第一部分

企业级应用的转折点

第 1 章 可取性、适用性、可行性：内存计算技术的影响

摘要：对于支持人机互动的应用程序来说，亚秒级的响应时间和实时分析是关键指标。我们预计，企业级应用的用户将像如今所有互联网用户与 Web 搜索引擎互动一样，很自然地与软件工具互动，可以在初始结果无法满足搜索需求时，实时完善搜索结果。本书的第一章将阐述这一实时提供业务数据的愿景，并介绍它的可取性、适用性和可行性。首先，我们将介绍对实时提供信息的渴求，并阐述企业级应用环境中的亚秒级响应时间。其次，我们将以充分利用现代计算机硬件的内存数据库为基础讨论适用性。最后，我们将从成本角度阐述内存数据管理的可行性。内存计算技术可以显著提高性能。因此，无论是从功能角度，还是从成本角度来说，它都将对企业级应用产生革命性的影响。有了它，企业开发人员可以创建全新的应用程序，而企业用户和管理人员可以用他们喜欢的新方式查看和存储数据。过去，为确保数据处理工作的及时进行，往往必须采取成本高昂的变通措施。而如今，随着性能的提高，这类措施已经没有必要。其中特别值得一提的是，无需再分离运营系统和分析系统。内存计算技术支持对运营数据进行分析，从而简化软件和硬件架构，最终降低总成本。

1.1 实时信息：随时随地获取任何信息

当今的 Web 搜索引擎向我们展现了实时分析大量数据的巨大潜能。用户输入查询问题，即可获得搜索结果。在这一方面，企业级应用力求达到相同的效果，但事实上几乎未能实现。例如，呼叫中心代理或经理需要在公司所有数据源内查找一些特定片段的信息，希望围绕客户制定更明智的产品计划，或者规划未来发展战略。与能够即时查询结果的 Web 搜索相比，企业级应用的运行速度较慢，用户会感到响应时间过长。毫无疑问，如果业务环境中的信息访问速度就像在 Web 搜索引擎环境下一样快速，那么业务用户的行为必定会改变。

Web 搜索和企业级应用的主要区别在于预期结果的完整性。在 Web 搜索中，尽管所有相关数据都会被扫描并在搜索结果中显示，但只有匹配度最高的那些搜索结果才是用户关注的。Web 搜索查询一组数据的索引，评估相关性并提取结果。相比之下，企业级应用必须执行其他数据处理，比如复杂的聚集。在许多应用场景（如分析或计划）中，数据必须在呈现给用户之前就已经准备妥当，特别是当数据来自于不同的源系统时。

为了分析企业数据，并且达到合适的查询响应时间，目前运营系统和分析系统是分开的。分析系统的数据预处理只能作用于整个企业数据集的一部分，这会限制相关报告的数据粒度。根据具体的准备步骤（例如：数据清理、格式设置或计算等），从数据进入运营系统到最终得到报告需要几个小时甚至数天（有关这种分离方式的原因、优势和劣势的详细信息，请参阅第 7.1 节）。当应用程序需要同时进行运营处理和分析时，这种延迟将对性能产生严重影响。例如，

第 2 章中介绍的可承诺量（ATP）、需求计划和催款应用程序需要同时进行运营处理和分析。在运营处理方面，它们必须在最新的数据上运行，并执行读写操作。在分析处理方面，它们需要处理大量数据，因为近期数据和历史数据都是分析所必须的。这些应用程序都将受益于交互式的假设情景分析能力。但目前市场上还没有一个系统能既支持亚秒级的响应时间，又灵活访问系统中的任意信息。

图 1.1 展示了信息“唾手可得”的场景。“信息唾手可得”，是比尔·盖茨 1994 年提出的一个术语，他展望在未来可从任何位置提取任意数据 [58]。该图表明，在不同的地点，与会者都可实时浏览、查询并操作相同的信息。信息交换的时间会大为缩短，同时还能随时响应用户的即席查询。

现在，我们一起来深入探讨亚秒级响应时间、实时分析和快速计算等主题。这些主题对实现上述情景至关重要。

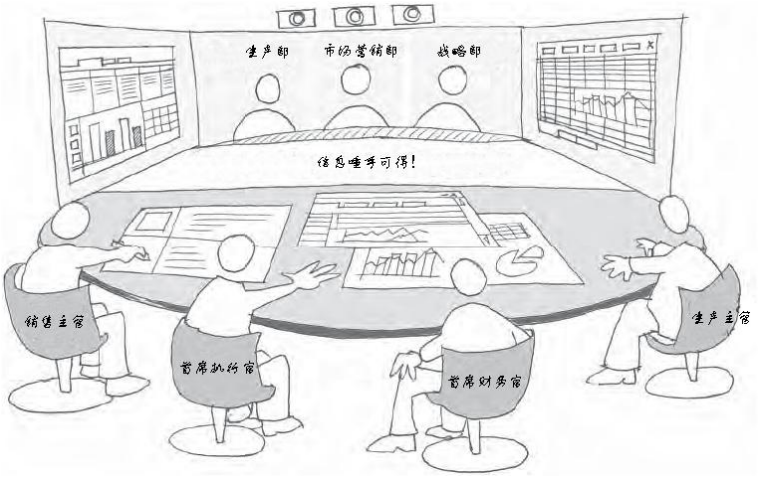


图 1.1 未来的管理层会议 [58]

