



Advanced

OWI *in* Oracle 10g

Advanced Oracle Wait Interface in 10g

高级OWI与

赵东郁 著
金鸿哲 译
任重模 审校

Oracle性能调整

高级 OWI 与 Oracle 性能调整

赵东郁 著
金鸿哲 译
任重模 审校

上海科学技术出版社

内 容 提 要

本书内容分为两大部分：第一部分是 OWI。介绍了 OWI 基本思想、Oracle 代表性的等待事件、Latch 和 Lock 的工作机制及相关等待事件、Oracle 的专用内存空间（SGA）的各区域工作机制等。第二部分是个别等待事件。介绍了各种等待事件的原因和解决方法。本书的主要特点是不仅把重点放在 Oracle 的最新版本 10g 上，而且提供了符合数据库实际状况的解决方案。

本书主要适合 DBMS 管理员、IT 系统管理员、程序开发人员，以及想要提高数据库相关知识的学生作为参考书使用。

图书在版编目（CIP）数据

高级 OWI 与 Oracle 性能调整 /（韩）赵东郁著；金鸿哲译. —上海：上海科学技术出版社，2007.8
ISBN 978-7-5323-9040-3

I. 高… II. ①赵…②金… III. 关系数据库 - 数据库管理系统, Oracle IV. TP311.138

中国版本图书馆 CIP 数据核字（2007）第 119873 号

上海世纪出版股份有限公司 出版、发行
上 海 科 学 技 术 出 版 社
（上海钦州南路 71 号 邮政编码 200235）
新华书店上海发行所经销
常熟市文化印刷有限公司印刷
开本 787×1092 1/16 印张 22
字数：310 千字
2007 年 9 月第 1 版 2007 年 9 月第 1 次印刷
印数：1 - 1250
定价：98.00 元

本书如有缺页、错装或坏损等严重质量问题，
请向承印厂联系调换

目 录

Part1 关于 OWI	1
第 1 章 OWI 介绍	3
1.1 何谓 OWI	3
1.2 OWI 的特点	6
1.2.1 OWI 是面向问题的 (problem-oriented)	6
1.2.2 OWI 是定量的	7
1.2.3 OWI 是征兆学性的	7
1.2.4 OWI 正在不断完善	8
1.3 OWI 工具	10
1.3.1 OWI 动态视图	10
1.3.2 其他重要动态视图	13
1.3.3 Extended SQL Trace	15
1.3.4 oradebug 与转储 (dump)	17
1.3.5 Automatic Workload Repository	21
1.4 OWI 数据收集	23
1.4.1 Logoff Trigger	23
1.4.2 利用 SQL 周期性收集	23
1.4.3 Automatic Workload Repository (AWR)	23

1.4.4 Direct Memory Access (DMA)	24
1.5 其他	25
1.5.1 PL/SQL	25
1.5.2 脚本 (script)	25
第 2 章 Latch 和 Lock.....	28
2.1 Oracle 的同步机制.....	28
2.2 锁存器	31
2.2.1 何谓锁存器	31
2.2.2 锁存器所保护的资源	32
2.2.3 锁存器工作机制	34
2.2.4 锁存器相关的动态视图 (dynamic views)	40
2.2.5 一般锁存器相关的等待事件.....	41
2.3 锁	43
2.3.1 锁的分类	43
2.3.2 锁保护的资源	45
2.3.3 锁工作机制	47
2.3.4 锁相关的动态视图	49
2.3.5 普通锁相关等待事件	51
第 3 章 Oracle 内部结构和 OWI.....	53
3.1 高速缓冲区 (buffer cache) 和 OWI.....	53
3.1.1 高速缓冲区结构	53
3.1.2 Working Set	55
3.1.3 Buffer lock.....	60
3.1.4 检索缓冲区	60

3.1.5 高速缓冲区转储 (buffer cache dump)	62
3.2 Shared Pool/Library Cache 和 OWI	65
3.2.1 共享池 (shared pool) 和堆	65
3.2.2 库高速缓冲区 (library cache) 结构	71
3.2.3 SQL 的执行	75
3.3 事务和 OWI	79
3.3.1 事务的概要	79
3.3.2 事务和块转储	81
3.3.3 事务和撤销块转储	86
3.4 段和 OWI	88
3.4.1 段的概要	88
3.4.2 手动段空间管理 FLM	89
3.4.3 自动模式的段空间管理 ASSM	94
3.5 I/O 和 OWI	97
3.5.1 I/O 概要	97
3.5.2 应用程序层 (application layer)	97
3.5.3 Oracle 内存层	98
3.5.4 Oracle 段层	100
3.5.5 OS/裸设备层 (raw device layer)	100
3.5.6 Direct path I/O	103
3.6 重做和 OWI	105
3.6.1 重做 (redo) 概要	105
3.6.2 重做缓冲区	108
3.6.3 重做日志	110

