内存数据库技术在移动实时累加系统中的应用①

邵 璐[1]，费洪晓[2]

[1](中南大学 信息科学与工程学院，长沙 410083)
[2](中南大学 软件学院，长沙 410083)

摘 要：针对移动实时累加系统对海量数据的实时性处理要求，提出了适用于通信行业的内存数据库服务模型，详细阐述了 MDB 服务体系的设计与实现，使其可以屏蔽业务与存储介质的关联度；同时模型采用 TTHA 故障恢复机制，保证内存数据库能够从故障中完整地恢复及实现高可用性。经过对比测试表明，整个服务模型在实时处理海量数据时实现了高效的性能要求。

关键词：内存数据库；实时性；海量数据；故障恢复；实时累加

Application of MDB Technology to Mobile Real-Time Accumulative System

SHAO Lu[1], FEI Hong-Xiao[2]

[1](Institute of Information Science and Engineering, Central South University, Changsha 410083, China)
[2](Institute of Software, Central South University, Changsha 410083, China)

Abstract: An MDB Service Model for Telecommunication is proposed to be used in a mobile real-time accumulative system to process mass data. The proposed model offers the ability of shielding the relationship between business and storage medium. The TTHA failure recovery mechanism makes it possible to make the memory database revert from malfunction and achieve high availability. A comparison experiment is also provided to prove the high performance of our model in a mobile real-time accumulative system.

Key words: memory database; MDB; real-time; mass data; failure recovery

1 引言

1.1 应用背景

实时累加系统作为移动业务支撑系统（BOSS）的关键应用系统之一，主要负责接收计费系统送来的本地用户使用移动通讯业务服务的记录数据，将其转换为三级帐单科目后，以用户为单位进行费用和使用资源量的累计，为后续计算用户使用移动业务的总费用提供基础数据。原有的累加系统仅使用基于磁盘的关系数据库（Disk Resident Database System, DRDB）完成每日一次的定时累加任务，并不具有实时性。但是，近年来手机用户的激增，以及用户对手机业务种类及功能越来越高的要求，不但使得 BOSS 系统的功能越来越复杂，而且对累加系统的实时性和处理海量话费账单的能力提出了更高的要求。

本文针对实时累加系统对海量数据的实时性处理要求，研究探讨内存数据库技术在该系统中的应用，分析内存数据库理论的实际应用经验及技术创新。

1.2 内存数据库概述

内存数据库（Memory Database, MDB）在访问数据时，拥有比磁盘数据库更高的访问效率，可以克服关系数据库的 I/0 瓶颈，使其更适合于需要快速响应和高事务吞吐量的应用环境。不同于磁盘数据库，内存数据库将全部或大部分数据放在内存中[1]，在内存中实现对数据的管理。内存数据库要求数据库“主版本”常驻内存，磁盘版本仅作为“工作版本”的后援[2]，而具有实时特性的内存数据库又要求事务和数据都具有时效性，因而通过采用一套完整的数据库技术策略和机制来确保事务执行过程中所要存取的数据都已驻留内存，从而消除事务执行过程中磁盘的输入输出，最大限度地确保事务和数据的定时限制。

① 收稿时间:2010-11-24;收到修改稿时间:2011-01-10

内存数据库的定义[3]：

设有数据库系统 DBS，DB 为 DBS 中的数据库，DBM(t) 为在时刻 t，DB 在内存的数据集，$DBM(t) \subseteq DB$。TS 为 DBS 中所有可能事务的集合，AT(t)为在时刻 t 处于活动状态的事务集，$AT(t) \subseteq TS$。Dt(T)为事务 T 在时刻 t 所操作的数据集，$Dt(T) \subseteq DB$。若在任一时刻 t，均有：$\forall T \in AT(t) Dt(T) \subseteq DBM(t)$

成立，则称 DBS 为一内存数据库系统，DB 为一内存数据库。

2　MDB技术在移动实时累加系统中的实现

2.1 MDB 服务子系统的设计与实现

MDB 服务子系统是移动实时累加系统的关键子系统，主要完成累加应用程序对内存数据库的访问请求，一方面提供与累加应用程序交互的接口，另一方面可对 TT（Timesten）数据库进行增删改等操作。将外部应用程序与内存数据库的交互独立出来，形成单独的 MDB 服务子系统，有利之处在于屏蔽业务与存储介质的关联度。无论是累加应用程序发生变动，还是 TT 数据库的升级都不会影响到对方的正常使用。同时，增加了 MDB 内存数据库系统的可扩展性。若外部程序新增非累加业务程序，MDB 服务不需任何改动。

