

# Oracle TimesTen 产品和技术

*Oracle 白皮书*

2005 年 12 月

**ORACLE**

引言.....	3
“满足实时应用场合的需要”.....	3
实时应用程序的发展.....	3
实时行业.....	3
实时企业.....	4
实时数据管理软件.....	4
应用程序层部署.....	4
产品.....	5
Oracle TimesTen In- Memory Database.....	5
Replication – TimesTen to TimesTen.....	5
Cache Connect to Oracle.....	5
内存中数据库技术.....	5
ORACLE TIMESTEN 的物理结构.....	6
应用程序层共享库.....	6
内存中数据结构.....	7
系统进程.....	8
管理程序.....	8
检查点和日志文件.....	8
数据复制技术.....	8
高速缓存技术.....	9
深入研究 IMDB 技术.....	11
查询优化.....	11
缓冲池管理.....	11
索引结构.....	12
差别体现在哪里.....	13
非凡的性能.....	13
可伸缩性.....	13
响应时间.....	14
实时功能.....	15
数据管理.....	15
查询处理.....	19
数据复制.....	19
高速缓存.....	21
事件处理.....	24
结论.....	25

# Oracle TimesTen 产品和技术

“Oracle 收购 TimesTen 是一次绝好的技术整合，它将 TimesTen 高性能内存中体系结构与 Oracle 巨大的数据库潜力结合起来”  
—Rob Hailstone,

## 引言

2005 年 6 月，Oracle 收购了内存中数据库软件的领先供应商 TimesTen, Inc.。Oracle 产品与 TimesTen 产品的结合为端到端数据管理提供了独有的单供应商解决方案。

本文介绍了 Oracle TimesTen 产品和技术及其与其他 Oracle 产品集成方面的问题，并使用该软件及其手册前的“书面”评估。

### “满足实时应用场合的需要”

Oracle TimesTen 产品为性能关键系统提供了应用层数据管理，并针对快速响应以及实时高速缓存 Oracle 数据进行了优化。公司可以使用 Oracle TimesTen 扩展其软件基础架构，以创建具有以下特性的系统：

- 即时响应性
- 高度可伸缩性
- 连续可用性

这些系统用于：

- 提高客户忠诚度
- 吸引新客户
- 简化操作
- 避免采用昂贵的专用软件开发方法。

自 1998 起，Oracle TimesTen 先后部署到网络、电信服务、运营支持系统、联系中心、航空和预订系统、指挥和控制系统以及证券交易系统的生产环境中，并在时间关键的行业和实时企业中发挥了出色的作用。全球范围内的数百家公司在生产应用程序中使用了 Oracle TimesTen，其中包括 Amdocs、Aspect、Avaya、Cisco、Ericsson、JP Morgan、Lucent、Nokia、Salesforce.com 和 Sprint。

## 实时应用程序的发展

### 实时行业

需要实时应用程序的行业包括网络设备制造商、电信运营商、证券交易所和经纪公司、航空公司、货运和物流公司以及国防和情报机构等

很多公司都离不开实时应用程序。它们对于公司运营必不可少。需要实时应用程序的典型行业包括网络设备制造商、电信运营商、证券交易所和经纪公司、航空公司、货运和物流公司以及国防和情报机构。以往，为这些行业构建实时应用程序还需要开发实时基础架构软件。商业性的实时基础架构软件当时并不

存在。只要对应用程序的要求保持不变，这些系统便可以起作用，尽管速度快但却不够灵活。然而，动态行业对功能的要求将很快超出静态应用程序的能力范围，而在存在商业基础架构软件的情况下，不值得为开发、测试以及维护专用基础架构软件进行投资。

### 实时企业

能否使用实时处理智能地捕获、分析和响应重要事件逐渐成为衡量优秀企业的标准。

随着企业网络传输的信息量不断增多，能否使用实时处理智能地捕获、分析和响应重要事件逐渐成为衡量优秀企业的标准。这不仅仅对业务关键流程的执行和管理很重要。客户希望任何与他们保持重要业务关系的公司能够提供高度定制的交互和最快的响应。

业务活动监视、复杂事件处理、RFID/基于传感器的应用程序、Web 门户以及 Web 服务有助于将应用程序扩展到企业边界。由于配置为由相互关联的例程的动态集合，因此这些应用程序构成面向服务体系结构 (SOA) 这一总体方法的一部分。尽管许多应用程序扩展到企业边界，但大多数数据源仍存在于后端，其中主要包含大量很少访问的旧数据以及少量当前活动的信息。SOA 概念的自然扩展包括应用程序层中的轻型、实时数据管理，连接到企业数据源以为当前活动的数据提供实时性能。

### 实时数据管理软件

仅仅收集并高速缓存与应用程序密切相关的数据是不够的，而将企业数据库与某个应用程序放到同一平台上也是不切实际的

作为实时处理最大的受益者，企业体系结构在应用程序层中提供了事件、数据和事务管理，从而使前端系统具备了快速响应性和更深入的洞察力。仅仅收集和高速缓存与应用程序密切相关的数据 — 第一代内部基础架构软件通常如此 — 还不够。此外，将企业数据库与某个应用程序放到同一平台也是不实际的。

所需要的是一代轻型基础架构软件，它应提供易于使用、功能强大的界面和广泛使用的查询语言 — 可以轻松地与现有后端数据库、消息处理系统以及应用服务器进行通信，可以充分利用当前内存丰富的联网计算平台的全部潜在性能。这正是 Oracle TimesTen 提供的用于管理实时数据的新一代基础架构软件。

### 应用程序层部署

所需要的是一代可以利用当前内存丰富的互联计算平台的全部潜在性能的轻型基础架构软件。Oracle TimesTen 产品无缝地集成到这些环境中，并提供了针对应用程序层部署优化的体系结构

当前的很多新应用程序开发都着力于提高与客户之间的交互或简化内部操作以消除延迟和多余的成本。它们是位于网络边界附近（在某些情况下，作为网络内部的托管服务）的实时应用程序。这是一个新的企业应用程序层，它在平台、性能和可用性方面的要求不同于旧的后端应用程序。业务事件包含在应用程序预订的网络消息中，用于触发实时处理以及其他消息发布。

要满足这些应用程序的响应时间和可伸缩性目标，通常必须将

基础架构软件 and 应用程序（包括驱动该应用程序的某些或所有数据）部署到同一平台上。Oracle TimesTen 产品可以无缝地集成到这些环境中，并提供了针对应用程序层部署优化的体系结构以及用于实现高度定制解决方案的配置选项。

## 产品

Oracle TimesTen 由三个基于内存中数据库、数据复制和高速缓存技术的产品组成

Oracle TimesTen 实时数据管理软件由三个基于内存中数据库、数据复制和高速缓存技术的产品组成。本文该部分简要介绍了这些产品和技术。后续部分将提供更多详细信息。

### Oracle TimesTen In-Memory Database

*Oracle TimesTen In-Memory Database* 是一个内存优化的关系数据库，它为应用程序提供了当今实时企业和行业（例如电信、资本市场和国防）所需的即时响应性和非常高的吞吐量。Oracle TimesTen In-Memory Database 作为高速缓存或嵌入式数据库被部署在应用程序层中，它利用标准的 SQL 接口对完全位于物理内存中的数据存储区进行操作。

### Replication – TimesTen to TimesTen

*Replication – TimesTen to TimesTen* 是 Oracle TimesTen In-Memory Database 的一个选项，它支持服务器间的实时数据复制，以获得高可用性和负载共享。数据复制配置可以是双机热备份 (active-standby) 或负载均衡 (active-active)，可以使用异步或同步传输，可以包含冲突检测和冲突解决以及在故障服务器恢复后自动重新同步。数据复制与 Cache Connect to Oracle 选项完全兼容。

### Cache Connect to Oracle

*Cache Connect to Oracle* 是 Oracle TimesTen In-Memory Database 的一个选项，它为位于应用程序层中的 Oracle 数据创建实时、可更新的高速缓存。它免除了后端系统的计算负担，并支持反应灵敏且可伸缩的实时应用程序。Cache Connect to Oracle 能够将 Oracle 数据的子集加载到 TimesTen 中，能够双向传播更新，能够使对非高速缓存数据的 SQL 请求的透传自动化，并能够在故障之后自动重新同步数据。Cache Connect to Oracle 与 Replication – TimesTen to TimesTen 选项完全兼容。

