第 3 章 SanssouciDB——企业内存数据库系统的未来蓝图

摘要：本章概述我们如何实现实时的为业务提供实时数据的美好愿景。我们将阐述如何使用专为企业级应用设计的 IMDB 实现显著的性能优势。此外，还将介绍我们所使用的特殊技术，并阐明这些技术为何是实现特定目标的最佳选择。本章还将展示如何最好地利用最新硬件实现最佳性能，以及如何有效组织数据从而高效使用数据库所在服务器上的大型主存。

在过去的 25 年里，商用 DBMS 的研发宗旨可以概括为“一刀切”[165]。传统的 DBMS 架构旨在支持具有通用数据管理功能的各种不同的应用程序。这些应用程序通常具有不同的特性，并对数据管理软件有着不同的需求，这使得对特定任务的 DBMS 优化难以实现。因此，如今市场上主要的和通用的数据库管理系统虽说“样样精通”，实则“样样稀松”。相反，那些专门为某些应用领域（如文本搜索、文本挖掘、流处理和数据仓储）量身打造的数据库系统，早已获得重大的改进 [165]。如果能够将特定应用领域的特性直接整合并融入到系统架构和数据布局中，则可以显著提升性能[1]。

正如在先前的章节中所强调的那样，我们的愿景是将运营处理和分析处理统一至企业级应用的数据库管理系统中。我们认为，这样做是克服现有企业数据管理解决方案的不足，并满足未来企业级应用需求的基本前提条件 [135] 。

我们将以 SanssouciDB 为例阐述我们的理念。SanssouciDB 是为了统一分析处理和事务处理的原型数据库系统[2]。SanssouciDB 秉承了面向特性的数据库系统的理念，即：它是专门为企业级应用量身定制的系统。我们将举例说明企业级应用的需求如何影响系统中的设计决策。本节将概述 SanssouciDB，此外，还将特别阐述 SanssouciDB 的目标硬件、架构和数据存储。

3.1 重点关注多核和主存

理想状态下，我们希望将企业的整个数据库安装到刀片服务器的单一刀片上，即安装到一台包含多个 CPU 的主板和大型主存模块阵列的计算机上。这样可以简化 SanssouciDB 的设计。但是，直到此书编写完成，市场上最大型的刀片也不支持我们这样做。因此，我们需要使用通过网络相互连接的多个刀片群集来达成所愿（见图 3.1）。

[1] 在下文中，我们将使用术语“特性导向的数据库系统”指代这些专门根据某些特殊应用领域的特性而量身定做的系统。

[2] SanssouciDB 概念的提出基于在 HPI 开发的原型和 SAP 现有的数据库系统。

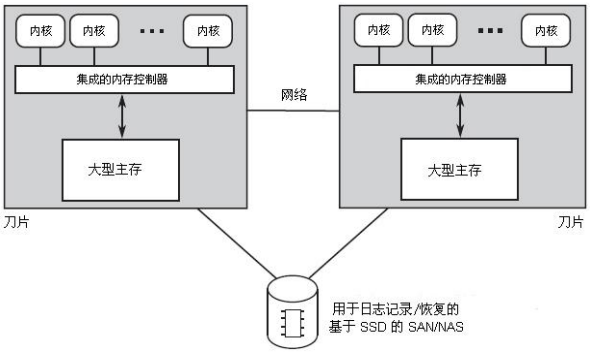

图 3.1 SanssouciDB 的物理架构

数据库系统要在这样的刀片群集上运行，前提是必须实现跨刀片的数据分割和存储。跨刀片管理数据使系统变得更加复杂，例如，必须使用可以跨刀片并且能并行访问分区的分布式查询处理算法。此外，通过网络访问数据的通信成本会高于通过本地刀片访问数据所造成的通信成本。最后，不同的数据分区策略将对查询性能和负载平衡产生影响。因此，我们需要不定期地重新分区，以实现更好的负载平衡，或适应特定的查询工作负载。

另一方面，群集架构也有众多优势：单一刀片法容易造成单点故障，而多刀片法则可带来高可用性。如果某个刀片意外出现故障，则备用刀片可以接管并加载故障刀片中的外部存储中的数据，并将这些数据从外部存储转移至主存中。如果没有备用刀片，则数据可以加载至群集中其余刀片的主存中（有关高可用性信息，请参阅第 5.9 节）。此外，多刀片法还支持横向扩展以防现有刀片数无法处理指定工作负载或数据量。在这种情况下，可以添加更多刀片，进一步分散数据和增强数据的并行处理（有关数据并行处理和可扩展性的信息，请参阅第 4.2 节）。