Part2 个别等待事件	113
第 4 章 高速缓冲区上的等待事件	115
4.1 latch: cache buffers chains	115
4.1.1 低效的 SQL	116
4.1.2 Hot Block	121
4.2 latch: cache buffers lru chain	126
4.3 buffer busy waits/read by other session	130
4.3.1 Select/Select 引起的 <i>read by other session</i>	133
4.3.2 Select/Update 引起的 <i>buffer busy waits/read by other session</i> ...	137
4.3.3 Insert/Insert 引起的 <i>buffer busy waits</i>	141
4.3.4 Update/Update 引起的 <i>buffer busy waits</i>	146
4.4 write complete waits	157
4.5 free buffer waits	164
4.6 enq:TC-contention	167
4.6.1 Parallel Query	168
4.6.2 Tablespace Hot backup	170
4.7 enq:CI-contention, enq:RO-contention	173
第 5 章 库高速缓冲区上的等待事件	178
5.1 latch: shared pool	178
5.2 latch: library cache	187
5.2.1 Hard Parsing 或 Soft Parsing 过多时	188
5.2.2 Version Count 高时	193
5.2.3 SGA 区域发生 Page Out 时	194
5.3 library cache lock 和 library cache pin	195

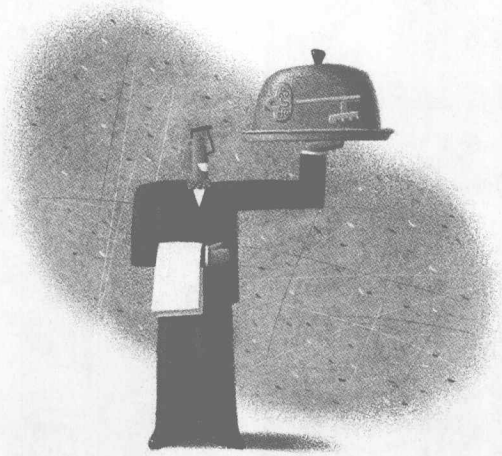
第 6 章 行高速缓冲区上的等待事件.....	213
6.1 row cache lock	213
6.2 enq:SQ-contention, DFS lock handle (SV)	218
第 7 章 事务上的等待事件.....	221
7.1 enq:TM-contention	221
7.1.1 无索引的外键.....	224
7.1.2 不当的 DDL 引起的 TM 锁争用.....	224
7.1.3 利用 Lock table...主动获取 TM 锁时	226
7.1.4 执行 Direct/Parallel Load 工作时	227
7.2 enq :TX-row lock contention, enq :TX-allocate ITL Entry, enq :TX-index contention	229
7.2.1 多个会话修改相同行时 (<i>enq:TX-row lock contention</i> , <i>mode=6</i>)	230
7.2.2 多个会话引起唯一键冲突时 (<i>enq:TX-row lock contention</i> , <i>mode=4</i>)	233
7.2.3 ITL 条目不足 (<i>enq:TX-allocate ITL entry, mode=4</i>)	235
7.2.4 多个会话引起位图索引冲突时 (<i>enq:TX-row lock contention</i> , <i>mode=4</i>)	239
7.2.5 索引叶节点上发生分割时 (<i>enq:TX-index contention</i> , <i>mode=4</i>)	242
7.2.6 其他 (<i>enq:TX-contention</i>)	243
7.3 enq:UL-contention, PL/SQL lock Timer.....	245
第 8 章 段上的等待事件.....	248

8.1 enq:HW-contention	248
8.2 enq: ST-contention, enq: TT-contention.....	257
8.3 enq:US-contention	262
第 9 章 I/O 上的等待事件	266
9.1 db file scattered read.....	266
9.1.1 应用程序层	268
9.1.2 Oracle 内存层.....	269
9.1.3 Oracle 段层.....	271
9.1.4 OS/裸设备层	272
9.2 db file sequential read.....	273
9.2.1 应用程序层	273
9.2.2 Oracle 内存层.....	274
9.2.3 OS/裸设备层	278
9.3 direct path read	279
9.4 direct path write	282
9.5 direct path read temp/direct path write temp	284
9.5.1 应用程序层	284
9.5.2 Oracle 内存层.....	284
9.6 direct path read (lob) /direct path write (lob)	289
9.7 db file parallel write.....	291
9.8 control file parallel write.....	293
第 10 章 重做缓冲区上的等待事件.....	296
10.1 latch: redo writing, latch: redo allocation, latch: redo copy.....	296
10.2 log file sync.....	303