## 内存中数据库技术

内存中数据库技术实现了这样一个关系数据库：其所有运行时数据都位于内存中且数据结构和访问算法利用该特性实现了性能突破

内存中数据库 (IMDB) 技术是 Oracle TimesTen 产品的基础技术。IMDB 技术实现了这样一个关系数据库：其所有运行时数据都位于内存中且数据结构和访问算法利用该特性实现了性能突破。与完全高速缓存的 RDBMS 相比，IMDB 技术使用的 CPU 资源少很多，这是因为它避免了用于管理内存缓冲区并处理多个数据位置（磁盘和内存）的开销。对 IMDB 技术

而言，磁盘用于实现持久保存和恢复，而不是用作主要的数据库存储位置。

Oracle TimesTen 内存优化的性能由提供事务属性功能、持久保存机制以及系统故障恢复所完善。有各种方法可用于锁定、多用户隔离和记录，从而满足了各种应用情形（从临时的查找高速缓存到核心事务交易和计费系统）。

TimesTen 通过将已提交事务中的更改记录到磁盘并定期更新数据库的磁盘镜像（称作“检查点”）实现了持久性。应用程序可以对日志写入磁盘的时间进行配置（要么与事务的结束时间同步，要么一直延迟到事务完成之后），从而实现了更高的性能。在许多情况下，高吞吐率要比同步记录重要，尤其是当事务的货币价值很低或事务数据的生存时间较短时（如，当在每几秒钟传输一次移动电话位置的网络中跟踪移动电话的位置时）。

Oracle TimesTen 的性能由提供事务属性功能、持久保存机制以及系统故障恢复所完善

Oracle TimesTen 产品支持的原生接口符合标准，且一般与其他符合标准的关系数据库兼容。应用程序通过 JDBC（Java 数据库连接）或 ODBC（开放数据库连接）接口发出 SQL（结构化查询语言）命令。用于定义数据存储区和复制配置的语句也遵守 SQL 语法惯例。此外，还使用 SNMP（简单网络管理协议）发出标准化的系统管理警报。

Oracle TimesTen 提供了具有标准 JMS 接口的开放式事务日志 API (XLA) 来读取事务日志。它对于创建响应数据库更新的应用程序很有用。在这方面，XLA 是一个轻型“触发器”。它还用于构建从 Oracle TimesTen 到其他数据库系统的自定义数据复制。

## ORACLE TIMESTEN 的物理结构

本部分从安装、运行和使用计算资源的角度介绍了 Oracle TimesTen 产品的系统组件。

Oracle TimesTen 产品由以下组合组成：

- 共享库
- 内存中数据结构
- 系统进程（后台程序/子后台程序/代理）
- 管理程序
- 磁盘上的检查点和日志文件

### 应用程序层共享库

SQL 操作包含一组共享库中，开发人员将这些库链接到应用程序

实施 SQL 操作的例程以及相关函数包含在一组共享库中，开发人员将这些库链接到应用程序并作为应用程序进程的一部分执行。此共享库方法与更常规的 RDBMS 不同，后者实现为与应用程序连接（通常通过客户端/服务器网络连接）的一组可执

行程序。通常，将数据管理器库嵌入到应用程序中可能会使数据存储区在应用程序进程异常终止时容易遭到破坏。Oracle TimesTen 解决了这一难题。Oracle TimesTen 库使用正在申请专利的算法（称作“MicroLogging”）保护自身不受应用程序进程故障的影响。内存中数据库保持一致，并且其他应用程序不受影响而继续运行。

IMDB 技术实现为由应用程序、实用程序和系统进程通过共享库例程访问的内存中数据存储区。可根据恢复目的有选择地维护日志和检查点（备份副本）的磁盘文件

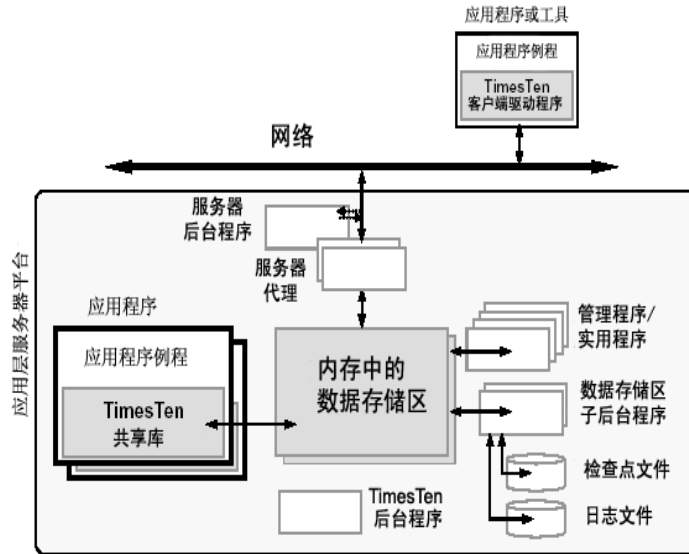


图 1: Oracle TimesTen In-Memory Database 产品的组件

应用程序还可以使用客户端/服务器连接访问 Oracle TimesTen 数据库，但在大多数情况下，使用直接链接的应用程序将实现最佳性能。对于客户端/服务器访问，服务器后台程序（后台进程）将远程用户连接到服务器代理，服务器代理包含代表远程客户端访问数据存储区的共享库。

### 内存中数据结构

内存中数据库在操作系统的共享内存段中维护，它包含所有用户数据、索引、系统目录、日志缓冲区、锁表和临时空间。多个应用程序可以共享一个数据存储区，而单个程序可以访问同一系统上的多个数据存储区。

内存中数据存储区在操作系统的共享内存段中维护，它包含用户数据、索引、系统目录、日志缓冲区、锁表和临时空间。多个应用程序可以共享一个数据存储区，而单个应用程序可以访问多个数据存储区

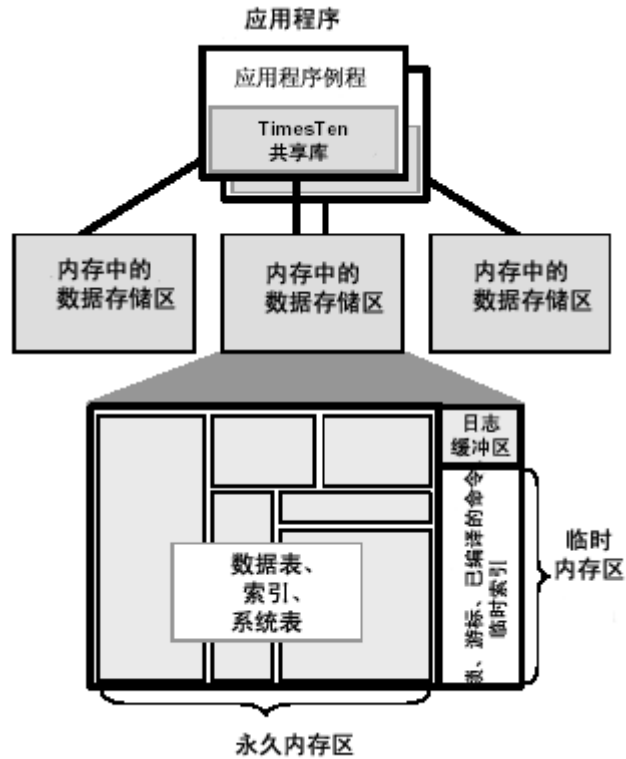


图 2：内存中数据存储区

### 系统进程

后台进程在系统级别为启动、关闭和应用程序故障检测提供服务，并在数据存储区级别为加载、检查点和死锁处理提供服务。每个数据存储区都有一个实例范围的 *TimesTen* 后台程序（一个系统可能有多个实例）和一个单独的数据存储区子后台程序。

### 管理程序

用户、脚本或应用程序显式调用实用程序来执行交互式 SQL、批量复制、备份/恢复、数据存储区移植和系统监视等服务。

并定期将数据存储区和事务日志的更改写入磁盘

### 检查点和日志文件

并定期将数据存储区和事务日志的更改写入磁盘。如果需要恢复数据存储区，Oracle TimesTen 将把磁盘上的数据存储区检查点与仍位于日志文件中的已完成事务合并在一起。检查点和日志文件使用普通的磁盘文件系统。在有限情况下，可以将 Oracle TimesTen 配置为执行无磁盘操作。

### 数据复制技术

当需要近乎连续的可用性或负载分配时，可以将数据复制配置为在两个或多个服务器间发送更新。将主服务器配置为发送更

新，将用户服务器配置为接收更新，而服务器可以同时用作主服务器和用户服务器来执行双向复制。基于时间的冲突检测和解决用于当同时在多个位置更新同一数据时（这种情况很少出现）建立优先级。

