IBM原厂"正规军"出品

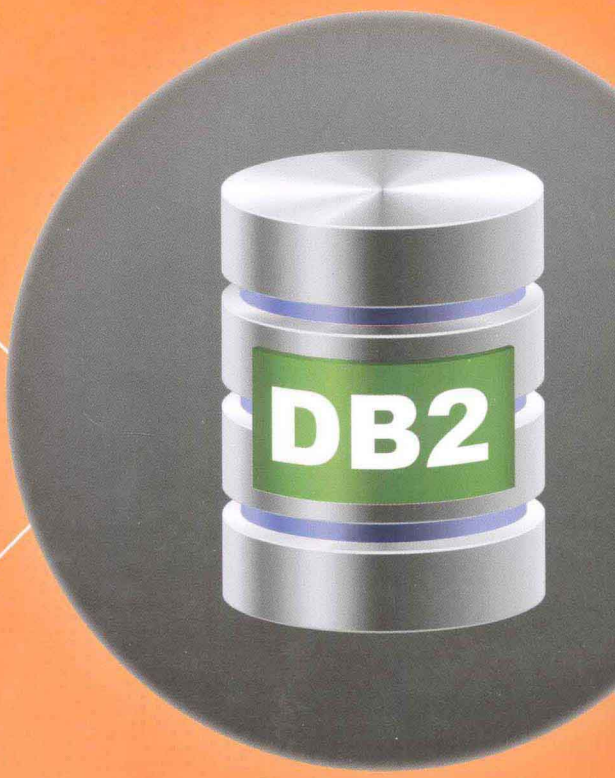
DB2

设计、管理与性能优化艺术

王飞鹏 李玉明 朱志辉 王富国 编著

本书特色：

- IBM原厂性能优化专著
- DB2 pureScale设计与优化专著
- DB2数据仓库设计与优化专著
- 涵盖DB2 HADR灾备最佳实践
- 涵盖DB2 V10最新设计与优化技术
- 将Oracle与DB2的对比贯穿全书
- 融汇金融、电信、政府等行业实战案例
- 表达新颖独特、内容深入浅出
- DB2从业人员的案头必备之作



清华大学出版社

DB2
设计、管理与性能优化艺术

王飞鹏 李玉明 朱志辉 王富国 编著

清华大学出版社
北 京

内 容 简 介

本书内容严谨精细、生动活泼，从内容来看，共分为四大部分，共11章。第一部分包括第1章和第2章，提出了两种性能优化方法学，包括理想化的自上而下方法学和救急专用的自下而上方法学，随后通过一个真实的实战案例，阐述了如何灵活运用方法学。第二部分是设计部分，包括第3章、第4章和第5章，分别谈到了物理设计、基础逻辑设计和高级逻辑设计，这是设计一个高质量的数据库系统所必须掌握的内容。第三部分是性能优化部分，包括第6章、第7章、第8章和第9章，讲述了如何对DB2进行性能监控，如何调整参数和优化维护工具，如何调整锁和日志来优化高并发系统，如何来优化最耗资源的SQL语句。第四部分是有关高级设计与优化内容，包括第10章和第11章，讲述了如何设计和优化大容量数据仓库，讲述了如何设计和优化DB2 pureScale集群。

图书在版编目(CIP)数据

DB2设计、管理与性能优化艺术 / 王飞鹏等编著. —北京：清华大学出版社，2013

ISBN 978-7-302-32661-8

Ⅰ. ①D… Ⅱ. ①王… Ⅲ. ①关系数据库系统—程序设计 Ⅳ. ①TP311.138

中国版本图书馆CIP数据核字(2013)第122415号

责任编辑：袁金敏
封面设计：陈晓兵
责任校对：胡伟民
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京嘉实印刷有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：33　　**字　　数**：845千字
版　　次：2013年9月第1版　　**印　　次**：2013年9月第1次印刷
印　　数：1～4000
定　　价：79.00元

产品编号：049972-01

序一

DB2 数据库是 IBM 公司数据管理产品线上最知名也是最成功的产品，不仅在大型主机市场处于绝对领导地位，而且在开放式平台的影响力更是与日俱增，不断地有客户从 Oracle 迁移到 DB2。

说起 DB2 的悠久历史，实际上它起源于一篇具有划时代意义的论文。1970 年，IBM 公司的研究员埃德加·考特(Edgar F. Codd)发表了名为“大型共享数据库的关系模型”（*A Relational Model of Data for Large Shared Data Banks*）的论文，从而正式拉开了关系型数据库的大幕。从那时起，IBM 公司就一直在数据库这个领域深耕细作，持续创新，通过不断研发优秀的产品屹立于强手如林的科技界，表现在 DB2 研发上更是不断的推陈出新。下面是 DB2 发展史上具有重要里程碑意义的版本，它们一经推出就获得了业界的广泛好评。

1983 年，DB2 正式问世，被 IBM 大型机所专用，之所以命名为 DB2，是由于以前已经有了一个著名的层次型数据库产品 Information Management System（IMS）。

1992 年，IBM 将 DB2 带向了多种开放式平台，包括 Linux、Unix 以及 Windows 服务器，也简称为 DB2 LUW。

2002 年，IBM 发布了 DB2 V8，这个版本开始支持数据库分区特性（Data Partitioning Feature，DPF），从而通过非分享（Share Nothing）的 MPP 架构为数据仓库提供更强的伸缩性。

2006 年，IBM 发布了 DB2 V9，这个全新的 DB2 是第一款“天然”存储 XML 的关系型数据库。

2008 年，IBM 发布了 DB2 V9.7，提供了 Oracle 兼容特性，这个版本完全支持 Oracle 语法，从而大大方便了应用和数据库迁移。

2009 年，IBM 发布了 DB2 V9.8，增加了 DB2 pureScale 特性，该特性利用了 z/OS 大型机上经过验证的 DB2 数据库集群技术，在开放平台上实现了共享磁盘（Share Disk）架构，以满足高吞吐量 OLTP 应用对高可用性、扩展性和负载均衡的需要。

