Oracle 白皮书

2011 年 5 月

Oracle Berkeley Database 11g R2

性能概述

目录

概述

当选择一个数据库时，其性能的好坏往往是我们要考虑的第一关键因素。本白皮书介绍了一些性能测定的方法，旨在帮助你理解从 Berkeley DB 数据库的一些常见配置预期会得到怎样的性能。你的应用程序的性能也取决于你的数据、数据访问的模式、缓存大小、其他配置参数、操作系统、以及硬件等。基准测试并不能反映某一个特定的应用程序的性能好坏，但它们可以提供一些基准，并为建立基本可行的期望提供指导和帮助。

介绍

通过一些在一定配置下进行的测试，本文给出了有关 Oracle Berkeley DB (BDB) 11gR2 (11.2.5.0.2.21) 吞吐量的信息。这些测试包括：不同配置下 BDB 键/值接口 (key/value API) 得到的吞吐量，和使用 Wisconsin 和类 TPC-B 基准对 BDB SQL 接口进行测试得到的性能数据。

实验环境

所有的实验结果都是用相同的硬件配置得到的，硬件信息如表 1 所示：

表 1. 硬件配置

处理器	操作系统	RAM	硬盘及速度	文件系统
Intel 酷睿 2 双核 E8400, 3.0GHz	Redhat Linux 5.4	4GB	SATA, 7200 RPM	EXT3

键/值接口的性能概述

Berkeley DB 是高可配置的，它可以配置来使用或禁止数据库操作的大多数特性，例如，事务日志（WAL）功能，并发控制的锁机制。由于这个原因，Berkeley DB 将在两种不同配置环境下进行实验：分别对应于非事务性及事务性。

第一组实验测试数据存储 Data Store(DS)。DS 是 Berkeley DB 的配置选项之一。DS 本质上是一种简单的、单线程、非事务性的存储系统。第二组实验测试事务性数据存储 Transactional Data Store(TDS)。TDS 是 Berkeley DB 的一种能提供所有事务语义(transactional semantics)的配置选项。表 2 给出这两种功能集的区别。

表 2. 数据存储与事务性数据存储属性

特点	数据存储（DS）	事务性数据存储（TDS）
访问方法（Access Method）	B 树	B 树
锁（locking）	无	页级锁
日志（logging）	无	带有 128MB 缓存的磁盘
事务（transactions）	无	同步（Synchronous）
共享内存（Shared Memory）	不共享（DB_PRIVATE）	不共享（DB_PRIVATE）
缓存	512MB	512MB

这两组实验都使用 512MB 的缓存，在数据库环境中设置 DB_PRIVATE 的标志，同时空间局部值（spatial locality）设为 10。使用 DB_PRIVATE 标志说明在数据库中使用堆内存（heap memory）作为其高速缓存（cache），这种配置只适用于单进程（可能有多个线程）的环境中执行。

最常用的数据库性能衡量标准是吞吐量（就是在固定时间段里读取或写入的记录数）。在我们的实验中，我们把吞吐量定义为是用 Berkeley DB11gR2 (版本号为 11.2.5.0.21)每秒能达到的操作次数。

在实验中，我们使用固定长度的记录。每条记录包括 64 字节的键值（key）和 64 字节的数据值（data）。所有的键值和数据值均为整数型。每个实验进行 57,600 批次，每个批次对应 10 个顺序的键值操作。比如，在插入测试中，随机产生第一个键的值（记住 k），并对它进行插入操作；接下来再插入 9 个记录， 这 9 个记录具有连续的键值（k+1, k+2, k+3, …, k+9）；这个过程将重复 57,600 次。每个测试都运行 5 次，并记录相应的吞吐量的平均值和标准差。

针对每个配置，下面是实验的步骤：

- **步骤一：**
  生成一个具有 576,000 条记录的数据库（就是 57,600 组，每组具有 10 个顺序键值的记录）。这个过程仅仅是生成用于实验的数据库和预热缓存，测试并不记录其消耗的时间。
- **步骤二：**
  检索 57,600 个随机记录组，每组具有 10 个连续键值的记录（读取操作）。
- **步骤三：**
  更新 57,600 个随机记录组，每组具有 10 个连续键值的记录（更新操作）。
- **步骤四：**
  删除数据库中的所有记录，以每组具有 10 个连续键值的记录为单位（删除操作）。
- **步骤五：**
  像步骤一那样，重新生成测试的数据库（插入操作）。