当需要近乎连续的可用性或负载分配时，可以将复制配置为在两个或多个服务器间发送更新。服务器可以同时用作主服务器和用户服务器以执行双向复制

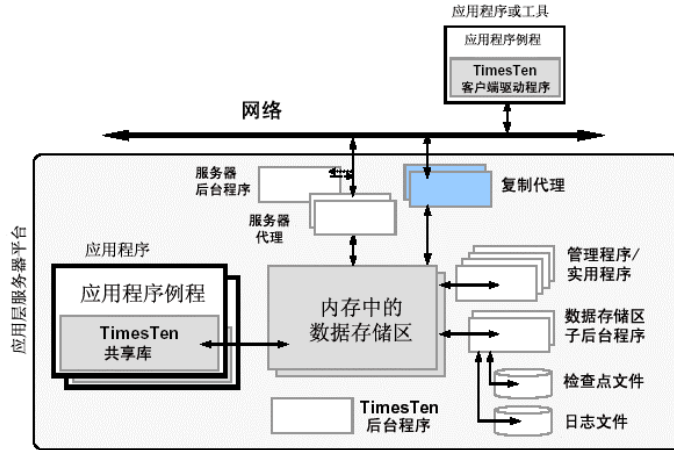


图 3：添加数据复制以获得高可用性

配置复制后，将为每个数据储存区启动复制代理进程。如果为复制而配置了同一服务器上的多个数据储存区，则每个数据储存区将有一个单独的复制代理。每个复制代理能够向一个或多个用户服务器发送更新，并从一个或多个主服务器那里接收更新。这些连接分别在复制代理进程的内部实现为一个单独的执行线程。复制代理通过 TCP/IP 流套接字进行通信。

为获得最高性能，复制代理通过监视现有的事务日志检测数据储存区更新，并在可能的情况下向用户服务器发送批量更新。只复制提交的事务。在用户服务器节点上，复制代理通过高效的低级接口更新数据储存区，从而避免了 SQL 层的开销。

### 高速缓存技术

当使用 Oracle TimesTen 将 Oracle 数据的某些部分高速缓存到内存中数据库时，将创建一个高速缓存组数据库结构来保存高速缓存的数据，应用程序将调用其他共享库例程，同时启动一个系统代理（即高速缓存代理），此代理在高速缓存数据库和 Oracle 数据库之间执行所有异步数据传输。

当使用 Cache Connect to Oracle 选项将 Oracle 数据库的某些部分高速缓存到 Oracle TimesTen 数据存储区时，将创建一个高速缓存组结构来保存缓存数据，应用程序将调用其他共享库例程，同时高速缓存代理在 TimesTen 高速缓存与 Oracle 数据库之间执行异步数据传输

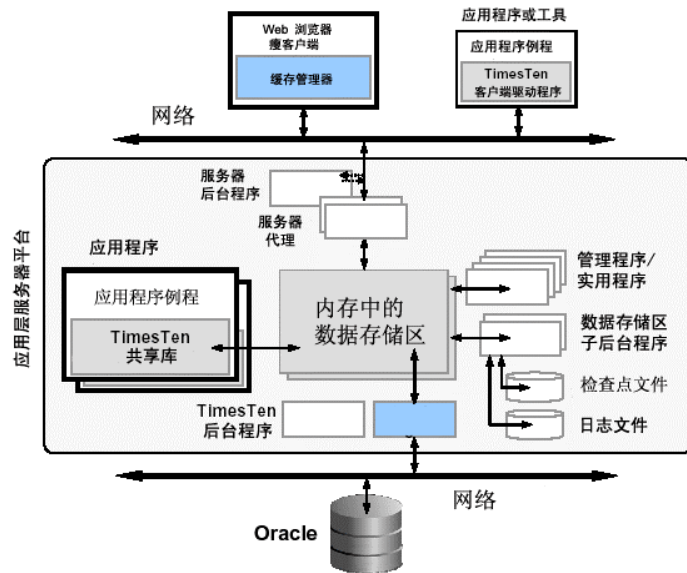


图 4: 为 Oracle 数据添加高速缓存

高速缓存组是由一个或多个通过主键/外键关系以逻辑层次结构排列的表的集合。高速缓存组中的每个表都与 Oracle 数据库表相关。高速缓存组中的每个表都与 Oracle 数据库表相关

高速缓存组是由一个或多个通过主键/外键关系以逻辑层次结构排列的表的集合。高速缓存组中的每个表都与 Oracle 数据库表相关。一个高速缓存组表可以包含相关 Oracle 表中的所有行和列或行和列的一个子集。可以通过基于 **高速缓存管理器** 的浏览器或通过 SQL 语句创建和修改高速缓存组。高速缓存组支持以下特性：

- 应用程序可以对高速缓存组执行读取和写入操作
- 可以自动或手动刷新高速缓存组（将 Oracle 数据库数据置于高速缓存组中）
- 可以自动或手动清理高速缓存组（将高速缓存更新传播到 Oracle 表）
- 可以自动跟踪对 Oracle 表或高速缓存组的更改

如果应用程序更新了高速缓存组中的行，则将 Oracle 表中的相应行作为同一事务的一部分同步更新，或在事务执行后立即异步更新，具体情况取决于所创建的高速缓存组的类型。异步配置显著提高了吞吐量并缩短了响应时间。

如果在同步提交至 Oracle 时因某种原因失败，则回滚事务，从而保持一致性，而异步配置将只记录 Oracle 失败，而不是自动回滚 Oracle TimesTen 事务（因为该事务已经提交）。而异步配置的优点（除性能外）在于，即使在与 Oracle 的连接断开的情况下，它仍将继续运行，并在恢复连接时自动将更新应用于 Oracle。

源自 Oracle 表的更改通过高速缓存代理刷新到高速缓存中。

对高速缓存数据的更新置于表中，并定期由代理放到高速缓存中。用户在高速缓存外部所定义数据的老化也是由代理执行的异步操作。

通过在内存中管理所有数据并对该环境进行优化，内存中数据库技术可以更高效地发挥作用，从而显著提高了性能

## 深入研究 IMDB 技术

内存中数据库技术通过更改有关数据在运行时所在位置的假设提供了引人注目的性能。通过在内存中管理所有数据并针对该环境进行优化，内存中数据库技术可以更高效地发挥作用，从而显著显著提高响应性和吞吐量。

但这种性能的核心是什么？应用程序能否只通过将其所有 RDBMS 数据置于主内存中而获得同一结果？

RDBMS 执行的大部分工作都是基于数据主要位于磁盘中这一假设完成的。优化算法、缓冲池管理以及索引化检索技术都有利于此基本假设。

而 Oracle TimesTen 知道数据位于主内存中，因此可以更直接地路由数据，从而降低了代码路径长度并简化了算法和结构。

在比较这些体系结构时，有一些重要的差别清楚地说明了如何实现数倍的性能改进。为了进行说明，在查询优化算法、缓冲池管理和索引结构中只提供了其中少数几个差别。

## 查询优化

由于磁盘 I/O 的开销远远高于内存访问，因此基于磁盘的 RDBMS 必须假设数据位于磁盘上

基于磁盘的系统与基于内存的系统使用不同的查询优化算法。RDBMS 优化决策依赖于数据主要位于磁盘上这一假设。在动态的运行环境中（数据在任何给定时刻都可能位于磁盘上或缓存在主内存中），基于磁盘的系统必须假设最差的情形。由于磁盘 I/O 的开销远远高于内存访问，因此基于磁盘的 RDBMS 必须假设数据位于磁盘上。它们针对减少性能瓶颈点（即磁盘 I/O）进行了优化。如果这就是所采用的假设，那么基于磁盘的优化器不会总为主要（或完全）位于主内存中的数据生成最佳计划。

而 IMDB 技术知道数据位于主内存中，并基于更简单的假设优化它的查询。它不需要根据磁盘驻留进行最差情形假设，因此它的开销估算可能会更简单并更准确。

## 缓冲池管理

尽管缓冲池在基于磁盘的数据管理解决方案中是必要的，但在基于内存的数据管理中却不是必要的。数据早已位于内存中

在常规的 RDBMS 体系结构中，必须为已经高速缓存到主内存中的数据维护高速缓冲池。当 SQL 查询处理器需要一页数据

时，访问方法必须先要在缓冲池中搜索该内存中数据。即使该数据位于缓冲池中，在许多情况下仍必须将它复制出缓冲池以便随后进行处理。此缓冲池的维护和管理以及额外的数据副本显著增加了将数据提供给应用程序的最初开销。尽管缓冲池在基于磁盘的数据管理解决方案中是必要的，但在基于内存的数据管理中却不是必要的。Oracle TimesTen 没有缓冲池。由于数据已经位于主内存中，因此不需要缓冲池。这样便降低了代码路径的长度和引擎的内存需求，避免了复制，简化了算法并更迅速地将数据传输给应用程序。