2.1.1 MDB 服务子系统的架构

MDB 服务子系统是 MDB 实时累加系统的核心部分，为实时累加提供数据存储及访问支持。

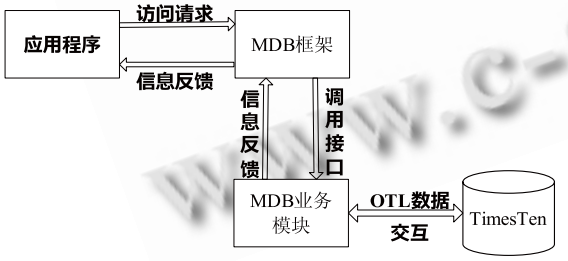

图 1　MDB 子系统的架构

如图 1 所示，它主要由两部分组成：MDB 框架、MDB 访问模块。

(1) MDB 框架

MDB 框架需要启动 MDB 服务，连接 TT 数据库，在指定端口监听对 MDB 的访问请求，启动 MDB 的转期及数据维护线程。停止 MDB 服务及断开 TimesTen 连接。除此之外，MDB 框架还需完成接收外部请求的任务，外部程序可以通过 socket 方法将访问数据包发送给 MDB 框架，框架解包后根据其中的消息类型调用 MDB 业务模块实现数据的处理并通过 socket 将数据打包返回给外部程序。

MDB 框架的实现是由定时脚本及多个接口组成。MDB 服务的启停、端口监听、转期启动和线程维护等由脚本控制，主要利用 TT 数据库现有的函数及 TTsql 命令完成；框架调用 MDB 业务模块完成实质的数据访问则需要调用接口来完成。

(2) MDB 业务模块

MDB 与 TT 数据库的交互是以 OTL 的方式进行，完成各种数据访问和处理操作，并调用 TT 的持久化方法确保数据的安全可靠。提供数据查询、更新以及删除的功能，对不同进程的访问进行同步，对共享数据进行保护。除此之外，MDB 访问模块还要实现转期及其它特定业务逻辑。

OTL(Oracle,Odbc and DB2-CLI Template Library)是一种利用 C++语言编写的访问关系数据库的开源模板库，几乎支持所有的当前各种主流数据库产品。MDB 访问模块利用 TimesTen 提供的 OCI 接口实现对 TT 数据库的操作，通过类 otl_stream、otl_connect、otl_exception、otl_long_string 等实现对数据库的操作。其访问数据库的流程可以简述为：首先调用 otl_connect 的 initialize 方法进行初始化，再利用 try……catch 调用 otl_exception 异常处理机制；然后调用 otl_connect 的 logon 方法登陆数据库，otl_nocommit_stream（非自动提交可以避免自动提交造成的访问冲突等问题）执行 sql 语句；最后调用 otl 的 commit 函数手动提交获取数据或更新数据库，再调用 otl_connect 的 logoff 方法关闭数据库连接。

2.1.2 多进程访问 MDB 数据库

前文中提到 MDB 实时累加系统会同时创建多进程，并且实际应用有着严格的时效性要求，因此 MDB 服务子系统如何控制这些进程访问 MDB 数据库成了一个关键问题。

MDB 服务子系统在 MDB 中建立进程信息表，方便 MDB 记录外部程序状态。如表 1 所示，表中记录了访问 MDB 进程的各种信息。

表 1 MDB 进程信息表

Name	Code	Data Type
进程 ID	PROCESS_ID	TT_BIGINT
MDB 服务线程	SERV_THREAD	TT_BIGINT
Socket 句柄	SOCK_HANDLE	TT_INT
IP 信息	IP_INFO	CHAR(32)
进程状态	PROCESS_STATUS	TT_SMALLINT
进程类型	PROCESS_TYPE	TT_SMALLINT
进程有效日期	BILL_DATE	DATE
状态时间	STATUS_TIME	TT_BIGINT