2012 年 4 月，IBM 发布了 DB2 V10，该版本提供了多时态表、多表星型连接、行列访问控制（RCAC）、多温度存储、自适应压缩等特性。在这个版本中，DB2 pureScale 技术得到了进一步完善，从而获得了很多行业客户的高度认可，目前 DB2 pureScale 已经在中国金融、航空、烟草等行业的生产库上得到了成功实施。

2012 年 10 月，IBM 正式推出了数据库一体机 PureData。该产品不同于竞争对手的支持混合负载一体机，而是针对交易处理、分析和操作分析这三种应用场合分别提供不同的软硬件预配置解决方案。

时光荏苒，我加入 IBM 中国开发中心已经快 20 年了，在这里非常荣幸见证了中国工程师 DB2 研发实力的快速提高。从刚开始接一些简单的开发工作，到现在参与 DB2 最新的核心技术

开发；从刚开始国外团队对中国工程师工作能力的半信半疑，到现在获得国外 DB2 团队的信任和认可，这说明中国开发中心在 IBM 全球的研发地位变得越来越重要了。当然，除了研发最新的 DB2 核心技术外，中国开发中心还作为培养中国 DB2 人才的“黄埔军校”，培养了一大批技术支持和服务客户的优秀工程师。

成绩是非常突出的，但是存在的一些问题也毋庸讳言。在数据库用户中，精通 Oracle 的人非常多，但是精通 DB2 开发、维护和优化的人就相对比较少了，这在一定程度上影响了 DB2 产品的使用。其实，说到底这是 DB2 生态系统的问题。作为原厂来说，IBM 公司是非常愿意通过各种方式和途径推广 DB2 产品技术的，例如通过和 db2china 社区的紧密合作，定期由来自中国研发中心的资深工程师在 db2china 社区上轮值来解答用户的棘手问题，中国开发中心也会和 db2china 社区定期举办 DB2 沙龙以更有效地传播 DB2 新技术，这些努力都取得了非常好的效果。

近几年来，我们还逐渐加大了对 IBM 客户和合作伙伴的支持力度。针对 DB2 客户，我们于 2011 年在大中华区成立了 DB2 客户顾问委员会，简称 DB2 CAC，目前来自银行、电信、保险、制造等不同行业的多个客户已经加入了这个委员会。这种方式加强了 DB2 实验室和客户之间的双向交流与合作，我们为客户安排了实验室顾问（Lab Advocate），向他们介绍 DB2 最新的技术发展，客户也可以把他们对 DB2 产品的需求提供给 DB2 开发团队，这种方式收到了非常显著的效果。另外，针对合作伙伴，我们今年成立了一个合作伙伴委员会（Business Partner Council）以帮助 IBM 重要合作伙伴在他们的业务系统中使用好 DB2。

实际上，除了 IBM 官方的积极推动外，类似《DB2 设计、管理与性能优化艺术》这样的 DB2 书籍也是一种非常有效的技术推广方式。这本书由来自 IBM 中国开发中心的 DB2 资深工程师编写，具备科学的性能优化知识体系，以大量实战案例为载体，不仅展示了 DB2 经典设计和优化方法学，还涵盖了 DB2 高级设计和优化技术，例如 DB2 HADR、DB2 pureScale 集群、数据仓库等。这对奋战在数据库应用开发与优化一线的 DB2 用户来说，可以说是找到了一把进入 DB2 技术圣殿的金钥匙，希望广大读者能从中获益。

IBM 中国开发中心 DB2 开发与客户支持总监

干毅民

2013 年 7 月 1 日

序二

去国离家十四载后，2009 年，当我再次从 IBM 多伦多实验室归来凝视上海这座不夜城时，不由地感慨世界真的变小了、变平了，中国作为地球村的一员，正以惊人的速度发展变化着，而我的工作也一样在发展变化着。从加拿大回国后于 2010 年正式加入 IBM 中国软件开发实验室，由之前的 DB2 数据管理工作，转而开始从事数据治理（Data Governance）方面的工作。

过去的十年是企业的 IT 系统数据量高速膨胀的时期，“大数据”时代已悄然来临：每天，遍布世界各个角落的传感器、移动设备、在线交易和社交网络生成上百万兆字节的数据；每个月，人类发布 10 亿条 Twitter 信息和 300 亿条 Facebook 信息。据统计，全球 90% 的数据都是在过去两年中生成的。

“大数据”时代必然会产生新的“大数据”问题。哪些数据是可信的？哪些数据需要进行清洗？如何从海量数据中获得业务洞察力，从而指导商业决策？如何确保新录入的信息不会产生冗余？如何以可复用的方式发布可信任的信息？如何使如此庞大的数据真正变成对企业有价值的信息？这些海量的、分散在不同角落的数据带来了资源利用的复杂和管理的困难等问题。

以上所述各种问题，最终的解决办法就是要靠数据治理。（在很多时候我们用一个更为精确的概念——信息治理）数据治理是专注于将数据作为企业的商业资产进行应用和管理的一套管理机制。良好的数据治理能够消除数据的不一致性，提高组织数据质量，建立规范的数据应用标准，实现数据广泛共享，并能够将数据作为企业的宝贵资产应用于业务、管理、战略决策中，发挥数据资产的最大商业价值。