针对每一个类型的操作（读取、更新、删除、插入），计时从第一个组操作前开始直到完成最后一组操作结束。这期间不包括用于打开和关闭数据库环境和数据库句柄所花的时间。

数据存储：单线程

第一组实验是衡量使用 Berkeley DB DS 来测试单线程应用程序所达到的吞吐量。表 3 是实验得到的结果。

表 3. BDB DS 单线程的性能

描述	插入(Insert)		读取(Fetch)		删除(Delete)		更新(Update)	
	操作数/秒	标准差	操作数/秒	标准差	操作数/秒	标准差	操作数/秒	标准差
DS	208,139	329	264,665	229	158,506	236	250,297	870

事务性数据存储：单线程

第二组实验统计了基于 Berkeley DB TDS 的单线程应用程序在不同的配置下所达到的吞吐量。这些配置包括：写持久性，日志持久性，和存储介质。每个配置将解释为：

- 写持久性

默认条件下，事务将被同步提交到磁盘。设置 DB_TXN_NOSYNC 标志将意味着异步提交（即数据不会同步到磁盘），而 DB_TXN_WRITE_NOSYNC 标志意味着把数据写入到文件系统但不会同步到磁盘。

- 日志持久性

默认测试，日志是存储在磁盘中；同时也测试了非持久的、内存中的日志信息。

- 不同的存储介质

默认的配置是采用一个传统的硬盘。我们将比较使用这种默认配置和使用 2GB 的 RAM 在存储数据库和日志文件方面得到的性能。在所有的情况下，事务都会同步提交。

表 4. BDB 单线程性能

描述	插入(Insert)		读取(Fetch)		删除(Delete)		更新(Update)	
	操作数/秒	标准差	操作数/秒	标准差	操作数/秒	标准差	操作数/秒	标准差
TDS SYNC	693	10	159,837	895	831	23	1,732	18
TDS RAMDISK[1]	49,673	912	162,848	1,588	44,279	277	60,973	291
TDS WNS[2]	37,664	460	160,307	293	36,110	454	52,922	258
TDS NS[3]	53,199	613	159,980	1,419	49,340	183	86,960	639
TDS INMEM[4]	66,435	102	163,229	815	58,845	101	97,602	396

1 2GB RAMDISK，数据库和日志均存在 RAM 磁盘中

2 DB_TXN_WRITE_NOSYNC，事务提交时设置的标志

3 DB_TXN_NOSYNC，事务提交时设置的标志

4 允许在内存中存储日志

在所有的实验中，我们使用足够大的缓存大小，所以磁盘的 I/O 操作并不影响读的吞吐量。通过对比 TDS 和 TDS RAMDISK 的结果，不难发现磁盘的 I/O 延时和性能会大大影响写操作。通过比较 TDS SYNC, TDS INMEM, TDS WNS 和 TDS NS 的结果，我们可以发现不同的持久性设置也会影响到写的吞吐量。

上述结果显示了对数据库来说很常见的一些有趣特点。其中一个特点是 I/O 是决定数据库性能好坏的最关键因素。I/O 操作尽管对数据库的读操作来说，仅会造成较小影响，但却是决定写操作性能的关键原因。在使用 TDS 进行的实验中，插入和更新操作会导致大量的 I/O 操作，所以在性能上会受到很大的影响。

这些实验表明了五种主要类别造成的延迟：

- 内存和处理器
- 从用户到内核空间之间的传递
- 文件系统
- 存储 I / O
- 介质

TDS INMEM 的实验是最快速的，因为所有的事务提交到内存的数据缓冲区，没有把数据转入到文件系统缓冲区或是磁盘所带来的开销。使用 TDS INMEM 单线程进行插入操作的实验为你的系统显示了最理想的速度，但是可能不是很有用，原因是此时数据不具有永久性。在进程意外结束时，内存中的数据将会被删除掉，所有没有持久性可言。