一个进程在正式访问 MDB 数据之前必须先向 MDB 注册进程信息。根据进程的有效日期 MDB 判断是否接受其访问请求，如果进程有效日期与 MDB 内有效日期相同，则允许访问，其它日期则不允许访问。进程信息表的作用在于 MDB 可以获知有哪些进程还在处理哪一天的数据，从而控制 MDB 自身有效日期的改变。

2.1.3 MDB 转期

MDB 子系统的转期是 MDB 通过调整内部有效日期，确保 MDB 数据处理和外部程序访问的一致性。一方面，MDB 实时累加系统对于数据处理有着很高的实时性要求，另一方面，MDB 实时累加系统各模块的处理不是同步进行的，因此 MDB 必须支持同时保存和处理不同日期的数据，并且对异步进行的相关处理进行同步控制。

MDB 的转期通过转期线程控制，转期线程作为 MDB 服务进程的子线程，由 MDB 框架启动，在 MDB 提供服务的过程中监控 MDB 状态，当满足转期条件时，对 MDB 进行转期操作。

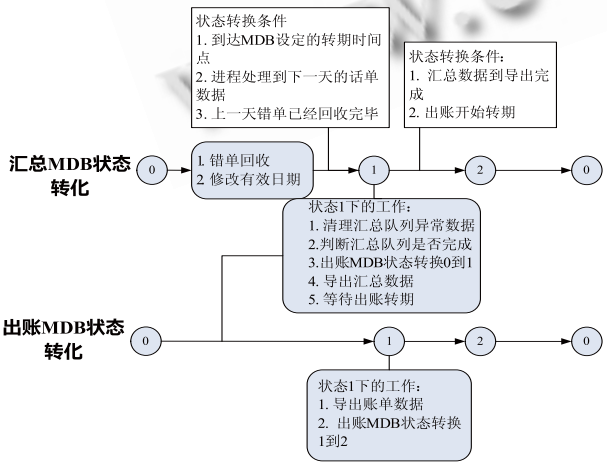

图 2 MDB 转期状态转换图

图 2 描述了 MDB 转期的状态转换条件及准备工作，它主要进行两阶段的衔接控制——累加结果汇总 MDB 转期及出账 MDB 转期。在实际的生产环境中，会出现实时日出账导出的累加结果不准确，为了解决这个问题转期的两个阶段有着衔接条件：汇总转期必需在出账转期完成后才可继续。下面对转期控制的算法做了详细描述。

Step1 ： MDB 服务子系统接收转期请求，判断 MDB 是否到达转期状态 1 的时间点，否则执行 Step8。

Step2 ：检查进程信息表，判断地市汇总 MDB 状态，为 1 则将汇总 MDB 的有效日期调整至下一天，并回收上一天的错单；否则否则执行 Step8。

Step3 ：清除汇总队列异常数据，完成汇总队列。

Step4 ： 出账 MDB 由状态 0 转换至 1。

Step5 ： 导出数据账单及账单明细数据，完成出账 MDB 转期。

Step6 ： 根据汇总表模板生成 sql 及 ctl 文件，并用户 ttbulkcp 导出汇总数据文件。

Step7 ： 汇总 MDB 完成转期。

Step8 ： 空转等待

2.2 **TTHA 故障恢复机制**

内存数据库能够从故障中完整地恢复和高可用性是用户选择数据库时重点的评测指标之一。TTHA（TimesTen High Avaliability）故障恢复机制能确保 TT 数据库由于各种原因发生故障时，快速完整地从故障中恢复，尽可能减少数据丢失，保证数据的完整，将故障造成的损失降到最低。

为了保证系统在发生故障时依然能正常运行，TTHA 故障恢复机制采用主备双机。一个作为主机，一个作为备机，两者部署相同的 MDB 实时累加系统，但是备机在正常状态下不启动 MDB 服务和累加进程，只有其 TT 数据库需要作为主机 TT 数据库的备份启动。除此之外，考虑到内存数据库的易失性，主备机同时与外存数据库（磁盘数据库）连接，每日转期后将内存数据库中的数据导入外存，用于主备同时故障时的恢复。在故障发生时，备机启动 MDB 服务及累加进程，备机转主机根据 log 文件继续运行。