这几年，数据治理在国内的研究和应用都取得了一定进展，但是也面临观念上和实践上的双重挑战。从观念上看，国内很多人认为数据治理只是 IT 部门的责任，只把数据治理当成软件，并未真正意识到业务、数据和软件之间的关系，也就不能从整体上将数据作为企业资产来看待。从实践上看，很多企业做了数据质量检查，做了数据归档，做了数据安全，但缺乏一个完整的体系来将各个部分串联起来，也就是说，缺乏将这些领域组织起来的方法论。

为了帮助企业更好地管理数据资产，应用大数据时代信息治理的挑战，IBM 推出了全新的信息管理和业务分析产品，并提供技术资源，致力于为企业及机构提供大数据分析、信息整合、主数据管理等数据治理解决方案。以下都是基于 IBM 数据治理方案的优秀软件。

IBM InfoSphere BigInsights 是 IBM 大数据平台的核心产品之一，它是一款以 Hadoop 为基础的、对海量数据进行存储、管理和分析的企业级平台。可以在 30 分钟内安装完毕并投入运行，可用来管理企业各种数据，比如大量来自社交网络、移动设备和传感器等不同来源的非结构化数据，并对这些数据进行深度挖掘和分析。

IBM InfoSphere Information Server 系列软件支持将大数据作为来源和目标进行整合，并凭借其成熟可靠的性能和并行引擎，提供大数据所需的强大可扩展性，包括元数据管理、数据清洗、

数据质量治理及分析、数据自动化分析、数据抽取转换加载、数据集成、监控与报告等一整套软件组成的企业级信息平台。通过它可分析、清洗和整合异构源中的数据信息，并且把经过分析、清理后的可信任信息以可复用的方式提供给用户，同时也对新录入的信息进行实时的数据清洗操作，保证新录入信息的正确性。

IBM InfoSphere Master Data Management 软件，其最核心的任务是导出企业的关键业务数据，也是绝对真实的数据。主数据管理旨在从企业的多个业务系统中整合最核心的需要共享的数据，集中进行数据清洗，并以服务的方式把统一、完整、准确的主数据分发给企业内的操作型应用和分析型应用，包括业务系统、业务流程和决策支持系统等。使用户更深入地理解生产链条上的各个要素——客户、产品、供应商、员工等之间的关系，为进一步分析和决策做重要支撑。

IBM InfoSphere Stream 是 IBM 大数据平台中专门针对快速产生的如流水般不间断的海量数据流的处理平台。它是一个支持开发和部署的应用程序平台。能够持续快速地分析实时产生的各种各样的海量流数据。具有低延迟、实时响应、跨多个数据流进行分析的特点，尤其适用于对响应时间有较高要求的应用，例如欺诈检测、网络管理，能很好地解决企业处理数据量大、存储成本高的问题。通过直接对数据进行分析，无须存储，从而实现对有价值数据进行深入分析的可能。能够在大规模的集群环境中并行、高性能地处理流数据，并具有近似于线性的可扩展性，是帮助企业处理实时化的海量流数据分析的好帮手。

谈了不少数据治理，那么它和 DB2 数据管理有什么不同？我想分享一下我的观点。如果说从事数据质量管理、主数据管理等数据治理工作的人是数据巨轮的船长，他们平时的工作就是站在舰桥上，穿着带金色肩章的白制服，用双筒望远镜了望远方以把握方向，那么从事 DB2 相关工作的 DBA 就是在轮机舱工作的船员，船长和船员双方共同为数据巨轮的稳定运行发挥重要作用。但是，在实际工作中，我经常发现这样的事情：当舰桥上传来船长焦急的指令，命令 DBA 加快数据流动的速度时，DBA 由于缺乏驾驭 DB2 的优化技巧，只能回答说，“DB2 引擎遇到性能瓶颈了，船长！”。虽然船长的指令很及时，但遇到这样的船员，数据巨轮恐怕也运转不佳了。

DB2 船员的实际水平制约了船长对数据巨轮的驾驭能力。本书就是一本供 DB2 船员行驶的实战手册，希望广大船员能真正理解和掌握书中讲述的 DB2 设计与性能优化艺术，当收到船长要求加速数据流动的命令时，可以让 DB2 引擎运行得更好，这样数据巨轮就能畅通无阻得行驶下去。

IBM 中国开发中心信息管理产品开发部
洪桦 资深主管经理
2013 年 6 月 8 日

序三

IBM 百年华诞，在 2011 年夺目九界，璀璨全球。创新，这是 IBM 能屹立于强手如林的科技界的关键，使得她能够适应科技时代发展的需要，不断创新求变，从而把握时代的脉搏，解决今天以及未来企业遇到或可能遇到的重大挑战。

进入 21 世纪，IBM 与其众多对手们不约而同地投入到了信息时代的竞争。随着信息技术的迅猛发展，作为其核心组成部分的数据之战已成为了 21 世纪“竞争”的新内涵，而作为承载、处理和加工这些数据的数据库软件行业就不可避免地成为了主战场。

为了满足客户各种需求，大家竞争的对象是极富生命力的数据。在数据的整个生命历程中，它会经历设计、开发、部署、运营、优化和治理的不同阶段，并且这不是一次性的过程，而是通过周期性的迭代方式，来发挥数据更大的价值。任何一家企业拥有了对数据强大的管控和支配，它就会在 21 世纪数据之战中立于不败地位，甚至引领信息时代的发展。

IBM 正在着手于实现一个战略计划，提供一个集成的模块化数据管理环境，帮助企业更高效准确地管理整个数据生命周期（从需求到报废）。我们将这个过程称为“集成数据管理”，管理数据生命周期的每个环节，并能够支持各种主流厂商提供的数据管理技术，这包括 DB2、Informix、Oracle 等。

