

Subscribe to DeepL Pro to edit this document.  
Visit www.DeepL.com/Pro for more information.

**商业智能(BI)软件**是企业决策支持技术的集合，其目的是使知识工作者(如行政人员、管理人员和分析人员)能够做出更好、更快的决策。在过去的二十年里，无论是在提供的产品和服务的数量上，还是在行业对这些技术的采用上，都出现了爆炸性的增长。这种增长得益于获取和存储大量数据的成本下降，这些数据来源包括银行、零售以及电子商务中的客户交易、用于库存跟踪的RFID标签、电子邮件、网站的查询记录、博客和产品评论。如今企业收集的数据粒度更细，因此数据量也更大。企业正在积极利用其数据资产，部署和试验更复杂的数据分析技术，以推动业务决策，并为客户提供个性化的报价和服务等新功能。今天，很难找到一家没有利用BI技术开展业务的成功企业。例如，BI技术被用于制造业的订单发货和客户支持，零售业的用户特征分析，以在结账时锁定杂货店的优惠券，金融服务的理赔分析和欺诈检测，运输业的用户特征分析等。

商业智能技术是当今企业运行的关键，而这项技术正在经历着巨大的变化。本文**由Surajit Chaudhuri、Umeshwar Dayal和Vivek Narasayy撰写**。在车队管理中，在电信中，用于识别客户流失的原因，在公用事业中，用于电力使用分析，在医疗保健中，用于结果分析。图1显示了在企业内部支持BI的典型架构（阴影框是我们在本文中重点讨论的技术）。执行BI任务的数据通常来自不同的来源--通常来自企业内部各部门的多个业务数据库，以及外部供应商。不同的数据源包含的数据质量不一，使用不一致的表示方式、代码和格式，这些数据必须进行协调。因此，在为 BI 任务做准备的过程中，整合、清洗和标准化数据的问题是相当具有挑战性的。高效的数据加载对于BI来说是势在必行的。此外，BI任务通常需要随着新数据的到来而逐步执行，例如，上个月的销售数据。这使得高效和可扩展的数据加载和刷新能力对企业BI来说势在必行。这些为商业智能准备数据的后端技术共同构成了**关键的洞察力 数据采集和数据存储的成本已经显著下降。这增加了企业获取非常大的数据量的胃口，以便从中获取尽可能多的竞争优势。新的大规模并行数据架构和分析工具超越了传统的并行SQL数据仓库和OLAP引擎。缩短数据获取和决策之间的时间差的需求正在推动商业智能技术的创新。90 acm通讯**｜2011年8月｜第54卷｜No.8篇评论文章被称为提取-转换-加载（ETL）工具。越来越多的人需要以近乎实时的方式来支持商业智能任务，也就是基于业务数据本身做出业务决策。为了支持这样的场景，出现了被称为复杂事件处理(CEP)引擎的专业化引擎。执行BI任务的数据通常被加载到一个称为*数据仓库的存储库中*，该存储库由一个或多个数据仓库服务器管理。存储和查询仓库数据的引擎的一个流行选择是关系数据库管理系统（RDBMS）。在过去的二十年里，已经开发了几种数据结构、优化和查询处理技术，主要用于在大量数据上执行复杂的SQL查询--这是BI的一个关键要求。这种临时SQL查询的一个例子是：寻找在过去一个季度内下过订单的客户，其金额至少超过平均订单金额的50%。大型数据仓库通常会部署并行的RDBMS引擎，这样SQL查询就可以在大量数据上以低延迟的方式执行。随着越来越多的数据诞生数字化，人们越来越希望架构低成本的数据平台，以支持比传统的RDBMS处理的数据量大得多的数据。这通常被描述为"大数据"挑战。在这一目标的推动下，基于MapReduce9范式的引擎--最初是为分析Web文档和Web搜索查询日志而构建的--现在正成为企业分析的目标。目前，这类引擎正在进行扩展，以支持传统企业数据仓库场景所必需的类似SQL的复杂查询。数据仓库服务器由一组*中层服务器*补充，这些*服务器为*不同的BI场景提供专门的功能。在线分析处理(OLAP)服务器有效地将数据的*多维*视图暴露给应用程序或用户，并实现过滤、聚合、钻取和枢轴等常见的BI操作。除了传统的OLAP服务器外，新的"内存式BI"引擎也出现了，它利用当今庞大的主内存大小，极大地提高了多维查询的性能。*报表服务器*可以实现报表的定义、高效执行和渲染--例如，报告今年各地区的总销售额，并与去年的销售额进行比较。