此外，这些 RDBMS 产品不会在运行时动态改变这些基于磁盘的假设。基于磁盘的假设在系统代码库中的交错过于紧密，以至于无法使用几个精心设计的 if-then-else 语句来简化它们。在以下两个体系结构的索引树结构中可以看到有关这个相互交错、基于磁盘的假设示例。

## 索引结构

### (i) B+ 树

将数据保存到内存中以后，索引模式的目标是减少 CPU 周期而非 I/O

在典型的 RDBMS B+ 树索引页中，键值和数据指针均保存在 B+ 树条目中。B+ 树节点（磁盘上的页）由许多 B+ 树条目组成，每个条目都保存索引数据值和下一个相应索引节点的页码或保存所搜索到的数据行的磁盘块页码。该结构使得树非常平和宽（非常适于减少磁盘 I/O）。在 B+ 树结构中，主要的目标是减少完成数据文件索引化查找所需的磁盘 I/O 数量。B+ 树实现此目标的方法是：(1) 将键值保存在 B+ 树节点本身中（减少磁盘 I/O），(2) 将尽可能多的索引条目保存到节点中（增加可以用单个 I/O 提供服务的 B+ 树条目数）。

该结构尽管对于磁盘上的数据和索引文件比较适合，但对于内存中的数据则不太适合。将数据保存到内存中以后，索引模式的目标是减少 CPU 周期而非 I/O。CPU 周期用于扩展和比较 B+ 树中的压缩索引值。大量的 CPU 周期还用来管理保存数据的缓冲区以及已经从磁盘读取到内存中的索引。

### (ii) T 树

在 Oracle TimesTen 中，情况要简单一些。T 树针对主内存访问进行了优化。就大小和算法而言，它的索引条目比 B+ 树的更经济。T 树节点之间的连接通过  $\leq$  和  $\geq$  指针实现。这两个指针引用内存位置，而非磁盘上的页。仅仅通过这两个比较，T 树搜索算法便知道它正在搜索的值是位于当前节点上还是位于内存中其他位置了。每次使用新的索引节点指针间接运算符都会使搜索区域减半 – 体现了真正的二分搜索。

T 树索引针对内存中的数据体系结构进行了优化。由于所有行已经位于内存中，因此 T 树不包含键值 - 仅指向实际行（本示例中显示的键值仅用于说明目的）

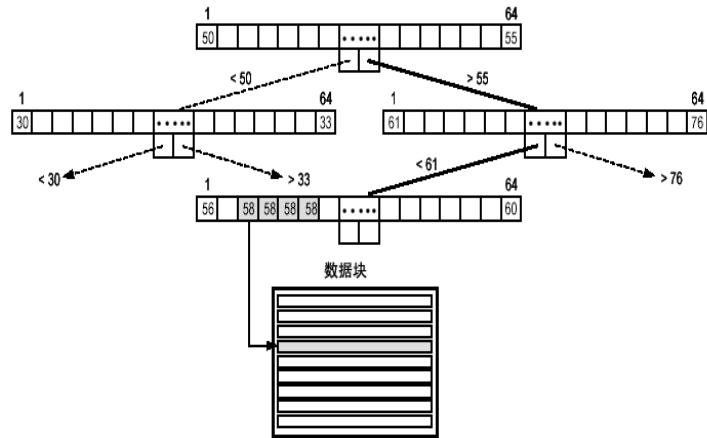


图 5：内存中数据存储区的 T 树索引

删除磁盘驻留假设后，一切变得更简单、更别致、更精巧和更快速

主内存数据管理的目标是减低空间要求、消除磁盘 I/O 以及简化算法、代码路径和内存使用量。T 树减少空间要求的方法是不在索引节点本身中保存键值，而是只指向已经位于内存的数据行中的值。由于索引树结构全部位于内存中，因此树没有磁盘 I/O。T 树是一个非常简单的算法，降低了代码复杂性和代码路径的长度。使用 T 树，您将获得基于 B+ 树访问的所有优势，并可以大量节省移动操作以及缩小大小。

### 差别体现在哪里

前面的体系结构差别示例表明，消除磁盘驻留性假设（或数据位置的模糊性）将显著降低复杂性。计算机指令数至少降低十倍，缓冲池管理不复存在，不需要额外的数据副本，索引页减小，并且它们的结构得到简化。当数据的内存驻留性成为基本事实假设时，一切变得更简单、更简洁、更精巧、更快速。

### (a) 全新的性价比

对于许多应用程序而言，利用内存中数据库技术为获得竞争优势、提高用户满意度以及增加投资回报提供了一个新契机。

将其处理时间的一半用于管理数据的应用程序可以使容量增长近一倍

性能将成倍而非少量增长。将其处理时间的一半用于管理数据并将另一半用于其应用程序区域或业务逻辑的应用程序可以通过使用内存中数据管理产品使容量增长近一倍。就市场策略、体系结构经济和系统选择而言，系统容量增加一倍显著扩大了候选对象的范围。

### 非凡的性能

#### 可伸缩性

Oracle TimesTen 的技术可利用对称多处理器计算机上的多个 CPU。下图显示了 Oracle TimesTen 6.0 在 4 CPU Linux/Intel

系统上的最大事务吞吐量。每个事务执行一个 SQL 更新或选择（读取）操作（如图所示）。该图显示了单 CPU、双 CPU 和 4 CPU 的结果。

Oracle TimesTen 的吞吐量极大，当事务包含一个 SQL 操作时，吞吐量的范围从每秒数万个更新事务到数十万个读取操作。这些结果是在具有 3.0 GHz 处理器的 4 CPU Linux/Intel 系统上记录的

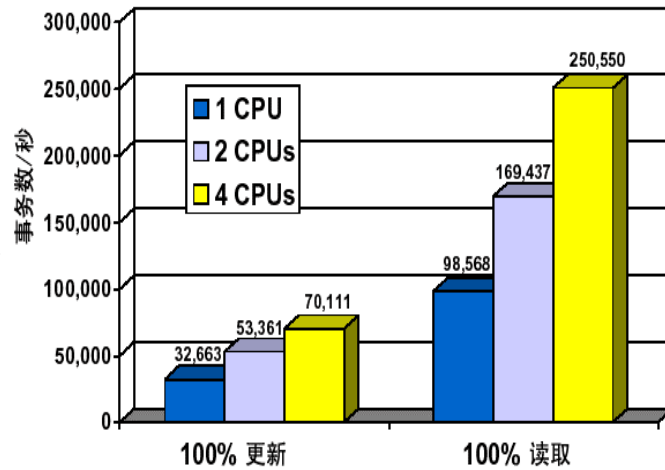


图 6: Oracle TimesTen 吞吐量

最简单也是最高容量的负载是“100% 读取”测试，该测试度量使用键值查找可以检索的单个记录数。在本示例中，平均每秒完成 250,550 次读取操作。在只使用 1 个 CPU 的情况下，速率接近 100,000。

更新操作需要的远非一个读取操作，部分是因为必须为恢复而记录更改。该结果表明，在 4 CPU 上，每秒更新的记录数超过 70,000，而在 1 个 CPU 上，每秒更新的记录数超过 30,000。

尽管这些纯负载不代表任何特定的业务应用程序，但这些结果却证明了 Oracle TimesTen 具有极高的性能，任何实际的混合 SQL 操作都有可能获得最大的吞吐量（从每秒数千次操作到每秒数万次操作不等）。如果需要更高的容量，Oracle TimesTen 可以扩展到使用 4 个以上 CPU 的系统。

#### 响应时间

吞吐量是单个事务响应时间的副产品。随着吞吐量增加，平均响应时间将缩短。在当前系统的各种数据管理负载下，吞吐量极高的 Oracle TimesTen 将获得微秒级的响应时间。1 微妙 = 1/1000000 秒。下面两个图形显示了前面描述的吞吐量负载的平均响应时间。

对于单记录操作，Oracle TimesTen 延迟（响应时间）以微妙（1/1000000 秒）计。数据密集型应用程序由于其总响应时间不仅仅包含数据管理部分而将成为最大受益者。这些结果是在具有 3.0 GHz 处理器的 4 CPU Linux/Intel 系统上记录的

