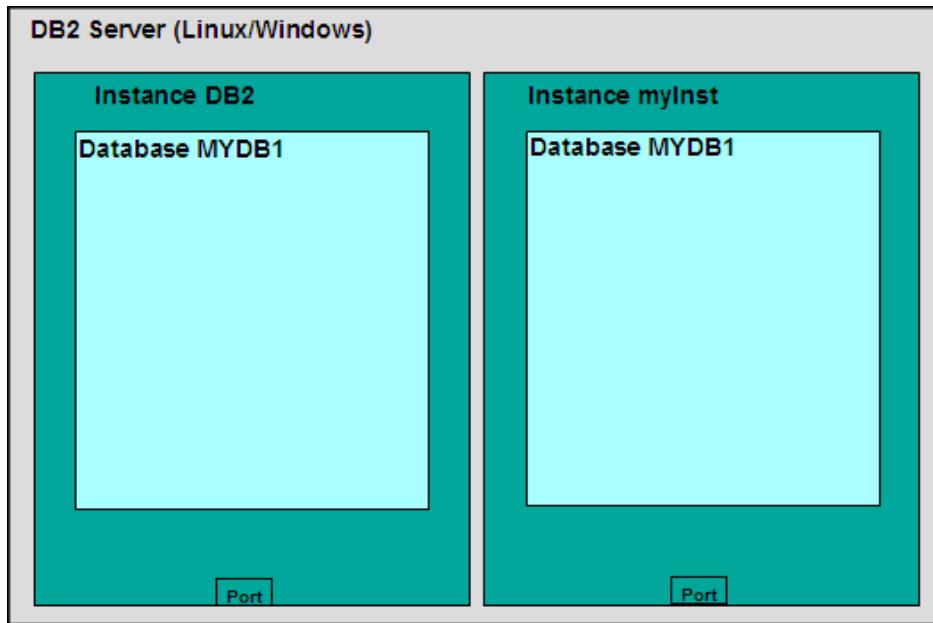


图 4.7 – 在“myInst”实例中创建的数据库“MYDB1”。

随着数据库的创建，有几个默认的对象也同时被创建：表空间，表，缓冲池，日志文件。因为创建这些对象需要一点时间，所以执行数据库创建过程需要几分钟。图 4.8 显示了三个默认被创建的表空间。在第 6 章 DB2 体系结构中会详细介绍表空间的相关细节，现在，可以把表空间看成是处于逻辑表和物理资源之间（如类似硬盘、内存等）的逻辑层。

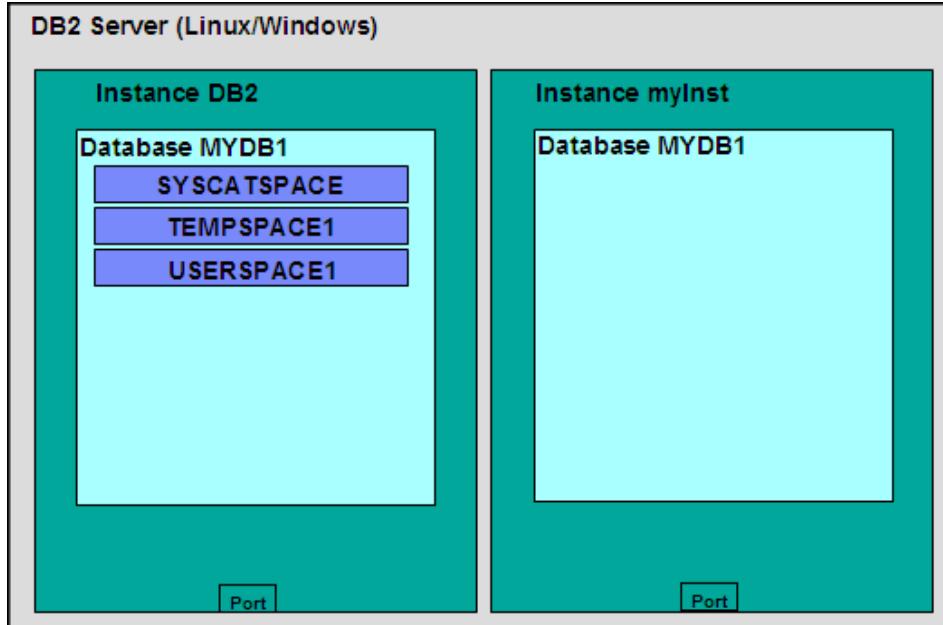


图 4.8 – 当创建一个数据库时默认被创建的表空间

SYSCATSPACE 表空间包含目录表（Catalog）。在于其他关系数据库管理系统中，这个目录被叫做数据字典。它包含的系统信息不可更改和不可删除，否则数据库无法正常工作。当 DB2 实现一些类似排序等需要额外空间的操作时，就会用到表空间 TEMPSPACE1。如果创建一个表的时候没有指定空间，系统通常会使用 USERSPACE1 来存储您的数据库表。

您也可以使用 `CREATE TABLESPACE` 语句创建自己的表空间。图 4.9 显示了一个表空间 `MYTBL1`，它创建在 `DB2` 实例的 `MYDB1` 数据库中。您在创建表空间时需要指定所用的硬盘和内存（缓冲池）。因此，如果您的某个表使用频繁，那么您可以将此表定位在具有最快的硬盘和最大内存的表空间上。

图 4.9 展示了另两个被默认创建的对象：名为 `IBMDEFAULTTBP` 的缓冲池和日志文件。

缓冲池是数据库使用的高速缓冲存储器。您可以创建多个缓冲池，但是至少应该有一个缓冲池，它的页的大小与现存的表空间的页的大小相同。数据页和页的大小将在第六章 `DB2` 体系结构中详细介绍。

日志文件用于恢复操作。数据库运行过程中，不仅仅把数据库信息存储到硬盘上，而且日志文件也会存储所有针对数据的操作。可以把日志文件看成是一个在“`autosave`”操作产生的临时文件。第十一章（备份和恢复）将对日志加以详细讨论。

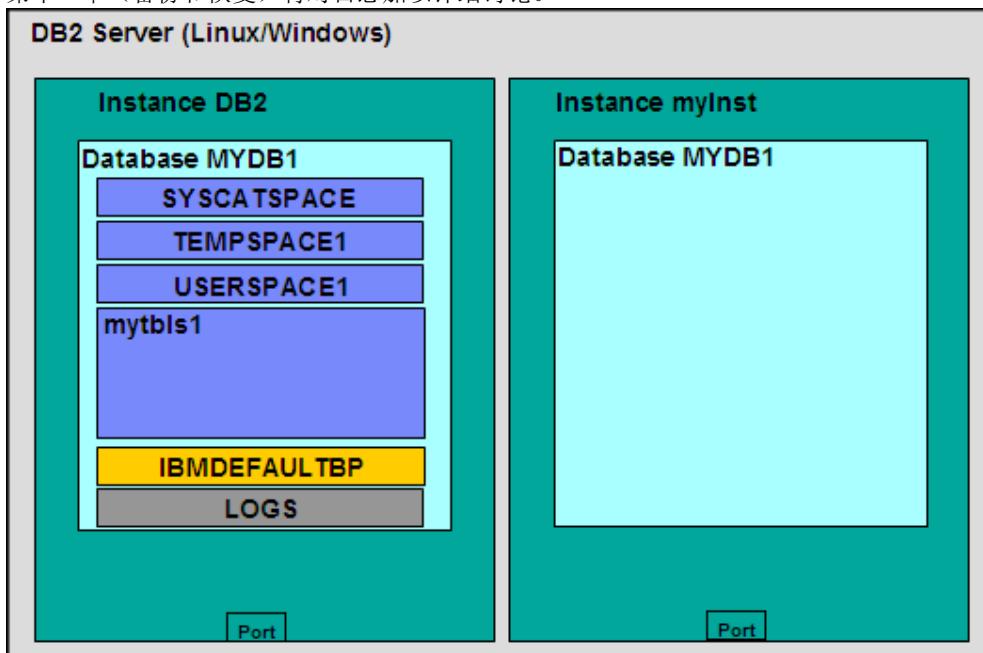


图 4.9 – 默认创建的缓冲池和日志

前面我们讨论了实例间是相互独立的，因此，几个不同实例内可以创建同名数据库。与实例类似，数据库之间也是相对独立的单元，所以，一个数据库内的对象和另一个数据库内的对象毫无关系。图 4.10 展示了存在于实例 `DB2` 的两个数据库 `MYDB1` 和 `SAMPLE` 中同名为“`mytbls1`”的表空间。注意由于受图片空间大小限制图 4.10 并没有显示数据库 `SAMPLE` 中其他被默认创建的对象。

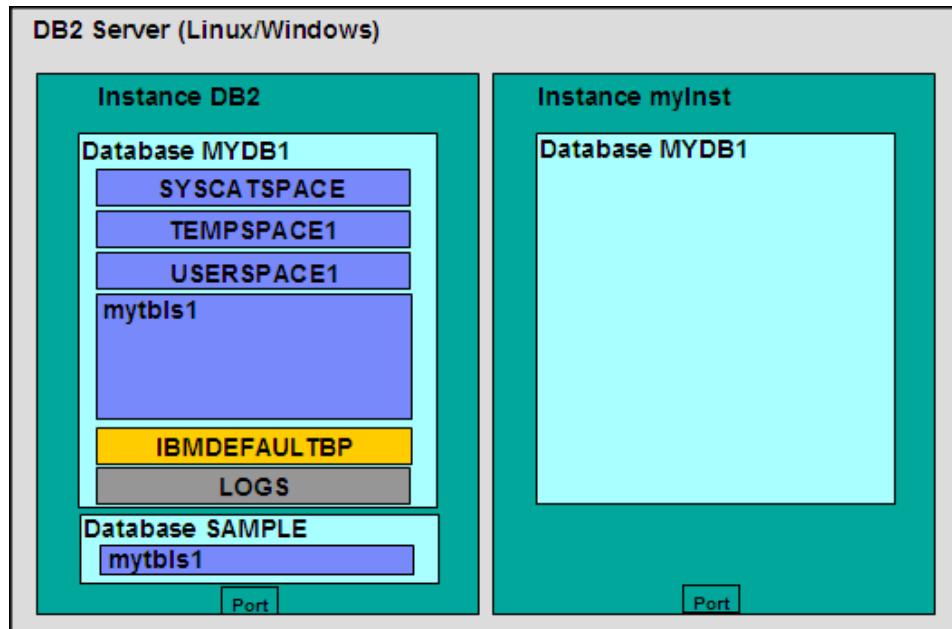


图 4.10 – 不同数据库的同名表空间

当您创建了表空间后，就可以在表空间中创建其它数据库对象，例如数据表，视图和索引。如图 4.11 所示。

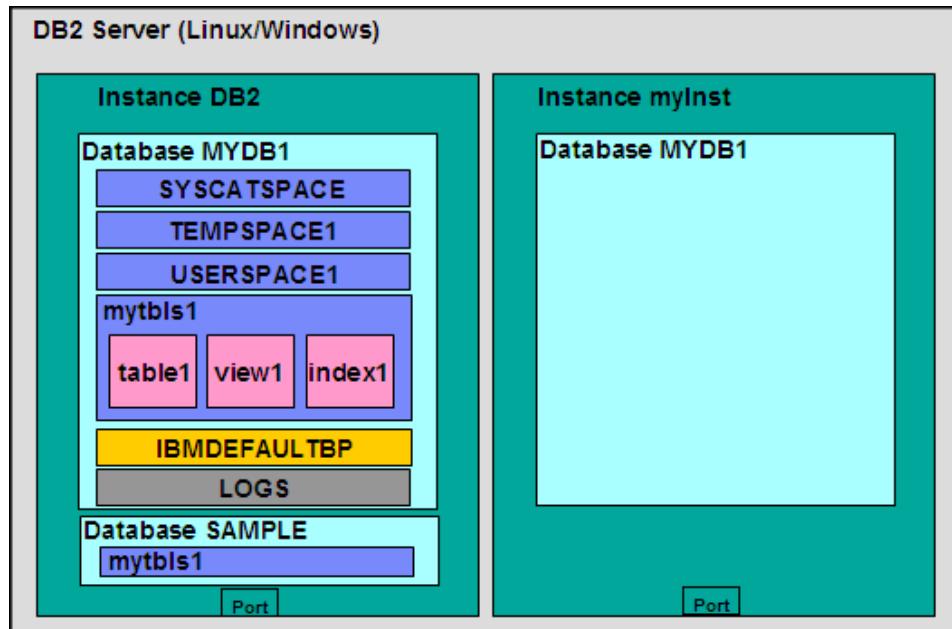


图 4.11 – 在表空间中创建的表，视图和索引

4.1 DB2 配置

使用配置向导工具（Configuration Advisor Tool）可以设置 DB2 参数。在控制中心右键单击数据库并选择“Configuration Advisor”，根据您对系统资源和工作载荷的描述，配置向导会提供一个 DB2 推荐参数列表，您可以阅读它们来获得关于 DB2 配置的细节信息，也可直接使用配置向导提供的数值。

一个 DB2 服务器可以在四个不同层面上加以配置：

- 环境变量
- 数据库管理器配置文件（dbm cfg）
- 数据库配置文件（db cfg）
- DB2 概要文件注册表（db2 profile registry）

如图 4.12 所示。注意图中每个方框所处的不同位置。例如，环境变量是在服务器的操作系统层设置的，数据库管理器配置文件变量是在实例层设置的。数据库配置变量是在数据库层设置的，DB2 概要文件注册表则可以在操作系统层或实例层设置。

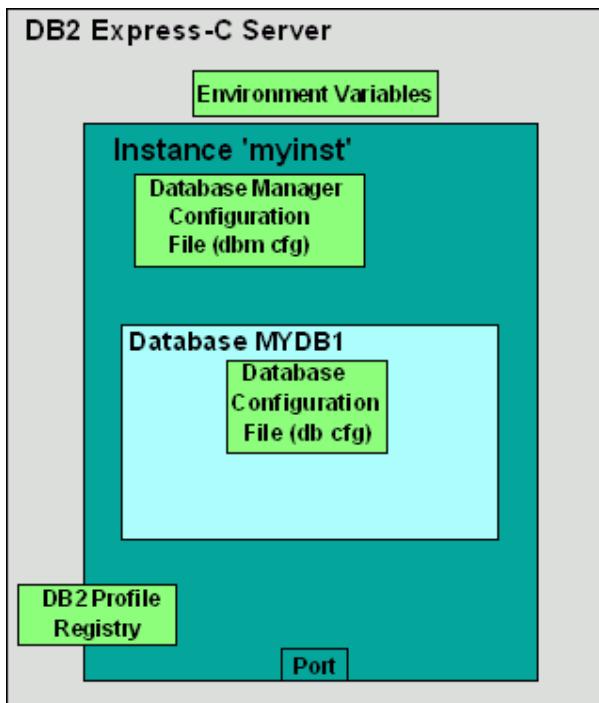


图 4.12 – DB2 配置

4.1.1 环境变量

环境变量是在操作系统层设置的变量。DB2INSTANCE 是其中一个主要的环境变量。这个变量显示了当前活动的实例——即您的 DB2 命令对之执行操作的实例。例如，在命令窗口中设置一个活动的实例“myinst”，您可以运行如下操作系统命令：set db2instance=myinst

4.1.2 数据库管理器配置文件 (dbm cfg)

数据库管理器配置文件 (dbm cfg) 包含一些参数，这些参数影响对应的实例和及其数据库。您可以通过命令行或者 DB2 控制中心查看和修改数据库管理器配置文件。

想要从控制中心启动 DBM CFG，可以在控制中心实例 (instance) 文件夹内选择实例对象，然后右键单击，在弹出菜单上选择配置参数 (Configure Parameters)。如图 4.13 所示。

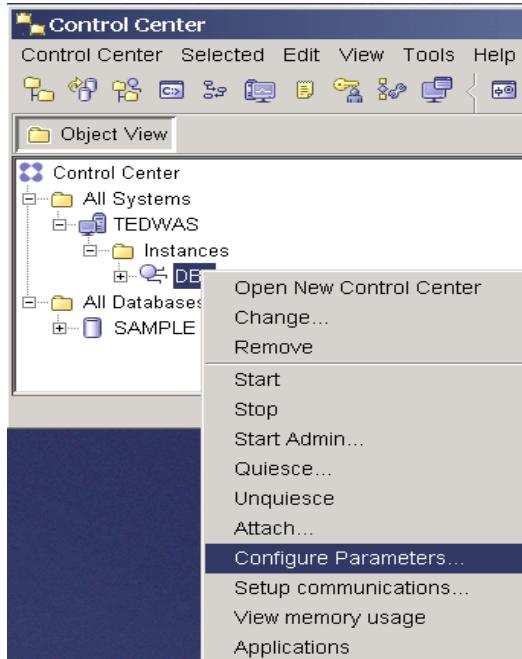


图 4.13 – 在控制中心配置 dbm cfg

在选择配置参数 (Configure Parameters) 之后，屏幕会显示 dbm cfg 参数列表，如图 4.14 所示。

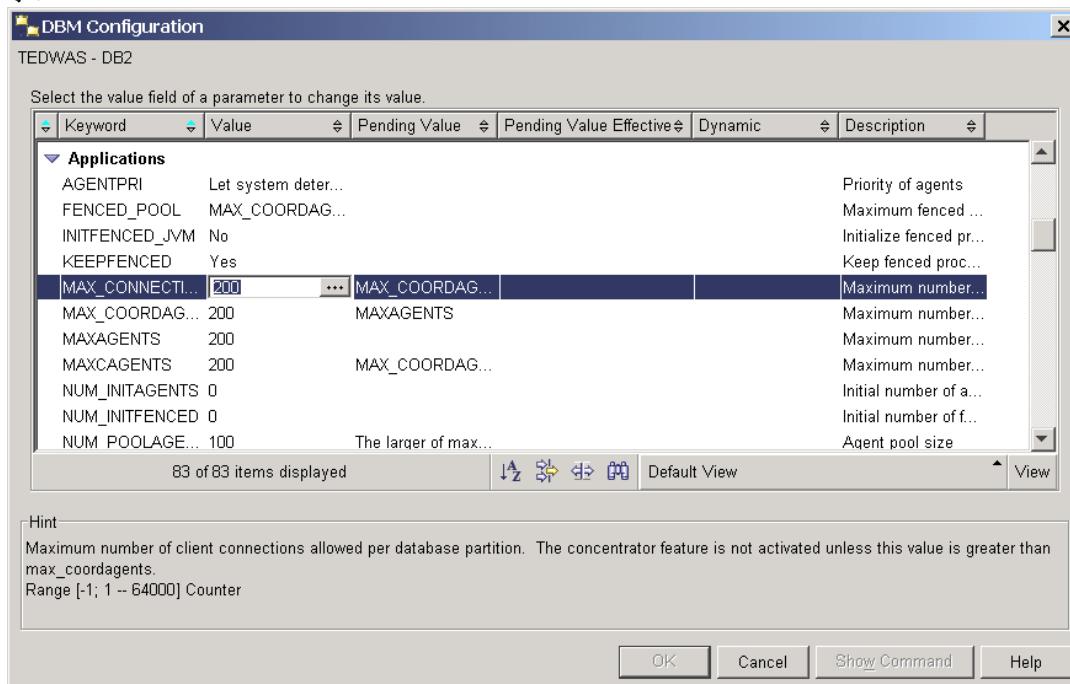


图 4.14 – dbm cfg 对话框

很多参数都是动态的，立即生效；但是，有一些参数的修改需要重启实例。可以在命令行使用 db2stop 和 db2start 命令来实现。

在停止一个实例之前，必须断开所有应用与实例的连接。如果您希望强制停止实例，可以使用 db2stop force 命令。

还可以通过控制中心来停止和启动实例——单击实例对象选择 Stop 或 Start 即可。

表 4.3 显示了一些用命令行管理 dbm cfg 的命令。

命令	描述
db2 get dbm cfg	得到 dbm cfg 的信息
db2 update dbm cfg using <parameter_name> <value>	更新 dbm cfg 的参数信息

表 4.3 – 操作 dbm cfg 的命令

4.1.3 数据库配置文件 (db cfg)

数据库配置文件 (db cfg) 包含影响对应数据库的参数。数据库配置文件也可以通过命令行或者控制中心来查看或者修改。

从控制中心启动 DB CFG，可以在数据库 (database) 文件夹中选择数据库对象，并右键单击弹出菜单，在菜单中选择配置参数 (Configure Parameters)。如图 4.15 所示。

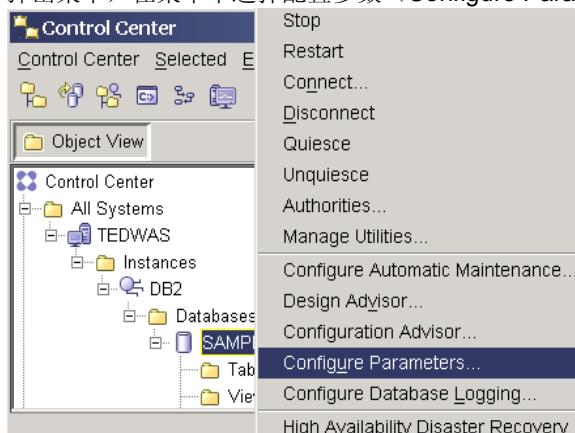


图 4.15 – 通过控制中心配置 db cfg。

选择配置参数后 (Configure Parameters)，屏幕会显示一个 db cfg 参数的清单。如图 4.16 所示。

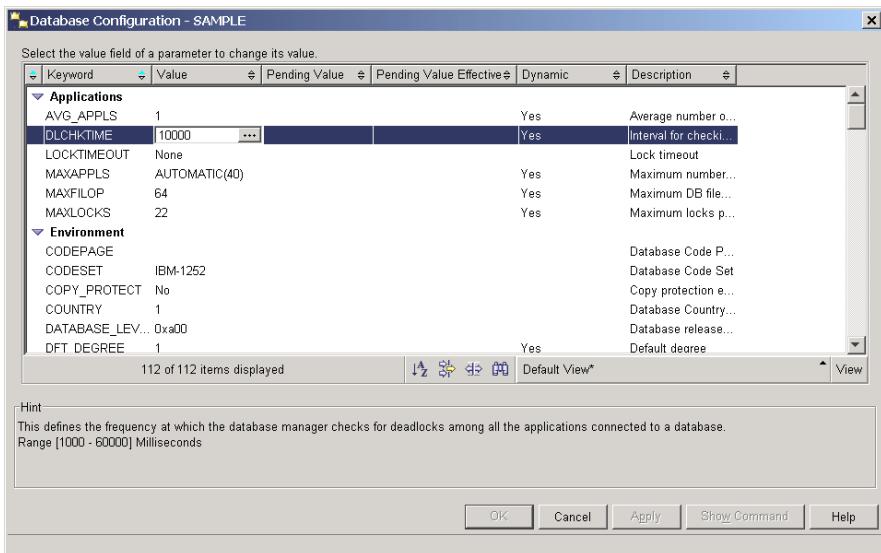


图 4.16 – db cfg

表 4.4 包含了命令行中使用的管理 db cfg 的命令。

命令	描述
get db cfg for <database_name>	得到指定数据库的 db cfg 信息
update db cfg for <database_name> using <parameter_name> <value>	更新 db cfg 参数的值

图 4.4 – 操作 db cfg 的命令

4.1.4 DB2 概要文件注册表

DB2 概要文件注册表包含了与平台相关的全局（影响所有实例）或者实例层次（只影响某个实例）的参数。

表 4.5 显示了操作 DB2 概要文件注册表的一些命令。

命令	描述
db2set –all	列表显示当前设置的所有 DB2 概要文件注册表变量
db2set –lr	列表显示所有 DB2 概要文件注册表变量
db2set <parameter>=<value>	把一个参数设置为指定值

表 4.5 - 操作 DB2 概要文件注册表的一些命令

表 4.6 展示一些最常用的 DB2 注册表变量。

Registry Variable	Description
DB2COMM	指定数据库管理器启动后的通讯管理器。
DB2_EXTSECURITY	Windows 操作系统上，通过锁住 DB2 系统文件来阻止无权限用户对 DB2 的访问
DB2_COPY_NAME	存储当前使用的 DB2 副本的名称。

要转换到不同的 DB2 副本，请运行 `installpath\bin\db2envvars.bat` 命令进行转换。本变量不能被用于这种转换副本的目的。

表 4.6 – 经常使用的 DB2 概要文件注册表变量

例如，若想用 TCPIP 协议实现通讯，可以将 DB2COMM 注册表变量用命令设置为 TCPIP，如下所示：

```
db2set db2comm=tcpip
```

4.2 DB2 管理服务器（不建议）

DB2 管理服务器（DAS）是一个运行在 DB2 服务器上的守护进程，此进程允许远程您连接并通过图形化管理器管理 DB2 服务器。每一台计算机只有一个 DAS，如图 4.16 所示。

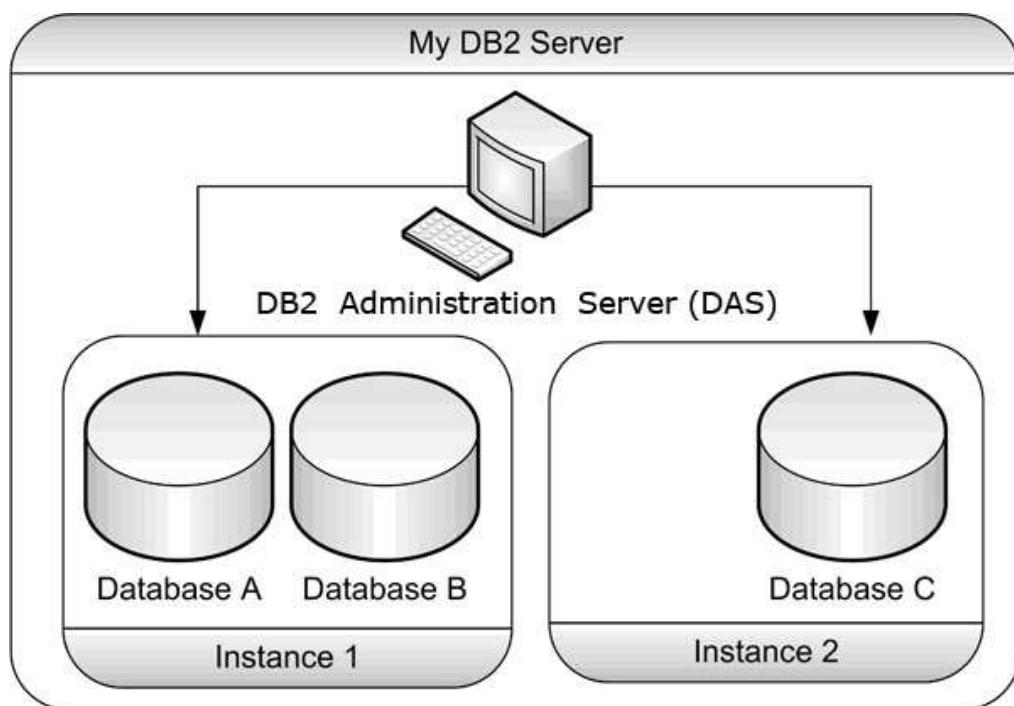


图 4.16 – DB2 管理服务器（DAS）

4.3 小结

在这一章中，我们探讨了 **DB2** 环境涵盖的概念和实例，以及数据库的建立，还有其常用命令。我们知道有关实例的其他重要方面，包括表空间；表空间的三个组成包括表，视图以及索引；缓冲池；日志记录。

最后，我们讨论了 **DB2** 的配置结构以及通过以下四方面对它进行改变：环境变量；数据库管理配置文件；数据库配置文件；**DB2** 配置文件注册表。

4.4 实验

本节中的实验将让您理解本章中讨论的概念，并介绍一些工具给您。

第一部分 – 使用创建数据库向导创建一个新的数据库

本实验中您会在控制中心用创建数据库向导（**Create Database wizard**）创建一个新的数据库。

实验过程

1. 在控制中心左侧对象树状面板中，右键点击 **All Databases** 文件夹，选择 **Create Database** 项，然后选择自动维护 **With Automatic Maintenance** 项。运行“创建数据库向导 **Create Database Wizard**”。
2. 在向导的 **Name** 页面上指定数据库名和存储位置。可以使用如下值：
 数据库名： **EXPRESS**
 默认驱动器（Windows）： **C**
 默认路径（Linux）： **/home/db2inst1**
 别名： 如果置空则默认规定为 **EXPRESS**
 备注： 此项可选且可为空

然后点 **Next** 按钮进入到下一页。



注：在 Windows 上，默认情况下只能在驱动器上而不能在路径上创建数据库。如果您想在一个路径上创建一个数据库，请设置 DB2 注册表变量 **DB2_CREATE_DB_ON_PATH**。

3. 在 **Storage** 页，不要做任何改动，直接点 **Next**。
4. 在 **Maintenance** 页，保留默认（**Yes, I can specify an offline ...**），然后点 **Next**。
5. 在向导的 **Timing** 页制定离线维护时间，设置为周一至周四每天早上 **1: 00am** 起的 6 小时。然后点 **Next** 按钮继续。
6. 在向导的 **Mail Server** 页面配置通知。如果 **DB2** 监测到问题或者异常情况，它可以自动发送邮件或者呼叫您。如果想配置此项，则需为 **DB2** 提供可用的 **SMTP** 服务器。在本实验中，我们没有 **SMTP** 服务器，所以此项置空并单击 **Next**。

在向导的 **Summary** 页面复查所有的已选项。点击 **Finish** 按钮开始数据库创建进程。数据库创建通常要花费几分钟时间，这期间可以看到执行的进度条。

第二部分 – 实例、数据库和配置管理

实验目的

在本实验中，您将在 Windows 系统下创建一个新的实例和一个新的数据库，然后更改 DB2 服务器的配置参数。您可以通过控制中心或者命令窗口来实现这些操作。在这里我们以使用命令窗口为例。

实验过程

1. 打开命令窗口： Start -> Programs -> IBM DB2 -> DB2COPY1（默认值）-> Command Line Tools -> Command Window。
也可以这样操作： Start -> Run， 键入 db2cmd。

2. 从命令窗口创建一个名为 *newinst* 的新实例
`db2icrt newinst`

3. 切换到 *newinst* 实例，并验证它确实是您的当前实例。然后启动它。
`set db2instance=newinst`（注意等号=前后不能有空格）
`db2 get instance`（确保了 *newinst* 就是当前实例）
`db2start`

4. 在实例 *newinst* 中创建一个默认值的数据库 *newdb*。这需要几分钟，因为 DB2 在数据库中创建内部对象，并提供一些初始配置。

`db2 create database newdb`

5. 连接和断开新的数据库 *newdb*。然后，删除它。

`db2 connect to newdb`
`db2 terminate`
`db2 drop db newdb`

6. 停止当前实例 *newinst*
`db2stop`

7. 列出所有在您的服务器上的实例
`db2ilist`

8. 切换到 DB2 实例，然后确认切换成功
`set db2instance=db2`
`db2 get instance`

9. 删除实例 *newinst*
`db2idrop newinst`

10. 找出 `dbm cfgFEDERATED` 参数的当前值。默认情况下，它应该有一个 NO 值。

`db2 get dbm cfg`

提示：在 Linux 上，您可以这样做：`db2 get dbm cfg | grep FEDERATED`

11. 把 `dbm cfg FEDERATED` 参数值更改为 YES 并且验证此更改

`db2 update dbm cfg using FEDERATED YES`

由于 `FEDERATED` 不是动态参数，更改不会生效，直到您停止和重新启动实例。但是，要停止该实例，您必须确保没有连接。确保的一个方法是发出以下命令：

`db2 force applications all`

```
db2 terminate
```

重新启动实例，并验证 FEDERATED 的新值：

```
db2stop
```

```
db2start
```

```
db2 get dbm cfg
```

12. 以您在操作系统中的用户 ID 和密码，连接到 SAMPLE 数据库。

```
db2 connect to sample user <userID> using <password>
```

13. 检查有多少应用程序运行在您的当前实例中

```
db2 list applications
```

14. 打开另一个 DB2 命令窗口，不指定用户 ID 和密码再次连接到 SAMPLE 数据库。随后检查您现在有多少个连接。

```
db2 connect to sample
```

```
db2 list applications
```

15. 强制关闭了来自 DB2 命令窗口的一个连接。这提供了一个例子：一个 DBA 如何可以强行终止某一用户（可能是一个占用的系统资源的用户）的工作。

```
db2 force application (<应用程序句柄 for db2bp.exe>)
```

应用程序句柄是您想要强制执行的应用程序的一个数字或'句柄'。您从 db2 list application 的输出中获得这个数字。应用程序 db2bp.exe 代表 DB2 命令窗口。

16. 验证 DB2 命令窗口的一个连接已被迫关闭。如果您不知道哪个 DB2 命令窗口被迫关闭，那么就对两个窗口都执行下面的语句。

```
db2 select * from staff
```

DB2 命令窗口被迫关闭应返回代码 SQL1224N 错误消息。其他 DB2 命令窗口应返回您查询的输出。

17. 删除并重新创建 DAS，启动它。

```
db2admin stop
```

```
db2admin drop
```

```
db2admin create
```

```
db2admin start
```

18. 看一看注册表值 DB2COMM 的当前值。

```
db2set -all
```

19. 取消设置 DB2COMM 注册表变量，并验证这项工作已完成

```
db2set db2comm=
```

```
db2stop
```

(提示：如果您有连接，您将会在 db2stop 上得到一个错误。您应该怎么做，请参考前面的步骤解决此问题。)

```
db2start
```

```
db2set -all
```

20. 在您的实例中设置 DB2 注册表变量 DB2COMM 为 tcpip 和 npipe 并且验证新的值

```
db2set db2comm=tcpip,npipe
```

```
db2stop
```

```
db2start
```

```
db2set -all
```

21. 检查 LOGSECOND db cfg 参数的当前值，然后将其更改为值 5 并且验证新的值

```
db2 connect to sample
```

```
db2 get db cfg
```

```
db2 update db cfg using LOGSECOND 5
```

db2 get db cfg

5

第五章 DB2 工具

在这一章，我们会讨论一些您可以使用的 DB2 工具。自 DB2 9.7 起，大多数在本章介绍的工具不推荐使用，虽然他们仍然可以使用，但不会再得到加强，而且未来发布的产品可能不包含这些工具。IBM Data Studio 是这些工具的替代品。

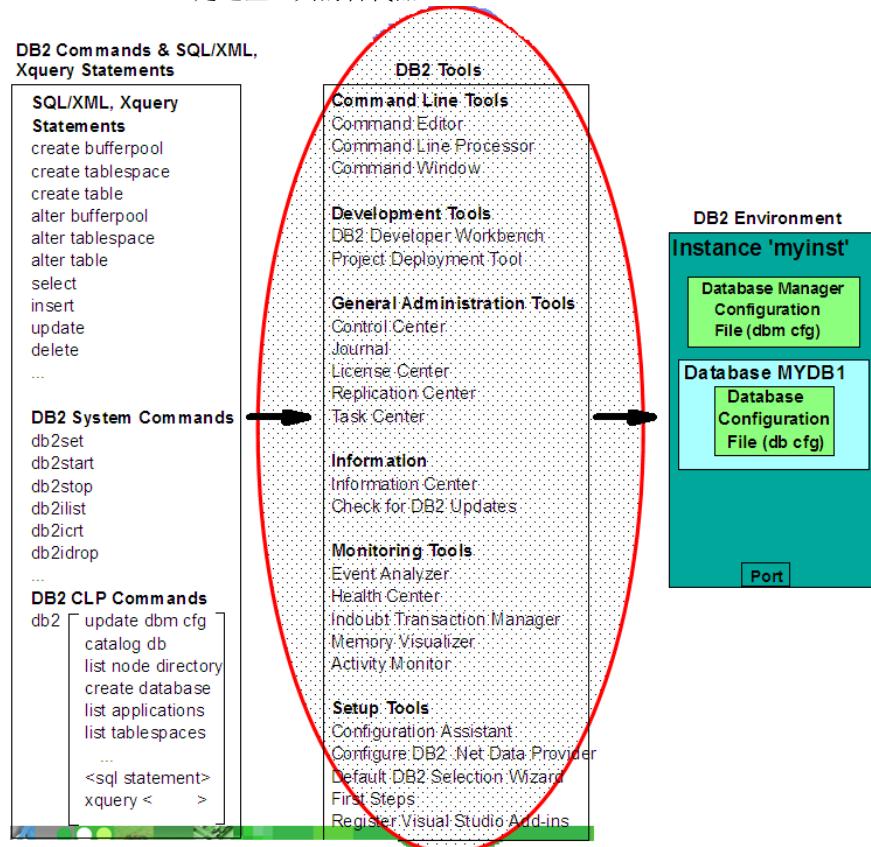


图 5.1 中用红线椭圆标识的是本章的重点内容。

注意：

更多关于 DB2 工具和脚本的信息，请参见以下视频：

<http://www.channeldb2.com/video/video/show?id=807741:Video:4202>

<http://www.channeldb2.com/video/video/show?id=807741:Video:4182>

图 5.2 列出了 IBM DB2 在开始菜单快捷方式能找到的所有工具。无论在 Linux 还是 Windows 操作系统，它们大部分都是相同的。

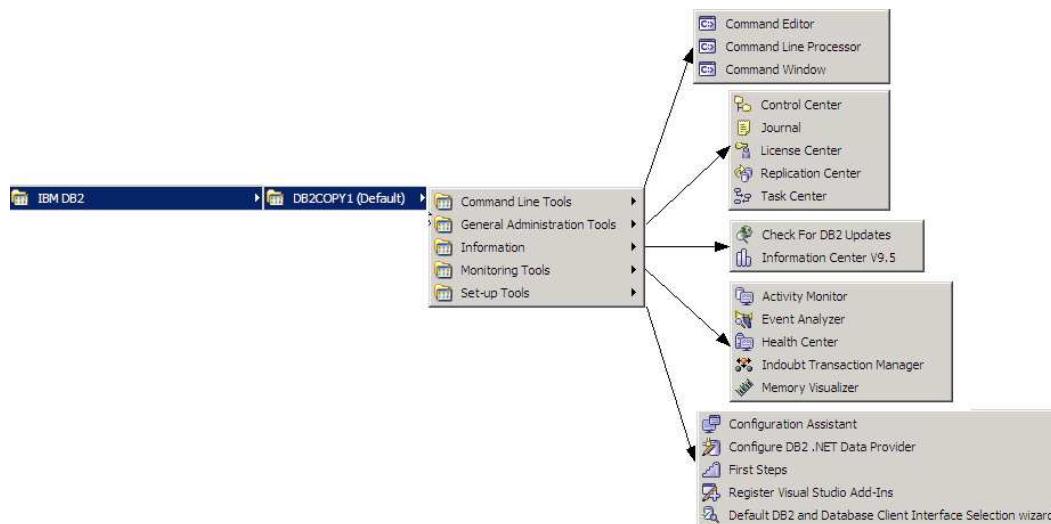


图 5.2---IBM DB2 开始菜单中的 DB2 工具

表 5.1 提供了一个快捷命令的列表，它可用于启动一些在 Linux 或 Windows 使用最频繁的工具。这张表还列出了在 DB2 9.7 中不建议的工具。

工具名称	命令	不建议的
命令编辑器 Command Editor	db2ce	是
命令行处理器 Command Line processor	db2	否
命令窗口 Command Window (仅限于 Windows 平台)	db2cmd	否
控制中心 Control Center	db2cc	是
任务中心 Task Center	db2tc	是
健康中心 Health Center	db2hc	是
配置助手 Configuration Assistant	db2ca	是
起始步骤 First Steps	db2fs	否

表 5.1 – DB2 工具对应的快捷命令

new in
V9.7

5.1 IBM 数据工作室

伴随 DB2 9.7 的发布，IBM Data Studio 已成为进行 DB2 数据库管理和开发的最主要工具。IBM Data Studio 是免费的。它可以在 Linux 和 Windows 上运行，并且是 IBM 的 Integrated Management Data 产品线的一部分。IBM Data Studio 开发的时间表与 DB2 的发布时间并没多大关系，但这两个产品的发行时间会尽可能地协调一致。例如，DB2 9.7 和 IBM DATA Studio 2.2 在 2009 年 6 月的同一天发布。

图 5.3 显示了 IBM Data Studio 的界面。

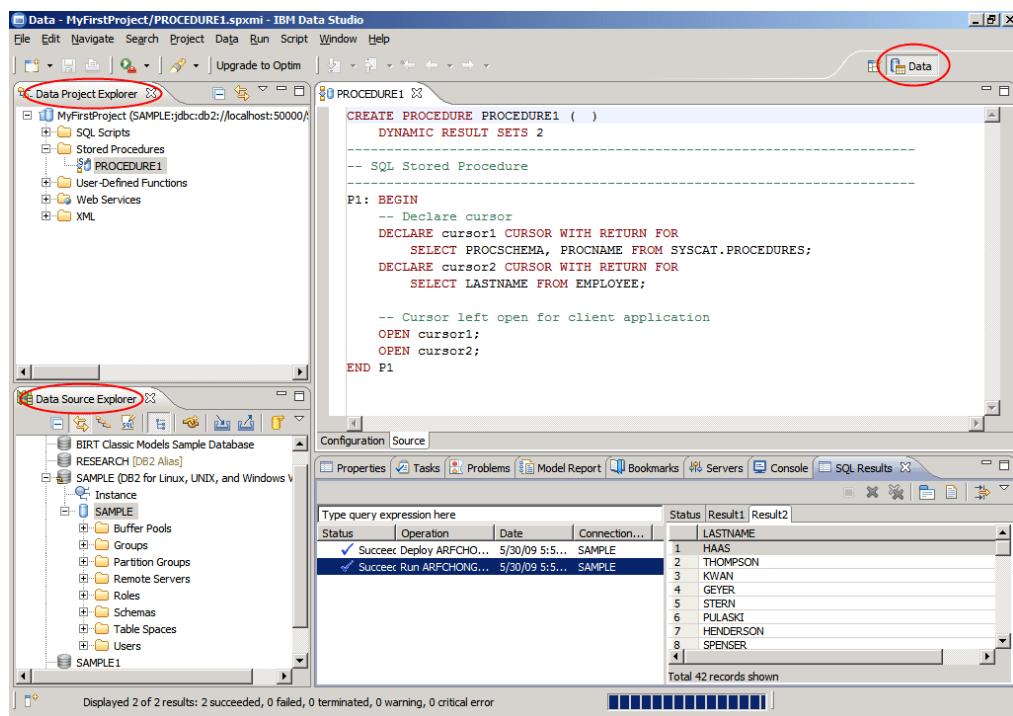


图 5.3 - IBM 数据工作室

如果您熟悉 Eclipse，您会注意到，Data Studio 是基于 Eclipse 的。使用 Data Studio 时，按照典型配置您将在数据透视窗口（在图中右上角突出的）工作。当然如果您正在开发一个 Java 程序，您还可以切换到 Java 透视图。图中有两个标出的地方：

- 1) 数据项目浏览器（左上）
- 2) 数据源浏览器（左下）

数据项目浏览器视图是用于数据库开发人员使用 SQL 脚本，XQuery，存储过程，UDFs 和数据网络服务。

数据源浏览器视图是 DB2 的数据库管理员用来管理实例和数据库。使用这个视图中，您可以执行以前可以在控制中心使用的大部分功能。

在图中，有着标题 PROCEDURE1 的视图是一个程序编辑器强调了在数据项目资源管理器。根据您正在执行的任务，编辑器或其他窗口将会出现，并让您编译代码或执行更多的配置。

IBM Data Studio 中有两种类型：

- 1) 独立包
- 2) IDE 包

独立包是比 IDE 包小很多，但不支持数据 Web 服务开发，也不能扩展到其他像 InfoSphere Data Architect 的基于 Eclipse 的 IBM 产品中。除此之外，界面和功能倒是相同的。

您可以在使用 IBM Data Studio 的同时使用其他数据服务器如 Informix。使用几种不同的数据服务并有一个小的数据库管理或数据库开发团队的公司现在能方便的用一个工具处理和管理所有问题。

注意：

有关数据 Studio 的详细信息, 请参阅入门使用 IBM Data Studio DB2 的免费电子书, 这是本系列丛书的一部分。

5.2 控制中心 (不建议)

在 DB2 9.7 之前, 主要的数据库管理工具是控制中心 (Control Center), 如图 5.4 所示。

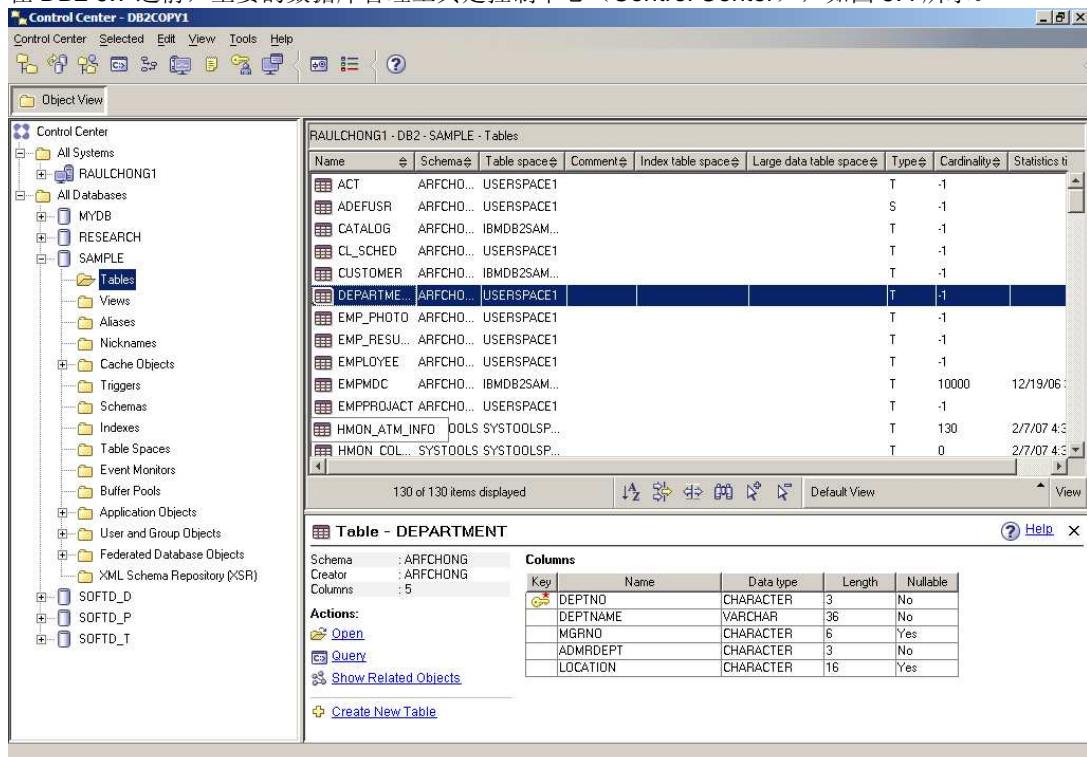


图 5.4 - DB2 控制中心

控制中心是一个集中式的管理工具, 使用它可以实现如下功能:

- 查看您的系统、实例、数据库、数据库对象
- 创建、修改、管理数据库和数据库对象
- 启动其他的图形化 DB2 工具

控制中心左边的面板提供了系统中数据库对象的层次表示, 用文件夹表示数据表或者视图等。

当您双击一个文件夹时, 例如双击 Table 文件夹, 如图 5.4 所示, 控制中心右上方的窗口中会列出所含的相关对象, 在本例中显示的在 SAMPLE 数据库中所有的表。如果您在右上的窗口中选中某一个数据表, 在右下的窗口会显示出关于这个数据表更多的详细信息。

在左边层次树的不同文件夹/对象上单击右键, 会弹出对应于这个文件夹/对象的菜单。例如, 右键单击一个数据库实例, 然后选择 “Configure parameters”, 就可以弹出窗口让您查看或者设置数据库管理配置文件。相似的, 当您右键单击一个数据库然后选择 “Configure parameters”, 就可以弹出窗口让您查看或者设置该数据库的配置文件。DB2 环境和设置参数在第五章的 DB2 环境中详细阐述。

当您第一次启动控制中心时, 您会被要求选择控制中心的视图。不同的视图会显示不同的数据库类型和对象。图 5.5 展示了控制中心视图对话框。

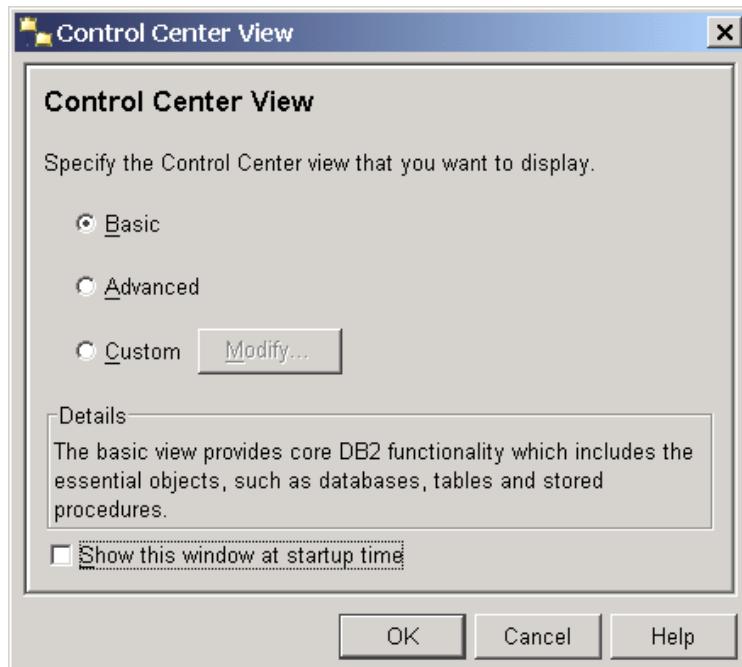


图 5.5 - DB2 控制中心视图对话框

基本视图（basic）提供了 DB2 的主要核心功能。

高级视图（advanced）提供了关于 DB2 更多的选项和特性。

自定义视图（custom）可以让您自己定义控制中心所显示的选项、对象和数据库特性。

此后，您可以在控制中心的“工具”菜单中选择 Tools->Customize Control Center 选项来弹出 DB2 控制中心视图对话框，如图 5.6 所示。

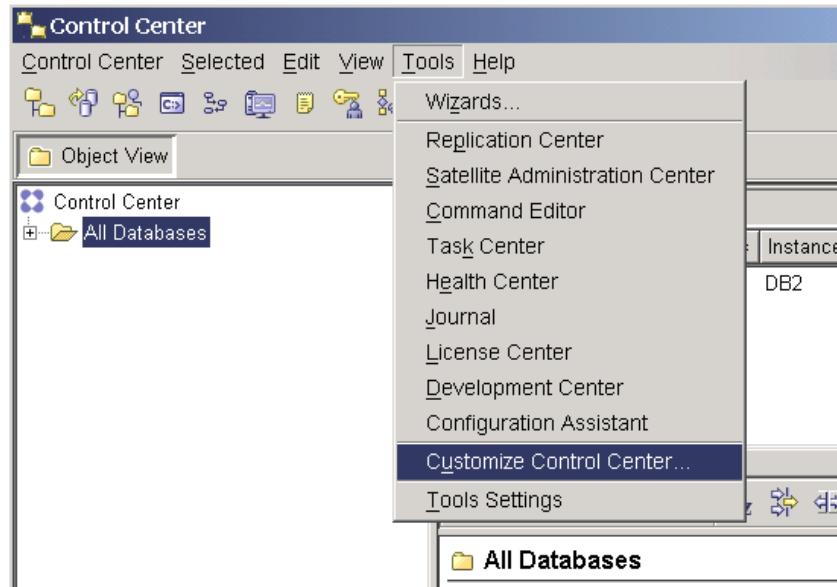


图 5.6---自定义控制中心

5.2.1 运行控制中心

有很多种方法来启动控制中心：

1. 在 Windows 开始菜单中找到控制中心的快捷方式来启动。

2. 在命令提示符中输入 db2cc 来启动。

3. 在任意的 DB2 GUI 工具中点击控制中心的图标  来启动。

4. 从 Windows 的系统任务栏的 DB2 图标中启动，如图 5.6 所示，右键单击 DB2 的绿色图标，然后在弹出的菜单中选择 DB2 Control Center 选项。

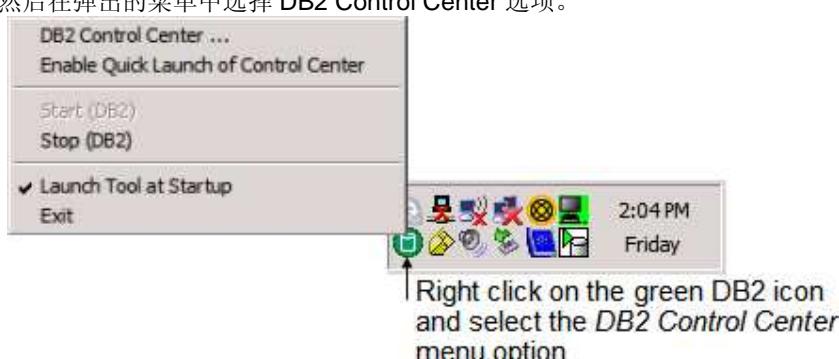


图 5.7--从 Windows 系统任务栏启动 DB2 控制中心

5.3 命令编辑器（不建议）

您可以使用 DB2 命令编辑器来执行 DB2 命令、SQL 和 XQuery 语句、分析语句的执行流程、查看或者更新查询结果集。图 5.8 展示了命令编辑器 Command Editor 并进行了解释。

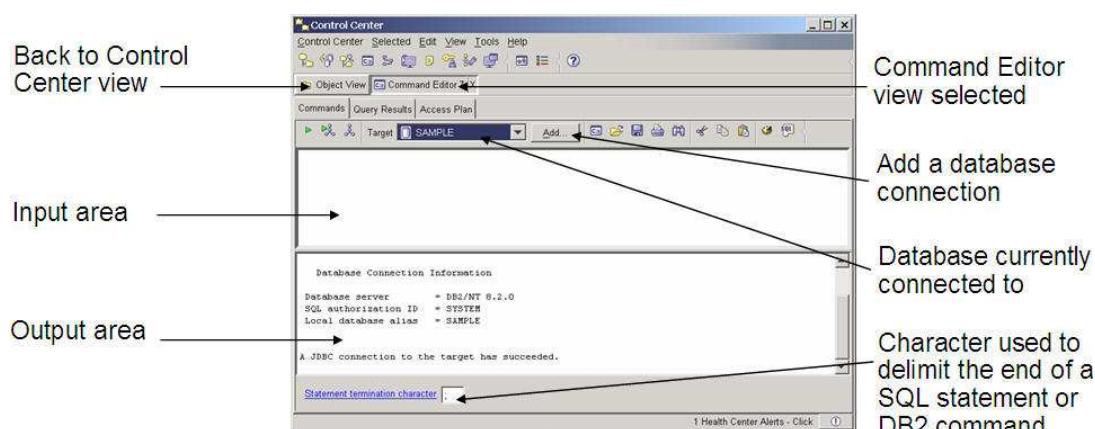


图 5.8---DB2 命令编辑器（Command Editor）

在输入框中，您可以输入若干条语句，每个语句后面使用一个终结字符结束。当您点击“执行 execute”按钮（如图 5.8 左边第一个按钮所示），语句会一条一条按顺序执行。当您高亮了某些语句时，只有选定高亮的语句会被执行。在任何 SQL 语句执行前必须连接到数据库，当然，也可以把数据库的连接语句放到要执行的 SQL 语句中。

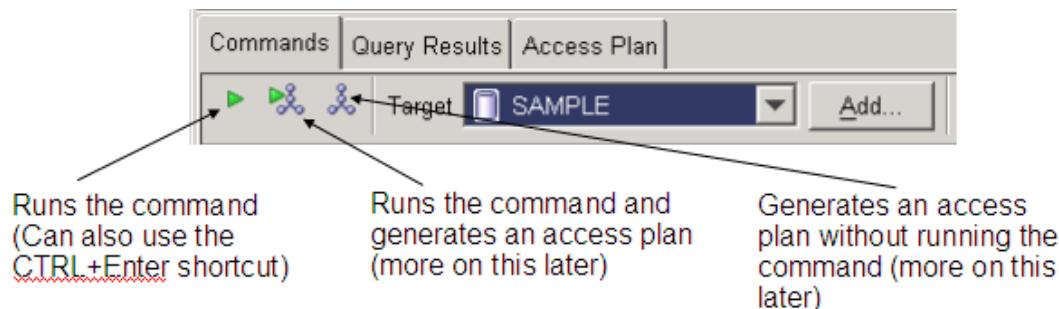


图 5.9 命令编辑器的 Commands 标签

5.3.1 运行命令编辑器

您可以通过几种方法类运行命令编辑器：

1. 从 Windows 开始菜单启动：Start -> Programs -> IBM DB2 -> DB2COPY1 (Default) ->Command Line Tools -> Command Editor
2. 在命令提示符中输入 db2ce
3. 从控制中心的“工具 tools”菜单中运行
4. 将命令编辑器嵌入到控制中心
 - 在控制中心的对象树窗口中右键单击 SAMPLE 数据库，然后选择 Query 选项。
 - 在任何时候，如果能够被查询的对象（如数据库、表、等等）被选中，您都可以单击在控制中心的对象明细 Object Detail 窗口中的 Query 链接来运行命令编辑器。
5. 在控制中心的工具栏中单击命令编辑器的图标 ，如图 5.10 所示，可以启动命令编辑器。

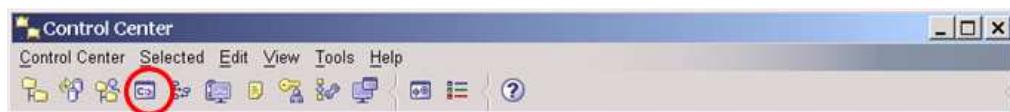


图 5.10 控制中心的命令编辑器图标

5.3.2 添加一个数据库的连接

为了给连接到一个数据库，单击命令编辑器的 Add 按钮（请查看图 5.7），会出现图 5.10 的对话框。

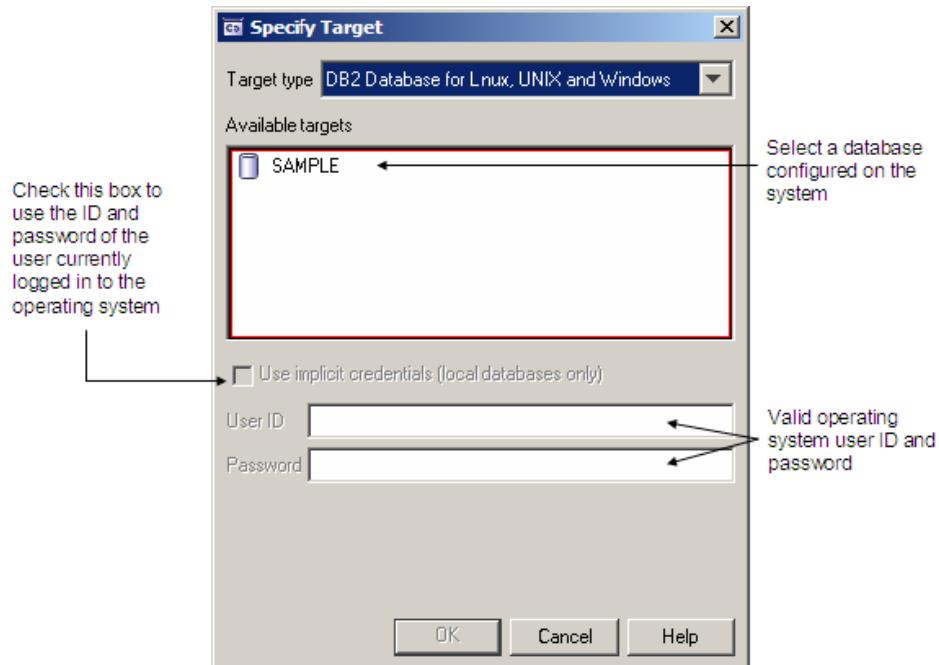


图 5.11 添加一个数据库连接

5.4 SQL 帮助向导（不建议）

如果您不熟悉 SQL 语言，您可以使用 SQL 助手或者向导来生成所需要的 SQL 代码。SQL 帮助向导（SQL Assist Wizard）可以在命令编辑器中启动，如图 5.11 所示，您可以通过单击 Commands 标签中的工具栏最后一个有 SQL 字样的按钮来启动它。此图标将只有在连接到一数据库后才会出现。

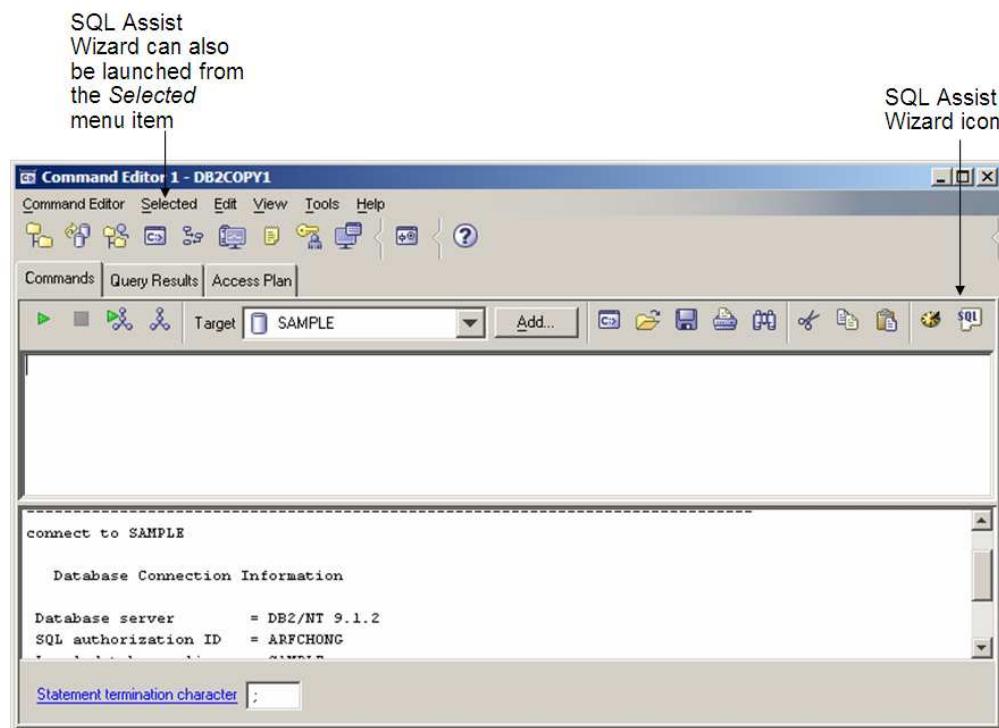


图 5.12 运行 SQL 协助向导

图 5.13 展示了 SQL 帮助向导 (SQL Assist Wizard)，它是非常简单易用的。首先指明您需要帮助的 SQL 语句类型 (SELECT, INSERT, UPDATE, DELETE)。根据您选的语句，会出现不同的选项，在向导对话框的底部窗口中，您可以看到根据您的选项而产生的 SQL 语句。

