

# ORACLE性能优化 思维与方法论

贾代平 吴丽娟 著

海外借



科学出版社

# ORACLE 性能优化 思维与方法论

贾代平 吴丽娟 著



科学出版社

北京

## 内 容 简 介

随着数据量的不断增加,传统的数据服务系统面临性能退化的挑战。作者在理论和实践的基础上从数据服务的核心层、逻辑层、物理层、实例层、管理层深入探讨 ORACLE 系统的性能调优问题,向读者展示其中的技术原理、手段和思维方式,并试图上升到方法论的层面揭示性能调优的技术路线图。利用此路线图,读者在处理性能问题时,按图索骥就可以找到解决工程问题的法门。核心层介绍了执行计划 (Execution Plan); 逻辑层、物理层分别从数据的存储结构和数据的访问方式上探讨引发性能问题的主要方面;实例层和管理层从数据服务系统整体运行的角度探讨性能问题。本书虽然结合 ORACLE 系统探讨性能问题,但书中讨论的调优技术、思维方式和解决问题的系列方法具有一般性,可供其他数据服务系统参考和借鉴。

本书读者对象为数据服务系统的规划、设计人员,ORACLE 系统的技术专家和高级用户;同时对数据管理、应用开发、系统维护等 DT 时代相关的 IT 从业人员也具有重要的参考价值。

### 图书在版编目 (CIP) 数据

ORACLE 性能优化思维与方法论/贾代平,吴丽娟著. —北京:科学出版社, 2017

ISBN 978-7-03-055672-1

I. ①O… II. ①贾…②吴… III. ①关系数据库系统 IV. ①TP311.138

中国版本图书馆 CIP 数据核字 (2017) 第 292821 号

责任编辑:赵丽欣 常晓敏 / 责任校对:马英菊  
责任印制:吕春珉 / 封面设计:东方人华平面设计部

科学出版社出版

北京东黄城根北街 16 号

邮政编码:100717

<http://www.sciencep.com>

三河市骏杰印刷有限公司印刷

科学出版社发行 各地新华书店经销

\*

2017 年 12 月第 一 版 开本:787×1092 1/16

2017 年 12 月第一次印刷 印张:18 1/2

字数:421 000

定价:69.00 元

(如有印装质量问题,我社负责调换〈骏杰〉)

销售部电话 010-62136230 编辑部电话 010-62134021

版权所有,侵权必究

举报电话:010-64030229; 010-64034315; 13501151303

# 前 言

信息系统的本质是数据服务，而 ORACLE 是数据服务领域当之无愧的主角。本书研究的是基于这个主角的性能问题。欢迎进入 ORACLE 性能调优的世界。

传统地，人们构建信息系统，关注的是“功能”(Function)，系统设计和应用开发主要为“功能”服务。随着数据量的增加和人们对“功能”品质要求的提升，“性能”(Performance)问题逐渐浮出水面，而且在各种数据服务系统百花齐放的情况下，性能成为在市场中致胜的关键要素之一。还记得中国铁路客户服务中心网站(www.12306.cn)的经历吗？系统花费巨资开发的火车票务服务的“功能”，却没有经受住“性能”的考验，投入运行的前三年，曾几度瘫痪。如果不是独家经营，消费者早就会将其弃如敝屣了。这是数据服务系统性能问题的经典案例。

关注“性能”问题，不仅可以提升数据系统的服务质量，还可以实现 IT 资源的有效利用，减少不必要的投资。在实践中见到很多的类似案例，如当系统遭遇性能瓶颈时，建设者往往考虑的是增加投资、实施服务器扩容。事实上，只有当必需的某种吞吐量(Throughput)得不到满足时，系统才需要扩容。大多数情况下通过性能优化、提高资源的均衡利用率，性能瓶颈可以得到缓解甚至消除。

“性能”问题虽然与“功能”有关，但它在功能问题之上。从软件的角度，对于功能，需要调试(Debug)，而对于性能，需要调优(Tune)。功能与性能、调试与调优，都只是一字之差，但其思维方式和处理方法却存在显著区别。调试的目的是发现软件功能上的缺陷(Bug)，可以让软件在运行过程中暂停，以进一步观察内部的细节信息，因此它可以是静态的；而调优的目的是发现软件在运行中的潜在问题，需要在过程中发现瓶颈，因此它是动态的。调试是解决局部的问题，而调优大多数情况下是要解决系统性的问题。在调优实践中，往往缓解或消除了某一环节的问题，却可能会引发另外一些环节的问题。消除了一个瓶颈，另外一些瓶颈又会出现。因此在处理性能问题时，既要熟悉系统的每个环节，又要对不同环节之间的协作方式有充分的了解；既要熟悉局部运行的细节，又要理解不同环节之间的轻重缓急。由此可以看出，处理调优问题要比处理调试问题难度更大，对 IT 人员的要求更高。工程实践中的调优过程不太可能一蹴而就，本质上是一个动态迭代的过程。