本文中提到的 IBM Optim Tools，就是应运而生的这样一个工具集，它除了提供对数据库基本的管理和开发功能外，还提供了强大的 DB2 监控和优化功能。IBM Optim Tools 最大的优势就在于对 DB2 数据库全面的支持，能够及时地反映并紧跟上 DB2 数据库的发展和更新，例如对 DB2 数据仓库和 DB2 pureScale 的支持。

本书的作者都有非常丰富的数据库管理和优化经验，使得本书具有极佳的实践性和可操作性，相信能为广大的数据库用户提供前所未有的帮助。

IBM 中国开发中心信息管理产品开发部
资深经理 孙冰江
2013 年 6 月 28 日写于北京

序四

我和飞鹏在几年前的一次 InfoSphere 培训中结识，基于对 DB2 的共同爱好，我们经常探讨关于 DB2 的各种技术，受益颇丰。如今飞鹏的舞动 DB2 系列专著已经陆续出版上市，作为已经阅读过舞动 DB2 系列书籍的读者，我想说，这套专著的理论水平毋庸置疑，实际案例更是经验和智慧的结晶，对于使用 DB2 的各个层次的读者都会有很大帮助。

这次，新书《DB2 设计与性能优化艺术》即将上市了，我受飞鹏之邀为本书作序，感到十分荣幸。

先讲讲我和数据库的故事。我从 1992 年开始使用数据库，最初是用 FOXBASE 开发了一个工资管理系统。1995 年开始基于 DB2 for AS400 的银行应用开发。1997 年开始基于 SYBASE 的证券应用开发。1998 年开始从一名开发者转向系统工程师，开始了中间件和数据库方面的技术支持。1999 年获得 DB2 V5 DBA 认证。1999 年~2006 年，先后做过 SYBASE、DB2、Oracle 数据库的规划设计、系统管理、故障排除、性能优化、开发指导等工作。2007 年开始主要关注和研究 IBM 软件，提供主流数据库、数据仓库、数据迁移、灾难备份的咨询及培训服务。到目前为止，我拥有 DB2 从版本 5 到版本 10 的 DBA 认证、DB2 版本 9 的高级 DBA 认证、Oracle 11G DBA OCP 认证、Informix 10 和 11 的 DBA 认证以及 IBM 的 InfoSphere 及 Tivoli 的各种认证 40 余个。目前担任 IBM 官方认证讲师，负责十多项产品的培训工作。

虽然我做过各种类型的数据库工作，但是数据库的设计与优化无疑是一项很重要且关键的工作。说起数据库优化，先给大家讲一个故事。华佗兄弟三人都精通医术，有一天一个人问华佗，你们家里兄弟三人谁的医术最高，这时华佗说：我们家大哥的医术最高明，其次是我的二哥，医术最差的就是我了。提问者十分不解地问：所有人都知道你是天下最有名、医术最高的人了，怎么还有比你医术更高明的人？于是华佗说了一段非常耐人寻味的话：“我大哥治病是在人们尚未察觉身体有病的时候为人们医治的，人们对他医术不甚了解。我二哥治病是在人们开始发病的时候通过望闻问切，开处方医治病人的，人们只是对他有所了解。我看病是在病人的病情非常严重的时候，才给病人下药，所以人们认为我能够让人起死回生，因此我最有名气，但是论医术水平我与我的兄长差距很大呀！”其实 DBA 就像是数据库的医生，数据库的性能优化也有三个阶段：第一个阶段是，数据库性能出现严重问题时去诊断分析、提出整改方案，即使能起死回生，但由于之前的规划设计缺陷很难达到持续稳定健康的运行；第二阶段是，在数据库运维过程及时发现问题解决问题，但同样也受限于之前的规划设计；第三个阶段是，在数据库规划设计阶段就开始考虑性能问题，这是性能优化的最高境界。所以设计与性能优化是分不开的，这也是本书所关注和阐述的。

通过十多年的数据库经验，强烈地感觉到数据库设计与性能优化需要了解数据库之外的很多相关知识，比如：操作系统、存储、网络、中间件，这些知识是相互关联和影响的，既而成

为一个既独立又统一的系统，理解和掌握的越多，在性能优化时考虑得越全面。DBA如同一个剑客，借用电影《英雄》的台词：“剑法有三种境界：第一种，手中有剑、心中却无剑，主要练就的是一招一式；第二，手中有剑、心中有剑，所谓人剑合一，练就的是剑气；第三，手中无剑、心中也无剑，是一种至大则空的平和。第三种被称为剑法的最高境界。” 第一种，手中有剑、心中却无剑：这种境界的DBA知道很多DB2的命令及用法，但对这些命令及用法的使用场景是否匹配合适则无所适从。第二种，手中有剑，心中有剑：知道很多命令及用法，也知道这些命令及用法的适用场景，也能做到活学活用。第三种，手中无剑、心中也无剑：不再使用现成的某某高手写的安装指南、运维指南等，不再仅关注于数据库，而更多地是利用DB2的信息中心，通过全面地系统感悟，做到IT系统为我服务，我来规划设计IT系统。到了这个境界，就不会纠结于要学习DB2还是Oracle，因为两者的很多东西都是相通的。希望本书能引导读者走向DBA的最高境界。

北京富通东方科技有限公司技术服务中心副总经理
张东焕
2013年7月15日写于北京

前 言

“三方演义”与性能优化

性能优化为什么这么难？

一个 IT 系统建设的过程，其实就是“三方演义”的过程。这里的三方，就是客户方、开发商和 IBM 原厂，通常是客户提需求，开发商负责应用设计和开发，IBM 提供软硬件产品，系统完成开发后，由客户或者第三方公司运行维护。一个 IT 系统的好坏，取决于这三方能否紧密合作，能否发挥各自独特的优势，从而实现 1+1+1>3 的效果。