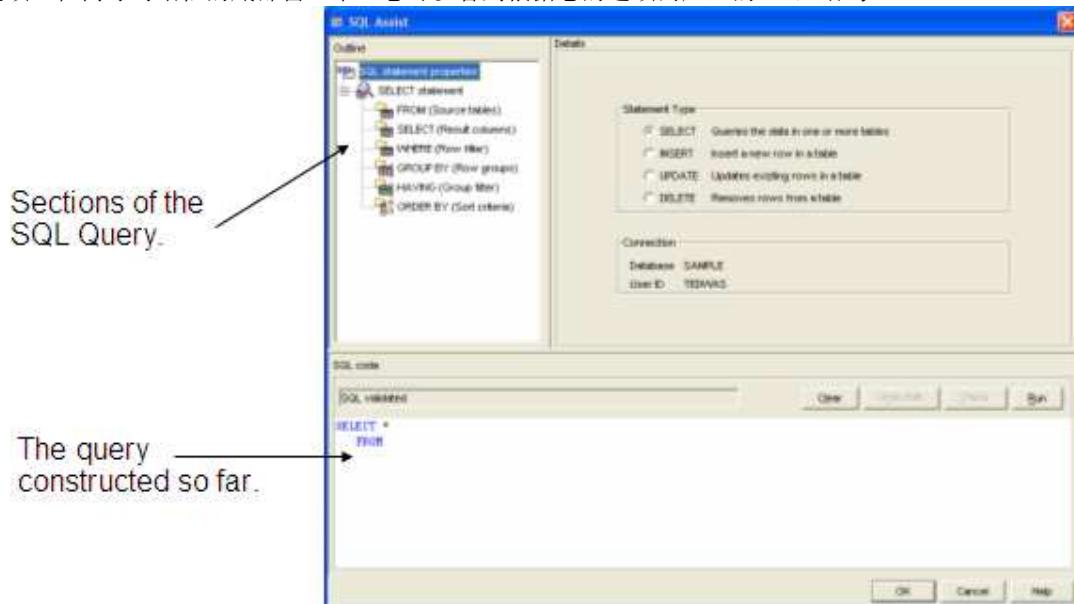


图 5.13 - SQL 帮助向导 (SQL Assist wizard)

5.5 显示 SQL 按钮（不建议）

在 DB2 中，大多数的 GUI 工具或者向导允许您查看当前所产生的 SQL 语句。为了看到 SQL 语句，在当前使用的工具中单击显示 SQL（Show SQL）按钮，如图 5.14 和 5.15 所示。



图 5.14 显示 SQL（Show SQL）按钮

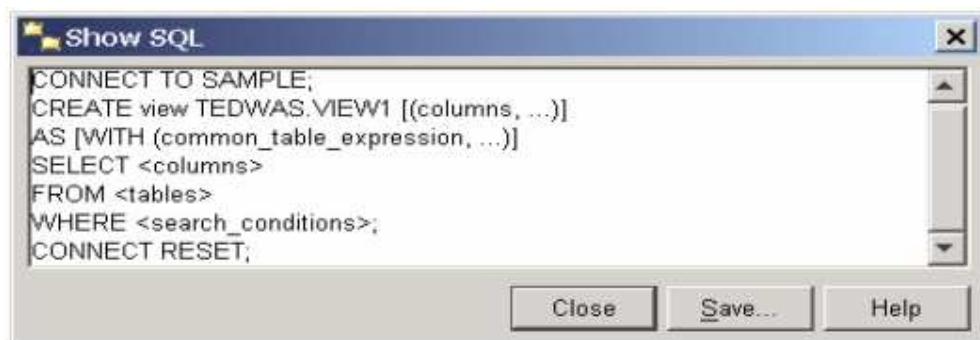


图 5.15 显示 SQL（Show SQL）的输出

查看 SQL 语句和命令的功能使您能够非常地方便学习 SQL 的语法，您也可以将这些命令或者语句保存下来，以便将来使用，也可以使用所生成的这些命令和语句来创建自己的脚本。

5.6 任务中心 Task Center（不建议）

GUI 的任务中心用于建立任务，任务指的是一连串操作（如运行 DB2 命令、操作系统命令或脚本）的集合。在任务成功或者失败时，任务中心会执行相应的动作。例如，有一项任务是在 3:00am 备份一个重要的数据库，如果这个任务执行成功，则向数据库管理员发送一封包含成功执行信息的电子邮件；如果备份失败，任务中心能够呼叫数据库管理员。任务中心如图 5.16 所示。

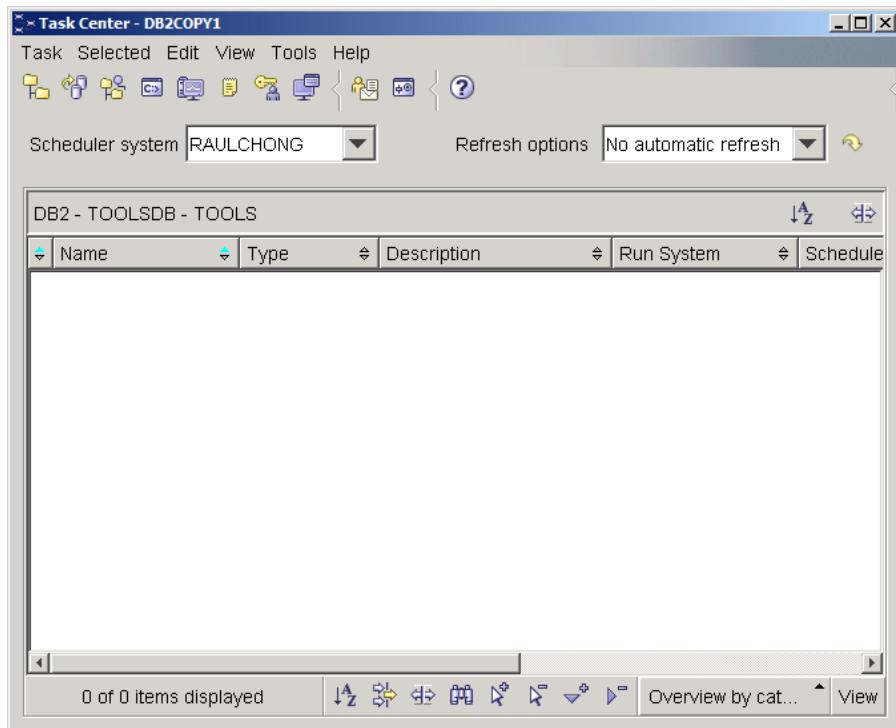


图 5.16 任务中心 (Task Center)

5.6.1 工具目录数据库 (不建议)

关于任务的所有详细信息都保存在 DB2 中叫 Tools Catalog 的数据库里。在安排任务前，这个数据库必须存在。您可以用下面的命令来创建 Tools Catalog 数据库。

```
CREATE TOOLS CATALOG systools CREATE NEW DATABASE toolsdb
```

在上面的例子中，`systools` 是数据库中所有数据表的模式名字，`toolsdb` 是数据库的名字。我们会在第 8 章对模式进行详细讲解。

5.6.1.1 运行任务中心

您可以在控制中心中启动任务中心。如图 5.16 所示 (Tools > Task Center)。同样，可以从 Windows 的开始菜单中启动任务中心 (Start -> Programs ->IBMDB2 ->DB2COPY1 (默认值) ->General Administration Tools->Task Center)。

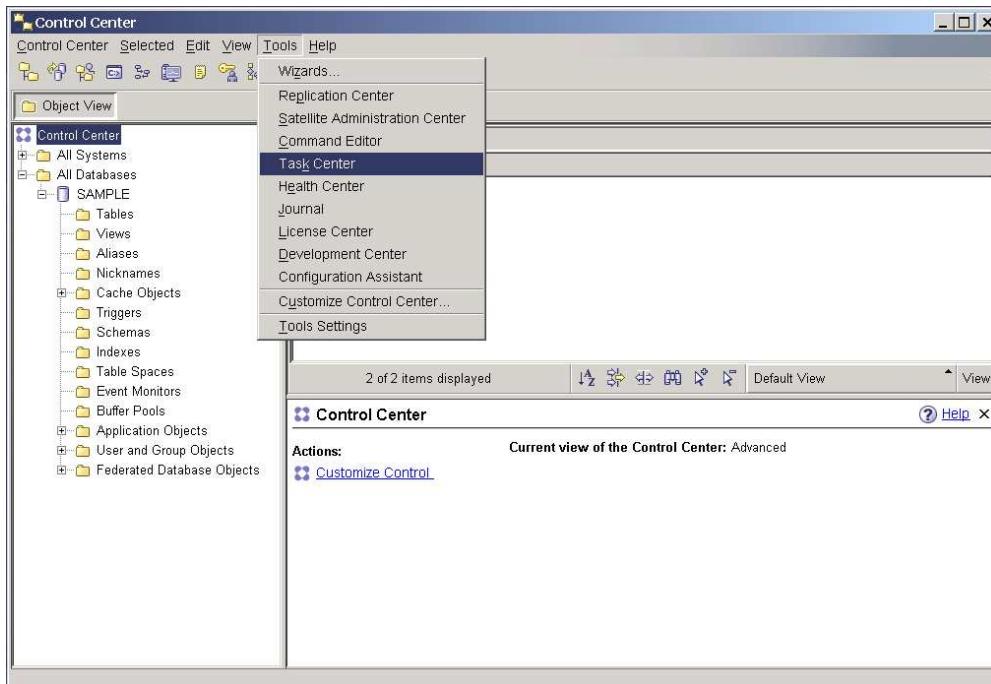


图 5.17 运行任务中心

5.6.1.2 使用任务中心来计划任务

可以使用任务中心来计划各种脚本的运行安排，无论此脚本是不是由 DB2 GUI 工具创建。各种任务按照它们的时间表来运行。我们建议您自行研究一下任务中心，其实建立一项任务是非常简单明了的。

5.7 日志（不建议）

DB2 GUI 的日志工具以在线的形式为管理员提供了数据库各种活动的日志。图 5.18 展示了日志工具，表 5.2 展示了您能够从日志工具获得的各种信息。

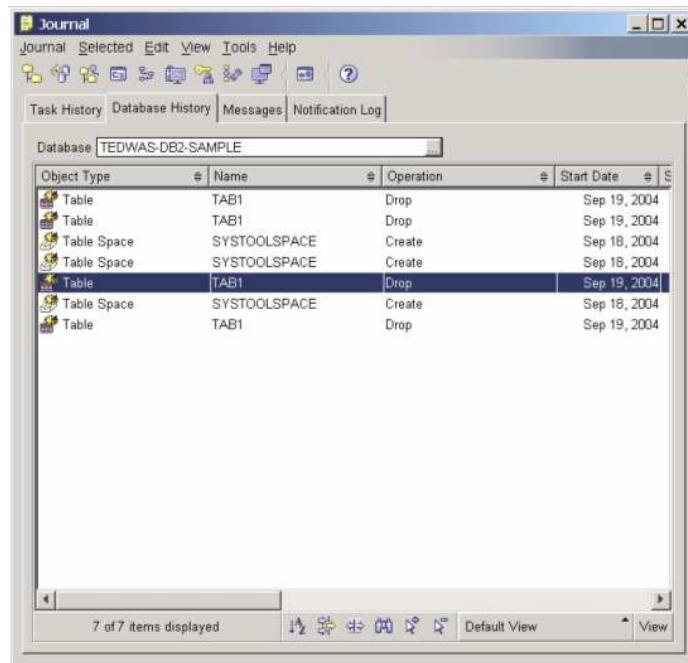


图 5.18 日志

信息类型	描述
Task History	所有执行的计划任务和它们的完成情况
Database History	数据库所有活动的记录，如备份、恢复、重组等等
Message	保存 DB2 工具返回的信息，这是非常有用的。如果您想重新执行某项操作并比较新旧的出错信息，或者有意无意关闭了一个对话框而您还来不及看清里面的信息，您可以在这里找到这些信息。
Notification Log	包含系统级别的信息，严重错误信息记录在这里

表 5.2 - 日志工具提供的信息

5.7.1 运行日志工具

您可以在控制中心中点击 Tools > Journal 来启动日志。如图 5.19 所示。或者，您可以从 Windows 开始菜单中启动该工具：

Start > Programs > IBM DB2 > DB2COPY1 > General Administration Tools > Journal

您可以在控制中心中启动日志工具。如图 5.18 所示（Tools > Journal）。同样，可以从 Windows 的开始菜单中启动日志工具（Start -> Programs -> IBM DB2 -> DB2COPY1(Default) -> General Administration Tools -> Journal）。

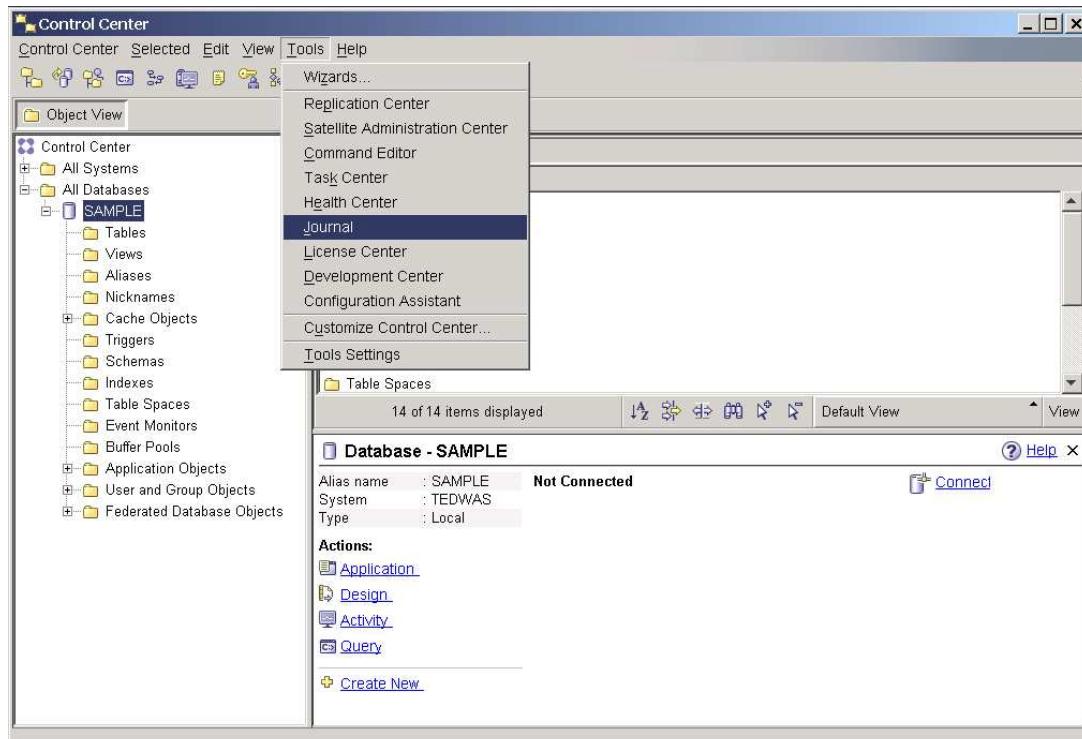


图 5.19 运行日志工具

5.8 运行状况监视器（不建议）

运行状况监视器（Health Monitor）是在 DB2 引擎中默认的状况监视工具，它全方位监控数据库的健康状况（包括内存、磁盘空间、预定义的活动等等）。如果某些方面超出健康阈值，则会提示异常引起数据库管理员的注意。运行状况监视器有三种报警等级：

- 注意（Attention）：系统轻微不正常状态
- 警告（Warning）：一个中等严重状态，并不需要马上处理，但是整个系统性能受到影
响。
- 严重警告（Alarm）：数据库受到严重影响，需要马上进行处理。

可以使用数据库管理配置参数 **HEALTH_MON** 来打开或者关闭运行状况监视器。

5.8.1 运行状况中心（不建议）

运行状况中心是一个与运行状况监视器交互的图形化工具。它按照实例、数据库、表空间三个等级来显示数据库环境的总体状态以及当前的所有报警，如图 5.20 所示。

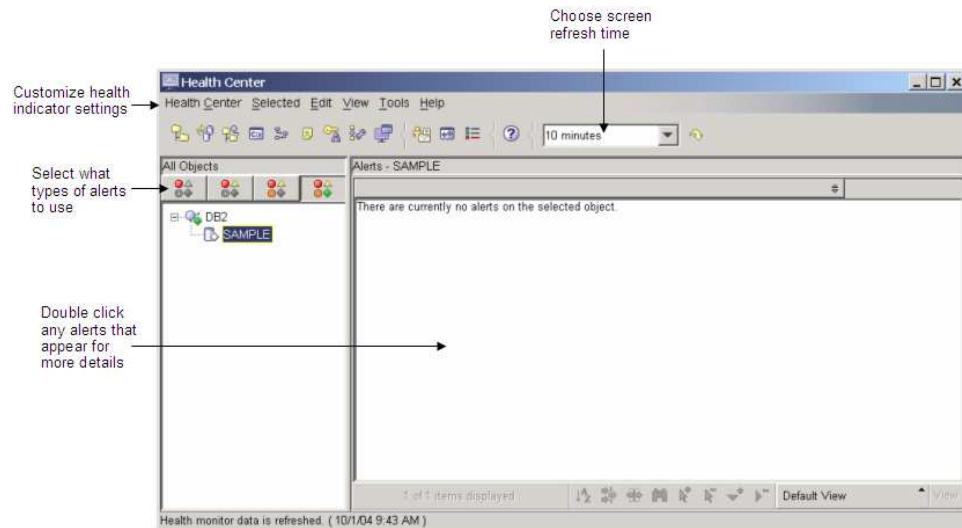


图 5.20 运行状况中心

5.8.1.1 运行运行状况中心

您可以在控制中心中启动运行状况中心，如图 5.20 所示（Tools > Health Center）。同样，可以从 Windows 的开始菜单中启动运行状况中心（Start -> Programs-> IBM DB2 -> DB2COPY1(Default) -> Monitoring Tools -> Health Center）。

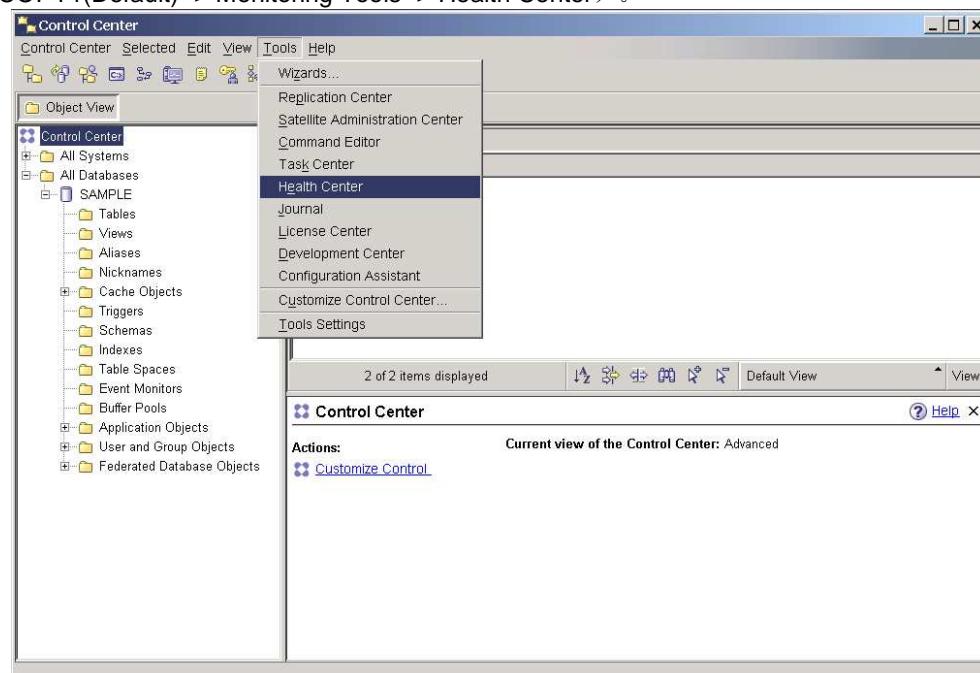


图 5.21 启动运行状态中心

5.8.1.2 配置运行状态报警通知

如果您启动了运行状态中心，您可以通过菜单 Health Center> Configure > Alert Notification 来配置报警通知，如图 5.22 所示。报警通知允许您输入出现报警时要联系的名字和 email 地址或者呼叫号码。

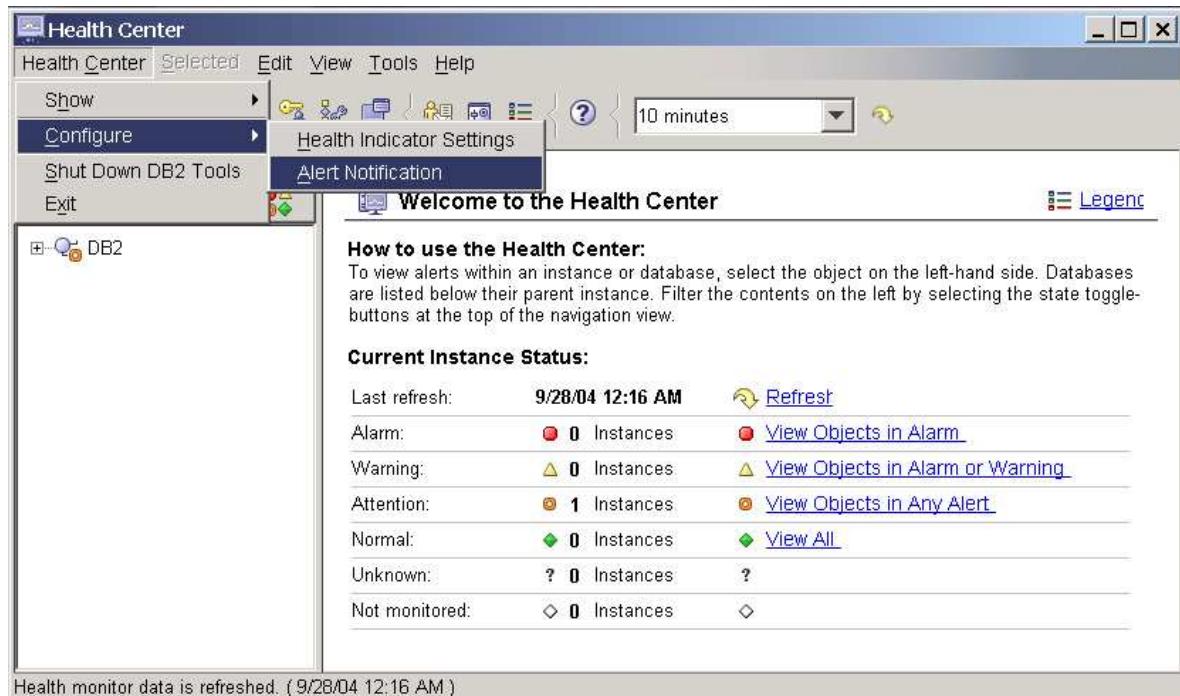


图 5.22 报警通知

5.9 自动调整内存管理

自动调整内存管理程序（STMM）是在 DB2 9 中引入的几个自主计算功能之一。它通过为几个内存配置参数自动赋值来简化内存配置任务。当启用时，该程序在几个内存使用者中为数据库动态的分配可用的内存资源。该程序通过调整内存配置参数值和缓冲池的大小来优化性能，响应负载特性的变化。

要启动 STMM，把 db cfg 的 SELF_TUNING_MEM 参数设置为 ON。其他诸如自动维护和自动储存的计算功能在本书其他地方讨论。

5.10 脚本

将常用的一连串 DB2 命令或者 SQL 语句编写成脚本是非常方便和有用的。例如，一个数据库管理员每天会运行一个脚本来检查重要数据表的行数。常用的有两种脚本类型：

1. SQL 脚本
2. 操作系统（shell）脚本

5.10.1 SQL 脚本

SQL 脚本包括查询语句和数据库命令。这类脚本比较简单易懂，并且具有平台无关性。但是 SQL 脚本不支持变量和参数。

例如，下面是在列表 5.1 中展示并保存在 script1.db2 文件中的命令。

```

CONNECT TO EXPRESS;
CREATE TABLE user1.mytable
  (    col1 INTEGER NOT NULL,
       col2 VARCHAR(40),
       col3 DECIMAL(9,2));

```

```
SELECT * FROM user1.mytable FETCH FIRST 10 ROWS ONLY;
COMMIT;
```

列表 5.1- 一个存储在 **script1.db2** 文件中的 SQL 脚本

在上面的脚本中，所有的语句都是 SQL 语句，每条语句都使用分号来隔开，文件扩展名不一定是 **db2**，任何扩展名都是可以的。

5.10.1.1 执行 SQL 脚本

可以在命令编辑器或者 Windows 的 DB2 命令窗口或者 Linux 的 shell 中执行 SQL 脚本。在 Windows 的 DB2 命令窗口或者 linux 的 shell 中执行 SQL 语句，您可以使用下面的命令：

db2 -t -v -f script1.db2 -z script1.log

或者：

db2 -tvf script1.db2 -z script1.log

在上面的命令中，

-t 表示语句使用默认的语句终结符——分号

-v 表示使用冗长模式，这样 DB2 会显示每一条正在执行命令的信息。

-f 表示其后就是脚本文件

-z 表示其后的信息记录文件用于记录屏幕的输出，方便以后的分析（这是可选的，但我们建议使用该选项）

当使用了**-t** 选项而没有标明语句终结符，则分号会默认为语句的终结符。有时可能会出现使用另外的终结符的情况，例如用 SQL PL 编写的脚本使用其它的符号而不是默认的分号，因为分号在 SQL PL 是用于定义数据库对象过程中的语句结束。

例如，如列表 5.2 中名为 **functions.db2** 的脚本包含了创建一个函数所需要的数据定义语言 (DDL)，在 **SELECT** 语句后有一个分号作为该语句的结束。而整个 **CREATE FUNCTION** 语句，则使用了叹号 (!) 作为终结符，因为如果我们依然使用分号作为这个 **FUNCTION** 对象的终结符，会在运行时产生冲突，DB2 将为此返回一个错误。

```
CREATE FUNCTION f1()
  SELECT ... ;
  ...
END!
CREATE FUNCTION f1()
  SELECT ... ;
  ...
END!
```

列表 5.2 - 脚本文件 **functions.db2** 的内容

为了使 DB2 使用其它的语句终结符，使用**-d** 选项结合**-t** 来声明其它的终结符如下的**-td!** 所示：

db2 -td! -v -f functions.db2 -z functions.log

至于其它选项的信息，可以在命令窗口或者 Linux 的 shell 中输入如下的命令来查询：

db2 list command options

5.10.2 操作系统 (shell) 脚本

相比于 SQL 脚本，操作系统脚本更加灵活和强大，因为您可以添加其它的程序逻辑。它们依赖于平台，不同的操作系统的脚本可能不相同，但是它们可以支持变量和使用参数。列表 5.3 是一个 Windows 操作系统 (Shell) 的脚本。

```
set DBPATH=C:
set DBNAME=PRODEXPR
set MEMORY=25
```

```

db2 CREATE DATABASE %DBNAME% ON %DBPATH% AUTOCONFIGURE USING
MEM_PERCENT %MEMORY% APPLY DB AND DBM
db2 CONNECT TO %DBNAME% USER %1 USING %2
del schema.log triggers.log app_objects.log
db2 set schema user1
db2 -t -v -f schema.db2 -z schema.log
db2 -td@ -v -f triggers.db2 -z triggers.log
db2 -td@ -v -f functions.db2 -z functions.log
set DBPATH=C:
set DBNAME=PRODEXPR
set MEMORY=25
db2 CREATE DATABASE %DBNAME% ON %DBPATH% AUTOCONFIGURE USING
    MEM_PERCENT %MEMORY% APPLY DB AND DBM
db2 CONNECT TO %DBNAME% USER %1 USING %2
del schema.log triggers.log app_objects.log
db2 set schema user1
db2 -t -v -f schema.db2 -z schema.log
db2 -td@ -v -f triggers.db2 -z triggers.log
db2 -td@ -v -f functions.db2 -z functions.log

```

列表 5.3 - 操作系统的脚本文件 **create_database.bat** 的内容

为了运行这段脚本，您可以在 Windows 的命令行中输入下面的语句。

```
create_database.bat db2admin ibmdb2
```

其中 db2admin 是脚本的用户 ID 和第一个参数，ibmdb2 是密码和脚本的第二个参数。

在 Windows 中使用“bat”作为扩展名，表示这是一个可执行的批处理命令。

在 Linux 中，您需要改变文件的属性，使用 chmod +x 命令来为文件添加可执行的属性。添加可执行属性后，您可以像上面那样将它运行。

5.11 基于 Windows Vista 的考虑

在 Windows Vista 中，用户访问控制（UAC）的功能会导致应用程序按照标准的权利启动，即使您的用户 ID 是本地管理员。

这意味着，任何您在 Vista 中启动的 DB2 工具或命令可能能运行，但却会通报有关访问权限的问题。

要避免此问题，使用特别在安装时为 Vista 用户创建的快捷方式“命令窗口 - 管理员”。

在这个窗口中，您可以运行其他命令并启动其他工具（使用在本章开头表 5.1 中所示的命令）或者，从 Windows Vista 开始菜单或其他 DB2 快捷方式，找到您要运行的所需的 DB2 工具，右键点击，并选择以管理员身份运行。

如果 DB2 扩展了安全模式，这是默认的（详见第 10 章，数据库安全），您还必须确保您的用户名是为了使用控制中心这类的图形化工具的 DB2ADMNS 组的一员。

5.12 小结

在这一章中，我们着眼于各种各样可以用来配置和管理您的 DB2 数据服务器的工具。

作为主要的管理工具，IBM Data Studio for DB2 9.7 的到来使数据库管理和开发工作达到了一个新的层面。

我们还讨论了一些现已过时的 GUI 工具：控制中心，SQL 协助向导，任务中心，日志，以及运行状况代理和监视器。

注意，命令行处理器和命令窗口工具在 DB2 9.7 之后的版本将继续被使用。

在 DB2 9.7 版本后，自动调整内存管理工具仍然是数据库的优化过程中的重要组成部分。

任何一个数据库管理员所需掌握的一项主要技能是使用脚本执行 DB2 命令和功能。在这一章中，我们深入了解 SQL 和操作系统（shell）脚本，特别是它们如何形成、存储和执行的。

最后，我们总结了如何确保 DB2 工具在 Windows Vista 中顺利运行。

5.13 实验

本节中的实验将让您练习在 DB2 中使用脚本。

第 1 部分：使用脚本填充 EXPRESS 数据库

在本实验，您会使用命令编辑器和两个脚本来填充 EXPRESS 数据库（之前已经创建）。

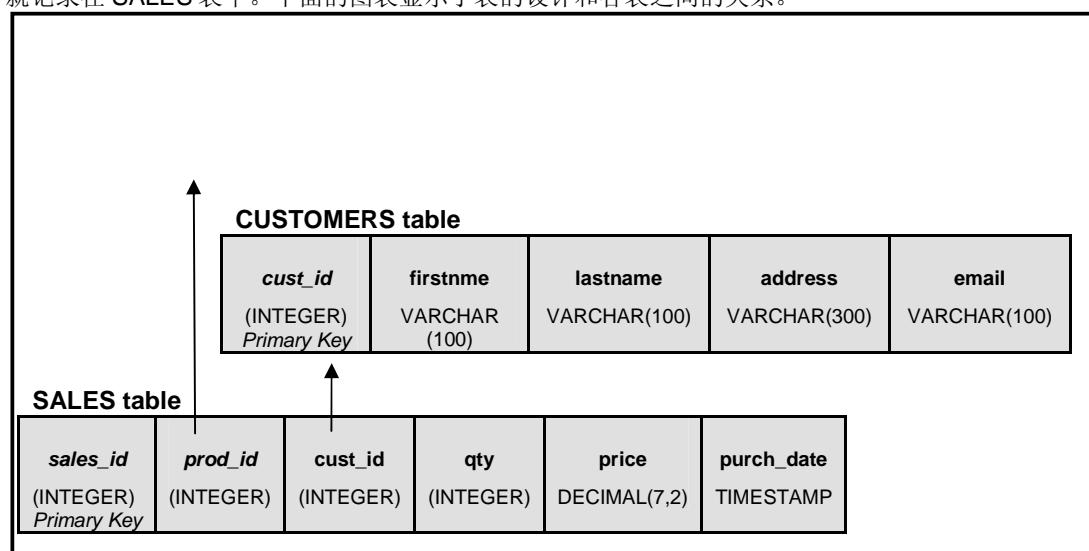
实验过程

1. 现在您需要用一些数据表和数据来填充这个 EXPRESS 数据库。为了您的方便，我们预先建立了两个脚本 Lab_Chpt5.db2 和 Lab_Chpt5.dat 供您使用。Lab_Chpt5.db2 包含了创建数据表的命令，所以必须先运行这个脚本。Lab_Chpt5.dat 脚本包含了将数据插入数据表的语句。这两个脚本都可以在 expressc_book_exercises_9.7.zip 随书文件中找到。要运行这些脚本，打开命令编辑器，确保在工具栏的下拉列表中选中那个新建的数据库。如果新建的数据库没有出现在列表中，单击 Add 按钮来添加一个到数据库的连接。

2. 在命令编辑器中单击 Selected -> Open，找到 Lab_Chpt5.db2 并单击 OK 按钮。这时，在命令编辑器的输入区域会显示 Lab_Chpt5.db2 文件的所有内容。单击运行（Run）按钮来执行这个脚本。检查在脚本运行过程中是否有出现什么错误。

3. 对 Lab_Chpt5.dat 文件重复上一个步骤。

您新建的这个数据库是用于一个非常简单的网上书店，BOOKS 表保存了书店中所有库存书目信息。CUSTOMERS 表保存了每一个顾客的信息。SALES 表保存了销售的数据。每当顾客买了一本书，就记录在 SALES 表中。下面的图表显示了表的设计和各表之间的关系。



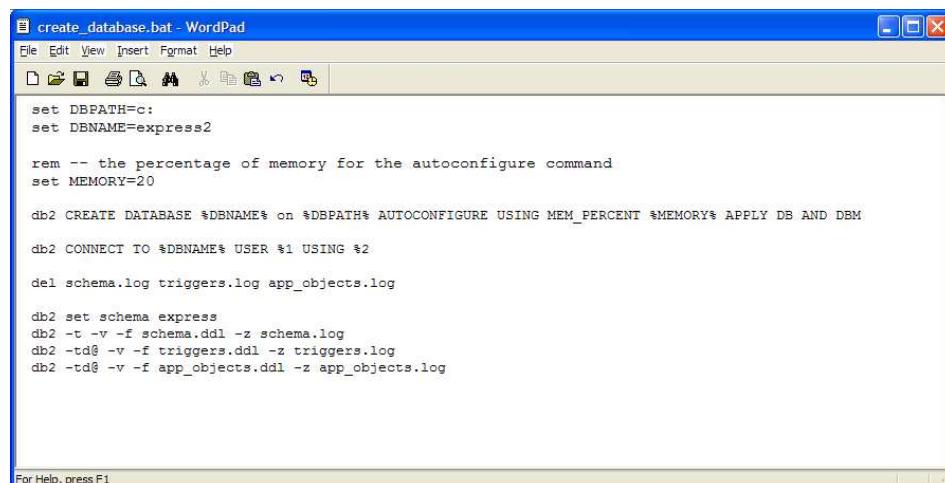
第 2 部分：为 EXPRESS 数据库创建一个安装脚本

脚本是处理重复工作的强大工具，例如收集数据库统计数据、数据库备份、数据库发布配置等存在大量重复劳动的工作可以使用脚本来方便地解决它们。操作系统脚本支持脚本参数，使得它更加灵活。在本实验中，您将会建立一个操作系统脚本来发布和配置 EXPRESS 数据库（将其配置为 EXPRESS2）。这个脚本会调用先前创建的 SQL 脚本来处理数据库对象。本实验的所有脚本和命令都是基于 Windows 平台，如果您想在 Linux 上进行实验，请将脚本和命令进行适当的修改。

实验过程

1. 打开一个文本编辑器，输入如下所示的信息。当输入下面的内容时，大多数人都会打错字。我们故意不把它作为一个单独的这文件，以便当您犯了这些错误时，可以学习解决这些问题！

请注意您也可能必须为 schema.ddl, triggers.ddl 和 app_objects.ddl 指定正确的路径，这些文件可以在 expressc_book_exercises_9.7.zip 随书文件中找到。



```

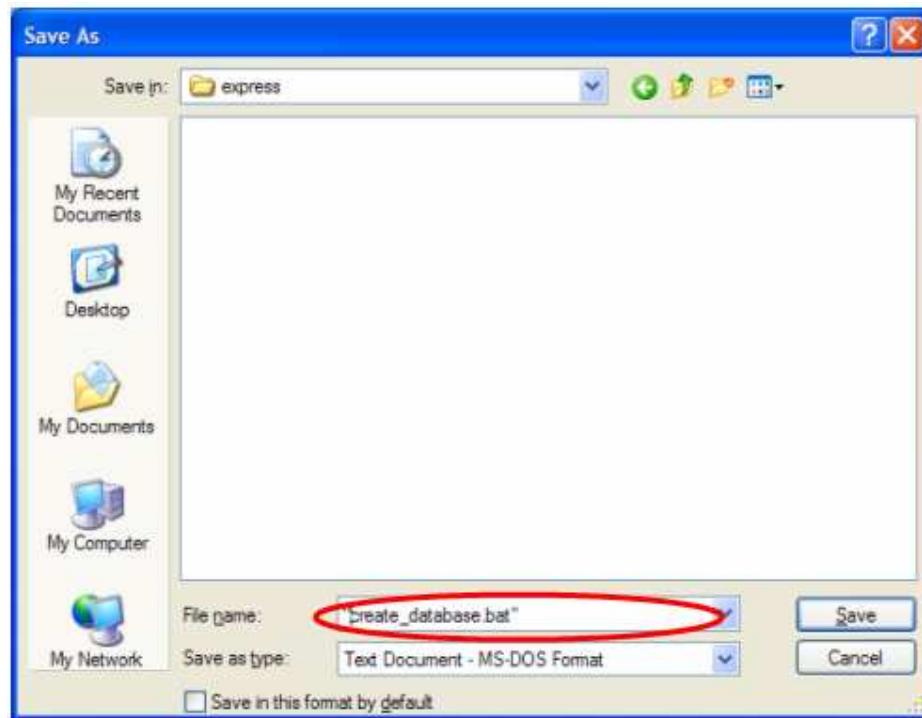
create_database.bat - WordPad
File Edit View Insert Format Help
New Open Save All Undo Redo Cut Copy Paste Select All Find Replace
set DBPATH=c:
set DBNAME=express2
rem -- the percentage of memory for the autoconfigure command
set MEMORY=20
db2 CREATE DATABASE %DBNAME% on %DBPATH% AUTOCONFIGURE USING MEM_PERCENT %MEMORY% APPLY DB AND DBM
db2 CONNECT TO %DBNAME% USER $1 USING $2
del schema.log triggers.log app_objects.log
db2 set schema express
db2 -t -v -f schema.ddl -z schema.log
db2 -td@ -v -f triggers.ddl -z triggers.log
db2 -td@ -v -f app_objects.ddl -z app_objects.log

```

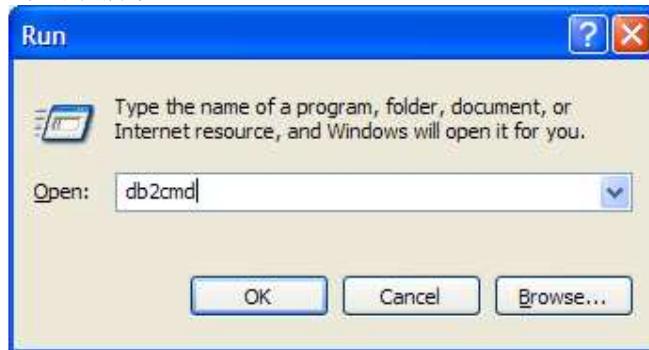
For Help, press F1

2. 把脚本文件保存于一个目录中，如：C:\express 并命名为 create_database.bat。

注：如果您使用的是写字板，在另存为对话框中，确保您选择的是 MS-DOS 格式。如果您选择了其它的格式，会在文件中插入一些不可见的字符，这会导致该脚本在运行中出错。同时，使用双引号将文件名括起来，这样 Windows 就不会将.txt 的扩展名添加到脚本文件名后面，如下图所示：

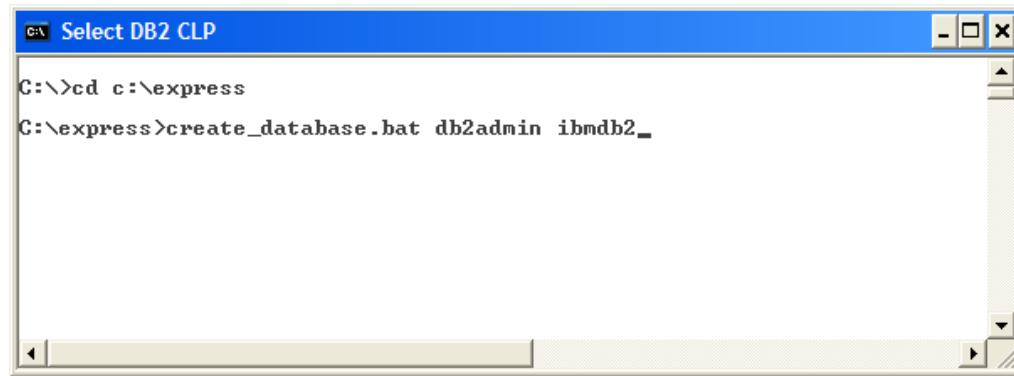


3. 为了与 DB2 交互地运行这段脚本，您必须先打开 DB2 命令行环境。您可以从 Start > Program Files > IBM DB2 > DB2COPY1（默认值）>Command Line Tools > Command Window 打开 DB2 命令行窗口。您也可以使用 Windows 的“运行”命令（Start > Run），键入 db2cmd 之后按回车键。如下图所示：



4. 输入下面的命令来执行脚本

```
cd C:\express
create_database.bat db2admin ibmdb2
```



The screenshot shows a Windows command-line interface window titled "Select DB2 CLP". The window contains the following text:

```
C:\>cd c:\express  
C:\express>create_database.bat db2admin ibmdb2
```

5. 请花些时间来理解上面创建的脚本，您能说出每一行的作用是什么吗？
6. 尝试回答下列问题：
 - A、数据库在哪里建立连接？
 - B、%1 和 %2 是什么意思？
 - C、下面的语句起什么作用？作用在哪里？为什么要这样做？
SET DBPATH=C:
 - D、下面的语句有什么作用？
del schema.log, triggers.log, app_objects.log
 - E、如果无参执行上面的脚本会发生什么事情？
 - F、为什么 SQL 脚本没有包含 CONNECT TO 语句？它们是怎么样连接到数据库的？

PART II – DB2 Express-C 数据库管理

6

第 6 章 – DB2 体系结构

这章我们简要介绍 DB2 体系结构：

- DB2 进程模型
- DB2 内存模型
- DB2 存储模型

注意：

更多有关 DB2 体系结构的信息请参看以下视频文件：

<http://www.channeldb2.com/video/video/show?id=807741:Video:4482>

6.1 DB2 进程模型

图 6.1 描述了 DB2 的进程模型，在这个图中，长方形代表了处理进程，而椭圆形代表了处理线程。DB2 中主进程是 db2sysc。在这个处理进程中有许多的线程，最主要的也称为 db2sysc。这个主要的线程派生了其他的子线程。当一个远程的应用程序尝试使用 SQL CONNECT 语句连接服务器时，通讯协议的远程监听器将接收这个请求，并联系 DB2 协调代理（db2agent）。DB2 代理是一个代表 DB2 实现一些操作的小处理单元。当发出请求的应用程序是本地的，也就是说，它与 DB2 在同一个服务器上运行，那么，整个过程中的步骤也是十分类似的，唯一不同的是 db2ipccm 代理取代了 db2tcpcm 线程处理本地应用程序的请求。有些情况下，比如发生并行时，db2agent 也许会生成其他代理诸如 db2agntp 线程等。图中所示的其他的一些代理包括：db2pfchr, db2loggr, db2dlock，它们都将应用于不同的目的。表 6.1 描述了大部分常见的进程，表 6.2 描述了大部分常见的线程。

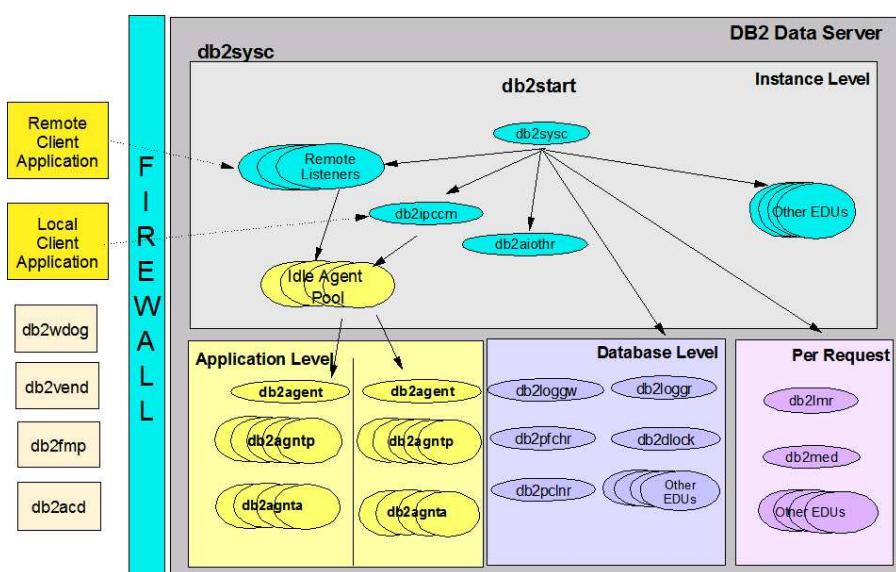


图 6.1 – DB2 进程模型

进程名称	描述
db2sysc (Linux) db2syscs (Win)	DB2 的主系统控制器或者引擎。在 DB2 9.7 中，对于一个完整的分区其中只有一个包含多线程的主引擎进程。所有的引擎可调度单元（EDU）都是进程中的线程。如果没有这个进程，数据库服务器是无法工作的。
db2acd	主管运行状况监视器和自动维护实用程序的自主计算守护程序。此进程以前称为 db2hmon。
db2wdog	DB2 的看门狗。看门狗是主引擎进程 db2sysc 的父进程。如果 db2sysc 进程非正常终止，它将清除其所占用的资源。
db2vend	在主引擎进程之外的设防方式进程，DB2 9.7 中，所有的第三方代码都在这个进程中运行。第三方的程序指那些非 IBM 的程序，它们可以和 DB2 交互，例如可以使用第三方程序执行记录归档功能，只需要将用户的出口参数指向这个程序。
db2fmp	设防方式进程，在防火墙外运行用户的存储程序和用户定义函数代码。此进程代替了 DB2 老版本中使用的 db2udf 和 db2dari 进程。

表 6.1 – 常见 DB2 进程

线程名称	描述
db2sysc	系统控制线程。这个线程主要负责启动实例、关闭和管理正在运行的实例。
db2tcpcm	TCP/IP 通信监听器
db2agent	代表应用程序实现数据库操作的协调代理（至少每连接一个，具体数量取决于是否使用连接集中器）
db2agntp	如果 INTRA_PARALLEL 的属性设置为 YES。那么会产生活动的副代理。它会为应用程序执行数据库操作。db2agent 将协调不同 db2agntp 副代理之间的工作

db2pfchr	DB2 异步 I/O 数据预取程序(NUM_IOSERVERS)。
db2pclnr	DB2 异步 I/O 数据写入程序(NUM_IOCLEANERS)

表 6.2 – 常见 DB2 线程

6.2 DB2 内存模型

DB2 内存模型由内存的不同区域构成，这里的内存是指存在于实例层、数据库层、应用程序和代理层中的内存，如图 6.2 所示。本书中不详细解释每个不同的内存区域，相应的，我们提供概要性的总结。

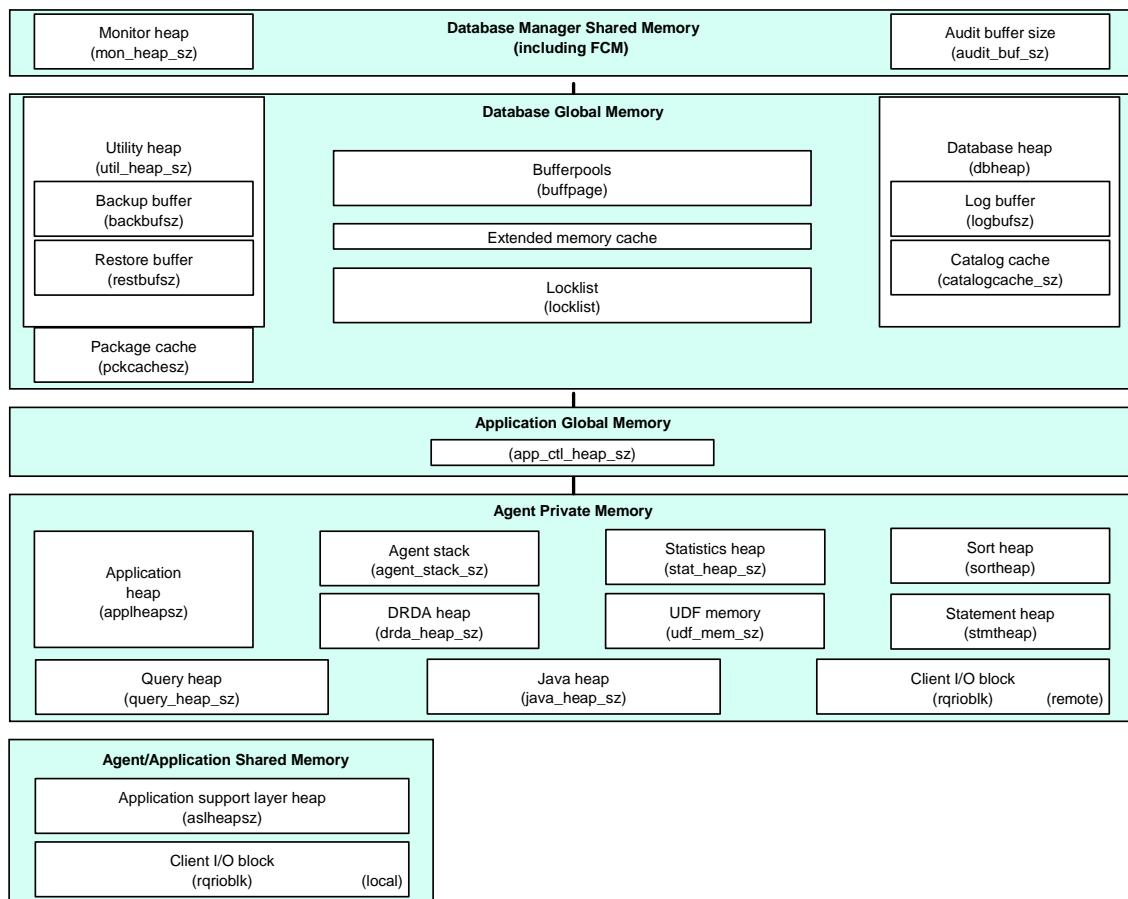


图 6.2 – DB2 内存模型

当一个实例启动后，系统就为数据库管理器分配共享内存，这通常不需要花费很大的空间。第一次连接数据库时，就会分配数据库全局内存（**Database Global Memory**）。在此区域（内存块）内，缓冲池是最重要的部分之一，其主要用于优化查询性能。缓冲池的大小将决定整个数据库全局内存（**Database Global Memory**）的大小。

每个DB2代理都使用一个代理私有内存。在没有连接集中器的情况下，每一个连接都需要一个代理。一般情况下，一个代理使用大约3-5MB内存。在有连接集中器的情况下，几个连接可以使用一个代理，因此减少了对物理内存的需求。

6.3 DB2 存储模型

本节将讨论以下概念：

- 数据页和扩展数据块
- 缓冲池
- 表空间

6.3.1 数据页和扩展数据块

页是在DB2中的最小存储单元。允许的数据页大小为：4K、8K、16K和32K。扩展数据块是一组数据页。DB2中，每次处理一页会影响数据库的性能，所以DB2以扩展数据块为单位进行处理。页大小和扩展数据块大小在使用缓冲池和表空间的时候定义。

6.3.2 缓冲池

缓冲池是表和索引数据在内存中的缓存。它通过减少直接的顺序IO存取，并提供异步读取（预取）和写入来提高系统的性能。也就是说，DB2预测那些所要用到的数据页并将它们预取到缓冲池中以备随时可用。

缓冲池在内存中以4K、8K、16K、32K的页面大小为单位。每一个数据库至少需要一个缓冲池，而且对于每个既定页面大小的表空间，至少存在一个相同页面大小的缓冲池与之相匹配。

6.3.2.1 创建缓冲池

您可以使用CREATE BUFFERPOOL语句来创建一个缓冲池。当然，您也可以使用控制中心来创建缓冲池。在控制中心中，右键单击某一数据库的缓冲池文件夹，然后选择Create，如图6.3所示。

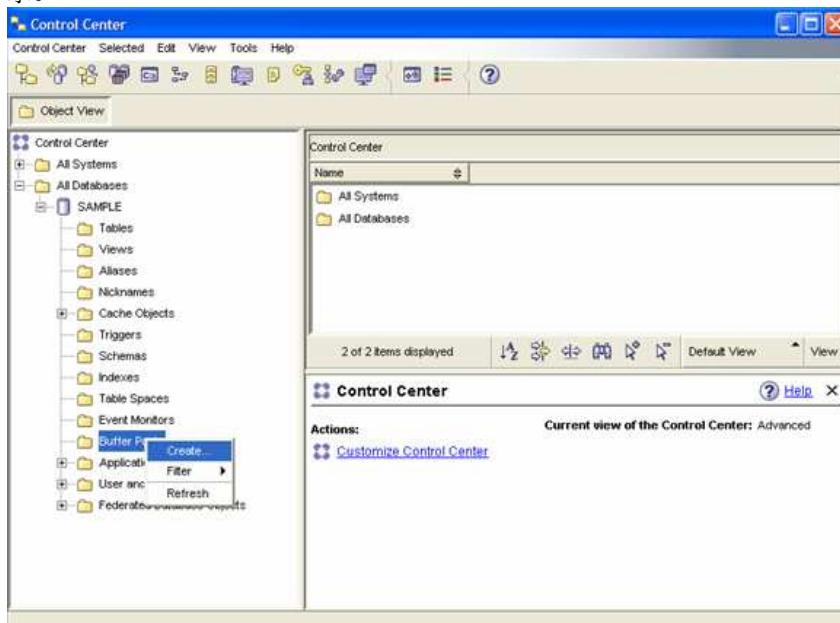


图 6.3 – 创建一个缓冲池

点击 Create 后，会出现如图 6.4 所示的创建缓冲池对话框。

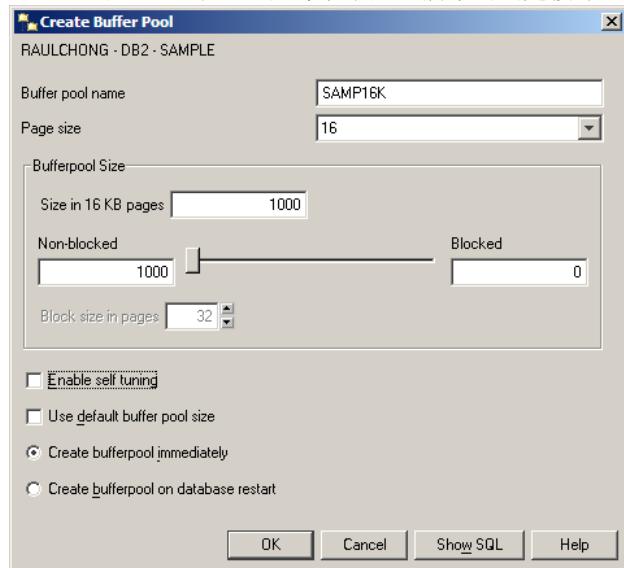
**图 6.4 – 创建缓冲池对话框**

图 6.4 中的大多数条目都很简单明了，Non-blocked 和 Blocked 区域表示数据页以块或者非块存在的数目。基于块的缓冲池保证硬盘上连续的页面也会连续地移动到块区域中，这能够提高性能。页面数必须小于 98% 的缓冲池页面数目。

当缓冲池创建成功后，它就会显示在控制中心里，如图 6.5 所示。

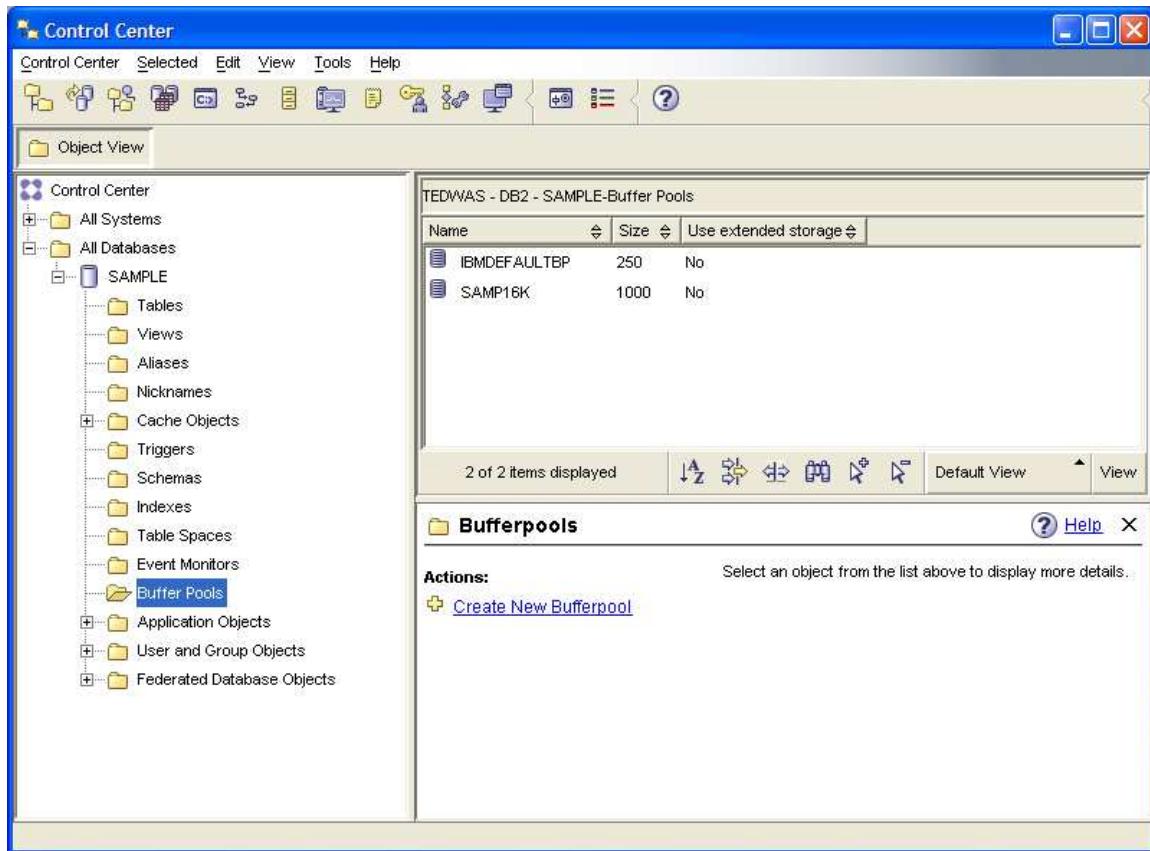


图 6.5 – 创建名为“SAMP16K”的缓冲池后的控制中心示意图

6.3.3 表空间

表空间是处于逻辑的数据表和系统物理内存（缓冲池）以及容器（硬盘）之间的逻辑接口。用 CREATE TABLESPACE 语句来创建一个表空间。创建时您能够制定如下的参数：

- 表空间的页大小（4KB, 8KB, 16KB, or 32KB）。页大小必须与其关联的缓冲池的大小相同。
- 分配给这个表空间的缓冲池名字。
- 扩展的大小。
- 预取的大小。

6.3.3.1 表空间类型

表空间有三种类型：

- 常规表空间

这种表空间用于用户的数据表。比如，默认创建的 USERSPACE1 表空间就是一个常规表空间。

- 大型表空间

These are used to optionally separate LOB data into its own table space 这种表空间是一种为可选的方法，它将 LOB 数据单独存储在自己的表空间。如果创建时指明 pureXML 的支持（数据库以 UNICODE 编码创建，并且使用 XML 数据类型作为列属性），它也能用作存储 XML 数据，这种情况下，大型表空间是默认的表空间。

- 临时表空间

临时表空间有两种：

- ▶ 系统临时表空间

系统临时表空间用于 DB2 的内部操作，比如排序等。**TEMPSPACE1** 就是一个默认创建的系统临时表空间，它在您创建一个数据库时自动建立。

- ▶ 用户临时表空间

用户临时表空间用于创建用户定义全局临时表（在内存中的临时表）。它们通常和系统临时表空间相混。在 **DGTT** 或 **CGTT** 可以用之前，用户必须建立用户临时表空间。

6.3.3.2 表空间的管理

表空间可以根据他们如何被管理来进行分类。管理方式能在使用 **CREATE TABLESPACE** 创建表空间时指定。

由系统管理

这种类型的表空间也称为系统管理存储（**System Managed Storage, SMS**）表空间。这意味着由操作系统来管理表空间的存储。这种方式的表空间是很容易管理的，表空间的容器就是 **OS** 文件系统的文件夹。存储空间不是预先分配的，但是表空间文件能够动态增长。当您指定容器后，它就会在创建表空间时固定，以后不能再添加其它的表空间容器，除非使用转向存储。当使用 **SMS** 表空间，表中的数据、索引、**LOB** 数据不能够跨越不同的表空间。

由数据库管理

这种类型的表空间也称为数据库管理存储（**Database Managed Storage, DMS**）表空间。这意味着由 **DB2** 管理表空间的存储。这种表空间的管理需要数据库管理员（**DBA**）的人工干预，能够预先分配容器或者裸设备。使用裸设备时，数据是直接写入到设备中而没有使用 I/O 缓存。

这种管理方式能够增加、删除容器，也能够改变容器大小。**DMS** 表空间对性能是最优的，数据表的数据、索引、**LOB** 数据能够分割到不同的表空间中以提高性能。

由自动存储管理

这种管理方式由自动存储来管理，它具有 **SMS** 表空间类似的自主管理性，又具有 **DMS** 数据表空间的灵活性和高性能表现。所以，从 **DB2 9** 开始，这是默认的表空间类型。使用这种表空间时，用户首先指明一个逻辑存储设备组，不需指明其容器。容器会在存储路径里自动创建。容器的增长和新建都是有 **DB2** 来管理的。如果在 **CREATE DATABASE** 命令中没有指定存储路径，数据库路径将作为存储路径。而数据库路径就是主数据库的定义路径。如果数据库路径也没有指定，那么就从数据库管理配置参数 **DFTDBPATH** 中获取存储路径。在 **Windows**，路径只能是一个驱动器而不是路径。

为了使用自动存储管理，您在创建一个数据库时必须允许自动存储管理（这是默认的行为），然后为其分配一个存储路径集合。这个数据库创建之后，如果您需要，可以重新定义存储的路径，这只需使用 **RESTORE** 操作即可。然后，您就可以创建自动存储管理的表空间（同样，这也是默认的行为）。

由自动存储进行管理与 **DMS** 表空间是很相似的，但是具体操作已经自动化了所以它们由 **DB2** 来管理；这些操作包括了容器的分配和自动调整大小。

我们来看一个由自动存储表空间管理的例子。首先，创建允许自动存储管理的数据库，如下面的例子所示：

```
创建数据库时默认使用的就是自动存储管理
CREATE DATABASE DB1
```

```
显式声明使用自动存储管理
CREATE DATABASE DB1 AUTOMATIC STORAGE YES
```

创建数据库时默认使用的就是自动存储管理，但存储路径是显示的。如果存储路径是目录，那么它必须提前创建。

Windows 的例子:

```
CREATE DATABASE DB1 ON C:/, C:/storagepath1, D:/storagepath2
```

注意，上一行中的第一项是驱动器，因为在 Windows 里，其代表的数据库路径只能是驱动器，而不是路径。

这一项也将作为存储路径使用。但，数据库路径是 C:/，而存储路径由 C:/, C:/storagepath1, D:/storagepath2 组成，而且后面 2 个目录必须提前创建。

Linux 的例子:

```
CREATE DATABASE DB1 ON /data/path1, /data/path2
```

显式禁用自动存储管理

```
CREATE DATABASE DB1 AUTOMATIC STORAGE NO
```

然后，创建使用自动存储管理的表空间，如下的例子所示:

创建表空间时默认使用的也是自动存储管理:

```
CREATE TEMPORARY TABLESPACE TEMPTS
```

显式声明使用自动存储管理的表空间:

```
CREATE TABLESPACE TS2 MANAGED BY AUTOMATIC STORAGE
```

隐式声明适用自动存储管理，并且指明初始化分配的大小、增长的大小、所能达到的最大容量。

```
CREATE TABLESPACE TS1
```

```
INITIALSIZE 500 K
```

```
INCREASESIZE 100 K
```

```
MAXSIZE 100 M
```

6.3.3.3 数据是怎么存储在表空间的

默认情况下，DB2 每次写入数据到硬盘都会跨越若干个容器。例如，您使用 4K 大小 DMS 表空间，扩展数据块数页数为 8，使用 3 个容器，这就是说，在写入下一个容器之前，会将 32K 数据（ $4\text{K} \times 8$ 页/扩展块 = 32K）写入当前容器中。图 6.6 展示了这种情况。注意，不同表不会共享相同的扩展数据块。

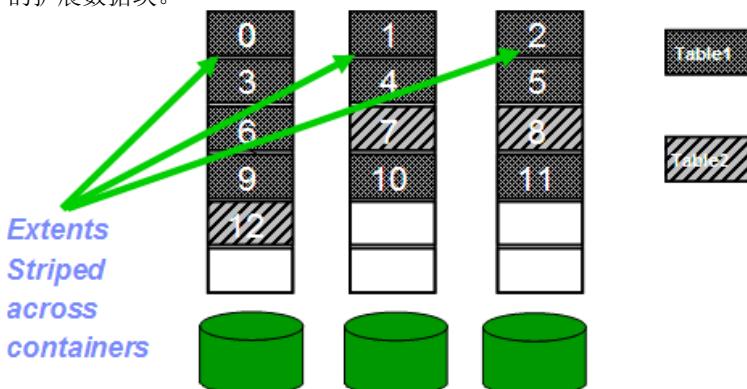


图 6.6 – 在表空间中写入数据

6.3.3.4 使用控制中心创建表空间

使用控制中心创建表空间，首先在某个数据库上右键单击表空间（Table Spaces）文件夹，然后选择 **Create**，如图 6.7 所示。这是会出现创建表空间向导（Create table space wizard），如图 6.8 所示。根据向导提示，即可完成表空间的创建。

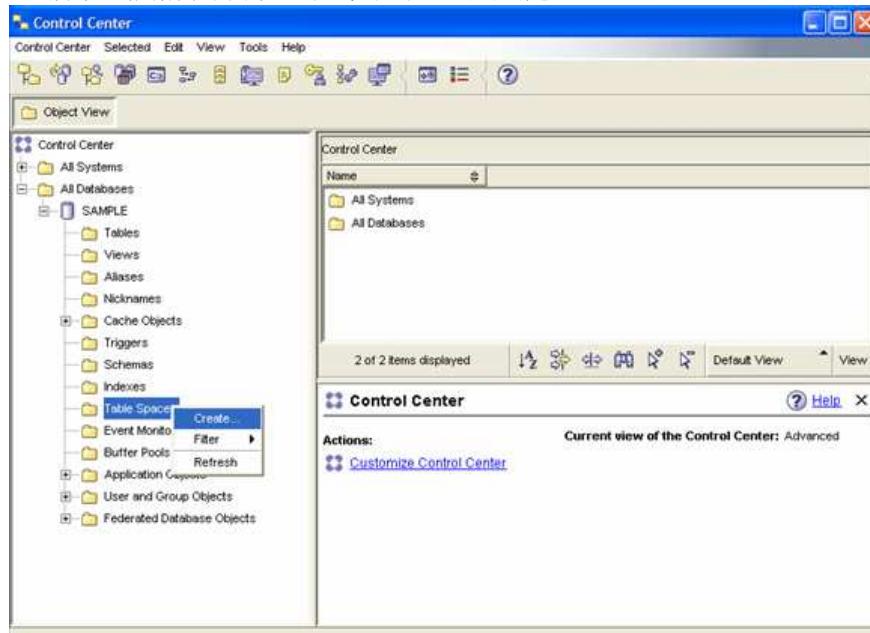


图 6.7 – 从控制中心创建表空间



图 6.8 – 创建表空间向导

图 6.8 显示的向导会一步一步指导您创建表空间。

6.4 小结

在这一章，我们探究了 DB2 体系结构的 3 个主要方面：DB2 进程模型，内存模型和存储模型。在进程模式中，我们了解了一些常见的进程和线程，其中包括 db2sysc，而缺少它 DB2 将不能运行。

我们深入地讨论了存储模式，其覆盖了三个最重要的方面：数据页和扩展数据块，缓冲池（具体的创建过程）和表空间。最后，我们介绍不同类型的表空间，并详细介绍它们是如何管理的（SMS，DMS，Automatic）和如何使用控制中心创建一个新的表空间。

6.5 实验

这个实验将会帮助您更好的理解在 Windows 上，DB2 的进程模型，内存模型和存储模型。您会复习到不同的进程和线程，监控内存使用情况并练习在 Windows 上创建一个使用自动存储和存储路径的数据库。理想情况下，存储路径将在磁盘（驱动器）中创建，但因为您的电脑可能没有配置多个磁盘，这个实验只使用 C:\ 驱动器。

实验过程：

1. 先看一下 Windows 的一些进程，首先打开 DB2 命令窗口（Start -> run -> db2cmd），输入命令：db2stop force 确保您的例子已经停止。
2. 打开 Windows 任务管理器，选择 Processes 选项卡，点击 Image Name 列按顺序排列，找到 db2sysc.exe 进程，如下表所示。

Image Name	User Name	CPU	Mem Usage
cmd.exe	SYSTEM	00	1,988 K
cmd.exe	arfchong	00	2,952 K
cmd.exe	arfchong	00	2,928 K
csrss.exe	SYSTEM	00	10,288 K
ctfmon.exe	arfchong	00	3,772 K
CUCore.exe	arfchong	00	7,824 K
cvrnd.exe	SYSTEM	00	8,308 K
db2bp.exe	arfchong	00	11,648 K
db2dasrrm.exe	SYSTEM	00	8,876 K
db2icrv.exe	SYSTEM	00	1,472 K
db2mgmtsvc.exe	SYSTEM	00	6,984 K
db2rcmd.exe	arfchong	00	8,504 K
db2systray.exe	arfchong	00	7,592 K
DefWatch.exe	SYSTEM	00	5,732 K
DIMon.exe	arfchong	00	3,388 K
DLG.exe	arfchong	00	3,856 K
Dot1Ccfg.exe	arfchong	00	16,452 K
EvtEng.exe	SYSTEM	00	13,936 K
explorer.exe	arfchong	01	39,592 K

Show processes from all users

Processes: 132 CPU Usage: 22% Commit Charge: 1380M / 3931M

3. 您应该找不到进程 db2syscs.exe，因为在步骤 1 中我们已经要求您停止 DB2 实例了。

4. 输入指令: db2start 开始 DB2 实例, 再重复第 3 步的操作, 现在您能找到 db2syscs.exe 进程吗?