曾几何时，在注重“功能”的年代，性能问题是个定性的问题，与用户的主观感受有关，性能的优劣仅影响用户体验而已。ORACLE 很早就显示出，性能问题不仅仅是用户体验问题，而是影响到用户选择的问题。因此在 ORACLE 产品升级过程中，越来越重视性能优化技术的开发。经过几个版本的更新与发展，ORACLE 系统的性能优化问题已经从早期的定性的问题发展为可以定量研究的科学问题。目前，ORACLE 系统的大部分的性能问题都可以定量地去度量，用数据说话。

基于对性能问题的上述理解，本书将 ORACLE 性能调优问题归纳为核心层、逻辑层、物理层、实例层、管理层 5 个不同的层次，并进一步分解为 10 章内容(第 2 章至第 11 章)，每一章讨论不同的优化方法与技术。首先，将系统全局的观点贯穿全书，在介绍具体的技术细节之前，注重讨论调优的思维方式和解决问题的一般方法。其次，都会在每一章节讨



论不同的概念、模型、观测和实验，以便展示其中的技术细节。下面简要介绍各章的主要内容。

第1章介绍基于 ORACLE 系统性能优化的总体框架，阐述性能与资源消耗的基本概念及其相互关系。第2章（核心层）讨论 ORACLE 性能调优的度量及其主要的跟踪技术，这是分析性能问题的基本手段。第3章（核心层）集中讨论 ORACLE 系统的优化器及其决策环境，这是数据服务系统的核心部件，它直接决定用户指令的执行效率。第4章（逻辑层）包括对优化器输出结果的分析与讨论、查看用户指令的执行过程、理解和分析执行过程的每一个环节及其资源消耗，这是性能调优的基本功。第5章（物理层）从数据库对象的物理存储的角度解析数据访问的性能问题，重点解析堆表的存储结构及影响其访问性能的主要方面。

现代数据服务系统离不开索引的支持，第6章（逻辑层、物理层）专门讨论 B 树索引的结构及其访问方式，是软件开发人员掌握 ORACLE 性能优化的关键一环。第7章（逻辑层、物理层）在堆表和 B 树索引的基础上，解析 ORACLE 系统常用对象的段（Segment）结构及其数据访问。第8章（逻辑层、实例层）集中讨论多用户环境和多进程环境的特殊问题，它们既可以阻碍性能，也可以提升性能，关键看如何利用。第9章（实例层）专门讨论 ORACLE 系统的运行结构，应重点关注内存结构的使用及对 ORACLE 系统性能产生的影响，这个影响是全局性的。第10章（实例层、管理层）等待事件接口是一种全新的研究性能问题的角度，它是基于时间因素的考量。利用 OWI 可以有效地观察 ORACLE 系统性能问题的瓶颈。第11章（管理层）讨论基于性能问题的内部统计手段及其性能问题的辅助诊断，它是基于统计意义的分析手段。

由于 ORACLE 系统的复杂性，面对调优问题时很多从业人员处于盲人摸象的状态。即使是某些细分领域的技术专家，在面对一些性能上的疑难杂症时，也会产生强烈的无力感。这就需要系统的研究与学习。作者希望本书是一座桥梁，通过向读者阐述 ORACLE 系统不同层次的调优技术，实现理论与实践的统一、原理和方法的贯通，使读者最终能够形成对 ORACLE 系统性能问题的全局性的认识，从而大大提升处理实际问题的能力。就像一位武林高手，游刃于各个门派和招式之间，融会贯通，出神入化，最后达到无招胜有招的高度。

本书的出版得到科学出版社的大力支持，在此一并致谢。衷心地希望读者能够通过本书，逐级跨越 ORACLE 性能调优的核心层、逻辑层、物理层、实例层、管理层的系列技术壁垒，迈上一个新台阶，达到一览众山小的境界。

由于作者水平有限、认识有限、时间有限，书中难免出现不妥或疏漏，希望读者给予批评指正。若能获得及时的反馈，作者深表谢意。联系邮箱：jiadp@oraclechina.org。