类似于 IT 系统建设，一个上规模的性能优化项目，也是“三方演义”的互动过程。我经历过一些烂尾项目，这种项目在启动的时候，大家坐在一起指点江山，高谈政治，拍脑袋做决定，当出现性能问题的时候非常浮躁，不能冷静客观地分析问题和解决问题，而是叫喊着用更高档的硬件，缺乏严谨、务实、执着和求真的精神，缺乏对性能优化项目最起码的敬畏感。

首先是客户方。钱是客户出的，客户是真正的甲方，所以客户总是最强势，客户领导经常在会议上发号施令，当遇到问题的时候把开发商骂得一塌糊涂：“看看你们写的垃圾代码，还想让我们购买更好的硬件？”或者骂原厂：“你们说过 DB2 提供的自动维护特性不需要 DBA 过多干预，但为啥根本不是那么回事呢？”骂归骂，出了问题其实客户也是有责任的，你为啥不弯下身段，真正参与到系统的设计、开发和维护的整个过程呢？这样即使出了问题，心里也有底了呀。但是，现实中又很难，因为这和客户内部的体制有关，通常在一个项目的不同阶段，各个部门就像“铁路警察，各管一段”，开发部门只管开发，运维部门只负责上线维护，运维部门和开发部门是各自为战，当出了问题后，相互踢皮球，协调起来非常困难。例如，金融业的客户，一般拥有自己专门的开发部门和运维部门，两个部门独立，不存在上下级隶属关系，遇到困难后很容易相互扯皮。至于电信业和政府部门，出了系统性问题更难协调：电信业的客户，通常只负责系统维护工作，应用开发通常由第三方的开发商负责；政府客户，只负责需求、系统规划和管理工作，开发和维护都是由第三方公司承担的。针对客户方，我想强调的是，没有专业分工的局面是可怕的，但太过于精细的分工同样效率低下，更可怕的是很多企业采用政治挂帅的方式启动和管理项目，这是导致烂尾的重要根源之一。

接下来谈谈开发商。开发商是挨客户骂最多的了，但并不妨碍他们喜欢拍胸脯，啥都敢承诺，啥都敢做，毕竟有其独特优势。开发商有人力资源的优势，手中有大量的开发人员可供使用；开发商还有对业务理解的优势，毕竟开发商长期和行业客户在一起，积累了很多业务经验

和知识；通常开发商还有关系的优势，毕竟从客户那拿项目，是要和客户关系融洽、有信誉才行。但是，开发商也有技术硬伤，那就是对系统软件，包括硬件、数据库、中间件等的使用水平还停留在安装配置的层面，缺乏对其内部机制的了解，所以在架构规划、数据库优化、高可用性测试、可扩展性测试等方面技术上做不到甚至没有这方面的意识，更不可能实施了。

最后谈谈原厂。原厂的人员也会挨骂，也会被开发商拿来当挡箭牌，这不新鲜。原厂有自己的核心优势，那就是对自身产品的深入理解，以及跨金融、电信、政府等各行业应用所积累的实施优势，这必将在架构规划、数据库优化、扩展性测试等需要技术深度和经验的领域发挥重要作用，也可以为客户和开发人员在设计、开发和运维等工作上提供至关重要的技术支持。但是，原厂也有做的不够好的地方，其实也是体制上的原因，那就是以围绕产品的技术支持为主，如果在推动最新的技术和产品给客户的同时能愿意多倾听客户的真正需求，如果能积极主动参与到项目规划、设计、开发、测试和运维的整个阶段，那将对 IT 系统质量的提升发挥重要作用。

在这样一个“三方演义”的架构下，要想完成性能优化工作有时候超越了技术本身，它更多地需要设计开发人员和运维人员的紧密配合。比如，当我发现 SQL 语句的问题，需要开发人员支持和配合的时候，却被告知项目已经被开发商交付给客户了，没有办法找到开发人员了。有时候我即使找到了开发人员，他们却对 SQL 语句的最终运行环境一点都不关注，他们只是强调 SQL 语句是业务逻辑的需要。于是，我不得不找运维人员寻求支持，但是运维人员让我更吃惊，他们只关注主机、操作系统和数据库本身，对上层应用缺乏了解，也不关注上层 SQL 语句是怎么写的。

这可能是国内 IT 系统建设的一个缩影吧，所以在此我仅提出一条呼吁性质的建议：在客户的推动下，设计开发人员和运维人员定期举行技术交流，多交流硬件、数据库、中间件以及应用开发中遇到的性能痛点，拿出一套行之有效的办法在当前项目中试点并在其他项目中加以推广。

某银行性能优化的真实案例

2012 年 12 月 15 日，星期六，早上 8 点钟，本人衣冠楚楚，匆匆出门，打算去参加一个老同学的婚礼。刚上出租车，就接到了公司销售的电话，说是国内某银行客户的 DB2 数据库上线试运行后出现了非常严重的性能问题，希望我赶赴浙江宁波现场做一下性能调优工作。前几年如果周末接到这种电话，我内心肯定会抱怨一番的，周末了都不让人好好休息，不过经历的多了也就适应了，也慢慢练就了乱云飞渡仍从容的气度。老同学的婚礼该参加还是继续参加，该喝酒还是继续喝酒，该定去宁波的机票还是继续定。

我是 12 月 16 日晚上抵达宁波的，到了之后和客户领导郑总约了第二天也就是 12 月 17 日早上 9 点在现场召开会议。

第二天也就是星期一上午 8 点 50 分，我刚到了客户会议室门口，就听见一个人在满腔怒火地训斥，从声音可以听出来这个训人者就是昨天和我通电话的客户领导郑总。等我刚踏入会议室，发现场面非常隆重，里面坐了很多人，除了客户领导外，还有客户方 DBA 小李、第三方顾问公司的架构师老张、以及来自几家应用开发商的大队开发人员，可谓各路英豪齐聚一堂啊。作为原厂的唯一代表，本人孤独地坐在了角落里。会议几乎在争吵中度过，下面是我对大