1.1.1 思想速度般的响应

在 Web 搜索中，用户使用关键词搜索来查询数据。关键词的含义可能含糊不清，因此用户需要根据所获得的结果重新定义搜索词。只有亚秒级的响应时间才会支持这种反复试验的行为。如果企业分析中也能够实现亚秒级响应时间，则用户也能使用相同的方法查询业务数据。准备数据和创建新报告的速度会很快，因此用户就可以交互式的定义和重定义报告查询条件。

通常观察者对刺激作出简单响应的平均反应时间为 220 毫秒[101]。其中一部分时间用于发现刺激，其余时间用于作出响应。识别需要的响应时间更长，因为它还需要理解和领悟。识别的平均响应时间为 384 毫秒。此外，识别响应时间会随着环境复杂程度的增加而增加。在比较复杂的环境中，550 至 750 毫秒时间内做出的响应可以被称为“思想速度般的响应”。经过专业培训、重复执行同一动作的用户的反应时间会更短。因此，较慢的系统响应时间对他们来说会显得更加漫长。

系统响应时间超过人反应时间的部分被视为等待时间。此时，用户会转移注意力。这是一个无法用意识控制的过程。等待的时间越长，用户放弃手头任务的可能性就越大。而亚秒级的响应时间有助于确保用户专注于一个主题，而不会转移注意力。任务切换很容易引起疲劳。即便是小任务，用户也必须找回原先的主题，并回忆其后续步骤。如果可以避免这种任务切换，那么用户可以集中注意力，专注于浏览和分析数据。如果用户可以根据历史查询结果自由地建立查询条件，而不因任务切换分心，他就能在更短的时间内挖掘出数据中更深层的信息。

亚秒级的响应时间意味着我们能够以全新的方式（如通过移动设备）使用企业级应用。移

动设备用户已无法忍受长达几秒的设备响应时间。如果企业级应用实现亚秒级的响应时间，则移动终端的响应时间（包括传输时间）会在可接受的范围内。这样，经理们便可一边等待航班，一边用手机查看催款结果。然后，他们可以直接呼叫最差的债务人，快速解决问题。而与此相比，传统系统的一次催款操作可能需要几个小时。

1.1.2 实时分析和动态计算

实时分析的基础是，所有资源在调用分析时都已就位 [142]。当前，高效的分析报告依赖于专门的物化数据结构（称为“数据立方体”）。这些数据立方体以固定的数据维度为基础。根据这些维度，分析报告可定义其结果集。因此，一个数据立方体只适用于一组特定的报告。如需要其他维度，则必须创建新的数据立方体，或者扩展现有的数据立方体。在最糟糕的情况下，数据立方体维度的线性增长会导致存储需求的指数级增长。如果扩展数据立方体，会导致已经使用该数据立方体的报告性能会下降。因此，需慎重决定是扩展还是重新构建。无论做出何种决定，在系统的生命周期内可能有大量的数据立方体，从而增加存储需求和维护工作。

业务用户应当能够制定即席报表，而不拘泥于预定义报表。它们应当涉及公司拥有的整个数据集，还有可能涉及外部数据源中的数据。假设拥有一个快速的内存数据库，则无需预先计算好的物化数据结构。只要将数据的变更提交到数据库，这些更改便会在报表中体现。如果报表仍然需要执行数据准备和转换步骤，则这些步骤将在查询过程中完成，所有计算都相当快速。报告期间的高速计算以不存储数据、仅提供报表接口的数据立方体为基础，它不但能解决困扰至今的问题，并能优化所有这类分析报表的性能。

1.2 最新硬件趋势的影响

现代硬件的发展一直处于不断变革和创新之中，而其中的最新发展是多核架构以及容量大、价格低的主存[1]的出现。要跟上这些技术的发展潮流，最大程度地挖掘基础硬件的潜能，必须对现有软件系统（如数据库管理系统）进行调整。本节将介绍数据库管理系统和硬件的最新发展趋势。内存数据管理有助于企业级应用实现真正的实时分析，我们将阐述内存数据管理的关键驱动因素。

1.2.1 企业级应用的数据库管理系统

我们认为，数据库管理系统（DBMS）是一系列支持用户创建和维护数据库[48]的程序的统称。DBMS 是一个促进不同用户和应用之间定义、构建、操作和共享数据库流程的软件系统。它是企业级应用中所有操作的基础.从整体上来说，企业级应用的性能很大程度上取决于 DBMS 的性能。

要实现整合分析系统，支持实时访问企业系统任意数据，关键是提高数据库层的性能。在本节中，我们将介绍与企业级应用最密切相关的数据库概念，阐述数据库层的硬件限制因素如何迫使商务智能（BI）应用程序与事务系统分离。此外，我们还将介绍硬件的最新发展趋势，探究如何通过内存计算提高性能。

无论是在学术领域，还是在商业领域，市场对企业级应用的需求已成为 DBMS 发展的主要动力。事实上，最早的商业 DBMS 之一是由 IBM 公司于 1968 年开发的信息管理系统 (IMS) [119]，当时专门用于美国国家航空航天局的“阿波罗太空计划”的库存管理。IMS 是全球第一