作 者  
2017 年夏

# 目 录

|                       |    |
|-----------------------|----|
| 第 1 章 ORACLE 性能优化思维概述 | 1  |
| 1.1 IT 系统的“功能”与“性能”   | 1  |
| 1.2 服务性能与资源配置         | 2  |
| 1.2.1 性能问题的一般情形       | 2  |
| 1.2.2 业界的服务水平管理       | 2  |
| 1.3 性能研究关注的侧重点        | 3  |
| 1.3.1 关注数据访问的执行过程     | 3  |
| 1.3.2 关注负载和性能之间的动态关系  | 4  |
| 1.4 负载—响应时间曲线         | 4  |
| 1.5 性能优化的一般方法         | 6  |
| 1.5.1 自下而上的优化方法       | 6  |
| 1.5.2 自上而下的优化方法       | 7  |
| 1.6 服务性能的基本问题         | 8  |
| 1.6.1 内存问题            | 8  |
| 1.6.2 CPU 利用率         | 9  |
| 1.6.3 I/O 问题          | 9  |
| 1.6.4 高资源消耗的 SQL      | 10 |
| 1.6.5 引发性能瓶颈的应用问题     | 11 |
| 1.6.6 OLTP 与 OLAP     | 13 |
| 第 2 章 性能度量的主要途径       | 14 |
| 2.1 性能调优的度量概述         | 14 |
| 2.2 EXPLAIN 解释 SQL    | 14 |
| 2.2.1 配置 EXPLAIN      | 14 |
| 2.2.2 获得执行计划          | 15 |
| 2.3 语句级跟踪 AUTOTRACE   | 17 |
| 2.4 会话级跟踪 SQL_TRACE   | 18 |
| 2.4.1 设置 SQL 跟踪       | 18 |

- 2.4.2 TKPROF 格式化跟踪文件 ..... 19
- 2.5 扩展的 SQL 跟踪 ..... 21
- 2.6 度量的阈值与告警 ..... 23
- 第 3 章 优化器及其决策环境 ..... 26
  - 3.1 游标及其处理过程 ..... 26
  - 3.2 优化器的成本核算 ..... 26
    - 3.2.1 ORACLE 成本估算模型 ..... 27
    - 3.2.2 执行计划中的相关概念 ..... 28
  - 3.3 数据访问的路径 ..... 29
    - 3.3.1 表的访问方法 ..... 29
    - 3.3.2 索引的访问方法 ..... 30
  - 3.4 行源的联接关系 ..... 33
    - 3.4.1 内联接和外联接 ..... 34
    - 3.4.2 嵌套循环联接 ..... 36
    - 3.4.3 排序融合联接 ..... 36
    - 3.4.4 散列联接 ..... 37
    - 3.4.5 星形转换 ..... 38
  - 3.5 优化器的决策环境 ..... 42
    - 3.5.1 影响优化器的主要参数 ..... 42
    - 3.5.2 数据库对象的统计信息 ..... 44
    - 3.5.3 系统统计信息 ..... 46
    - 3.5.4 统计信息的维护与管理 ..... 47
- 第 4 章 执行计划的分析与干预 ..... 50
  - 4.1 观测执行计划 ..... 50
    - 4.1.1 查看执行计划 ..... 50
    - 4.1.2 定制执行计划的输出 ..... 51
  - 4.2 认识执行计划 ..... 53
  - 4.3 对多表联接的分析 ..... 55
    - 4.3.1 多表联接概述 ..... 55
    - 4.3.2 联接条件和类型 ..... 55
    - 4.3.3 两两联接的方法 ..... 58