TTHA 故障恢复机制的第二步策略是 TT 同步超时复制。数据库复制技术有两种复制方式：同步复制和异步复制。

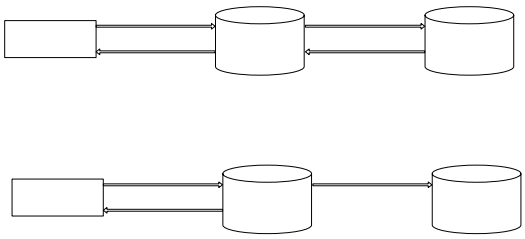

图 3　数据库复制方式 COMMIT

图 3（a）是同步复制方式，数据库事务T提交到主机 A 的 TT 后，A 将交易数据传送给备机 B 的 TT 数据库，B 收到数据后执行提交，返回结果。A 收到反馈信息，根据反馈结果提交或回滚，事务 T 结束。同步复制可以保证 0 丢失数据，并且在主机发生故障时由人工切换，而无需数据恢复。但是由于主机必须等待备机执行成功后才执行，所以备机一旦发生故障，则会阻塞主机的正常运行，即便无故障发生，在主备双机的距离较长时，也会出现延迟。

图 3（b）是异步复制方式，事务 T 提交给主机 A 的 TT 后， A 执行提交，事务 T 结束，同时复制交易数据与 log 文件到备机 B 的 TT 数据库，不关心备机 B 的执行结果。异步复制在备机发生故障时，不会影响主机的正常运行，而且主备长距离连接不会出现延迟，但是主机发生故障主备人工切换时，需要主备机保持数据一致性及完整性，所以异步复制易出现数据丢失的现象。

两种方式各有优缺点，最终采用何种方式，要结合实际应用来决定。MDB 实时累加系统首先要保证结果的正确，所以数据必须 0 丢失。在此基础上，要求高性能、高效率。TTHA 故障恢复机制采用同步复制，但必须解决备机延迟和阻塞的问题。主备双机间通过网络联接，设置 timeout（提交超时），当主机同步复制等待备机执行，等待时间超过 timeout 设定的时间，主机结束等待执行本地提交。在备机异常排除后，再将主机数据复制到备机，保持主备双机的数据一致性和完整性。

3　性能测试及结果分析

为了验证 MDB 实时累加系统的应用效率和性能，需要对比测试应用传统数据库的累加系统和 MDB 实时累加系统的性能，两套系统处理同一批移动话单数据。

MDB 实时累加系统的测试主机配置为两台 IBM P595，每台配置是 64 位 CPU，192G 内存。主机 A 和 B 上各建一套 Datastore，配置成 ACTIVE-STANDBY 的同步模式。软件环境为：AIX UNIX 操作系统+oracle9i。老累加系统的测试主机只有一台，不采用主备双机机制，其他软硬件配置同 MDB 实时累加系统。

两套系统均由计费系统提供已批价话单，因考虑到 MDB 实时累加系统效率，计费系统会将原来应该入库后再处理的话单转成 XDR 文件，直接提交给 MDB 实时累加系统。

图 4 上半部分是某条话单对应的 xdr 文件格式，下半部分的表格是该条话单对应的已入库详单。累加应用程序会对 xdr 文件的各段内容自动进行解析，并且获取其所需的信息。