选择使用多刀片系统作为硬件基础之后，接下来的问题是：是使用大量、性能较低的低端刀片还是使用少量、性能较强大的高端刀片呢？对于 SanssouciDB，我们选择了后者，原因有两点：高端刀片更可靠（请参阅第 5.9 节），所需的数量更少，并且每个刀片都可以管理更大量的数据，支持更多的本地刀片数据处理，从而避免通过网络通信访问远程刀片中的数据而带来的高昂的通信成本。在我们的目标硬件配置中，标准刀片将拥有 2 TB 的主存，且内核多达 64 个。如果拥有 25 个这种刀片，我们便可以处理全球最大公司的所有企业数据。

SanssouciDB 同时选用了无共享/共享内存的方法，即：在每个刀片上运行一个 SanssouciDB 实例，每个实例负责处理特定分区中的数据。因此，从全局角度来看，我们将这种系统定性为“无共享”系统。但是，在单一刀片内，多个内核都会访问存储在刀片内存中的数据。因此，从局部来看，SanssouciDB 又是一个共享内存的系统（请参阅第 4.2 节）。请注意，如图 3.1 所示，所有刀片均与同一台非易失性存储设备相连，例如网络附加存储（NAS）或存储区域网络（SAN）。由此，用户可能会希望将该架构归类为共享磁盘。但是，对于 SanssouciDB 而言，主数据副本十分重要。这是因为主数据副本保存在主存中，而不是非易失性存储设备中，所以我们还是将该系统归类为非共享系统。

为了高效利用此架构，SanssouciDB 在所有级别中都推行并行处理。其中包括分布式查询处理（在各刀片之间）、并行查询处理算法（在刀片的内核之间），甚至在处理器级别中利用特殊的单指令多数据（SIMD）指令。第 4.2 节和第 5.7 节将详细介绍针对这种架构进行的并行处理。此外，在这两节中我们还会介绍 SanssouciDB 中两个重要的分布式和并行查询处理算法。

设计内存系统时，必须考虑的一个重要因素是，刀片上高速缓存和主存的不同延迟时间。第 4.1 节将详细阐述内存层次结构对内存数据管理的影响。最后，为了帮助提高性能，我们采用了基于闪存的存储解决方案，而未使用传统的硬盘，以确保持久性。

3.2 内存数据库系统设计

几乎所有企业级应用都依赖于第一章中简要介绍了的关系数据模型，因此，我们将SanssouciDB 创建为关系数据库系统。SanssouciDB 中存储的关系永久驻留在主存中，这是因为主存的访问速度比磁盘数据库系统的速度明显更快（见表 1.1）。内存数据库系统（IMDB）[56]可以消除传统磁盘数据库系统在读取操作上的主要性能瓶颈。

图 3.2 是 SanssouciDB 的概念示意图。如上所述，SanssouciDB 被设计运行在刀片群集上。在每个刀片上运行一个服务器进程。该服务器进程本身可以运行和管理多个线程，刀片中的每个物理内核均可运行并管理一个线程。调度程序（图 3.2 中未显示）负责分配作业包至不同的线程（请参阅第 5.10 节）。

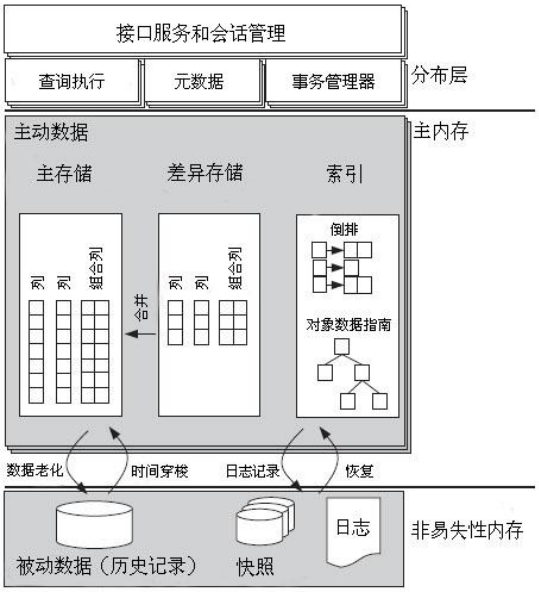

图 3.2 SanssouciDB 概念示意图

为了与客户端和其他服务器进程通信，SanssouciDB 服务器进程拥有接口服务和会话管理器。会话管理器可以跟踪客户端连接和相关参数（如连接超时）。接口服务则提供 SQL 接口并支持存储过程（请参阅第 5.1 节）。此外，接口服务还利用应用程序编程接口（API）转发请求至其他客户端或服务器端进程。该服务在分布层上运行。分布层负责协调分布式元数据处理、分布式事务处理和分布式查询处理。要支持快速查看本地刀片的元数据，分布层必须在不同刀片上的服务器进程之间复制和同步元数据。元数据包含数据表的存储位置及其分区的相关信息。由于数据分割于不同的刀片之中，因此，SanssouciDB 将提供分布式事务和分布式查询处理。此外，分布层还包括事务管理器。在图中，它被标记为“事务管理器”。