|                     |     |
|---------------------|-----|
| 4.4 干预执行计划          | 65  |
| 4.4.1 优化提示的使用       | 65  |
| 4.4.2 与优化模式有关的 Hint | 66  |
| 4.4.3 与表有关的 Hint    | 67  |
| 4.4.4 与索引有关的 Hint   | 68  |
| 4.4.5 与行源联接有关的 Hint | 70  |
| 4.4.6 其他常见 Hint 举例  | 71  |
| 4.5 管理执行计划          | 72  |
| 4.5.1 SQL 概要文件      | 72  |
| 4.5.2 SQL 计划基线      | 77  |
| 4.6 关注高能耗 SQL       | 85  |
| 第 5 章 数据存储与段结构      | 89  |
| 5.1 堆表的存储结构概要       | 89  |
| 5.1.1 段结构           | 89  |
| 5.1.2 堆表的存储结构       | 90  |
| 5.2 表结构中的数据块        | 90  |
| 5.2.1 块结构及其控制参数     | 91  |
| 5.2.2 行迁移与行链接       | 92  |
| 5.3 正确设置参数 PCTFREE  | 92  |
| 5.4 行迁移与行链接         | 94  |
| 5.4.1 行迁移与行链接的检测    | 95  |
| 5.4.2 行迁移与行链接的消除方法  | 95  |
| 5.5 消除行迁移和行链接的实例    | 96  |
| 5.6 高水位线 HWM        | 97  |
| 5.7 表存储统计实验         | 99  |
| 5.7.1 验证表结构         | 99  |
| 5.7.2 收集统计信息        | 99  |
| 5.7.3 表分析实验         | 100 |
| 5.8 表存储访问效率实验       | 105 |
| 5.8.1 存储访问实验过程      | 106 |
| 5.8.2 重构表的存储        | 112 |



|                              |     |
|------------------------------|-----|
| 第 6 章 索引及相关性能结构分析 .....      | 113 |
| 6.1 ORACLE 索引概述 .....        | 113 |
| 6.2 B 树索引 .....              | 114 |
| 6.2.1 B 树索引结构 .....          | 114 |
| 6.2.2 对 NULL 值的索引 .....      | 115 |
| 6.3 聚簇因子 .....               | 116 |
| 6.3.1 计算聚簇因子 .....           | 116 |
| 6.3.2 对访问性能的影响 .....         | 118 |
| 6.4 索引分析与重建 .....            | 122 |
| 6.4.1 索引分析与统计 .....          | 122 |
| 6.4.2 索引重建 .....             | 125 |
| 6.5 与索引有关的参数 .....           | 126 |
| 6.6 访问索引的方式 .....            | 127 |
| 6.6.1 索引扫描方式 .....           | 127 |
| 6.6.2 两类数据块扫描 .....          | 130 |
| 6.7 B 树索引的维护机制 .....         | 131 |
| 6.7.1 INSERT 操作的 B 树维护 ..... | 131 |
| 6.7.2 DELETE 操作的 B 树维护 ..... | 139 |
| 6.7.3 UPDATE 操作的 B 树维护 ..... | 144 |
| 6.8 复合索引的使用 .....            | 145 |
| 6.8.1 使用原则 .....             | 145 |
| 6.8.2 复合索引和 order by .....   | 148 |
| 6.9 关于索引使用的建议 .....          | 148 |
| 第 7 章 面向性能的对象分析 .....        | 150 |
| 7.1 索引组织表 .....              | 150 |
| 7.1.1 IOT 的主要选项 .....        | 150 |
| 7.1.2 IOT 的使用特性 .....        | 151 |
| 7.1.3 IOT 上的二级索引 .....       | 151 |
| 7.1.4 IOT 的应用提示 .....        | 153 |
| 7.2 聚簇表 .....                | 153 |
| 7.2.1 聚簇的基本概念 .....          | 154 |

|                              |            |
|------------------------------|------------|
| 7.2.2 索引聚簇 .....             | 154        |
| 7.2.3 散列聚簇 .....             | 156        |
| 7.2.4 聚簇表的使用建议 .....         | 157        |
| 7.3 位映射索引 .....              | 157        |
| 7.3.1 位映射索引的结构 .....         | 158        |
| 7.3.2 位映射索引的应用建议 .....       | 159        |
| 7.4 分区表与分区索引 .....           | 159        |
| 7.4.1 分区概述 .....             | 159        |
| 7.4.2 表分区的基本类别 .....         | 160        |
| 7.4.3 分区索引技术 .....           | 164        |
| 7.4.4 分区表与索引的维护 .....        | 168        |
| 7.4.5 分区交换及其应用 .....         | 173        |
| 7.4.6 联机分区处理 .....           | 176        |
| <b>第 8 章 并发处理与并行执行 .....</b> | <b>181</b> |
| 8.1 并发处理与锁 .....             | 181        |
| 8.2 ORACLE 数据库的锁类型 .....     | 182        |
| 8.3 数据访问过程中的加锁 .....         | 183        |
| 8.4 与锁有关的字典参数与指令 .....       | 187        |
| 8.4.1 有关锁的数据字典视图 .....       | 187        |
| 8.4.2 有关锁的初始化参数 .....        | 188        |
| 8.5 事务的隔离级别 .....            | 189        |
| 8.6 锁争用与死锁 .....             | 190        |
| 8.6.1 量测锁争用 .....            | 190        |
| 8.6.2 处理死锁 .....             | 191        |
| 8.7 锁存器 .....                | 194        |
| 8.7.1 锁存器机制 .....            | 194        |
| 8.7.2 检查锁存器争用 .....          | 194        |
| 8.8 并行处理技术概述 .....           | 195        |
| 8.9 SQL 语句的并行处理 .....        | 196        |
| 8.9.1 串行处理与并行处理 .....        | 196        |
| 8.9.2 并行处理的主要概念 .....        | 197        |