家发言概要的整理。

郑总发言："这个项目是银监会重点督导的项目，具有重要的政治意义，如果这个项目在宁波试点成功了，那么可以推广到华东甚至是全国。但是，开发商在开发阶段就屡次拖延工期，这次上线试运行后，又暴露了严重的性能问题，业务人员没法使用，对此我是非常不满意的，需要拿出彻底的整改办法来。"

开发商代表发言："首先，接受领导的批判。但是，这个项目的业务需求改动了好几次，另外我们的队伍对 Weblogic 应用服务器和 Oracle 数据库非常熟悉，但是对 DB2 数据库开发和优化的技术积累非常薄弱啊，没有原厂的支持是不行的。"

郑总发言："原厂的工程师已经到了，王工，你终于出山了，昨天从北京过来的吧，你可是 DB2 领域的领军人物了，解决这个性能问题可以说是举手之劳啊……"

我发言："多谢大家的信任，初来乍到，我先了解一下情况再发表建议吧。"

客户方 DBA 小李发言（发言非常激烈）："这个应用上线试运行后，正常情况下还好，但是一旦达到了 1000 并发用户时，系统的平均响应时间从 1s 一下急剧增加到 4s 左右，业务人员根本就没法用，要知道现在的硬件可是 16 内核的双机 Power 740 了！"

开发商代表接着发言（已经快哭了）："现在双机 Power 740 配置太低了，难以满足性能需要，请按业务最高峰值配置硬件资源，使用最高档 Power 780，另外存储设备也要升级，用 IBM V7000！"

郑总发言："这算什么回事？不是国庆节前后刚从 x3850 升级到 Power 740 吗，当时你们可是告诉我，Power 740 肯定够用了，现在又要到 Power 780，你们真好意思说出口！"

开发商代表小声说："这个不会是 DB2 的 Bug 导致的吧……"

第三方顾问公司的架构师老张发言："现在是双机 Power 740，一台运行 Weblogic 应用服务器，一台运行 DB2，使用 HACMP 来实现 HA，当出现故障时，一台机器接管另外一台。现在 DB2 有新的技术了，可以用 DB2 pureScale 试试，如果还不行的话，可以考虑 DB2 一体机方案 pureData，应用跑上去肯定快几倍！"

我寻思着，首先感谢这个架构师对 IBM 新技术新产品的信任，不过他也有点太着急了，目前是响应时间慢的问题，不是事务吞吐量遇到瓶颈的问题，在没有真正分析客户的性能瓶颈之前，这样去硬推销 DB2 pureScale 或者 pureData 一体机，不仅不会给客户留下好感，有时候反而让客户对这些新技术或者新产品产生强烈的逆反心理。

会开到这个份上，其实已经陷入了僵局，我坐在角落里，大脑里反复出现郑总的话，经验告诉我，当客户遇到危难猛夸我的时候，其实已经把我推到了风口上了，因为终于有了可以堵枪眼的人了。当然，我也在考虑解决办法，内心也发出这样的感慨：相当一部分客户和开发商，什么都敢用，但是都用的不够专业，不够精细，这样一旦出现性能问题时，他们就只能高谈政治意义，随后习惯性地拍脑袋做决定，很难冷静客观地分析问题和解决问题。

快到中午了，郑总下午还有其他日常安排，开发商提出的用最高档 Power 780 的建议即将被一锤定音的时候，我赶紧发言，把我想说的用最快的速度说了出去："首先，我认为这个系统目前不应使用 Power 780 和 IBM V7000 存储，那是一种对资源的浪费，Power 740 跑这样的负载绰绰有余；其次，我认为 DB2 还有优化空间，我有办法调整一些参数让 DB2 把硬件的能力发挥出来；最后，也是最重要的，应用软件还有很大优化空间，至少在 1000 并发用户的情况

下，响应速度急剧变慢，就和应用有关。”

我说出上面的话后，整个会议室突然宁静了下来，开发商的人员面面相觑，客户领导这时反而笑了，笑的我有点发毛，他和蔼的问我：“王工，你有几成把握？这个项目可是非常紧急的，不能意气用事啊”，我内心其实也虚不过还是表面非常镇定地回答：“立足于现有的硬件环境，请给我 5 天时间吧，当我优化数据库和改造应用的时候，请 DBA 和开发人员配合我，谢谢！”

可能是外来的和尚会念经吧，另外我估计郑总以前就被那些劝他升级高档硬件 Power 780 的人吓怕了。非常出人意料，喜欢张嘴骂人的郑总竟然采纳了我的建议，也对我提出的让 DBA 和开发人员配合这样的要求全部满足。不过，最后他走出会议室的时候，丢下了一句狠话给我：“我现在骂人都骂累了，如果没有搞定，就走人吧。”

星期一：应用自下而上方法学，制定优化计划

立下了军令状，优化工作也就正式开始了。

下午的时候，我和小李以及来自开发商的几个开发人员在会议室进行了深入讨论，确定优化方法和实施计划。开发人员刚要开始给我详细介绍应用逻辑的时候，我立刻拒绝了，现在没有时间听这个。

其实，他们都想知道我葫芦里卖的什么药，想看看我有什么办法解决这个燃眉之急。我内心其实也是有点忐忑的，不过还是先给他们上了一课：“性能优化的方法有两种：自上而下方法学和自下而上方法学。自上而下的思路是早发现早解决，越到后面发现，优化的成本就越高，因为它是贯穿设计、开发和维护的所有阶段的，不过目前已经处于试运行阶段，自上而下显然是不可行了，所以只能采用自下而上方法学。”