当事务配置了 DB_TXN_NOSYNC 标志时，在事务提交时，数据是不会从日志缓存区提交或同步到文件系统的。相反，日志会被保留在日志缓存区中直到该缓存满为止，此时一个文件系统的写操作会把日志从用户内存传输到内核或是文件系统内存中。接下来，由文件系统自己决定数据是否要写进以及何时写进磁盘介质中。所以，在 TDS INMEM 和 TDS NS 之间的不同性能正好反映了当日志缓存区满时把日志记录从用户级别的日志缓存区传输到内核或是文件系统内存所带来的开销。

当事务配置了 DB_TXN_WRITE_NOSYNC 标志时，Berkeley DB 在每个事务提交时都会把日志从用户级别的日志缓存区提交或同步到操作系统或是文件系统。这样只要在把数据写到磁盘之前该操作系统不挂掉，就能提供持久性的保证，。因此，在应用程序挂掉时，只要操作系统能继续在运行的话，数据就不会丢失。但是，如果此时操作系统或是硬件也挂掉的话，已写入操作系统但没有写入磁盘的数据将会被丢失。

在 TDS WNS 和 TD SNS 的实验中，插入操作得到的不同结果就是反映了把日志从 Berkeley DB 日志缓存区拷到文件系统时造成事务提交的时间上的差别。最后，通过对比 TDS SYNC 和 TDS RAMDISK 的结果，你就会观察到往内存中写和往磁盘上写带来的性能差别。在 TDS SYNC 这个实验中，SATA 总线和硬盘驱动带来的开销会大大降低性能。

相当有趣的是，在所有的实验中，读操作的性能基本上是一致，平均值占标准差只有1%（标准差=1655.92，平均=161240.2，百分比=1.026989482）。这说明了在所有的实验中，缓存造成的影响是一样的，因此在对内存中的数据进行读操作时带来的事务开销是可以忽略不计的。 尽管没有统计，但是 I/O 操作的开销决定了对在缓存外的数据进行读操作所需要的时间。

事务性数据存储：扩展对称多处理系统

下表显示的是 BDB TDS 在对称多处理系统（SMPs）时性能的数据。针对多线程访问，每个线程都使用一个数据库句柄（DB handle）以减少线程间的竞争。每个线程使用一个计时器来计算其吞吐量。下表显示的是所有这些信息的集合。

当引入第二个线程时，每个操作的吞吐量将相应地减少，原因是系统将进行一些锁操作，所以会导致一些相应的开销。总体上来说，写操作的吞吐量将随着线程数加大而呈按比例递增。但是，当读线程超过 4 个的时候，读操作的吞吐量将下降，原因是此时 CPU 的利用率是 100%，而额外的线程需要一些不必要的上下文交互和硬件缓存的回收。

表 5. 在 SMP 系统上，配置不同线程数对应的 TDS 性能

描述	插入(Insert)		读取(Fetch)		删除(Delete)		更新(Update)	
	操作数/秒	标准差	操作数/秒	标准差	操作数/秒	标准差	操作数/秒	标准差
TDS 1 个线程	693	10	159,837	895	831	23	1,732	18
TDS 2 个线程	647	7	106,196	432	767	20	1,634	27
TDS 3 个线程	690	9	126,762	762	863	20	1,909	32
TDS 4 个线程	721	8	134,581	497	882	18	2,085	54
TDS 8 个线程	859	26	128,928	423	967	27	2,497	32
TDS 12 个线程	930	24	112,735	518	1,013	20	2,658	149
TDS 16 个线程	968	13	99,860	820	1,085	34	2,837	117