|        |                        |     |
|--------|------------------------|-----|
| 8.10   | 并行处理的性能提升              | 198 |
| 8.11   | 并行处理的适应性               | 199 |
| 8.11.1 | 多 CPU 主机系统             | 199 |
| 8.11.2 | 分布式存储                  | 199 |
| 8.11.3 | 资源密集型 SQL              | 199 |
| 8.11.4 | 批量数据扫描                 | 200 |
| 8.12   | 控制并行处理                 | 200 |
| 8.12.1 | 确定并行度                  | 200 |
| 8.12.2 | 使用并行提示 Hint            | 202 |
| 8.12.3 | 调整与并行处理有关的参数           | 203 |
| 8.13   | 并行处理的执行计划              | 203 |
| 8.14   | 实时的并行处理信息              | 205 |
| 8.15   | 并行处理的跟踪                | 206 |
| 8.16   | 并行处理实例                 | 207 |
| 8.16.1 | 并行数据更新                 | 207 |
| 8.16.2 | 并行数据添加                 | 208 |
| 8.16.3 | DDL 的并行处理              | 210 |
| 8.16.4 | 并行索引访问                 | 211 |
| 8.17   | 并行处理的优化                | 212 |
| 8.17.1 | 并行处理的一般性原则             | 212 |
| 8.17.2 | 部分并行化与完全并行化            | 213 |
| 8.17.3 | 监控实际运行中的并行度            | 215 |
| 8.17.4 | 并行处理进程的负荷分配            | 216 |
| 8.17.5 | RAC 环境下的并行处理           | 218 |
| 第 9 章  | 实例结构的分配与优化控制           | 220 |
| 9.1    | ORACLE 的实例架构           | 220 |
| 9.1.1  | 实例的内存结构                | 220 |
| 9.1.2  | 实例的进程结构                | 223 |
| 9.1.3  | 实例的存储结构                | 226 |
| 9.2    | 最近最少使用算法               | 227 |
| 9.2.1  | Cache Hit 与 Cache Miss | 227 |

|        |                  |     |
|--------|------------------|-----|
| 9.2.2  | LRU 与 MRU        | 228 |
| 9.2.3  | 表扫描的处理           | 228 |
| 9.2.4  | 直接路径读            | 229 |
| 9.3    | 实例缓存的配置与优化       | 230 |
| 9.3.1  | 计算缓存命中率          | 230 |
| 9.3.2  | 使用多类型缓存          | 232 |
| 9.3.3  | 设置缓存的大小          | 233 |
| 9.3.4  | ASMM 与内存抖动       | 234 |
| 9.4    | 共享缓冲池的配置与优化      | 235 |
| 9.4.1  | 共享缓冲池的构成         | 236 |
| 9.4.2  | SQL 解析及其执行       | 236 |
| 9.4.3  | 关注游标共享           | 236 |
| 9.4.4  | 检查共享池效率          | 240 |
| 9.5    | 用户工作区的调整         | 242 |
| 9.5.1  | PGA 与 UGA        | 242 |
| 9.5.2  | PGA 的使用与限制       | 243 |
| 9.5.3  | 监控 PGA 的性能       | 244 |
| 第 10 章 | 基于等待事件的诊断分析      | 248 |
| 10.1   | 基于等待事件的性能问题描述    | 248 |
| 10.1.1 | 性能的时间因素          | 248 |
| 10.1.2 | 等待接口与信号量         | 249 |
| 10.2   | 用户响应的时间模型        | 250 |
| 10.2.1 | CPU 服务时间         | 250 |
| 10.2.2 | 等待事件与等待时间        | 251 |
| 10.3   | 统计项与等待事件         | 251 |
| 10.4   | DB Time 与 DB CPU | 253 |
| 10.5   | Top SQL 说明       | 254 |
| 10.6   | 等待事件直方图          | 255 |
| 10.7   | 性能与等待事件          | 256 |
| 10.8   | 常见的等待事件及其描述      | 257 |