[1]主存是指可通过 CPU 直接访问的硅基存储设备，而内存数据管理则是指将数据库的主数据副本存储在主存中。主存是易变的，在电源故障时数据会丢失。

款针对特定应用而构建的典型商业数据库系统。[2]当时，开发复杂软件系统 [19] 意味着 DBMS 要能为尚未开发的应用程序层提供各种通用功能。很明显，通过创建全新的 DBMS，解决每个新企业级应用的需求，是一项耗时且低效的工作。让 DBMS 支持各种企业级应用，这才是明智的选择。因此其发展已成为学术研究和工业生产领域中的共同目标。

1.2.1.1 企业级应用和关系模型

早期系统中使用的 IMS 等层次化数据模型可以很好地处理简单的事务，但涉及对数据库中的数据进行分析时（这是企业级应用的基本要求），这类模型便会暴露出很多缺点。尤其是在整合叶节点中存储的信息时，效率十分低下。在层次较多的大型结构中，这一操作需要占用大量时间。

1970 年，科德（Codd）率先提出了“关系数据模型”的概念，以及以代数算子为基础的一组关系型运算，可以灵活地表示几乎所有实体类型之间的依赖性和关系[27]。科德的模型在学术界迅速获得了认可，但直到 1974 年首次引入“结构化英语查询语言”（SEQUEL）[21]（它支持以相对用户友好的语言编制关系数据的查询公式）时，这种关系模型才得到普遍推广。随后，这种查询语言简称为“结构化查询语言”（SQL）。通过与关系模型结合使用，它可用于各种应用程序。在此模型基础之上建立的系统称为“关系数据库管理系统”（RDBMS）（请参阅第 5.1 节）。

后来，随着“事务”概念的引入，RDBMS 的研发迎来了另一重大突破[61]。事务是指具有明确定义的开头和结尾、有固定执行顺序的一组操作。这一概念[73]催生了术语 ACID ，它可用来描述事务的属性。ACID 是原子性(Atomicity)、一致性 (Consistency)、隔离性 (Isolation) 和持久性 (Durability) 的首字母缩写。“原子性”是指将数据库中的一系列不同操作作为单一操作提供给用户，且所有不同操作均应全部执行，或者全部不执行。“一致性”旨在确保数据库在事务开始之前以及结束之后保持一致状态。系统通过隔离来确保并发事务间的一致性，因为这样才能满足以下要求：只有事务本身才能访问其临时数据，除非事务尚未完成。“原子性”、“一致性”和“隔离性”将影响 DBMS 处理数据的方式。“持久性”是维持成功交易的保障，使其不受任何系统故障的影响。以上都是企业级应用的必要特性。

RDBMS 支持 ACID 事务，这为存储和检索企业数据提供了一种高效可靠的方式。20 世纪 80 年代，基于这类系统的企业级应用可用于处理运营数据（即“联机事务处理”（OLTP）），并执行联机分析处理（OLAP） [170]。在本书的后续章节中，我们将交替使用术语 OLTP 和“运营数据处理”以及术语 OLAP 和“分析处理”。

1.2.1.2 事务处理与分析处理的分离

随着数据量的不断增加，RDBMS 无法再高效地满足所有各类企业级应用的需求，特别是 DBMS 本身无法及时为整个事务数据库提供即席查询服务。

其中一个原因就是，这种数据库模式设计已成为大多数事务型企业级应用的基础。OLTP 模式高度规范化，旨在最大程度地减少数据输入量，加快插入、更新和删除的速度。但当涉及检索数据时，这种高度规范化就成为一个缺点，因为系统可能需要连接多个表才能找到全部所需信息。创建这些连接并从多个表中读取数据会严重影响性能，因为这需要多次读取磁盘的操作。与此同时，分析查询必须访问数据库中很大一部分数据，因此传统的解决方案运行时间过长。

[2]应当指出的是，重新设计的 IMS 版本可以用于很广泛的场景下。

在这种情况下，OLAP 系统应运而生，旨在解决大型企业即时分析数据的需求。这些系统以专用的数据结构 [170] 为基础，旨在优化读取性能，并快速处理复杂的分析查询。这样一来，数据必须从企业的事务系统传输至分析系统，并针对预定义报表作相应处理。

这种传输以周期性批量的方式执行，反复运行“提取、转换、加载”（ETL）流程 [181] 。所需报表可能包含许多来自不同源系统的数据。这些数据必须抽取并转换为一种适用于转换处理的格式。随后，数据在转换阶段必须遵循相应的规则，才能加载到目标 OLAP 系统中。这些规则分别承担各种不同的功能，例如，删除重复数据、排序和聚集。最后，将转换后的数据加载到一个优化的目标模式中，以便快速生成报表。

整个流程的最大局限性在于，由于分析查询的对象是 OLAP 系统中的数据副本，而 OLAP 系统中并不包含最新事务，所以无法执行实时分析。

1.2.1.3 内存计算技术与磁盘技术相比所具有的性能优势

当前的 RDBMS 之所以无法快速执行所需查询，其主要原因在于，系统必须从磁盘中检索查询数据。现代系统广泛使用缓存技术，将经常访问的数据存储在主存中。但是，对于那些处理大量数据的查询事务，仍然需要读取磁盘。磁盘访问和读取操作会花费大量的时间。表 1.1 分别列出磁盘与主存访问和读取的时间（此数据基于 [40]）。