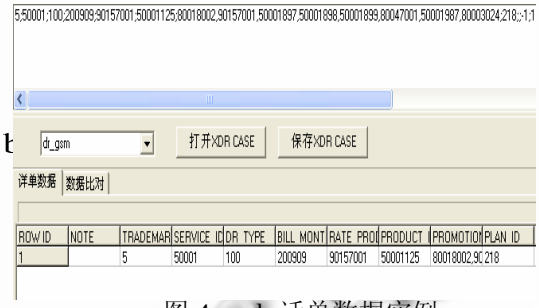

图 4　xdr 话单数据实例

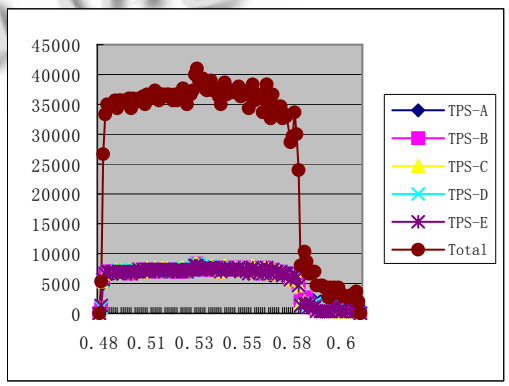

图 5　MDB 实时累加系统测试结果

图 5 及图 6 的横轴表示运行时间，纵轴表示数据吞吐量，TPS-A 至 TPS-E 是数据库根据业务要求分成 5 个库，分别并行数据处理，图中也显示了其各自的运行性能。图中波形的平滑程度显示了两个测试系统

的 I/O 等待情况。

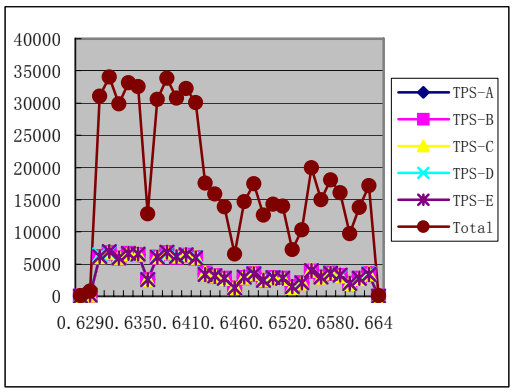

图 6 磁盘数据库累加系统测试结果

图 5 显示的 MDB 实时累加系统在整个数据处理过程中，波形比较平滑，无明显的 I/O 操作，系统的吞吐量也无明显的下降，I/O 瓶颈问题得以解决可以很大的提高系统的效率；而图 6 的 I/O 传输太过频繁，出现了大量的等待 I/O 操作，而且系统吞吐量有明显的下降趋势，这对处理每天上亿条话单的海量数据很不利，无法满足实时高效的要求。

由对比测试的结果可以看出，应用了内存数据库技术的 MDB 实时累加系统可以满足目前实时处理海量话单的要求，可以达到预期的效果。

4 结语

本文研究并详细分析了内存数据库在实时累加系统中的应用技术，以及为适应具体应用环境而作出的创新，并且从测试结果中也可以看出，内存数据库技术在解决海量数据处理及实时应用中有着很大的优势。

目前，电信行业已经进入 3G 时代，随着 3G 技术越来越成熟，移动 BOSS 系统要处理来的数据量必然激增。业务种类的增多、用户群的增长、通信技术的革新等等，都对 BOSS 系统的提出挑战，而本文对内存数据库面向应用的研究，在这种发展背景下有着一定的实用价值。

参考文献

1 王珊,肖艳芹,刘大为,覃雄派.内存数据库关键技术研究.计算机应,2007,27(10):2353–2357.

2 肖迎远.分布式实时数据库技术.北京:科学出版社,2009.94–103.

3 肖迎远,刘云生,廖国琼.主动实时内存数据库系统的数据交换策略及实现.计算机工程与应用,2004,40(29):11–14.

4 Hong DK, Chakravarthy S, Johnson T. Incorporating load factor into the scheduling of soft real-time transactions for main memory databases. Information Systems, 2000,25(4):309–322.

(上接第 187 页)

进，提高了算法的安全性和执行的效率，目前正在试用于我们正在研发的智能 Agent 安全机制[4-6]中。

参考文献

1 Stallings W. Cryptography and Network Security Principles and Practices.北京:电子工业出版社,2006.

2 Van Rossum G. An Introduction to Python. Network Theory Ltd 2003,4.

3 http://pydes.sourceforge.net.

4 Li AN, Zhao ZM. Research and design of security mechanism for multi-grade Agent system. The 11th IEEE International Conference on Communication Technology Proceedings. 2008.777-780.

5 李爱宁,赵泽茂.分布式网络中智能代理的安全迁移机制.计算机工程与科学,2009,31(1):101-103.

6 李爱宁,赵泽茂.基于 RBAC 模型的多等级移动 Agent 系统访问控制机制.计算机系统应用,2009,18(7):23-27.