我看他们似懂非懂，满脸迷茫的样子，随后接着解释自下而上方法学：“它是一种应急的办法，分别从硬件和应用入手，硬件上通过合理配置让 DB2 发挥硬件最大能力，这个不难，我用一天时间就能解决好。比较费劲的是应用优化，第一，我没有时间了解应用逻辑的细节，当务之急是需要花时间把最影响性能的模块找出来，再从这个模块里面找出前 10 位执行时间最长、执行次数最多的 SQL 语句，分析它们的访问计划，随后运用索引、表连接、分区等技术有效解决它们；第二，在开发应用的时候，开发人员对 DB2 的锁机制估计考虑的不周，上午小李提到的有 1000 并发用户时，系统响应时间急剧增加，这个很可能和锁有关，但本质上牵涉到应用代码。”

于是，大家一起制定了优化计划：星期二，调整参数，发挥硬件的处理能力；星期三，优化执行时间最长、执行次数最多的 SQL 语句；星期四和星期五，从应用角度解决锁问题。

星期二：调整参数，发挥硬件处理能力

我和小李来到了客户机房，一进到机房，让我大吃一惊。作为一名常年在客户现场服务的 DB2 工程师，我去过北上广很多大客户的机房，包括最大电信运营商的和最大银行的，这个客户在宁波，我原以为硬件投入上比不上北上广那些客户，但是没想到进去后，全是清一色的 IBM 服务器、HP 服务器，还有 EMC 存储服务器，足有 100 多台。我忍不住一声叹息，都这么多硬件了，还想忽悠客户继续升级硬件，这也太浮躁了。

言归正传，考虑到业务系统的性能不仅仅由数据库决定，它涉及存储、主机、DB2 数据库和 Weblogic 应用服务器，所以，对这些内容都要进行性能监控。

首先监控了 Weblogic 应用服务器的运行情况。主机的 CPU 和内存利用率一切正常，这个也是开发商的强项，他们对 Weblogic 非常熟悉，所以没有问题也是意料之中的事。

其次是存储规划。这块是小李搭建的，一共 16 块盘，每块盘 300GB 左右，划分为 4 个 RAID 组，每个 RAID 组是 3D+1P，其中两个 RAID 组存放数据，一个 RAID 组存放索引，最后一个 RAID 组存放事务日志，这是一种教科书式的规划。我通过 SSH 命令登录到 DB2 服务器上，使用 dd 命令对每个 RAID 组的 I/O 吞吐能力进行了测速，可以达到 200 M/s 以上，暂时没有调整的必要。

```
dd if=/dev/zero of=/data1/test.file bs=8192 count=5000000
5000000+0 records in
5000000+0 records out
40960000000 bytes (41 GB) copied, 179.411 seconds, 228 MB/s
```

接着是 CPU 利用率。在 1000 并发用户的情况下，拥有 16 个内核的 Power 740 的 CPU 利用率也就是 20%左右，显然是足够了。

最后是监控内存。使用 get snapshot 命令抓取了数据库快照和应用快照，发现缓冲池的命中率竟然只有 60%，而且有大量的排序溢出和编目缓存溢出。我仔细检查了一下，发现服务器总的物理内存为 64GB，平时可用内存大约 48GB 左右，但是发现 DB2 仅仅申请了 2G 内存，用于缓冲池、排序堆、包缓存、编目缓存、锁列表等！把这么多内存空余下来想干啥？这是最大的浪费啊，而且缓冲池命中率这么低。想想也觉得这没有什么奇怪的，很多 DBA 只知道 DB2 提供的 STMM 内存自调优，但并不了解 STMM 对并发访问量非常大的交易系统的自调整有一定的滞后，而且它本身也有一定的开销。

凭借我的技术和经验，我对下面的参数进行了手工调整。其实真正调整的就是下面几条语句，但就是这几条语句，客户、开发商和第三方顾问公司要求我写书面的调整建议书，随后反复评估了整个下午，先在测试环境验证，最后我和小李在晚上负载低的时候进行了正式实施。

调整缓冲池大小，将 DataBuf 设置为 102400 个页面，由于页面大小为 16K，所以总大小为 16GB，IndexBuf 设置为 512000，即 8GB，再次监控，缓冲池命中率达到了 99%！

```
--pagesize 为 16K
ALTER BUFFERPOOL  DataBuf IMMEDIATE size 1024000
ALTER BUFFERPOOL  IndexBuf IMMEDIATE size 512000
```

增加 sortheap 大小直到不出现排序溢出为止，最终调整为 819200。

```
update dbm cfg using SHEAPTHRES 0
update db cfg using sheapthres_shr 1638400
update db cfg using sortheap 819200
```

同样的办法，增大 CATALOGCACHE_SZ 直到编目缓存不出现溢出为止，最终调整为 102400。

```
update db cfg using CATALOGCACHE_SZ 102400
```

同样的办法，增大 PCKCACHESZ 直到包缓存不出现溢出为止，最终调整为 102400。

```
update db cfg using PCKCACHESZ 102400
```

星期三：优化前 10 位执行时间最长、执行次数最多的 SQL 语句

这一天是我过的最开心的一天，也是最顺利的一天。早上刚到机房，小李就把昨晚的对比结果发出来了：经过参数调整后，在 1000 并发用户的情况下，系统的响应时间从 4s 减少到了 2s。