表1.1 磁盘和主存的访问和读取时间

操作	时间
主存访问	100 纳秒
内存顺序读取 1 MB	250 000 纳秒
磁盘寻道	5 000 000 纳秒
磁盘顺序读取 1 MB	30 000 000 纳秒

主存或内存数据库自 20 世纪 80 年代 [56] 就已出现，但动态随机存取存储器（DRAM）的价格直到最近才变得不那么昂贵，因而这类数据库系统成为大型企业系统的一个可行的选择。企业级应用中的数据库层能够快速处理大量数据，这对于我们整合分析系统是至关重要的，它能帮助我们实现让所有业务查询都达到亚秒级响应时间的目标。基于最新硬件的内存数据库可以提供这种功能，它是我们在第 3 章中提出的数据库架构的基础。

1.2.2 主存是新磁盘

由于内存数据库利用服务器主存作为主要的存储位置，因此，主存组件的大小、成本和访问速度是至关重要的。借助数据压缩技术，如今的标准服务器系统已经具备相当大的主存容量，可以容纳任何公司的全部运营数据（请参阅第 4.3 节）。随着单位内存价格的不断降低，以主存作为主要存储的做法越来越有吸引力。这类数据库可以直接对主存访问进行优化，不再需要用特殊的算法来优化磁盘访问。

图 1.2 概括的是主存、磁盘和闪存的价格随时间变化的情况。磁盘和主存的单位容量价格呈指数级下降。例如，2001 年，1 MB 磁盘空间的价格已经跌至 0.01 美元以下，与 1970 年 250 美元以上的价格相比，算是急剧下降。从图中还可以看出，主存的价格也有类似的变化。除了可以将公司的所有业务运营数据全部存放在主存中，并且优化和简化数据访问过程，主存的访问速度也要比磁盘速度快 4 个数量级：一次主存访问需要 100 纳秒[41]，而目前磁盘的读写寻道时间通常为 5 毫秒 [81, 35]。

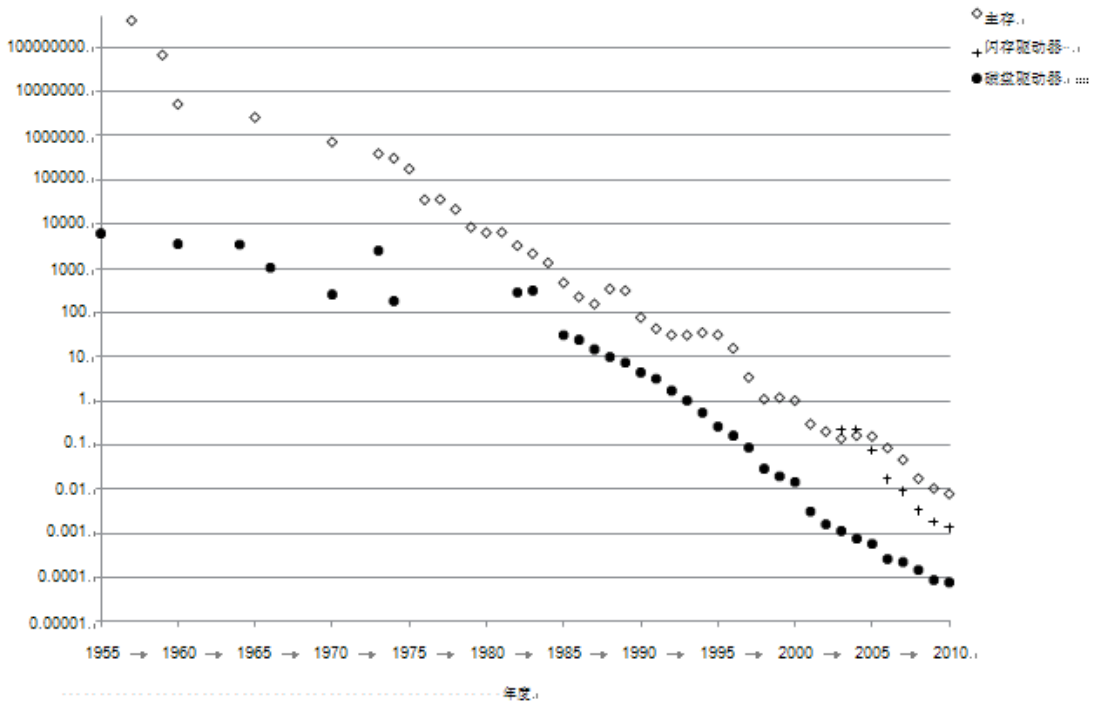


图 1.2 存储价格的历史变化

1.2.3 从最大化 CPU 速度到多核处理器

1965 年，英特尔（Intel）联合创始人之一戈登•摩尔（Gordon E. Moore）曾对集成电路复杂度的增长进行了著名的预测 [117]。该预测被称为“摩尔定律”。如今，它已成为技术迅猛发展的代名词。根据“摩尔定律”，单一芯片上的晶体管数目大约每两年增加一倍 [89]。