10.2.1 提交次数和 log file sync	303
10.2.2 I/O 系统的性能和 log file sync	306
10.2.3 重做数据量和 log file sync	306
10.2.4 重做缓冲区的大小和 log file sync	308
10.3 log file parallel write	309
10.4 log buffer space	311
10.5 log file switch completion, log file switch (checkpoint incomplete) , log file switch (archiving needed) , log file switch (private strand flush incomplete)	315
第 11 章 网络上的等待事件	317
11.1 网络速度缓慢时	318
11.2 SQL 执行次数异常地过高时	319
11.3 应用程序的实现方式上存在问题时	322
参考文献和资料说明	323
术语索引	326
图索引	338
表索引	339
测试索引	340

Part 1
关于 OWI





第1章 OWI 介绍

1.1 何谓 OWI

Oracle 在执行某项工作过程中，没有如愿获得需要的资源时，将一直等待与相应资源对应的事件，直到解除对资源的占有为止。假如，某进程欲修改其他进程已修改的特定行（row），需要等到获取之前进程已拥有的对事务（transaction）资源的锁（TX 锁）为止。进程在系统上记录名为 *enq: TX-row lock contention* 的事等待件。我们可以对此等待现象进行观察，进而推断进程发生了何种问题。综上所述，记录和观察进程所经历的等待现象的功能和界面及方法论，统称为 OWI，也就是 Oracle Wait Interface。最初 Oracle 7.0.1 中对 OWI 进行过介绍，这时的等待事件总共为 104 个。但是 Oracle 不断改善代码，并细分了事件，因而事件数量在 Oracle 8.0 中是 140 个，Oracle 8i 中是 220 个，Oracle 9i 中则是 400 个。而今，Oracle 10g 中则提供了 800 个以上的事件。Oracle 中所使用的等待事件随着版本的提升一路攀升。这意味着 Oracle 正向准确报

告性能问题的方向发展，所以对 OWI 的应用也会逐步提升。

进程没有获得特定资源而等待时称为“等待 XXX 事件 (event)”。如果，将一个块 (block) 从磁盘物理读取的进程发出请求后，“等待 *db file sequential read* 事件”，直到对实际请求的块的 I/O 请求结束为止。举一个其他例子。某进程欲改变其他进程正在修改的缓冲区时，需要等待“*buffer busy waits* 事件”，直到其他进程结束。事件或等待事件可定义为直到进程获得特定资源为止等待发生的“事件”。

等待事件通过 P1、P2、P3 这三个参数表现当前等待的资源。等待事件的 P1、P2、P3 值可以通过 V\$SESSION_WAIT、V\$SESSION 视图等动态(dynamic)视图或 SQL Trace 文件观察。例如在 *db file sequential read* 事件中，P1=file#、P2=block#、P3=request block count。利用 P1 和 P2 两个值可以确认对哪个段(segment)的哪个块发生 I/O 等待。各事件的 P1、P2、P3 的意义都不尽相同，可通过 V\$EVENT_NAME 视图进一步确认。Oracle 世界里非常广泛地使用“资源(resource)”这个词，与一般我们所理解的“资源不同”。OWI 观点上的“资源”，只限于事件的 P1、P2、P3 三个参数表达的范围。假设 *db file sequential read* 事件的 P1=file#、P2=block#、P3=request block count，事件所表示的资源就是“特定数据文件的特定块”。*enq: TX-row lock contention* 事件的 P1=name|mode、P2=usn<< 16||slot、P3=sequence，它所表示的资源是“以 usn、slot、sequence 确定的对事物处理的锁(lock)”。*latch free* 事件的 P1=latch address#、P2=latch#、P3=tries，此事件表现的资源就是“以 latch address 形式表现的锁存器(latch)对象”。

Oracle 利用等待次数、超时次数、等待时间三个值表示等待现象，因此可以表述如下：进程在特定时间内总共等待 10000 次 *latch: library cache* 事件，其中发生五次超时，其总等待时间是 10 秒。从用户角度看，等待次数、超时

次数、等待时间中，感觉最为有用的是“等待时间”。特别是锁存器等资源，较频繁地试图获得锁存器，因此以等待次数判断争用（contention）是没有任何意义的，通过等待时间可最为准确地分析争用程度。