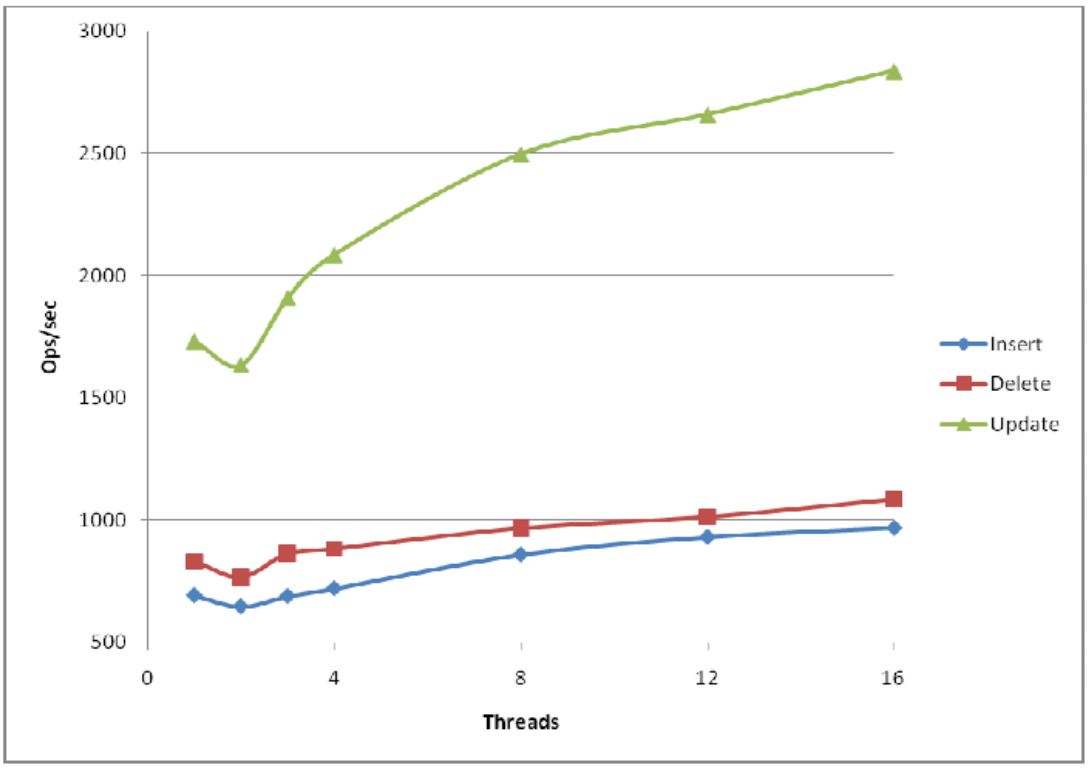

图 1. 在 SMP 系统上，配置不同线程数对应的 TDS 性能

SQL 接口性能概述

目前有很多流行的基准（benchmarks）是专门为基于 SQL 的关系数据库而设的。为适应 Berkeley DB 的 SQL 接口，我们参考实现了类似的基准。

我们的实验启发自两种流行基准: 事务处理性能委员会的 TPC- B 基准 [1] 和 D. J. DeWitt 的 Wisconsin 基准 [2]。

这些基准实现和测试数据，没有被第三方验证或审核，而仅仅作为我们在不需要其它软件系统介入的情况下试图去认识有关 BDB 性能特征所做的努力。

TPC-B 基准

TPC-B 基准的测试需要实现 TPC B 规范要求的一些模式（Schema）和标准事务。帐户（accounts）的数目是按比例下降的，这样做的目的是在运行的时候方便管理。它有 1000 家分支机构（branches），总共拥有 10000 个出纳员（tellers）和 100,000 个帐户。