|         |                          |     |
|---------|--------------------------|-----|
| 10.9    | 等待事件不能反映的信息 .....        | 258 |
| 10.10   | 收集等待事件信息 .....           | 258 |
| 10.11   | 利用等待事件发现性能瓶颈 .....       | 260 |
| 10.11.1 | 案例 1：一个慢速查询的处理 .....     | 260 |
| 10.11.2 | 案例 2：耗时的调度批处理 .....      | 262 |
| 10.11.3 | 案例 3：客户服务器应用中的等待事件 ..... | 265 |
| 10.11.4 | 案例 4：疲于应付的数据库服务器 .....   | 266 |
| 第 11 章  | 服务性能管理与性能统计 .....        | 269 |
| 11.1    | AWR .....                | 269 |
| 11.1.1  | AWR 的控制 .....            | 270 |
| 11.1.2  | AWR 报告解读 .....           | 271 |
| 11.2    | ADDM .....               | 275 |
| 11.2.1  | ADDM 诊断框架 .....          | 275 |
| 11.2.2  | ADDM 诊断案例 .....          | 276 |
| 11.3    | ASH .....                | 279 |
| 11.3.1  | ASH 采样框架 .....           | 279 |
| 11.3.2  | ASH 主要应用 .....           | 280 |
| 参考文献    | .....                    | 284 |



## 优化思维概述

## 1.1 IT 系统的“功能”与“性能”

今天的企业和机构比以往任何时刻都更依赖于 IT 系统提供的各类服务，其中，数据服务是构建 IT 系统的核心价值。数据服务系统的“性能”关系着日常生活和工作的方方面面，关系着工作效率和生活质量。长期以来，IT 领域习惯于关注数据服务系统能够实现的功能，而一定程度上忽视了数据服务系统的“性能”。没有满足用户需求的“性能”表现，服务的“功能”往往失去意义，这在一些大中型数据服务系统中表现尤为突出。事实上，大多数的数据服务系统在实现了业务处理功能、投入运行的那一刻起，其“性能”表现就会直接影响用户体验和用户满意度。这里大家所熟知的典型案例就是中国铁路客户服务中心（[www.12306.cn](http://www.12306.cn)），该系统在 2011 年正式投入运营后，其性能表现广受用户诟病，其售票系统在第一个“春运”期间几度瘫痪，无法提供正常的购票服务。无独有偶，2009 年 6 月 26 日，美国 TMZ.com 网站率先报道了摇滚歌星迈克尔·杰克逊（Michael Jackson）病逝的消息，大量歌迷涌入网站，访问量骤增，导致该网站陷入瘫痪状态。类似的案例在国内外都曾出现过，概括地说都是由于“负荷”增加、“性能”减退，最终导致“功能”失效。

数据服务系统的“功能”问题是一个纯粹的技术问题，而数据服务系统的“性能”问题不仅仅是一个技术问题，也是一个 IT 工程问题。随着服务范围的扩大，用户数量的增加，“性能”问题几乎困扰着每个数据服务系统的运营方。从纯技术的角度，所有的数据服务系统都是高度相似的，这就给系统地研究数据服务系统的“性能”问题提供了科学依据。

本书讨论的对象是基于 ORACLE 的独立数据服务系统，它是一个逻辑上的概念，不仅是指物理上独立的系统，也可指工程中的集群服务系统、云端服务系统等。不论物理上的外在形式如何，从应用的角度，它是一个具有确定边界的数据服务系统，为用户服务的内在处理机制是完全一致的，因此关于此类系统的“性能”问题，完全可以用一种统一的观点去描述、研究。

## 1.2 服务性能与资源配置

### 1.2.1 性能问题的一般情形

在 IT 项目的工程实践中，服务资源的配置具有一定的盲目性，在这里，项目设计人员的工程实施经验起着主导作用，往往要等到项目上线后才能发现服务资源配置不足或资源配置不均衡（导致潜在的性能瓶颈）。研究性能问题的意义在于：根据“性能”要求预先配置服务资源，或根据已知的服务资源确定可以达到的“性能”目标，让“性能”指标在服务资源的配置中起决定性作用，避免服务资源配置的错位或浪费。