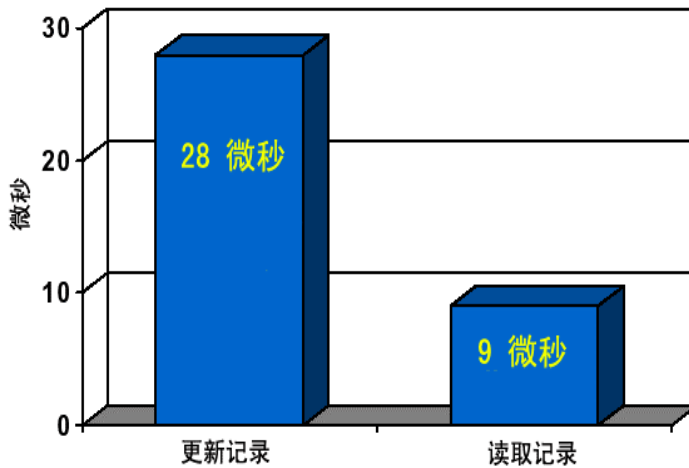


图 7: Oracle TimesTen 延迟

必须注意，事务的响应时间是组成事务的各处理步骤（如业务逻辑、联网、I/O 和数据管理）响应时间的总和。Oracle TimesTen 部分的性能并不是总响应时间的唯一决定因素。有时，某个步骤（通常是 I/O 或联网）将影响其他步骤。

例如，如果使用客户端/服务器网络，而不是在应用程序与数据管理器之间直接建立连接，则网络中的每次往返响应时间将增加 1 毫秒或更多。同样，如果使用同步 I/O 记录更改，则响应时间将包含相对较慢的磁盘操作。在这两种情况下，如果数据管理部分在几微妙内完成，总应用程序响应时间将仍是网络和 I/O 延迟的一个因素。

对于数据管理部分起决定因素的应用程序而言，Oracle TimesTen 技术的高速性将显著缩短响应时间。性能最高的应用程序是指直接连接的、能够使用异步日志记录并频繁进行数据管理的应用程序。典型例子包括频繁执行搜索操作的在线预订系统、基于在线状态和位置的通信服务以及联系中心路由引擎。

通常，许多因素（如事务大小、非数据访问逻辑的数量以及平台性能）将对性能结果产生重要影响。

## 实时功能

本部分介绍了 Oracle TimesTen 产品的功能以及开发人员在编写实时应用程序时使用的接口和特性。

### 数据管理

#### 关系数据库模型

在关系模型中，用户数据存储为表中的行（记录）。行被定义为一列的列（数据字段），每个字段都有一个指定的数据类型

Oracle TimesTen 数据管理器采用了一个真正的关系数据库模型。它不是一个混合模型，也不是一个移植了关系视图的其他模型。熟悉常见 RDBMS 产品的开发人员很快便可以熟练使用该模型

型。可以显式或隐式地创建索引，以便加快对表执行的基于键的搜索或值范围搜索。Oracle TimesTen 支持散列索引以及前面介绍的 T 树索引，并自动实施定义的各种约束，如唯一性或引用完整性（表之间的父子关系）。

在表中，可以指定由一列或多列组成的主键。主键是行的唯一标识符，Oracle TimesTen 将自动禁止插入具有同一主键值的其他行。还可以为每个表指定一个或多个外键，每个外键都与另一表的主键关联。外键建立两个表之间的父子关系。例如，如果某一行有一个主键，而另一个表中的关联外键与该主键的值相同，则不会删除该行。

物化视图是可以定义的可选结构。物化视图是源自常规（基）表的只读表。它可以包含一个或多个表中的部分或所有数据，并可以包含经计算得到的值（如总数和平均数）。由于基表每次更改时系统都会自动更新物化视图，因此物化视图使应用程序能够更快速地访问频繁更新的数据。如果没有物化视图，则每当应用程序需要读取数据时，将需要执行联接和其他计算。物化视图对于支持事件处理（稍后将对其进行介绍）尤其有用。

高速缓存组是由一组表组成的层次结构，这些表映射为一组相应的 Oracle 表并包含某些或所有相同数据。可以对高速缓存组执行读写操作，并且对数据的更新可以在高速缓存组和 Oracle 数据库之间自动传播

如果使用 Cache Connect to Oracle，则可以在内存中数据存储区中定义一个或多个“高速缓存组”。高速缓存组是由一组表组成的层次结构（通过外键实现），这些表映射到一组相应的 Oracle 表并包含一些或所有相同数据。可以对高速缓存组执行读写操作，并且对数据的更新可以在高速缓存组和 Oracle 之间自动传播。

数据管理操作使用行业标准的结构化查询语言 (SQL) 表示。主要操作包括 SELECT(读)、UPDATE、INSERT 和 DELETE。还支持关系联接操作（将多个表中的列组合为一个结果集），并且支持大多数在基准 SQL92 标准中定义的 SQL 选项。

表、索引、物化视图和高速缓存组在数据存储区中创建和维护。同一计算机系统上可能存在多个由同一应用程序访问的数据存储区

表、索引、物化视图和高速缓存组在数据存储区中创建和维护。应用程序连接到数据存储区并对其中的数据请求数据管理操作。同一计算机系统上可能存在多个由同一应用程序访问的数据存储区。但任何单个操作（例如，两个表之间的联接操作）都无法跨越多个数据库。应用程序可以通过 Oracle TimesTen 支持的分两阶段提交的标准 XA（或 JTA for Java）接口参与分布式事务。

前面对 Oracle TimesTen IMDB 技术的介绍解释了数据管理服务的“实时”性。值得重申的是，并不是因为整个数据库全部位于内存中才消除了 I/O — 实际上与此无关。确保数据存储区位于内存中可以免去处理指令，否则这些指令可能导致数据位置模糊不清。这些指令包括间接的搜索算法、缓冲区管理开销、数据副本以及最佳的索引结构和执行计划。

此外，为了实现快速处理并节省内存，还对 Oracle TimesTen 数

数据库内部的信息布局进行了优化。在基于磁盘的 RDBMS 中，磁盘结构和内存结构几乎相同；内存页的大小与 I/O 块的大小相等，从而最大限度地降低了 I/O 延迟。对 TimesTen 而言，这并不重要，因此可以使内存结构最大限度地减少处理指令。

#### 持久性和并发性

Oracle TimesTen 符合数据管理系统的“ACID”属性：原子性、一致性、孤立性和持久性。这些属性确保在多用户系统中，每个事务在运行时就好像此时只有它运行一样，并且系统可以确保不丧失已提交事务的效果。这些属性是数据管理器所需的最严格的属性，而 TimesTen 能够完全具有这些属性。

Oracle TimesTen 符合数据管理系统的“ACID”属性：原子性、一致性、孤立性和持久性

一个常见的误区是，内存中数据管理器无法防止因系统故障而导致的数据丢失。实际上，TimesTen 使用了与常规数据库中持久保存事务和数据的技术相同的技术。在所有面向事务的系统中，可以组合使用更改日志记录以及定期刷新数据存储区的磁盘版来实现持久性。

一个常见的误区是，内存中数据管理器无法防止因系统故障而导致的数据丢失。实际上，TimesTen 使用了与常规数据库中持久保存事务和数据的技术相同的技术。其主要差别在于，TimesTen 为应用程序提供了对持久性和总吞吐量进行权衡的控制

其主要差别在于，Oracle TimesTen 为应用程序提供了对持久性和总吞吐量进行权衡的控制过多的强调某一方将减弱另一方，而 Oracle TimesTen 在权衡二者方面提供了选择。

#### 磁盘操作

如果应用程序要求不能丢失更改，则在提交事务的过程中将把日志记录保存到磁盘上。如果性能最大化的重要性超过丢失一些事务的可能性，则不用频繁地将日志记录写入磁盘（与每次提交事务异步进行）。在任一情况下，Oracle TimesTen 数据管理器都将尝试将多个事务一起“分组提交”，从而最大限度地降低磁盘写操作。

应用程序可以定期请求 Oracle TimesTen 检查数据存储区的磁盘镜像变化。在必要情况下，应用程序可以控制记录检查点的频率，这将决定完成恢复操作所用的时间。应用程序也可以选择允许系统执行自动检查点记录。支持在后台运行的“模糊匹配”检查点，该检查点对正在运行的应用程序的影响很小。从系统故障恢复是指将日志记录与最新的检查点文件合并。