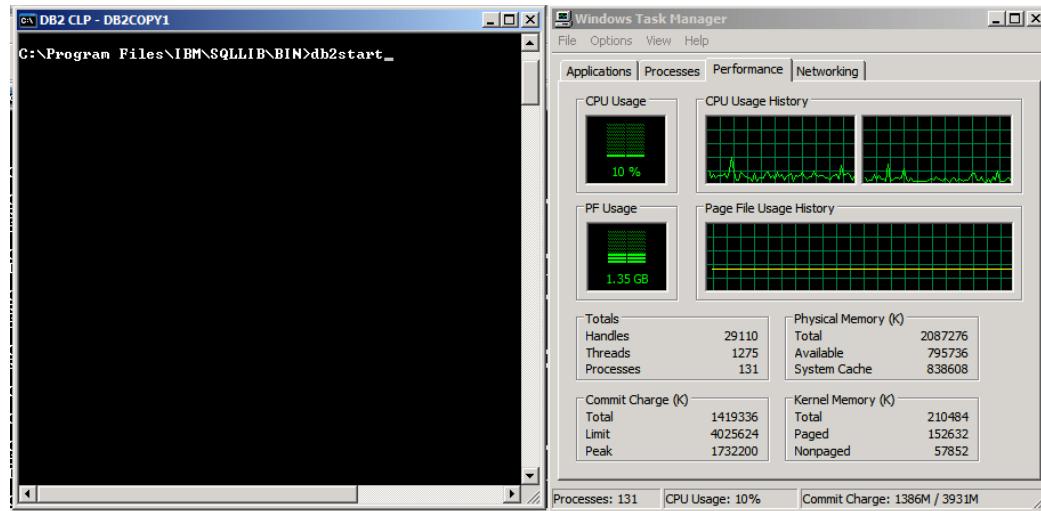
5. 我们再来查看下 CPU 和内存的占用情况, 进行以下步骤:

A 关掉所有的其他应用程序, 确保您的系统中没有什么在运行。

B 打开一个新的 DB2 窗口, 输入: db2stop force

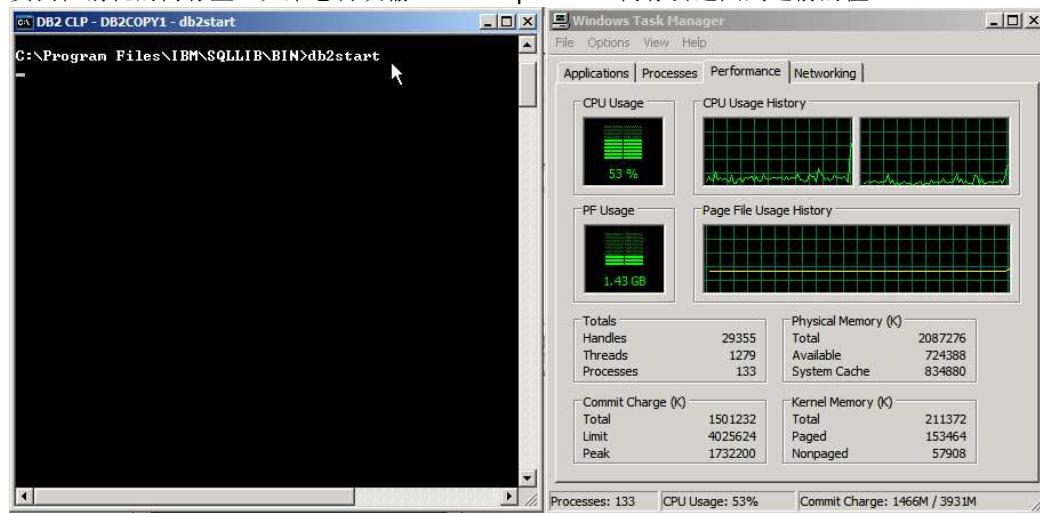
C 在任务管理器里, 切换到性能选项卡

D 让任务管理器和 DB2 命令窗口如下图一样并列打开着, 写下可用内存的数量和 CPU 的使用量。



E 输入 db2start, 在您执行这条命令的同时监视 CPU 和内存的使用率。

您将会在 CPU 使用率中看到一个短暂的顶峰, 可用内存将下降 50MB 到 70MB, 这是这个 DB2 实例在消耗的内存量。如果您再次输入 db2stop force, 内存会返回到之前的值。



6. 重复之前的步骤, 但这次, 在连接到 SAMPLE 数据库后, 监视有什么变化。在 DB2 命令窗口中输入如下命令:

```
db2start
db2 connect to sample
```

在您连接到 SAMPLE 数据库后，您将马上看到可用物理内存的减少。那是因为，一旦您连接到一个数据库，数据库全局内存（缓冲池， 目录缓存， 等等）将被分配。

7. 重复之前的步骤，但这次，在创建一个任意大小的缓冲池后，监视会有什么变化。确保您的电脑中没有超过物理内存。这样做后，DB2 不会立即分配缓冲池，但 DB2 会延缓创建直到数据库停用。

除此之外，一个小的系统缓冲池将会代替使用，DB2 将一直使用这个缓冲池直到内存足够。比如，要创建一个 160MB 的缓冲池，当连接到 SAMPLE 数据库时，输入以下命令：

```
db2 create bufferpool mybp immediate size 5000 pagesize 32k
```

8. 创建一个数据库 mydb1，这个数据库使用自动存储管理，其数据库路径为驱动器 C:，存储路径为 C:，C:\mystorage1，C:\mystorage2。在 DB2 命令窗口输入：

```
db2 create database mydb1 on C:\, C:\mystorage1, C:\mystorage2
```

您输入上述命令后，可能会收到一个错误，因为您必须先创建目录 C:\mystorage1，C:\mystorage2。

创建目录再试一遍吧。

7

第 7 章 – DB2 客户端的连接

本章将介绍使用 TCP/IP 协议从 DB2 的客户端连接到 DB2 服务器端所需的步骤。因为 DB2 服务器自带了客户端组件，所以 DB2 服务器也可作为一个客户端与其它的 DB2 服务器相连。DB2 客户端有几种不同方式的连接设置，不过在本章中我们只讨论最简单的一种——使用配置助手（Configuration Assistant）。

注意:

更多有关 DB2 结构的信息，请参见以下视频：

<http://www.channeldb2.com/video/video/show?id=807741&Video:4222>

7.1 DB2 目录

DB2 目录是一些二进制文件，文件中存储的是关于您的机器可以访问到哪些数据库的信息。DB2 共有四个不同的目录：

1. 系统数据库目录（System database directory）

这个目录就像一本书的目录。它显示了所有您可以连接到的本地或远程数据库。对于本地数据库，将会有个指针来指向这个本地数据库目录（Local database directory）。对于远程数据库，它将有一个指针指向结点目录（Node directory）。可以使用如下命令查看目录内容：

```
list db directory
```

2. 本地数据库目录（Local database directory）

这个目录包含的信息是关于您可以连接到的本地数据库。可以用如下命令查看目录内容：

```
list db directory on <drive/path>
```

3. 结点目录（Node directory）

这个目录包含的信息是关于如何连接到一个给定的数据库。例如，如果使用的是 TCP/IP 协议，一个 TCP/IP 结点项就将包含您试图连接的 DB2 数据库所在服务器的 IP 地址，以及这个数据库的实例所用的端口。可以用如下命令查看此目录的内容：

```
list node directory
```

4. 数据库连接服务目录（DCS directory， DCS: Database Connection Services directory）

这个目录只有在您已经安装了 DB2 的连接软件（用于连接到 DB2 z/OS（大型机）版或 DB2 i5/OS 版）时才会出现。查看此目录要用到如下命令：

```
list dcs directory
```

以上所有目录的内容都可以通过 GUI 配置助手（Configuration Assistant）工具来进行查看和修改。

7.2 配置助手（Configuration Assistant）

使用配置助手，您可以轻松地配置 DB2 客户端与服务器端之间的连接。

在 Windows 系统中运行配置助手，您只要选择“开始>程序>IBM DB2>DB2COPY1>设置工具>配置助手”（*Start > Programs > IBM DB2 > DB2COPY1 > Set-up Tools > Configuration Assistant*）

您也可以用 db2ca 命令从命令行启动这个工具。配置助手如图 7.1 所示：

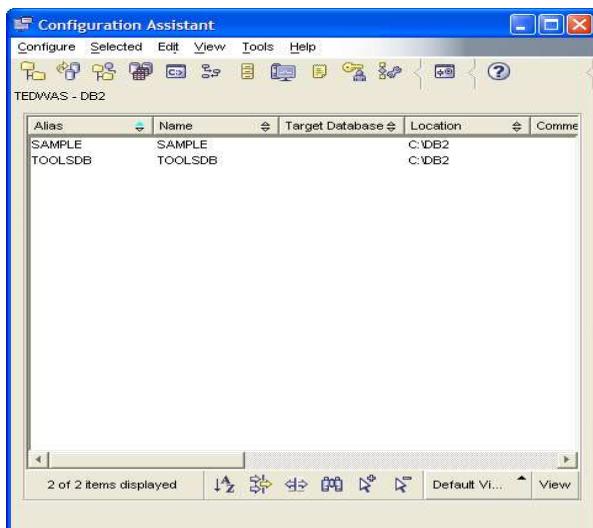


图 7.1 – 配置助手

7.2.1 服务器端的安装要求

服务器端有两处需要设置：

1) DB2COMM

这个注册值决定了应当由哪一个通信协议控制器来监听来自客户端的请求。通常 TCP/IP 是最常用的通信协议。改变这个参数需要重启实例。如需在配置助手中查看或改变 DB2COMM 的值，请选择配置->DB2 注册表（*Configure -> DB2 Registry*）（如图 7.2, 图 7.3 所示）。

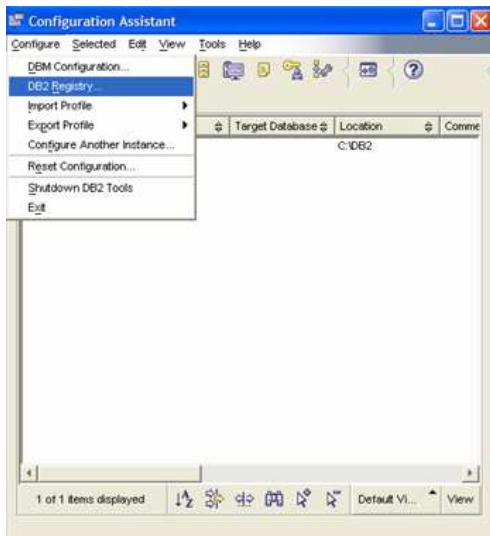


图 7.2 – 访问 DB2 注册表

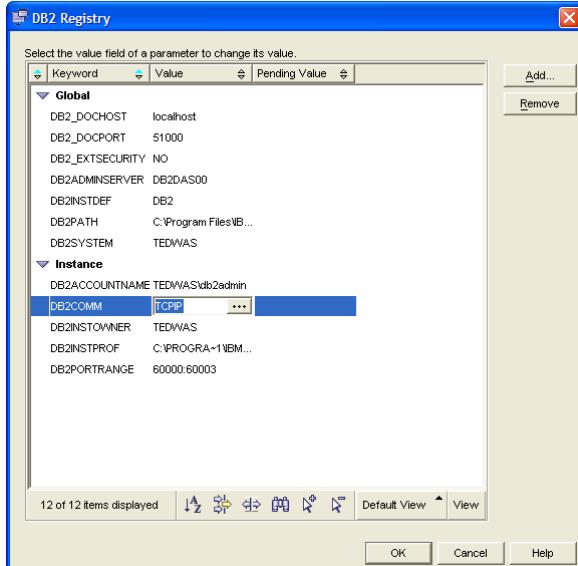


图 7.3 – 查看 DB2 注册表的 DB2COMM 值

2) SVCENAME

这个 DBM 配置参数应当被设为服务器名（与在 TCP/IP 服务文件中定义的服务器名一致），或者设为您想在这个实例中访问数据库时所使用的端口号。如图 7.4 所示，在配置助手中，选择配置>DBM 配置（Configure > DBM configuration）。

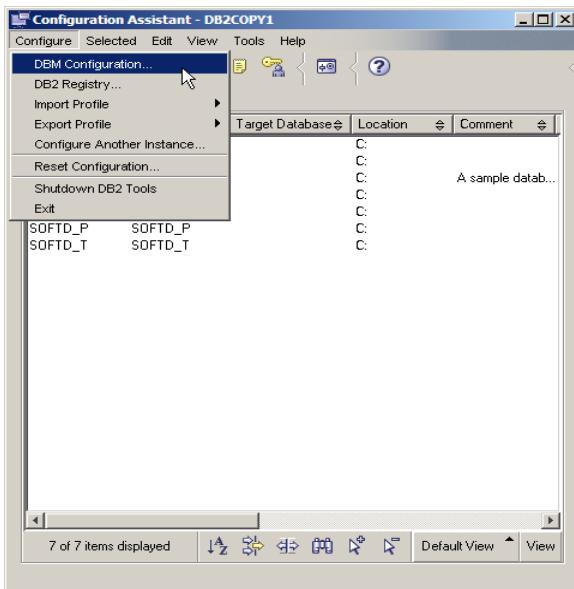


图 7.4 – 在配置助手中查看 DBM 配置

进入 DBM 配置窗口，找到“Communications”项，然后寻找 SVCENAME。如有必要，您可以把这个值改为一个与端口号相同的字符串。如图 7.5。

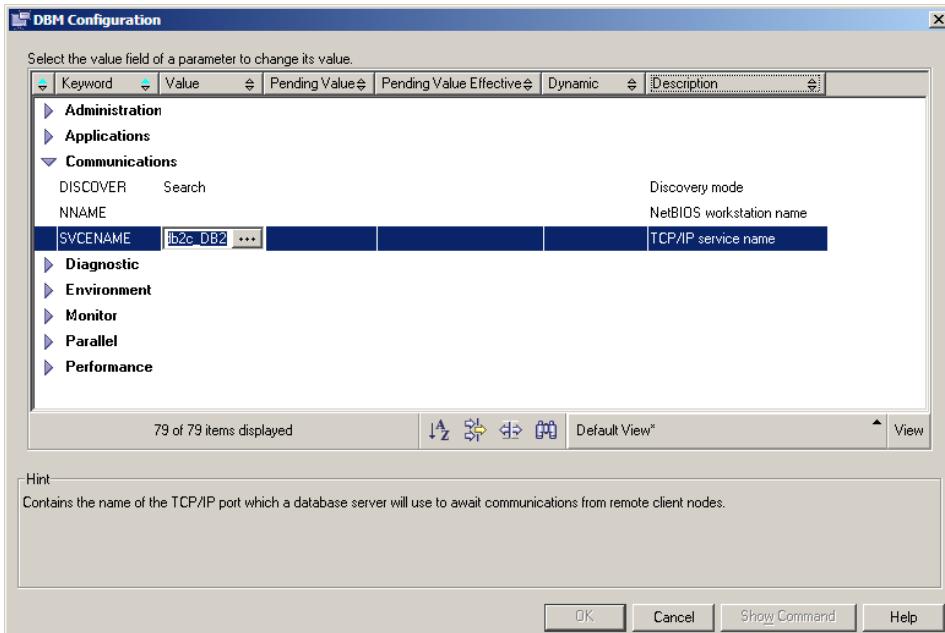


图 7.5 – 查看 DBM 配置的 SVCENAME 参数

7.2.2 Setup required at the client 客户端的安装要求

在客户端，您需要事先了解以下信息：

1. 您要连接的数据库的名称
2. 数据库所在的服务器上 DB2 实例所使用的端口号。您也可以用服务名，只要它与 TCP/IP 服务文件中的项相匹配。

3. 知道操作系统用户名和密码以便连接到数据库时使用。用户名必须已经在服务器端建立。

以上信息可以使用配置助手从 DB2 客户端输入。首先启动添加数据库向导：选择“所选>使用向导来添加数据库”（Selected -> Add Database Using Wizard）选项。（如图 7.6）。

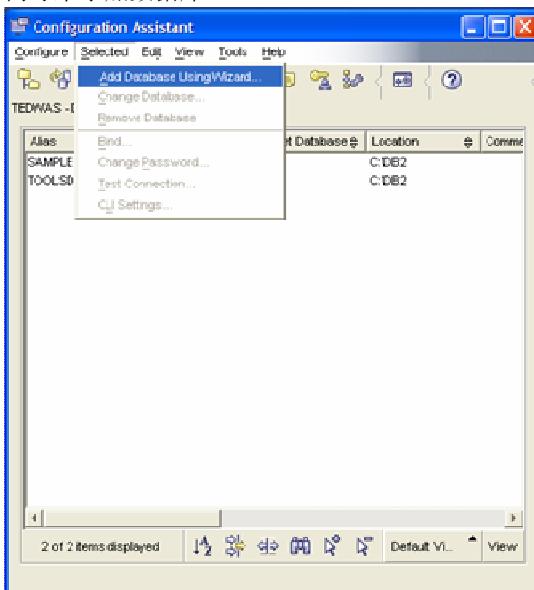


图 7.6 – 运行“使用向导来添加数据库”

您也可以用右键单击配置助手的空白处并选择“使用向导来添加数据库”的方式打开这个向导。

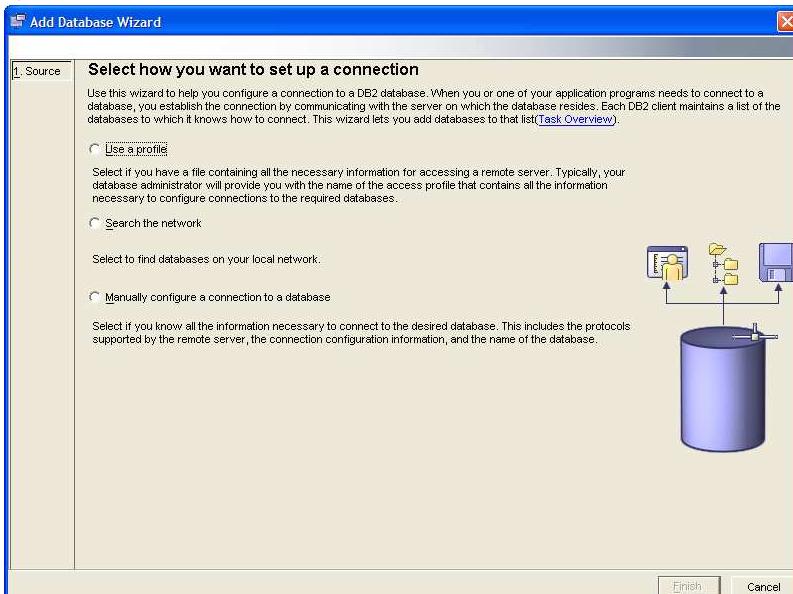


图 7.7 – 添加数据库向导

在添加数据库向导中，有三个选项：

使用概要文件

在有些情况下，您可能需要配置多个客户端到同一个服务器的连接。在这样的情况下，一个方便的做法是在一个客户端上进行所有的配置，并且把这些配置存入一个“概要文件”，然后在其它客

客户端上直接使用该文件进行配置。如果您选择了“使用概要文件”，您将从一个已经存在的概要文件加载信息。本章稍后将介绍具体该如何建立客户端与服务器的概要文件。

搜索网络

这一方式也称做“探索（Discovery）”，令DB2在网络中搜索给定的服务器、实例和数据库。为使这种方式正常运行，每一个DB2服务器上必须运行DAS（DB2管理服务器，DB2 Administration Server），使得服务器能够被发现。通常有两种方法来执行搜索：

- 搜索网络（Search）

搜索整个网络。如果您的网络很大并且有许多的集线器，那么不推荐您这样做。因为检索每一个系统的数据将会耗费相当长的时间。

- 已知系统（Known）

根据您提供的地址在网络中搜索已知的系统。

图 7.8 描述了这两种方式。

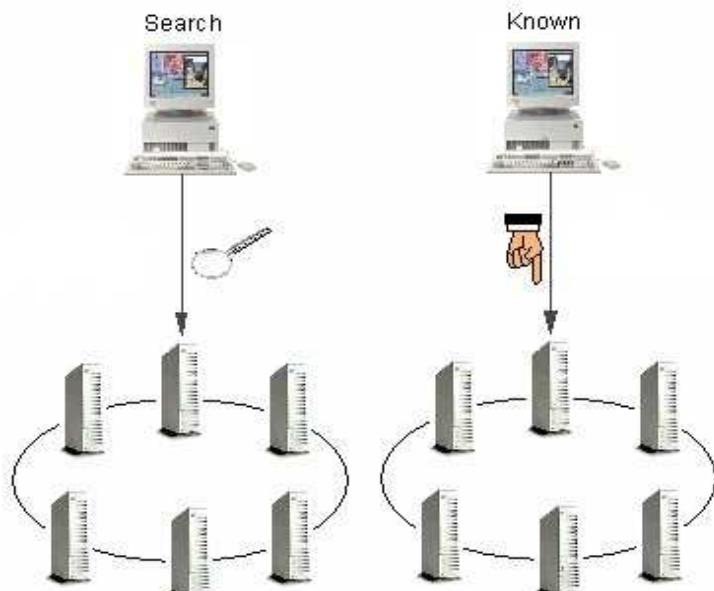


图 7.8 – “搜索网络”与“已知系统”的搜索（或探索）方式

可能在一些情况下，管理员并不希望客户端的用户搜索到网络中含有保密信息的数据库。DAS可以在实例级（Instance level）或者数据库级（database level）上做到这一点。详见图 7.9。

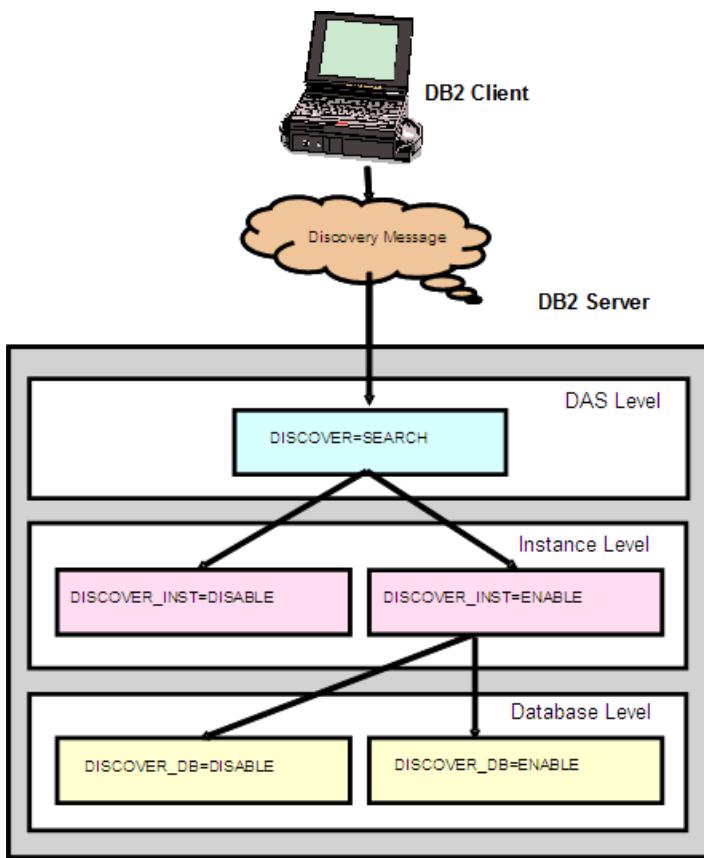


图 7.9 – 为搜索设置参数

如图 7.9 所示，您可以在不同的级别上允许或禁止搜索。在 DAS 级，您可以将 DISCOVER 赋值为 SEARCH 或 KNOWN。在实例级，DBM 配置参数 DISCOVER_INST 可以设为 ENABLE（允许）或 DISABLE（禁止）。这些参数的设定为您提供了所需的数据库搜索的精细程度。

手动配置与数据库的连接

使用这一方式，您将手动把主机名、端口号和数据库信息添加到配置助手中，随后会生成一条“catalog”命令来执行这些连接配置。配置助手将不会检查信息的正确性。只有当您无法连接到服务器时配置助手才会意识到错误的存在。所以您要确保用来连接到远程数据库的用户名和密码的正确性。在默认情况下，身份验证是在试图连接的 DB2 服务器端进行，所以，您必须提供该服务器上的用户名和密码。

7.2.3 建立客户端与服务器端概要文件

如果您正在配置大量的服务器（或者客户端），而又不想独立地配置每一个服务器端（或客户端）。您可以先设定一个服务器（或者客户端）的配置，然后将其导出为概要文件（即配置文件），再将此概要文件应用到其它的客户端或服务器上。这在设置运行环境时会节省管理员大量的时间。

如图 7.10 所示，要新建一个定制的概要文件，只需在配置助手中点击“配置（Configure）”菜单，然后选择“（导出概要文件=>定制）Export Profile => Customize”。

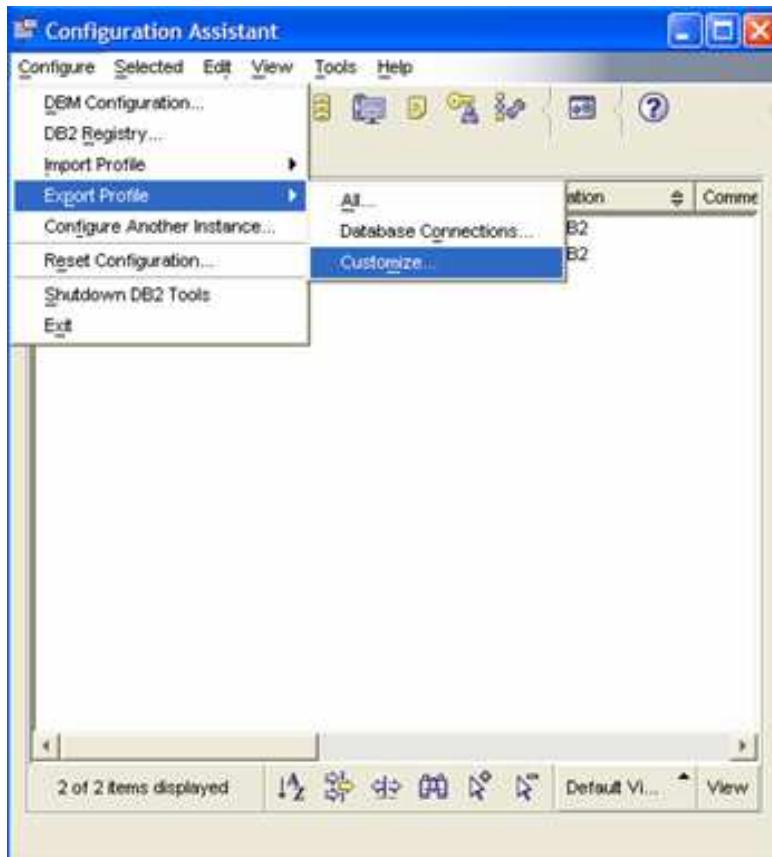


图 7.10 – 导出概要文件

图 7.11 所示为导出概要文件时需要填写的字段。

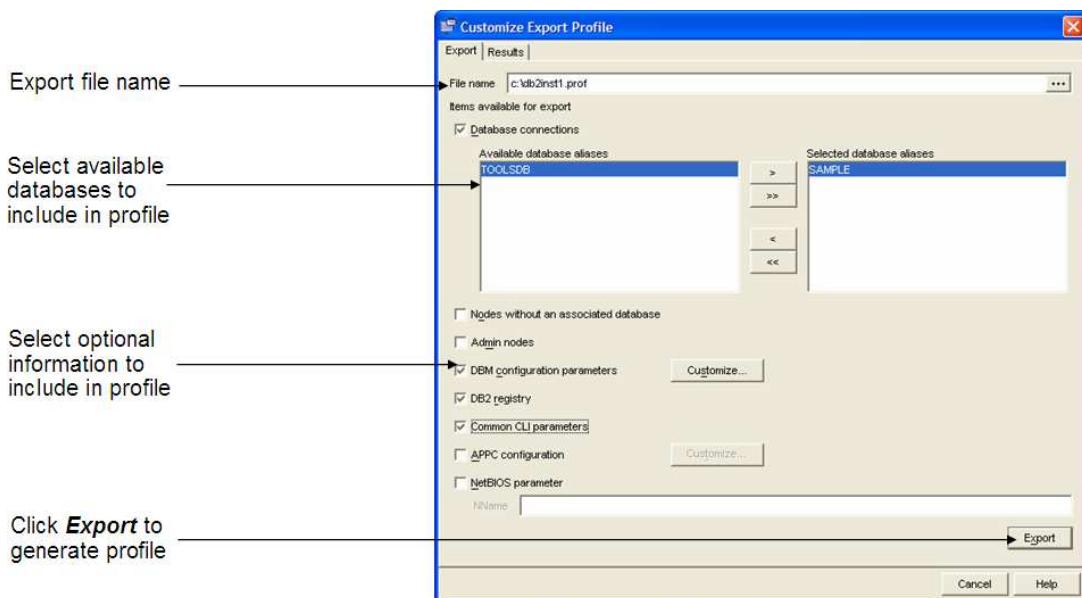


图 7.11 – 定制导出概要文件对话框

图 7.12 所示为点击定制概要文件 (Customize Export Profile) 对话框的“导出”按钮后的结果。

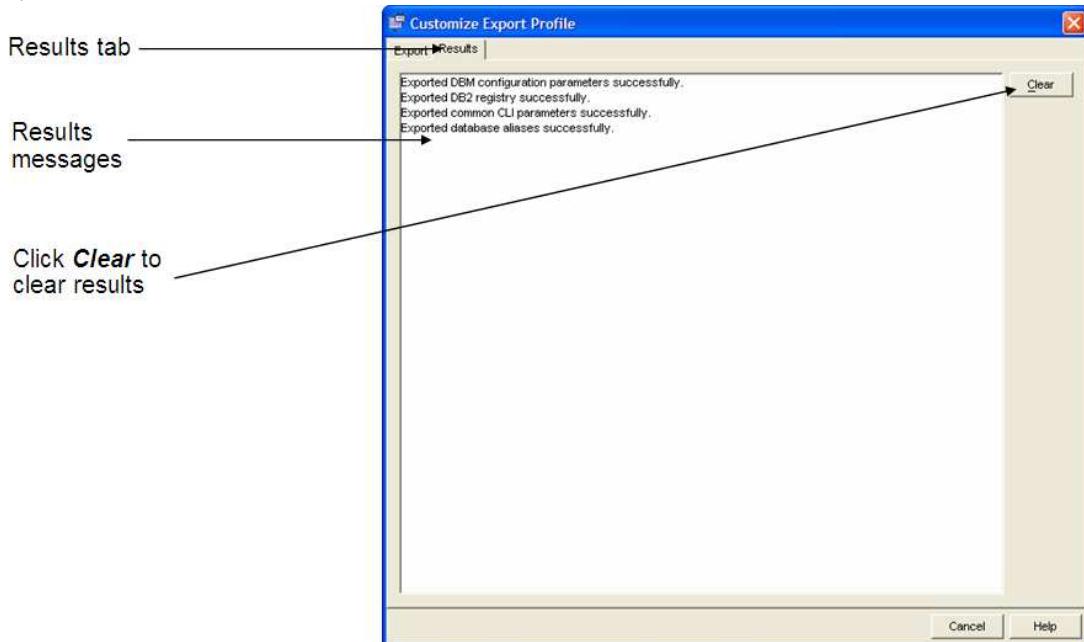


图 7.12 – 导出概要文件的结果

如要从配置助手导入一个定制概要文件，就点击配置菜单 (Configure)，然后选中“导入概要文件=>定制” (Import Profile => Customize)，如图 7.13 所示。

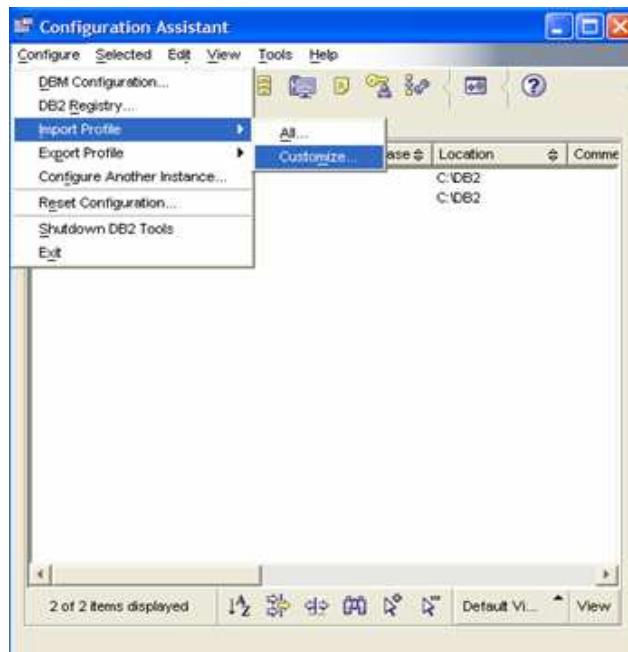


图 7.13 – 导入概要文件

图 7.14 所示为导入概要文件时需填写的字段。

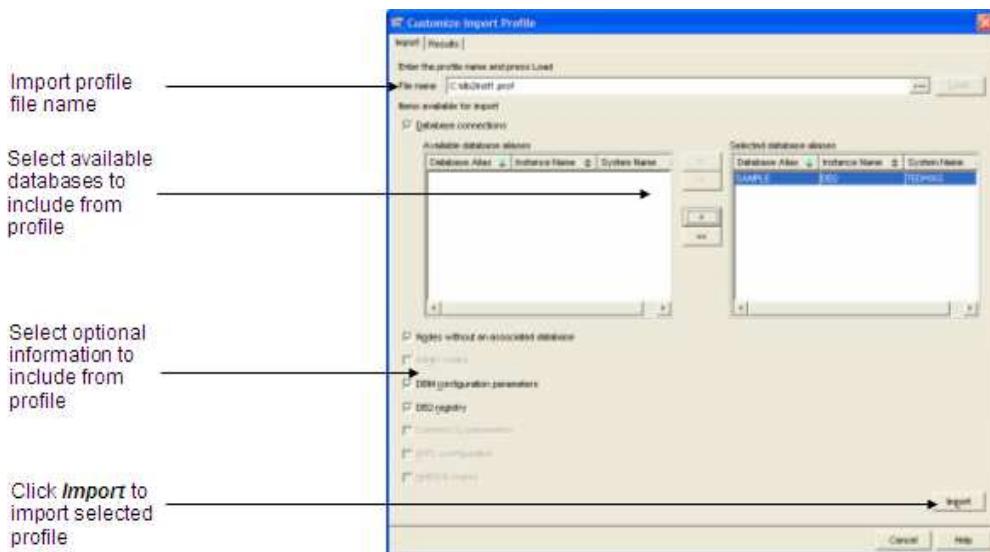


图 7.14 – 定制导入概要文件

7.3 小结

从数据客户端连接到服务器是一个关系型数据库管理的核心环节。在这一章中，我们从数据库的用途、内容和节点目录分析客户端与 DB2 之间的关系。

接下来，我们讨论了有关使用 GUI 配置助手来设置客户服务器连接，包括在连接两端时所需要的配置。

我们还介绍了通过“添加数据库向导”可以使服务器通过三种不同的方法来进行连接：使用已存储的概要文件、网络搜索（即 Discovery），或者手动键入服务器信息。客户端和服务器配置文件的建立包含了许多更详细的细节。

7.4 实验

配置助手可以用来简便地设置远程数据库连接。在本次实验中，您将把一个位于远程 DB2 服务器（用另一台工作站代替）上的数据加入目录，使用搜索网络和已知系统搜索两种方式添加目录。一经加入目录，您就可以象访问本地系统一样访问它。DB2 在后台完成所有的通信过程。

本实验假设您是工作在一个网络内。如果不是，您可以同时把您的计算机用作客户端和服务端，然后按照下面配置说明连接到您自己的系统。

实验过程

1. 向另一台工作站用户询问以下的信息：

远程数据库信息：

(PR)	Protocol 协议	<u>TCPIP</u>
(IP)	IP Address or hostname	<u></u>
	IP 地址或主机名	
(PN)	Instance Port Number 实例的端口号	<u></u>
(DB)	Database Name 数据库名	<u>SAMPLE</u>

提示：

- 在 Windows 系统命令行键入 hostname 来获取主机名
- 在 Windows 系统的命令行键入 ipconfig 来获取 IP 地址

2. 打开配置助手。 (提示: 可以从"开始"菜单访问)
3. 打开 "所选 (Selected)" 菜单并选择 "使用向导来添加数据库 (Add Database Using Wizard)"。
4. 在向导的 "源 (Source)" 页面中选择 "手工配置与数据库的连接 Manually Configure a Connection to a Database"。
5. 在向导的 "协议 Protocol" 页面, 选择 TCP/IP 选项。点击 "下一步" 按钮进入向导的下一页。
6. 在向导的"TCP/IP"页, 输入您在第(1)步中记下的完整的主机名或 IP 地址、端口号, 点击"下一步"按钮进入下一页。

注意: 仅在您的本地服务文件有一个与远程服务器上实例监听的端口号一致的端口号的条目时, 才可以用 "服务名称" 选项。当您使用这个选项时, DB2 将会在本机上而不是服务器上查找服务文件。如果您想要用此选项, 必须向本地服务文件中添加一个项。

7. 在向导的 "数据库 (Database)" 页上, 在 "数据库名称" 处输入您第 (1) 步记录的定义在远程服务器上的数据库名。注意 "数据库别名" 会自动填充相同的值。数据库别名通常是指本地应用程序用来与这个数据库连接时所使用的名称。既然您已经定义了一个叫作 "SAMPLE" 的数据库, DB2 将不会允许您在目录中以同样的名称登记一个数据库。因此, 您必须使用一个不同的别名。在这个例子中, 我们把数据库别名改为 SAMPLE1。有必要的话您可以添加一个注释。点击 "下一步" 按钮进入向导的下一页。
8. 在 "数据源 Data Source" 页上, 您可以选择把这个新的数据库 (数据源) 注册为一个 ODBC 数据源。它会自动地在 Windows 的 ODBC 管理器中为您进行注册。在这里, 取消 "为 CLI/ODBC 注册此数据库" 的勾选。点击 Next 按钮进入向导的下一页。
9. 在向导的 "节点选项 Node" 页, 指定数据库所在的远程服务所用的操作系统。因为这个实验中所有的工作站都使用微软的 Windows 系统, 所以确认您选中了下拉菜单里的 "Windows" 项。把这里的 "实例名" 字段设为 "DB2"。点击 Next 按钮进入向导的下一页。
10. 在向导的 "系统选项 System Options" 页, 您将会确认系统和主机名是否正确, 并核对操作系统的设置。点击 Next 按钮进入向导的下一页。
11. 向导的 "安全性选项 Security Options" 页面, 您可以指定用户认证在何处进行, 以及您想采用的认证方式。选中 "使用服务'DBM'配置中的认证值 Use authentication value in server's DBM Configuration"。这样就将采用在远程实例配置文件中 AUTHENTICATION 参数指定的认证方式。点击 "完成 Finish" 按钮, 将远程数据库加入目录并关闭向导。这时应该出现一个确认对话框。点击 "测试连接 Test Connection" 按钮来确认是否可以成功连接该数据库, 请确认您提供的定义在远程服务器上的用户名和密码是有效的 (前提是服务器的 AUTHENTICATION 参数值被设为 SERVER)。如果测试成功, 意味着您已成功地将远程数据库加入目录。否则, 需要返回向导确认所有指定的值都是正确的 (点击"修改"按钮回顾向导中的设置)。
12. 打开控制中心, 尝试查看新加入目录的远程数据库的各个表。
13. 回到配置助手, 尝试向目录中添加一个不同的数据库。这次选用 "搜索网络 Search the Network" 选项。在向导中采用与刚才同样的方式进行设置。注意, 在比较大的网络中, 搜索可能会耗较长的时间才返回结果。

8

第 8 章 – 数据库对象

本章讨论模式、数据表、视图、索引、序列等数据库对象。一些高级的数据库应用对象诸如触发器、用户定义函数和存储过程将在第 14 章讨论，SQL PL 存储过程、直接插入 SQL PL、触发器在第 15 章讨论。

注意：

更多关于数据库对象的信息，请参见以下视频：

<http://www.channeldb2.com/video/video/show?id=807741:Video:4242>

8.1 模式

模式是一组数据库对象的命名空间。它主要用于：

- 提供对象所有权的标识或某个应用的关系。
- 合理地将相关对象组合在一起。

所有的 DB 数据库对象标识名都可以一分为二，模式就是此名字的前半部分。

<模式名>.<对象名>

一个标识名称必须是独一无二的。在没有指定模式的情况下连接数据库并创建或引用数据库对象时，DB2 使用用户 ID 作为模式名称来连接数据库。例如，使用 arfchong 用户名连接 SAMPLE 数据库，然后用 CREATE TABLE 语句创建表：

```
CREATE TABLE artists ...
```

所建的数据库的标识名称即为 arfchong.artists。

您可以使用 SET SCHEMA 语句来声明一个会话模式。列表 8.1 提供了一个例子。

```
connect to sample user arfchong using mypsw
select * from staff ## This looks for arfchong.staff
set schema db2admin
select * from staff ## This looks for db2admin.staff
```

列表 8.1 – 一个使用 SET SCHEMA 语句的例子

一个“竞赛系统”能被用来说明模式的使用。假如一家单位正在举行一场让参加者创建各自的表并执行一些 SQL 操作的竞赛，所有的参加者都被给予了相同的 ID 来连接数据库，并且使用同样的脚本语言来创建表，到这里，所有的对象都还不正确的，因为他们没有一个模式名。在选手登入竞赛系统后系统根据时间戳生成一个模式名。这样虽然选手 A 会和选手 B 操作相同名字的表，但是由于是在不同的模式下，所以在他们操作的过程中不会发生冲突。

8.2 公用同义词（或别名）

DB2 9.7 中新出现的是公用同义词的概念，也被称为公用别名。公用同义词允许您在不指定一个模式的情况下引用对象。列表 8.2 提供了一个实例。

```
connect to sample user arfchong using mypsw
create public synonym raul for table arfchong.staff
select * from raul
select * from arfchong.raul ## Error
connect to sample user db2admin using psw
select * from raul
```

列表 8.2 一个公用同义词的例子

在列表 8.2 中，您先连接 arfchong 用户并参照表 arfchong.staff 创建公用同义词 raul。同义词本身不使用模式。如果您想使用模式，您将会得到一个错误。其他用户像列表 8.2 实例中 db2admin 也可以使用公用同义词 raul。

在这个实例中，如果关键字 public 没被使用，被创建的同义词将变成一个私有同义词。在列表 8.3 中，让我们仔细观察相同例子，但是用的是私有的同义词。

```
connect to sample user arfchong using mypsw
create synonym raul for table arfchong.staff
select * from raul
select * from arfchong.raul ## OK, it also works
connect to sample user db2admin using psw
select * from raul ## Error, cannot find db2admin.raul
select * from arfchong.raul ## OK, this works
```

列表 8.3 一个私有同义词的例子

在列表 8.3 中，您可以发现由于同义词是私有的，所以在没有指定模式的情况下，它是不能被其他连接着的用户所参考的。

8.3 表

表是一组由相关数据逻辑安排的行和列。下面是一个用 CREATE TABLE 语句创建表的例子。

```
CREATE TABLE artists
  (artno SMALLINT not null,
   name VARCHAR(50) with default 'abc',
   classification CHAR(1) not null,
   bio CLOB(100K) logged,
   picture BLOB(2M) not logged compact
  )
IN mytbls1
```

列表 8.4 一个使用 CREATE TABLE 语句的例子

在接下来的部分，我们将说明这个 CREATE TABLE 语句的主要部分。

8.3.1 数据类型

图 8.1 列出 DB2 支持的数据类型。

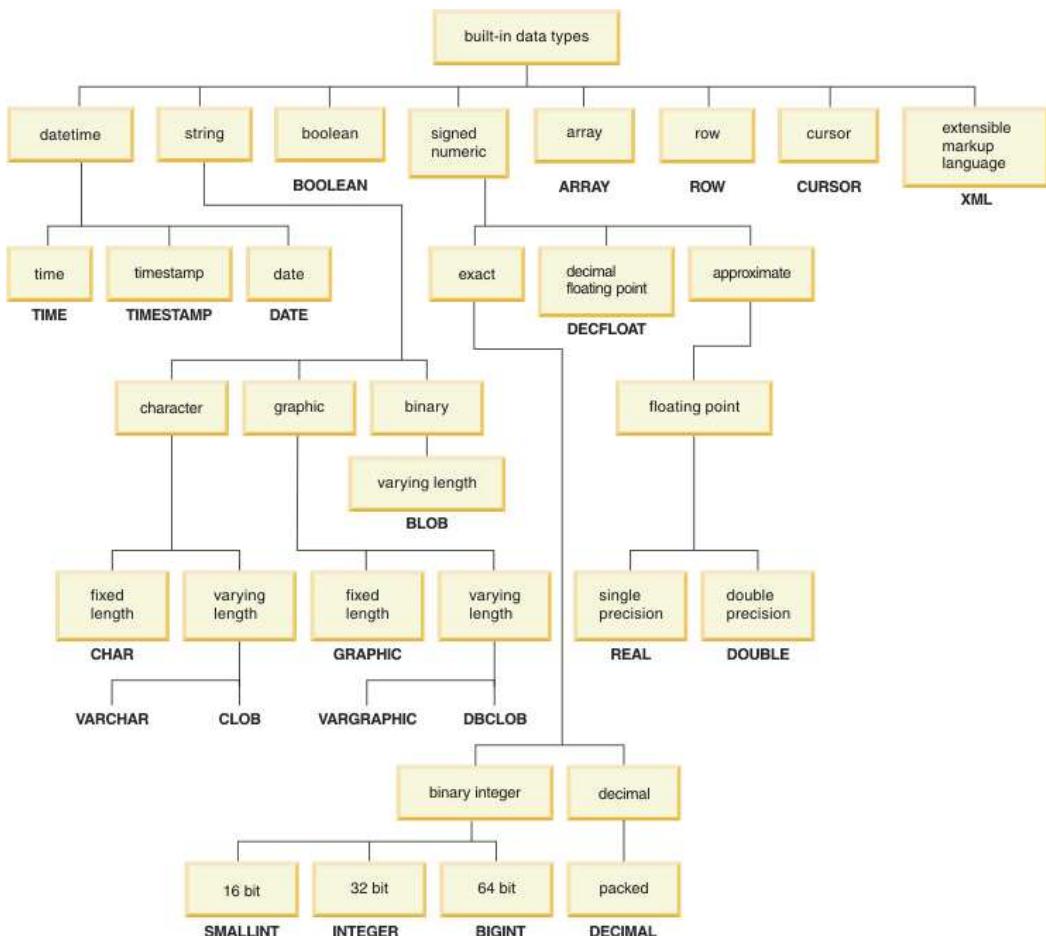


图 8.1 – DB2 内置数据类型

图8.1中出现的数据类型在DB2帮助文档中有详细的说明。它们中的绝大多数在其他相关数据库管理系统中是常见或是非常相似的，所以在这里就不多介绍了。另一方面，一些数据类型对初学者来说比较陌生，比如说大对象数据类型——LOB。

大对象数据类型用来存储大字符串、大二进制串或文件，如图8.2所示。

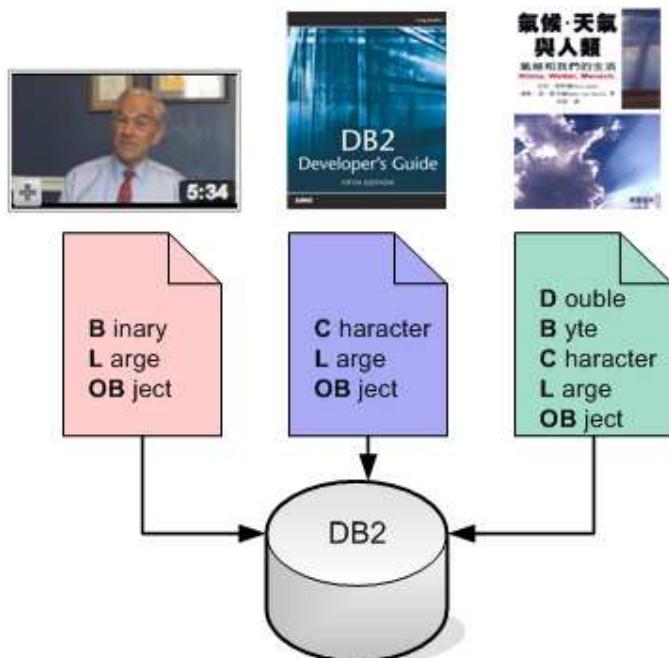


图 8.2 – LOB 数据类型

这些大对象二进制类型常简写为下面的形式：二进制大对象简写为 BLOB，字符大对象简写为 CLOB，双字节字符大对象简写为 DBCLOB。

**new in
V9.7**

图8.1也列出了在DB2 9.7中新增的数据类型：

- BOOLEAN
- ARRAY
- ROW
- CURSOR

这些数据类型是Oracle数据库服务器的数据类型的一部分，现在DB2也支持他们了。Oracle数据库服务器的数据类型稍后将在本章中详细讨论。

8.3.1.1 用户定义类型

DB2 允许您使用内置数据类型定义自己的数据类型，它们称为用户定义类型（UDT），UDTs 常出现在以下情况：

- Distinct 类型
- Structured 类型
- Reference 类型
- Array 类型
- Row 类型
- Cursor 类型

**new in
V9.7**

Reference、array、row和cursor类型是在DB2 9.7中新出现的并且使用在一般的SQL过程中。Distinct这个用户自定义类型是建立在内置数据类型上的，这些UDTs在以下情况下是有用处的：

- 需要为值建立上下文
- 需要 DB2 用强制类型执行数据类型

列表 8.5 中的语句序列阐述了怎样和何时使用 UDT:

```
CREATE DISTINCT TYPE POUND AS INTEGER WITH COMPARISONS
CREATE DISTINCT TYPE KILOGRAM AS INTEGER WITH COMPARISONS
CREATE TABLE person
(f_name      VARCHAR(30),
 weight_p    POUND NOT NULL,
 weight_k    KILOGRAM NOT NULL )
```

列表 8.5 一个使用 distinct 类型的例子

在这个例子中，有两个 UDT 被创建：POUND 和 KILOGRAM。这两个都是基于 INTEGER 内置类型。WITH COMPARISONS 子句表明作用于该数据类型的比较转换函数也将会被分别创建。

表 person 在列 weight_p 和 weight_k 分别使用这两个新的 UDT。如果我们使用以下语句。

```
SELECT F_NAME FROM PERSON
WHERE weight_p > weight_k
```

您会得到一个错误信息，因为两个正在作比较的列使用的是不同的数据类型。即使 weight_p 和 weight_k 分别使用 POUND 和 KILOGRAM，并且两个 UDT 均是基于 INTEGER 数据类型，这种类型的比较也是不可行的。这其是正是我们所对面要的，因为在现实生活中，磅和公斤之间的比较会有什么意义呢？这种比较不具有任何意义。

在接下来的例子中，您可能想将 weight_p 和一个触发器进行比较，但是这两个数据类型是不同的，因此除非使用一个转换函数，否则也将会收到一个错误信息。

正如下面的语句所示，进行我们想要的比较，要使用转换函数 POUND() 进行转换。正如早先说明的，当调用 CREATE DISTINCT TYPE 语句的 WITH COMPARISONS 子句时 POUND() 转换函数与 UDT 一同被创建。

```
SELECT F_NAME FROM PERSON
WHERE weight_p > POUND(30)
```

**new in
V9.7**

8.3.1.2 Oracle 数据库服务器数据类型

下面这些在 Oracle 数据库服务器中使用的数据类型现在 DB2 也能支持：NUMBER、VARCHAR2、TIMESTAMP(n)、“DATE”、BOOLEAN、INDEX BY、VARAY、Row Type、Ref Cursor。要使用这些数据类型，您首先需要像下面这样启用 DB2_COMPATIBILITY_VECTOR 这个注册表变量：

```
db2set DB2_COMPATIBILITY_VECTOR=FF
db2stop
db2start
```

一旦这个注册变量被启用，新的数据库就能支持这些数据类型。其中一些类型只有在有使用 SQL 过程的上下文中有效。

注意：

如果您使用的是支持 SQL 兼容（见第二章）的 DB2 版本，在使用了 PL/SQL 的地方，您也

可以用这些数据类型。在这种情况下，应该把上面例子中对“DB2_COMPATIBILITY_VECTOR”变量的赋值从FF改成FFF。



8.3.1.3 隐式转化或弱类型

很多动态语言，比如Ruby on Rails、PHP，都允许隐式转化，但是在DB2中由于其强类型机制的要求，这变成了一个难题。在DB2 9.7版本下，规则已经放宽，隐式转化或弱类型都允许出现。这意味着，例如，您现在可以将一个字符串赋值给一个numeric类型，或者用字符串类型与numeric类型进行比较，如下面这个例子：

```
create table t1 (col1 int)
select * from t1 where col1 = '42'
```

在这个例子中，字符串“42”现在可以与int类型的列col1进行比较。

此外，在DB2 9.7中现在允许您指定无类型参数或者无类型空值。而在以前，您不得不把他们明确的转化成相对应的类型。现在，下面这条语句可以顺利执行：

```
select ?, NULL, myUDF(?, NULL) from t1
```

8.3.1.4 零值（Null）

一个零值代表一个未知的状态。不过，CREATE TABLE 语句可以使用 NOT NULL 子句定义一列。这就保证了该列都包含已知的数据值。如果某列定义了 NOT NULL，您也可以为该列赋予一个默认值。如下示例：

```
CREATE TABLE Staff (
    ID      SMALLINT NOT NULL,
    NAME    VARCHAR(9),
    DEPT   SMALLINT NOT NULL with default 10,
    JOB    CHAR(5),
    YEARS  SMALLINT,
    SALARY DECIMAL(7, 2),
    COMM   DECIMAL(7, 2) with default 15
)
```

在这个例子中，ID和DEPT这两列被声明为NOT NULL，DEPT列同时被声明当没有值被赋予的时候，使用默认值10。

8.3.2 标识列

一个标识列就是一个可以为每个插入行自动产生唯一值的数字列。每个表的标识列不得多于一个。

有两种根据定义为标识列产生数值的方法：

- **常规产生（Generated always）：** 值常被 DB2 产生。应用程序不允许提供确切值。
- **默认产生（Generated by default）：** 值可以明确地由应用程序提供，或者应用程序不提供时由 DB2 产生。DB2 不能保证唯一性。这个选项是为生成数据、卸载或重载表提供的。

让我们看一看接下来的例子：

```
CREATE TABLE subscriber(subscriberID INTEGER GENERATED ALWAYS AS
                        IDENTITY (START WITH 100
                                  INCREMENT BY 100),
                        firstname VARCHAR(50),
                        lastname  VARCHAR(50) )
```

在这个例子中，列 `subscriberID` 是 `INTEGER` 类型，被定义为标识列并且是常规定义。其值将从 100 开始，按增量 100 增加。

8.3.3 序列对象

虽然序列对象的概念和表的概念是独立的，但是因为他们的工作方式与标识列的是相似的，所以也放在这一节讨论。跟标识列不一样的是，序列对象的取值在整个数据库唯一，而标识列在一个表中唯一。下面提供一个例子：

```

CREATE TABLE t1 (salary int)

CREATE SEQUENCE myseq
    START WITH 10
    INCREMENT BY 1
    NO CYCLE

INSERT INTO t1 VALUES (nextval for myseq)
INSERT INTO t1 VALUES (nextval for myseq)
INSERT INTO t1 VALUES (nextval for myseq)

SELECT * FROM t1

SALARY
-----
10
11
12
3 record(s) selected.

SELECT prevval for myseq FROM sysibm.sysdummy1

1
-----
12
1 record(s) selected

```

列表 8.6 一个序列的例子

`PREVVA` 提供序列的一个当前值，而 `NEXTVAL` 提供下一个值。上面的例子也使用 `SYSIBM.SYSDUMMY1`。这是一个包含一行一列的系统目录表。如果一个查询要求只得到一个输出，可以使用该表。系统目录表在下节进行阐述。

8.3.4 系统目录表

每个数据库都有它自己的系统目录表和视图。它们存储关于数据对象的元数据。您可以像使用普通的数据库表一样查询这些表。有三种模式用来识别系统目录表：

- **SYSIBM:** 基本表，对 DB2 使用进行最优化
- **SYSCAT:** 基于 `SYSIBM` 表的视图，对平常轻负载使用进行优化
- **SYSSTAT:** 数据库分析

下面是一些目录视图的例子

- `SYSCAT.TABLES`
- `SYSCAT.INDEXES`

- SYSCAT.COLUMNS
- SYSCAT.FUNCTIONS
- SYSCAT.PROCEDURES

8.3.5 已声明全局临时表 (DGTTs)

已声明全局临时表是内存中创建的表，供应用程序使用，当应用程序终止时就会自动删除。这些表只可以被创建它们的应用程序访问。它们在 DB2 目录表中并不存在相应记录条目。由于没有目录的争夺，没有行的锁定，也没有默认日志（日志是可选的）和没有权限检查，存取这些表是非常高效的。临时表支持索引，可以在临时表上建立标准的索引，也可以在临时表上运行 RUNSTATS 收集统计信息。

已声明全局临时表创建在用户的临时表空间中，这个空间必须在创建任何已声明临时表之前被创建。以下语句创建 3 个已声明全局临时表。

```
CREATE USER TEMPORARY TABLESPACE apptemps
    MANAGED BY SYSTEM USING ('apptemps');

DECLARE GLOBAL TEMPORARY TABLE employees
    LIKE employee NOT LOGGED;

DECLARE GLOBAL TEMPORARY TABLE tempdept
    (deptid CHAR(6), deptname CHAR(20))
    ON COMMIT DELETE ROWS NOT LOGGED;

DECLARE GLOBAL TEMPORARY TABLE tempprojects
    AS ( fullselect ) DEFINITION ONLY
    ON COMMIT PRESERVE ROWS NOT LOGGED
    WITH REPLACE IN TABLESPACE apptemps;
```

列表 8.7 DGTT 的操作

当一个已声明全局临时表被创建，它的模式必须被指定为 SESSION。用来创建临时表的用户 ID 必须具有在表上所有的权限。每个创建临时表的应用程序都具有它自己的独立拷贝，如图 8.5 所示。



图 8.3 DGTTs 的作用域

列表 8.8 阐明了已声明全局临时表的范围限制，它假定用户临时表空间已经给定。
在 DB2 命令窗口 #1:

```

db2 connect to sample
db2 declare global temporary table mydgtt (col1 int, col2 varchar(10))
on commit preserve rows
db2 insert into session.mydgtt values (1,'hello1'),(2,'hello2'),
(3,'hello3')
db2 select * from session.mydgtt
COL1 COL2
-----
1 hello1
2 hello2
3 hello3
3 record(s) selected.
在 DB2 命令窗口 #2:
db2 connect to sample
db2 select * from session.mydgtt
SQL0204N "SESSION.MYDGTT" is an undefined name. SQLSTATE=42704

```

列表8.8 DGTT的作用域

从**列表8.8**中，您可以发现，当您尝试在第二个会话（DB2 命令窗口 #2）使用 SESSION.MYDGTT 时，您会得到一个错误，因为这个DGTT没有在这个会话中声明。请注意：因为在DB2 命令窗口里每个输入的语句都会有这种默认值， DGTT 定义是使用 ON COMMIT PRESERVE 子句的。

8.3.6 创建全局临时表 (CGTTs)

虽然DGTTs允许您创建一个临时表，但是这个表不能被不同的连接或者不同的会话共享。每当一个会话被建立，DECLARE GLOBAL TEMPORARY TABLE语句就会被执行。使用CDTTs，临时表定义只需要创建一次，因为一旦被创建，它将会在DB2目录中永久储存。这意味着其他连接不必再创建它，可以直接使用该表了。尽管表结构可以被直接使用，但是对于不同的连接，表中的数据是互相独立的，并且当连接断开的时候，数据也会随之消失。举个例子，让我们看看**列表8.9**，假定临时表空间已经创建好了。

```

在 DB2 命令窗口 #1:
db2 connect to sample
db2 create global temporary table mycgtt (col1 int, col2 varchar(10))
on commit preserve rows
db2 insert into mycgtt values (1,'hello1'),(2,'hello2'), (3,'hello3')
db2 select * from mycgtt
COL1      COL2
-----
1 hello1
2 hello2
3 hello3
3 record(s) selected.
在 DB2 命令窗口 #2:
db2 connect to sample
db2 select * from mycgtt
COL1      COL2
-----
0 record(s) selected.