数据服务系统的“性能”问题起源于如下两个方面：一是随着系统负荷的增加，用户请求的响应时间开始缓慢增加，用户体验开始逐步变差，当超过某个临界值时，用户选择离开，这在面向市场的信息服务行业中是致命的；二是一旦系统负荷突破了某个所能承受的区域，数据服务的性能急剧下降，负荷的微小变化都会使“性能”变得敏感，最终系统不胜负荷，导致系统服务停滞或挂起（Hang）或“功能”的实现被无限拖延，最终导致数据服务系统瘫痪或死机。就目前国内外此领域的服务现实而言，第一种情形时刻都在发生，由于是正常的定量变化，大多数情况下无须特别关注；对于第二种情形，由于数据服务方尚没有系统的、完善的应对负荷急剧增加的策略，系统瘫痪的情形时有发生，在 IT 行业的新闻中常会看到此类报道。也就是说，在目前的条件下，完全杜绝这类事件的发生几乎是不可能的，所能做的是尽可能地降低此类事件发生的概率。

### 1.2.2 业界的服务水平管理

目前 IT 行业和学术界对性能问题的研究主要集中在计算机系统的服务水平管理（Service Level Management, SLM）上，它属于介于纯粹的技术学科和工程管理学科之间的交叉学科。SLM 涵盖如下五个方面的内容。

1) 容量管理（Capacity Management）：在满足特定数据服务需求的情况下应有的容量支持，它依赖于性能指标、负荷监控、应用的业务处理等，包括存储容量、网络容量、计算容量、I/O 带宽等。

2) 高可用性（High Availability）：现代数据服务系统一旦投入运行大多需要保持 7×24h 运行，高可用性研究的是保障数据服务系统持续运行的一系列解决方案，如构建数据服务集群（Cluster）、服务冗余（Service Redundancy）、主从冷热备用等。

3) 系统保障与恢复（Data Guard and Recovery）：此方向关注的是数据服务系统的可靠性、可维护性、备份策略与可恢复性、应变能力，核心是通过各种技术手段确保业务数据的安全。

4) 风险评估（Risk Assessment）：系统运行过程中，负荷和用户需求都是动态变化的，而服务资源则是相对静态的，系统需要以不变应万变，确保事先设定的服务标准，显然这充满不确定性。风险评估包含识别或预测潜在的风险和缓解风险两层含义，此处当然包含“性能”不满足用户需求的风险。

5) IT 投资分析（IT Investment Analysis）：此方向研究的是 IT 基础设施的成本效益核算（Cost Benefit Accounting, CBA）和投资回报率（Return On Investment, ROI），这是数

据服务“工程”区别于“技术”的显著方面。

显然,上述五个方面都与本书要研究数据服务的“性能”问题密切相关,国内外对这些分支的研究各有侧重。

首先,容量问题是制约数据服务“性能”的一个方面,然而单一的容量无法解决现实中遭遇的“性能”问题,甚至出现系统经过扩容后性能不但没有增加反而下降的案例。

其次,高可用性是数据服务系统提供连续不间断服务的保证,此方向的研究重点是避免系统出现各种极端的情况,防止意外事件的发生,如系统计划外停机、死机、崩溃等,另外某些高可用性方案也与容量密切相关。例如,集群数据服务系统既可增加系统的容量,也可提高系统的可用性。

再次,系统保障与恢复方向的重点在于“数据”本身,关注用户数据的安全,避免数据在意外情形下的不可访问和数据丢失。作者涉足此领域较早,曾就 ORACLE 系统撰写了一篇关于数据恢复的论文,讨论在各种意外情形下的完全数据恢复(零数据丢失)问题,被业界和同行广泛引用。

接着,风险评估方向探讨的是在满足服务需求方面可能面临的风险,本质上讨论的是未来的“性能”和未来的“负荷”之间的关系。在目前的学术研究和行业实践中,对于评估的结果,比较通行的做法是给出风险的显著性等级,显然这对于精细化的服务管理是远远不够的,需要将定性风险转换为数字化的“性能”与数据服务系统要素之间的定量关系。

最后,IT 投资分析研究的是在有限投资的约束条件下数据服务系统的实施情况,是一个典型的工程问题,目前基于数据服务系统的财务分析(包括 CBA 和 ROI)主要面向 IT 投资决策领域。

本书讨论的主题是基于 ORACLE 系统的性能优化问题,研究的是在资源限定的情况下如何改善或优化用户的性能体验。

## 1.3 性能研究关注的侧重点

### 1.3.1 关注数据访问的执行过程

用户数据访问的“性能”取决于用户请求在系统内部的执行过程。概括地说,用户的需求是 What to do,而这个过程是 How to do,如果能够详细探究这个过程,性能问题的细节信息就会展现在开发者面前。