接下来按照计划开始了优化 SQL 语句。本来以为能抓住什么能装满好几页的 SQL 语句，假如我自己解决不了的话，可以请教加拿大多伦多实验室的 SQL 专家，但没想到我从 snapdyn_sql 管理视图里面抓出的前 10 位执行时间最长的 SQL 语句竟然都非常简单，但是执行次数非常多，达到了上亿次！

```
    select  TOTAL_EXEC_TIME, NUM_EXECUTIONS,STMT_TEXT from sysibmadm.snapdyn_sql order
by TOTAL_EXEC_TIME desc fetch first 10 rows only
    NUM_EXECUTIONS       TOTAL_EXEC_TIME      STMT_TEXT
    3292321              293118181
    SELECT ORDERID, ORDERTIME FROM BANK.ORDER WHERE ORDERNumber = ?  AND ORDERTIME > ?
AND ORDERTIME <= ? ORDER BY ORDERTIME DESC
    ...                  ...                  ...
```

首先来看第 1 条执行时间最长的 SQL 语句。这个 SQL 语句很简单，就是对表 ORDER 上的一个动态查询，ORDER 这个表有 3 亿多条记录，它竟然执行了 3292321 秒，执行了 293118181 次！

解决办法是什么？其实很简单，为这条语句创建如下索引，这样不用再对 ORDER 表进行表扫描，可以通过索引扫描取得结果：

```
    CREATE INDEX "BANK"."IQUERY" ON "BANK"." ORDER"
            ("ORDERNumber" ASC,
             "ORDERTIME" ASC)
            MINPCTUSED 10
    ALLOW REVERSE SCANS
```

创建完毕后，运行 runstats 命令，重新收集统计信息，这样优化器在生成访问计划的时候就可以用上索引了。

其他 9 条执行时间最长的 SQL 语句，也是通过索引技术进行了优化。随后，我整理了报告发给了客户、开发商和第三方顾问公司供他们评估使用。很快，他们就做了答复，同意先在测试环境验证。完成验证后，已经夜深人静了，最后我喝着咖啡指导小李在生产库上进行了成功实施。

星期四：解决锁问题

早上来的时候，路上堵车，晚到了 20 分钟。当我刚到机房门口就听见郑总在用非常洪亮的声音给小李和开发商的开发人员讲话。他刚看到我就说：“王工，这几天工作成果很显著嘛，听小李说，现在平均响应时间已经优化到 1s 左右了，看样子大功告成了，晚上请你吃宁波菜！”

看样子郑总的心情不错，但是，我知道还有优化空间，因为星期二调整参数的时候发现了大量的锁等待。凭借我的经验，锁问题的产生通常是由于表的不合理设计或者事务对表的不合理访问导致的，这才是影响并发的关键所在。我告诉郑总，革命尚未成功，宜将余勇追穷寇，

我再加把劲，看看能否再提升一下。

于是，在小李的配合下，我多次使用 db2pd 工具分析锁，发现在高并发的情况下，应用会千万次的执行同一事务逻辑，即查询热表 SALESDATA 中的某一行，随后再修改这个热表中的同样的行，这样当多个事务争抢同一行时，就会出现大量的锁等待。

通过抓取应用快照，发现这些同一事务的业务逻辑也不复杂，就是两条 SQL 语句，都是对表 SALESDATA 操作的，这个表有 8000 万左右的记录数。

```
--开始事务
业务逻辑代码...
--对表 SALESDATA 进行查询
SELECT ARRIVALATTENDED, ARRIVALFIRCODE, ARRIVALHOLIDAYSURCHARGE, NIGHTARRIVAL,
VICINITYAPPROACHCOUNTVFR, VICINITYDEPARTURECOUNTVFR, WORKSTATIONIDENTIFIER FROM
BANK.SALESDATA WHERE FLIGHTIDENTIFIER = ? AND RECORDTYPE = ?
业务逻辑代码...
--对表 SALESDATA 进行更新
UPDATE BANK.SALESDATA  SET ARRIVALATTENDED = ?, ARRIVALFIRCODE = ?,
ARRIVALHOLIDAYSURCHARGE = ?, NIGHTARRIVAL = ?, VICINITYAPPROACHCOUNTVFR = ?,
VICINITYDEPARTURECOUNTVFR = ?, WORKSTATIONIDENTIFIER = ? WHERE FLIGHTIDENTIFIER
= ? AND RECORDTYPE = ?
--结束事务
```

遇到这种情况，只能和开发人员沟通一下了。开发人员告诉我，这是业务逻辑的要求，没有办法修改代码的，所以他们建议调整一下锁有关的参数而不是修改代码。我知道这么做，只是治标不治本，不过调整也能取得一定的效果，也就答应了。

最终将 LOCKTIMEOUT 从 90 调整为 30，这里的单位是秒，90 秒的时间太长了，设置为 30 秒比较合理，这样锁等待超过 30 秒后，就会回滚事务并报锁超时，从而提升事务吞吐量。

```
update db cfg using LOCKTIMEOUT 30
```

将 LOCKLIST 调大为 40960，MAXLOCK 调大为 60，这样为锁分配更多的内存资源。

```
update db cfg using LOCKLIST 40960
update db cfg using MAXLOCKS 60
```

这个事务的频繁执行，会写大量的日志到磁盘上，分别增大了日志缓冲区（LOGBUFSZ）、日志文件大小（LOGFILSIZ）、主日志文件个数（LOGPRIMARY）和辅助日志文件个数（LOGSECOND）。

```
update db cfg using LOGBUFSZ 10240;
update db cfg using LOGFILSIZ 102400;
update db cfg using LOGPRIMARY 50;
update db cfg using LOGSECOND 30;
```

晚上，郑总开车过来，带着我、小李和几个开发人员一起去了一家著名的宁波菜馆。由于 1s 的平均响应时间已经达到了，所以大家也比较放松，吃的不错，也喝了酒。我借着酒劲，告诉郑总，越到后面，调优的成本越高，但我还要再猛攻一下，看看能否达到 0.9s 左右，至少要把锁等待消灭一批才行。