可以在 Oracle TimesTen 中创建永久或临时的数据存储区，并可以将其指定为独占访问或共享访问

可以将 Oracle TimesTen 中的数据库创建为永久数据库或临时数据存储区，并可以将其指定为独占访问（即单用户）或共享（多用户）访问。永久数据存储区（检查点）在不使用时保存在磁盘中。临时数据存储区只位于内存中，并在不使用时将其删除。这两种形式的数据存储区都可以将更改记录到磁盘上或只记录在内存中。日志记录起两个作用。首先，日志记录可以在系统发生故障后根据永久数据存储区恢复事务。其次，在共享访问模式下使用时，日志记录使 Oracle TimesTen 数据管理器可以检测和消除死锁。

为基于磁盘的日志记录配置的数据存储区可以在事务提交（称

为基于磁盘的日志记录配置的数据存储区可以在事务提交操作过程中将日志记录写入磁盘，或允许操作显式确定何时写入日志记录

为持久提交)操作过程中将日志记录写入磁盘,也可以覆盖日志记录,并通过调用内置过程使应用程序能够确定何时写入日志记录。并非所有应用程序都需要持久提交。权衡之处在于性能。采用持久提交的高容量应用程序将导致磁盘到日志的吞吐量出现瓶颈,并很快将耗尽现代处理器的 CPU 带宽。较快的日志设备(如具有电池备份的高速缓存磁盘阵列或固定状态磁盘)将在一定程度减轻瓶颈。

无磁盘数据存储区适用于未连接磁盘存储器的系统,而故障后的恢复必须依赖从数据的远程主副本重新创建数据存储区这一功能

无磁盘数据存储区适用于未连接磁盘存储器的系统,如嵌入式网络处理器。对于多用户访问,将创建和维护内存中日志文件(但并非用于恢复),因此应用程序可以终止并回滚事务,并且可以实施多用户并发控制(如死锁检测)。因此,无磁盘数据存储区可用于读写操作,但故障恢复依赖从数据的远程“主”副本重新创建数据存储区(如通过复制工具维护的另一 Oracle TimesTen 数据存储区或另一数据源)这一功能。

非记录数据存储区提供最快的性能。但在禁用日志记录后,将无法回滚或恢复事务,并且只能在数据存储区级别放置锁(不允许使用行级或表级锁定)。禁用日志只适用于在发生故障时可以从头重新启动的单用户操作。

永久、共享的磁盘记录数据存储区是最常部署的配置。而在应用程序大多使用临时数据的情况下,系统出现故障后恢复数据存储区没有任何意义,这是通常使用临时数据存储区。呼叫中心应用程序中高度动态的“状态”信息就是其中一个例子。

#### 锁定和隔离

Oracle TimesTen 使用锁来防止用户更改其他用户当前正在读取或更改的数据。锁是在数据存储区、表或行等级别置于数据存储区对象中的“保留地”。

锁被放置在数据存储区、表或行等级别的数据存储区对象中。行级锁定提供最大程度的多用户并行性,并且是 Oracle TimesTen 的默认设置

行级锁定提供最大程度的多用户并行性,并且是 Oracle TimesTen 数据存储区的默认设置。应用程序可以在运行时调用过程将锁级别更改为行级或数据存储区级。当 TimesTen 优化器确定其有利或当应用程序调用一个指示优化器在事务持续期间应用表级锁定的过程时将使用表级锁定。

“隔离”指定为响应读取操作而采用的锁行为。应用程序有两个隔离级别可以选择:可串行化或读提交(默认设置)。

应用程序有两个隔离级别可以选择:可串行化或读提交

可串行化隔离用于需要在事务生存期中获得一致的、可预测的数据值的事务。使用可串行化隔离,在提交或回滚事务前,将禁止其他用户更新或删除不常读取的记录。此外在该时刻,如果新记录是这些读取操作结果集的一部分,则不允许其他用户插入这些新记录。换言之,可串行化隔离确保可以使用同一结果重复读取操作。使用数据存储区级锁定,事务将在默认情况下有效地处理可串行化隔离。

读提交隔离获得最大程度的多用户并行性。它确保读取者将只

读提交隔离获得最大程度的多用户并行性。它确保读取者将只看到已提交的数据值,但不用等待写入者释放锁,也不用将锁置于读取的记录之上

看到已提交的数据值，但不用等待写入者释放锁，也不用将锁置于读取的记录之上。为实现此效果，Oracle TimesTen 创建了已更新的一个记录的两个副本：适用于读取者的预更新版本（即“提交的”值）以及适用于写入者的可更新版本。不会阻止读取者访问“读”版本，并且写入者不用等待读取者。

不同的用户可以同时对同一数据存储区使用这两个隔离级别（因为它们兼容）。

### 查询处理

大多数专门为获得高性能设计的产品都需要使用专有的、“隐蔽”的 API。Oracle TimesTen 则不用这样，它的主要目标始终是采用相关的开放式标准。SQL（结构化查询语言）、JDBC（Java 数据库连接）、ODBC（开放式数据库连接）以及 JMS（Java 消息服务）是主要标准，并且是访问数据库的唯一方法。这些接口的实现已经针对 Oracle TimesTen 的体系结构进行了高度调整。

TimesTen 的主要目标是采用相关的开放式标准。SQL、JDBC、ODBC 和 JMS 是应用程序与 Oracle TimesTen 数据库交互所使用的接口。这些接口的实现已经针对 TimesTen 的体系结构进行了高度调整

SQL 多年来一直是关系数据库查询语言的标准。SQL 的主要长处之一就是其从底层存储和索引详细信息中进行抽象。它还是一种易于使用的语言。该语言用于表示做“什么”而“如何做”。SQL 不涉及索引、数据类型或物理布局——而只涉及表、列和搜索条件。Oracle TimesTen 的优化器特性决定了基于各种因素（如是否存在索引以及键值的分布）响应查询的最快方法。

该级别的抽象可以调整或扩展基本数据模型，而不会影响现有的应用程序。只需添加应用程序模块以及所需的任何其他数据表和列便可以为生产环境快速添加新服务。如果没有类似 SQL 这样的使应用程序不受数据管理器内部结构影响的查询语言，大多数应用程序将受到新添加的数据字段的影响。

### 数据复制

所谓的复制就是在数据库之间复制数据。Oracle TimesTen 复制主要是为了使任务关键的应用程序能够在尽可能不影响性能的情况下连续使用数据。除了在故障恢复中的重要作用外，复制在多个数据库之间分配用户负载从而实现最高性能和简化在线升级和维护方面也很有用。

复制通过 SQL 语句进行配置，可以应用于指定的表或整个数据存储区。为提高效率并降低开销，TimesTen 使用一个基于事务日志的复制模式

Oracle TimesTen 遵循“主服务器-用户服务器”复制模型，即将提交的更改从它们的源复制到一个或多个用户服务器数据存储区。复制通过 SQL 语句进行配置，可以应用于指定的表或整个数据存储区。为提高效率并降低开销，TimesTen 使用一个基于事务日志的复制模式。复制使用无磁盘日志记录或基于磁盘的日志记录。

每个主服务器数据存储区和用户服务器数据存储区上的复制都由通过 TCP/IP 流 socket 通信的复制代理控制。主服务器数据存储区上的复制代理从其事务日志中读取记录，并将检测

到的对复制元素的任何更改转发给用户服务器数据存储区的复制代理。然后，用户数据存储区的复制代理将把更新应用于其数据存储区。如果主服务器复制代理转发更新后，用户服务器复制代理并未运行，则主服务器复制代理将把更新一直保存在它的日志中，直到可以在用户服务器复制代理上应用这些更新。

主服务器数据存储区和用户服务器数据存储区使用内部机制确认用户服务器数据存储区已经成功接收和提交更新。这些机制可以确保只在用户数据存储区上应用一次，但它们对于应用程序是完全透明的。

#### 平衡性能和一致性

默认情况下，Oracle TimesTen 复制是一个异步机制。当使用异步复制时，一个应用程序将更新主服务器数据存储区并继续工作，而不等待用户服务器数据存储区接收更新。主服务器数据存储区和用户服务器数据存储区使用内部机制确认用户服务器数据存储区已经成功接收和提交更新。这些机制可以确保只在用户服务器数据存储区上应用一次，但它们对于应用程序是完全透明的。

异步复制提供了最高的性能，但应用程序与被复制的元素在用户服务器数据存储上的接收过程完全分离。Oracle TimesTen 还为需要对复制的数据在主服务器数据存储区和用户服务器数据存储区上的一致性抱有更高信心的“悲观的”应用程序提供了两个 *返回服务* 选项：