```

列表 8.9 CGTT 的作用域

在**列表 8.9**中，我们看到在DB2命令窗口#2（即另一个会话或者连接）中，不需要再创建 CGTT，您可以简单地引用它。但是没有行被返回，因为第一个会话的数据是它专有的。

8.4 视图

视图是表中数据的一种表现形式。视图中的数据不是单独存储在一起的，但当视图被调用时就可以被获得。嵌套视图是一基于其它视图创建的视图。所有有关视图的信息保存在DB2 的目录视图中: SYSCAT.VIEWS, SYSCAT.VIEWDEP 和 SYSCAT.TABLES。列表 8.10是一个创建视图和使用视图的例子:

```
CONNECT TO MYDB1;

CREATE VIEW MYVIEW1
  AS SELECT ARTNO, NAME, CLASSIFICATION
    FROM ARTISTS;
SELECT * FROM MYVIEW1;
Output:
ARTNO          NAME           CLASSIFICATION
-----          -----
10            HUMAN          A
20            MY PLANT        C
30            THE STORE       E
...
```

列表 8.10 视图的使用

8.5 索引

索引是有序键值的集合，每一个键值指向表的一行。索引的值可以唯一，它改善了数据库的性能。在索引上可以定义如下的一些特性:

- 索引顺序可以递增也可以递减。
- 索引键可以是唯一值的，也可以不唯一。
- 一些列可以一起用作索引（这被称作混合索引）。
- 如果索引和物理数据串聚在一个相似的索引序列中，它们就成为簇索引。

例如:

```
CREATE UNIQUE INDEX artno_ix ON artists (artno)
```

8.5.1 Design Advisor

Design Advisor 是在给定SQL 工作量的前提下如何优化数据库设计的绝好工具。Design Advisor 可以帮助设计索引、物化查询表 (MQT)，多维集群 (MDC) 和数据库分区特性。Design Advisor 可以从控制中心中调用，右键点击数据库并如图8.4所示，选择Design Advisor。

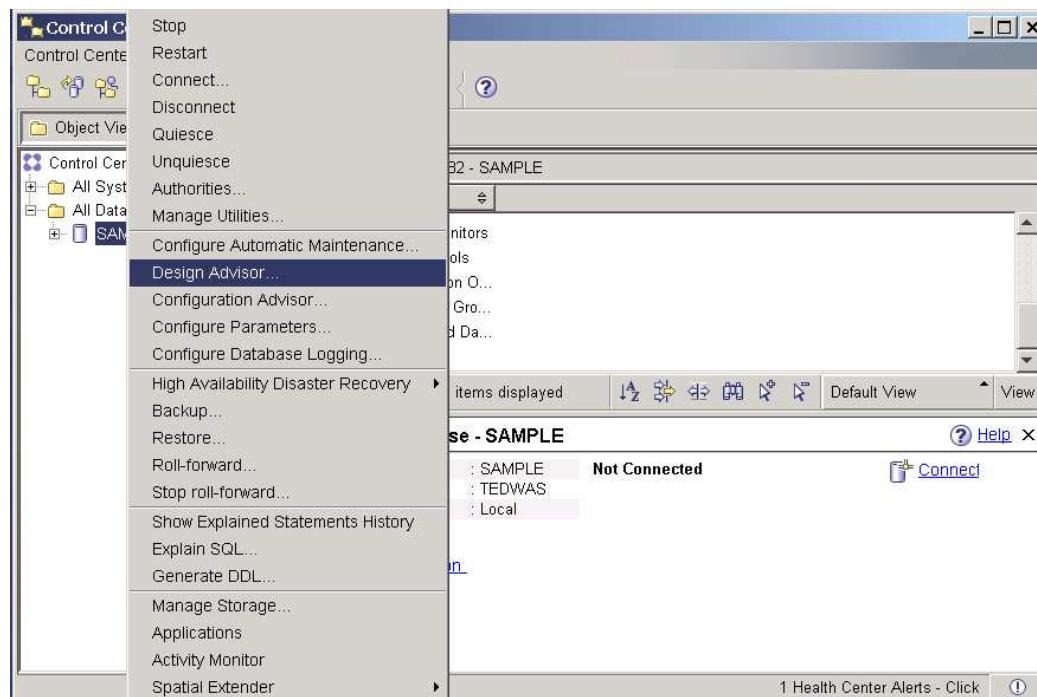


图8.4从控制中心运行Design Advisor

图 8.5 展示了 Design Advisor。在 DB2 中，跟随向导获得设计建议。

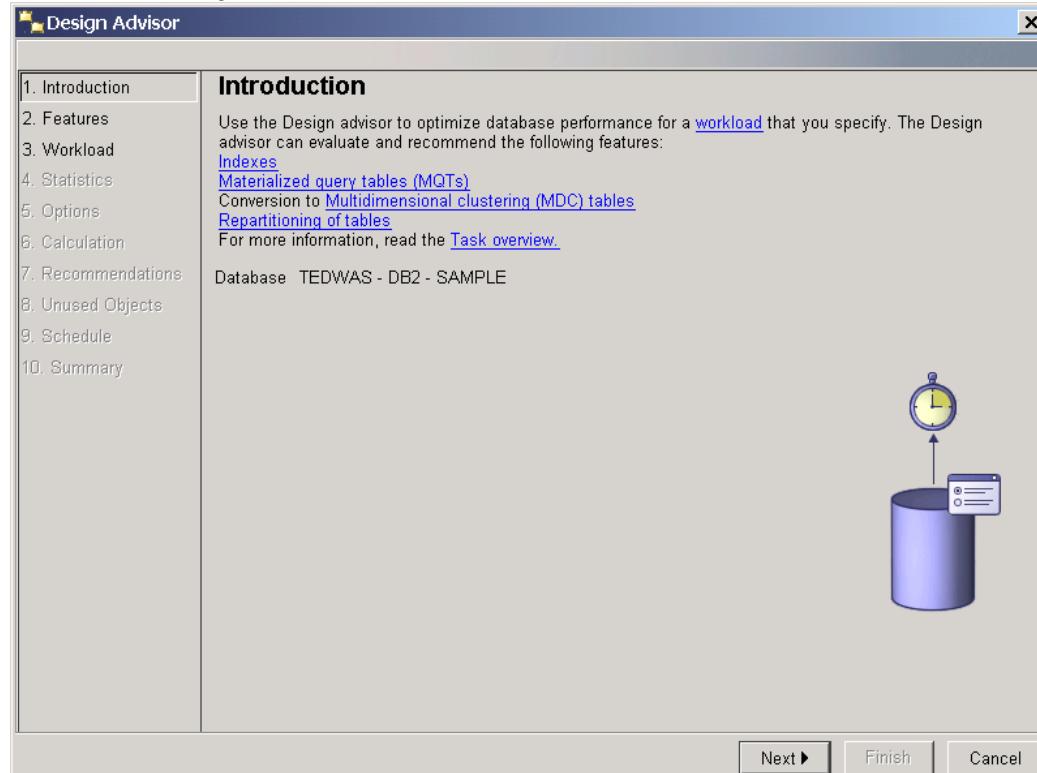


图8.5 Design Advisor

8.6 参照完整性

参照完整性允许您管理数据库表之间的关系。例如可以建立如图8.8 所示的父子表关系。在这个图中，有两个通过部门号相关联的表，DEPARTMENT 和 EMPLOYEE。EMPLOYEE 表的WORKDEPT 列仅能包含已经存在于DEPARTMENT 表中的部门号。这个关系中，DEPARTMENT 是父表，EMPLOYEE 表是子表或者依赖表。图中也显示了EMPLOYEE 表的CREATE TABLE 语句必须包含这种关系的定义。

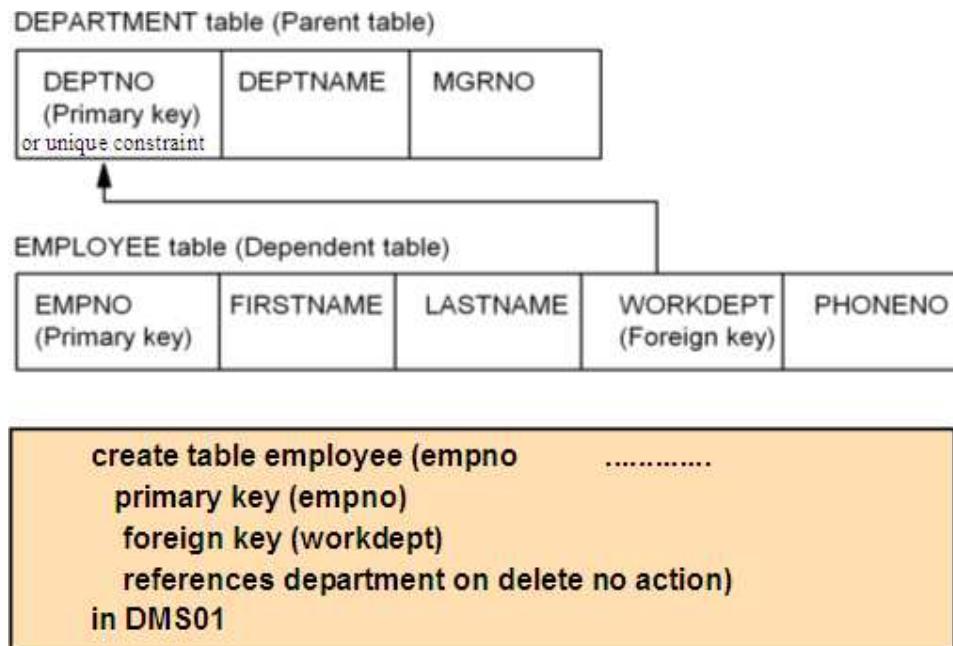


图8.6 参照完整性的示例

关于参照完整性，常用概念如表8.1所示。

概念	描述
父表	父键所在的数据表
从属表	依赖于父表中的数据的表，它含有外键。对于从属表中的任一行数据，在父表中都有与之相匹配的行数据存在。
主键	定义在父表上的主键，它不可以是空值（NULL）而且所有的值必须唯一。主键可以由表的一个或多个列组成。
外键	表中的一列或几列，这些列的值必须只要与父表中的一个主键值或唯一键值匹配。

表8.1 参照完整性关键字概念

表中的数据可以与一个或多个表中的数据建立引用完整性。数据的取值可以附加约束，使他们符合一定的财产或业务规则。举个例子，如果一个表属性存储性别，可以强制约束的允许值只有“M”（男性）或者“F”（女性）。

new in
V9.7

8.7 模式演变 当业务需求改变时，相应的信息技术的基础设施和系统结构也必须随之改变。也就是说，现有的数据库可能需要新建、修改或者删除一些表，一些触发器的逻辑也需要改变等等。虽然看上去这些操作是简单的，但是事实上有可能是相当困难和繁琐的。因为要实施复杂和带风险性的操作，应用程序要有长期维护的窗口。以前要实现这些变化困难的一个原因是，DB2要求所有

对象必须时刻保持逻辑一致。如果您想改变的对象A，但是有对象B依赖于A，那么单单改变A是不允许的，或者导致删除B。图8.7提供了一个案例。

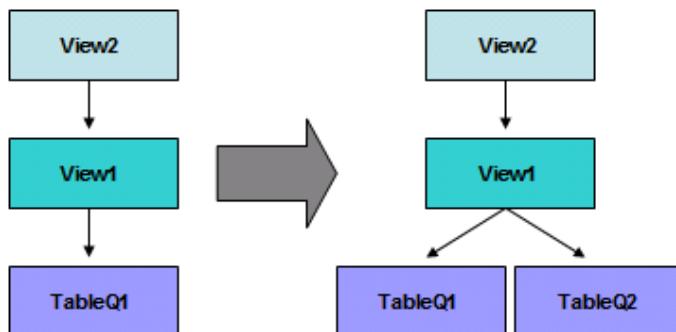


图8.7 模式升级的案例

如图8.7所示，为了使View1（视图）从仅仅参考TableQ1（表）改变到同时参考TableQ1和TableQ2，您需要删除View1，然后重新定义它。但是View2是依赖于View1的，在DB2 9.7以前，DB2不允许您删除View1，因为存在View1的依赖对象View2。所以您不得不先删除View2，然后才能删除View1，最后重新定义它们。DB2 9.7 规则已经放宽了，允许改变存在依赖对象的对象。不过在依赖对象（本例中是View2）被使用前必须重新生效，当然，为了提供安全性，这是自动执行的。这就是所谓的自动重新生效。您可以使用数据库配置参数AUTO_REVAL来打开或者关闭这个功能，甚至制定何时进行重新生效。例如，如果您设置这个参数值为DEFERRED_FORCE，重新生效将被推迟，直到无效对象或者依赖对象被访问，但是CREATE语句是允许的（有一个警告）即使依赖对象并不存在。

其它会影响依赖模型的变化包括一些功能性的操作，像对视图、函数、存储过程、触发器、别名等的CREATE 和 REPLACE 语句，例如：

```
create or replace procedure p1 begin ... end
```

在这个语句中，如果一个对象（在本例中是P1）不存在，它就会被创建；如果它已经存在，就用新的来替换它。已经存在并被替换的情况就涉及到依赖性。当P1被替换的时候，依赖P1的对象会进行自动重新生效。类似的情况在新属性RENAME COLUMN或者用ALTER COLUMN更改数据类型时也会发生，都已经能够支持所涉及的依赖性问题。

还有一个相关的概念是：软失效（soft invalidation）能让用户删除那些正在被其他事务使用的对象，其他事务对已经删除对象的访问被禁止。

8.8 小结

在本章中，我们了解了DB2中的数据库对象，包括：它们是什么，怎么创建它们和怎么样使用它们。我们介绍了数据库的模式并用公用同义词进行比较，同时我们还比较了公用同义词和私有同义词的区别。

接着，我们进一步讨论了表和相关元素构成：数据类型（内置类型和用户自定义类型）、标识列、序列对象和全局临时表。然后我们了解了视图、索引和如何使用配置助手来改善对表数据访问和恢复性能。

最后，我们研究了怎么使用参照完整性来定义表中的关系和一个新概念——模式演变，它能方便的改变数据对象。

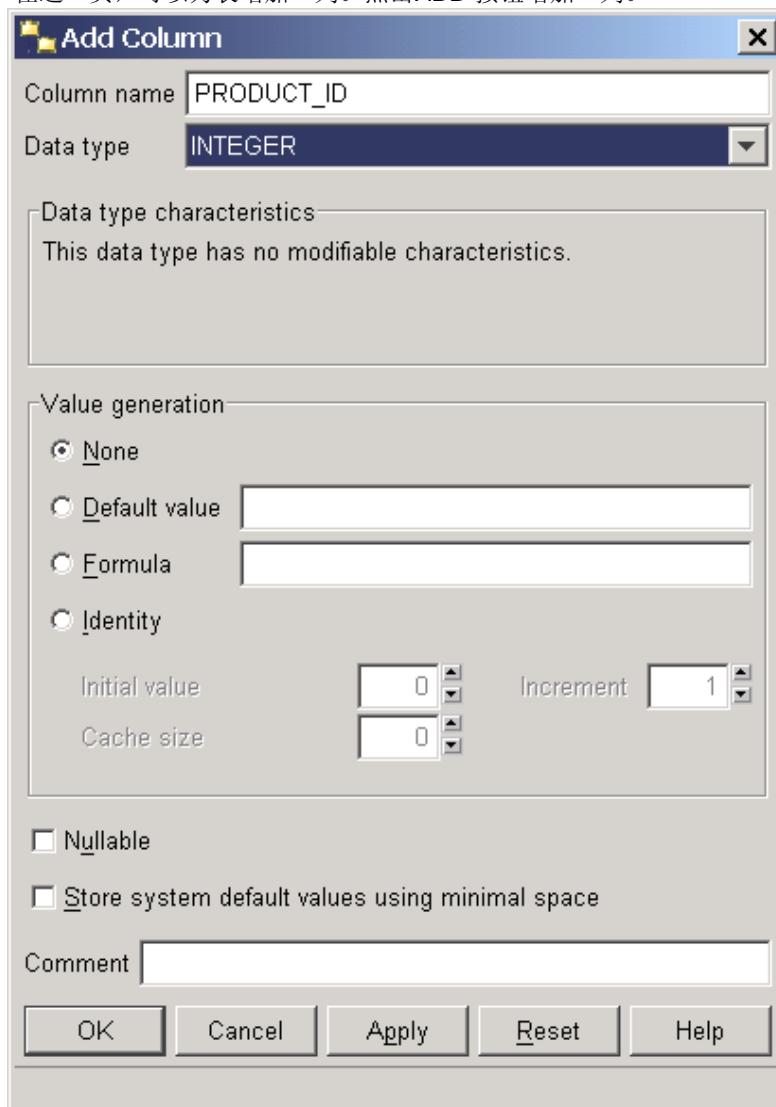
8.9 实验

到目前为止，您已经使用过SAMPLE 数据库中现有的数据库。本实验，您需要创建自己的数据库和表。您可以使用创建数据表向导（Create Table Wizard）在 SAMPLE 数据库中创建两个新表。

实验过程

- 按本章前面介绍的方法，打开Create Table Wizard (*Control Center -> All Databases ->SAMPLE -> (右键点击) Tables object -> Create*)

2. 定义表名，列定义和其它的约束。这个表存储在SAMPLE 数据库中，记录项目的办公用具采购信息。每次采购，表中都相应该增加一行。这个表有六列：
 - product_id: 被购买商品的唯一标识
 - description: 商品描述
 - quantity: 采购量
 - cost: 花费
 - image: 所购买商品的图片（可为空）
 - project_num: 购买这些商品的目标工程
3. 在这个向导的第一页，在模式名中，输入当前登录的用户ID，并且使用表名：SUPPLIES。您也可以输入注释。点击Next按钮继续向导的下一页。
4. 在这一页，可以为表增加一列。点击ADD按钮增加一列。



5. 输入列名“product_id”，并选择数据类型：INTEGER。Nullable 置空，点击Apply按钮来定义此列。
6. 使用下图所示的选项重复这一步骤完成此表中其它列。当所有的列都已经添加完成，点击OK按钮，得到刚才所创建列的总结。点击Next按钮继续向导的下一步。

列名	列属性
product_id (completed)	INTEGER, NOT NULL
description	VARCHAR, length 40, NOT NULL
quantity	INTEGER, NOT NULL
cost	DECIMAL, Precision 7, Scale 2, NOT NULL
image	BLOB, 1MB, Nullable, NOT LOGGED
project_num	CHAR, length 6, NOT NULL

注意：当声明LOB 列时选项NOT LOGGED 可以被指定。它强制约束列可以大于1GB。对大于10MB 的LOB 进行更改会很快填满日志文件，这种情况下，常常推荐使用此选项。即使使用NOT LOGGED 选项，事务过程中对LOB 类型的更改仍可以成功回滚。同时注意image 列是唯一一个被定义为Nullable 的列。您认为为什么这个列要这样定义呢？

7. 此时，提供创建表的所有的必要信息都已提供。可以直接跳过其它的页，以使用系统的默认选项。表建成以后也可以增加键和约束。点击Next 按钮继续向导的下一步。
8. 为表增加一约束条件来限制quantity 列的值。在向导的Constraint 页，点击ADD 按钮。在Check Name 部分输入：valid_quantities。在Check Condition 部分输入：quantity > 0。点击OK 按钮。就可以在向导的Constraint 页看到刚才所设置的约束信息列表。
9. 继续操作向导来改变表的其它参数。也可以直接跳到Summary 页或简单地点击Finish 按钮来创建表。
10. 在控制中心，点击SAMPLE 数据库下的Tables 文件夹。刚才所创建的表现在可以在列表中找到。有时候需要刷新控制中心来查看其中的改变。
11. 现在让我们用示例数据库中的staff 表来测试一下隐式转化。尝试一下内容：

```
C:\>db2 describe table staff
注意ID列的类型是SMALLINT
```

```
C:\>db2 select * from staff where id = '350' --> Note '350' is a string
C:\>db2 select * from staff where id = 350 --> Note 350 is a number
```

这两种情况返回的应该都是：

ID	NAME	DEPT	JOB	YEARS	SALARY	COMM
350	Gafney	84	Clerk	5	43030.50	188.00

在第一个查询中“350”是字符串类型，DB2隐式转化成了数字型(SMALLINT)。

9

第 9 章 – 数据迁移工具

本章介绍的工具或者命令用于在不同情况中的数据迁移，这些情况包括在同一个数据库中的数据迁移，同一个平台上不同数据库的数据迁移以及在不同平台之间的数据迁移。图 9.1 展示了这些数据迁移工具。

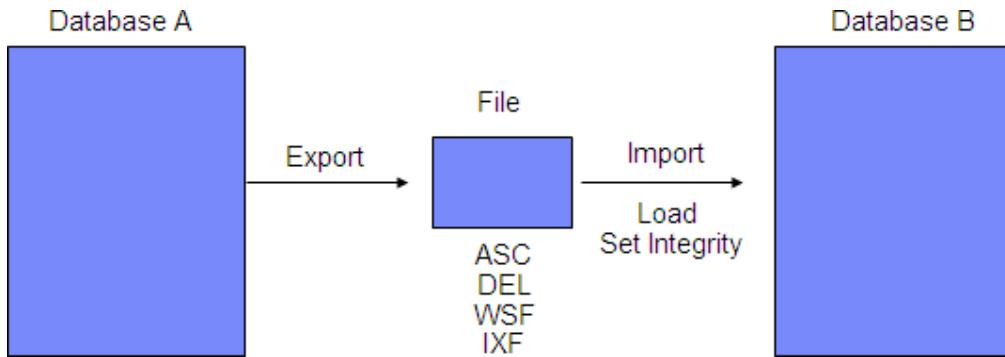


图 9.1 数据迁移工具

图 9.1 中有两个数据库，分别为 A 和 B，使用 Export 工具，可以将数据库中的表导出为文件，导出的文件格式可以为：

ASC = ASCII

DEL = Delimited ASCII

WSF = Worksheet format

IXF = Integrated Exchange Format

ASC 和 DEL 格式的文件是文本文件，可以用任何文本编辑器打开。WSF 格式的文件可以将数据迁移到电子表格软件中，例如 Excel, Lotus® 1-2-3。IXF 格式文件包括了数据表的数据描述语言（DDL）和里面的数据。使用 IXF 格式是非常方便的，利用它可以重建数据表，而其他格式则没有办法这么做。

当数据导出到文件后，使用 Import 可以将数据由文件导入到数据表中。如果使用 ASC, DEL 和 WSF 格式的文件作为中间文件，在它们导入之前数据表必须存在。而使用 IXF 格式的文件在导入前不需要存在相应的数据表。将数据导入到表的另外一个方法就是使用 Load 语句。Load 语句比 Import 更加快，因为它不与 DB2 数据引擎交互而直接操作数据数据库页面。然而，使用 Load 的方法导入数据不会检查数据的约束，也不会引发触发器。为了保证所导入数据的一致性，在使用了 Load 之后通常会运行 Set Integrity 命令。

下一节会详细介绍 Export, Import 和 Load。

注意：

更多关于数据迁移工具的信息，请参见以下视频：

<http://www.channeldb2.com/video/video/show?id=807741:Video:4262>

9.1 导出（EXPORT）工具

Export 用于将数据从数据表中导出到前面介绍那几种格式的文件中。其实，它执行了一个 SQL 的 SELECT 操作。下面的例子将一个 **employee** 数据表中的 10 行数据导出到 **employee.ixf** 文件中。

```
EXPORT TO employee.ixf OF IXF
  SELECT * FROM employee
    FETCH FIRST 10 ROWS ONLY
```

我们建议您尝试操作上面的例子。**Employee** 表是 **SAMPLE** 数据库（在前面的章节中建立）其中的一个表，所以您必须先连接这个数据库。

如果您选择使用 GUI 工具。**Export** 工具可以在控制中心中通过点击右键来调用，如图 9.2 所示。

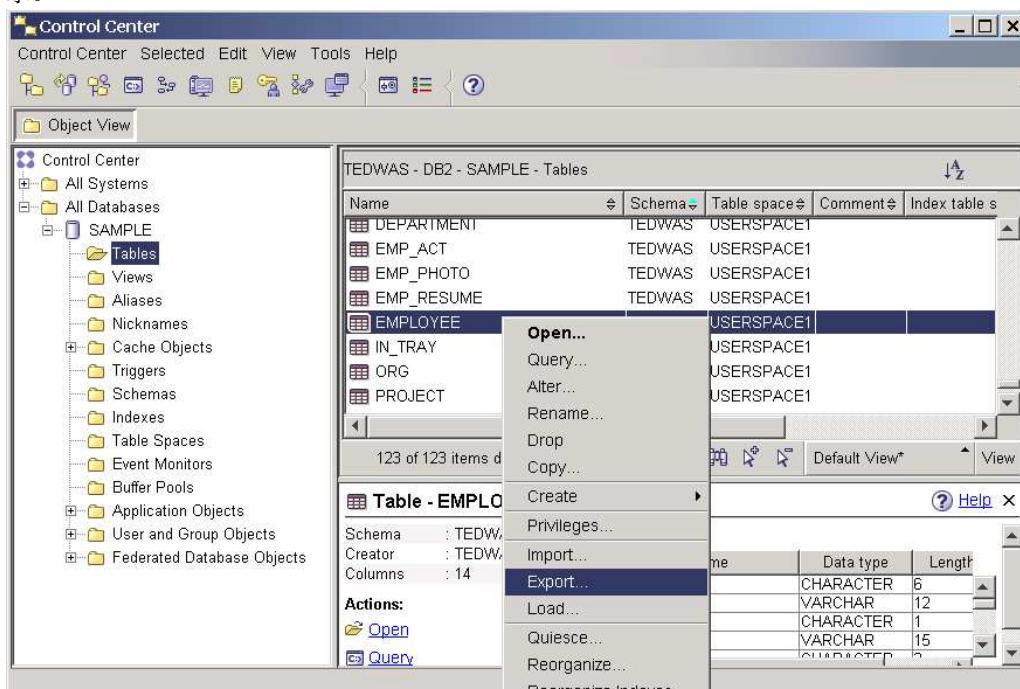


图 9.2 运行导出数据表对话框

如图所示，您首先单击选中 **employee** 表，然后点击右键，弹出菜单，在里面您可以看到 **Export** 选项。点击 **Export** 选项之后会出现一个向导，根据向导提示操作即可一步步完成导出操作。

9.2 导入（IMPORT）工具

Import 用于将前面介绍的数据文件导入到数据表中。它其实执行了 SQL 的 **INSERT** 操作。和 **INSERT** 操作一样，**Import** 执行的时候会激活触发器，所有的约束会强制实现，而且会使用数据库的缓冲池。下面的例子将 **IXF** 格式的 **employee.ixf** 中所有的数据导入到 **employee_copy** 数据表中。我们建议您尝试操作上面的例子，但是您必须先像前面一节那样执行 **Export** 工具将数据导出。

```
IMPORT FROM employee.ixf OF IXF
  REPLACE_CREATE
  INTO employee_copy
```

REPLACE_CREATE 操作是 Import 工具提供的众多参数之一。这个参数表示如果 **employee_copy** 数据表已经存在，则先清空数据表中的数据然后将 IXF 中的数据导入，如果 **employee_copy** 数据表不存在，则会先建立该表，然后将数据导入。

如果您选择使用控制中心，您可以任意选定一个表然后单击右键，选择 Import 选项来调用 Import 工具，如图 9.3 所示。

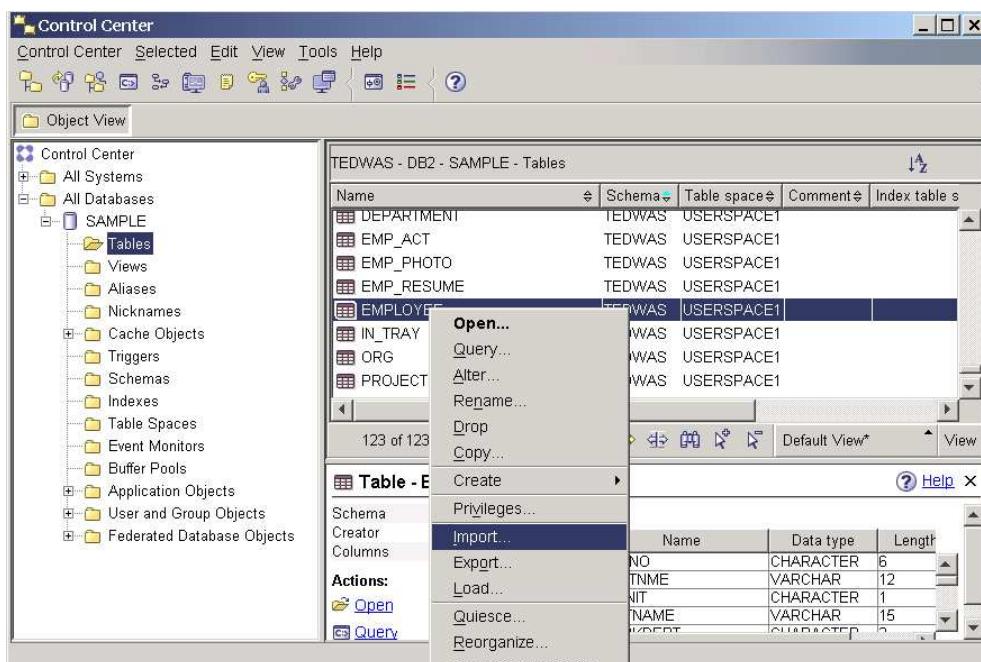


图 9.3 运行导入对话框

9.3 使用 LOAD 来导入

Load 工具可以更快的将数据文件导入到数据表中。正如前面讨论过的那样，Load 工具不会与 DB2 数据引擎发生交互，所以当使用 Load 工具时，不会触发触发器也不会使用缓冲池，而且必须单独实现数据表的约束。Import 工具执行起来比 Load 慢是因为它是低层次的数据操作工具，它分 LOAD, BUILD, DELETE 三个阶段对硬盘上的数据页面来进行直接的处理。

下面的例子将 IXF 格式的 **employee.ixf** 文件里面的所有数据导入到表 **employee_copy**。
REPLACE 是 Load 工具所提供的众多选项之一。它表示将替换 **employee_copy** 表中的所有数据。

```
LOAD FROM employee.ixf OF IXF
    REPLACE INTO employee_copy
```

执行完上面的命令后，该表进入 CHECK PENDING 状态。这时您必须运行 SET INTEGRITY 命令来检查数据的一致性，下面是执行 SET INTEGRITY 的例子：

```
SET INTEGRITY FOR employee_copy
    ALL IMMEDIATE UNCHECKED
```

如果您选择使用控制中心，您可以任意选定一个表，然后单击右键，依次选择 Load 和 Set Integrity 选项来调用它们，分别如图 9.4 和 9.5 所示。

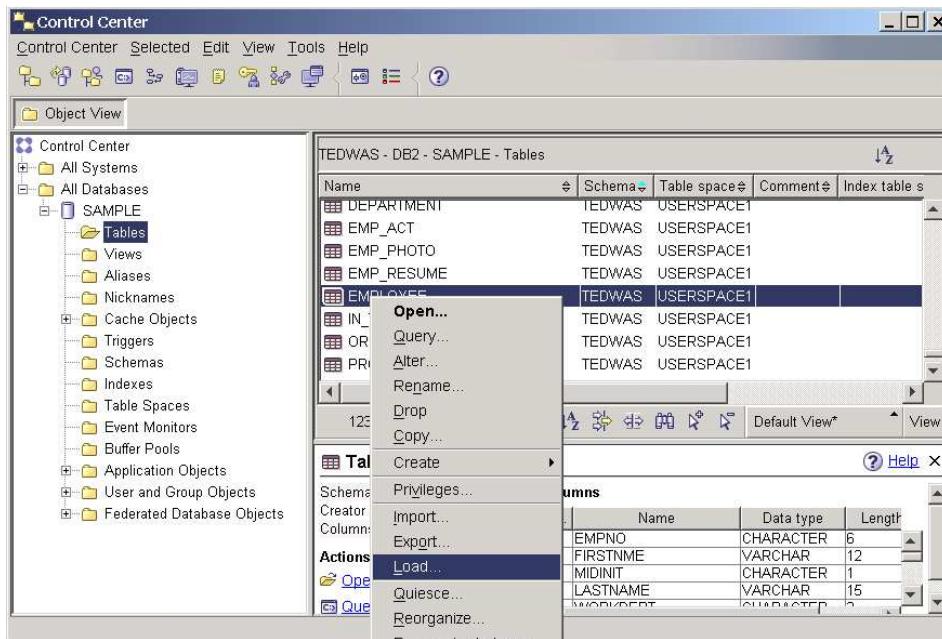


图 9.4 运行 LOAD 向导

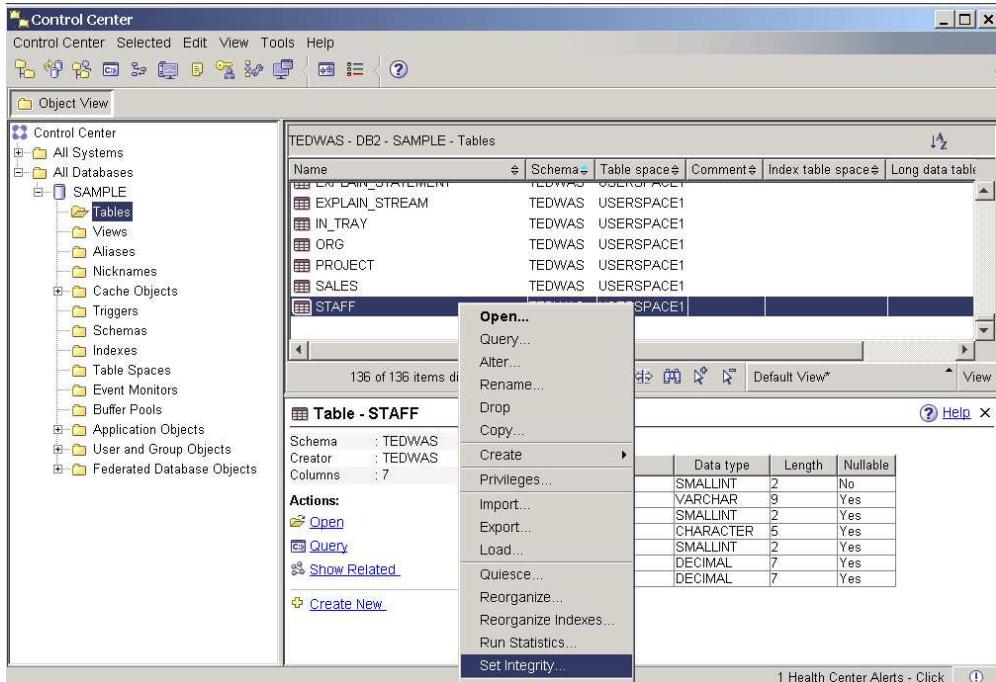


图 9.5 运行 Set Integrity 向导

9.4 db2move 工具

Export, Import 和 Load 每次都只对一个表进行操作。但是您若使用它们来写一段脚本，就可对一个数据库中的所有的表进行操作。另一个工具 **db2move** 可以更方便地完成同样的工作。**db2move** 工具只兼容 **IXF** 格式的文件，而且 **IXF** 文件的名字由 **db2move** 自动生成。下面的例子展示怎么样使用 **db2move** 并结合 **export** 和 **import** 选项来对 **SAMPLE** 数据库进行整体操作。

```
db2move sample export
```

```
db2move sample import
```

db2mov 不可以通过控制中心来调用。

9.5 db2look 工具

由前面可以知道 Export、Import、Load 和 db2move 工具可以在同一个数据库或者跨数据库的数据表之间进行数据的迁移。而 db2look 工具则可以将 DDL 语句、数据库统计状态、表空间参数导出到一个脚本文件中，这个文件可以用于不同系统的数据库。

例如，您想将 Linux 系统上 DB2 服务器的一个数据库克隆到 Windows 平台的 DB2 服务器，第一步，您在 Linux 系统中的 DB2 服务器上运行 db2look 工具，得到数据库的结构并将这个结构存储到脚本文件中，然后将这个脚本文件拷贝到 Windows 平台上的 DB2 服务器上运行，运行后得到一个克隆的数据库。这时，Windows 平台上的这个克隆数据库和 Linux 平台上的源数据库有着相同的结构。第二步，在 DB2 Linux 服务器上运行带 Export 选项的 db2move 工具，并将所产生的所有文件拷贝到 DB2 Windows 服务器上，再次运行 db2move 工具，利用 Import 或者 Load 选项将所有的文件导入到克隆的数据库中，执行完这个步骤后，这两个不同平台上的数据库就是一模一样的了。

当您的工作需要处理不同操作系统上的数据库的时候，您可能会进行上面的步骤。如果数据库服务器运行在同样的平台上，您很可能倾向于使用备份和恢复命令，这两个命令使这个整个数据迁移过程更加简单和直接。备份和恢复系统将会在后面的章节中进行详细的讨论。

下面的例子使用 db2look 工具，将 SAMPLE 数据库的表空间、缓冲池、DDL 语句导出并存储到 sample.ddl 中。我们建议您运行下面的命令并且查看它的输出文件 sample.ddl。

```
db2look -d sample -l -e -o sample.ddl
```

database name
layout
(tablespaces
& bufferpools)
extract DDL
output file

db2look 命令有非常多的选项，本书不能一一列举，您可以通过-h 选项来获得关于所有可用选项的简要介绍。

```
db2look -h
```

db2look 工具也可以在控制中心启动，如图 9.6 所示。

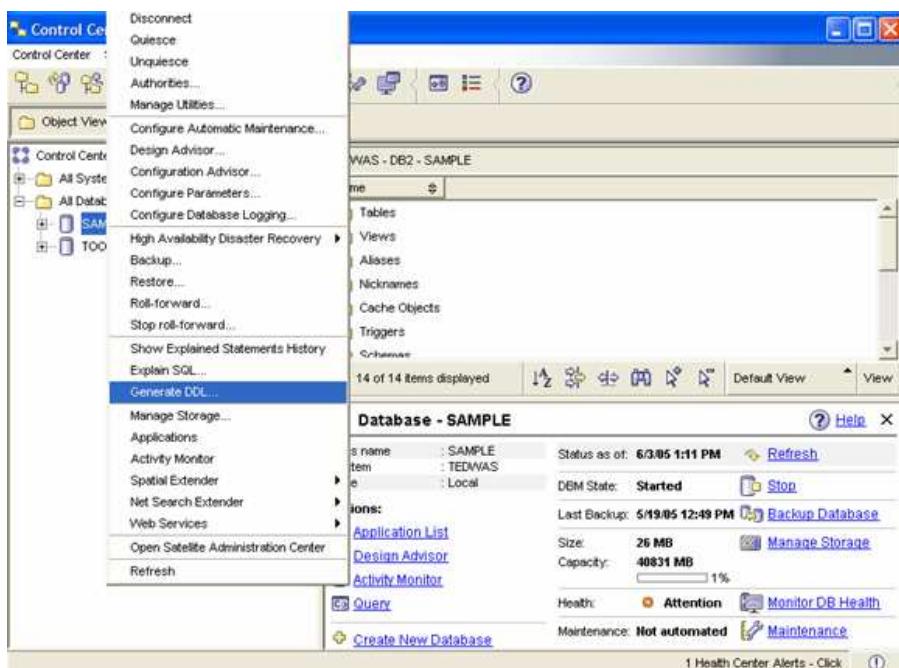


图 9.6 在控制中心中导出 DDL

在图 9.6 中，选择您想要生成 DDL 的数据库，用右键点击它，然后选择“Generate DDL”选项，这时会弹出“Generate DDL”窗口，里面有若干选项，如图 9.7 所示。

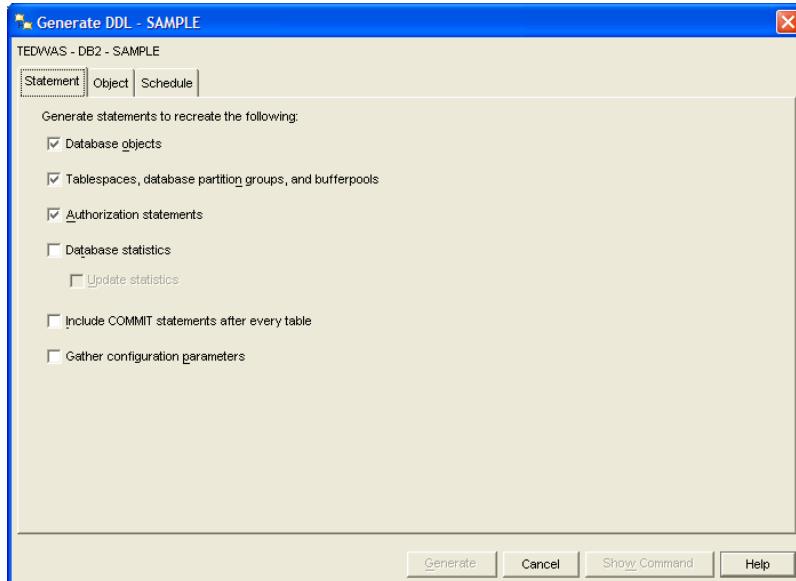


图 9.7 生成 DDL 窗口

9.6 小结

在这一章中，我们讨论了DB2的各种导入及导出功能。首先我们了解了各种导出的文件格式(ASC、DEL、WSF和IXF)，随后我们又对导出工具做了深入的检验。然后我们讨论了导入工具中的IMPORT工具和LOAD工具，以及使用LOAD工具时对SET INTEGRITY选项的需要。

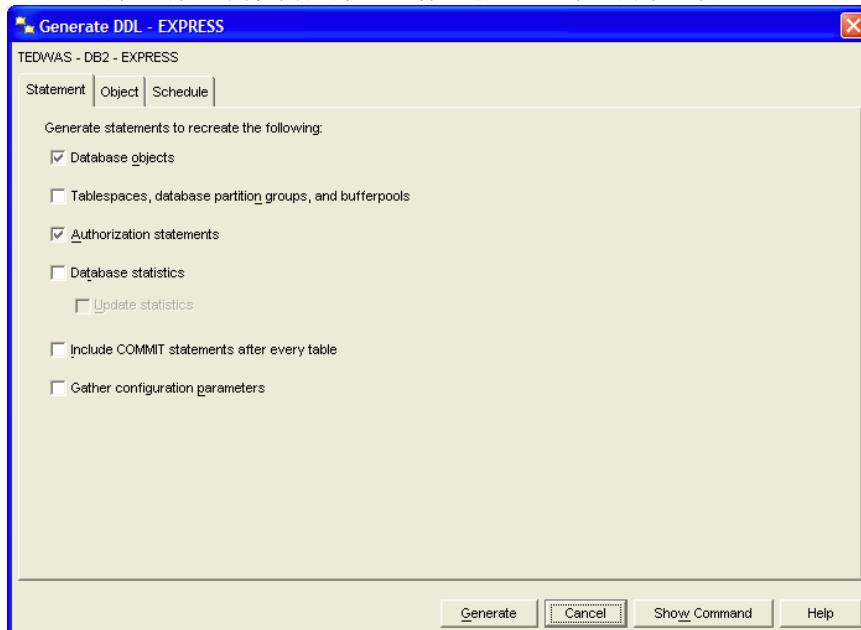
Db2move工具则为您提供了一种方式来简化导入导出的转换过程。一个更复杂一些的工具，db2look，则允许您在必要时提取并存储所需的所有数据库元素来独立新建一个完整的数据库。

9.7 实验

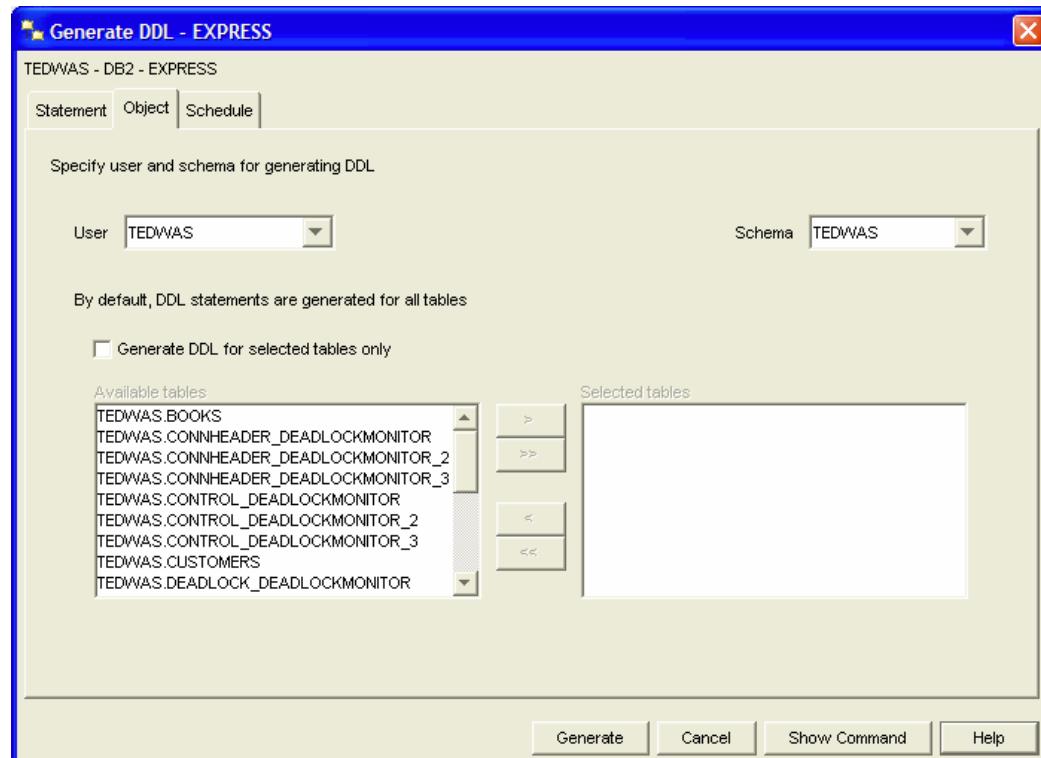
当您克隆一个数据库时，您希望将这个重建数据库的工作能够尽量简单明了和具有重复性。通常，使用SQL脚本能够达到这个目的，只要安装好DB2，SQL脚本就能够执行来进行一系列的数据操作。在本实验中，将会使用控制中心来进行这些工作，这里要求您使用控制中心来导出EXPRESS数据库（在前面的一个实验中）的DDL。

实验过程：

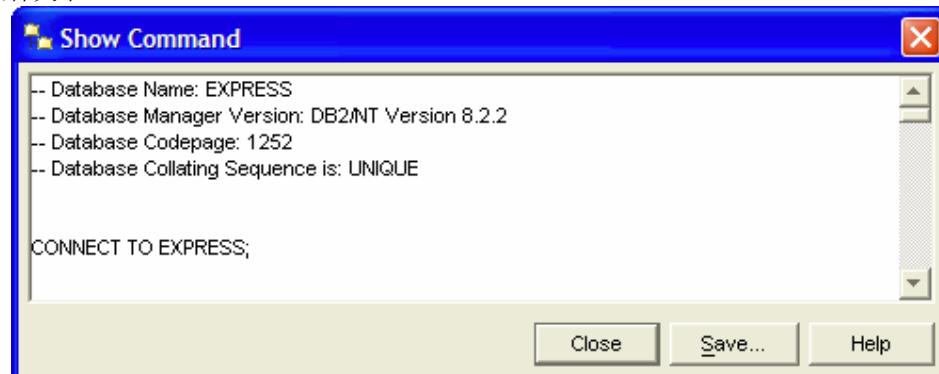
1. 打开控制中心。
2. 在左边窗口的所有数据库中右键单击EXPRESS数据库，在弹出的菜单中选择“GenerateDDL”项，这会启动“Generate DDL”窗口。
3. “Generate DDL”窗口如下所示，里面是导出DDL的选项。如果您在数据库中创建了其它的对象，比如表空间、缓冲池等等，您可以在这里选择您需要导出的对象。如果您没有创建这些对象，那么请不要选中相应的项。这里没有选中“导出数据库的统计信息”项，因为生产环境中的数据库可能和开发环境中的数据库有不同的统计信息。同样，不同环境下的设置参数也很可能不同，所以也没有选中“Gather configuration parameters”项，如果在您的环境中，所有的设置都和即将部署的设置一样，您可以选中这两个选项。



4. 单击“Object”标签，您可以指定要产生DDL的对象。在本实验中，选择您创建数据库所使用的用户和模式，然后选择生成这个数据库模式中所有对象的DDL，最后单击“生成”按钮即可开始生成DDL。



5. 查看生成的DDL。上一步的生成结果是一个单独的脚本，里面记录了所选择生成对象的全部SQL语句。现在您可以将这个脚本进行逻辑归类。
6. 新建一个文件夹 C:\express，单击“保存”按钮，将文件命名为schema.ddl 并保存在这个新建的文件夹中。



7. 在命令编辑器中打开这个新保存的文件。（提示：在命令编辑器中选择“文件”，然后点选“打开”，在对话框中选择要打开的文件）。
8. 虽然我们需要的是数据表的DDL，但是您可以看到这个文件中包括了所有数据库对象的DDL。这时，将所有的CREATE TRIGGER语句移动到一个新文件中，并将这个新文件命名为 triggers.ddl。这里尽管我们只生成了一个触发器，但是将不同的对象分开是一个最好的工程实践。
9. 现在，我们建议删除下面的语句：
 - 数据库连接语句CONNECT TO
 - 断开连接语句DISCONNECT
 这时，您有两个脚本了，
C:\express\schema.ddl 包含了所有数据表、视图、索引、约束的DDL
C:\express\triggers.ddl 包含了触发器的DDL

10. 整理脚本：

- 删除不需要的注释，例如 `-CONNECT TO...`
- 将函数和过程分离到独立的文件中。尤其在有很多函数和过程的时候特别有用。您可以根据他们的功能或者应用将它们分类，例如分成 `billing.ddl`，`math.ddl`，`stringfunc.ddl` 等等。

11. 您可能会注意到，触发器、函数、过程的结尾都使用了一个特殊的字符@。这是为了区分 `CREATE <object>` 语句和对象内部的过程语句。

10

第 10 章 – 数据库安全

本章讨论 DB2 的安全性，从图 10.1 可以大概的看出 DB2 对于安全性的处理。

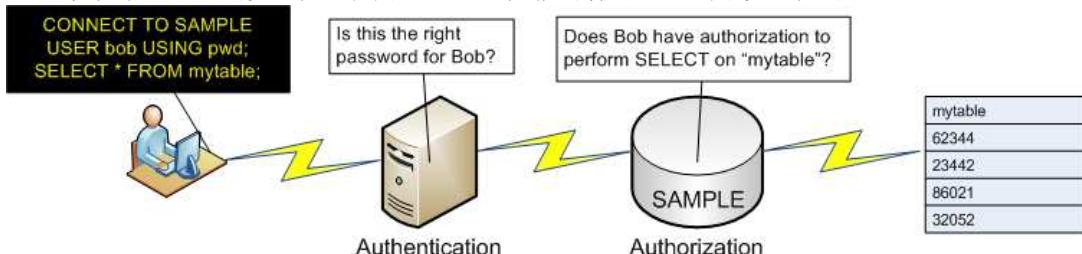


图 10.1 DB2 安全性概览

正如图 10.1 所示，DB2 的安全性由两方面组成：

认证

系统验证用户标识的过程。用户认证是通过 DB2 数据库系统外部的安全性工具来完成的，此安全性工具通常是操作系统的一部分或是一个单独的产品，比如一些网络认证方式或者第三方的认证产品。默认情况下使用的是基于操作系统的用户认证。当使用基于操作系统的用户认证时，先将用户的 id 和密码发送到数据库的服务器（例如，作为连接数据库语句的一部分），然后数据库服务器调用操作系统的用户认证来验证用户 id 和密码的正确性。

授权

在这一阶段，DB2 获取有关已认证的用户的信息，验证该用户是否有权限执行其请求的操作。授权的信息存储在一个 DB2 数据库目录和一个 DBM 设置文件中。

例如，在图 10.1，用户“bob”使用下面的语句连接到 SAMPLE 数据库

```
CONNECT TO sample USER bob USING pwd
```

用户名“bob”和密码 pwd 会发送到操作系统或者外部的认证设施来执行用户认证，用户认证首先会检验用户名“bob”是否已经定义，然后检测所发送过来的密码是否和这个用户名相对应。如果用户认证通过，操作系统会将安全控制权交给 DB2，接着，当用户“bob”执行一条语句，如：

```
SELECT * FROM mytable
```

此时 DB2 会接管安全控制，对用户进行授权验证，确定用户“bob”是否有权限在数据表 mytable 上执行 SELECT 操作。如果授权验证成功，那么 DB2 就会在 mytable 上执行该操作，否则，DB2 会返回一个出错信息。

注意：

更多关于 DB2 安全性的信息，请参见以下视频：

<http://www.channeldb2.com/video/video/show?id=807741:Video:4267>

10.1 认证

尽管实际的认证过程是由操作系统通过默认的安全性插件（或者外部安全工具）来执行，但是DB2可以决定在哪个层次上进行用户认证。

DB2服务器中，DBM CFG的AUTHENTICATION参数能设定为一系列的值，比如当这个值设为 SERVER（默认）时，认证过程由在服务器端的操作系统/外部安全工具来执行，如果这个值设置为 CLIENT 时，认证过程在客户端执行，如图 10.2 所示。

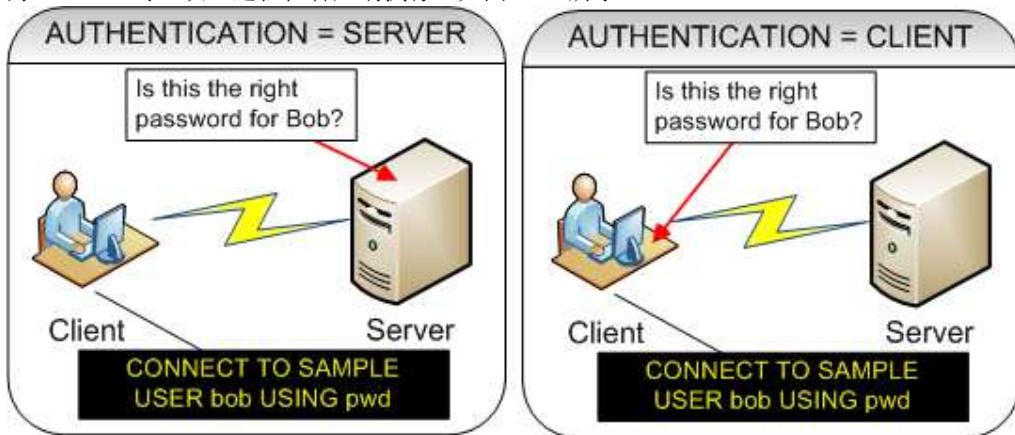


图 10.2 DB2 对执行用户认证的控制

AUTHENTICATION参数允许值列表在表10.1中。

命令	描述
SERVER（默认）	认证在服务器端执行
CLIENT	认证在客户端执行
SERVER_ENCRYPT	和 SERVER 参数相似，而且用户的 id 和密码都经过加密
KERBEROS	认证使用 Kerberos 安全机制
SQL_AUTHENTICATION_DATAENC	在服务器端进行认证，数据库连接时必须使用数据加密
SQL_AUTHENTICATION_DATAENC_CMP	与上面类似，但当条件不允许的情况下，可以不对数据进行加密
GSSPLUGIN	使用外部的基于 GSS API 插件的安全工具进行认证

表 10.1 AUTHENTICATION 参数的允许值

10.2 授权

授权由特权、权限、任务和基于标号的访问控制证书（LBAC）组成，这些信息都存储在DB2的系统表中，并由DB2来管理。

特权允许用户对数据库执行单一的操作，比如 CREATE, UPDATE, DELETE, INSERT 等等。

任务允许您将不同的特权集为一组后授予一个用户，一个用户组或其他的任务。

权限是预定义的规则，由若干的特权组成。

基于标号的访问控制（LBAC）证书包括支持指定用户对特定行或列进行访问的策略和标号。

LBAC不包含在DB2 EXPRESS-C中，但是您可以在第二章了解更多相关内容。

10.2.1 特权

图10.3 展示了DB2 的不同特权和权限。

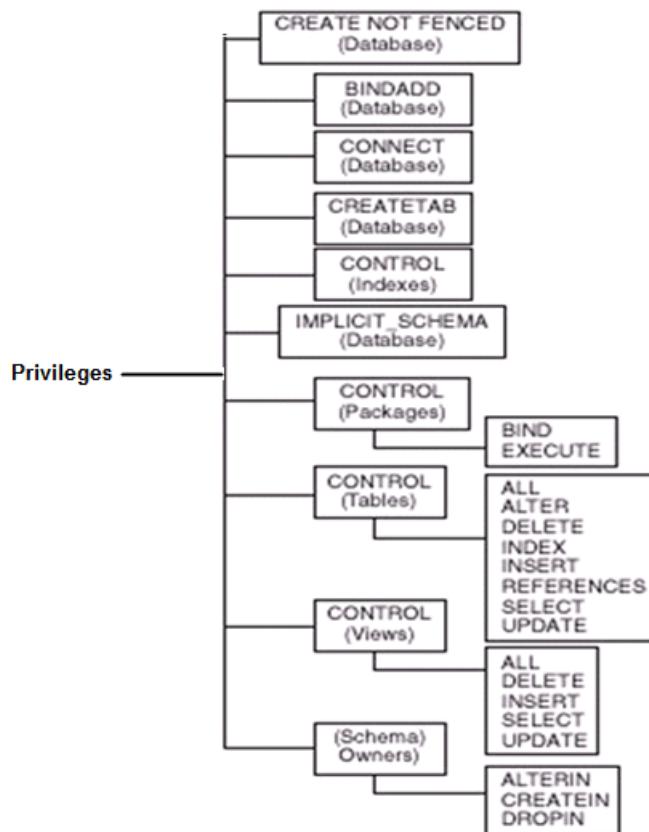


图 10.3 DB2 一些特权

一个用户或用户组得到CONTROL特权，就表示他们同样可以对其他用户或用户组授权。其他不同的特权的信息参见DB2信息中心。

10.2.2 权限

权限分为两种：

- 实例层次的权限：这些权限只能在实例层次上操作。比如 **SYSADM** 权限。
- 数据库层次的权限：这些权限只能在数据库层次上操作。比如 **DBADM** 权限。

10.2.2.1 实例层次的权限

表10.2列出了实例层次的权限。

权限	描述
SYSADM	整体管理实例
SYSCTRL	管理一个数据库管理实例
SYSMAINT	在一个实例中维护数据库

SYSMON	监督实例及其数据库
--------	-----------

表10.2-实例层次的权限

为了将 SYSADM, SYSCTRL, SYSMAINT 的权限授予一个组，可以在DBM CFG 中，将 SYSADM_GROUP, SYSCTRL_GROUP, SYSMAINT_GROUP 赋值为操作系统的用户组。