事实上，中央处理器（CPU）的性能平均每 20 个月就提高一倍。计算机架构师们取得了辉煌的成就，他们创造了运行速度更快的晶体管，这加快了处理器的时钟速度。此外，还增加了每平方米每个 CPU 上的晶体管数目，由于采用了高效的生产方式并降低了原料损耗，晶体管的价格明显下降。于是，产品性能提升了，而制造成本却和原来大致相同。例如，1971 年，一个处理器包含 2 300 个晶体管，而 2006 年的一个处理器包含了大约 17 亿个晶体管，两者的价格却大致相同。性能提升不仅依靠晶体管数量的增加，还依靠更高效的电路。新一代产品的单核性能比前一代提升两倍，晶体管的数目却大致保持不变。

图 1.3 概括的是从 1971 年到 2010 年处理器时钟速度和晶体管数目的发展情况（以上数据摘自 [77、88、44]）。如图所示，处理器的时钟速度在过去 30 年来一直呈指数增长，直到 2002 年开始停滞。功耗、热分布、热耗散和光速已成为摩尔定律的制约因素 [121]。过去以指数增长的前端总线（FBS）速度，也开始停滞。

2001 年，IBM 推出了第一款能够同时独立计算多个线程的处理器芯片。IBM Power 4 [13] 是专为高端服务器市场推出的产品，它是 IBM Regatta 服务器系列的一部分。Regatta 是多芯片模块的代号，每个模块包含 8 个内核 [128]。2002 年，英特尔推出其专有的超线程技术，这项技术在单一内核中提供线程级的并行处理，优化了处理器的利用率。凭借超线程技术，人们可以从一个物理处理器中创建多个具有重复架构状态的逻辑处理器，这样一来，多个任务几乎可以并行处理，因而可明显提高处理器的利用率。但是，任务其实并没有真正得到并行执行，因为执行资源依旧是共享的，只有在资源使用上可兼容的多条指令才可以在同一个处理步骤中执行。超线程同时适用于单核和多核处理器。

直到 2005 年，在家庭和企业计算机领域中占据主导地位的仍然是单核处理器。在消费市场

上，多核处理器到 2005 年才开始出现，首先出现的是双核处理器，例如 AdvancedMicroDevices（AMD）的 Athlon 64 X2。图 1.4 展示了多核处理器的发展历程和硬件厂商对多核技术未来发展的预测。2006 年秋，英特尔在其开发人员论坛中展示了一款 80 个内核的处理器原型，同年，IBM 推出了含 10 个内核的 CellBroadbandEngine [70]。CellBroadbandEngine 由两种不同的处理单元组成，包括一个双核的 PowerPC 处理单元，和多达 8 个协处理器在所有抽象层次中提供并行处理。2008 年，Tilera 推出了 Tile64，它面向高端嵌入式系统市场、包含 64 个内核 [173] 的多核处理器。3Leaf 也推出了以 HyperTransport 架构为基础的 192 核产品 [3]。预计未来单一芯片上的内核数会更多。Tilera 曾在 2008 年大胆预测，2017 年的嵌入式系统市场将出现 4 096 个内核的芯片；而 Sun 则预测，到 2018 年，服务器将采用含 32 到 128 个内核的芯片 [18]。