特定进程在执行某工作过程中，实际工作的时间（service time）是 100 秒，而与此时间相反，此工作执行期间内光是等待事件的时间就是 50 秒，用户实际感受到的响应时间为 $100+50=150$ （秒）。也就是：

$$\text{响应时间 (Response Time)} = \text{工作时间 (Service Time)} + \text{等待时间 (Wait Time)}^1$$

只要降低工作时间和等待时间，响应时间自然地随之降低，并增加了用户满意度。有些问题只要降低工作时间，等待时间自然也会降低。但是有些时候，不可能降低工作时间，这时只有降低等待时间一种方法。

¹ 摘自：《利用 OWI 的 Oracle 诊断和调优》，EXEM 译。

1.2 OWI 的特点

1.2.1 OWI 是面向问题的 (problem-oriented)

使用 OWI 方式与使用其他性能诊断方式最大的不同，就是 OWI 是面向问题的。接下来会提及其重要性已经远不如以前的 buffer cache hit ratio。什么时候听着都非常顺耳的这个用词，只能说明命中率 (hit ratio) 高，但不能说明当前如何影响性能。若正在影响性能，也不能提供发生此种情况的原因或任何线索。与此相比，等待时间这个概念能直观地表现当前发生何种性能问题。比如，整个 DB 的运行时间是 100 小时，其中 *latch: cache buffers chains* 事件等待时间是 30 小时，这说明多个会话 (session) 过多地发生逻辑读取 (logical reads)，从而引发了严重的争用。

Oracle 每当发生问题时，也就是在资源获得失败时会等待特定事件，以此表示当前发生何种问题、为何发生。而且通过进一步分析，可对引发问题的进程进行判断或分析。因此，只有着眼于当前系统发生的等待现象，才可最直观地掌握性能问题。利用从 Oracle 10g 开始提供的 AWR (Automatic Workload Repository) 性能历史记录管理功能，以 OWI 为基础，可以对过去特定时刻的性能问题进行分析。

一直到 Oracle 8i，buffer cache hit ratio (以下简称 BCHR) 被视为诊断高速缓冲区性能最为重要的因素。但是 Oracle 9i 之后，就不能再利用 BCHR 诊断性能。BCHR 的内存价格贵，I/O 速度又很慢，而且 CPU 数量有限的时候是很有意义的指标。但是在千兆内存和已明显得到改善的 I/O 设备，还有数十个 CPU 的服务器上，比起 BCHR，减少锁存器争用更为重要。Oracle 为了减少锁存器争用，不断地改善高速缓冲区的管理方法。Buffer Pinning、多重缓冲池、多种块大小等就是其具有代表性的方法。也正因为这样，传统的计算方式中的 BCHR，也基本上失去了意义。

1.2.2 OWI 是定量的

OWI 不是“推测”，而是基于“事实”和“数据”，将性能分析变为可能的方法论。OWI 上的事实是以特定事件的等待次数、超时次数、等待时间来表示的。当前 DBMS 用户饱受响应时间严重下降痛苦。其原因是什么呢？回答此问题的一种方法是这样的：“查看会话数，感觉会话数太多，CPU 使用率也是相当高，是不是因为错误地使用索引发生的问题呢？或者是 Hard Parsing 引起的问题？”这种分析方式会引起许多推测，因此在经过许多执行错误后才能弄清楚问题原因。但是如果使用了 OWI 方法论，则得出以下答案：“分析过去 5 分多钟的等待现象，在所有 DB CPU 使用时间中 *latch: library cache* 事件的等待时间占总时间的 50% 以上。还有，*latch: shared pool* 等待事件基本上没有发生。因而，不是因为 Hard Parsing 所引起性能问题的概率较高。平时，*parse time (total)* 统计值平均每秒是 100 次，而性能问题发生以后每秒是 1000 次以上。所以当前性能问题的原因是过多的 Soft Parsing。”如上所述，OWI 不仅能确定问题为何发生，也能确定具体引起了多么严重的性能问题。

虽然上面举的例子的前提是非常单纯的情况，但是从上可知：如果灵活使用了 OWI，则可以摆脱性能问题的模糊感，利用“观察的事实和数值”，可以将问题以简洁的数据表示。也就是通过 OWI，我们可以将复杂的性能问题，解释为任何人都易于理解的值、定量的值。如果保存了等待事件和性能指标的历史记录，就可以更为准确地进行分析。

1.2.3 OWI 是征兆学性的

Oracle 中发生等待现象，极端地比喻就与人的身体发生异常症状相似。就