例如，将SYSADM 权限赋予db2admns 这个组，您可以输入如下的命令：

```
update dbm cfg using SYSADM_GROUP db2admns
```

每一个 DB2 实例都有自己的权限定义。在 Windows 系统中，这些参数默认为空，这意味着本地的Administrators 组具有SYSADM 的权限级别。在DB2 9.7中，DB2ADMNS组（在扩展安全激活时）和LocalSystem Account也将具有SYSADM的权限。LocalSystem Account 的授权ID是 SYSTEM。在Linux 系统中，SYSADM 组默认是实例owner 组。

从DB2信息中心截得的截图10.4展示了不同的实例层次的权限以及他们所能发挥的不同职能。正如图片说明的，一个SYSADM权限包含且多于SYSCTRL的所有功能，一个SYSCTRL权限包含且多于SYSMAINT的所有功能，等等。

new in
V9.7

SYSADM

- Update and restore a database manager configuration parameters (DBM CFG) including specifying groups that have SYSADM, SYSCTRL, SYSMAINT AND SYMON
- Grant and revoke table space privileges
- Upgrade and restore a database

SYSCTRL

- Update a database, node, or distributed connection services (DCS) directory
- Restore to a new or existing database
- Force users off the system
- Create or drop a database (NOTE: automatically gets DBADM authority)
- Create, drop, or alter a table space
- Restore to a new or existing database
- Use any table space

SYSMAINT

- Back up a database or table space
- Restore to an existing database
- Roll forward recovery
- Start or stop an instance
- Restore or quiesce a table space, and query its state
- Run tracing
- Database system monitor snapshots
- Reorganize tables
- Use RUNSTATS and update log history files

SYMON

- GET DATABASE MANAGER MONITOR SWITCHES
- GET MONITOR SWITCHES
- GET SNAPSHOT
- LIST commands: ACTIVE DATABASES, APPLICATIONS, DATABASE PARTITION GROUPS, DCS APPLICATIONS, PACKAGES, TABLES, TABLESPACE CONTAINERS, TABLESPACES, UTILITIES--
- RESET MONITOR
- UPDATE MONITOR SWITCHES
- APIs: db2GetSnapshot and db2GetSnapshotSize, db2MonitorSwitches, db2mtrk, db2ResetMonitor
- All snapshot table functions, without running SNAP_WRITE_FILE
- Can connect to a database

图10.4-实例层次的权限和他们的职能

10.2.2.2 数据库层次的权限

表10.3列出了数据库层次的权限。

权限	描述
SECADM	在一个数据库内管理安全
DBADM	管理数据库
ACCESSCTRL new in V9.7	授予、撤销权限和特权（除了SECADM, DBADM, ACCESSCTRL, 和 DATAACCESS权限）。注意授予和撤销这些权限时必须有SECADM权限。
DATAACCESS new in V9.7	提供访问数据库中数据的能力
SQLADM	监督并协调SQL查询
WLMADM	管理工作量
EXPLAIN	需要解释查询计划的用户（EXPLAIN权限自身无法访问该数据）

表10.3-数据库层次的权限

要授予一个数据库层次的权限，可使用GRANT语句。例如，在数据库SAMPLE中，将DBADM权限授予用户bob，可用：

```
connect to sample
grant DBADM on database to user bob
```

在上面的例子中，您首先要连接到数据库，本例中为sample数据库，然后，您可以将DBADM权限授予用户。DBADM权限或其他数据库层次的权限的授予者必须有SYSADM权限。

请注意：DBADM权限不能够建立表空间，尽管表空间是数据库里面的对象。因为表空间与其提供者（磁盘）和缓冲池（内存）相关联，而这些是系统的物理资源。这些可由SYSADM权限实现。

由DB2信息中心截得的截图10.5展示了不同的数据库层次的权限和他们能发挥的职能。

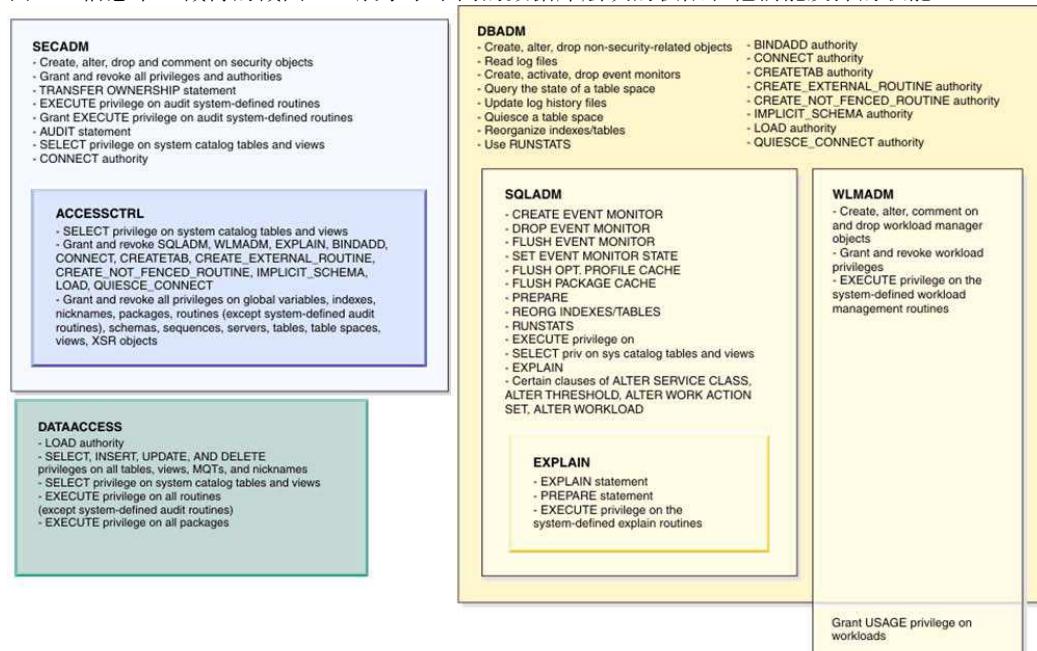


图10.5-数据库层次的权限和他们的职能

new in
V9.7

注意:

在 DB2 9.7 中, 为了增强数据的保密性和可管理性, 授权模式被升级以便更清晰地区别系统管理员、数据库管理员和安全管理员的责任。

大体来讲, 一些权限的职能范围与以前版本的 DB2 相比减小了。例如一个 SYSADM 权限不再有权访问任何数据库中的数据。一个 DBADM 权限不再有权访问它管理的数据库中的数据。相反, SECADM 则得到了更多的职能, 例如授予和撤销用户、任务和用户组的权限和特权的能力。

考虑到更多的粒度和更好的控制系统安全, DB2 9.7 中也创建了新的权限。这种做法通过不授予用户他们需求以外的权限方式将数据泄露的风险降到了最低。

DB2 9.7 的更新也包括了审查方面的完善来保证过去的数据库活动的再次进行。例如, 如果您需要知道一个几年前给出的命令对某些表格有何影响时, 您可以使用数据库审查信息来得到您分析所需的信息。

10.2.2.3 使SYSADM和DBADM以DB2 9.7之前版本的方式工作

如果您想让一个 SYSADM 权限如 DB2 9.7 以前版本一样, 可考虑如下两种情况

- 如果 SYSADM 权限是数据库的创建者, 那么它将自动为该数据库得到 DATAACCESS, ACCESSCTRL, SECADM 和 DBADM 权限
- 如果 SYSADM 权限不是数据库的创建者, 那么就得到和 DB2 以前版本 (除了 SECADM 权限) 同样的能力; 一个 SECADM 权限必须将 DBADM、DATAACCESS 和 ACCESSCTRL (默认) 权限授予给出的数据库中的 SYSADM 权限。

对于 SECADM 权限须考虑以下情况

- 默认 SECADM 是数据库的创建者
- 如果一个有 SECADM 权限的用户将 SECADM 权限授予一个有 SYSADM 权限的用户, 那么 SYSADM 用户就能将 SECADM 权限授予其他用户。
- 如果一个有 SECADM 权限的用户将 DBADM 权限授予另一个用户, 那么 DBADM 用户默认自动得到 DATAACCESS 和 ACCESSCTRL 权限。

如果您迁移了一个 DB2 9.5 数据库, 那么 SYSADM 和 DBADM 能力将不会改变, 因为在迁移中 DB2 自动授予 SYSADM 用户组 DBADM, DATAACCESS 和 ACCESSCTRL 权限。DB2 在迁移中也会自动授予每个有 DBADM 权限的授权 ID DATAACCESS 和 ACCESSCTRL 的权限。另外, 当数据库中没有 USER 类授权 ID 拥有 SECADM 权限时, DB2 自动授予正在迁移中的用户 ID SECADM 权限。

SYSADM 权限会失去它隐含的授予或撤销 DBADM 和 SECADM 权限的能力, 此时这项只能由 SECADM 权限来发挥。

10.2.3 任务

任务允许一个安全管理员来为多个用户或用户组分配特权或权限。任务和用户组十分相似, 但任务是在 DB2 内部定义, 因此提供了很多便利。例如, 在您创建视图、触发器等项目时, 授予任务的特权或权限总是被用到, 而这时是与用户组无关的。相反, 您不能分配类似 SYSADM 的实例层次的权限给一个任务, 只能分配特权和数据库层次的权限。然后对于一个用户组来讲, 则可以分配所有的特权和权限。

关于任务的工作需要按照以下步骤:

1. 一个安全管理员 (SECADM) 必须首先用如下命令来创建一个任务

```
CREATE ROLE TESTER
```

2. 接下来，一个DBADM权限授予任务特权或权限。例如，在SAMPLE数据库中的STAFF和DEPT表为TESTER任务授予SELECT特权，则有：

```
GRANT SELECT ON TABLE STAFF TO ROLE TESTER
GRANT SELECT ON TABLE DEPT TO ROLE TESTER
```

3. 然后，安全管理员将TESTER任务授予用户RAUL和JIN：

```
GRANT ROLE TESTER TO USER RAUL, USER JIN
```

4. 下一步，如果用户JIN将要离开TEST部分，安全管理员将会从用户JIN处撤回TESTER任务。

```
REVOKE ROLE TESTER FROM USER JIN
```

10.3 关于组特权

如果您要用用户组代替任务，则要考虑以下情况：

- 当一个用户组获得特权后，它的所有成员也会隐式地获得这些特权。
- 当一个用户从用户组中移除后，这个用户会失去从用户组隐式获得的那些特权，但是对于原先显式赋予的特权依然保留。显式地将特权赋予一个用户，则必须显式地将这个特权撤销。

10.4 PUBLIC 组

DB2 定义了一个内部的组，叫 PUBLIC。操作系统或者网络认证服务定义的用户都隐式的属于 PUBLIC 组。当一个数据库建立时，下面的特权都会自动的授予 PUBLIC 组：

- CONNECT,
- CREATETAB,
- IMPLICIT SCHEMA,
- BINDADD

为了提高安全性，我们建议撤回 PUBLIC 组以下的权限：

```
REVOKE CONNECT          ON DATABASE FROM PUBLIC
REVOKE CREATETAB        ON DATABASE FROM PUBLIC
REVOKE IMPLICIT_SCHEMA  ON DATABASE FROM PUBLIC
REVOKE BINDADD          ON DATABASE FROM PUBLIC
```

10.5 GRANT 和 REVOKE 语句

GRANT 和 REVOKE 语句是标准 SQL 的语句，它们用于授予/撤销用户或者用户组的特权。下面是一些例子：

赋予 USER1 T1 表的 SELECT 特权：

```
GRANT SELECT ON TABLE T1 TO USER user1
```

赋予 GROUP1 对于 T1 表的所有权限：

```
GRANT ALL ON TABLE T1 TO GROUP group1
```

撤销 GROUP1 对于 T1 表的所有权限：

```
REVOKE ALL ON TABLE T1 FROM GROUP group1
```

赋予 USER1 对于存储过程 p1 的 EXECUTE 特权：

```
GRANT EXECUTE ON PROCEDURE p1 TO USER user1
```

撤销 USER1 对于存储过程 p1 的 EXECUTE 特权:

```
REVOKE EXECUTE ON PROCEDURE p1 FROM USER user1
```

10.6 查看授权和特权

查看授权和权限的最简单方法就是通过控制中心来查看。图 10.4 展示了怎样在控制中心里启动数据表 EMPLOYEE 的表特权 (Table Privileges) 对话框。

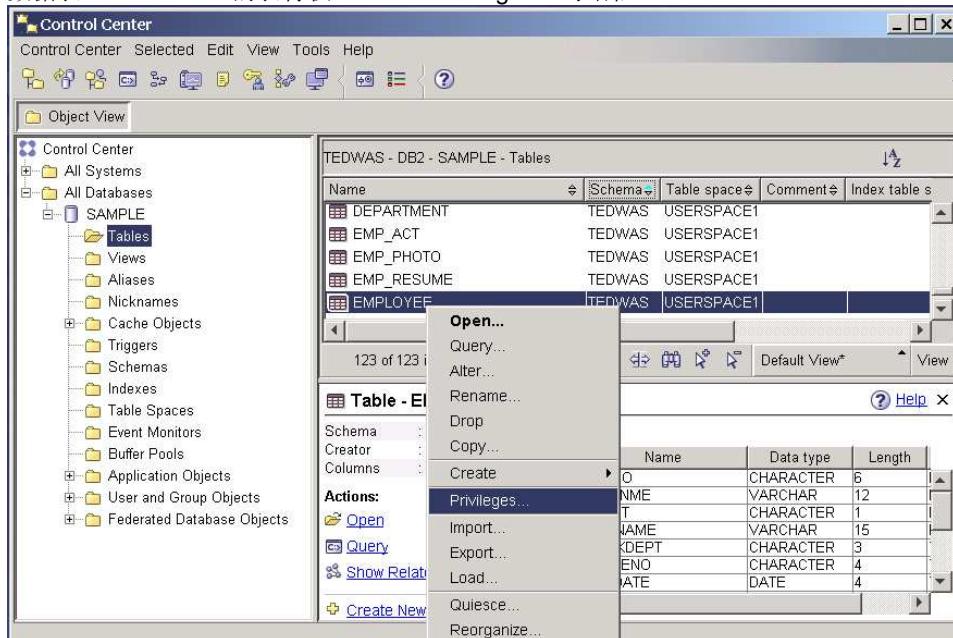


图 10.6 运行表特权对话框

如图 10.4 所示，您首先选择想要的数据表，右键点击它，然后选择“Privileges”，选择后将会出现表权限 Table Privileges 对话框，如图 10.5 所示，图中的文字对不同区域和窗体部件进行了解释。

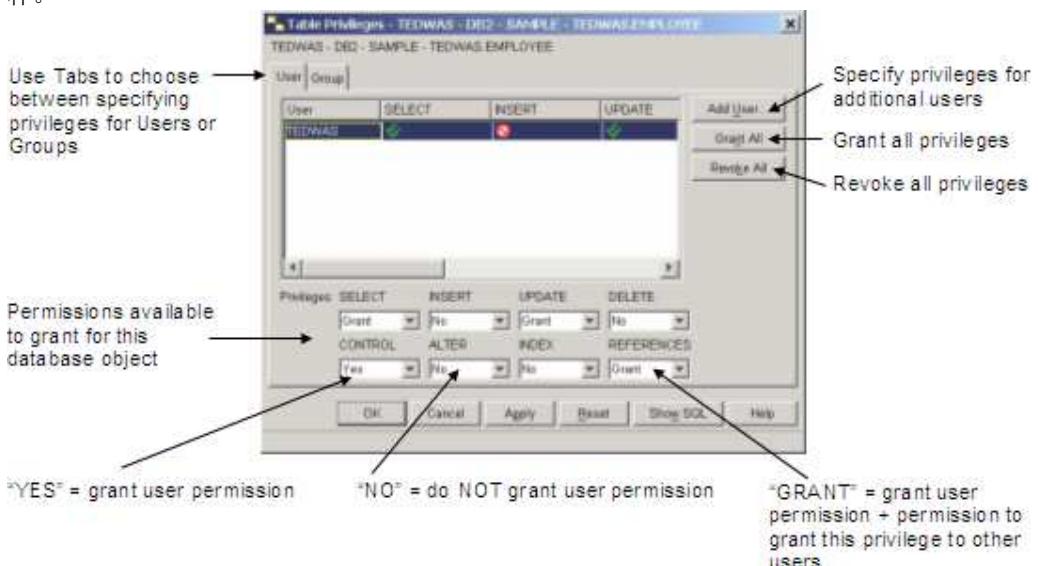


图 10.7—表特权对话框

另一方面，您可以查询 **DB2 SYSCAT** 目录视图，它保存了所有授权的信息。例如您想知道用户 **DB2ADMIN** 是否有表 **T2** 的 **SELECT** 权限，如果有，是哪位用户将这个特权授予 **DB2ADMIN**，您可以像下面例子这样查询：

```
SELECT grantor, grantee, selectauth
  FROM syscat.tabauth
 WHERE tablename = 'T2'

GRANTOR          GRANTEE          SELECTAUTH
-----
ARFCHONG        DB2ADMIN         Y
```

在上面的例子中，用户 **ARFCHONG** 将这个 **SELECT** 特权授予用户 **DB2ADMIN**。

10.7 在 Windows 上的扩展安全

为了防止Windows操作系统访问DB2文件和目录（例如DB2存储实例信息的位置），DB2安装时有默认的扩展安全。扩展安全会创建两个用户组：

- **DB2ADMNS**: 这一用户组及本地管理员能够通过操作系统完整访问所有DB2项目
- **DB2USERS**: 这一用户组只能通过操作系统以只读的方式执行对DB2所有项目的访问

new in V9.7 在 DB2 9.7 中，如果扩展安全可用，并且数据库配置参数 **SYSADM_GROUP** 还没设置，则 DB2ADMNS 用户组的成员自动获得 **SYSADM** 权限。

10.8 小结

这一章主要叙述了**DB2**的安全性。首先我们做了一个理解性的讨论，关于授权和认证的区别和重要性。在这里我们看到了不同的权限等级为实例及数据库提供了不同级别的安全性。

接下来，我们讲述了任务的新定义，以及在考虑到安全性以及用户组安全的局限性时，他们如何被有效的利用。此外，我们还讨论了**PUBLIC**组，以及关于如何保护它，保证普通用户不接触到数据服务器的建议。

另外，我们还验证了**GRANT**和**REVOKE**语句，最后，我们还看了如何使用控制中心和系统目录表来核对特权和权限的级别。

10.9 实验

目前为止，您一直都是使用**DB2** 的管理员帐户 (**SYSADM**) 来运行所有的数据库命令。该管理员帐户拥有对所有的数据库工具、数据、数据库对象的控制权限，所以这个账户必须小心保护好以避免有意或者无意的数据丢失。在大多数情况下，您希望另外建立一个新的帐号或者用户组，并赋予它们有限的权限。在本实验中，您将会建立一个新的用户帐号，然后赋予它特定的一些特权。

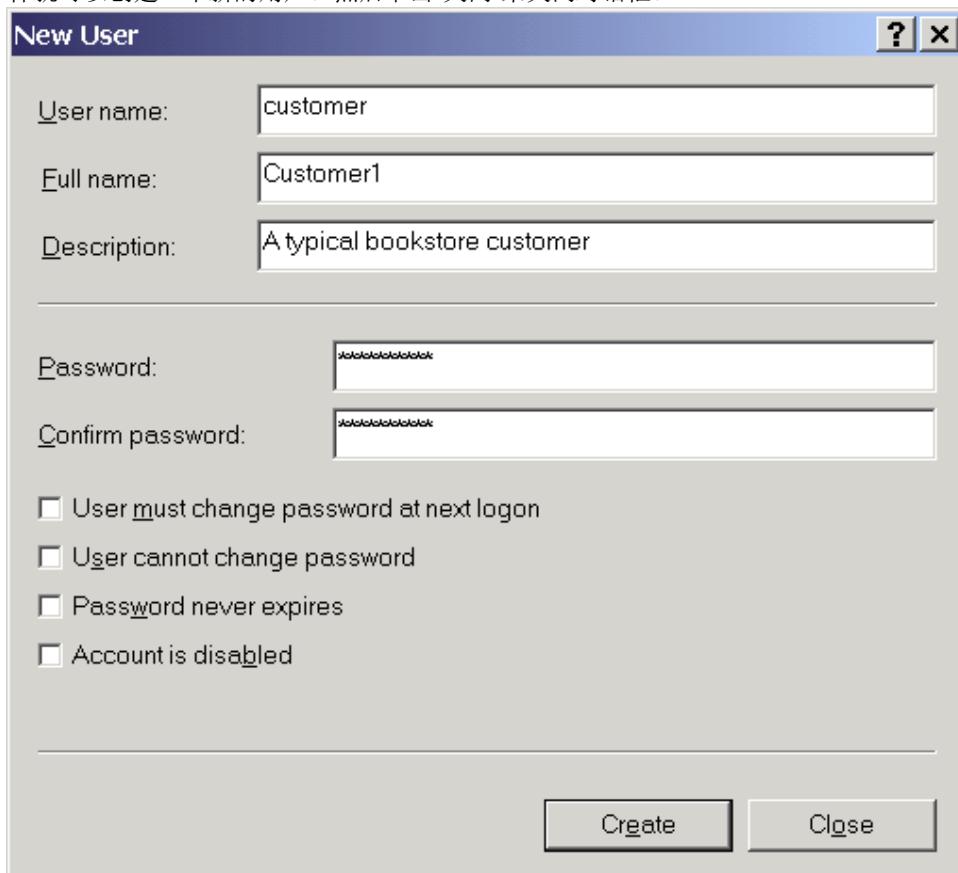
第一部分—使用特权

在这一部分中您将练习如何通过控制中心向用户授予和撤销特权。

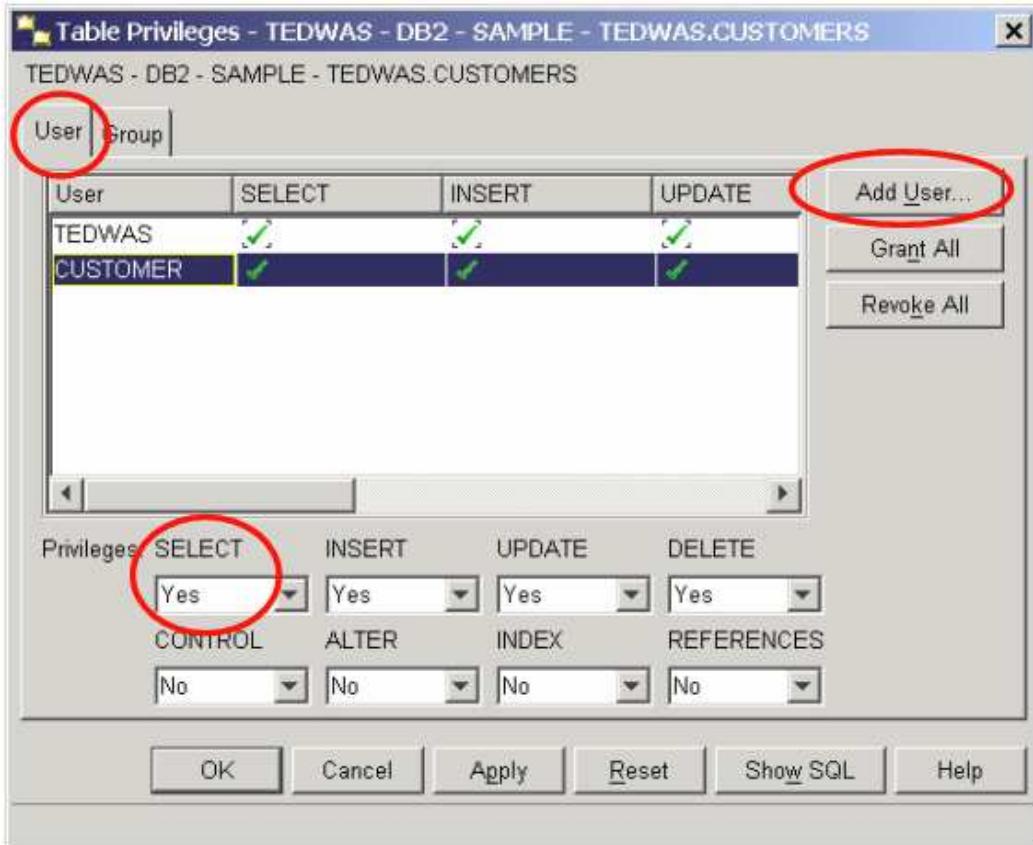
实验过程：

5. 在桌面上右键单击“我的电脑”图标，在弹出菜单中选择“管理”项，出现“计算机管理”(Windows Computer Management) 窗口。

6. 在“计算机管理”的左边窗口的控制树中展开“系统工具”，然后再其下展开“本地用户和组”文件夹，右键单击用户文件夹，然后选择“新用户”。
7. 在“新用户”对话窗中进行以下步骤：在“用户名”一栏中输入“customer”；在“全名”一栏中输入“Customer1”；在“描述”栏中输入“A typical bookstore customer”；在“密码”和“确认密码”栏中输入“ibmdb2ibm”；去掉“用户下次登陆时必须修改密码”前面的钩；最后单击“创建”按钮，这样就可以创建一个新的用户。然后单击“关闭”来关闭对话框。



8. 打开控制中心，选择高级视图。为了转换到高级视图，在控制中心工具 Tools 菜单里选择“Customize Control Center”，然后设置“Advanced”选项并单击“OK”按键。
9. 将控制中心左边的对象树展开为：All Databases > EXPRESS > Tables。
10. 授予新建用户所需的特权。在EXPRESS 数据库中右键选择CUSTOMERS 表，然后在弹出菜单中选择Pririleges 项，这时会弹出Table Privileges 对话框。
11. 单击“Add User”按钮，然后选择刚才新建立的“customer”用户。单击“OK”关闭“AddUser”对话框。
12. 您会注意到，用户customer 已经添加到用户列表中，但是并没有分配相关特权。为了将 SELECT、INSERT、UPDATE、DELETE 特权赋予用户，可以在对应的下拉列表中选择 Yes。网上的顾客必须能够浏览、添加、更新、删除与他们账户相关的数据。我们不必给他们其他的权限因为他们并不需要。单击“OK”确定您做的更改并关闭Table Privileges 对话框。



13. 在BOOKS 和SALES 表上重复第7-9步。对于BOOKS 表，只需要赋予customer 账户SELECT 权限，因为顾客不能够更改商店的存货信息。对于SALES 表，只需要赋予customer账户SELECT 和INSERT 特权而没有DELETE 和UPDATE 特权，因为只有商店的雇员才可以修改销售交易信息。
14. 用上面建立的用户ID 连接数据库，在DB2的命令窗口输入如下命令：

```
Db2 connect to express user customer using ibmdb2ibm
```

尝试从customer 表选择（SELECT）数据，发现了什么？尝试删除（DELETE）或者更新（UPDATE）SALES 表，又发现了什么？

第二部分—使用SYSADM, DBADM和SECADM权限

在这一部分您将练习如何分配SYSADM和DBADM权限，并理解这些权限是如何工作的。

实验过程：

1. 按第一部分中的步骤创建一个新用户：mysysadm
2. 创建mysysadmgp窗口用户组。仍按原来的步骤创建用户，但这次不是右键点击“用户”文件夹，而是右键点击“用户组”文件夹并选择“新用户组”。在组名位置输入mysysadmgp，在成员选择处，单击“增加”按钮来增加一个新成员，然后输入mysysadm。单击“检查用户名”按钮来确认您是否正确地输入了成员名。如果是，单击“创建”，然后单击“关闭”。
3. 至此您已经创建了一个用户mysysadm和它所属于的用户组mysysadmgp。这些都已经在

Windows 操作系统上实现了。现在我们需要通知DB2我们要让mysysadmgrp用户组做 SYSADM 组，这就要在DB2的命令窗口中用到下面的命令：

```
db2 update dbm cfg using SYSADM_GROUP mysysadmgrp
```

由于SYSADM_GROUP的参数不是动态的，您需要先停止再开始这个实例。Db2stop 的选项“force”将会保证所有的连接都被移除到db2stop中。

```
db2stop force  
db2start
```

4. 在DB2的命令窗口中，使用mysysadm用户来连接SAMPLE数据库，并在表STAFF上输入一个SELECT *语句。注意您需要使用创建表格时所用的相应的模式。在下面的例子中我们用arfchong作为模式。

```
db2 connect to sample user mysysadm using ibmdb2ibm  
db2 select * from arfchong.staff
```

您应该得到了一个这样的错误消息。为什么？难道您没有SYSADM权限？

```
SQL0551N "MYSYSADM" does not have the required authorization or  
privilege to perform operation "SELECT" on object  
"ARFCHONG.STAFF". SQLSTATE=42501
```

从DB2 9.7开始，SYSADM并不会默认得到DBADM权限，这就是您收到这条错误报告的原因。

5. 使用创建SAMPLE数据库的Windows用户来连接数据库。在例子中，arfchong是用户。接下来授予mysysadm用户DBADM权限，但不授予DATAACCESS权限，然后以mysysadm身份在表STAFF上尝试SELECT。这生效吗？为什么？

```
db2 connect to sample user arfchong using ibmdb2ibm  
db2 grant dbadm without dataaccess on database to user mysysadm  
db2 connect to sample user mysysadm using ibmdb2ibm  
db2 select * from arfchong.staff
```

如您所见，即使mysysadm被授予了DBADM权限，您仍收到了错误信息。这种现象是必然的，因为其中包含语句WITHOUT DATAACCESS意味着DATAACCESS权限是不包含在内的，因此仍然没有访问数据的权限。这是一个如何限制DBADM访问数据的例子。

6. 让我们授予DATAACCESS权限并再次尝试SELECT。

```
db2 connect to sample user arfchong using ibmdb2ibm  
db2 grant DATAACCESS on database to user mysysadm  
db2 connect to sample user mysysadm using ibmdb2ibm  
db2 select * from arfchong.staff
```

现在您的SELECT语句应该生效了！这个实验向您展示了DB2 9.7中SYSADM和DBADM权限的全新表现。您应该从这个实验中得到的主要思想是明白现在数据访问中存在区别，以及SYSADM和DBADM权限都能够做什么。

7. 重置YSADM_GROUP的值为空，这样本地管理员用户组和LocalSystem account就又具有SYSADM权限：

```
db2 update dbm cfg using sysadm_group NULL  
db2stop force  
db2start
```


11

第 11 章 – 备份和恢复

本章我们将讨论 DB2 数据库的日志记录、如何使用 **BACKUP** 功能制作一个数据库的完整或部分拷贝、以及如何使用 **RESTORE** 功能恢复您的数据。

注意:

更多关于日志记录、备份、恢复的信息，请参见以下视频：

<http://www.channeldb2.com/video/video/show?id=807741:Video:4282>

11.1 数据库的日志记录

当您使用一个文本编辑器进行编辑工作时，每次都需要点击“**save**”按钮以确保您的文档被成功保存。而在数据库的世界里，一个“**COMMIT**”语句正是完成这同样的工作。每次当一个

“**COMMIT**”语句被执行的时候，就保证了所有已经对数据做出的修改都被保存在某个特定位置。

同样的，当您使用文本编辑器时，有时您会看到窗口的右下角显示一条“**auto-saving**”的短消息。在数据库里，这同样会发生，因为您对数据做出的任何操作，如“**UPDATE**”，“**INSERT**”或者“**DELETE**”，都会在您执行该操作的时候被保存在某个特定位置。

上文提到的所谓“特定位置”是指数据库的日志。数据库日志存储在磁盘上，用来记录事务（**transactions**）的处理行为。如果发生系统崩溃或者数据库崩溃，日志被用来在恢复过程中恢复崩溃前被提交的事务。

图 11.1 形象地介绍了使用一个采用日志记录的数据库时发生的情况。

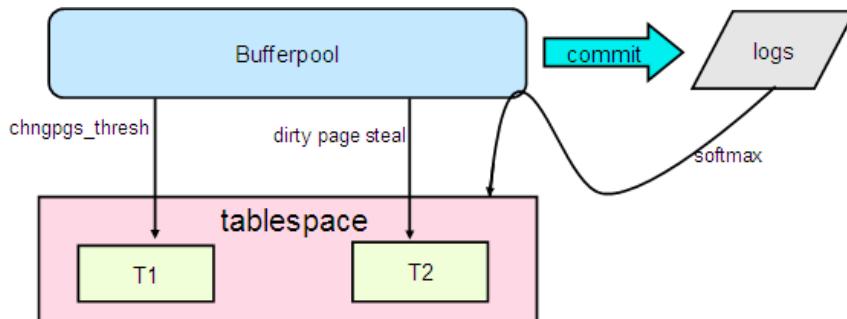


图 11.1 – 数据库日志记录

在图 11.1 中，我们看到一个表空间和数据库的日志。它们都被存储在磁盘上，我们建议把它们应存储到不同的磁盘上。举例来说，当一个 **UPDATE** 操作发生的时候，当前被操作的数据行记录将被移动到缓冲池（位于内存中），**UPDATE** 操作将会在缓冲池中执行，而同时该行的旧值和新值会被存储到日志文件中。有时候这个存储是立即进行的，有时候是当一个日志缓冲区满了以后才进行。而当在 **UPDATE** 操作后立即执行一个 **COMMIT** 操作时，该数据行的旧值和新值就会被立即存储到日志文件中。这样的过程在数据库的很多其他 SQL 操作上反复发生。只有当碰到某些情况的时候，比如被修改的数据页数目达到了 **CHNGPGS_THRESH** 数值，缓冲池中的这些修改的数据页记录

就会被“外部化具体化”或者被写到表空间磁盘上。参数 CHNGPGS_THRES 表示了缓冲池中所谓“脏”记录的百分比，也就是被修改了的记录的百分比。

从性能的角度来看，为每一个 COMMIT 操作执行两次磁盘写入是毫无意义的：一次写到日志中，另一次写到表空间所在磁盘中；这就是为什么数据“外部化”到表空间所在磁盘的操作仅仅发生在到达阈值参数的时候，即修改的数据页达到 CHNGPGS_THRES 的时候。

11.2 日志的类型

有两种日志类型：

主日志（Primary logs）

主日志是预分配的，可用的数目由 db cfg 的 LOGPRIMARY 参数规定。

辅助日志（Secondary logs）

辅助日志是根据 DB2 的需要动态分配的。辅助日志的最大数目由 db cfg 的 LOGSECOND 参数规定。动态分配日志是很消耗系统资源的；所以，为了保证日常使用的性能，应确保日志都在所分配的主日志范围内。辅助日志文件会在所有到数据库的连接全部关闭后被删除。

如果设定 LOGSECOND 为 -1，则不限制记录日志的大小；然而，我们不推荐这样做，因为您的文件系统空间会因此而耗尽。

11.3 日志记录的类型

有两种日志记录类型：循环日志记录（默认）和归档日志记录。

11.3.1 循环日志记录

图 11.2 演示了循环日志记录是如何工作的。

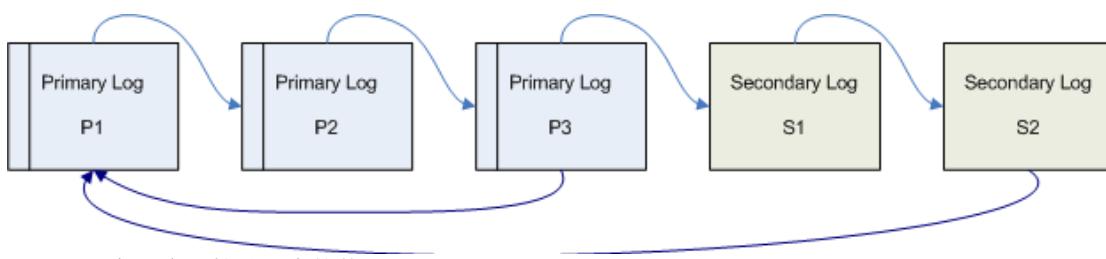


图 11.2 - 主日志和辅助日志的使用

在图 11.2 中，我们可以假设 LOGPRIMARY 参数被置为 3，于是图中有三个主日志。为简单起见，我们在这个例子中只讨论仅有一个事务被执行的情况。当这个事务被执行时，开始占用日志文件 P1 的空间，接着是 P2 的空间。如果一个提交操作发生，接着信息被外部化到表空间磁盘中，那么 P1 和 P2 就可以被覆盖了，这是因为崩溃系统恢复（本章的后面将更详细的谈到）已经不再需要这些信息了。另一方面，如果事务很长，以至于它占用了 P1, P2, P3 的空间后，由于该事务既没有被提交也没有被外部化而需要更多的日志空间，那么辅助日志（图中的 S1）就会被动态的分配。如果该事务依然在继续，更多的辅助日志就会被分配，直到 LOGSECOND 所规定的最大辅助日志数目都被分配完毕。这时候如果还需要更多的日志，用户就会看到一个显示日志已满的错误信息，并且当前事务将会被回滚。作为另外一种选择，您可以将 DB2 的参数设置为 BLK_LOG_DSK_FUL，从而在某些事务被挂起的同时可以继续每隔五分钟向日志写入，这种方法可以给 DBA 提供时间去找新的空间，使事务可以继续下去。

循环循环日志记录可以帮助系统故障修复，但是如果您希望恢复到指定的时间点，最合适的时间点就是您上一次的离线备份。

11.3.2 归档日志

在归档日志（也被称为日志保留记录）中，日志文件不会被覆盖，而会在线或离线保存。在线归档日志与系统恢复所需的活动日志保存在一起，离线归档日志被移动到另外的存储媒介，例如磁带中。这个操作可以由 **USEREXIT** 程序，Tivoli 存储管理程序，或第三方日志类产品完成。

若要激活归档日志，把数据库配置参数 **LOGARCHMETH1** 和（或）**LOGARCHMETH2** 设置成 OFF 以外的值。另一个方法是把设置参数 **LOGRETAIN** 设置成 RECOVERY。这样会自动地将参数 **LOGARCHMETH1** 设置为 **LOGRETAIN**。但是 **LOGRETAIN** 参数主要用于与 DB2 老版本的兼容性，并不建议使用。

归档日志通常被用在生产系统中，并且由于日志都被保存，使得数据库可以在大多数情况下被恢复到最早的日志状态。有了归档日志，DBA（数据库管理员）可以使数据库从人为造成的错误中恢复。举例来说，如果一个系统用户不经意的执行了一个错误的事务，几天后，当这个问题被检测到时，DBA 可以使系统恢复到这个问题发生以前的状态。然而，为了保证正确恢复这个事务，可能需要一些手动处理的过程。

在需要向前回滚恢复和在线备份的时候需要归档日志。图 11.3 描述了归档日志的过程。

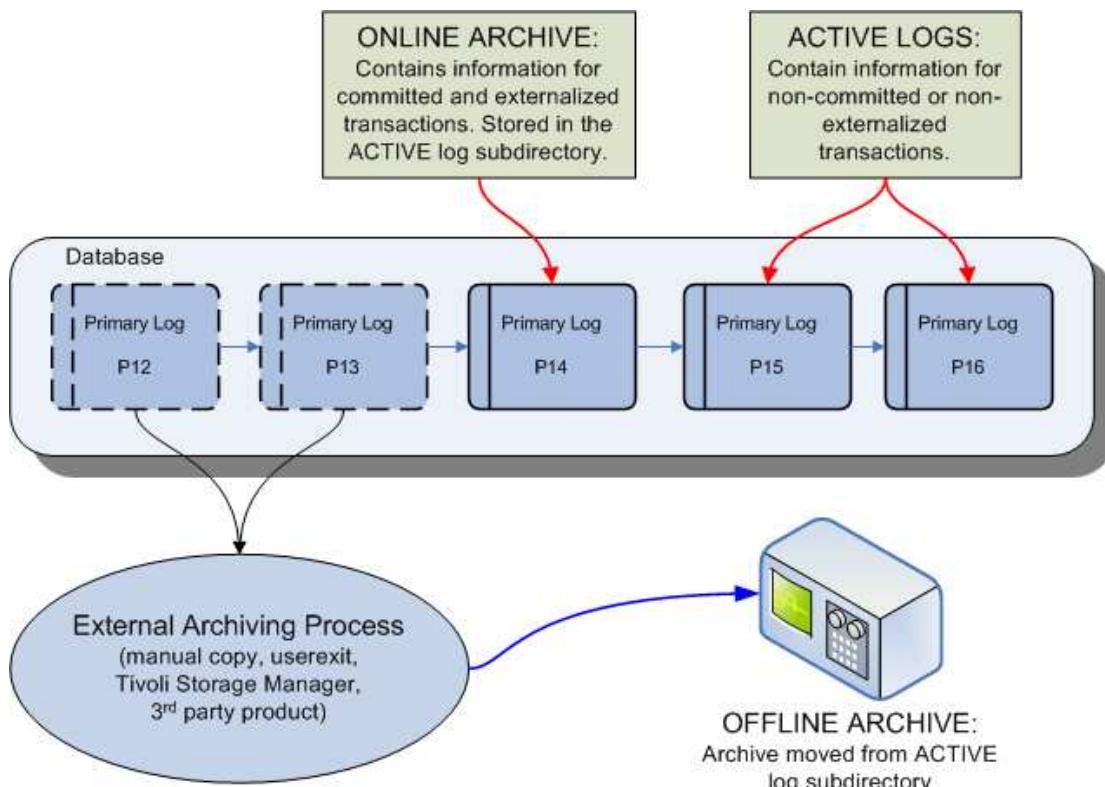


图 11.3 - 档案日志记录

11.4 从控制中心配置数据库日志

在控制中心，您可以通过右键单击数据库，选择“配置数据库日志（Configure Database Logging）”来配置数据库的日志记录。图 11.4 进行了详细描述。

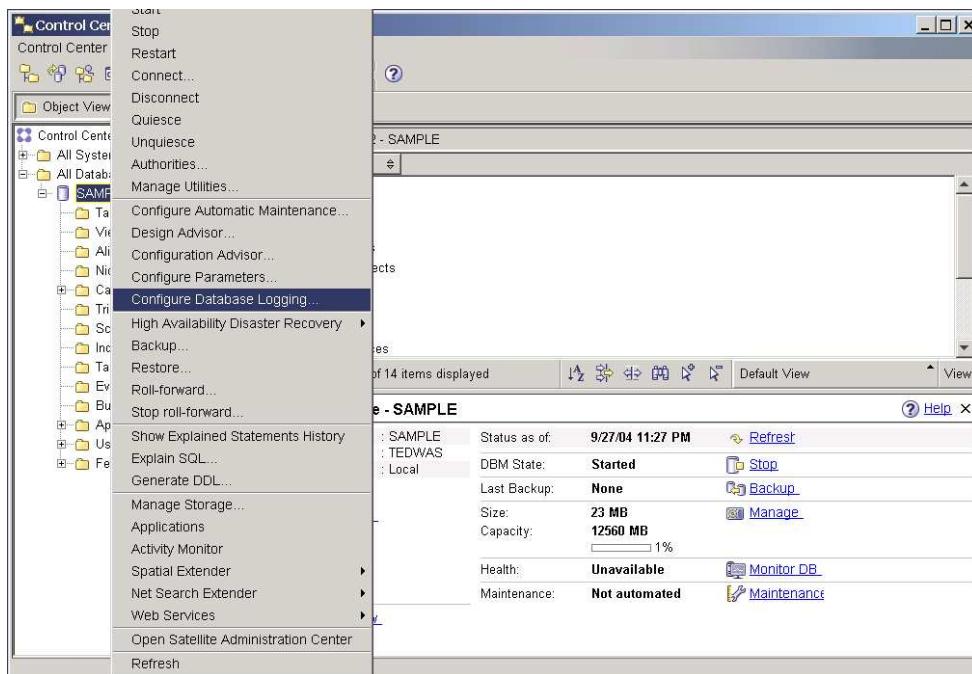


图 11.4 - 在控制中心中进行数据库日志配置

图 11.5 中显示了数据库的日志记录配置向导，在这里您可以选择循环日志记录或者归档日志。

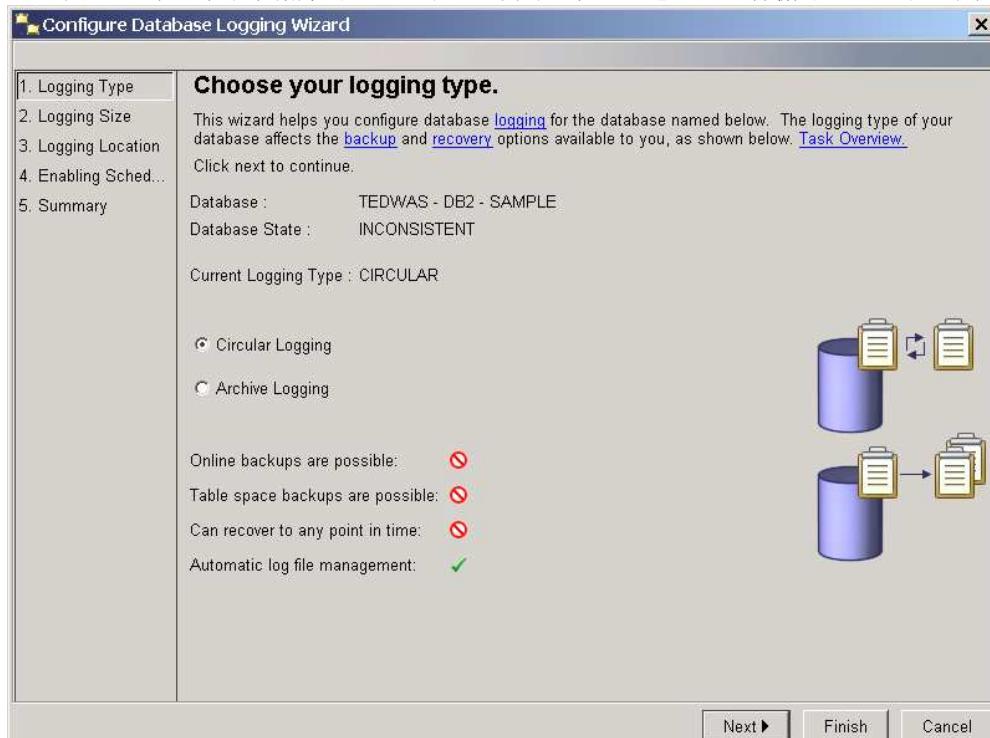


图 11.5 – 数据库日志记录向导

11.5 日志的参数

有一些关于日志的数据库配置（DB CFG）参数。表 11.1 列出了其中主要的参数。

参数	描述
<code>logbufsz</code>	在日志记录被写到磁盘之前，被用来给日志记录做缓冲的内存大小
<code>logfilsz</code>	每个设定的日志的大小，以 4KB 页为单位
<code>logprimary</code>	被创建的大小为 <code>logfilsz</code> 的主日志的数目
<code>logsecond</code>	在需要时被创建的用于系统恢复的辅助日志的数目
<code>newlogpath</code>	数据库活动日志和在线归档日志最先建立在数据库目录下的 <code>SQLOGDIR</code> 子目录里，您可以通过修改这个配置参数值指向不同的目录和设备。
<code>mirrorlogpath</code>	为了保护主日志路径上的日志不受到磁盘错误和意外的删除操作的影响，您可以指定主日志存放点的辅助（映像）路径。
<code>logarchmeth1</code> <code>logarchmeth2</code>	指定一个与数据库活动日志不同的位置存放归档日志文件。如果两个参数都指定，则每个日志文件会存储两次，这意味着您将在两个不同的位置拥有两份归档日志文件的备份。可能的值是 <code>OFF</code> （意味着激活档案日志记录）， <code>LOGRETAIN</code> ， <code>USEREXIT</code> ， <code>DISK</code> ， <code>TSM</code> ， <code>VENDOR</code> 等等。
<code>loghead</code>	当前活动日志文件的文件名
<code>userexit</code>	允许 <code>userexit</code> 离线拷贝日志
<code>softmax</code>	限制系统崩溃恢复的开销
<code>overflowlogpath</code>	与 <code>ROLLFORWARD</code> 命令的 <code>OVERFLOW LOG PATH</code> 选项相似；然而，您可以只用设定一次该配置参数，而不用为了每一个 <code>ROLLFORWARD</code> 命令都指定一次 <code>OVERFLOW LOG PATH</code> 选项。
<code>blk_log_dsk_ful</code>	设定该参数用以在 <code>DB2</code> 不能够在归档日志路径创建一个新的日志文件时避免磁盘严重错误的产生。取而代之的是， <code>DB2</code> 会尝试着每五分钟创建新日志文件，直到成功。只读的 <code>SQL</code> 就可以继续执行。
<code>max_log</code>	事务的最大活动日志空间所占的百分比
<code>num_log_span</code>	一个活动的工作单元（UOW）的活动日志文件数目
<code>mincommit</code>	写到磁盘之前的提交数目

表 11.1 – 日志记录的参数

11.6 数据库备份

`DB2` 备份命令允许您在该命令执行时获取一个您的数据库的快照拷贝。您执行这条命令的最简单的语法是：

```
BACKUP DATABASE <dbname> [ TO <path> ]
```

大多数命令和工具可以在线或者离线执行。在线意味着其它的用户可能在您执行您的命令时正连接到数据库并执行其他数据库上的操作。离线表示当您执行您的操作时，没有其他的用户连接到数据库。要允许一个在线操作，把关键词 `ONLINE` 加在命令语法里，否则，默认情况下该命令会假定您在离线执行它。

比如，如果您想要备份 `SAMPLE` 数据库到路径 `C:\BACKUPS`，您可以在 `DB2 Window/Linux` 命令行解释器中执行这条命令：

```
db2 BACKUP DB sample TO C:\BACKUPS
```

注意，`C:\BACKUPS` 目录必须在执行这条命令之前就已经存在。同时要确定在您执行上面这条指令时没有其他到数据库的连接，否则您就会收到一条错误信息，因为一次离线的备份不能在有其他连接时执行。

要找出当前是否有其它到数据库的连接，在 DB2 命令行解释器或者 Linux 的 shell 中执行下面命令：

```
db2 list applications
```

要强制到一个实例的所有数据库的所有连接，在 DB2 命令行解释器或者 Linux 的 shell 中执行下面命令：

```
db2 force applications all
```

在一个有很多用户的实用环境中，您可能并不想运行这最后一条命令，否则您会收到许多来自您的生气同事们的电话！同时也要注意到，最后一条命令是异步运行的。这意味着当您以后尝试着执行备份命令时，刚才那条命令可能仍然没有起作用。如果您第一次收到一条错误，等待几秒钟，重复备份命令。

在成功执行备份命令之后，一个包含备份好的数据库映像的文件被生成。该文件的文件名遵循图 11.6 中所示的规则。

Linux/UNIX/Windows



图 11.6 – 备份映像的命名规则

类型“0”表示这个备份是一个完全备份。类型“3”则表示这仅仅是一个表空间备份。节点被定为 NODE0000，表示没有被分割的数据库，这正是带有 DPF 特性的除 DB2 企业版之外的所有 DB2 版本的情形。目录节点也被定为 CATN0000。您可以参考 DB2 手册获取更多细节。

当获取了几个备份且都存储在同一个路径下的时候，文件名最后的时间标记部分被用来区分这些备份映像。正如我们将在下一部分看到的，RESTORE 命令可以利用这个时间标记来从其中的特定备份恢复。

11.7 数据库恢复

一次数据库恢复是指从一个备份和（或者）日志中重建您的数据库。如果您刚从一个备份中重建数据库，您所做的就相当于重新创建一个与您备份的时候一模一样的数据库。

如果归档日志记录在备份前被激活，您就不能仅仅是使用一个备份映像重建数据库，您还要使用日志。正如我们将要在下一部分看到的一样，一个向前回滚的恢复允许您从备份中重建数据库，然后再应用（向前回滚）全部日志进行恢复，或者恢复到某一个特定的时间点。

注意术语“恢复”在这一部分使用很频繁，但是用于恢复的命令却是“RESTORE”。

11.7.1 恢复类型

有三种恢复类型：

- **崩溃或重启恢复**

假设您正在使用桌面电脑运行重要的 DB2 数据库事务。突然系统掉电，或者某些人不小心拔去了电源线：那么数据库会发生什么呢？

下一次您启动电脑并启动 DB2 时，崩溃恢复就会自动执行。在崩溃恢复中，DB2 会自动运行命令 RESTART DATABASE，并且基于活动日志，重新进行或者取消事务。当这条命令完成

后, DB2 保证您的数据库将会被维持在与以前一致的状态, 也就是说, 任何被提交的操作都会被保存, 任何没被提交的操作都会被回滚。

▪ 版本或映像恢复

这种类型的恢复意味着您仅从备份映像中恢复数据库; 所以, 您的数据库会被置于与备份的时候相一致的状态。任何在备份后进行的事务都会丢失。

▪ 向前回滚恢复

使用这种类型的恢复, 您不仅仅从备份映像中恢复数据库, 同时还将运行 ROLLFORWARD 命令在备份恢复的基础上利用日志来恢复数据, 这样一来, 您就可以恢复数据库到一个特定的时间点。这种类型的恢复使数据库损失降到最低。

11.7.2 数据库恢复

使用 RESTORE 命令来从一个备份映像中恢复数据库。以下语法是这个命令的最简单应用:

```
RESTORE DATABASE <dbname> [from <path>] [taken at <timestamp>]
```

比如, 如果您有一个 sample 数据库的备份映像, 名称如下:

Alias	Instance			Year	Day	Minute	Sequence
SAMPLE.0	.DB2INST.NODE0000.CATN0000	20060314131259.001					
Type	Node	Catalog Node		Month	Hour	Second	

您可以执行下面的命令:

```
RESTORE DB sample FROM <path> TAKEN AT 20060314131259
```

11.8 其他关于备份和恢复的操作

以下列出了使用备份和恢复命令还能做的一些事情。我们建议您查阅 DB2 手册以获取更多信息。

- 在一个 32 位 DB2 上备份数据库, 并在一个 64 位 DB2 上重建它
- 恢复并覆盖一个已存数据库
- 使用重定向恢复把数据库恢复到一个与备份映像中所描述的磁盘数目不同的系统
- 仅备份或者恢复表空间, 而不是整个数据库
- 可以进行少量备份或者增量备份; 增量备份仅仅记录从上一次备份到下一次备份之间的改动, 而增量备份记录所有改动并把它们追加到每一个备份映像上。
- 从闪存拷贝中备份 (需要相关硬件支持)
- 恢复删除的表 (如果给定表的该选项被激活)
- 不能从一个操作系统中备份 (比如 windows) 再到在另一个操作系统 (如 Linux) 中重建。在这种情况下应使用 db2look 和 db2move 工具。

11.9 小结

这一章我们探讨了 DB2 日志记录的功能, 包括两种类型的日志 (主日志和辅助日志) 和两种类型的日志记录 (循环日志记录和归档日志记录), 与日志记录有关各种数据库参数。对于每种日志记录类型, 我们讨论了什么时候和为什么使用它, 以及如何从控制中心建立它。

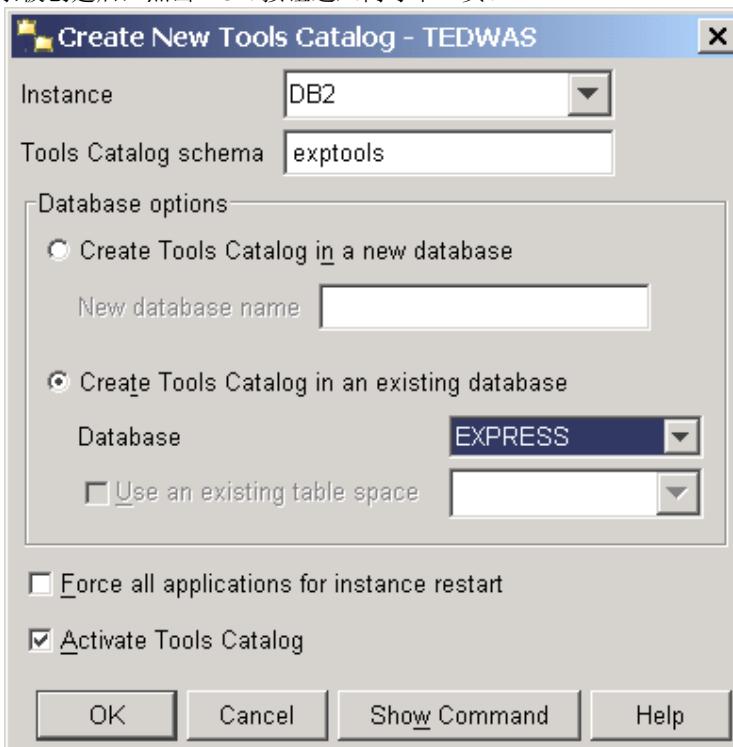
我们也探讨了如何使用 DB2 命令行进行备份和恢复，深入探讨了三种类型的数据库恢复：崩溃，版本，回滚。

11.10 实验

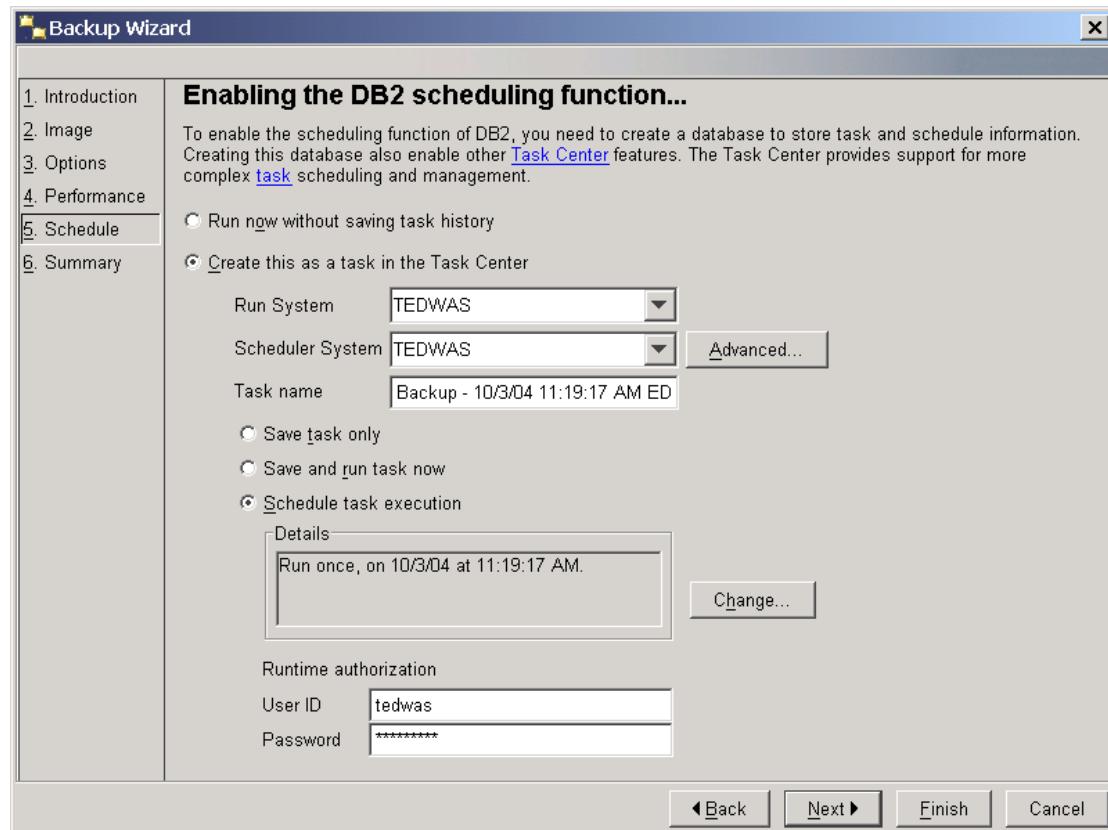
虽然 DB2 能使数据库许多维护工作自动进行，但有些时候您想定制某些维护工作。在这个实验中，您将为 EXPRESS 数据库创建一个定制的夜间备份安排。

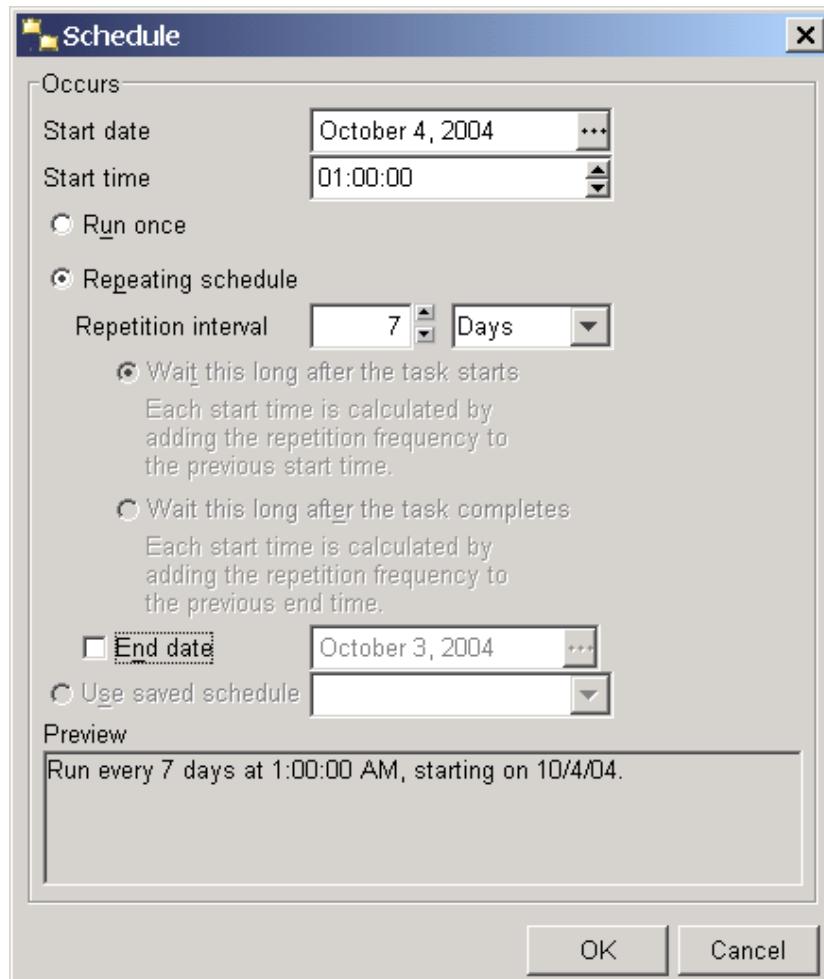
实验过程

1. 在控制中心左边的对象树中找到 Control Center -> All Databases。右键单击 EXPRESS 数据库并选择 Backup 选项。这会启动备份向导 Backup Wizard。
2. 这个向导的 Introduction 页概述了当前数据库的状态，包括最后一次备份的时间和日志记录的方法。点击 Next 按钮进入下一页。
3. 在该向导的 Image 页，选择映像文件要存储的目标位置。通常，您应该选择一个与现存数据库所在位置不同的物理驱动器。现在，在文件系统中创建一个新文件夹 C:\db2backup，并设定该文件夹为备份位置。在向导中，从媒介类型 Media Type 下拉列表里选择文件系统 File System。点击 Add 按钮，选择您刚刚创建的文件夹，并点击 OK。点击 Next 进入下一页。
4. 您可以仔细研究 Options 页和 Performance 页，但是默认的选项常常足够了，因为 DB2 自动地以最佳方式进行数据库的备份。在您设置完毕后，请导航至 Schedule 页。
5. 在计划 Schedule 页中，如果计划任务没有被激活，现在就选择激活它。选择要创建工具目录 (tools catalog) 的系统并创建一个新工具目录。给工具目录指定一种模式并且选择在已有的 EXPRESS 数据库中创建它。这个工具目录包含有关每个计划任务的元数据。点击 OK 按钮继续。当工具目录被创建后，点击 Next 按钮进入向导下一页。