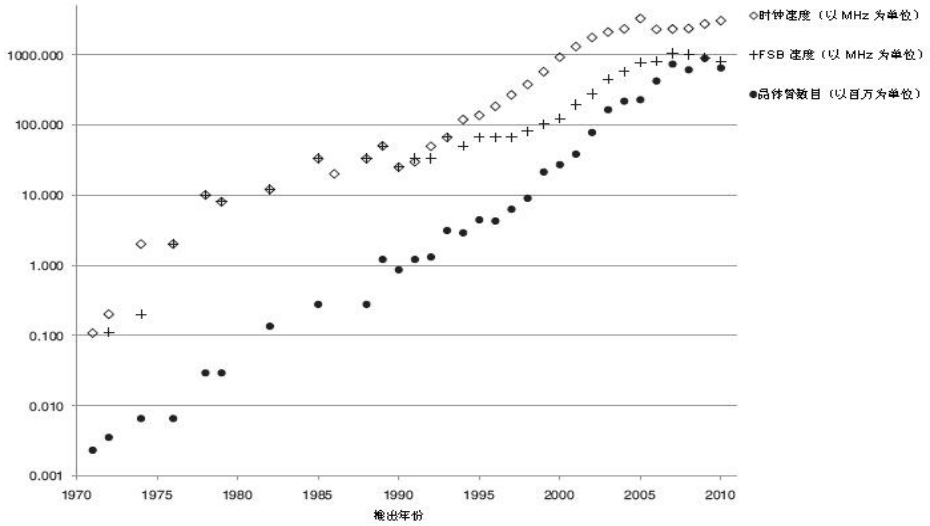


图 1.3 时钟速度、FSB 速度和晶体管数目的发展

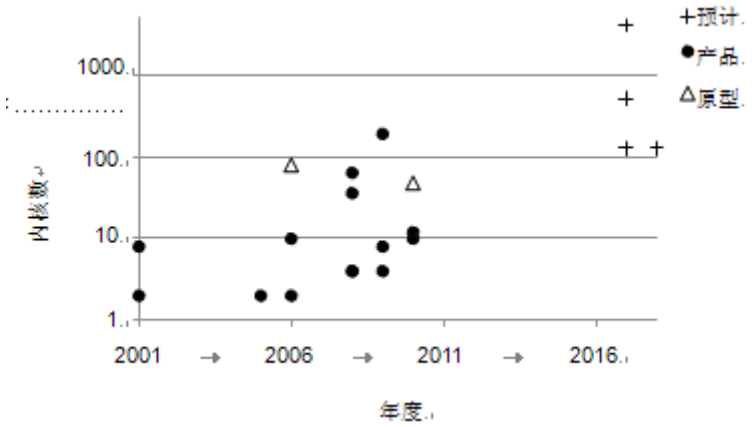


图 1.4 内核数目的发展变化

开发人员在研发当前和未来的软件系统时，必须考虑多核与多处理器计算机的应用。过去几十年来，软件程序一直得益于处理器时钟速度的加快而获得性能的提升，但如今，再也不可

能通过利用处理器技术的进步，提升软件性能的空间，换句话说，“免费午餐的时代结束了”[168]。软件必须结合运用并行化处理，将算法拆分至多个计算单元或同时执行多个运算，即使是在数据库系统中处理同一组数据时也是如此（请参阅第 4.2 节）。随着单一处理单元数量的增加，编程范式得以很好的扩展，现在已经包含了无锁数据结构和算法 [167]（请参阅第 5.6 节）。随着多核架构的发展，内核时钟速度停滞不前的问题也得以弥补，技术进步和软件开发朝着全新的方向迈进。

1.2.4 增加的 CPU 和主存之间的带宽

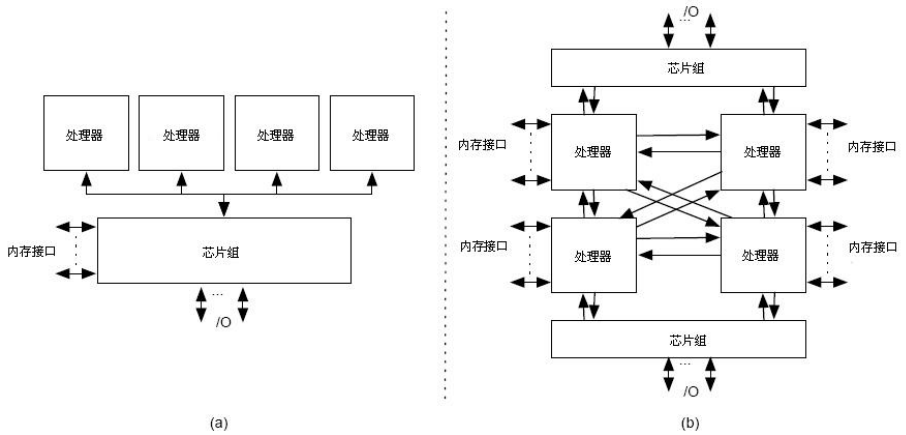
从图 1.3 可以看出，目前 FSB 作为 CPU 与主存和其他所有输入/输出（I/O）组件的唯一接口，其性能的提高已经跟不上处理器性能的指数级增长。计算内存理论吞吐量的重要因素是时钟速度（即“每秒周期数”）和数据总线带宽。

当时钟速度增加，且每台计算机使用多个内核，处理器消化数据的能力与基础设施提供数据的能力之间的差距会加大。尽管这种不断扩大的差距会产生瓶颈，但内存计算和列存储可以利用额外的处理能力。列存储具有很高的压缩比，可以更好地利用带宽。内存数据存储可以利用预取等增强算法来访问数据。在本书后面的章节（第 4 章）中，我们将介绍数据库系统的内存计算和列优先存储。

使用压缩数据和处理压缩数据的算法属于标准技术。实践证明，这项技术完全可以在内核数较少的计算机中处理数据供应的瓶颈问题，但它并不适用于内核数更多的情况。人们曾在列式压缩内存存储和数据密集型应用程序上做过试验，结果表明，在 8 核计算机上，FSB 能得到很好的利用而不至于阻塞。但是，如果在 24 核的计算机上运行同一应用程序，则 FSB 的数据供应能力就无法满足数据处理的需要。从这些试验中我们可以看出，为了避免出现数据供应瓶颈，内核数较多的计算机需要采用新的内存访问策略。处理资源往往不能得到充分利用，而内存延迟和处理器频率之间不断扩大的性能差距又会加剧处理资源利用不足的情况 [37]。

图 1.3 概括了 FSB 速度增加的发展过程。英特尔提高了可用传输速率，使一个周期内可传输的数据量增加了一倍，或在多处理器主板上增加额外的独立总线。2001 年，超微半导体公司 (Advanced Micro Devices, AMD) 推出了 HyperTransport 协议 [3]，将内存控制器集成到处理器中。2008 年下半年，英特尔推出了与 HyperTransport 协议类似的“快速通道互联”（Quick Path Interconnect, QPI）技术[86]。QPI 是内存和多个处理内核间的点对点系统接口，用来取代 FSB。除了使用特殊总线在处理器之间传输数据以外，每个处理器还可使用一个或多个多通道内存控制器来访问主存。英特尔 2007 年推出的 FSB 带宽为 12.8 GB/每秒，而 2008 年推出的 QPI 将可用带宽增加到了 25.6 GB/每秒 [86]。在英特尔的 Nehalem EP 芯片中，每个处理器含 3 条内存控制器和物理内存之间的通道 [54]。在英特尔的 Nehalem EX 芯片中，每个处理器的通道数已增加至 4 个 [55]。