下表显示的是由 TPC B 生成的事务吞吐率（以每秒的事务数（TPS）为单位）。这些结果是从一个 256 兆字节大小的数据库缓存中得到。

表 6- TPC-B 事务吞吐率

线程数	事务吞吐率（TPS）
1	1846.71
2	2310.84
3	2508.26
4	2678.14
5	2808.51
10	2859.15

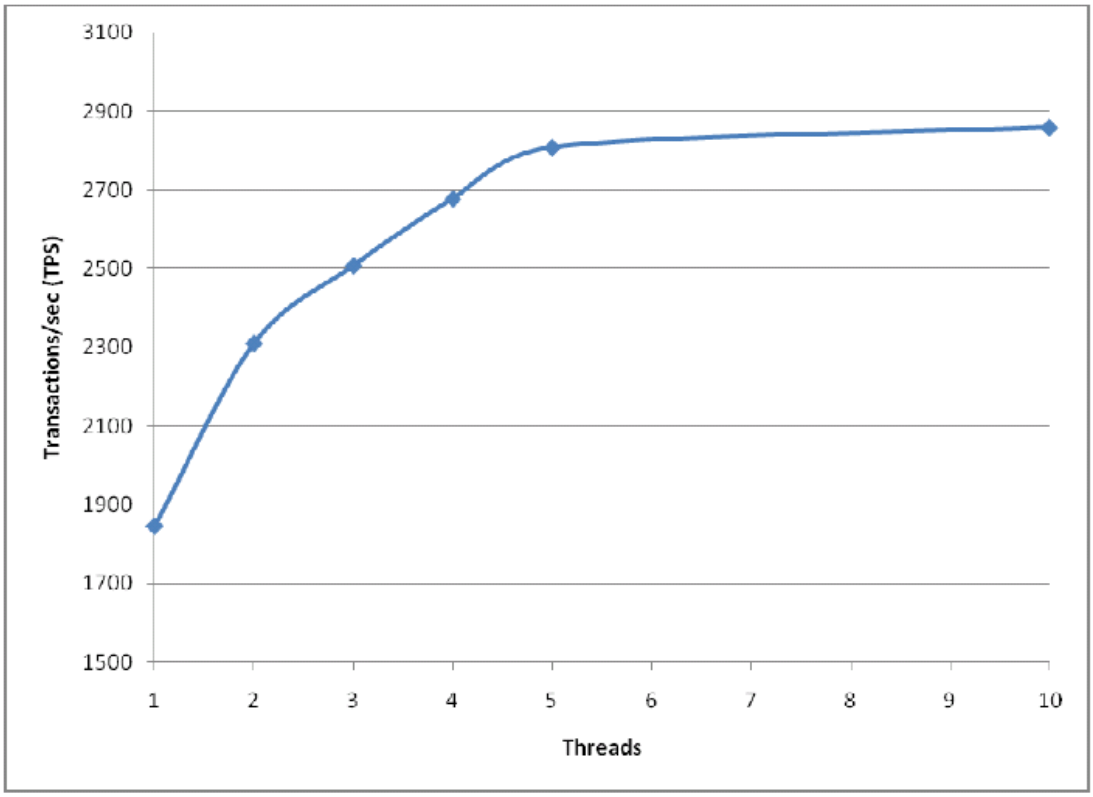

图 2. 由类 TPC-B 基准衡量的每秒事务数（TPS）以及对应的线程数

Wisconsin 基准

本测试尝试按照 Wisconsin 基准规范规定的模式和查询进行，但是并没有尝试去降低由数据库系统和操作系统引起的缓存效应。

这个实验衡量了在标准数据库上进行大量不同测试案例的单用户性能。所有的测试案例组合的结果可用于分析一个数据库的优势和弱势。对于查询的描述，请参阅链接 [4] 中的附录一。

为了这些实验能够正常运行，数据库是配置了针对单线程和单用户访问的编译优化选项。每个查询运行 10 次，下表显示的是平均需要的运行时间，以毫秒为单位。

表 7. 由 Wisconsin 基准衡量的平均运行时间

案例编号	运行时间（毫秒）	案例编号	运行时间（毫秒）	案例编号	运行时间(毫秒)	案例编号	运行时间（毫秒）
1	8.175	9	31047.220	17	16.944	25	18.163
2	13.875	10	3160.824	18	58.801	26	0.907
3	0.879	11	5335.219	19	136.595	27	3.844
4	6.749	12	9.920	20	3.944	28	4.380
5	1.368	13	9.942	21	32.566	29	1.081
6	11.017	14	11.585	22	32.774	30	1.273
7	0.244	15	19.407	23	0.102	31	1.252
8	1.373	16	14.759	24	17.742	32	1.081

参考

1 http://www.tpc.org/tpcb/default.asp
2 http://firebird.sourceforge.net/download/test/wisconsin_benchmark_chapter4.pdf
3 http://www.tpc.org/tpcb/spec/tpcb_current.pdf
4 http://firebird.sourceforge.net/download/test/wisconsin_benchmark_chapter4.pdf

结束语

Berkeley DB 键/值接口无论在单线程还是在多线程的应用程序上都执行地非常好。同时，如果一个应用程序并不需要完全的持久性的话，它将能非常显著地提高其性能。

Berkeley DB 在 11gR2 5.0 的版本中增加基于键/值接口的 SQL API。你可以从这里下载 Oracle Berkeley DB: http://www.oracle.com/technology/software/products/berkeley-db/index.html

你也可以在 Oracle Technology Network (OTN) 论坛上发表相关评价以及提交问题：
http://forums.oracle.com/forums/forum.jspa?forumID=271

有关销售或者产品支持的信息，请发送 Email 到：berkeleydb-info_us@oracle.com
想了解有关最新产品发布信息，请发送 Email 到：bdb-join@oss.oracle.com