6. 在 Schedule 页上，选择为任务的执行创建一个计划。安排备份每天从凌晨 1 点开始执行。点击 Next 按钮进入下一页。





7. 在总结页面，您可以检查一下将要被创建的计划任务。一切妥当后，点击 **Finish** 按钮创建这个任务。
8. 启动任务中心 **Task Center** 查看或者修改刚刚创建的备份任务。

12

第 12 章 – 维护任务

本章讨论维护数据库的各种任务。一般来说，DB2 会自动执行其中大部分任务的。DB2 Express-C 版本，和其他所有现存 DB2 版本一样，拥有这些自动维护的能力。这种自我管理能力对于不能雇用一个全职数据库管理员来管理数据服务器的中小型公司是十分有好处的。另一方面，假使雇用了数据库管理员，他或她也会因此而获得更多的时间，来处理其他对公司有益的高级活动。

注意：

更多关于数据库维护的信息，请参见以下视频：

<http://www.channeldb2.com/video/video/show?id=807741&Video:4302>

12.1 重组 (REORG)、运行统计 (RUNSTATS)、重绑定 (REBIND)

DB2 有三个主要的维护任务，在图 12.1 中给出了描述：重组 (REORG)、运行统计 (RUNSTATS)、重绑定 (REBIND)。

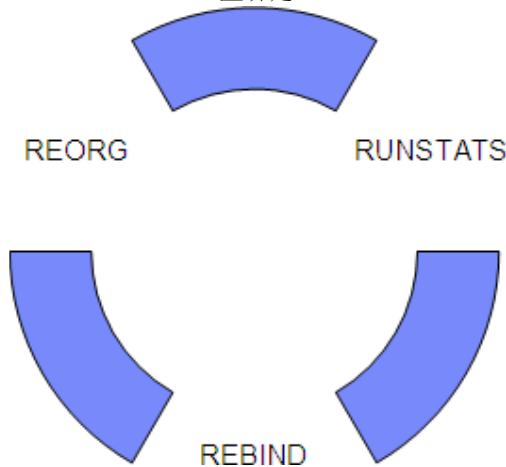


图 12.1 – 维护任务：重组 (REORG)、运行统计 (RUNSTATS)、重绑定 (REBIND)

图 12.1 显示了各种维护任务是以一种循环的方式进行的。如果执行了 REORG，那么建议您接着运行 RUNSTATS 和 REBIND。一段时间之后，数据库中的表会被 UPDATE, DELETE 和 INSERT 等操作修改。这时候这个循环就会从 REORG 开始重新启动。

12.1.1 重组 (REORG) 命令

随着时间流逝，您不断在数据库上执行 UPDATE, DELETE 和 INSERT 等操作，您的数据在数据库页之间变得越来越支离破碎。REORG 命令回收浪费的空间并重新组织数据，从而获得更高的运

行效率。被频繁修改的表能从 REORG 命令中获得最大的利益。您还可以重组索引，而且，REORG 命令在线或者离线都可以执行。

离线 REORG 命令更快更有效，但不允许对数据表访问。而在线的 REORG 命令允许对表的访问，但会消耗大量系统资源，这对于小型的表最为有利。

语法：

```
REORG TABLE <tablename>
```

例子：

```
REORG TABLE employee
```

REORGCHK 命令可以在 REORG 之前使用以检查一个表或者一个索引是否需要被重组。

12.1.2 运行统计 (RUNSTATS) 命令

DB2 优化器是 DB2 的“大脑”。它为定位或者获取数据找到最有效的路径。优化器是系统的价值所在，它使用存储在目录表中的数据库对象统计信息来最优化数据库的性能。目录表存有关于一个表中当前有多少列，多少行，表有多少个索引，索引是什么类型之类的统计信息。

统计信息不是动态更新的。因为您不会想要 DB2 在每次数据库操作后都更新统计信息，这有可能对整个数据库性能产生不利的影响。取而代之的是，DB2 提供了 RUNSTATS 命令来更新统计信息。保持数据库统计信息的更新十分重要。如果 DB2 优化器认为一个表有一行而不是一百万行，就可能从根本上改变访问路径。如果数据库的统计信息是最新的话，DB2 可以选择一个更好的访问策略。更新统计数据的频率应该由表格中数据的变动频率决定。

语法：

```
RUNSTATS ON TABLE <schema.tablename>
```

例子：

```
RUNSTATS ON TABLE myschema.employee
```

12.1.3 绑定 / 重新绑定

在成功执行 RUNSTATS 命令之后，并不是所有的查询都会使用最新的统计信息。静态的 SQL 访问策略在执行 BIND 命令时确定，因此这时候使用的统计信息有可能并不是和当前一致的。图 12.2 举例说明了这个问题。

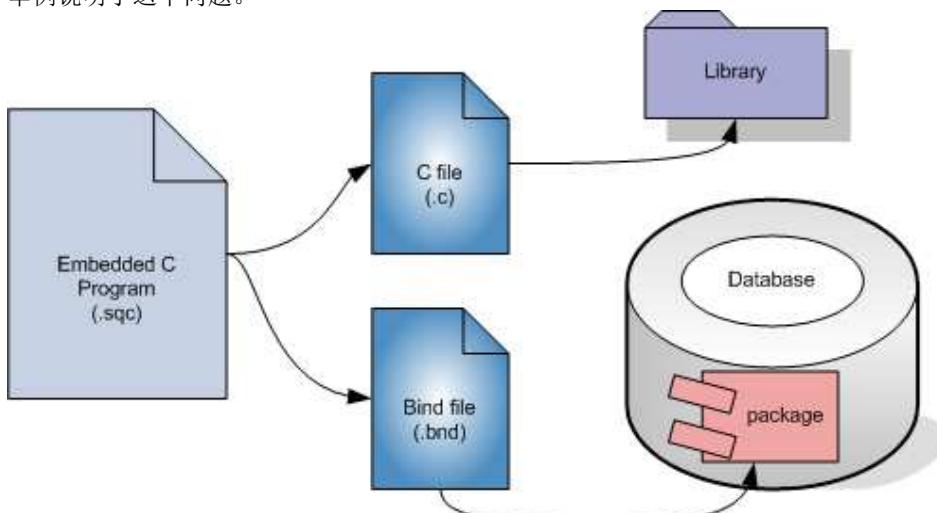


图 12.2 – 静态的 SQL 绑定过程

在图 12.2 中一段嵌入的 C 程序（存储在“sqc”文件类型中）被预编译。预编译后，会产生两个文件，一个“.c”文件包含了 C 代码以及所有的 SQL 注释；一个“.bnd”文件包含了所有的 SQL 语句。扩展名为“.c”的 C 语言文件像以往一样由 C 编译器编译，创建一个如图中右上角所示的“库”。“.bnd”文件进行类似的处理，生成一个存储在数据库中的包。绑定相当于以当时可用的统计信息为基础，用一种最快的访问策略编译 SQL 语句，并把它们存在一个包里。

那么，用个嵌入的 C 程序调用 SQL 把一百万行插入到一个正在使用中的表中时会发生什么呢？在插入之后，若执行了 RUNSTATS 命令，统计信息就会被更新。然而这个绑定的包是不会利用更新的统计信息来重新计算访问路径的。可以使用 db2rbind 命令利用更新的统计信息重新绑定所有的包。

语法：

```
db2rbind database_alias -l <logfile>
```

例子：

要重新绑定 sample 数据库里所有的包，并把输出日志存储在一个 log.txt 文件中，执行这条命令：

```
db2rbind sample -l mylog.txt
```

12.1.4 在控制中心执行维护工作

在控制中心中您可以执行 REORG 和 RUNSTATS。图 12.3 显示了如何来操作。

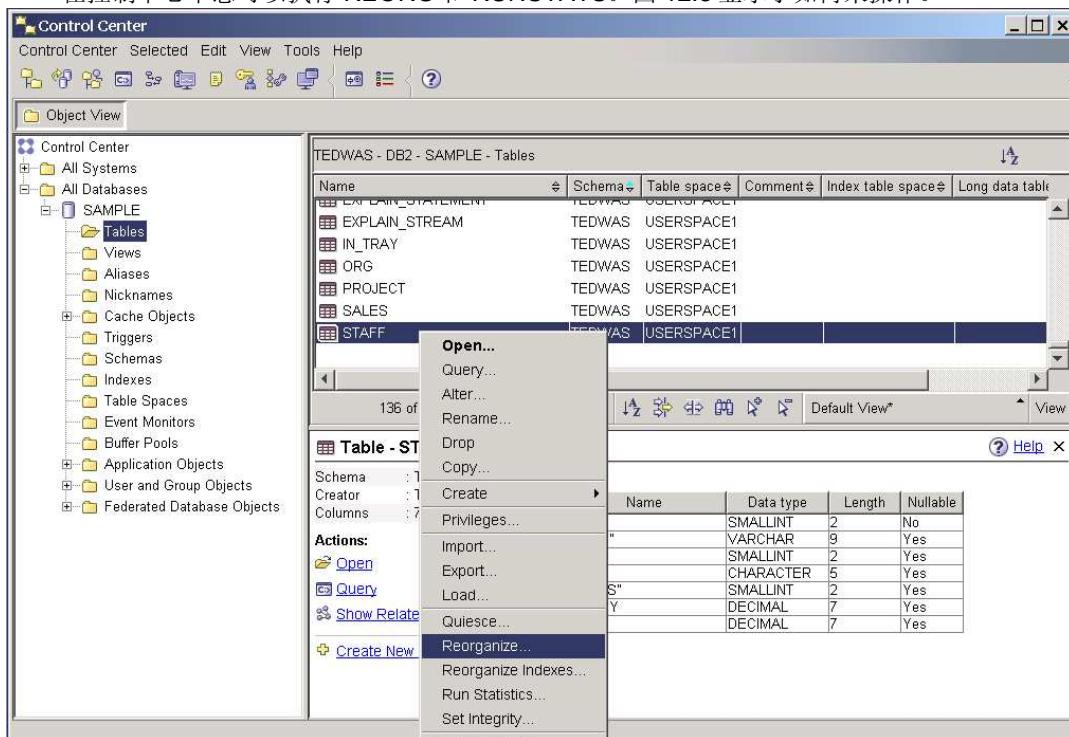


图 12.3 – 控制中心里的 REORG 和 RUNSTATS

选择您要操作的表，右键单击它并选择 Reorganize（即 REORG）或者 Run Statistics（即 RUNSTATS）。

数据库的操作视图

当您选中一个数据库，控制中心右下角的数据库操作视图就会提供有关数据库的信息，比如它的大小，最后一次备份是在什么时候，是否设定了自动维护等等。这个视图允许您快速识别出您的数据库需要的维护。图 12.4 显示了这些信息。

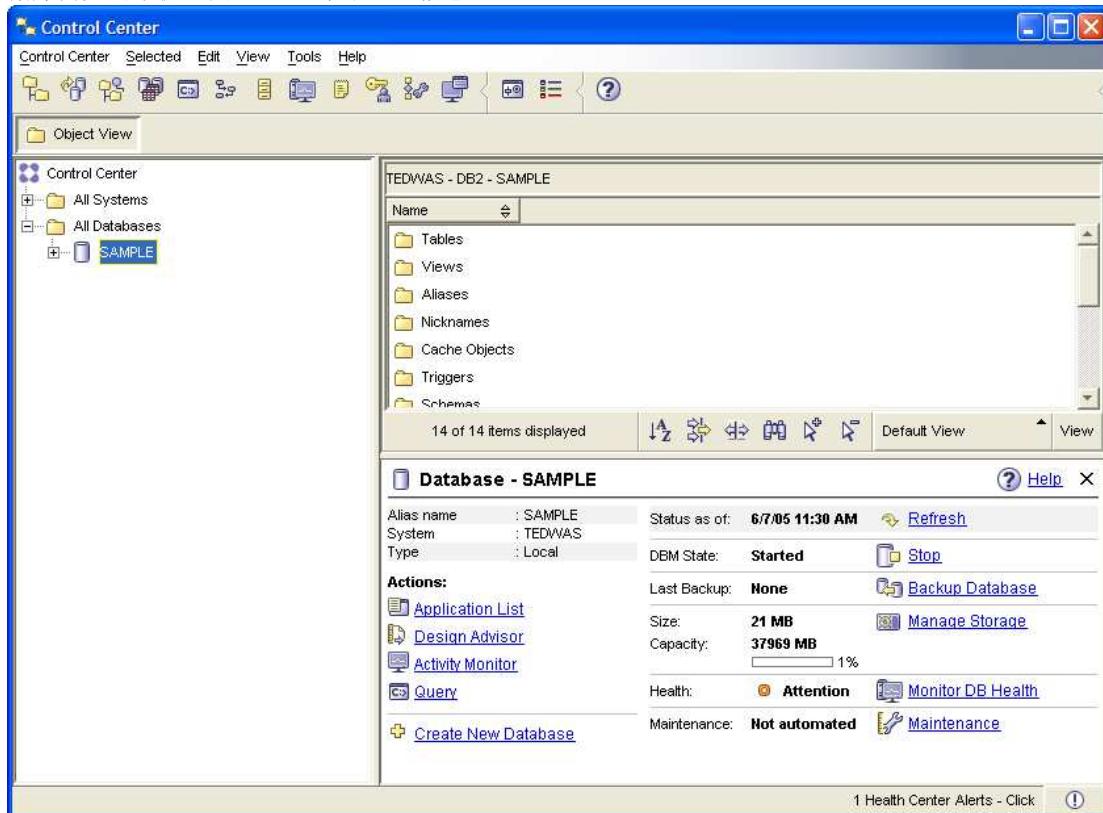


图 12.4 – 控制中心的数据库操作视图

12.2 维护方式

有三种方式进行数据库的维护：

1. 手动维护

在需要的时候进行手动维护。

2. 创建维护脚本

您可以用维护命令来创建脚本，并让它们按时间表执行。

3. 自动维护

让 DB2 自动的为您执行维护 (REORG, RUNSTATS, BACKUP)。

这部分我们主要关注自动维护。

自动维护由以下组成：

- 用户定义一个维护窗口期，在其中可以最小干扰的执行维护任务。比如，如果系统在周日凌晨 2 点到 4 点活动最少，这个时间段就可以用作一个维护窗口期。
- 有两种维护窗口期，一种适用于线上操作，一种适用于线下操作。
- 在维护窗口期，仅当有维护需要时 DB2 才自动执行维护操作。

在控制中心中，您可以启动图 12.5 所示的自动维护配置向导 (Configure Automated Maintenance Wizard)。

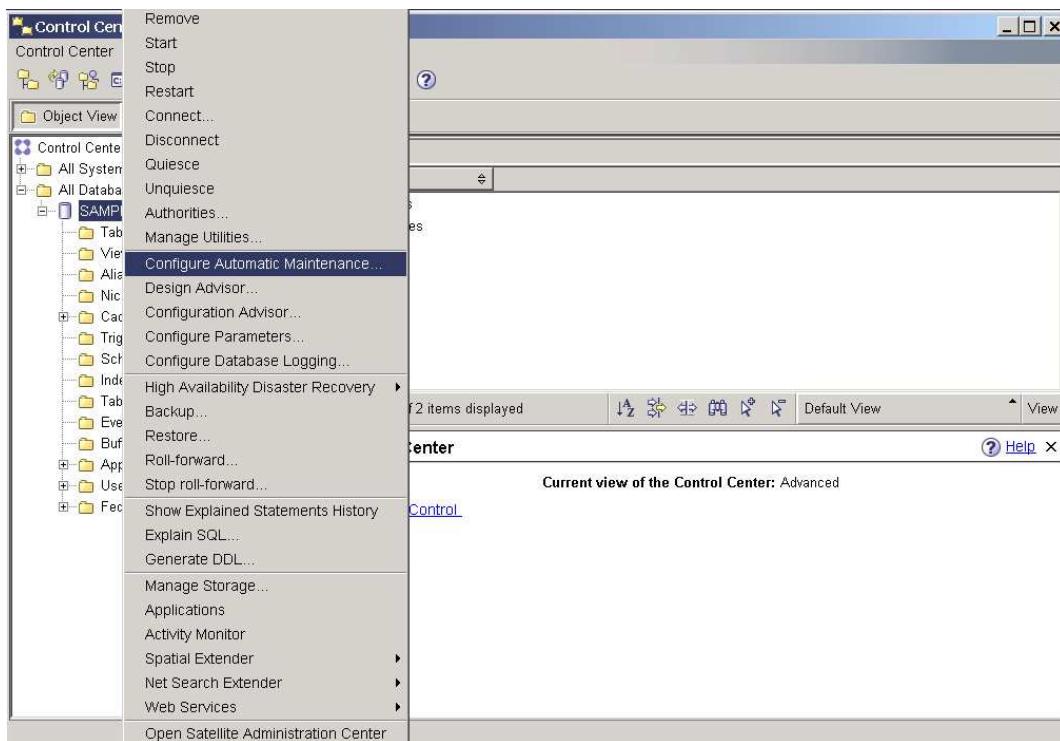


图 12.5 – 运行自动维护配置向导 (Configure Automated Maintenance Wizard)

图 12.6 展示了自动维护配置向导 (Configure Automated Maintenance Wizard)。

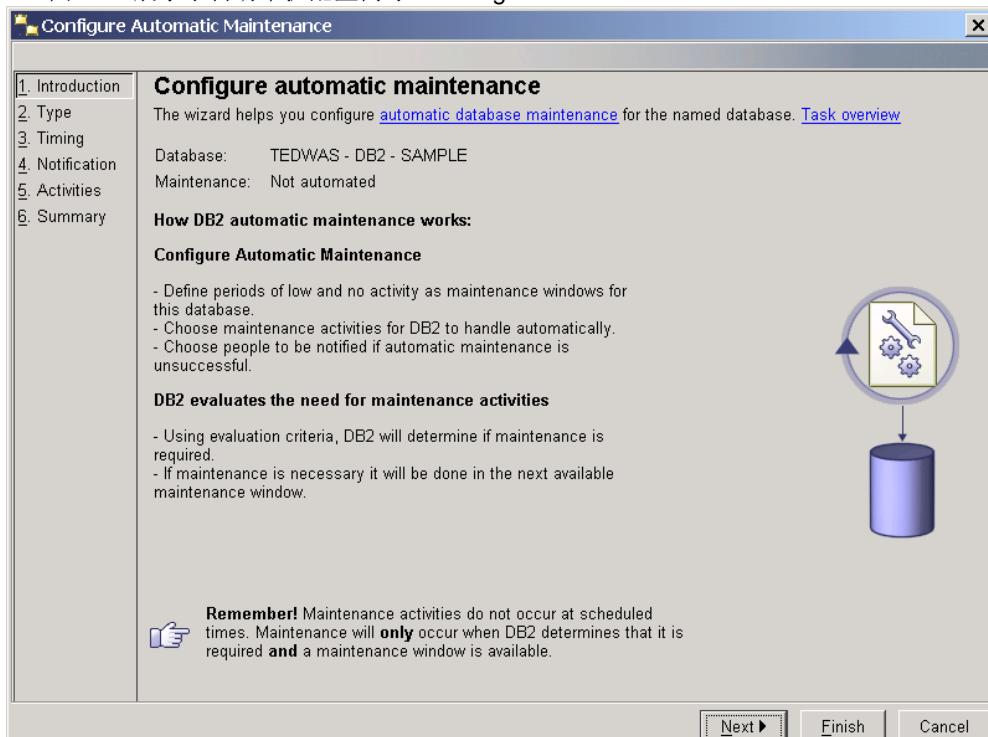


图 12.6 自动维护配置向导 (Configure Automated Maintenance Wizard)

12.3 小结

本章讨论了维护对您的数据库的重要性，包括 REORG, RUNSTATS, REBIND 循环的作用。REORG 命令，顾名思义，就是整理您的数据以消除碎片和实现高速数据检索。RUNSTATS，即通过 DB2 优化工具更新统计信息来提高数据的性能。BIND 或 REBIND 过程更新数据库包的最近访问路径。

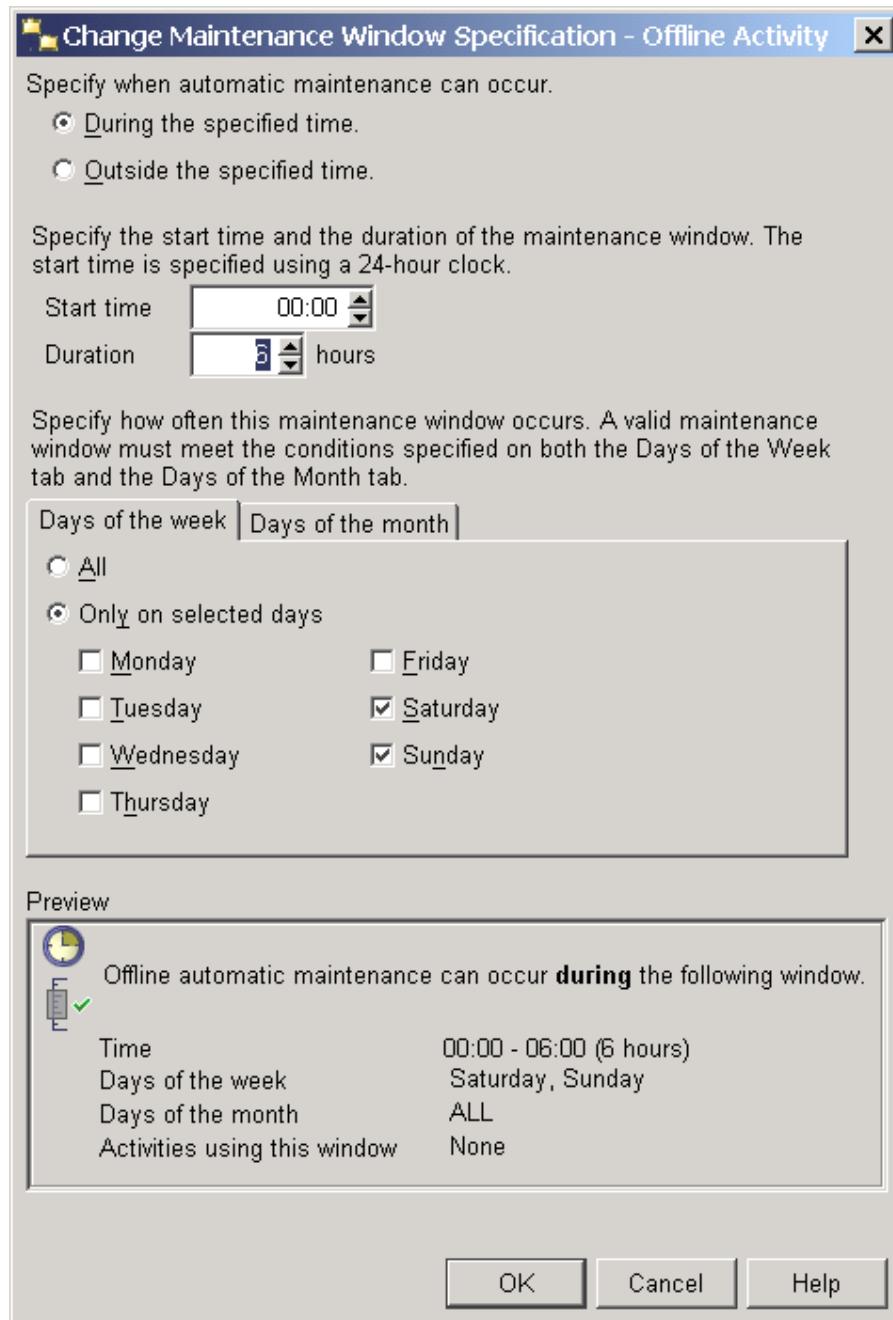
我们也讨论了 DB2 控制中心提供的图形工具，通过手动，脚本，自动模式进行维护活动。

12.4 实验

在本实验中，您将对 DB2 的 SAMPLE 数据库配置自动维护。

实验过程

1. 在控制中心对象树中，右键单击 SAMPLE 数据库并选择配置自动维护（Configure Automatic Maintenance）菜单项。这将运行自动维护配置向导。
2. 向导的起始页（Introduction）显示了当前自动维护的设定。如果您创建数据库时使用了自动维护选项，自动维护就已经配置过了。您可以使用该向导重新配置自动维护的选项。点击 Next 按钮进入下一页。
3. 向导的类型页（Type）让您选择关闭所有自动维护还是变更自动维护选项。选择变更自动维护选项。点击 Next。
4. 时间选择页（Timing）让您设定维护窗口期。按照下图设定离线窗口期为每周六和周日的午夜到早上六点。点击离线维护窗口期预览面板旁边的 Change 按钮并设定想要的时间。在设定这些必要信息后，点击 OK 按钮回到向导。不改变在线窗口期设定（在线维护随时都可以进行）。点击 Next 按钮。



- 5.在向导的通知页（Notification），您可以建立一个联系方式，当自动维护失败时会联系您。现在跳过这一步，点击 Next 按钮。
- 6.在向导的活动页（Activities），您可以选择某些活动为自动或者某些活动不自动，也可以选择进行某些活动要通报。在这个例子中，请确保所有的自动（Automate）选框都被选中，所有的通报选项都被取消。点击 Next 按钮。
- 7.在进入下一页之前，您应该要配置数据库备份的地址。理论上，您应当把备份存储在一个不同的物理磁盘上，防止磁盘错误。在活动页（Activities），选择数据库备份选项（Backup database），点击配置设定 Configure Settings 按钮。

8. 在配置设定（Configure Settings）对话框的备份标准（Backup Criteria）标签，选择平衡数据库恢复和运行性能（Balance Database Recoverability with Performance）选项。在备份地址栏，选择当前地址并点击 Change 按钮。设定一个不同的地址进行备份（请确保该磁盘有足够的空间）。在备份模式（Backup Mode）标签，请确保选择了离线备份（Offline Backup）。点击 OK 按钮关闭备份标准标签。点击 Next 按钮。
9. 配置自动维护向导的总结页包含了您的所有选择的总结。点击 Finish 按钮同意并执行这些改变。

13

第 13 章 – 并行与锁定

本章讨论如何允许多用户互不干扰地同时访问一个数据库，并且确保他们操作的完整性。我们将会讨论事务、并行和锁定的概念。

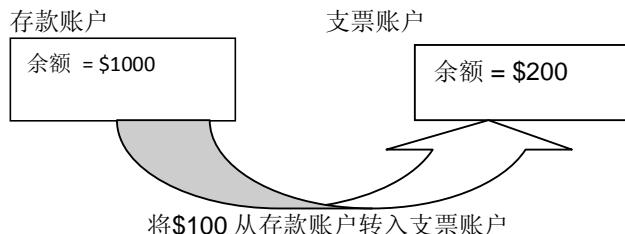
注意:

更多关于并行与锁定的信息，请参见以下视频：
<http://www.channeldb2.com/video/video/show?id=807741:Video:4322>

13.1 事务 (Transactions)

一个由若干条 SQL 语句组成的事务或者工作单元在执行时应被视为一个独立的整体；即，如果事务中有一条语句执行出错，整个事务都会失败，事务内部在故障点前执行的语句也会回滚。

COMMIT 语句是一个事务的结束，也标志着一个新事务的开始。图 13.1 是一个事务的例子。



- 从存款账户取出\$100
- 向支票账户存入\$100

图 13.1 – 一个事务的例子

如图 13.1 所示，您想要把 100 元从您的储蓄账户转入您的支票账户。为了完成这项工作，需要以下的事件序列：

从储蓄账户取出\$100
向支票账户存入\$100

如果不将上面的事件序列看作一个单独的事件单元，想象一下，在把\$100 从存款账户取出后，若在存入支票账户前突然断电，会发生什么？您将损失\$100！

13.2 并行 (Concurrency)

并行意味着多个用户可以在同一时间操作同一个数据库对象。DB2 是一个多用户数据库。它使用一种机制合理协调多个用户对数据库访问，每一个用户不会感觉到其他用户的存在，并能确保数据的完整性与一致性。如图 13.2 所示：

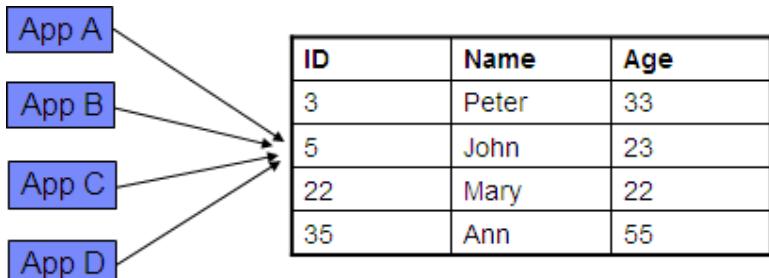


图 13.2 – 一个并行的例子，需要并发控制

在图 13.2 中，有四个应用程序（App A, App B, App C, 和 App D）试图访问表中的同一行（第二行）。如果没有并发控制，所有的程序可能会对同一行进行操作。假设所有的程序都想把第二行中的“Age”项改为不同的值，那么最后一个更新 Age 列的程序似乎将为这种情形下的“胜出者”。很明显地，在本例中，需要某种并发控制来保证结果的一致性。这个并发控制是基于使用锁定的。

锁定与并行的概念是紧密关联的。锁定暂停其它程序的操作直到当前操作完成。一个系统里的锁定越多，并行的可能性也就越小。反之，锁定越少，并行的可能性越大。

锁定会在需要一个事务时被自动获取，在事务终止时被释放（使用一条 COMMIT 或 ROLLBACK 命令）。锁定可以是针对行或列的。锁定有两种：

- 共享锁定（S 锁定） - 当程序企图读并且禁止其它程序修改同一行时需要用到的锁定
- 互斥锁定（X 锁定） - 当一个程序修改，插入或删除一行时需要用到

现在考虑图 13.3，它与图 13.2 很接近，但它现在展示的是一个锁定。

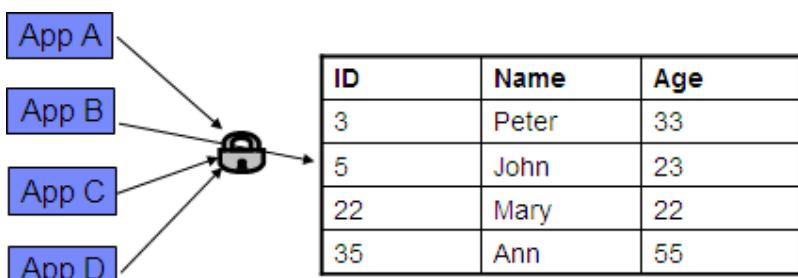


图 13.3 – 一个并行需要锁定的例子

例如在图 13.2 中，如果 App B 最先访问第二行并且要进行修改，App B 就掌握着对该行的 X 锁定。当 App A, App C 和 App D 试图访问同一行时，由于 X 锁定的缘故，它们就无法对其进行修改。这项控制可以保证数据的一致性与完整性。

13.3 无并行控制导致的问题

如果没有并发控制机制，数据库可能会遇到以下问题：

- 丢失更新（Lost update）
- 未落实的读（Uncommitted read）
- 不可重复读（Non-repeatable read）
- 幻象（Phantom read）

13.3.1 丢失更新 (Lost update)

本节中曾解释了在何种情况下最后进行修改操作的程序会成为“胜出者”，丢失更新的问题与此相似。

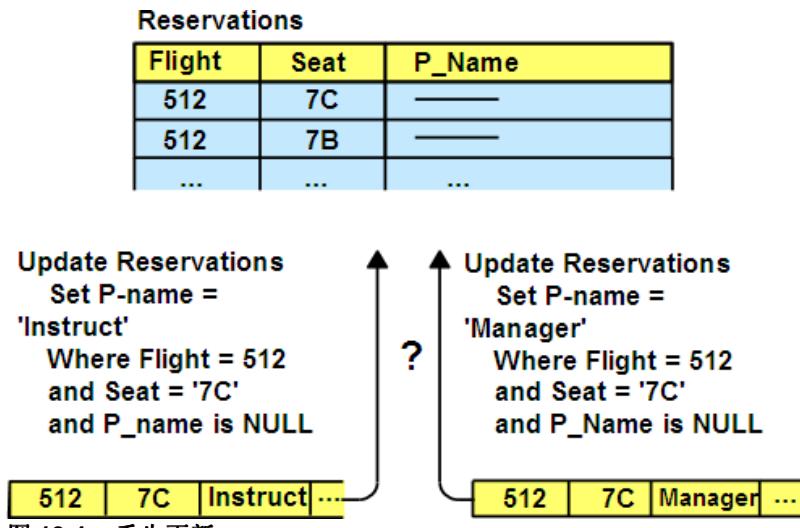


图 13.4 – 丢失更新

在图 13.4 中，有两个程序企图修改同一行。左边的一个是程序 App1，右边的是程序 App2。事件发生的序列如下：

1. App1 修改一行
2. App2 修改同一行
3. App1 提交
4. App2 提交

App2 进行更新时，App1 的更新就丢失了。所以它们“丢失了更新”。

13.3.2 未落实的读 (Uncommitted read)

一个未落实的读，或谓之“脏读”允许一个程序读取未提交的信息，因此读出的数据不保证准确性。

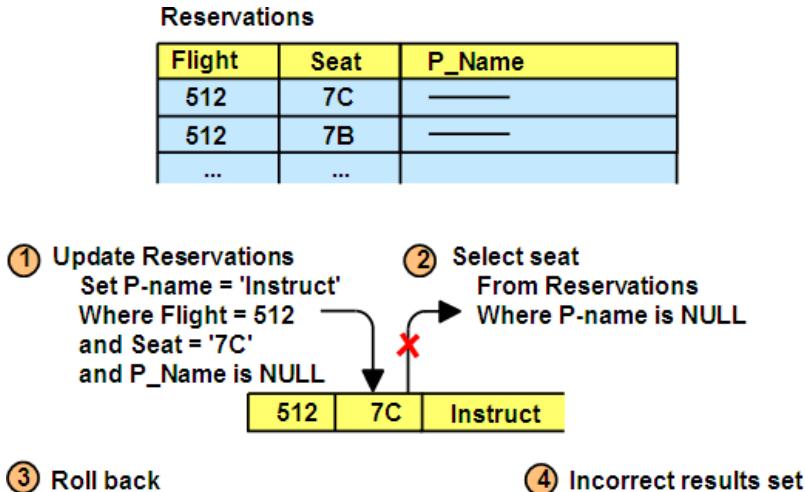


图 13.5 – 未落实的读

图 13.5 依照的是如下的事件序列:

1. App1 修改一行
2. App2 读取该行的新值
3. App1 回滚它对该行的更改

App2 读取的是未提交的数据，所以该数据是无效的，这便是为什么这个问题会被称作“未落实的读”。

13.3.3 不可重复读 (Non-repeatable read)

一个不可重复的读意味着您无法在执行完这一读操作之后，再得到同样的结果。

FLIGHT	SEAT	NAME	DESTINATION	ORIGIN
512	7B	—	DENVER	DALLAS
....				
....				
814	8A	—	SAN JOSE	DENVER
....				
134	1C	—	HONOLULU	SAN JOSE
....			

Figure 13.6 – 不可重复读

在图 13.6 中，考虑如果您正试图订购从达拉斯 (Dallas) 到火奴鲁鲁 (Honolulu) 的机票，事件的序列是：

1. App1 打开一个游标（亦即结果集）获取您在图 13.6 中看到的内容。
2. App2 删除游标限定的一行（例如目的地 (destination) 是 “San Jose” 一行）。
3. App2 提交更改。
4. App1 关闭并重新开启游标。

在此情况下，因为 App1 在一次重复的读中不会得到同一份的数据，所以它不能够重新产生这个数据集；这就是为何这个问题被称作“不可重复读”。

13.3.4 幻象 (Phantom read)

幻象的问题与不可重复读相似，不同之处在于随后的行读取上。您可能会获取额外一些行。

图 13.7 展示了这个问题的一个例子：

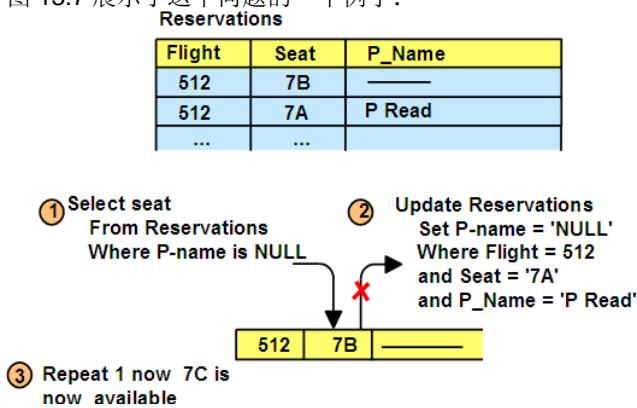
**Figure 13.7 – 幻象**

图 13.7 展示了如下的事件序列：

1. App1 开启一个游标
2. App2 向数据库中添加了一个行，该行与游标匹配
3. App2 提交更改
4. App1 关闭并且重新开启游标

在此情况下，App1 在重复的读取中获取的将不是同一数据，它会得到更多的行。这就是为何此问题被称作“幻象”。

13.4 隔离级别 (Isolation Levels)

您可以把隔离级别设想为一些锁定规则。依据隔离级别的不同，一个程序会产生不同的数据库锁定行为。

DB2 提供了不同的保护级别来隔离数据：

- 未落实的读 (UR)
- 游标稳定性 (CS)
- 读稳定性 (RS)
- 可重复读 (RR)

13.4.1 未落实的读

未落实的读亦称“脏读”。它是最低的隔离级别，并且提供最高的并行性。除非另一个程序企图删除 (drop) 或者更改 (alter) 整个表，否则读操作时没有行锁定；而修改 (update) 操作与游标稳定性级别相同。

此隔离级别仍存在的问题：

- 未落实的读
- 不可重复读
- 幻象

此隔离级别所防止的问题：

- 丢失更新

13.4.2 游标稳定性

游标稳定性是默认的隔离级别。它提供了低程度的锁定。在这一隔离级别中，游标的“当前”行是被锁定的。如果该行只是被读的，锁定会一直持续到一个新行被访问或者该工作单元终止。如果该行被修改，锁定会一直持续到该工作单元终止。

此隔离级别仍存在的的问题：

- 不可重复读
- 幻象

此隔离级别所解决的问题：

- 丢失更新
- 未落实的读

new in
V9.7

13.4.2.1 当前提交

在 DB2 9.7 使用光标稳定性隔离级别之前，一个写操作（UPDATE）将阻止同一行的读操作（SELECT）。其中的逻辑是，写操作正在修改行，读操作应该等到更新完成后看到最终提交的值。在 DB2 9.7 中，有一个新数据库的游标稳定性隔离级别的默认行为。实施这一新的行为时，将使用当前提交 CC（currently committed）的语义。有 CC 之后写操作不会阻止访问同一行中的读操作。如果您使用了隔离级别未提交的读操作（UR），这种情况过去是可能发生的。但现在的差别是，UR 读操作读到的是未提交的值，CC 读操作读到的是提交的值。当前提交值是致力于写操作的开始前的值。例如，表 T1 具有以下内容：

FIRSTNAME	LASTNAME
Raul	Chong
Jin	Xie

现在，您的应用程序 AppA 执行这个语句，但不提交：

```
update T1 set lastname = 'Smith' where firstname = 'Raul'
```

接下来，应用程序 AppB 执行此提交：

```
select lastname from T1 where firstname = 'Raul' with CS
```

在 DB2 9.7 之前，这条语句将挂起，因为它正在等待由 AppA（写操作）更新语句的排他锁被释放。在 DB2 9.7 当前提交（新数据库默认）能让语句返回当前提交的值，即 **Chong**。请注意，即使 CS 是默认值，为清楚起见我们还是把“with CS”包括在提交语句中。我们将在后一章节讨论这个子句。

如果 AppB 尝试这个语句：

```
select lastname from T1 where firstname = 'Raul' with UR
```

由于 UR 的隔离使用，其结果将是未提交的值 **Smith**。这个例子说明，CC 程序具有更好的并发功能使读操作能够读到一个正在更新的行。另一个在 DB2 9.7 之前会引起争议的情况是读操作阻止写操作访问行。这就是为什么即使是读操作也建议使用一个提交的原因之一，因为这将确保共享（S）锁被释放。有了 CC 之后这不再是一个问题，读操作不会阻塞写操作。对于那些没有提交的 INSERT 操作，读操作将默认地跳过他们，结果就是不显示这些行。对于 DELETE 命令，读操作应该也可以跳过（忽略）受影响的行，但行为取决于 DB2 注册表的变量值 DB2_SKIPDELETED。其他注册表变量和 BIND 和 PREPARE 命令属性可以改变 CC 的默认行为。记住：当前提交意味着它只会显示当前提交的信息，因此未提交的 INSERT 或 DELETE 操作将被忽略。如前所述，在新数据库 CC 是默认的。如果您想将其关闭，或把一个早期版本创建的数据库升级到 DB2 9.7，您可以更新数据库配置 CUR_COMMIT 的值。例如，将 SAMPLE 数据库的 CC 关闭可以写成：

```
db2 update db cfg for sample using CUR_COMMIT off
db2stop
db2start
```

13.4.3 读稳定性

使用读稳定性隔离级别时，在同一个工作单元中的一个程序进程所检索的全部行都会被锁定。对于一个给定的游标，它要锁定所有与结果集匹配的行。例如，如果您有一个含 1000 行的表并且查询返回 10 行，那么只有那 10 行会被锁定。读稳定性使用中等级别的锁定。

此隔离级别仍存在的问题：

- 幻象

引隔离级别所解决的问题:

- 丢失更新
- 未落实的读
- 不可重复的读

13.4.4 可重复读

可重复读是最高的隔离级别。它提供了最大程度的锁定和最少的并行。产生结果集的所有行都会被锁定，也就是说，即使不必出现在最终结果集中的行也会被锁定。在该工作单元结束前，任何其它的程序都不能修改、删除或插入一个会影响结果集的行。重复读确保程序在一个工作单元中多次进行的同一项查询都返回相同的结果。

此隔离级别仍未解决的问题:

- 无

此隔离级别解决了的问题

- 丢失更新
- 未落实的读
- 不可重复的读
- 幻象

13.4.5 隔离级别对比

图 13.8 比较了对于一次访存，不同隔离级别的作用。如图所示，未落实的读（UR）级别没有锁定。游标稳定性（CS）级别在访问行 1 时将它锁定，但是在访问行 2 时即将此锁定释放，等等。对于读稳定性（RS）级别或是可重复读（RR）级别，任何被访问的行都将被锁定，并且直到事务结束（一个提交点）时才会被释放。

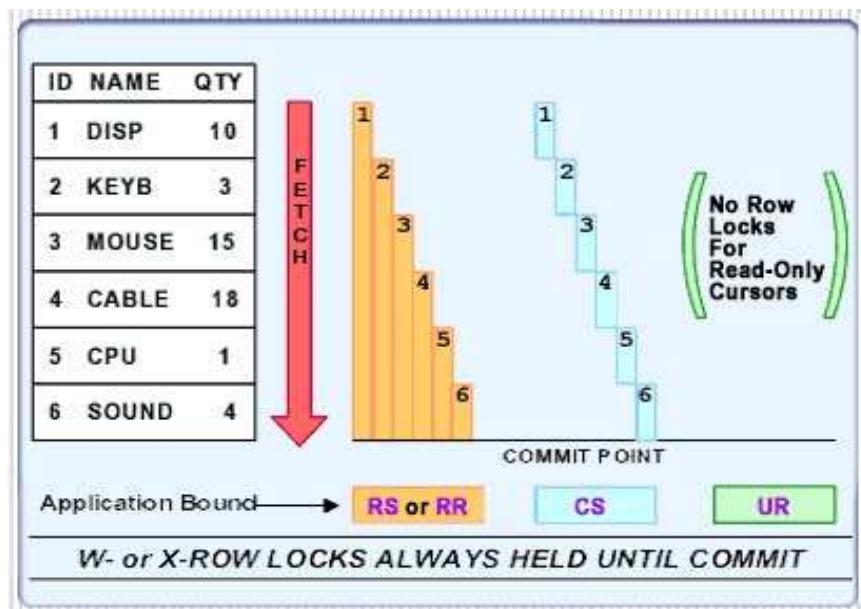


图 13.8 – 隔离级别对比

13.4.6 设定隔离级别

隔离级别可以作用于不同的级别：

- Session (application)会话级（程序级）
- Connection 连接级
- Statement 语句级

隔离级别通常被定义在会话或程序级别。在您的程序中如果没有定义隔离级别，它的默认值为游标稳定性等级。例如，表 13.1 展示了对于.NET 和 JDBC 程序而言可能的隔离级别以及这些设定属性如何匹配 DB2 的隔离级别。

DB2	.NET	JDBC
Uncommitted Read 未落实的读(UR)	ReadUncommitted	TRANSACTION_READ_UNCOMMITTED
Cursor Stability 游标稳定性(CS)	ReadCommitted	TRANSACTION_READ_COMMITTED
Read Stability 读稳定性(RS)	RepeatableRead	TRANSACTION_REPEATABLE_READ
Repeatable Read 可重复读(RR)	Serializable	TRANSACTION_SERIALIZABLE

表达 13.1 – 隔离级别术语对比

语句级隔离级别可以通过 WITH{隔离级别}子句设定。例如：

```
SELECT ... WITH {UR | CS | RS | RR}
```

例如有如下的情形：

一个程序需要粗略地知道表中的行数。性能最为重要。游标稳定性级别要求一条 SQL 异常语句的配合。

```
SELECT COUNT(*) FROM tab1 WITH UR
```

对于嵌入式的 SQL，该级别是在绑定时设定的，对于动态 SQL，该级别是在运行时设定的。

隔离级别的选择取决于您的程序。如果您的程序像上面的例子那样不需要确切的数字，就选用未落实的读 (UR) 级别。如果您的程序要求严格控制它作用的数据，选择可重复读 (RR) 级别。



在绑定或准备时要使用当前提交的语义，使用以下句法：

BIND:

```
>---+-----+---+
'--CONCURRENTACCESSRESOLUTION--+--USE CURRENTLY COMMITTED--+--'
                                '--WAIT FOR OUTCOME-----'
```

PREPARE:

```
concurrent-access-resolution:
| +-USE CURRENTLY COMMITTED+-+-----+-----|
    '-WAIT FOR OUTCOME-----'
```

在 JDBC 应用程序中使用 IBM 数据服务器驱动程序来服务 JDBC 和 SQLJ，您可以使用属性 concurrentAccessResolution 来激活当前提交。

13.5 锁定升级

DB2 做的每次锁定都要消耗一定量的存储空间。当优化器认为一次锁定整个表格比锁定多个行更好时，就出现了锁定升级。

图 13.9 描述了这样情况。

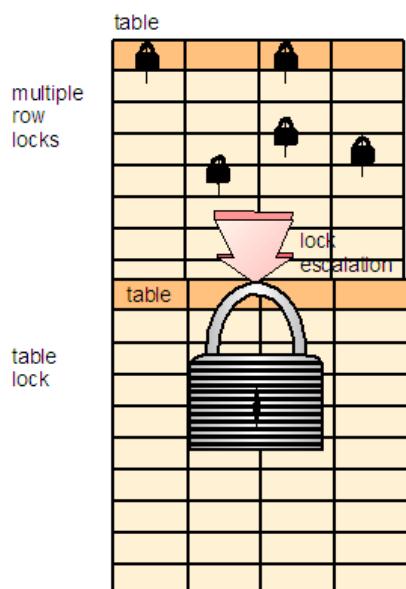


图 13.9 – 锁定升级

有两个主要的数据库配置参数与锁定升级有关：

LOCKLIST – 预留存储空间的数量（以 4k 页为单位），该空间用于管理所有连接的进程的锁定。Windows 中的默认值 50，即 50 个 4K，总共有 $50 \times 4K = 200K$ 个页。

MAXLOCKS – 单个程序在整个锁定列表（lock list）中可以占用的最大比例。默认值为 22%。

因此，如果使用默认值，当单个程序请求多于 $44K(200 K * 22\% = 44K)$ 的空间用于保存锁定，就会发生锁定升级。如果在默认配置下经常发生锁定升级，请增大 LOCKLIST 和 MAXLOCKS 的值。

锁定升级会减少并行，因而导致性能的下降。DB2 诊断记录文件（db2diag.log，通常保存在 C:\Program Files\IBM\SQLLIB\DB2 文件夹中）能够用于监测锁定升级的发生情况。

13.6 锁定监视

您可以使用 DB2 程序锁定快照来监视锁定的使用情况。使用如下命令来为锁定开启快照：

```
UPDATE MONITOR SWITCHES USING LOCK ON
```

在这个开关被打开后，监视信息会被收集起来。使用如下命令来获得给定时间的锁定情况报告：

```
GET SNAPSHOT FOR LOCKS FOR APPLICATION AGENTID <handle>
```

图 13.9 展示了一个程序的锁定快照输出的内容。

Application Lock Snapshot	
Snapshot timestamp	= 11-05-2002 00:09:08.672586
Application handle	= 9
Application ID	= *LOCAL.DB2.00B9C5050843
Sequence number	= 0001
Application name	= db2bp.exe
Authorization ID	= ADMINISTRATOR
Application status	= UOW Waiting
Status change time	= Not Collected
Application code page	= 1252
Locks held	= 4
Total wait time (ms)	= 0
 List Of Locks	
Lock Name	= 0x050007000480010000000000052
Lock Attributes	= 0x00000000
Release Flags	= 0x40000000
Lock Count	= 255
Hold Count	= 0
Lock Object Name	= 98308
Object Type	= Row
Tablespace Name	= TEST4K
Table Schema	= ADMINISTRATOR
Table Name	= T2
Mode	= X

图 13.9 – 程序的锁定情况快照

new in
V9.7

注意:

在 DB29.7 中，正努力把数据库管理从系统管理中移除，快照技术能让 SQL 访问内存，如工作负载管理表函数和 IBM 数据工作室工具。更多信息见 DB2 官方文档。

13.7 锁定等待

当两个或两个以上程序需要对同一个对象进行操作，它们中的一个可能不得不等待以获取它所需要的锁定。默认情况下，程序会无限期地等待。一个程序等待锁定的时间是由数据库配置参数 **LOCKTIMEOUT** 来决定的。默认值是-1（无限期地等待）。

CURRENT LOCK TIMEOUT 寄存器可以用于设置给定连接的锁定等待时间。这个寄存器默认情况下会被设为 **LOCKTIMEOUT** 的值。它的值用 **SET LOCK TIMEOUTWG** 语句来修改。该寄存器的值一旦被设定到一个连接，它会一直持续于整个事务，如：

```
SET LOCK TIMEOUT=WAIT n
```

13.8 死锁的引发与侦测

当连接到同一个数据库的两个以上的进程无限期地等待某一资源时，便可能发生死锁。因为每个进程都保持着另一个进程需要的资源，所以这种等待永远都无法解除。死锁是大多数程序设计中都要解决的问题。图 13.10 描述了一个死锁。

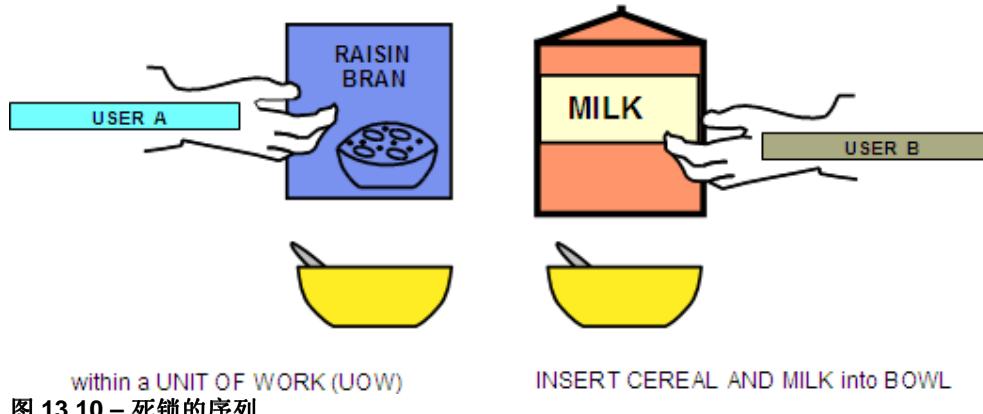


图 13.10 – 死锁的序列

在图 13.10 中，用户 A 拿着葡萄干麦片，直到他拿到牛奶才肯释放。另一方面，用户 B 拿着牛奶，直到他拿到葡萄干麦片才肯释放。于是，我们就有死锁的情况。

**new in
V9.7**

在 DB2 9.7 中，使用当前提交大大减少了死锁的发生，因为一个应用程序不需要等待其他应用程序释放它的锁，而可以直接访问当前提交值。

为了模拟在 DB2 死锁的情况，按照下列步骤进行：

1. 关闭当前提交：

```
db2 update db cfg for sample using cur_commit off
db2stop force
db2start
```

2. 打开两个 DB2 命令窗口（我们将称之为“CLP1”和“CLP2”。）代表两个不同的连接到数据库的应用从 CLP1 发出命令：

```
db2 connect to sample
db2 +c update employee set firstnme = 'Mary' where empno = '000050'
```

3. 在 CLP1 中执行以下命令：

```
db2 connect to sample
db2 +c update employee set firstnme = 'Mary' where empno = '000050'
```

首先我们连接到了数据库 SAMPLE，在 employee 表对所有“empno = '000050'”的行执行了一条更新（update）语句。语句中的“+c”选项表示我们不希望 DB2 命令窗口自动提交语句。我们这样做的目的是为了保持锁定。

4. 在 CLP2 中执行如下命令：

```
db2 connect to sample
db2 +c update employee set firstnme = 'Tom' where empno = '000030'
```

在表示第二个程序的 CLP2 窗口中，我们也连接到 SAMPLE 数据库，但是要更新 employee 表中的另一行。

5. 在 CLP1 中执行:

```
db2 +c select firstnme from employee where empno = '000030'
```

在键入回车键执行上述 SELECT 语句后, SELECT 仿佛被挂起了。它实际上不是挂起, 而是在等待 CLP2 在步骤 4 中对这一行锁定的释放。在此处, 如果 LOCKTIMEOUT 被保留为默认值-1, CLP1 进程就会一直等待下去。

6. 在 CLP2 中执行:

```
db2 +c select firstnme from employee where empno = '000050'
```

通过执行上述 SELECT 语句, 我们现在制造出了一个死锁。这个 SELECT 语句也会仿佛被挂起了, 而事实上它是在等待 CLP1 在步骤 2 中对该行锁定的释放。

在上面死锁序列中, DB2 将会核对数据库的配置参数 DLCHKTIME。这个参数设置的是检查死锁等待的间隔时间。例如, 如果这个参数被设为 10 秒, DB2 将每 10 秒检查一次是否存在死锁。如果确实发生了死锁, DB2 会用一个内部算法来决定两个事务中的哪一个应该被回滚, 哪一个应该继续。

如果您正在体验数目众多的死锁, 您应该重新检查现存的事务并且检查是否可能重新进行构造。

13.9 并行与锁定的最佳实践:

遵循以下建议有助于最佳地实现可能的并行性:

1. 如果您的应用程序的逻辑允许, 使用当前提交 (CC)。
2. 使事务尽可能地短小。这可以通过在您的程序逻辑允许的情况下频繁地使用 COMMIT 语句 (即使对只读的事务也一样) 来实现。
3. 只有必要时才记录事务信息。
4. 快速清理数据:

```
ALTER TABLE ACTIVATE NOT LOGGED INITIALLY WITH EMPTY TABLE
```

**new in
V9.7**

或使用 DB2 9.7 的 TRUNCATE 命令:

```
TRUNCATE <table name>
```

5. 成批或成组地修改数据, 例如:

```
DELETE FROM (
    SELECT * FROM tedwas.t1 WHERE c1 = ... FETCH FIRST 3000 ROWS ONLY)
```

6. 利用 DB2 的数据转移工具中的并行特性。

7. 设置数据库级的 LOCKTIMEOUT 参数 (推荐值为 30~120 秒之间)。不要保留为默认值-1。您也可以使用基于会话的超时锁定。

8. 不要检索不需要的数据。例如, 在 SELECT 语句中使用 FETCH FIRST n ROWS ONLY 子句。

注意：

并行与锁定的最佳实践的更多信息，请见最佳实践的文档：
<http://www.ibm.com/developerworks/data/bestpractices/>

13.10 小结

这一章我们关注了如何通过事务控制、用户并发控制和锁定级别来维护数据完整性。不同的并行级别会影响对数据的访问和管理。

我们也详细关注了如何设定隔离级别来解决这些问题和如何操纵隔离级别来提供您的应用程序和数据所需的最佳灵活性。

除此之外，我们还关注了锁定升级、锁定等待和锁定监视，以及数据库死锁的引发、检测和解决方法。

最后，我们讨论了一些针对您的并行需求，获得尽可能好的结果的一些最佳实践思路。

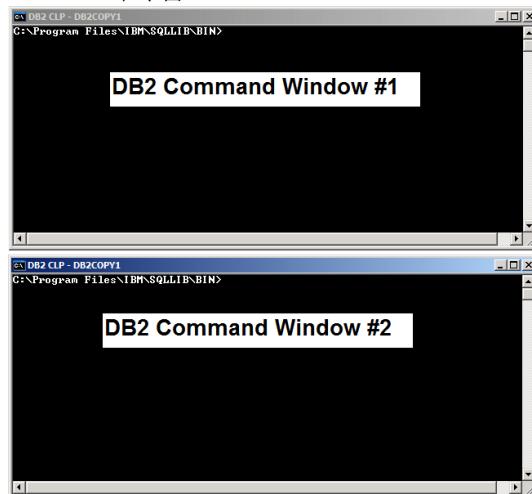
13.11 实验

在本实验中您将要使用 DB2 命令窗口实践本章所讨论的并发和锁定的内容。这个工具使用的是默认的 CS 隔离级别。执行完一条 SQL 语句后，命令窗口将自动做出提交（即 `autocommit` 自动提交）。为了说明问题，我们将使用 `+c` 来关闭自动提交，并在一些 SQL 语句后使用 `WITH <isolation level>` 子句来重载默认隔离级别 CS。

第一部分：测试幻象问题和隔离级别 RR

实验过程：

- 按照下图所示打开2个DB2命令窗口。我们把上面的窗口叫做“DB2 命令窗口 #1”，下面的叫做“DB2命令窗口 #2”



- 在DB2 命令窗口 #1 中输入：

```
db2 connect to sample
db2 +c select * from staff
这里将返回35个记录
```

- 在DB2命令窗口 #2 输入：

```
db2 connect to sample
```

```
db2 +c insert into staff (id, name) values (400, 'test')
```

4. 在 DB2 命令窗口 #1 输入:

```
db2 +c select * from staff
```

仍然返回35个记录

5. 在 DB2 命令窗口 #2 输入:

```
db2 commit
```

6. 在 DB2 命令窗口 #1 输入:

```
db2 +c select * from staff
```

结果将返回36个记录!