图 1.5 列出了不同架构的概况。图 1.5（b）显示，在 QPI 中，每个处理器均有自己的专有内存。在 4 个处理器的 Intel XEON 7560（Nehalem EX）系统中，基准测试显示每秒的吞吐量已经超过 72 GB [55]。图 1.5（a）显示，本地内存（相邻插槽）和与其他处理单元相邻的远程内存之间的内存访问时间各有不同，这一点和使用 FSB 的情况正好相反。因此，基于 FSB 的架构被称作“一致内存访问”（UMA），而新的架构被称为“非一致内存访问”（NUMA）。缓存一致的非一致内存访问（ccNUMA）和缓存不一致的非一致内存访问系统是有区别的。在 ccNUMA 系统中，所有的 CPU 缓存均共享可用内存的同一视图，并通过由硬件实现的协议确保一致性。缓存不一致的 NUMA 系统则需要软件层来处理内存访问冲突。由于大多数可用的标准硬件都只提供 ccNUMA，因此，我们只详细介绍这一系统。



（a）共享的 FSB；（b）英特尔的“快速通道互联”

图 1.5 FSB 不同架构的概况[35]

要全面利用非一致内存访问（NUMA），应用程序必须主要从与处理器本地连接的内存中加载数据。对于内存受限的应用程序，如果只进行远程内存访问，而不是本地内存访问，可能会导致性能下降达 25% [55]，原因在于内核与非近邻内存间的数据传输可造成 QPI 链路饱和，或是随机访问远程内存时出现高延迟。由于内存缓存和数据预取可以缓解远程内存造成的影响，所以性能并不会全面下降。假设某项作业可拆分为多项并行的任务，在任务并行执行时，数据的分布很重要。只有当执行的任务只访问本地内存时，才能获得最佳的性能。如果数据分布零乱，而且许多任务都需要访问远程内存，处理器之间将会充斥着过量的数据传输。

除了数据密集型应用程序，有些供应商还使用 NUMA 来创建分布式系统的替代品。通过 NUMA，多台物理机可以整合为一台虚拟机。注意这里所说的虚拟机和常用术语“虚拟机”的概念是不同的，常用术语指的是把物理机的一部分作为虚拟机。用 NUMA 可以将多台物理机全面组合为一台虚拟机，用户的感觉就好像在用一台超大型服务器。有了这种虚拟机，所有节点和所有 CPU 的主存均可作为本地资源加以访问。用户可以通过扩展操作系统有效地扩大系统规模，而无需特殊的远程通信（它们需要操作系统或应用程序去处理）。在很多情况下，通过保留部分本地内存缓存部分远程内存，可提高远程内存的访问速度。进一步的研究将证明这些解决方案的性能与分布式解决方案相比，是否能有所提升。例如，3Leaf 是一家使用专业硬件的供应商，而其他公司，如 ScaleMP[120]，则单纯依靠软件解决方案来构建虚拟系统。总之，我们注意到，CPU 内核时钟速度的增长已经趋于停滞，目前主要是依靠在计算机上添加更多的内核来取得技术进步。但正如先前提到的，由于其他的瓶颈依然存在，比如内存访问速度与 CPU 时钟速度之间存在差距，增加内核数目并不能完全解决目前存在的所有问题。压缩技术可减少这种差距的影响，但计算周期却会增加。于是 NUMA 被研发出来，它是一种通过多核进行内存访问的互连策略，可以替代原先的技术。要利用 NUMA 并通过增加更多内核来提升性能，必须采用相应的软件。内存数据库与列数据存储的结合使用，特别适用于多核架构。列数据存储原本就可提供垂直分区，支持运算符并行（请参阅第 4.2 节）。

1.3 通过内存数据管理降低成本

本节将讨论使用内存计算技术进行企业数据管理的可行性以及设置和运行企业系统所需的财务成本。在对成本因素进行概括之后，我们将阐述基于内存计算技术的架构如何帮助企业降低成本。

1.3.1 总体拥有成本

总体拥有成本（TCO）是一个业务公式，用于评估购买和运营资源（在本例中指“企业软件系统”）的生命周期成本。决定购买和实施的软硬件会对企业产生严重的影响，因此准确估算成本相当重要。由于很大一部分成本发生在系统生命周期的后期，仅仅根据设备的购买成本来做 IT 决策是远远不够的，在这种情况下，TCO 衡量指标便应运而生。TCO 分析包括直接成本（如硬件购买成本）和间接成本（如针对最终用户的培训活动成本）[96] 的分析。

引入 TCO 概念的主要目的在于：找出所有隐藏的成本因素，并提供准确而透明的成本模型。这种模型不仅帮助用户尽早发现潜在的成本问题，还可使用户从计算投资回报率或分析成本收益中获益。这些业务公式超越 TCO 分析，还考虑到货币利润或其他收益，因此可以作为技术变更或优化活动的决策依据。

估算 TCO 是一项艰难的任务。具体体现在很难创建准确的成本模型并估算隐性成本。在很多情况下，TCO 分析有助于企业在一次性投资和较高的持续成本之间做出平衡决策，这就是“TCO 平衡法”[39]。通常，集中化和标准化的投资有助于简化运营，降低开销，从而减少支持、升级和培训成本。另一方面，初始投资购买成本高昂的高端硬件，则有助于显著提升系统性能，促进开发、管理和所有其他运营事项。