- *return receipt 服务* 通过在复制确认用户数据存储区已经接收到了更新前阻止应用程序，使应用程序与复制机制松散耦合或“异步化”
- *return twosafe 服务* 通过在复制确认用户数据存储区上已经收到并提交了更新前阻止应用程序实现完全的同步复制

异步复制提供了最高的性能。TimesTen 还为需要在更高的级别确认复制数据在主服务器数据存储区和用户服务器数据存储区之间一致的应用程序提供了两个“返回服务”选项

使用返回服务的应用程序用损失一些性能的方法来确保主服务器数据存储区和用户服务器数据存储区之间具有更高级别的数据完整性和一致性。在主服务器数据存储区发生故障时，应用程序可以在更大程度上确信主服务器数据存储区提交的事务保存在了用户服务器数据存储区中。与 *return twosafe* 复制相比，*return receipt* 复制需要更少的同步，并且对性能的影响也更小。

#### 复制拓扑

通过将一个数据存储区同时指定为主服务器和用户服务器，可以配置双向复制此外，Oracle TimesTen 还允许多节点“N 路”复制，从而提供了各种可能的复制拓扑，其中包双机热备份配置和负载均衡配置。后者还可以分成分割式负载（其中的每个复制表只有一个主服务器）和分布式负载（其中的每个表有多个主服务器）。在分布式负载配置中，应用程序负责在两个系统之间进行分工，以避免发生复制冲突。如果发生了冲突，则基于时间戳的冲突检测和解决机制可以防止出现不一致的复制。

TimesTen 的复制工具可以实现各种数据分发和同步的拓扑。基本的复制模型是 N 路“主服务器-用户服务器”，并在表级别或数据存储区级别指定了复制。复制可以是单向或双向的（主服务器也可以是其他主服务器的用户服务器），从而实现了一个简单的主备用配置或负载均衡配置。对于相同表具有多个主服务器的情况，TimesTen 使用基于时间戳的冲突检测/解决机制

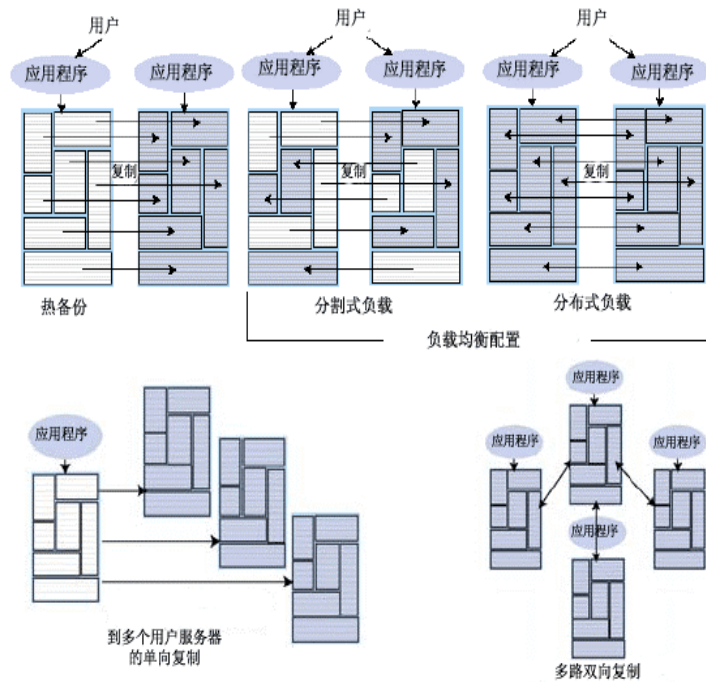


图 8：复制配置选项

还可以定义一个用作 *传播服务器* 的用户服务器，该服务器从主服务器接收复制的更新并将其传递给它自身的用户服务器。传播服务器对通过较低带宽的网络连接（如内联网中的服务器之间的连接）优化复制性能很有用。

### 高速缓存 自定义的集成

Oracle TimesTen 通常与基于磁盘的 RDBMS 部署在一起，当需要实时捕获或处理数据时，将使用 TimesTen。当数据转换到非实时状态（例如，当完成了股票交易或评估了呼叫详细记录的等级后），将把信息从 TimesTen 传到 RDBMS。有多种方法可以实现此集成。

“领先的服务提供商依赖我们的产品来管理时间关键的计费数据，因此无法容忍瓶颈和停机。使用 Oracle TimesTen 和 Oracle 真正应用集群，我们实现了一个可靠的、可伸缩的、经验证的实时数据管理基础来更好地为客户提供服务”

—Ed McKee,  
Interact, Inc. 应用程序总监

应用程序可以连接到 Oracle TimesTen 和 RDBMS，并通过常规的 API 请求移动数据。这是最灵活的方法，但对应用程序来说也是最不透明的方法，它需要复杂的编程。例如，需要高速缓存所有活动用户服务器的应用程序可以在 Oracle TimesTen 中先请求一个记录。如果生成“not found”消息，则连接到 RDBMS 并重复同一请求，然后将结果插入到 Oracle TimesTen 中。随后对该信息的访问将非常快。如果数据有更新，则应用程序可以在两个数据存储区中重复这些更新。真正

的困难是由对 RDBMS 中的数据直接进行更改造成的，它可能会产生高速缓存一致性问题。

由以上方法派生的方法是通过一根发布和订阅消息总线连接 Oracle TimesTen 和 RDBMS, 并编写独立于现有应用程序的新模块（“侦听”要发布到总线的更改并复制从总线中选择的更改）。应用程序可以使用 Oracle TimesTen 的事务日志 API (XLA) 注册并接收对数据存储区更新的通知。大多数 RDBMS 都提供了一个可用于发出更改信号的触发器特性，并提供了一个类似于 XLA 的事务日志 API。

#### Cache Connect to Oracle

集成 Oracle TimesTen 数据存储区和 Oracle 数据库的一个更简单、更透明的方法是通过它内置的连接添加 Cache Connect to Oracle。

与大多数只读的高速缓存不同，Cache Connect to Oracle 支持对 Oracle 数据的高速缓存进行读/写操作，并支持在 Oracle TimesTen 和 Oracle 数据库之间双向传播更新

与大多数只读的高速缓存不同，Cache Connect to Oracle 支持对 Oracle 数据的高速缓存进行读/写操作，并在 Oracle TimesTen 和 Oracle 数据库之间双向传播更新。另一个不同之处是“高速缓存组”这一概念，它描述了一组与 Oracle 数据库中所有表或表子集相对应的 Oracle TimesTen 表。一个高速缓存组可以包含这些 Oracle 表中的所有行和列或行和列的一个子集。

Cache Connect to Oracle 可以控制 Oracle 数据在高速缓存中保存的时间。除了能够设置自动功能之外，可以通过 SQL 语句在需要时加载、刷新、清除和卸载高速缓存组。

Cache Connect to Oracle 与 Oracle 数据库进行交互来执行所有同步高速缓存组操作，如创建高速缓存组、从 Oracle 中加载高速缓存组以及在高速缓存组与 Oracle 之间传播更新。此外，还有一个称为 *Oracle 代理* 的系统进程，该进程执行所有异步高速缓存操作，如将 Oracle 中的更新自动传播给高速缓存组。

Cache Connect to Oracle 应用程序可以通过单一连接向高速缓存组或 Oracle 发送 SQL 语句。此单一连接功能由 *pass-through* 特性启用，该特性检查高速缓存表能否在本地处理 SQL 语句或者是否必须将其重定向到 Oracle。可以在 Oracle TimesTen 高速缓存组或 Oracle 数据库中更新高速缓存的数据。Cache Connect to Oracle 能够将更新从高速缓存组自动传播给 Oracle，并可以从 Oracle 传播给高速缓存组。

应用程序可以通过单一连接将语句发送给高速缓存组或 Oracle。此单一连接功能由 pass-through 特性启用, 该特性检查高速缓存表能否在本地处理 SQL 语句或者是否必须将其重定向到 Oracle