DB2 命令窗口#1代表一个打开一个游标或结果集 (select * from staff) 并获得35个记录的应用程序。在同一个事务中（因为在这个窗口中我们没有输入任何提交语句），这个应用程序打开相同的游标，就算在DB2命令窗口#2中的应用程序插入了（但并没有提交）一个新的记录，您仍然只能看到35个记录。

接下来，DB2命令窗口#2应用程序提交这个插入，所以DB2 命令窗口#1第三次打开这个游标，结果集将返回多一行（幻象）并获得36个记录。这个例子说明了幻象问题。在同一个事务中，打开相同的游标将返回更多的行。这里我们使用的是CS隔离级别，正如本章之前所提，CS并不能阻止幻象。

7. 在我们进行接下来的步骤之前，先清除插入的记录。

在DB2 命令窗口#1中输入:

```
db2 rollback
```

在 DB2 命令窗口#2 输入:

```
db2 delete from staff where id = 400
```

```
db2 select * from staff
```

还是返回35个记录

8. 现在我们来看看RR隔离级别是否能阻止幻象问题的产生。

在 DB2 命令窗口 #1 输入:

```
db2 connect to sample
```

```
db2 +c select * from staff with RR
```

返回35个记录

在 DB2 命令窗口 #2 输入:

```
db2 connect to sample
```

```
db2 +c insert into staff (id, name) values (400, 'test')
```

这条语句将会如预期一样挂起。

因为子句WITH RR被添加到DB2命令窗口#1的SELECT语句中，此隔离级别将会阻止会改变结果集的行所进行的插入。这个例子说明了RR隔离级别确实能阻止幻象问题。

9. 继续进行此实验的第二部分前请先清除记录。

在 DB2 命令窗口 #2 输入:

```
Ctrl-C (to interrupt)
```

关闭窗口

在 DB2 命令窗口 #1输入:

```
db2 rollback
```

关闭窗口

第二部分：测试当前已落实 (CC) 级别和未落实的读 (UR) 级别

步骤1：分析没有当前提交 (currently committed) 的CS隔离级别的作用

1. 打开一个DB2命令窗口并输入以下语句：

```
db2 connect to sample
db2 select * from staff
```

查看表STAFF的内容，特别是要找到值为'10'的ID。相应的NAME列中有个值为Sanders。然后关闭这个窗口。

2. 当前提交 (Currently committed) 是新数据库的默认级别，确认它为SAMPLE数据库设置好了：

```
db2 get db cfg for sample
```

在输出内容的末尾有如下一行：

Currently Committed	(CUR_COMMIT) = ON
---------------------	-------------------

如果当前提交CC的值是ON，先设置其为OFF，然后我们就可以分析隔离级别CS的作用，即使是DB2 9.7之前的版本：

```
db2 update db cfg for sample using CUR_COMMIT off
db2 force applications all
```

添加 force选项确保没有连接，这样CUR_COMMIT的更新将会在下一个连接中生效。

确定CUR_COMMIT 已被禁止，应该是：

Currently Committed	(CUR_COMMIT) = DISABLED
---------------------	-------------------------

3. 像第一部分一样打开2个DB2命令窗口，这样一个窗口在上，一个在下。我们来看看当一个更新（写）和一个选择（读）对同一行进行操作时，隔离级别CS在关闭当前提交下如何工作。

注意我们不必在语句后面添加“WITH CS”（因为是默认的）。

在DB2命令窗口#1中输入（这也就是写）：

```
db2 connect to sample
db2 +c update staff set name = 'Chong' where id = 10
```

在DB2命令窗口#2中输入（这也就是读）：

```
db2 connect to sample
db2 +c select * from staff
```

这个SELECT将挂起，等待DB2命令窗口#1应用程序释放专属的(X) 锁定。正如您所见，DB2 9.7 之前的CS默认作用允许更少的并发。

在DB2命令窗口#2中输入：

CTRL-C (press these keys to interrupt)
关闭窗口

在DB2命令窗口#1中输入：
db2 rollback
关闭窗口

步骤2：分析有当前提交（currently committed）的隔离级别CS的作用

把当前提交设置为ON：
打开一个DB2命令窗口并输入：

```
db2 update db cfg for sample using CUR_COMMIT on  
db2 force applications all
```

关闭窗口。

如步骤1所示打开2个DB2命令窗口，这样一个在上一个在下。然后输入如下内容：

在DB2命令窗口#1中输入：
db2 connect to sample
db2 +c update staff set name = 'Chong' where id = 10

在DB2命令窗口#2中输入：
db2 connect to sample
db2 +c select * from staff

现在这条SELECT语句就能运行，没有被挂起，显示的值为Sanders，这正是当前提交的值。

在DB2命令窗口#1中输入：
db2 rollback
关闭窗口。

在DB2命令窗口#2中输入：
db2 rollback
关闭窗口。

步骤3：分析隔离级别UR的作用

1. 如第一部分一样打开2个DB2命令窗口，这样一个在上面一个在下面。然后输入以下内容：

在DB2命令窗口#1中输入：
db2 connect to sample
db2 +c update staff set name = 'Chong' where id = 10

在DB2命令窗口#2中输入：
db2 connect to sample
db2 +c select * from staff with UR

这SELECT语句运行，但您会注意到显示的值是Chong，这是未提交的值。

在DB2命令窗口#1中输入：

db2 rollback

关闭窗口。

在DB2命令窗口#2中输入：

db2 rollback

关闭窗口。

PART III – 学习 DB2: 应用程序开发

14

第 14 章 –DB2 应用程序开发介绍

IBM DB2 是用来管理相关数据和 XML 数据的强大的数据服务器软件。它对数据库管理员和数据库开发者都提供了灵活性。不管您用哪一种语言来开发您的程序，DB2 都提供您所需要的驱动程序，适配器和扩展作为您的数据库应用程序的一部分。您在开发应用程序时可以免费的、没有数据库大小限制使用 DB2 Express-C，并且相同水平的编程语言也支持 DB2 的其他版本。开发使用一次 DB2 Express-C，您可以在任意 DB2 版本上运行您的应用程序并且不用任何修改。

注意：

这部分只是介绍 DB2 应用程序的开发。超过 25 本的免费网上图书系列正在编写，作为 DB2 校园系列图书的一部分，其中有一本书是针对 DB2 的应用程序开发的。这个系列的其他书还包括非 DB2 的课题，如 Java, PHP, Ruby on Rails, Python, Perl, Web 2.0, SOA, Eclipse, 开放源码，云计算等！还有一些书涵盖了 IBM 技术，如详细探讨 pureQuery、Data Studio，和 InfoSphere Data Architect 的书。这些书于 2009 年 10 月开始推出。

14.1 DB2 应用程序的开发：概述

DB2 为数据库开发者提供了灵活性，以便于利用服务器端的开发功能优势，例如存储过程和用户自定义函数。应用程序开发者可以使用他们自己选择的编程语言来开发客户端应用程序。关于这种灵活性图 14.1 所示。

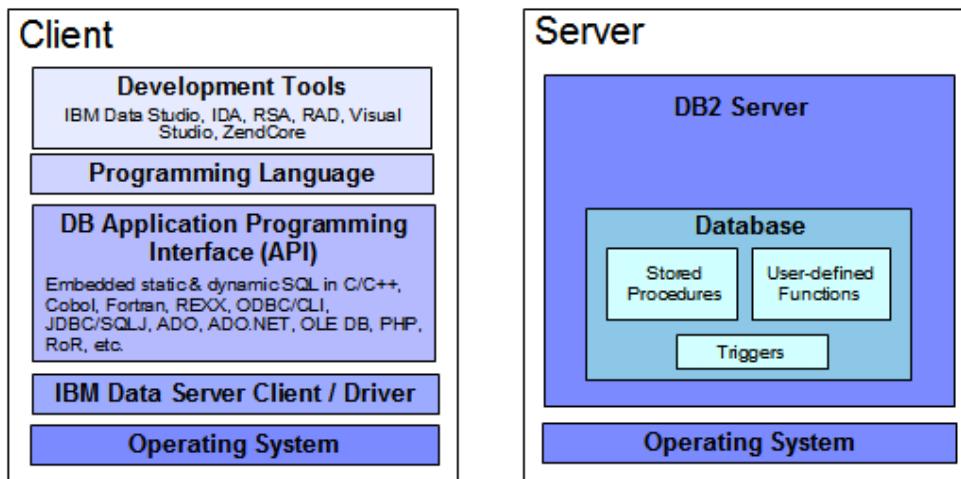


图 14.1 - DB2 对数据库和应用程序开发者都有用

在图 14.1 中，左图表示供应用程序员开发并运行所编写程序的客户端机器，在这个客户端机器中，除了操作系统之外，还可以根据开发应用程序的类型来安装 IBM Data Server Client。IBM Data Server Client 包括所需的连接驱动程序，如 JDBC 驱动程序和 ODBC / CLI 驱动程序。您可以

在访问 IBM DB2 Express-C 网站 ibm.com/db2/express 时下载这些驱动程序。无论用什么编程语言您都可以使用这些编程工具如 IBM Data Studio, InfoSphere Data Architect (IDA), Rational Software Architect (RSA), Rational Application Developer (RAD)等开发您的应用程序。IBM Data Server Client 同样支持着应用程序编程接口 (API) 库中支持的语言。当您连接到一个 DB2 服务器，使用这些 API 接口，DB2 会把所有程序的指令翻译成能够被 SQL 或 XQuery 的理解的语句。表 14.1 为所提到的工具做了一个简短的描述。

工具名称	描述
IBM Data Studio	IBM Data Studio 是一个基于 Eclipse 的工具，它允许用户管理他们的数据服务器、开发存储过程、开发函数和数据网络服务。本书前面已经介绍过 IBM Data Studio。
InfoSphere Data Architect (IDA)	IDA 是一个数据建模工具。它可以帮助您建立数据库的逻辑设计和物理设计。
Rational Software Architect (RSA)	RSA 是一个可以帮助您开发 UML 图形的基于 Eclipse 的软件开发工具。
Rational Application Developer (RAD)	RAD 是一个给软件开发者的基于 Eclipse 的快速应用程序开发工具。
Visual Studio	Microsoft Visual Studio 是一个集成开发环境，允许用户在 Windows 系统中使用微软的技术来开发应用程序。
ZendCore	开发 PHP 应用程序的免费集成开发环境，正式名称叫做 IBM 的 ZendCore。

表 14.1 - DB2 中开发应用程序的工具

图 14.1 右侧说明一个 DB2 服务器包含一个数据库。这个数据库中，有存储过程、用户自定义函数和触发器。接下来，我们对这些对象进行更加详细的描述。

14.2 服务器端的开发

DB2 中服务器端的开发就是用 DB2 的数据库来开发应用程序对象，并将应用程序对象存储在数据库中。下列应用对象我们将简单的讨论一下：

- 存储过程
- 用户自定义函数
- 触发器

14.2.1 存储过程

存储过程是一个数据库的应用程序对象，它可以封装 SQL 语句和业务逻辑。为减少应用程序与数据库之间网络流量，通过保持一部分数据库中的应用逻辑来提供性能改进。此外，存储过程提供一

个集中的位置来存储您的代码，所以其他应用程序可以重复使用相同的存储过程。使用 **CALL** 语句能够调用存储过程。在 DB2 中，您可以使用 SQL PL, Java, C/C++, CLR, OLE，以及 COBOL 这几种语言来开发存储过程。下面举一个简单的例子来介绍一下如何使用 DB2 命令在 Windows 或 Linux 界面中创建和调用一个 SQL PL 的存储过程：

```
db2 create procedure P1 begin end
db2 call P1
```

在这个例子中，P1 是一个没有任何功能的空存储过程。这个例子说明了如何简单的创建一个存储过程。如果要开发逻辑更复杂的存储过程，我们建议您使用含有调试器的 IBM Data Studio（数据工作室）。

14.2.2 用户自定义函数

用户自定义函数 (UDF) 是一个数据库应用程序对象，允许用户用自己的逻辑扩展 SQL 语言。一个函数总是返回一个或多个值，通常作为业务逻辑层的结果包含在函数中。使用一个 SQL 语句或 **values** 函数就能够调用函数。在 DB2 中，您可以使用 SQL PL, Java, C/C++, OLE, CLR 这几种语言来开发用户自定义函数。下面举一个简单的例子来介绍一下如何使用 DB2 命令在 Windows 或 Linux 界面中创建和调用一个 SQL PL 的用户自定义函数：

```
db2 create function F1() returns integer begin return 1000; end
db2 values F1
```

在这个例子中，函数 F1 是一个返回整型值 1000 的函数。**values** 语句可以用来调用该函数。和开发存储过程一样，我们建议您使用 IBM Data Studio 创建函数。

14.2.3 触发器

触发器 是一个自动执行对表或视图的操作的对象。在一个触发器定义的对象上执行触发动作会导致触发器被触发。触发器通常不被视为一个应用程序对象，因此，数据库开发者也就不会触发代码，而数据库管理员可以触发。由于有编程需要，我们会在本节中介绍触发器。下面是一个触发器的例子：

```
create trigger myvalidate no cascade before insert on T1
referencing NEW as N
for each row
begin atomic
    set (N.myxmlcol) = XMLVALIDATE(N.myxmlcol
        according to xmlschema id myxmlschema);
end
```

在这个例子中，在表 T1 上的 INSERT 操作之前触发器已经被触发。触发器将插入这个值（这是一个 XML 文件），但也会调用 XMLVALIDATE 函数来利用给定的模式验证这个 XML 文件。在第 15 章 DB2 pureXML 中，会谈到更多有关 XML 和 XML 模式的问题。

14.3 客户端的开发

正如题目中写的一样，在客户端的开发中，应用程序的开发者在客户端上编写自己的程序，然后使用 DB2 提供的应用程序接口 (API) 来连接和访问 DB2 数据库。在本节中我们将讨论：

- 嵌入式 SQL
- 静态 SQL 与 动态 SQL
- CLI 与 ODBC

- JDBC, SQLJ 与 pureQuery
- OLE DB
- ADO.NET
- PHP
- Ruby on Rails
- Perl
- Python

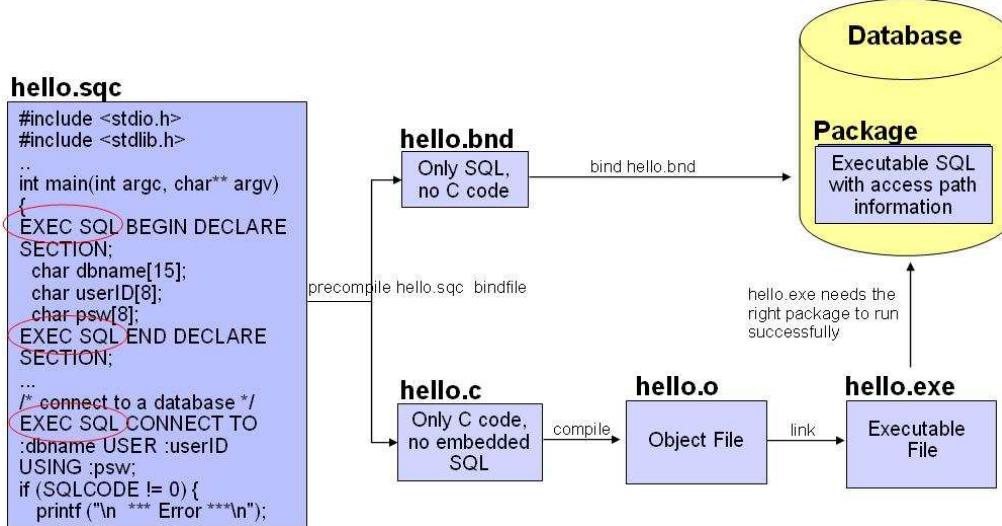
14.3.1 嵌入式 SQL

嵌入式 SQL 应用程序就是将 SQL 嵌入到宿主语言（如 C, C++, COBOL）中的应用程序。一个嵌入式 SQL 应用程序可以包括静态或动态的 SQL，这也正是我们下一节将会描述的。图14.2展示了嵌入式 SQL 应用程序是如何建立的。

图14.2 - 嵌入式 SQL 应用程序的建立

在图中，C 语言程序 `hello.sqc` 包含嵌入式 SQL。C 语言的嵌入式 SQL 应用程序接口（API）使用 EXEC SQL（图14.2中红圈画出）让一个预编译程序来区分嵌入式 SQL 语句和实际的 C 语言代码。您可能还注意到在 `hello.sqc` 代码中，一些变量前面加上了一个冒号，如`:dbname`, `:userID`, 和`:psw`。这些被称为宿主变量。宿主变量是宿主语言在嵌入式 SQL 语句中引用的变量。用 `precompile` 命令（也称为 `prep` 命令）时使用 `bindfile` 选项会生成两个文件，`hello.bnd` 的绑定文件只包含 SQL 语句，`hello.c` 的绑定文件也只包含 C 代码。使用 `bind` 命令就可以编译绑定文件来获得存储在数据库中的数据包。一个数据包包括编译后的/可执行 SQL 和 DB2 检索数据所遵循的存取路径。当发出 `bind` 命令时必须连接数据库。在该图的底部，`hello.c` 文件和正常的 C 语言程序一样的被编译和链接。生成的可执行文件 `hello.exe` 必须匹配数据库中存储的数据包才能成功地执行。

14.3.2 静态 SQL 与动态 SQL



静态 SQL 语句所在的 SQL 结构在预编译时是完全已知的。举个例子：

```
SELECT lastname, salary FROM employee
```

在这个例子中，在语句中引用的列 (`lastname`, `salary`) 和表(`employee`)的名称在预编译时是完全已知的。下面的这个例子也是一个静态 SQL 语句：

```
SELECT lastname, salary FROM employee WHERE firstname = :fname
```

在第二个例子中，宿主变量 :**fname** 是嵌套 SQL 语句的一部分。虽然宿主变量的值在运行前是未知的，但是其数据类型看程序就可以得知，并且所有其他对象（列名，表名）也完全是事先已知的。DB2 使用这些宿主变量的估计值来提前计算存取方案，因此，这种情况下语句仍然被认为是静态 SQL。

您预编译，绑定和静态编译可执行的 SQL 语句，然后才能运行您的应用程序。静态 SQL 最好用在数据没有很大变化的数据库。现在我们再看一个例子：

```
SELECT ?, ? FROM ?
```

在这个例子中，语句引用的列名和表名在运行前是未知的。因此，存取方案只在运行时利用当时的数据计算。这些类型的语句就是**动态 SQL** 语句。

不管 SQL 语句是否包括已知对象，一些程序的 API（如 JDBC 和 ODBC）总是使用动态 SQL。举个例子，语句

```
SELECT lastname, salary FROM employee
```

提前拥有所有已知的列名和表名，但通过 JDBC 或 ODBC，您不能预编译语句。所有存取方案中的语句还是在运行时计算的。

一般来说，用以下两种命令执行的语句会被视为动态的 SQL 语句：

PREPARE: 本命令通过准备或编译的 SQL 语句计算出的存取方案来检索数据

EXECUTE: 本命令执行 SQL

另外，您也可以用下面这个单独的语句来执行 **PREPARE** 和 **EXECUTE**:

EXECUTE IMMEDIATE

列表 14.1 显示了一个准备和执行嵌入式 C 语言动态 SQL 语句的例子。

```
strcpy(hVStmtDyn, "SELECT name FROM emp WHERE dept = ?");
PREPARE StmtDyn FROM :hVStmtDyn;
EXECUTE StmtDyn USING 1;
EXECUTE StmtDyn USING 2;
```

列表 14.1 - 嵌入式 C 语言 SQL 语句中使用 PREPARE 和 EXECUTE

列表 14.2 显示了与列表 14.1 相同的例子，但使用的是 **EXECUTE IMMEDIATE** 语句。

```
EXECUTE IMMEDIATE SELECT name from EMP where dept = 1
```

```
EXECUTE IMMEDIATE SELECT name from EMP where dept = 2
```

列表 14.2 - 嵌入式 C 语言 SQL 语句中使用 EXECUTE IMMEDIATE

在许多动态编程语言中（如 PHP 或 Ruby on Rails），其中 SQL 动态运行，程序员往往用不同的字段值来写如下所示的相同的 SQL 语句：

```
SELECT lastname, salary FROM employee where firstnme = 'Raul'
SELECT lastname, salary FROM employee where firstnme = 'Jin'
```

```
...
```

在这个例子中，这些语句除了 **firstnme** 列的值以外都是相同的。在 DB2 中认为这两个动态 SQL 语句是不同的，因此在运行时，DB2 编制并执行每个单独的语句。编制几次相同语句的开销可能会导致性能下降，因此 DB2 9.7 以前的版本，建议把语句编写成如下形式：

```
SELECT lastname, salary FROM employee where firstnme = ?
```

语句中的问号（?）称为参数标记。使用参数标记时，程序只需要编制语句一次，然后将不同的值提供给参数标记执行 **EXECUTE** 语句。

**new in
V9.7**

在 DB2 9.7 中，DB2 引入了一种叫语句集中器（**statement concentrator**）的技术，所有除了字段值以外相同的语句，会利用参数标记自动集中到一个单独语句中，然后用不同的值执行 **EXECUTE** 语句。语句集中器可以合理地确定何时不将语句整合在一起；举个例子，当您故意加入一些子句来影响 DB2 的优化器时，语句就不能被集中。

关于性能方面，通常静态 SQL 比动态 SQL 能更好地执行，因为在静态 SQL 中存取方案是在预编译时而不是在运行时完成的。然而，在有许多插入（**INSERT**）和删除（**DELETE**）等动作的环境中，预编译时计算出的统计数据未必是最新的，因此，静态 SQL 的存取方案可能不是最优的。假定频繁执行 **RUNSTATS** 命令来收集当时的统计数据，在这种情况下，动态 SQL 可能是更好的选择。

注意：

许多用户认为嵌入式 SQL 只能是静态的。实际上，它可以既可以是静态的也可以是动态的。

14.3.3 CLI 与 ODBC

调用层接口 (CLI) 最初是由 X/Open 国际联盟有限公司以及 SQL 接入组群开发的。不管关系型数据库管理系统 (RDBMS) 供应商是谁，CLI 是一个用于开发可移植的 C/C++ 应用程序所需可调用 SQL 接口的通用规格。在 X/Open 公司调用层接口草案的基础上，微软公司开发了**开放数据库互连 (ODBC)**。之后，X/Open 公司的大多数调用级接口规范得到了 ISO 国际 CLI 标准的认证。如图 14.3 所示，DB2 的 CLI 的开发基于两点：ODBC 与 SQL/CLI 国际标准。

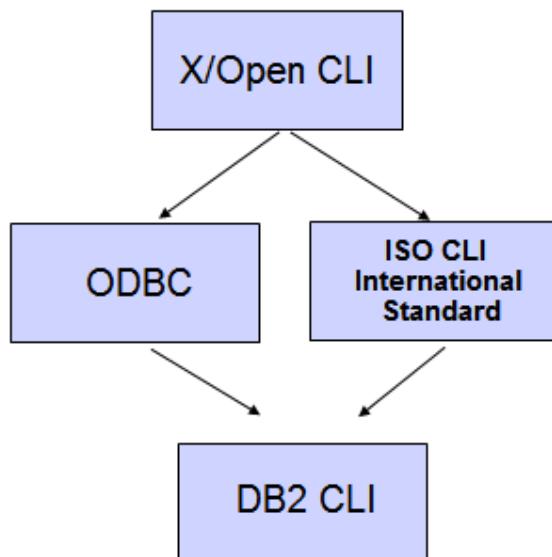


图 14.3 - DB2 的 CLI 的开发 ODBC 与 SQL/CLI 国际标准

DB2 中的 CLI 符合 ODBC 3.51 版本，当 ODBC 驱动程序管理器加载使用 CLI 时，CLI 还可作为 ODBC 的驱动程序。图 14.4 可以帮助您概览一下 DB2 CLI 如何支持 ODBC。

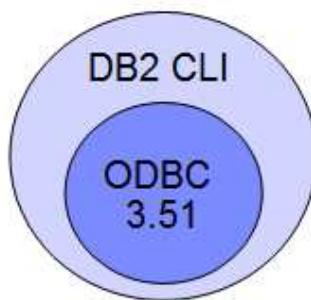


图14.4 - DB2的 CLI 符合 ODBC 3.51版本

CLI/ODBC 有以下特点:

- □代码可以轻松地在几个 RDBMS 供应商之间移植
- □与嵌入式 SQL 不同, 它不需要预编译器和宿主变量
- □运行动态 SQL
- □非常流行

运行一个 CLI/ ODBC 应用程序所需要的是 DB2 CLI 驱动程序。此驱动程序从下列客户端和驱动程序中安装, 这些客户端与驱动程序可从 www.ibm.com/db2/express 免费下载并使用:

- IBM Data Server Client
- IBM Data Server Runtime Client
- IBM Data Server Driver for ODBC and CLI

开发一个 CLI/ ODBC 应用程序所需要的不仅是 DB2 CLI 驱动程序, 还需要适当的库。这些只能在 IBM Data Server Client 中找到。

让我们来看下面的例子, 通过例子您能更好地了解如何为您的应用程序设置 DB2 CLI 驱动程序。图14.5描述了三个不同的计算机, 他们分别在印尼, 巴西以及加拿大。

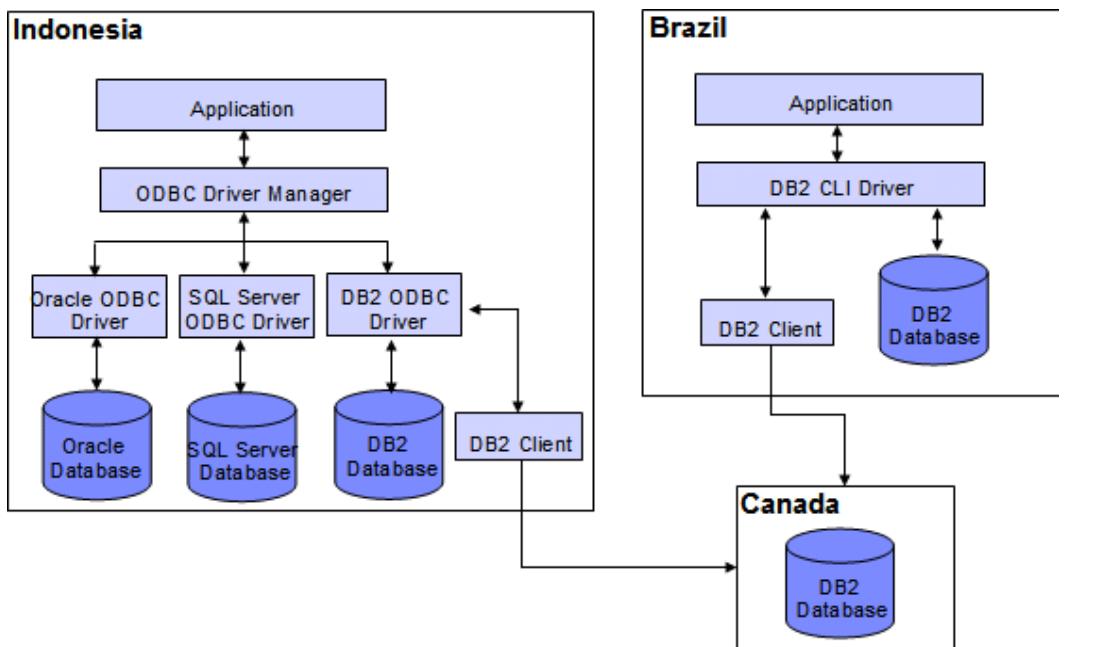


图14.5 - DB2 CLI/ODBC 的示例脚本

该图说明了两种情况：

在左侧，比方说在印尼的计算机上正在运行的 ODBC 应用程序可以在任意的 RDBMS 中工作，如 Oracle, Microsoft SQL Server 或 DB2。ODBC 驱动程序管理器将依据正在访问的数据库来装载适当的 ODBC 驱动程序。在应用程序要访问加拿大 DB2 数据库的情况下，连接要经过一个具有远程连接组件的 DB2 客户端。

在右侧，比方说在巴西的计算机上运行一个 CLI 应用程序。之所以说它是 CLI 应用程序，是因为它可能会使用到一些不可用在 ODBC 中的特定函数，也因为该应用程序将只能对一个 DB2 数据库工作。CLI 应用程序将通过 DB2 CLI 驱动程序运行。应用程序可以连接到巴西本地 DB2 数据库。当它需要连接到在加拿大的远程数据库时，就必定要经历 DB2 客户端。

最后在本节提出的最后一点就是之间的比较。图 14.6 说明了这一比较。

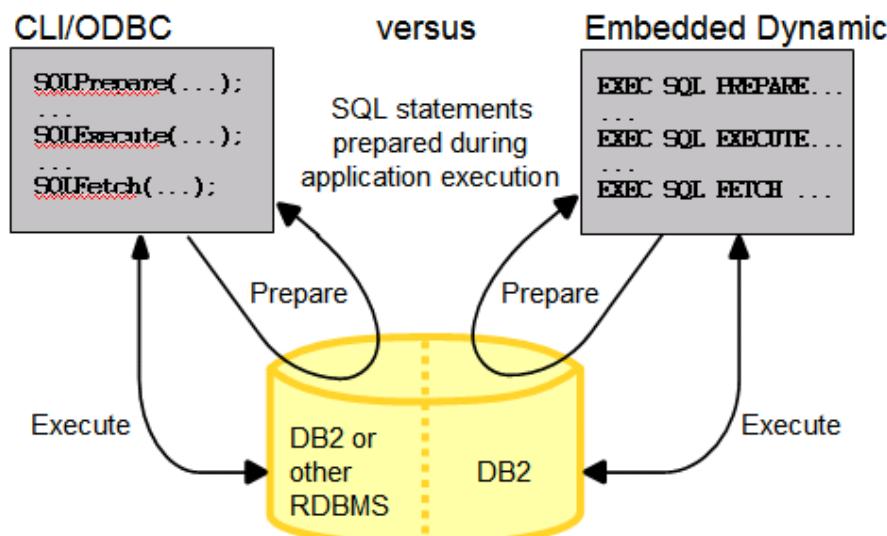


图 14.6 - CLI/ ODBC 应用程序与嵌入式 C 语言动态 SQL 应用程序

如图 14.6 所示，CLI/ ODBC 与嵌入式 C 语言动态 SQL 之间的唯一不同的就是，CLI/ ODBC 的代码是可移植的，可以通过改变连接字符串来轻易接入其他的 RDBMS，而在嵌入式 C 语言动态 SQL 中版本，您就要为 DB2 编写的特定内容。当然，其他不同之处就是调用 **PREPARE**，和 **EXECUTE** 函数的方法不同。

14.3.4 JDBC, SQLJ 与 pureQuery

Java数据库连接 (JDBC) 是一种 Java 编程 API，提供工作方法和访问数据库的标准。通过 JDBC，代码可以简单地在一些 RDBMS 之间移植。对代码唯一需要做变化就是加载不同的 JDBC 驱动程序和连接字符串。JDBC 只使用动态 SQL，这是很流行的。

SQLJ 是在 Java 程序中嵌入式 SQL 的标准。它主要用于静态 SQL，尽管它可以与 JDBC 相互操作（如图 14.7 所示）。虽然它通常比 JDBC 程序更简洁，并提供更好的性能，但是它并没有被广泛地接受。在编译 SQLJ 程序之前必须通过预处理器（SQLJ 转换器）才能运行程序。

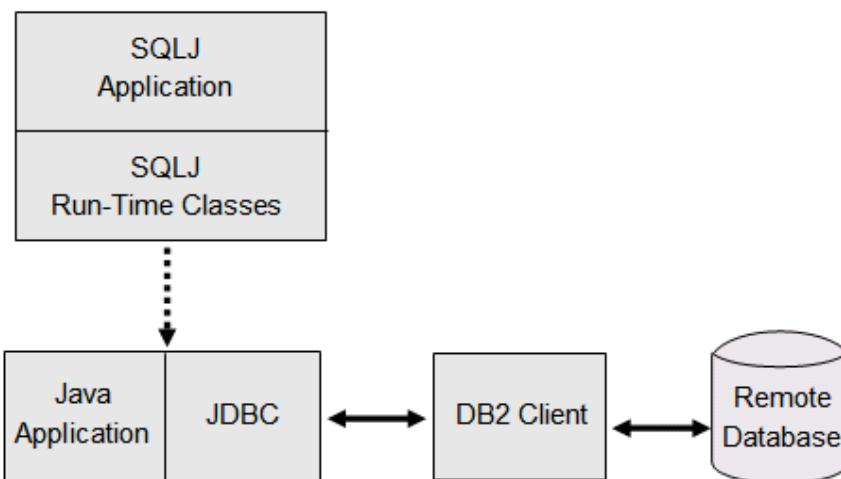


图 14.7 - SQLJ 与 JDBC 应用程序之间的关系

在图 14.7 中，是否需要一个 DB2 客户端取决于 JDBC 驱动程序类型，这点在本节后部分讨论。

pureQuery 是一款 IBM 基于 Eclipse 的插件，用于将关系型数据作为对象来处理。自 2007 年推出以来，pureQuery 可以自动生成代码来建立面向对象的代码与对象关系型数据库之间的对象关系映射（ORM）。您先用 Optim Development Studio（ODS）创建一个 Java 项目，连接到 DB2 数据库，然后 ODS 就会发现所有的数据库对象。通过 ODS 的 GUI，您可以选取一个表，然后选择生成 pureQuery 代码来将一个关系表中的任何基础实体转换成 Java 对象。生成的代码用来创建相应的 SQL 语句和封装这些语句的上层 Java 对象。所生成的 Java 对象和所包含的 SQL 语句可以被进一步定制。使用 pureQuery，您可以在运行时决定要运行的 SQL 是静态或动态模式。pureQuery 对 Java 和.NET 都支持。

14.3.4.1 JDBC 与 SQLJ 驱动程序

虽然 JDBC 驱动程序有几种类型，如类型 1, 2, 3 和类型 4；类型 1 和类型 3 不常用，DB2 对于这两种类型的支持也已经老旧（deprecated）。接下来我们将描述类型 2 的两个驱动程序，但其中的一个驱动程序之一也已经老旧。

DB2 支持类型 2 和类型 4，如表 14.2 所示。类型 2 的驱动程序需要有安装 DB2 客户端，因为驱动程序要使用它来建立与数据库的通信。类型 4 是一个纯 Java 客户端，所以不需要 DB2 客户端，但驱动程序必须安装在有 JDBC 应用程序运行的计算机上。

驱动程序类型	驱动程序名字	打包成	支持	Java 对 SDK 的最低水平要求
类型 2	Linux, UNIX 与 Windows（不建议）的 DB2 JDBC 类型 2 驱动程序	db2java.zip	JDBC 1.2 与 JDBC 2.0	1.4.2
类型 2 与类型 4	JDBC 与 SQLJ 的 IBM Data Server Driver	db2jcc.jar 与 sqlj.zip	JDBC 3.0 兼容	1.4.2
		db2jcc4.jar 与 sqlj4.zip	JDBC 4.0 以下版本	6

表 14.2 - DB2 中 JDBC 与 SQLJ 驱动程序

正如前面提到并显示在表 14.2 中的一样，类型 2 由两个不同的驱动程序所提供，但文件名为 db2java.zip 的 Linux, UNIX 和 Windows 的 DB2 JDBC 类型 2 驱动程序已经可以取消。

当您为 JDBC 和 SQLJ 安装 DB2 服务端，DB2 客户端或 IBM Data Server Driver 时与 JDBC 3.0 兼容的 db2jcc.jar 和 sqlj.zip 文件就会自动添加到您的 classpath。

14.3.5 OLE DB

数据库对象连接与嵌入（OLE DB） 是一组访问不同的数据源的接口。这个 OLE DB 设计是为了代替 ODBC，但扩大到支持更为广泛的数据源，其中包括非关系型数据库，例如面向对象的数据和电子数据表。OLE DB 通过使用组件对象模型（COM）技术实现。

OLE DB 用户可以利用 IBM OLE DB 的 DB2 供给程序（IBM OLE DB Provider for DB2）访问 DB2 数据库。这个供给程序有以下特点：

- 供给程序名称：IBMDADB2
- 支持 OLE DB 供给程序规范的 0 级，也支持 1 级里的一些接口
- 符合 Microsoft OLE DB 2.7 或更高版本的规范
- 必须安装 IBM Data Server Client 和 Microsoft Data Access Components(MDAC)

- 如果 IBMDADB2 没有明确指定，就利用默认的微软 OLE DB 驱动程序（MSDASQL）。MSDASQL 允许客户端利用 OLE DB 来使用 ODBC 驱动程序访问非微软 SQL 服务端数据源，但并不保证 OLE DB 驱动程序的全部功能都能实现。

14.3.6 ADO.NET

.NET Framework 是微软公司用来替代组件对象模型（COM）技术的。使用.NET Framework，您可以使用超过四十种不同的编程语言编写.NET 应用程序；最流行的是使用 C# 和 Visual Basic .NET 语言。

.NET Framework 类库为您提供建立.NET 应用程序的基石。该类库与语言无关，并提供操作系统和应用服务的接口。.NET 应用程序（不论语言）编译成中间语言（IL），也就是一种字节码类型。

公共语言运行时（CLR）是.NET Framework 的核心，联机编译 IL 代码，然后运行它。在运行已编译的 IL 代码时，CLR 将激活对象，验证其安全检查，分配其内存，并执行它们，每执行完毕一次就清理它们的内存。对于 Java 如何工作的问题，我们来打个比方，在 Java 中，一个程序可以在很少或没有修改的情况下，运行在多个平台：一种语言，但多个平台。在.NET 中，任意四十种支持的语言之一所编写的程序可以在很少或没有修改的情况下，运行在一个像 Windows 这样的平台上：多种语言，但一个平台。

ADO.NET 是.NET Framework 中用来支持数据存取的。ADO.NET 支持连接和未连接两种情况下的存取。在 ADO.NET 中未连接时数据存取的关键部分是 **DataSet** 类，它是作为数据库缓存驻留在您的应用程序内存中的实例。

对于连接和未连接两种情况下的存取，您的应用程序使用数据库通过了所谓的数据供给程序。各种数据库产品里都有自己的.NET 数据供给程序，Windows 的 DB2 当中也有。

每一个.NET 数据供给程序都会以实现下述基本类为特征：

- **Connection:** 建立和管理数据库连接。
- **Command:** 对数据库执行 SQL 语句。
- **DataReader:** 对数据库读取并返回结果数据集。
- **DataAdapter:** 将 DataSet 实例链接到数据库。通过 DataAdapter 的实例，DataSet 可以读取和写入数据库表中的数据。

能够使用 DB2 工作的三种数据供给程序如表 14.3 所示。

数据提供者	特点
ODBC .NET Data provider (不推荐)	<ul style="list-style-type: none"> - 使 ODBC 使用 DB2 CLI 驱动程序调用 DB2 数据源。 - 与 DB2 CLI 驱动程序支持和限制的关键词相同。 - 在.NET Framework1.1, 2.0 或 3.0 版本中可使用。
OLE DB .NET Data provider (不推荐)	<ul style="list-style-type: none"> - 使用 IBM DB2 OLE DB 驱动程序 (IBMDADB2) - 与 DB2 OLE DB 驱动程序支持和限制的关键词相同。 - 只能.NET Framework1.1, 2.0 或 3.0 版本中使用。
DB2 .NET Data provider (推荐)	<ul style="list-style-type: none"> - 扩展了 ADO.NET 接口的 DB2 支持。 - DB2 管理提供程序实现了标准 ADO.NET 类和方法的相同设置。 - 根据 IBM.DATA.DB2 来定义命名空间 - 能够通过下载以下任一程序获得: <ul style="list-style-type: none"> - ODBC, CLI 与 .NET 的 Data Server Driver - IBM Data Server Runtime Client - DB2 Data Server

表 14.3- ADO.NET 数据供给程序

14.3.7 PHP

PHP 超文本预处理器 (PHP) 是一个开源的，与平台无关的脚本语言，它是为 Web 应用程序开发而设计的。它可以嵌入在 HTML 中，一般运行在可以运行 PHP 代码并创建输出网页的 Web 服务器上。

PHP 是一种模块化的语言。您可以使用扩展来定制需要的功能。一些最流行的 PHP 扩展是用来访问数据库的。IBM 通过两个扩展来支持访问 DB2 数据库：

- **ibm_db2:** ibm_db2 扩展提供了一个应用程序接口，除了广泛的访问数据库元数据外，还能够进行创建，读取，更新和写入数据库操作。它可以在 PHP 4 或 PHP 5 上编译。
- **pdo_ibm:** pdo_ibm 是一个 PHP 数据对象 (PDO) 扩展，能够通过 PHP 5.1 中引入的标准面向对象的数据库接口来访问 DB2 数据库。它可以针对 DB2 库直接编译。

PHP 的扩展和驱动程序由 PECL 库免费提供 (<http://pecl.php.net/>)，或在 IBM 数据服务器客户端中提供。ibm_db2 和 pdo_ibm 都是基于 IBM 的 DB2 CLI 层的。

IBM 还与 Zend 科技公司合作支持 ZendCore。ZendCore 是用于 PHP 和 DB2 Express- C 开发的、免费的、已经完成环境配置的工具包。ZendCore 捆绑了 PHP 库，Apache Web 服务器和 DB2 Express - C。从 <http://www.ibm.com/software/data/info/zendcore> 可以下载 ZendCore。

14.3.8 Ruby on Rails

Ruby 是一个开源的面向对象的语言。Rails 是使用 Ruby 的 Web 框架而建立的。Ruby on Rails (RoR) 是一种开发数据库支持的基于 Web 的应用软件的理想手段。这个热门的新技术采用基于模型、视图和控制器 (MVC) 的体系结构并遵循敏捷软件开发原则。

Rails 并不需要特别的文件格式或集成开发环境 (IDE)，您可以从文本编辑器起步。但是许多集成开发环境已经提供了对 Rails 的支持，如 RadRails，就是 Eclipse 的 Rails 环境。如果需要 RadRails 的更多信息，请访问 <http://www.radrails.org/>。

DB2 支持 Ruby 1.8.5 版本及更高版本以及 Ruby on Rails 1.2.1 版本及更高版本。IBM_DB gem 包括 IBM_DB Ruby 驱动程序和 Rails 适配器，让您可以使用 DB2 工作，并基于 CLI 层。这个 gem 必须与 IBM Data Server Client 一起安装。您就可以使用 Ruby gem 或作为一个 Rails 插件安装 IBM_DB 驱动程序和适配器。

14.3.9 Perl

Perl 是一种流行的解释型编程语言，在许多操作系统中可以免费使用。它使用动态 SQL，它是设计应用程序原型的理想选择。

Perl 提供了一个名为数据库接口 (DBI) 模块的标准模块来访问不同数据库。可在 <http://www.perl.com> 了解到。这个模块与不同的数据库供应商的驱动程序“对话”。在 DB2 中，它是 DBD: : DB2 驱动程序，可以从 <http://www.ibm.com/software/data/db2/perl> 得到。

14.3.10 Python

Python 是一种常用于脚本的动态语言。它强调代码的可读性并支持多种编程模式，包括过程，面向对象，面向方面，meta 和函数编程等。Python 是快速开发应用程序的理想选择。

表 14.4 显示了能够从 Python 应用程序存取 DB2 数据库的扩展。

表 14.4 - IBM Data Server - Python 扩展

扩展	描述
ibm_db	由 IBM 定义 为高级功能提供了最好的支持。 允许您发出 SQL 查询，调用存储过程，使用 pureXML 和访问元数据信息。
ibm_db_dbi	实现了 Python Database API Specification v2.0。 它不提供 ibm_db API 所支持的一些高级功能。 如果您有一个驱动程序支持 Python Database API Specification v2.0 的应用程序，您可以轻松地切换到 ibm_db。ibm_db 和 ibm_db_dbi 的 API 是包装在一起的。
ibm_db_sa	支持 SQLAlchemy，是流行的开源 Python SQL 工具包和对象关系映射 (ORM)。

14.4 XML 与 DB2 pureXML

可扩展标记语言 (XML) 是 Web2.0 工具和技术的基础，同时也是面向服务的体系结构 (SOA) 的基础。IBM 公司很早就认识到了 XML 的重要性，并投入大量资金研发了 pureXML 技术——一项更好地支持在 DB2 中存储 XML 文档的技术。

在 2006 年推出的 DB2 9 是一款混合数据服务器：它允许本机存储关系数据，以及分层数据。虽然以前版本的 DB2 和市场上的其他数据服务器可以存储 XML 文档，但是在 DB2 9 中采用的存储方法具有更好的性能和灵活性。使用 DB2 9 的 pureXML 技术，XML 文档在内部利用解析分层方式存储，就像一棵树一样；因此，XML 文档实用性大大增强。DB2 的较新版本，例如 DB2 9.5 和 DB2 9.7，进一步改善了对 pureXML 的支持。在第 15 章 DB2 pureXML 中，会详细探讨这个问题。

14.5 Web 服务

作为一个简单的定义，Web 服务被作为一个可以通过网络调用的函数，您不需要知道开发它的编程语言，不需要知道运行该函数的操作系统，不需要知道它运行的位置。Web 服务允许一个应用程序使用基于 XML 的可扩展的工业标准协议与另一个应用程序交换数据，如图 14.8 所示。

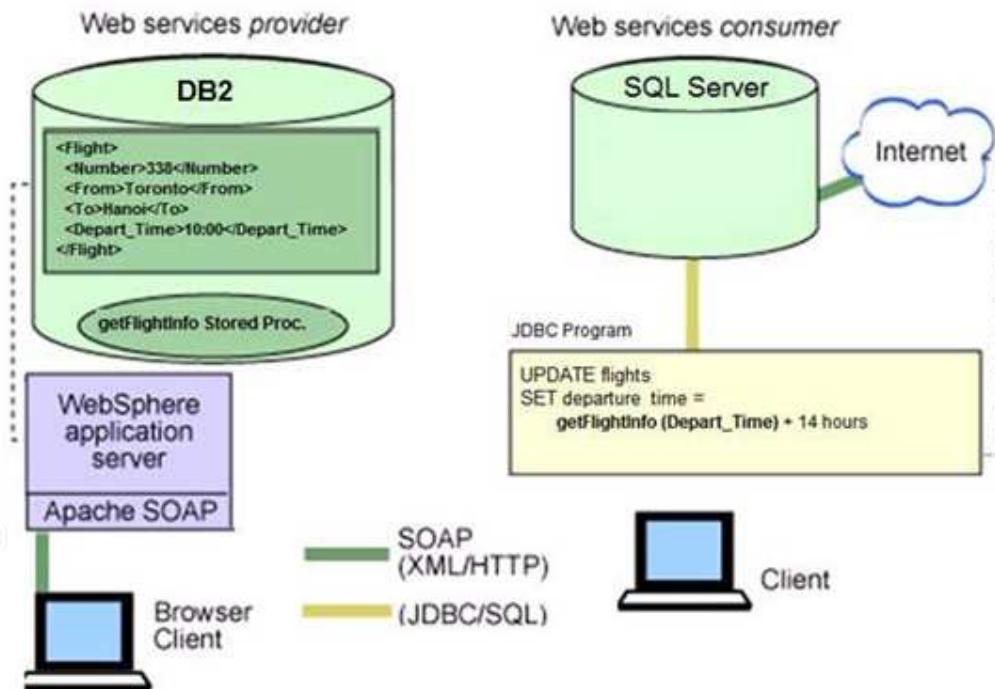


图 14.8 – Web 服务的工作过程

在图中，假设左边代表着一个虚构的航空公司系统，Atlantis 航空公司是在 Linux 中使用 DB2，并以 XML 格式存储其航班信息在 DB2 数据库中。在右侧我们虚构另一个航空公司，Discovery 航空公司是使用 SQL Server 运行于 Windows 上。现在让我们假设 Atlantis 航空公司和 Discovery 航空公司签署一项合作协议，其中两家公司要共享调度和价格信息用以协调他们的航班。共享信息给这两家使用不同操作系统（Linux, Windows）、不同数据服务器（DB2, SQL Server）的公司是一个挑战。当 Atlantis 航空公司改变从多伦多到北京的航班时刻表，Discovery 航空公司如何自动调整其从北京转机到上海的航班时刻表？答案在于 Web 服务。Atlantis 航空公司可以通过创建一个 **Data Web service** 公开一些航班信息，它从 DB2 数据库输出返回一个航班信息的存储过程（getFlightInfo 存储过程）。**Data Web service** 是一个基于数据库信息的 Web 服务，当 **Data Web service** 被部署到一个应用服务器如 WebSphereApplication Server 的时候，客户或者合作伙伴如 Discovery 航空公司可以非常容易地使用浏览器访问 Atlantis 航空公司的航班信息。在这个例子中，Atlantis 航空公司作为 Web 服务提供者，制定并提供 Web 服务，而 Discovery 航空公司作为 Web 服务消费者，消费或使用 Web 服务。

Discovery 航空公司也能从自己的 JDBC 应用程序调用 Web 服务，以便于使用它的 SQL Server 数据库中的数据执行计算。例如，如果从多伦多飞往北京的时间平均为 12 小时，Discovery 航空公司可以通过 Atlantis 航空公司航班离开多伦多的起飞时间，加上飞行持续时间和一些缓冲时间来计算从北京转机到上海的出发时间。需要使用的缓冲时间量可以存储在 Discovery 航空公司系统的 SQL Server 数据库中，在 JDBC 应用程序中使用的简单公式是这样的：

Air Discovery Flight Departure time (Beijing to Shanghai)	=	Air Atlantis Flight Departure time (Toronto to Beijing)	+	Air Atlantis Flight Duration (Toronto to Beijing) 12 hours average	+	Air Atlantis Flight Buffer time (Toronto to Beijing) 2 hours
---	---	---	---	---	---	---

如果 Atlantis 航空公司变更航班的起飞时间，当 Discovery 航空公司系统调用 Web 服务时该信息会自动传递给它。

14.6 管理 API

DB2 提供了大量的管理 API，开发人员可以使用它来建立自己的实用程序或工具。例如，要创建一个数据库，您可以调用 `sqlcrea` API；启动一个实例，使用 `db2InstanceStart` API；或导入数据到表中，使用 `db2Import` API。完整的列表可以从 DB2 信息中心得到。请参阅 DB2 信息中心网站的资源部分。

14.7 其他发展

DB2 Express-C 的一些用户还与第三方产品交互作用，如 MS Excel 和 MS Access 创建简单的表单连接到 DB2。在本节中我们将描述如何使用这些产品和 DB2 Express-C。

DB2 Express-C 在 Mac OS X 上也可用，所以您可以在 Mac 上使用 DB2 开发数据库应用程序。这对已经接受 Mac 平台的 RoR 社区可能是很大的吸引力。

14.7.1 使用 Microsoft Access 和 Microsoft Excel

Microsoft Access 和 Microsoft Excel 是用来生成报表，创建表单，开发简单的提高数据商业用途的应用程序的常用工具。DB2 很容易与这些工具交互。DBA 可以将公司数据存储在一个安全的 DB2 服务器中，普通的 Access 或 Excel 用户可以访问这些数据并生成报告。这如图 14.9 所示。

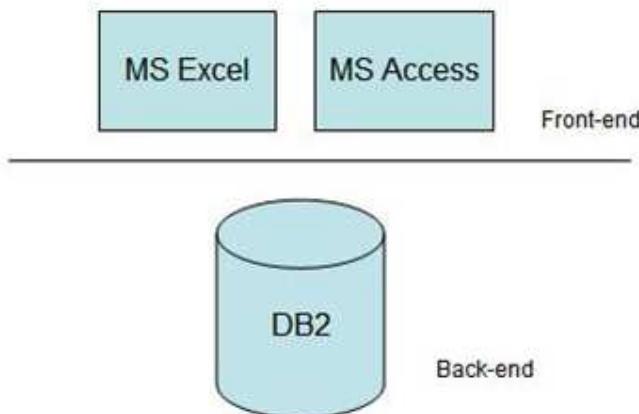


图 14.9 – 使用 Excel、Access 和 DB2

在图中，Excel 和 Access 可以用于开发前端应用，而 DB2 作为后端应用负责数据的安全性，可靠性和性能。将所有数据集中在 DB2 中的数据创建一个简化存储模型。

在 Excel 中，最简单的方法来访问 DB2 数据是使用 OLE DB 驱动程序，如 IBM OLE DB Provider for DB2。该驱动包括在可以从 DB2 Express-C 的 www.ibm.com/db2/express 网站免费下载的 IBM Data Server Client 中。一旦安装，您需要从 MS Excel 菜单中选择合适的 OLE DB 供给程序（OLE DB provider）数据源。选择数据->导入外部数据->导入数据。接下来的步骤都记录在 IBM® DB2® Universal Database™ 和 Microsoft® Excel 应用程序开发入门[1]中。详见参考文献。

在使用 Microsoft Access 的情况下，您也应该安装以下三个软件之一：

- IBM 数据服务器（IBM Data Server）客户端，
- IBM 数据服务器的 ODBC，CLI 和 .Net 驱动程序，
- IBM 数据服务器的 ODBC 和 CLI 驱动程序。

IBM 数据服务器的 ODBC，CLI 和 .Net 驱动程序和 IBM 数据服务器的 ODBC 和 CLI 驱动程序也称为 IBM DB2 ODBC 驱动程序，和 DB2 CLI 驱动程序一样。这个驱动程序用来从 Access 连接到 DB2。安装驱动程序后，创建一个 Access 2007 项目，选择 ODBC 数据库，在表格工具带中选择外

部数据标签的可选选项。接下来的具体步骤参见 *DB2 9 and Microsoft Access 2007, Part 1: Getting the Data [2]*。当使用 Microsoft Access 中的链接表时, Access 2007 的用户可以得到数据, 但数据本身仍存储在 DB2 数据服务器上。

对于 2007 年之前的 Access 版本, 安装有些不同, 您可以参考 *Use Microsoft Access to interact with your DB2 data [3]*。

14.8 开发工具

Microsoft Visual Studio 和 Eclipse 是两种被开发人员使用的集成开发环境 (IDE)。这两种集成开发环境与 DB2 交互工作得很好。

对于 Microsoft Visual Studio, DB2 提供了一个 Visual Studio 加载项, 这样安装后, IBM 工具被添加到 Visual Studio 菜单中。这样, 开发人员不需要切换到其他工具来使用 DB2 数据库。您可以从 DB2 Express-C 的 www.ibm.com/db2/express 网站下载 Visual Studio。

对于 Eclipse, IBM 发布了 IBM Data Studio, 一个基于 Eclipse 的免费工具, 允许您开发 SQL 和 XQuery 脚本, 存储过程, UDF 和 Web 服务。由于它是基于 Eclipse 平台, 许多开发人员可以利用他们现有的知识来使用这个工具。

14.9 示例程序

为了帮助您了解如何在不同的语言中使用 DB2 作为数据服务器, 您可以参考 *SQLLIB\samples* 目录里与 DB2 服务器安装程序在一起的应用程序示例。图 14.10 显示了一些 Windows 平台 DB2 的示例程序。

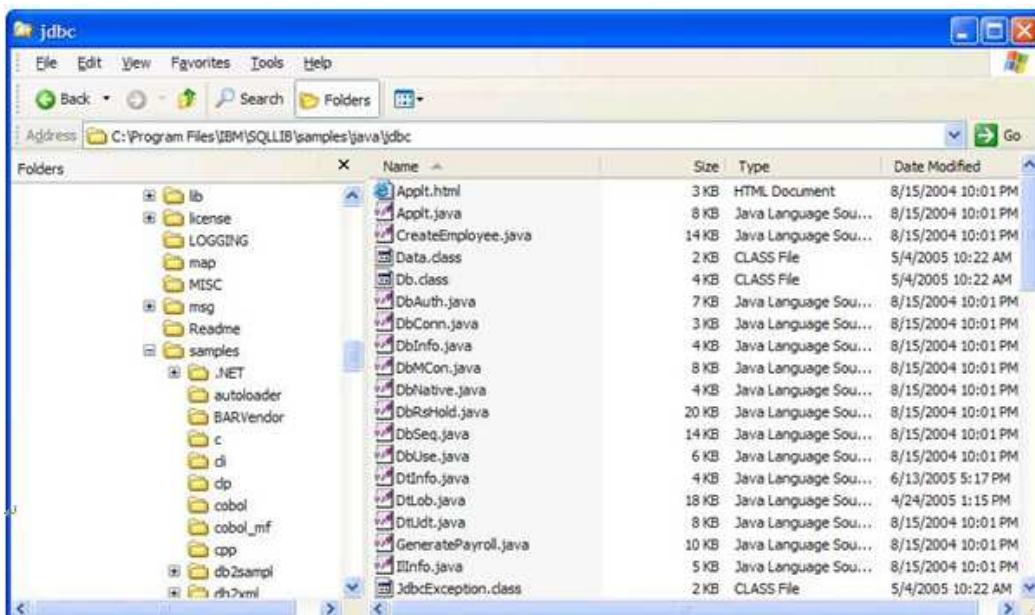


图 14.10 – 示例程序与 DB2

14.10 小结

在这一章中, 我们看到了 DB2 如何给服务器上的数据库, 或通过客户端应用程序与 DB2 数据服务器连接的应用程序提供灵活性。

服务器端的应用包括存储过程, 用户定义的函数和触发器。

在客户端, 我们讨论了很多 DB2 应用程序开发的编程接口和方法, 再次展现了 DB2 作为数据服务器的高灵活性和能力。

15

第 15 章 – DB2 pureXML

本章我们将讨论 pureXML，DB2 9 引入这一新技术来支持 XML 原生存储。本章中很多示例和概念皆引自于 IBM 红皮书：《DB2 9: pureXML 概览和快速启航》。更多信息请参阅“参考资源”这一节。图 15.1 的 DB2 概览图指示出我们在本章要讨论的内容。

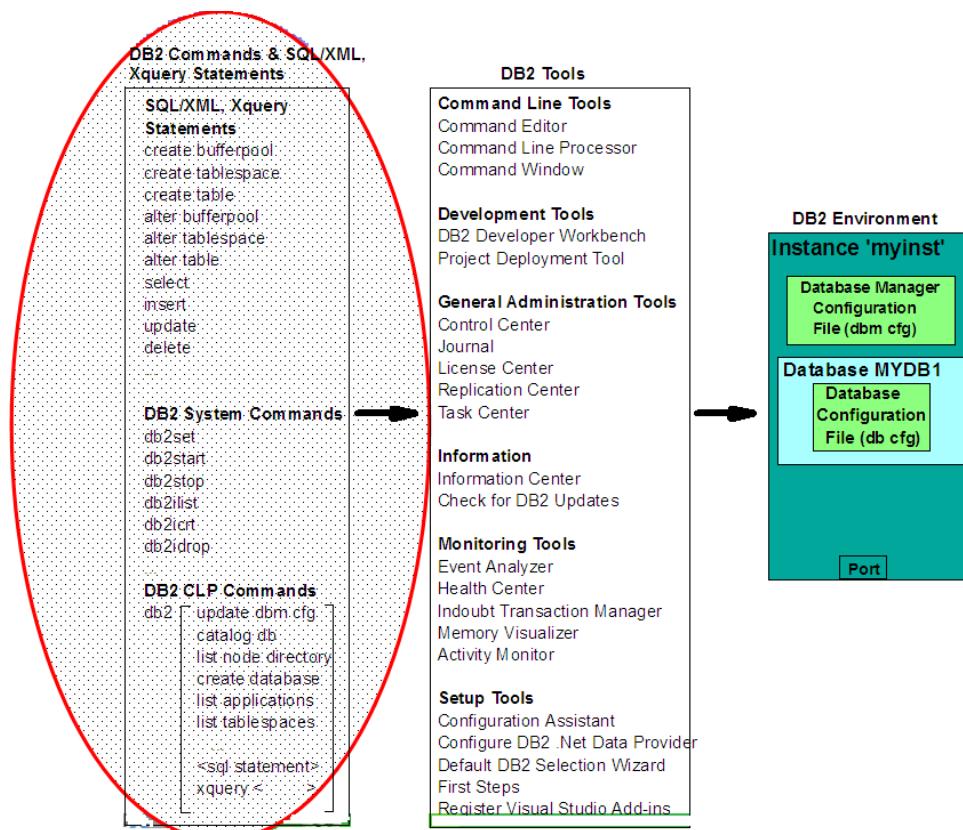


图 15.1 – DB2 概览图: DB2 命令, SQL/XML 和 XQuery

注意:

更多关于 pureXML 的信息，请参见以下视频：

<http://www.channeldb2.com/video/video/show?id=807741:Video:4382>

15.1 在数据库中使用 XML

XML 文档可以存储在文本文件、XML 库或者数据库中。许多公司更倾向于把它们存储在数据库中，这主要基于两个原因：

- 管理海量 XML 数据是一个数据库的问题。XML 与其他数据相似，只是格式不同罢了。使用数据库存储关系型数据的原因也同样适用于 XML 数据：数据库提供了高效的数据检索和获取途径，对于数据持久化的健壮支持，数据备份和恢复，事务处理支持以及性能保证和可扩展性。
- 集成：通过将关系型数据和 XML 数据存储在一起，您可以把新的 XML 数据和现有的关系型数据集成起来，同时把 SQL 和 XPath 或者 XQuery 组合成为单一查询。另外，关系型数据可以用 XML 格式发布，反之亦然。通过集成，数据库可以更好的支持 Web 应用、SOA 和 Web 服务。

15.2 XML 数据库

有两种类型数据库可以存储 XML 数据：

- 启用 XML 的数据库
- 原生 XML 数据库

15.2.1 启用 XML 的数据库

启用 XML 的数据库使用关系模型作为其核心数据存储模型。这需要在 XML（层次型）数据模型和关系数据模型之间建立映射，或者把 XML 数据存储为字符大对象。尽管稍显“陈旧”，很多数据库厂商依然在使用这一技术。图 15.2 更细致的解释了启用 XML 的数据库存储数据的两种方式。

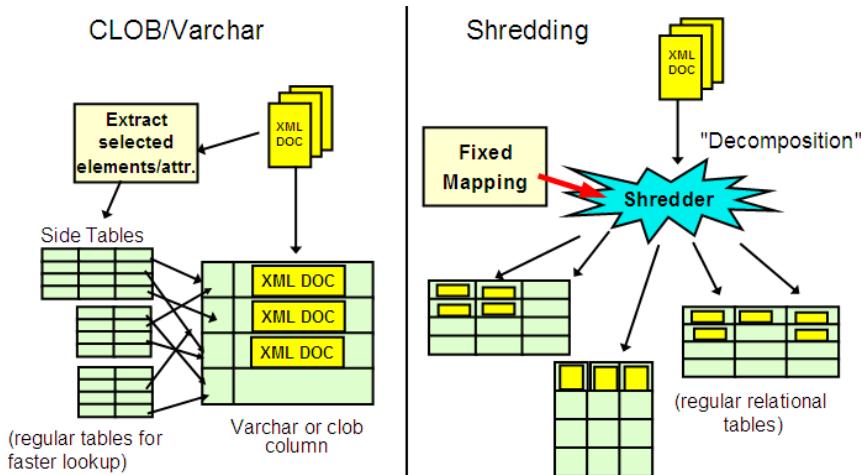


图 15.2 –启用 XML 的数据库的两种存储 XML 的方式

图 15.2 左半部分描述了使用“字符大对象和变长字符串”存储 XML 的方法。这种方法以字符大对象或者变长字符串的形式把 XML 文档存储为未经解析的字符串。如果 XML 文档以字符串形式存储，那么欲获取其部分内容，应用程序就必须获取整个字符串，然后完成解析的工作。显然，这种方法缺少灵活性。

图 15.2 右半部分描述了启用 XML 的数据库的另一种存储 XML 数据的方法，称为“裂分或者解构”，整个 XML 文档会被裂分作存储在数据表中的小块。通过这种方法，XML 文档的层次模型被转化成关系模型。这样同样无益于灵活性：XML 文档的一处改动不能轻易传递到相应的数据表，而且

很可能需要创建额外的许多表。这种方法对性能的影响同样不能让人满意：当您需要取回原始 XML 文档时，您必须执行代价可观的 SQL 操作，而当牵涉到更多的表时，这一代价将更加惊人。

15.2.2 原生 XML 数据库

原生 XML 数据库在内部使用层次型数据模型存储、处理 XML 数据。存储格式与处理格式一致，即层次型数据模型不再映射为关系数据模型，XML 文档也不再存储为映像。XPath 或者 XQuery 交由本地引擎直接处理，而不再转化成 SQL。“原生 XML 数据库”由此得名。自 DB2 V9 起，这是目前提供这一支持的唯一商业数据服务器。

15.3 DB2 中的 XML

图 15.3 大致描述了自 DB2 V9 起，DB2 是如何将关系型数据和层次型数据（XML 文档）存储在一起。假定图中 dept 表定义如下：

```
CREATE TABLE dept (deptID CHAR(8),...,deptdoc XML);
```

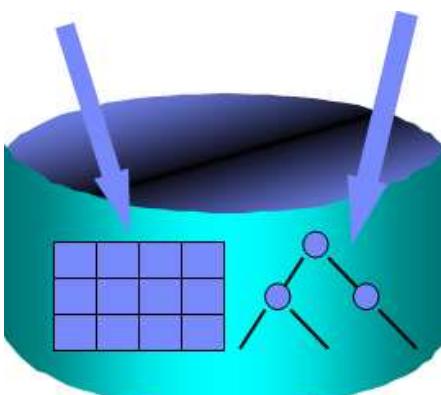


图 15.3 –DB2 中的 XML

注意 deptdoc 列采用了新的数据类型，XML。图中左侧箭头表明 deptID 列存储在关系表中，而 deptdoc 列以经解析的层次型格式存储。

图 15.4 说明从 DB2 9 开始，有四种方式访问数据：

- - 使用 SQL 访问关系型数据
- - 使用带有 XML 扩展的 SQL 访问 XML 数据
- - 使用 XQuery 访问 XML 数据
- - 使用 XQuery 访问关系型数据

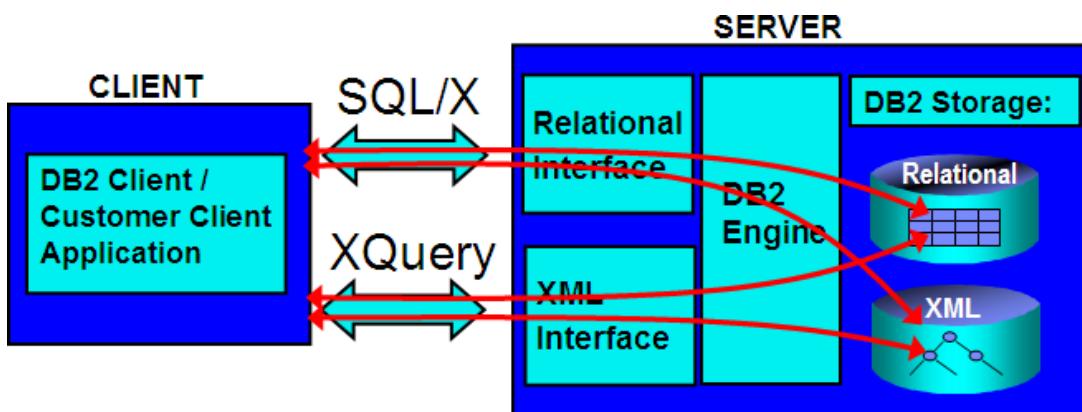


图 15.4 –存取 DB2 数据的四种方法

因此，惯用 SQL 的人会把 DB2 看作世界级的关系型数据库管理系统，同时扩展了对于 XML 的支持；而 XML 的使用者会认为 DB2 是世界级的 XML 数据库，同时对 SQL 给予了支持。

请注意 IBM 使用 pureXML 而不是“native XML”来定义该创新技术。因为其他厂商使用“字符大对象/变长字符”或者“解构”的早期技术来存储 XML 文档，并且称之为“native XML”。为了避免混淆，IBM 决定使用 pureXML 并且注册了商标，这样其他数据库或者 XML 技术厂商不能再使用这一术语来为某个相异的技术命名。在 DB2 9.1 中，pureXML 仅支持 Unicode 数据库。DB2 9.5 解除了这个限制，无论是 unicode 还是非 unicode 环境，您都可以自由使用 pureXML。