1.3.2 企业系统中的成本因素

建立和维护企业系统的成本因素包括购买和运营硬件基础设施的成本，购买、管理和运行软件的成本。

1.3.2.1 硬件基础设施和功耗

企业需要什么样的硬件，这取决于应用程序的具体要求。比较复杂的企业级应用，会有许多用户对海量数据执行复杂的计算查询，因此对应用程序的可用性和响应时间都会有很高的要求。要满足这些极具挑战性的要求，通常意味着要使用高端硬件。

另一方面，服务器和冷却系统的电力成本是持续成本的重要组成部分。运行一个大型数据中心，电力成本以及配电和冷却设施的成本约占每月总成本的 30%，服务器成本（以三年为期分摊）约占 60%。以上数字是参照超大型数据中心（约 50 000 台服务器）计算得出的，并假定其电力使用效率非常高 [75]。因此，对于许多小型的数据中心，其电力成本和电力基础设施的成本可能会占据更高的比例。

1.3.2.2 系统管理

系统管理对成本的影响主要取决于它所需要的时间和人力，而这些因素是由系统性能和任务的复杂程度所决定的。首批任务是根据客户的业务结构对系统进行初始设置、配置和自定义。升级和扩展会影响到系统中的每一个组件，当系统变得越来越复杂时，升级和扩展所需的成本也会变得更高昂。为了尽早发现问题（如资源瓶颈），还需要对所有的系统组件进行监控，这项成本也会随着系统复杂度的增加而增加。在升级或扩展现有系统时，扩展和更改复杂系统所涉及的成本和风险是两个主要的限制因素。

管理任务通常包括扫描、转换或复制大量数据。例如，在创建新的企业系统时，经常需要从旧系统中迁移数据。在系统升级和扩展期间，需要对所有完整的表进行转换和重新格式化，还要创建完整的系统副本用于系统备份和测试。

1.3.3 内存计算的性能促进成本降低

如果所有数据都能存储在主存而不是磁盘中，那么数据的操作性能将得到明显改善，特别是在数据量大的情况下。这对系统的各个方面都有影响，包括对硬件组件的选择以及软件架构和基本的软件开发模式。

很多性能的驱动要素，如冗余的物化数据视图，都只是为了优化分析查询的响应速度。其弊端在于，企业必须通过支付大量的费用，维护这种冗余性（请参阅第 7.1 节）。但如果系统运行速度非常快，快到能够处理快速扫描海量数据的请求，那么这些维护费用就显得微不足道了。将数据密集型操作引入数据库，充分利用数据库的功能，并避免把海量的数据传输到数据库外去计算，就可以简化应用程序的软件堆栈。如果数据迁移和海量数据操作的速度能得以加快，系统的复制、备份、存档和升级速度都会自动加快，进而降低成本。

提升海量数据的操作性能，不仅有助于分析查询，还可以改善上述的日常管理任务。我们曾指出，简化软件系统架构可以对成本产生许多积极的影响。简言之，更简洁的系统（例如：架构层次较少、组件较少）往往价格更实惠，设置速度更快，操作、扩展和更改更简便，产生的故障也相对更少。

从我们目前使用内存计算技术进行应用程序开发的经验来看，应用程序代码可减少多达 75%。企业不再需要协调所有层次，如高速缓存和物化，算法已深入至数据库，操作更接近所需的数据。

在用户与系统的交互方面，性能的提升和复杂度的降低能直接降低成本。每当人们必须等待系统做出响应，或投入大量时间来了解错综复杂的流程，这种时间的浪费都会导致成本增加。增加的成本将影响软件生命周期的各个阶段，从软件开发到系统管理，再到业务应用的最终用户。

基于磁盘的高端系统可为数据库计算提供必要的性能，与这类系统相比，内存系统的初始成本和运行成本与之大致相同。基于磁盘的高端系统通过磁盘的冗余提供计算所需的带宽。尽管磁盘能够以更低的成本提供更多的空间，但若要使其性能与内存计算相媲美，还必须强制扩展磁盘数量并加以有效管理。

总之，可以肯定地说，使用内存计算技术有助于降低企业级应用程序的总体拥有成本，因为它不仅可以降低应用程序软件层的复杂度、支持执行数据密集型任务时更靠近数据源，而且能够加快响应速度以帮助用户更高效地利用时间，并快速完成涉及海量数据的管理任务。

1.4 结论

内存计算和多核技术不仅可以提升企业级应用的性能而且还可提升业务价值。在本章中，我们介绍了内存计算技术对企业级应用造成的潜在影响，并阐述为何应当从现在开始这一转变。

首先确定了，在性能提升之后，企业级应用程序可以增加的用户所需的新功能。其中，最主要的是能够直接对事务数据进行分析，而不必另行使用分离的 BI 系统。

在确定这些新功能之后，我们又分析了利用基于内存计算技术重新设计的 DBMS ，在企业级应用中提供这些功能的适用性。我们展现了如何利用当前的硬件趋势来实现愿景，以及这些新硬件的开发如何进一步补充完善我们的数据库设计。

最后，我们评估了基于内存计算技术的 DBMS 会对企业系统的 TCO 产生怎样的影响，并以此来讨论我们方案的可行性。在结尾处，我们描述了这类系统所带来的性能提升如何降低 TCO。