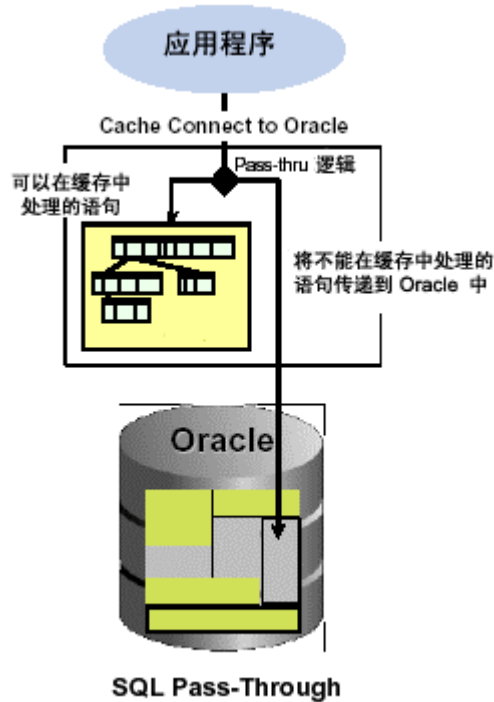


图 9: Cache Connect to Oracle 的单一连接特性

高速缓存组有两个基本类别:

- **系统管理**的高速缓存组, 它提供预先确定的高速缓存行为, 这些行为全部由 Cache Connect to Oracle 管理。
- **用户管理**的高速缓存组, 它允许用户从所有属性和 SQL 语句中进行选择来定义定制的高速缓存行为。用户能够完全控制加载、过期和传播机制。

系统管理的高速缓存组有两种基本类型:

- **直写**高速缓存组在创建时从 Oracle 中加载高速缓存的表数据。然后, 将把高速缓存组的所有更新自动传播给 Oracle。直写高速缓存组可以是异步的

(AWT) 或同步 (SWT) 的。SWT 高速缓存组在高速缓存中提交之前等待 Oracle 提交。AWT 高速缓存组提交高速缓存中的更改而不等待 Oracle 中的提交。

- **只读**高速缓存组保存只读数据, 因此不允许对高速缓存组中的表进行更新。更新必须在 Oracle 中执行。这些更新既可以直接在 Oracle 中完成, 也可以在高速缓存中使用 pass-through 特性重定向到 Oracle。默认情况下, 在 Oracle 中执行更新时, 将自动刷新只读高速缓存组。刷新频率由应用程序控制。

为实现最大的灵活性，高速缓存组行为可以是预定义的或由 Cache Connect to Oracle 通过系统管理的，也可以是用户控制的。系统管理的高速缓存组采用预定义的只读配置或直写配置

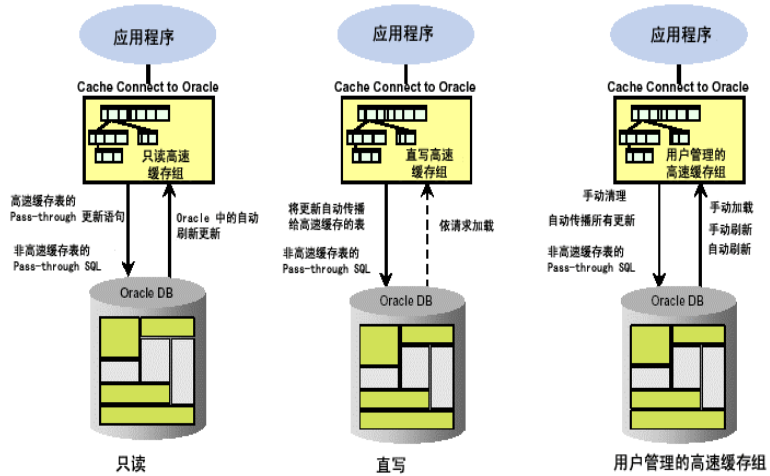


图 10: 系统管理的高速缓存组和用户管理的高速缓存组

## 事件处理

事件处理被定义为能够“感知和响应”具有业务相关性的事件。

根据 Gartner 对实时企业 (RTE) 进行的最新调查<sup>1</sup>，“RTE 监视、捕获和分析对其成功、快速引发事件比较关键的根本原因和公开事件，从而确定新机会、避免异常以及最大限度地降低核心业务流程中的延迟。随后，RTE 将利用该信息逐步消除管理中的延迟及其关键业务流程的执行。”

TimesTen 产品可以通过多种方法支持事件处理应用程序。在最基础的级别，它们可以支持位于网络边缘（即消息和业务事件到达的位置以及可以最先检测到它们的位置）的应用程序。能够在各种计算服务器上与应用程序共存的能力为支持事件处理应用程序提供了灵活性。

但简单事件本身通常没有揭示其相关性，而必须将其与其他信息关联或一直等到事件的模式明显后才能执行。在这些情况下，必须在数据管理器中捕获事件数据（可以与从其他数据源中提取的引用数据进行比较）。识别需要动作的事件后，在许多情况下必须立即做出相应的响应。此类处理比较适合 Oracle TimesTen 产品的功能。

Oracle TimesTen 事务日志 API (XLA) 支持数据存储区更新的检测。应用程序使用 XLA 监视数据存储区更改并根据这些更改采取措施。例如，在以下情况下可能需要通知：

- 客户当天的交易总额超过 1,000,000 美元
- “A”列表上的客户进行了采购

<sup>1</sup> Gartner 更新了它对实时企业的定义，K. McGee，2004 年 3 月 25 日

- 某笔预支付的欠款已经减少
- 某个网络元件已经脱机
- 某个航班更改了登机口

多个应用程序可以同时读取事务日志更新，并且每个应用程序可以在日志文件中维护它自身的“书签”来维护它的位置。书签在数据存储区连接、关机和系统故障中处于不变状态，以便事件通知可以从中断位置继续执行。**XLA** 通常用于为非 **Oracle TimesTen** 数据存储区构建一个自定义数据复制解决方案。

可以使用触发器和存储过程在传统的 **RDBMS** 中实现 **Oracle TimesTen** 的事件处理功能。而 **XLA** 可以降低开销和提高性能，这与实时预期保持一致。此外，当与物化视图功能结合使用时，可以在非常小的数据存储区子集中定向 **XLA** 事件。传统的数据库触发器在对表中的记录进行更改时执行触发器的逻辑。

尽管有广泛用途，但在 **Oracle TimesTen** 中使用的物化视图主要用于细粒度的事件通知，即物化视图将把与要采取操作的特定事件相关的数据放在一个位置。**XLA** 应用程序只需要从一个物化视图中监视相关的更新记录。如果没有物化视图，**XLA** 应用程序将必须监视基表中的所有更新记录，其中包括反映与应用程序无关的行和列更新的记录。

## 结论

随着企业网络传输的信息量不断增多，能否使用实时处理智能地捕获、分析和响应重要事件逐渐成为衡量优秀企业的标准。这不仅仅对业务关键流程的执行和管理很重要。客户希望任何与他们保持重要业务关系的公司能够提供高度定制的交互和最快的响应。

所需要的是一代轻型基础架构软件，它应提供易于使用、功能强大的界面和广泛使用的查询语言 — 可以轻松地与现有后端数据库、消息处理系统以及应用服务器进行通信，可以充分利用当前内存丰富的联网计算平台的全部潜在性能。这正是 **Oracle TimesTen** 提供的用于管理实时数据的新一代基础架构软件。

**Oracle TimesTen** 产品为性能关键系统提供了应用程序层数据管理，并针对快速响应以及实时高速缓存 **Oracle** 数据进行了优化。全球范围内的数百家公司在生产应用程序中使用了 **Oracle TimesTen**，其中包括 **Amdocs**、**Aspect**、**Avaya**、**Cisco**、**Ericsson**、**JP Morgan**、**Lucent**、**Nokia**、**Salesforce.com** 和 **Sprint**。**Oracle** 产品与 **TimesTen** 产品的结合为端到端数据管理提供了独有的单供应商解决方案。



Oracle TimesTen 产品和技术  
2005 年 12 月

Oracle Corporation  
全球总部  
500 Oracle Parkway  
Redwood Shores, CA 94065  
U.S.A.

全球咨询热线:  
电话: +1.650.506.7000  
传真: +1.650.506.7200  
oracle.com

版权所有 © 2005, Oracle。保留所有权利。  
本档只用于提供信息，其中的内容如有更改，恕不通知。本档不保证没有错误，也不受其他任何口头表达或法律暗示的担保或条件的约束，包括对特定用途的适销性或适用性的暗示担保和条件。我们特别声明：拒绝承担与本档有关的任何责任，本档不直接或间接形成任何合约职责。未经预先书面许可，不允许以任何形式或任何方式（电子方式或机械方式）、出于任何目，复制或传播本档。Oracle、JD Edwards、PeopleSoft 和 Retek 是 Oracle Corporation 和/或其子分支机构的注册商标。其他名称可能是其各自所有者的商标。