在 SanssouciDB 中，主动数据保存在主存中。数据库的主副本包含主存储、差异存储和索引集（请参阅第 3.3 节）。被动数据被保存在非易失性主存中，这有助于确保数据库的持久性。

分布层的问题是有趣而且是非常重要的，SanssouciDB 中的数据接口服务、会话管理和分布技术已经在传统的磁盘系统环境下开发出来了。因此，我们将不再赘述这些问题，但会在第 4.2 节中详细介绍分布式查询处理。在本书的第二部分中，我们将重点阐述内存数据的高效组织方法和访问方法。在以下各节中，我们将做一个简要介绍，并在以后的章节中提供更多的细节。

3.3 SanssouciDB 中数据的组织与访问

从传统上来说，数据库中的数据值以行形式存储，其完整的元组存储在磁盘的相邻块中或主存中[38]。列存储与行存储不同，它以列、而不是行形式存储在相邻块中。这一点在图 3.3 中有所体现。

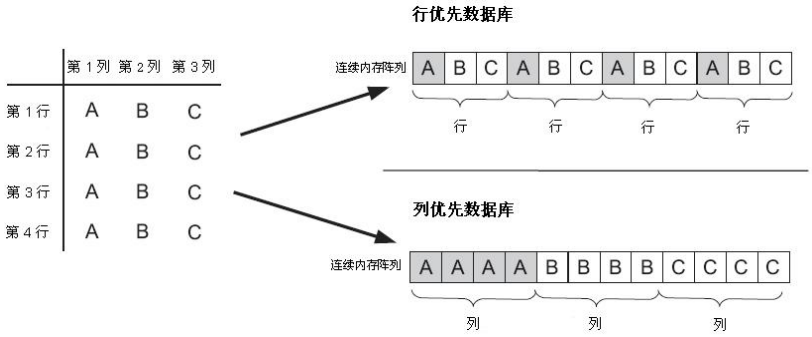

图 3.3 行优先和列优先数据布局

行存储支持快速读取单一元组，但不适合从单一列中读取一组结果。图 3.4 的上半部分以行存储中的行操作和列操作为例，阐述了这一点。

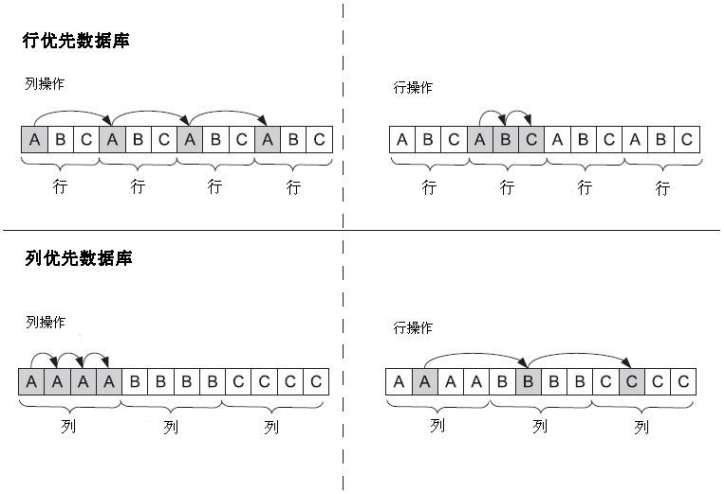

图 3.4 行和列优先数据库的操作

然而，对企业级应用中的数据库访问进行分析的结果表明，这些基于组的读取是最常见的操作类型 [135]，这使得行优先数据库不适合企业级应用程序。另一方面，将数据存储在列、而不是行中的列优先数据库 [2, 1]则非常适合这些基于组的操作。特别是当需要对按列顺序排列的数据进行扫描时，可非常高效地完成。图 3.4 的下半部分证明了这一点。卓越的扫描性能使列优先数据库成为分析处理的明智选择。事实上，许多商用的列优先 DBMS 都是专门针对分析市场而设计的，例如 SAP Business Warehouse Accelerator 和 Sybase IQ。这些系统的缺点在于，基于行操作（例如读取或写入一个完整的元组）的性能往往比较低下。为了结合二者的优点，SanssouciDB 允许将特定的列存储在一起，例如把那些经常查询的列作为一组（请参阅第 4.4 节）。在下面的内容中，我们将这些列组称为“组合列”。通过将这些列类型与列优先数据组织的优势结合在一起，既能确保快速读取，又能确保出色的写入性能。