15.3.1 pureXML 技术优势

pureXML 提供了许多技术上的优势。

1. 考虑到关系表使用新的 XML 数据类型存储 XML 文档，这种一致性使得先前在关系型数据库上的投资不会贬值。
2. 减少代码复杂性。例如，图 15.5 展示了使用和不使用 pureXML 的同一段 PHP 脚本的变化。使用 pureXML（绿色方框）减少了代码行数。这不仅降低了代码的复杂性，而且因为解析和维护的代码更少，应用的整体性能得到提升。

```

<?php
$conn = db2_connect($dbname, $dbuser, $dbpass);

/* Insert Customer Documents */

$stmt = db2_prepare($conn, "VALUES (NEXT VALUE FOR Cid)");
db2_execute($stmt);
list($Cid) = db2_fetch_array($stmt);

$fileContents = file_get_contents("customers/c1.xml");

$stmt = db2_prepare($conn, "INSERT INTO xmlecustomer(Cid, Info) VALUES (?, ?)");
if(!db2_execute($stmt, array($Cid, $fileContents)))
    echo db2_stmt_errormsg($stmt);

/* Insert Product Documents */

$fileContents = file_get_contents("products/p1.xml");
$dom = simplexml_load_string($fileContents);

$prodID = (string) $dom["pid"];

$stmt = db2_prepare($conn, "INSERT INTO xmiproduct(Pid, Description) VALUES (?, ?)");
if(!db2_execute($stmt, array($prodID,
)

```

```

    $stmt = db2_execute($stmt);
    list($Cid) = db2_fetch_array($stmt);

    $fileContents = file_get_contents("customers/c1.xml");
    $dom = simplexml_load_string($fileContents);

    $custName = (string) $dom->name;
    $custCountry = (string) $dom->addr->country;
    $custStreet = (string) $dom->addr->street;
    $custCity = (string) $dom->addr->city;
    $custProvince = (string) $dom->addr->("prov-state");
    $custZip = (string) $dom->addr->"PCODE-ZIP";
    $custPhone = (string) $dom->phone;

    $stmt = db2_prepare($conn, "INSERT INTO sqicustomer(Cid, Name, Country, Street, City, Province, Zip, Phone, Info) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");
    if(!db2_execute($stmt, array($Cid, $custName, $custCountry, $custStreet, $custCity, $custProvince, $custZip, $custPhone, $fileContents)))
        echo db2_stmt_errormsg($stmt);

    /* Insert Product Documents */

    $fileContents = file_get_contents("products/p1.xml");
    $dom = simplexml_load_string($fileContents);

    $prodID = (string) $dom["pid"];
)

```

图 15.5 – 使用或不使用 pureXML 的代码复杂性对比

3. 使用 XML 和 pureXML 技术使变更数据库模式更加容易。图 15.6 是 pureXML 带来更多灵活性的例子。图中假定数据库由一张雇员表和一张部门表组成。通常在非 XML 数据库中，如果经理让您为每一位雇员存储不止一个电话号码（除了家庭电话还有手机号码），那么您会在雇员表中增加一列来存储手机号码。然而，这种做法违背了关系型数据库的规则。如果不违背规则，那么您只能另外创建一个电话表，把所有电话信息迁移到这张表中。然后您可以把手机号码添加到这张表中。创建这张新表的代价很高，不只由于需要移动之前存在的大量数据，同时应用程序中所有相关的 SQL 都必须修改指向这张新表。

而图的左侧展示了如何通过 XML 解决这个问题。如果雇员“Christine”同时有一个手机号码，那么可以添加一个新的标签来对应这条信息。“Michael”没有手机号码，那么我们什么都不需要做。

<pre> <DEPARTMENT deptid="15" deptname="Sales"> <EMPLOYEE> <EMPNO>10</EMPNO> <FIRSTNAME>CHRISTINE</FIRSTNAME> <LASTNAME>SMITH</LASTNAME> <PHONE>408-463-4963</PHONE> <PHONE>415-010-1234</PHONE> <SALARY>52750.00</SALARY> </EMPLOYEE> <EMPLOYEE> <EMPNO>27</EMPNO> <FIRSTNAME>MICHAEL</FIRSTNAME> <LASTNAME>THOMPSON</LASTNAME> <PHONE>406-463-1234</PHONE> <SALARY>41250.00</SALARY> </EMPLOYEE> </DEPARTMENT> </pre>	<p>Requires:</p> <ul style="list-style-type: none"> • Normalization of existing data ! • Modification of the mapping • Change of applications <p>Phone</p> <table border="1"> <thead> <tr> <th>EMPNO</th> <th>PHONE</th> </tr> </thead> <tbody> <tr> <td>27</td> <td>406-463-1234</td> </tr> <tr> <td>10</td> <td>415-010-1234</td> </tr> <tr> <td>10</td> <td>408-463-4963</td> </tr> </tbody> </table> <p>Department</p> <table border="1"> <thead> <tr> <th>DEPTID</th> <th>DEPTNAME</th> </tr> </thead> <tbody> <tr> <td>15</td> <td>Sales</td> </tr> </tbody> </table> <p>Employee</p> <table border="1"> <thead> <tr> <th>DEPTID</th> <th>EMPNO</th> <th>FIRSTNAME</th> <th>LASTNAME</th> <th>PHONE</th> <th>SALARY</th> </tr> </thead> <tbody> <tr> <td>15</td> <td>27</td> <td>MICHAEL</td> <td>THOMPSON</td> <td>406-463-1234</td> <td>41250</td> </tr> <tr> <td>15</td> <td>10</td> <td>CHRISTINE</td> <td>SMITH</td> <td>408-463-4963</td> <td>52750</td> </tr> </tbody> </table>	EMPNO	PHONE	27	406-463-1234	10	415-010-1234	10	408-463-4963	DEPTID	DEPTNAME	15	Sales	DEPTID	EMPNO	FIRSTNAME	LASTNAME	PHONE	SALARY	15	27	MICHAEL	THOMPSON	406-463-1234	41250	15	10	CHRISTINE	SMITH	408-463-4963	52750
EMPNO	PHONE																														
27	406-463-1234																														
10	415-010-1234																														
10	408-463-4963																														
DEPTID	DEPTNAME																														
15	Sales																														
DEPTID	EMPNO	FIRSTNAME	LASTNAME	PHONE	SALARY																										
15	27	MICHAEL	THOMPSON	406-463-1234	41250																										
15	10	CHRISTINE	SMITH	408-463-4963	52750																										

图 15.6 – 使用 XML 增加数据灵活性

4. 提升 XML 应用程序的性能。对比测试表明，使用 pureXML 技术后几种不同类型 XML 应用程序的性能都会得到显著改善。测试针对一家由其他早期技术迁移到 pureXML 的公司，表 15.1 给出了测试结果。中间列对应早期 XML 处理技术，第三列是使用 pureXML 的效果。

任务	其它关系数据库	DB2 使用 pureXML
商用搜索和查找过程的开发	CLOB: 8 小时 Shred: 2 小时	30 分钟
相应的 I/O 代码的行数	100	35 (减少 65%)
在模式中添加域	1 星期	5 分钟
查询	24 - 36 小时	20 秒 - 10 分钟

表 15.1 – 使用 pureXML 技术能够提升应用程序的性能

15.3.2 XPath 基础

XPath (XML Path Language, XML 路径语言) 是一种用来查询 XML 文档的语言。列表 15.1 是一个 XML 文档的例子，图 15.7 经过解析后的层次结构 (也称作“节点”或“叶片”)。我们将用层次结构解释 XPath 如何工作。

```
<dept bldg="101">
  <employee id="901">
    <name>John Doe</name>
    <phone>408 555 1212</phone>
    <office>344</office>
  </employee>
  <employee id="902">
    <name>Peter Pan</name>
    <phone>408 555 9918</phone>
    <office>216</office>
  </employee>
</dept>
```

列表 15.1 – 一个 XML 文档

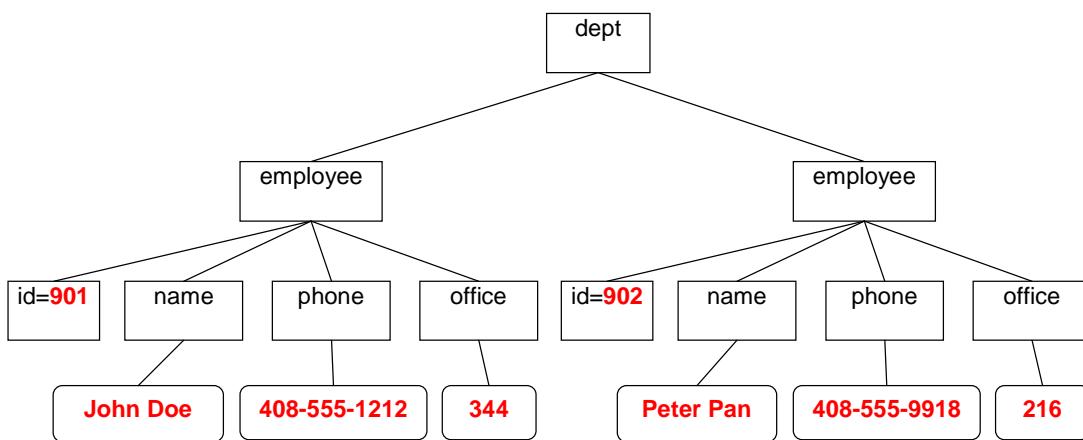


图 15.7 – 列表 15.1 的 XML 文档经过解析后的层次结构

比较 XPath 和 MS-DOS 或者 Linux/UNIX 中的更改目录命令是学习 XPath 的快捷途径。在 MS-DOS 或者 Linux/UNIX 中，使用如下 cd 命令展开目录树：

```
cd /directory1/directory2/...
```

同样，XPath 使用斜线“/”从 XML 文档的一个元素定位到另一个。例如，对于图 15.9 所示的文档可以使用如下查询获得所有雇员的姓名：

```
/dept/employee/name
```

15.3.2.1 XPath 表达式

XPath 表达式使用完整限定路径指定元素和（或者）属性。符号“@”用来指定属性。若仅仅需要获得一个元素的值（文本节点），则使用“text()”函数。表 15.2 列出了针对列表 15.1 中 XML 文档的 Path 查询和相应地结果。

XPath	结果
/dept/@bldg	101
/dept/employee/@id	901 902
/dept/employee/name	<name>Peter Pan</name> <name>John Doe</name>
/dept/employee/name/text()	Peter Pan John Doe

表 15.2 – XPath 表达式举例

15.3.2.2 XPath 通配符

XPath 中有两个通配符：

- “*”匹配任意标签名
- “//”是“子或自身”通配符

表 15.3 列出了针对列表 15.1 的文档更多使用通配符的例子。

XPath	结果
/dept/employee/*/text()	John Doe 408 555 1212 344 Peter Pan 408 555 9918 216
/dept/*/@id	901 902
//name/text()	Peter Pan John Doe
/dept//phone	<phone>408 555 1212</phone> <phone>408 555 9918</phone>

表 15.3 – XPath 通配符举例

15.3.2.3 XPath 谓词

谓词部分放置在[]中。作为对照，可以认为它们等价于 SQL 中的 WHERE 子句。例如 `[@id="902"]` 会被解释成“WHERE 属性 id 值等于 902”。在一个 XPath 表达式中可以有多个谓词。要指定特定位置的项，使用 `[n]`，这表明同一级元素的第 n 项会被选中。例如，`employee[2]` 表示第二个雇员被选中。表 15.4 列出了更多示例。

XPath	结果
<code>/dept/employee[@id="902"]/name</code>	<code><name>Peter Pan</name></code>
<code>/dept[@bldg="101"]/employee[office >"300"]/name</code>	<code><name>John Doe</name></code>
<code>//employee[office="344" OR office="216"]/@id</code>	<code>901 902</code>
<code>/dept/employee[2]/@id</code>	<code>902</code>

表 15.4 –XPath 谓词举例

15.3.2.4 XPath 的父元素

与 MS-DOS 或者 Linux/UNIX 相似，在表达式中使用“.”表示引用当前上下文，使用“..”表示引用上一级上下文。更多示例参见表 15.5。

XPath	结果
<code>/dept/employee/name[../@id="902"]</code>	<code><name>Peter Pan</name></code>
<code>/dept/employee/office[.>"300"]</code>	<code><office>344</office></code>
<code>/dept/employee[office > "300"]/office</code>	<code><office>344</office></code>
<code>/dept/employee[name="John Doe"]/../@bldg</code>	<code>101</code>
<code>/dept/employee/name[.= "John Doe"]../../@bldg</code>	<code>101</code>

表 15.5 – XPath 的父元素

15.3.3 XQuery 的定义

XQuery 是为 XML 创建的查询语言。XQuery 支持路径表达式从而为 XML 层次结构提供导航。事实上，XPath 是 XQuery 的子集；因而先前关于 XPath 的所有知识也适用于 XQuery。XQuery 同时支持类型化和无类型数据。因为 XML 文档不包含遗失的或者未知数据所以 XQuery 不提供空值。XQuery 返回 XML 数据序列。请注意 XQuery 和 XPath 表达式都区分大小写，这一点非常重要。

XQuery 支持 FLWOR 表达式。FLWOR 与 SQL 中 SELECT-FROM-WHERE 表达式相似。下一节将会探讨 FLWOR 的更多细节。

XQuery 的 FLWOR 表达式

FLWOR 是几个词首字母的缩写：

- FOR：对序列进行迭代
- LET：绑定变量

- WHERE: 定义过滤器
- ORDER: 将过滤结果排序
- RETURN: 返回查询结果

该表达式允许对 XML 文档进行操作，同时返回另一个表达式。例如，假定一张表定义如下：

```
CREATE TABLE dept(deptID CHAR(8),deptdoc XML);
```

将如下 XML 文档插入到 deptdoc 列：

```
<dept bldg="101">
  <employee id="901">
    <name>John Doe</name>
    <phone>408 555 1212</phone>
    <office>344</office>
  </employee>
  <employee id="902">
    <name>Peter Pan</name>
    <phone>408 555 9918</phone>
    <office>216</office>
  </employee>
</dept>
```

然后可以像如下使用 FLWOR 表达式：

```
xquery
for $d in db2-fn:xmlcolumn('dept.deptdoc')/dept
let $emp := $d//employee/name
where $d/@bldg > 95
order by $d/@bldg
return
<EmpList>
{$d/@bldg, $emp}
</EmpList>
```

返回结果如下：

```
<EmpList bldg="101">
  <name>
    John Doe
  </name>
  <name>
    Peter Pan
  </name>
</EmpList>
```

15.3.4 插入 XML 文档

可以使用 SQL INSERT 语句或者导入工具向 DB2 数据库插入 XML 文档。此处 XQuery 无能为力因为在 XQuery 标准中没有定义这项操作。

```
db2 -tvf table_creation.txt
```

可以使用 db2 –tvf table_creation.txt 命令从 DB2 的命令窗口或者 Linux Shell 中运行下面这段脚本：

```
table_creation.txt
-- (1)
drop database mydb
;

-- (2)
create database mydb using codeset UTF-8 territory US
;

-- (3)
connect to mydb
;

-- (4)
create table items (
    id      int primary key not null,
    brandname  varchar(30),
    itemname   varchar(30),
    sku       int,
    srp      decimal(7,2),
    comments   xml
);
;

-- (5)
create table clients(
    id      int primary key not null,
    name     varchar(50),
    status    varchar(10),
    contact   xml
);
;

-- (6)
insert into clients values (77, 'John Smith', 'Gold',
    '<addr>111 Main St., Dallas, TX, 00112</addr>')
;
;

-- (7)
IMPORT FROM "D:\Raul\clients.del" of del xml from "D:\Raul" INSERT INTO CLIENTS (ID,
NAME, STATUS, CONTACT)
;
;

-- (8)
IMPORT FROM "D:\Raul\items.del" of del xml from "D:\Raul" INSERT INTO ITEMS (ID,
BRANDNAME, ITEMNAME, SKU, SRP, COMMENTS)
;
;
```

注意这个脚本和其他相关文件位于随书光盘的 **expressc_book_quicklabs.zip** 文件中。接下来我们逐行解释这段脚本。

1. 删除“mydb”数据库。这一步执行清理。如果“mydb”数据库不存在，会得到错误提示，忽略即可。

2. 使用 UTF-8 字符集创建“mydb”数据库。Unicode 和非 Unicode 数据库都支持 pureXML。
3. 连接新建立的“mydb”数据库。接下来才可以在数据库中创建对象。
4. 建立“items”表。注意表的最后一列 “comments” 被定义为 XML 列，使用了新的 XML 数据类型。
5. 建立“clients”表。最后一列同样使用新的 XML 数据类型进行定义。
6. 用 SQL 的 INSERT 语句将 XML 文档插入 XML 列。在 INSERT 语句中 XML 文档被放在一对单引号里当作字符串来传递。
7. 用 IMPORT 命令可以把几个 XML 文档连同关系型数据一起导入到数据库中。第(7)部分导入的数据来自 clients.del 文件（一个有界 ascii 文件），该行同时指明了 clients.del 文件引用的 XML 数据存放何处。

首先来看目录 D:\Raul 下有什么内容（图 15.10），clients.del 文件我们稍后关注。

Name	Size	Type
Client3227.xml	1 KB	XML Document
Client4309.xml	1 KB	XML Document
Client5681.xml	1 KB	XML Document
Client8877.xml	1 KB	XML Document
Client9077.xml	1 KB	XML Document
Client9177.xml	1 KB	XML Document
ClientInfo.xsd	2 KB	XML Schema
clients.del	1 KB	DEL File
Comment3926.xml	1 KB	XML Document
Comment4023.xml	1 KB	XML Document
Comment4272.xml	1 KB	XML Document
items.del	1 KB	DEL File

图 15.10 - D:\Raul 目录的内容，里面包含了 del 文件和 xml 文件

这是 clients.del 文件的内容

clients.del

```
3227,Ella Kimpton,Gold,<XDS FIL='Client3227.xml' />,
8877,Chris Bontempo,Gold,<XDS FIL='Client8877.xml' />,
9077,Lisa Hansen,Silver,<XDS FIL='Client9077.xml' />,
9177,Rita Gomez,Standard,<XDS FIL='Client9177.xml' />,
5681,Paula Lipenski,Standard,<XDS FIL='Client5681.xml' />,
4309,Tina Wang,Standard,<XDS FIL='Client4309.xml' />
```

在这个文件里，“XDS FIL=”用来指向某一 XML 文档。

运行上面的脚本后，控制中心如图 15.11 所示。

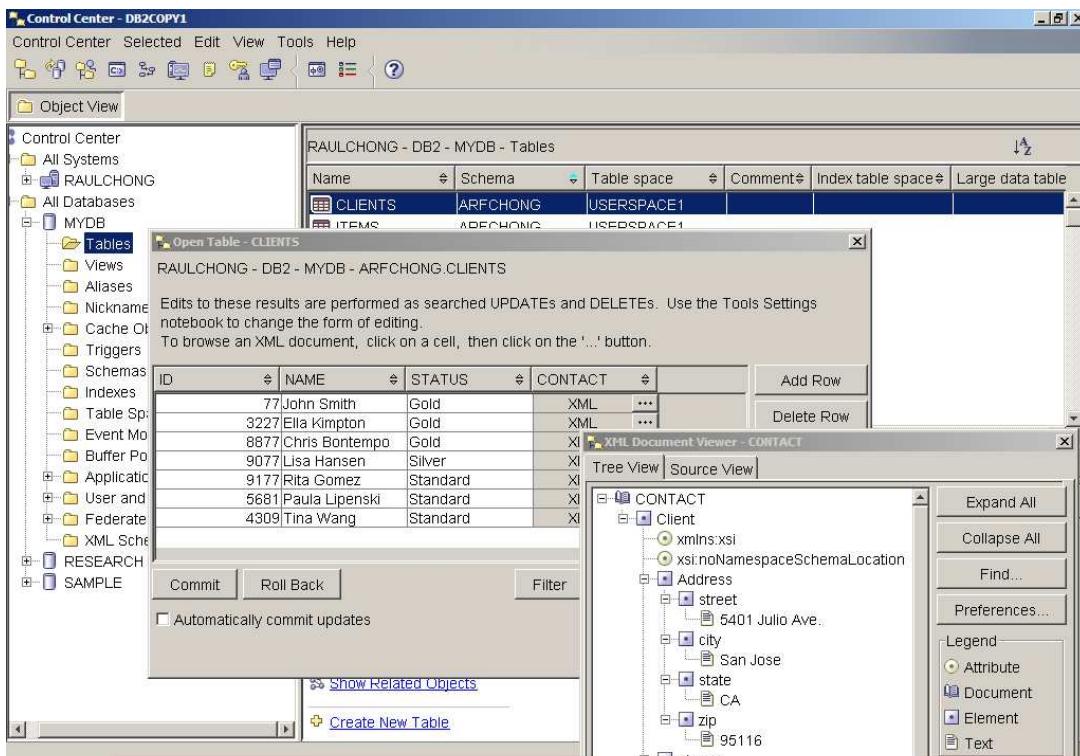


图 15.11 –执行完 table_creation.txt 后的控制中心视图

注意图中显示了 clients 表的内容。最后一列“Contact”是 XML 列。点击有三个点的按钮，将会在另一个窗口显示 XML 文档的内容。图中该窗口位于右下角。

15.3.5 查询 XML 数据

在 DB2 中有两种方式查询 XML 数据：

- 使用带有 XML 扩展的 SQL(SQL/XML)
- 使用 XQuery

无论那一种方式，DB2 都遵循 XML 的国际标准。

SQL/XML 查询 XML 数据

普通的 SQL 语句可以处理行和列。SQL 语句也可以处理完整的 XML 文档；然而，当要获取文档的部分内容时，普通的 SQL 语句就捉襟见肘了。在这种情形下，需要使用带有 XML 扩展的 SQL(SQL/XML)。

表 15.5 列举了符合 SQL2006 标准的部分 SQL/XML 函数。

函数名	描述
XMLPARSE	解析字符或者大对象二进制数据，产生 XML
XMLSERIALIZE	将 XML 值转换为字符或者大对象二进制数据
XMLVALIDATE	根据 XML schema 校验 XML 值的有效性
XMLEXISTS	检测 XQuery 是否返回结果（如由一个或多个项所组成的序列）
XMLQUERY	执行一个 XQuery 并以一个序列返回结果
XMLTABLE	执行一个 XQuery，以关系数据表形式返回结果（如果可

XMLCAST 表 15.5 – SQL/XML 函数	能的话) XML 类型转换
--------------------------------	------------------

可以使用先前创建的“mydb”数据库测试下面的示例。

例一

这是一个查询示例。设想您要获得居住在某一地区的所有客户的姓名。clients 表在 XML 列里存储了客户的地址和邮编。使用 XMLEXISTS 函数，可以在 XML 列搜索目标邮编，根据条件返回结果集。

```
SELECT name FROM clients
WHERE xmlexists(
  '$c/Client/Address[zip="95116"]'
  passing clients.contact as "c"
)
```

第一行 SQL 语句指定要从“clients”表的 name 列获取信息。

在 WHERE 子句中调用 XMLEXISTS 函数，参数指定了 XML 路径表达式，DB2 据此导航到 XML 文档邮编元素并且查找值 95116。

“\$c/Client/Address”指定了 DB2 在 XML 文档中定位邮编元素的路径。美元符号(\$)用来指定变量。passing clients.contact as "c"对变量 c 进行定义。这里，“clients”是表名，“contact”是 XML 列的列名。换言之，XML 文档传递给了变量 c。

DB2 检索“contact”列中的 XML 数据，从根节点“client”向下，通过节点“Address”到节点“zip”，经过目标邮编匹配检查判断该客户是否居住在目标地区。若 XMLEXISTS 函数返回“true”，则 DB2 返回该行的客户姓名。

DB2 9.7 简化了上面的查询：

```
SELECT name FROM clients
WHERE xmlexists(
  '$CONTACT/Client/Address[zip="95116"]'
)
```

DB2 自动生成与 XML 列同名的变量。在上例中，CONTACT 变量是 DB2 自动生成的。变量名与 CONTACT 列名相同。

例二

考虑一下如何生成一份列举了所有金牌顾客邮件地址的报告。运行下面的查询：

```
SELECT xmlquery('$c/Client/email' passing contact as "c")
  FROM clients
 WHERE status = 'Gold'
```

第一行表明我们要从 XML 文档的元素中（不是关系列）提取邮件地址。同上例，“\$c”是指向 XML 文档的变量。XMLEXISTS 函数可以用在 WHERE 后面，类似的可以在 SELECT 后面使用 XMLQUERY 函数。

例三

有时您想用表来呈现 XML 数据。XMLTABLE 函数帮助完成这一工作，演示如下。

```
SELECT t.comment#, i.itemname, t.customerID, Message
```

```

FROM items i,
xmltable('$c/Comments/Comment' passing i.comments as "c"
columns Comment# integer path 'CommentID',
CustomerID integer path 'CustomerID',
Message varchar(100) path 'Message') AS t

```

第 1 行指定了包含在结果集中的列。以变量 t 作为前缀表明该列的值是 XML 文档的元素值。

第 2 行调用 XMLTABLE 函数，参数指定了包含目标数据的 DB2 XML 列，以及目标数据在 XML 文档中的路径。

第 4 行至第 6 行的“columns”语句将 XML 元素映射到结果集中相应各列。映射还指定了 XML 元素值将要转换成的数据类型。示例中，所有 XML 数据都被转换成传统 SQL 数据类型。

例四

现在我们看一个简单的例子，该例子在 XMLQUERY 函数中使用 XQuery FLWOR 表达式。

```

SELECT name, xmlquery(
  'for $e in $c/Client/email[1] return $e'
  passing contact as "c"
)
FROM clients
WHERE status = 'Gold'

```

第 1 行指定在结果集中包含顾客姓名和 XMLQUERY 函数的输出。第 2 行指出“Client”元素的第一个“email”子元素将被返回。第三行指定 XML 数据源（“contact”列）。第四行告诉我们该列来自“clients”表；第五行指出查询条件是金牌顾客。

例五

此例再一次演示了使用 XQuery FLWOR 表达式的 XMLQUERY 函数的用法，但请注意这一次返回的不仅是 XML，而且还有 HTML。

```

SELECT xmlquery('for $e in $c/Client/email[1]/text()
  return <p>{$e}</p>'
  passing contact as "c")
FROM clients
WHERE status = 'Gold'

```

XQuery 的 return 语句确保 XML 输出按需转换。第一行使用 text() 函数仅获取金牌顾客的第一个邮件地址的元素值部分。第二行把这个信息放在了 HTML 段标签中。

例六

这个例子使用 XMLEMENT 函数创建一系列 XML item 元素，每一项都包含 ID、brand name 和 stock keeping unit(SKU)三个子元素，它们的值由“items”表相应列获取。基本上，当把关系数据转化成 XML 数据时都可以使用 XMLEMENT 函数。

```

SELECT
xmlelement (name "item", itemname),
xmlelement (name "id", id),
xmlelement (name "brand", brandname),
xmlelement (name "sku", sku)
FROM items
WHERE srp < 100

```

上面的查询输出如下：

```
<item>
  <id>4272</id>
  <brand>Classy</brand>
  <sku>981140</sku>
</item>
...
<item>
  <id>1193</id>
  <brand>Natural</brand>
  <sku>557813</sku>
</item>
```

使用 XQuery 查询 XML 数据

前一节，我们叙述了如何使用带有 XML 扩展的 SQL 进行 XML 数据查询。一直以来 SQL 都是最主要的查询手段，XPath 也被嵌入其中。本节，我们讨论如何使用 XQuery 查询 XML 数据，XQuery 将会是主要的查询手段，一些例子也使用了嵌入 SQL 的 XQuery（使用“db2-fn:sqlquery”函数）。使用 XQuery 时，我们将会调用几个函数，同时使用 FLWOR 表达式。

例一

返回顾客联系方式的简单 XQuery 示例

```
xquery db2-fn:xmlcolumn('CLIENTS.CONTACT')
```

给 XQuery 表达式加上“xquery”命令前缀，这样 DB2 才能调用 XQuery 解析器，否则 DB2 假定您尝试运行的是 SQL 表达式。**db2-fn:xmlcolumn** 函数从由参数指定的列获取整个 XML 文档。这与下面的 SQL 语句等价，此时获取的是整列内容：

```
SELECT contact FROM clients
```

例二

本例使用 FLWOR 表达式获取客户传真数据。

```
xquery
for $y in db2-fn:xmlcolumn('CLIENTS.CONTACT')/Client/fax
return $y
```

第一行调用 XQuery 解析器。第二行告诉 DB2 在 CLIENTS.CONTACT 列包含的传真子元素中进行遍历。每一个传真元素都绑定到变量\$y。第三行返回每一次遍历的“\$y”值。

查询的输出与此类似（默认情况下输出可能会包含命名空间，但我们在此没有显示它，否则输出将会跨越多行而变得难以阅读）：

```
<fax>4081112222</fax>
<fax>5559998888</fax>
```

例三

这个例子查询 XML 数据并以 HTML 形式返回结果。

```
xquery
<ul> {
```

```

for $y in db2-fn:xmlcolumn('CLIENTS.CONTACT')/Client/Address
order by $y/zip
return <li>{$y}</li>
}
</ul>

```

示例返回的 HTML 如下所示：

```

<ul>
<li>
<address>
<street>9407 Los Gatos Blvd.</street>
<city>Los Gatos</city>
<state>ca</state>
<zip>95302</zip>
</address>
</li>
<li>
<address>
<street>4209 El Camino Real</street>
<city>Mountain View</city>
<state>CA</state>
<zip>95302</zip>
</address>
</li>
...
</ul>

```

例四

这个例子演示了如何使用 `db2-fn:sqlquery` 函数将 SQL 嵌入到 XQuery 中。`db2-fn:sqlquery` 函数执行 SQL 查询并返回选中的 XML 数据。传递给 `db2-fn:sqlquery` 函数的 SQL 查询必须只返回 XML 数据。返回的 XML 数据可以被 XQuery 进一步处理。

```

xquery
for $y in
db2-fn:sqlquery(
  'select comments from items where srp > 100'
)/Comments/Comment
where $y/ResponseRequested='Yes'
return (
  <action>
    {$y/ProductID
     $y/CustomerID
     $y/Message}
  </action>
)

```

示例中，SQL 查询根据条件“srp”列的值大于 100 过滤行。在经过过滤的行中，XML 列“comments”会被选出来。下一行 XQuery（或者说 XPath）用来遍历 Comment 子元素。

注意：

DB2 不区分大小写，它将所有表名、列名都当作大写来处理，而 XQuery 是大小写敏感的。示例中使用的都是 XQuery 接口函数，因此传递给函数的表名和列名都应该是大写的。传递小写的对象名可能会导致一个“未定义的对象名”错误。

15.3.6 使用 SQL/XML 执行联合操作

这一节讨论如何在不同表的两个 XML 列之间或者 XML 列与关系数据列之间执行联合操作。假定已用如下语句创建了两张表：

```
CREATE TABLE dept (unitID CHAR(8), deptdoc XML)

CREATE TABLE unit (unitID CHAR(8) primary key not null,
    name CHAR(20),
    manager VARCHAR(20),
    ...
)
```

可以选择如下两种方法之一执行联合操作：

方法一：

```
SELECT u.unitID
FROM dept d, unit u
WHERE XMLEXISTS (
    '$e//employee[name = $m]'
    passing d.deptdoc as "e", u.manager as "m")
```

name 是表“dept”中“deptdoc”列 XML 文档的子元素，同时 manager 是表 unit 的列，语句在 dept 表和 unit 表上执行了联合操作。

方法二：

```
SELECT u.unitID
FROM dept d, unit u
WHERE u.manager = XMLCAST(
    XMLQUERY('$e//employee/name'
        passing d.deptdoc as "e")
    AS char(20))
```

在这个方法中，关系列位于联合操作表达式的左侧。此情形允许使用关系索引替代 XML 索引。

15.3.7 使用 XQuery 执行联合操作

假定已经创建了两张表：

```
CREATE TABLE dept(unitID CHAR(8), deptdoc XML)
CREATE TABLE project(projectDoc XML)
```

如果使用 SQL/XML，查询如下所示：

```
SELECT XMLQUERY (
    '$d/dept/employee' passing d.deptdoc as "d")
FROM dept d, project p
WHERE XMLEXISTS (
    '$e/dept[@deptID=$p/project/deptID]'

    passing d.deptdoc as "e", p.projectDoc as "p")
```

使用 XQuery 的相同查询如下：

```
xquery
for $dept in db2-fn:xmlcolumn("DEPT.DEPTDOC")/dept
for $proj in db2-fn:xmlcolumn("PROJECT.PROJECTDOC")/project
where $dept/@deptID = $proj/deptID
return $dept/employee
```

第二种方法更容易解释：变量“\$dept”保存了“dept”表中“deptdoc”列的 XML 文档。变量“\$proj”保存了“project”表中“projectdoc”列的 XML 文档。语句第四行在第一个 XML 文档的属性与第二个 XML 文档的元素间执行联合操作。

15.3.8 更新与删除操作

有两种方式可以对 XML 数据执行更新和删除操作：

- 使用 SQL UPDATE 和 DELETE 语句
- 使用 TRANSFORM 表达式

第一种方式，更新和删除是文档级的，即整个 XML 文档都被更新的文档替换。例如，在下面的例子中我们想改变的仅仅是<state>元素，然而实际上整个文档已经被替换。

```
UPDATE clients SET contact=
  xmpparse(document
    '<Client>
      <address>
        <street>5401 Julio ave.</street>
        <city>San Jose</city>
        <state>CA</state>
        <zip>95116</zip>
      </address>
      <phone>
        <work>4084633000</work>
        <home>4081111111</home>
        <cell>4082222222</cell>
      </phone>
      <fax>4087776666</fax>
      <email>newemail@someplace.com</email>
    </Client>')
  )
WHERE id = 3227
```

第二种方式用 TRANSFORM 表达式执行子文档更新，效率高很多。这种方法可以替换、插入、删除或者重命名 XML 文档节点。同样也可以更改节点值而不用替换节点本身，典型的应用便是更改元素或者属性值，这类更新相当常用。这是 DB2 9.5 新添加的特性。

TRANSFORM 表达式是 XQuery 语言的一部分，通常可以在任何使用 XQuery 的地方使用它，譬如在 FLWOR 表达式或者 SQL/XML 语句的 XMLQUERY 函数中。当需要修改 XML 列中的一个 XML 文档时，将 TRANSFORM 表达式用在 SQL UPDATE 语句中，这是最典型的应用。

TRANSFORM 表达式的语法如下：

>>-transform--| copy clause |--| modify clause |--| return clause |-><

copy clause

```

-----.
|--copy---$VariableName--:---CopySourceExpression+-----|
modify clause
|--modify--ModifyExpression-----|
return clause
|--return--ReturnExpression-----|

```

Copy 语句用来把要处理的 XML 文档赋值给变量。在 **modify** 语句中，可以使用 **insert**, **delete**, **rename** 或者 **replace** 表达式执行操作。这些表达式可以对 XML 文档执行更新。例如，使用 **insert** 表达式向文档添加新的节点，使用 **delete** 表达式删除节点，使用 **rename** 表达式重命名元素或者属性，使用 **replace** 表达式把已有节点替换为新的节点。表达式的替换值只能用以更改元素或者属性值。**Return** 语句返回 **transform** 表达式的值。

这是一条 UPDATE 语句，语句使用了 **tranform** 表达式。

```

(1)-- UPDATE customers
(2)-- SET contactinfo = xmlquery( 'declare default element namespace
(3)--           "http://posample.org";
(4)--   transform
(5)--   copy $newinfo := $c
(6)--     modify do insert <email2>my2email.gm.com</email2>
(7)--       as last into $newinfo/customerinfo
(8)--   return $newinfo' passing contactinfo as "c")
(9)-- WHERE id = 100

```

在上面的例子中，第(1), (2), (9)行使用的依然是 SQL UPDATE 语句的语法。第(2)行的 XMLQUERY 函数调用第(4)行的 **transform** 表达式。(4)至(8)行是 **transform** 表达式，它向 XML 文档插入一个新节点 **email2**。注意 DB2 不支持通过视图更新 XML 文档的元素。

从数据表删除整个 XML 文档一如在 SQL/XML 中使用 **SELECT** 语句一样简单。使用 SQL **DELETE** 语句同时指定需要的条件即可。

15.3.9 XML 索引

在 XML 文档中，可以为元素、属性或者值（文本节点）创建索引。示例如下，假定已经创建下面的表：

```
CREATE TABLE customer(info XML)
```

同时假定这是存储的 XML 文档之一：

```

<customerinfo Cid="1004">
  <name>Matt Foreman</name>
  <addr country="Canada">
    <street>1596 Baseline</street>
    <city>Toronto</city>
    <state>Ontario</state>
    <pcode>M3Z-5H9</pcode>
  </addr>
  <phone type="work">905-555-4789</phone>

```

```

<phone type="home">416-555-3376</phone>
<assistant>
  <name>Peter Smith</name>
  <phone type="home">416-555-3426</phone>
</assistant>
</customerinfo>

```

- 1) 这条语句为属性“Cid”创建索引

```

CREATE UNIQUE INDEX idx1 ON customer(info)
  GENERATE KEY USING
    xmppattern '/customerinfo/@Cid'
    AS sql DOUBLE

```

- 2) 这条语句为“name”元素创建索引

```

CREATE INDEX idx2 ON customer(info)
  GENERATE KEY USING
    xmppattern '/customerinfo/name'
    AS sql VARCHAR(40)

```

- 3) 这条语句为所有“name”元素创建索引

```

CREATE INDEX idx3 ON customer(info)
  GENERATE KEY USING
    xmppattern '//name'
    AS sql VARCHAR(40);

```

- 4) 这条语句为所有文本节点（值）创建索引。这种做法并不提倡，因为在 update, delete 和 insert 操作中维护索引代价高昂，同时索引也未免太大了。

```

CREATE INDEX idx4 ON customer(info)
  GENERATE KEY USING
    xmppattern '//text()'
    AS sql VARCHAR(40);

```

15.4 使用 XML Schema

DB2 允许您插入一个正确的 XML 文档到数据库中。如果它是错误的，您会在插入时收到一个错误。另一方面，DB2 不强迫您验证 XML 文档。如果您想验证一个 XML 文档，您有多种选择，这是我们这一节将讨论的。

15.4.1 注册您的 XML Schema

XML Schema 存储在 DB2 数据库的所谓 XML Schema 仓库中。您可以使用 REGISTER XMLSCHEMA 命令来添加一个 XML Schema 到一个库中。

例如，假设您有一个 XML 文档存储在文件 order.xml 中，如图 15.10 所示。

```

<?xml version="1.0" encoding="UTF-8"?>
<po:PurchaseOrder xmlns:po="http://www.test.com/po">
    <Header>
        <Id>1</Id>
        <date>2004-01-29</date>
        <description>purchase order</description>
        <value>20</value>
        <status>shipped</status>
    </Header>
    <Items>
        <Item>
            <ItemDescription color="red" weight="5">
                <Name>Widget C</Name>
                <SKU>1</SKU>
                <Price>30</Price>
                <Comment>no comment</Comment>
            </ItemDescription>
            <NumberOrdered>1</NumberOrdered>
        </Item>
    </Items>
    <Customer type="regular">
        <Name>Manoj K Sardana</Name>
        <Address>ring road, bangalore</Address>
        <Phone>918051055109</Phone>
        <email>msardana@in.ibm.com</email>
    </Customer>
</po:PurchaseOrder>

```

图 15.10 – 包含一个 XML 文档的 order.xml 文件

现在假设您有一个 XML Schema 文档存储在文件 order.xsd 中，如图 15.11 所示。

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.test.com/po"
    xmlns:po="http://www.test.com/po"
    xmlns:head="http://www.test.com/header"
    xmlns:prod = "http://www.test.com/product"
    xmlns:cust = "http://www.test.com/customer">

    <xsd:import namespace="http://www.test.com/product" schemaLocation="product.xsd" />
    <xsd:import namespace="http://www.test.com/customer" schemaLocation="customer.xsd" />
    <xsd:import namespace="http://www.test.com/header" schemaLocation="header.xsd" />
    <xsd:complexType name="itemType">
        <xsd:sequence>
            <xsd:element name="Item" minOccurs="1" maxOccurs="unbounded" >
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="ItemDescription" type="prod:prodType" />
                        <xsd:element name="NumberOrdered" type="xsd:integer" />
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="potype">
        <xsd:sequence>
            <xsd:element name="Header" type="head:headerType" />
            <xsd:element name="Items" type="po:itemType" />
            <xsd:element name="Customer" type="cust:customerType" />
        </xsd:sequence>
    </xsd:complexType>

    <xsd:element name="PurchaseOrder" type="po:potype" />
</xsd:schema>

```

图 15.11 – 包含一个 XML Schema 的 order.xsd 文件

在这个 XML Schema 文档中，对标记椭圆处我们强调如下：

<xsd:schema ...>: 表明它是一个 XML Schema 文档。

<xsd:import ...>: 我们导入其他 xsd 文件（其他 XML Schema），将成为这一更大的 XML Schema 的一部分。

minOccurs="1": 这是 XML Schema “规则”的一个例子，Item 元素应该出现至少一次，或换言之，应该至少有一个 Item 项目元素。

其次，为了把 XML Schema 注册到数据库，可以使用一个与列表 15.33 类似的脚本。该脚本包含注释，使人不言自明。

```
-- CONNECT TO THE DATABASE
CONNECT TO SAMPLE;
-- REGISTER THE MAIN XML SCHEMA
REGISTER XMLSCHEMA http://www.test.com/order FROM D:\example3\order.xsd AS order;
-- ADD XML SCHEMA DOCUMENT TO MAIN SCHEMA
ADD XMLSCHEMA DOCUMENT TO order ADD http://www.test.com/header FROM
D:\example3\header.xsd;
-- ADD XML SCHEMA DOCUMENT TO MAIN SCHEMA
ADD XMLSCHEMA DOCUMENT TO order ADD http://www.test.com/product FROM
D:\example3\product.xsd;
-- ADD XML SCHEMA DOCUMENT TO MAIN SCHEMA
ADD XMLSCHEMA DOCUMENT TO order ADD http://www.test.com/customer FROM
D:\example3\customer.xsd;
-- COMPLETE THE SCHEMA REGISTRATION
COMPLETE XMLSCHEMA order;
```

列表 15.33 – 显示注册 XML Schema 步骤的示例脚本

以后想要查看此信息，您可以从目录表 SELECT 信息，如列表 15.34 所示。

```
SELECT CAST(OBJECTSCHEMA AS VARCHAR(15)), CAST(OBJECTNAME AS VARCHAR(15))
FROM syscat.xsrobjects
WHERE OBJECTNAME='ORDER';
```

列表 15.34 – 从 DB2 目录表检索 XML Schema 信息**15.4.2 XML Schema 验证**

一旦您的 XML Schemas 在 DB2 注册，您可以通过两种方式验证您的 XML 文档：

- 在 INSERT 过程中使用 XMLVALIDATE 函数
- 使用 BEFORE 触发器

图 15.12 显示了根据如图 15.11 所示的 XML Schema 验证如图 15.10 所示的 XML 文档。

```

DROP TABLE t1;
CREATE TABLE t1 (po XML);

INSERT INTO t1 VALUES(xmlvalidate(xmlparse(document('<?xml version="1.0" encoding="UTF-8"?>
<po:PurchaseOrder xmlns:po="http://www.test.com/po">
<Header>
<Id>1</Id>
<date>2004-01-29</date>
<description>purchase order</description>
<value>20</value>
<status>shipped</status>
</Header>
<Items>
<Item>
<ItemDescription color="red" weight="5">
<Name>Widget C</Name>
<SKU>1</SKU>
<Price>30</Price>
<Comment>no comment</Comment>
</ItemDescription>
<NumberOrdered>1</NumberOrdered>
</Item>
</Items>
<Customer type="regular">
<Name>Manoj K Sardana</Name>
<Address>ring road, bangalore</Address>
<Phone>918051055109</Phone>
<email>msardana@in.ibm.com</email>
</Customer>
</po:PurchaseOrder>')) ACCORDING TO XMLSCHEMA ID order));

```

图 15.12 – 使用 XMLVALIDATE 验证 XML Schema

- 要测试一个 XML 文档是否已通过验证，您可以在 CHECK 约束中使用 “IS VALIDATED” 谓词。
- 您可以使用不同的 XML Schema 验证一个列中的 XML 文档。这对于 XML Schema 从版本 1 到版本 2 的轻松迁移是重要的。在同一个 XML 列中，您也可能会发现没有验证的 XML 文档。这是有用的，如果收到的文档有些来自受信任的来源和有些来自非信任的来源，那就只有后者需要模式验证。

15.4.3 其他 XML 支持

小型 XML 文档现在可以与基表统一处理。这意味着 XML 数据与关系数据存储在同一个地方，并且与常规关系数据利用相同的压缩机制。大型的 XML 文档存储在一个单独的内部对象中，也可以被压缩。

DB2 还支持 XML Schema 进化。这意味着如果您的 XML Schema 更改了，您可以使用 UPDATE XMLSCHEMA 命令方便地更新 XML Schema。如果 XML Schema 的更改太大了，您可能会得到错误信息。

DB2 也支持 XML 的分解或“切碎”。这是在数据库中存储 XML 的“旧”方法，也是其他厂商用来存储 XML 的方法。如果您想使用它，DB2 中仍然支持这个方法，但我们建议 pureXML。DB2 还支持 XML Extender，也是使用旧方法来存储 XML，但此 Extender 不再改版了。

new in
V9.7

DB29.7 中 pureXML 的所有好处已经扩展到通常用于数据仓库数据库分区。数据库分区功能 (DPF) 在 DB2 企业版里提供。

15.6 小结

本章向您介绍了 XML 和 pureXML 技术。由于 Web2.0 工具和技术以及 SOA, XML 文档的使用率成倍增长。通过存储在 DB2 数据库中的 XML 文档, 您使用 pureXML 可以利用安全性, 性能和编码的灵活性。pureXML 是这样一种技术, 它允许您在例如树状结构的解析分层格式中存储文档, 这是在数据库插入时完成的。在查询时, 没有必要在处理前解析 XML 文档来建立一棵树。XML 的文档树已经建成并存储在数据库中。此外, pureXML 技术采用了理解 XQuery 的原生 XML 引擎; 因此, 没有映射 Xquery 到 SQL 的需要, 而其他 RDBMS 产品是映射 Xquery 到 SQL 的。

本章还谈到了如何插入, 删除, 更新和使用 SQL/XML 和 Xquery 查询 XML 文档。还讨论了 XML 索引, XML Schema, 以及如压缩和 XML Schema 进化的其他特征。

15.7 实验

在本章中, 您已经看到为数不少的 SQL/XML 和 XQuery 语法示例, DB2 命令编辑器和 IBM DataStudio 也已经做过介绍。本实验中, 您可以测试您对 SQL/XML 和 XQuery 掌握的程度, 同时获得这些工具的动手经验。您将使用 “mydb” 数据库, 这个数据库是用本章前面介绍的 table_creation.txt 脚本文件创建的。

实验过程

1. 创建 “mydb” 数据库并载入 XML 数据, 像在前面的章节里讨论的一样, “table_creation.txt” 文件包含在附带压缩文件 “Expressc_book_exercises_9.7.zip” 第二章的文件夹中。从 DB2 的命令窗口中或 Linux shell 下按如下的方法运行脚本文件 table_creation.txt:

```
db2 -tvf table_creation.txt
```

2. 如果脚本在任何一个步骤运行失败, 请试着从错误信息中找出问题所在。一个典型的问题是当运行脚本时, 您也许需要改变文件的路径, 因为它们也许在不同的目录下。您可以在 DB2 命令行窗口或者 Linux shell 中使用以下的命令删除并重启数据库:

```
db2 drop database mydb
```

3. 如果您因为存在活跃的连接, 而在停止数据库的过程中遇到错误, 请首先输入以下的命令:

```
db2 force applications all
```

4. 成功运行脚本之后, 使用 DB2 控制中心或者 IBM DataStudio 确认 items 表和 clients 表已经创建, 并且分别含有 4 行和 7 行。

5. 在创建了 “mydb” 数据库并载入了两个表之后, 您就可以对它连接, 并执行从列表 15.7 到列表 15.19 所示的查询了。

A

附录 A — 排除故障

此附录要讨论的是如何检测并维护在使用 DB2 中可能遇到的问题。图 A.1 展示了发生问题时通常采取的措施。

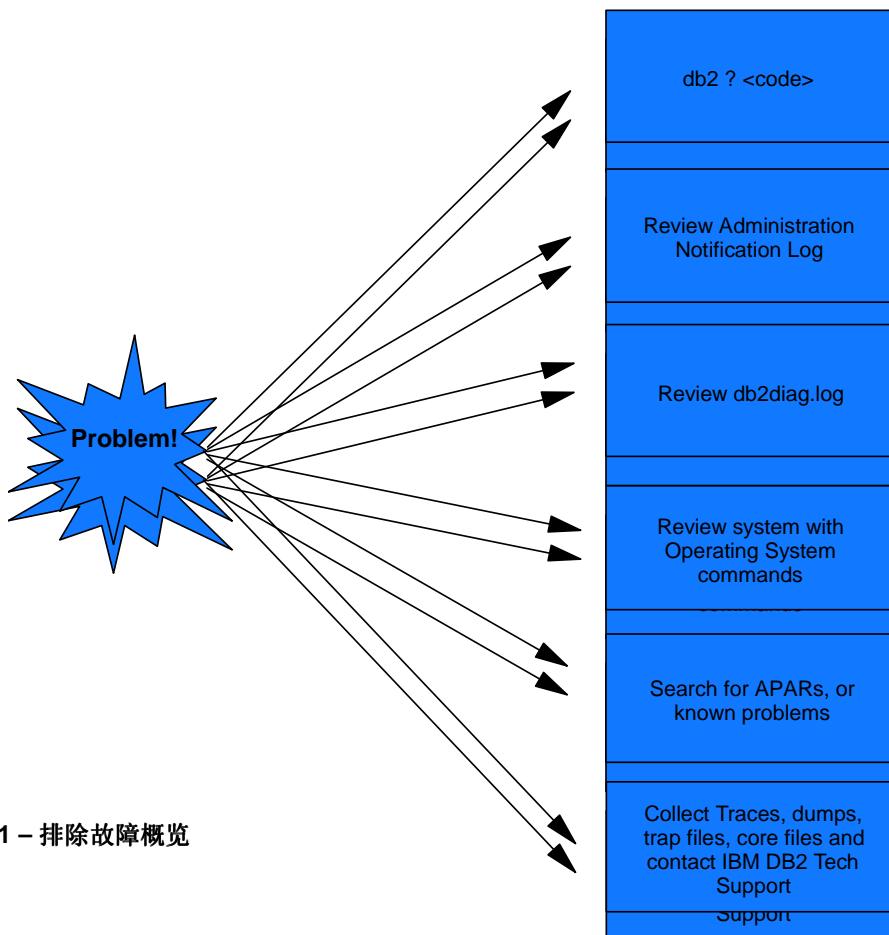


图 A.1 – 排除故障概览

注意:

更多有关排除故障的信息，请参见以下视频：

<http://www.channeldb2.com/video/video/show?id=807741:Video:4462>

A.1 查找错误代码的更多信息

想了获取更多有关故障代码的信息，在命令编辑器处输入该代码，并在前面加上问号。如图 A.2 所示

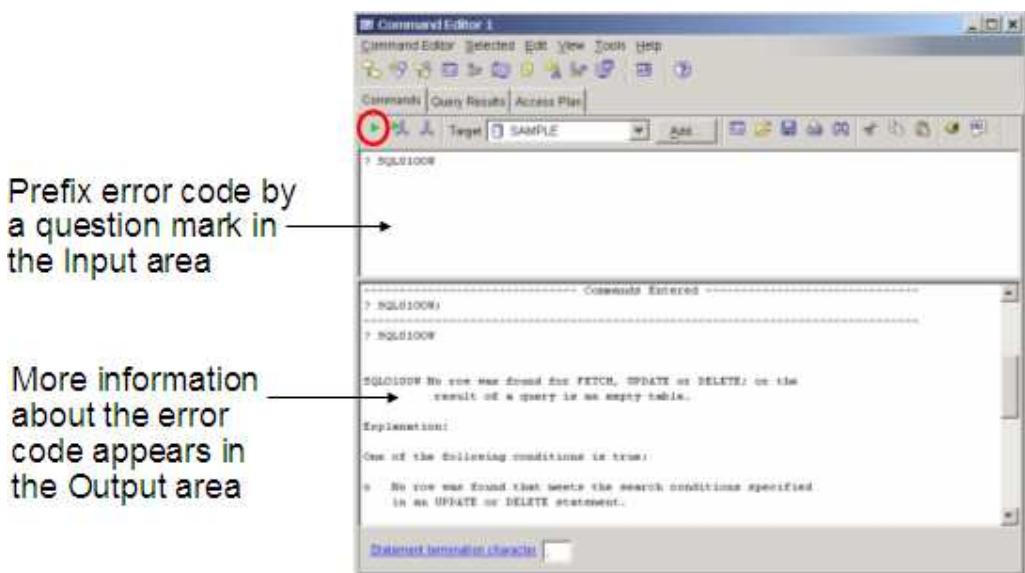


图 A.2 – 查找 DB2 错误代码的信息

问号（?）会调用 DB2 的帮助命令。以下是一些调用它来获取帮助的例子。例如：您想知道 SQL 错误代码“-104”的信息，可使用如下语句查询（以下例子都是等价的）：

```
db2 ? SQL0104N
db2 ? SQL104N
db2 ? SQL-0104
db2 ? SQL-104
db2 ? SQL-104N
```

A.2 SQLCODE 与 SQLSTATE

SQLCODE 是在每一条 SQL 语句执行后收到的代码。这些值的意义如下：

SQLCODE = 0; the command was successful 该命令执行成功。

SQLCODE > 0; the command was successful, but returned a warning 该命令执行成功，但返回了一个警告。

SQLCODE < 0; the command was unsuccessful and returned an error 该命令没有成功执行，并且返回了一个错误。

SQLSTATE 是一个遵守 ISO/ANSI SQL92 标准的长为 5 个字符的字符串。前两个字符就是 SQLSTATE 类代码：

- 类代码 00 表示该命令执行成功。
- 类代码 01 意味着有一个警告。
- 类代码 02 意味着有未检测到的情况。
- 其它的类代码都被认为是错误。

A.3 DB2 管理通知日志

DB2 管理通知日志提供了关于故障点的错误诊断信息。在 Linux/UNIX 平台上，管理通知日志是一个名为<实例名>.nfy 的文本文件（如“db2inst.nfy”）。在 Windows 平台，所有的管理通知消息都被写入 Windows 事件日志（Windows Event Log）。

DBM 配置参数 **notifylevel** 允许管理员指定要记录的信息的级别:

- 0 --不捕获任何管理通知消息
- 1 --致命的或不可恢复的错误
- 2 --要求立即采取措施
- 3 --重要信息，不要求立即响应（默认级别）
- 4 --普通消息

A.4 db2diag.log

db2diag.log 提供了比 DB2 管理通知日志更多的信息。它通常由 IBM DB2 技术支持或者经验丰富的数据库管理员使用。在 **db2diag.log** 中包含了如下内容:

- 出现错误的 DB2 代码的位置
- 应用程序标识符，它能让您将事件记录与服务器端和客户端的应用程序相对应
- 诊断信息（以“DIA”开头），提供了对错误的解释说明
- 其它的支持信息，例如 SQLCA 数据结构以及指向外部堆或者异常捕捉文件的指针

在 Windows 系统中，**db2diag.log** 默认保存在下面的路径:

```
C:\Program Files\IBM\sqlllib\<instance name>\db2diag.log
```

在 Linux/UNIX 系统中，**db2diag.log** 默认保存在下面的路径:

```
/home/<instance_owner>/sqlllib/db2dump/db2diag.log
```

诊断信息文字的长短和详细程度由 **dbm cfg** 的 **DIALEVEL** 参数决定，参数值可取 0~4，0 表示最短信息记录，而 4 表示最长最详细的信息记录。默认等级为 3。

A.5 CLI 追踪

对于 CLI 和 Java 应用程序，您可以打开 CLI 追踪来捕获和解决故障。在承载应用程序的服务器端修改 **db2cli.ini** 文件可以打开 CLI 追踪。典型的 **db2cli.ini** 文件如下所示:

```
[common]
trace=0
tracerefreshinterval=300
tracepathname=/path/to/writeable/directory
traceflush=1
```

当然也可以使用底层的追踪 (**db2trc**)，但是通常这只对 DB2 技术支持人员有用。

A.6 DB2 缺陷与补丁

有时候您遇到的问题可能是由 DB2 的缺陷引起，IBM 会定期发布补丁包来修复这些缺陷 (APARs)。补丁包的文档描述了对应补丁包所修复的缺陷。在开发新应用时，我们建议您安装好最新的补丁包。您可以用两种方式查看当前的版本和补丁包信息：1、在控制中心中点击“Help”菜单，然后选择“About”项；2、在命令行窗口输入“**db2level**”命令。注意：要获得 DB2 Express-C 的补丁包和 IBM DB2 技术支持，必须购买 12 个月使用许可证。



附录 B — 参考文献和其他资源

B.1 参考文献

- [1] ZIKOPOULOS, P. IBM® DB2® Universal Database™ and the Microsoft® Excel Application Developer... for Beginners, dbazine.com article, April 2005 <http://www.dbazine.com/db2/db2-disarticles/zikopoulos15>
- [2] ZIKOPOULOS, P. DB2 9 and Microsoft Access 2007 Part 1: Getting the Data..., Database Journal article, May 2008 <http://www.databasejournal.com/features/db2/article.php/3741221>
- [3] BHOGAL, K. Use Microsoft Access to interact with your DB2 data, developerWorks article, May 2006. <http://www.ibm.com/developerworks/db2/library/techarticle/dm-0605bhogal/>
- [4] SARACCO, C. et all. IBM Redbook DB2 9: pureXML overview and fast start July 2006. <http://www.redbooks.ibm.com/abstracts/sg247298.html>

B.2 网站

1. DB2 Express-C 网站:
www.ibm.com/db2/express
可以从该网站上面下载 DB2 Express-C servers、DB2 clients、DB2 drivers、用户手册，也可以查阅开发团队 blog，注册邮件列表等等。
2. DB2Express 论坛:
www.ibm.com/developerworks/forums/dw_forum.jsp?forum=805&cat=19
当您在用户手册上找不到答案时，请使用该论坛来张贴技术问题。
3. DB2 信息中心（DB2 Information Center）
<http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp>
该信息中心提供在线访问手册。这是最及时的信息来源。对于每个 DB2 版本有一个相应的 DB2 信息中心：
 - DB2 9.1: <http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp>
 - DB2 9.5: <http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp>
 - DB2 9.7: <http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp>
4. developerWorks
<http://www-128.ibm.com/developerworks/db2>
这个网站免费为开发人员和数据库管理员提供优质的资源，如技术文章、使用指南等等。
5. alphaWorks
<http://www.alphaworks.ibm.com/>
该网站提供了直接访问 IBM 的新兴技术的途径。在这里人们可以找到来自 IBM 研究的最新技术。
6. planetDB2

www.planetDB2.com

这里收集了大量贡献者关于 DB2 的博客文章。

7. DB2 技术支持 (DB2 Technical Support)

http://www.ibm.com/software/data/db2/support/db2_9/

您可以在上面寻找缺陷、问题的报告 (APAR) 和其他技术上的资料。

8. ChannelDB2

ChannelDB2 是 DB2 社区的社会网络，它提供了与 DB2 相关的视频、demo、博客、播客、讨论、其它资源等。涵盖的范围包括 Linux、UNIX、Windows、z/OS、i5/OS 等方面。

<http://www.ChannelDB2.com/>

B.3 书籍

1. 免费红皮书：《DB2 Express-C: The Developer Handbook for XML, PHP, C/C++, Java, and .NET》
Whei-Jen Chen, John Chun, Naomi Ngan, Rakesh Ranjan, Manoj K. Sardana,
August 2006 - SG24-7301-00
<http://www.redbooks.ibm.com/abstracts/sg247301.html?Open>
2. 免费红皮书：《DB2 pureXML Guide》
Whei-Jen Chen, Art Sammartino, Dobromir Goutev, Felicity Hendricks, Ippei Komi,
Ming-Pang Wei, Rav Ahuja, Matthias Nicola. August 2007
<http://www.redbooks.ibm.com/abstracts/sg247315.html?Open>
3. 红皮书：《Developing PHP Applications for IBM Data Servers》
Whei-Jen Chen, Holger Kirstein, Daniel Krook, Kiran H Nair, Piotr Pietrzak
May 2006 - SG24-7218-00
<http://www.redbooks.ibm.com/abstracts/sg247218.html?Open>
4. 《Understanding DB2 – Learning Visually with Examples V9.5》
Raul F. Chong, et all. January 2008
ISBN-10: 0131580183
5. 《DB2® SQL PL: Essential Guide for DB2® UDB on Linux™, UNIX®, Windows™, i5/OSTM, and z/OS®》第二版
Zamil Janmohamed, Clara Liu, Drew Bradstock, Raul Chong, Michael Gao,
Fraser McArthur, Paul Yip
6. 《DB2 9: pureXML overview and fast start》
Cynthia M. Saracco, Don Chamberlin, Rav Ahuja June 2006 SG24-7298
<http://www.redbooks.ibm.com/abstracts/sg247298.html?Open>
7. 《Information on Demand - Introduction to DB2 9 New Features》
Paul Zikopoulos, George Baklarz, Chris Eaton, Leon Katsnelson
ISBN-10: 0071487832
ISBN-13: 978-0071487832

B.4 联系我们

DB2 Express-C 邮箱: db2x@ca.ibm.com
DB2 校园计划邮箱: db2univ@ca.ibm.com

DB2 9.7快速入门的捷径

本书的用途：

- 通过免费的**Express-C**版了解**DB2**
- 理解**DB2**体系结构、工具和安全性
- 学会管理**DB2**数据库
- 编写**SQL**、**XQuery**、存储过程
- 使用**DB2**开发数据库应用程序
- 亲身练习每个实验

IBM 公司的 DB2 Express-C 是一款免费的 DB2 数据服务器，它能够轻松管理关系型和 XML 数据。免费意味着 DB2 Express-C 可以免费下载、免费用来创建应用程序、免费配置到产品中、也可以随着您的解决方案免费嵌入和发放。DB2 没有数据库大小、数据库数量、用户数量的任何人为限制。

DB2 Express-C 能够运行在 Windows 和 Linux 系统上，并为包括 C/C++、Java、.NET、PHP、Perl 和 Ruby 在内的各种编程语言提供应用程序驱动。DB2 还提供低费用的订购和支持以满足额外的需求。如果您需要更大的扩展性或者更先进的功能，您可以无缝地将基于 DB2 Express-C 的应用程序配置到如企业版、工作组版的其它的 DB2 版本上。

免费 DB2 版本适合开发人员、技术顾问、独立软件开发商、数据库管理员以及其他想开发、测试、配置、分发数据库应用程序的人。请马上加入到不断壮大的 DB2 Express-C 社区，试用 DB2 Express-C，开始体验如何创建下一代应用程序以及创新的解决方案。

更多信息和下载 DB2 Express-C，请到：<http://ibm.com/db2/express>

与 DB2 社区互动、访问 DB2 视频和博客，请到 <http://channelDB2.com>



9 780986 628351

定 价： 24.99 美元