对于性能问题,无论是由内部资源争用(Contention)还是由内部资源耗尽(Exhausted)导致的“性能”问题,最终都表现为用户请求响应时间的增加,即请求指令执行过程的延长。而这里的执行过程可以划分为计算过程和等待过程两个方面,其中计算过程反映了数据服务系统为响应用户请求实施的一系列内部操作及其资源消耗,根据现代资源调度和最优决策理论,可以将其转换为某些定量的指标,如计算路径(Computing Path)、计算成本或计算代价(Computing Cost)等。由于“性能”问题是一个多变量约束的问题,通过对计算过程中获取的路径和代价的分析,可以探索这个计算过程的最优路径和最小代价,将数据服务系统的“性能”问题转换为具有普遍意义的最优化问题。

探讨执行过程的另外一个方面就是分析等待事件(Wait Events)。前面的计算过程是一个资源消耗的过程,而这里的等待过程是一个资源需求不能满足的过程,即资源等待。在

之前的研究中发现，每当数据服务系统出现显著的“性能”问题时，其内部总是伴随着大量的等待过程（如果出现严重的等待，业界有一个直观但非正式的词——Hang）。在一个高度并发的数据服务系统中出现资源等待的情形不可避免，关键是首先要了解这些等待的情形，其次要了解出现这些内部等待发生的原因，在此基础上才有可能设法减少这些内部等待出现的次数，或缩短这些内部等待持续的时间。

这是从数据服务内在特性的角度研究“性能”问题的两个方面，即计算与等待。如果能够通过对资源的优化和调度，一方面减少资源的消耗，另一方面减少内部出现的等待，那就可以在性能优化方面做出改进。

### 1.3.2 关注负载和性能之间的动态关系

1.3.1 小节从数据服务系统的内部研究“性能”问题，另外一个角度是从数据服务系统的外部研究“性能”问题，即关注数据服务系统的输入/输出关系，这里的“输入”是指用户的各种数据处理请求给系统带来的“负载”，“输出”是指人们关心的“性能”。

对于一般意义上的计算机系统，“负载”和“性能”都是定性的概念，往往随着讨论问题的侧重点不同而有所不同。要研究数据服务系统的动态特性，首先要能够精确地计量“负载”和“性能”。对于本书要研究的数据服务系统，“负载”和“性能”的真正含义是什么？目前关于“负载”的说法众说纷纭：指某一时刻的并发会话（Session）数，单位时间内的事务处理的数据量，单位时间内的事务到达率，用户请求导致的逻辑读和物理读的数量，消耗的 CPU 时间，数据的查询量，数据的更新量，队列的长度或等待时间等。对于“性能”的描述同样如此：用户请求的响应时间，系统的 CPU 利用率、I/O 带宽利用率，缓存的命中率，事务处理的平均时间，吞吐量，单位数据块的读写时间等。

这就需要结合数据服务系统的特性，研究如何统一化、归一化地描述系统的“负载”和“性能”，并能适应于不同应用环境的需求；同时为了方便重复实验和数据采样，还有必要研究如何模拟一种用于“性能”测试目标的标准负载。与此同时，不论是“负载”还是“性能”，同样需要研究它们精确的计量方法。

通过对“负载”和“性能”的系列描述，可以将数据服务系统抽象为一个复杂的动态系统，而这里的“负载”和“性能”就是动态系统的输入和输出。如果把不同侧面的“负载”描述作为这个动态系统的输入向量，而把不同角度描述的“性能”计量作为输出向量，通过研究输出向量序列对输入向量序列的依赖关系，理论上就可以很好地把握数据服务系统的动态特性。这里，统计建模和序列计算理论（包括时间序列的理论与方法）都可以用来处理数据服务系统的输入向量序列、输出向量序列及其输入/输出关系。

## 1.4 负载—响应时间曲线

诚然，在一个复杂的数据服务系统里，实现性能优化需要关注的“点”有很多，而且这些“点”不是一成不变的，随着讨论问题的变化而变化，如讨论系统整体的性能、讨论系统某个局部的性能等。在讨论性能优化的一般性思维时，是否存在需要普遍关心的普适物理量呢？回答是肯定的。就性能表现来说，IT 系统无论是整体还是局部，其运行的机制是相似的，有输入、有资源消耗、有输出，而且任何输出都是在特定输入量（压力或负载）、限定资源消耗的情况下产生的。