尽管主存的容量已快速增长，但为了将大型企业级应用程序的所有数据保存在主存中，仍然需要采用数据压缩技术。压缩还能减少在非易失性存储设备和主存之间必须传递的数据量

（如系统启动时）。在 SanssouciDB 列存储中，数据值并不会直接存储，而是通过引用包含不同列值且已排序和压缩后的字典来实现。这些引用经过编码，可进一步压缩以节省空间（请参阅第 4.3 节）。在重复出现多个值（如国家/地区名称）的企业环境中，压缩技术可带来绝佳的压缩率。此外，由于许多操作可直接针对压缩数据进行，所以读取性能也会有相应提高。

然而，如果要向该字典添加新值，则写入性能将变得非常低，这是因为有时需要重新压缩整列。为了解决此问题，如图 3.2 所示，我们使用一种名为“差异存储”的针对写优化的数据结构。差异存储保留那些已添加至数据库，但尚未与主存的数据进行集成的数据。它仅存储对字典的引用，但在这种情况下，字典并未排序，因此无法支持快速更新。差异存储与主存一起表示数据库的当前状态。因此，查询必须同时传递给这两者。由于差异存储保存每条记录所需的存储空间往往超过主存储的空间，因此，我们不希望差异存储的空间增长得过大。为了避免这一点，我们使用一个合并流程，定期将数据从写入优化的差异存储移至主存储中。我们将简要探讨 SanssouciDB 一些其他的关键特性。

当某些查询谓词具有较高的选择性时，即仅当返回所有行中的一小部分时，即使在列存储下频繁扫描这些返回的结果也会带来巨大的开销。对于那些经常使用到高选择性谓词查询的列（例如主键列或外键列），SanssouciDB 支持倒排索引规范（请参阅图 3.2 和第 5.1 节）。

当服务器崩溃或出现电力故障时，IMDB 中的主数据副本将会丢失。为了从这样的场景中恢复，SanssouciDB 会编写日志，并将数据库快照保留在非易失性内存中。这是专门为内存列优先数据库量身定制的，我们将在第 5.9 节中进行详细描述。

由于 SanssouciDB 是专门针对业务应用程序而量身定制的，因此需要经常存储并重建业务对象。这些业务对象采用分层结构并映射至关系模型，从而形成表的层次结构。为了加快业务对象的操作速度，SanssouciDB 可提供对象数据指南。我们将在第 5.3 节进行详细阐述。

为了减少锁需求并保留所有数据库变更的历史记录，我们采用只插入方法（请参阅第 5.6 节）。这样所带来的一个影响是数据量将随着时间的推移而不断增加。我们的目标是始终将所有相关数据保留在主内存中，然而由于新数据随着时间的推移不断增加，这一目标就不可能实现了。为了确保以低延迟的方式访问最新数据，我们利用数据老化算法将数据分为始终保留在主存中的主动数据，以及可以根据需要迁移至闪存存储设备的被动数据（请参阅第 5.2 节）。保留在非易失性存储中的历史存储将负责跟踪这些被动数据。保留历史记录之后，SanssouciDB 可以执行时空穿梭查询，从而反映数据库在用户指定的时间点的状态。

要充分利用 SanssouciDB 改进的性能，必须采用全新的应用程序开发范例，特别是必须将尽可能多的应用程序逻辑推入到数据库中执行（请参阅第 6.1 节）。在第三部分中，我们将对此进行详细说明，并阐述诸如 SanssouciDB 中所使用的数据库架构如何改变企业级应用的编写和使用方式。

3.4 结论

在这一节中，我们介绍了设计企业级应用数据库层的新型架构方法，该架构依托少量高端多核大内存的商业刀片群集。我们的原型数据库系统 SanssouciDB 将表存储在压缩列中，并将主数据副本全部存储在主存中。外部存储器仅用于日志记录和恢复，并支持对历史数据的查询操作。商业应用程序的诸多要求已影响到 SanssouciDB 的设计和功能，其中包括尽量减少锁操作以便支持并发访问数据库，支持时空穿梭查询，并优化业务对象管理。在本书后面的章节中，我们将深入阐述此方法为何是当前以及未来企业级应用的首选。