文本数据的可用性和重要性日益增加，如产品评论、电子邮件和呼叫中心的文字记录，为BI带来了新的挑战。*企业搜索引擎*支持在仓库中的文本和结构化数据上的关键词搜索范式（例如，查找与特定客户相关的电子邮件信息、文档、购买历史和支持电话），并在过去十年中成为BI的宝贵工具。*数据挖掘引擎*可以对数据进行深入分析，这种分析远远超出了OLAP或报表服务器所提供的范围，并提供了建立预测模型的能力，以帮助回答一些问题，例如：哪些现有客户可能会对我即将进行的目录邮寄活动作出回应？*文本分析引擎*可以分析大量的文本数据（例如，调查回复或客户的评论），并提取有价值的信息，否则需要大量的人工努力，例如，调查回复中提到了哪些产品，以及与这些产品相关的经常讨论的话题。用户通过几种流行的前端应用来执行BI任务：电子表格、用于搜索的企业门户、使决策者能够使用可视化仪表盘跟踪业务关键绩效指标的绩效管理应用、允许用户提出特别查询的工具、数据挖掘模型的查看器等。快速、特设的数据*可视化*可以实现对模式、异常值的动态探索，有助于发现BI的相关事实。此外，还有其他的BI技术(图1中未显示)，如*Web分析*，可以了解公司网站的访问者是如何与网页进行交互的;例如哪些登陆页面有可能鼓励访问者进行购买。同样，客户关系管理(CRM)等垂直打包的应用也被广泛使用。这些应用程序通常支持内置的分析功能，例如，CRM应用程序可能会提供将客户划分为最有可能和最不可能再次购买特定产品的客户的功能。**图1.典型的商业智能架构。典型的商业智能架构。**外部数据**源 数据源 数据移动、流式引擎 数据仓库服务器 中层服务器 前端应用** 操作数据库 提取 转化 加载 (ETL) OLA P服务器 数据挖掘、文本分析引擎 企业搜索引擎 搜索 电子表格 仪表盘 特设查询 报表服务器 复杂事件处理引擎 关系型DBMS MapReduce引擎 评论文章 2011年8月 | 第54卷 | no.**今天，很难找到一个成功的企业没有在其业务中利用商业智能技术。**另一个新兴但重要的领域是*移动商业智能*，它为知识工作者在移动设备上实现新颖和丰富的商业智能应用提供了机会。在这篇短文中，我们无法全面覆盖BI中使用的所有技术（关于其中一些技术的更多细节，请参见Chaudhuri等人5）。因此，我们选择将重点放在研究能够发挥或历史上发挥过重要作用的技术上。在某些情况下，这些技术已经成熟，但仍然存在具有挑战性的研究问题--例如，数据存储、OLAP服务器、RDBMS和ETL工具。在其他情况下，技术相对较新，有一些开放的研究挑战，例如，MapReduce引擎、近实时BI、企业搜索、数据挖掘和文本分析、云数据服务。**数据存储**访问结构。决策支持查询需要进行过滤、加入、聚合等操作。为了有效地支持这些操作，在RDBMS中开发了特殊的数据结构（OLTP查询通常不需要），在此进行描述。后面将讨论不使用RDBMS的专用OLAP引擎中使用的访问结构。*索引结构。*索引可以根据特定列的值进行关联访问。当一个查询有一个或多个过滤条件时，可以通过*索引扫描*（例如，StoreId列上的索引可以帮助检索StoreId=23的所有销售）和*索引交叉*（当存在多个条件时）来利用这些条件的选择性。这些操作可以大大减少，在某些情况下可以消除对基础表的访问需求，例如，当索引本身包含回答查询所需的所有列时。*位图*索引支持高效的索引操作，如联合和交叉。列上的位图索引对该列域中的每条记录使用一个位。要使用位图索引处理一个形式为*column1 = val1* AND *column2 = val2的*查询，我们通过对各自的位向量进行位和来识别符合条件的记录。虽然这样的表示方式对于低卡数域（例如，性别）非常有效，但也可以使用位图压缩来处理高卡数域。*物化视图。*报表查询通常需要汇总数据，例如，最近一个季度和当前财政年度的总销售额。因此，对摘要数据进行预计算和物化（也称为物化视图）有助于显著加快许多决策支持查询。物化视图最大的优势在于它能够通过有效地缓存某些查询的结果来专门针对某些查询。然而这种优势也会限制它的适用性，也就是说，对于稍微不同的查询，可能无法使用物化视图来回答该查询。这一点与索引不同，索引是一种更通用的结构，但其对查询性能的影响可能没有物化视图那么显著。通常情况下，一个好的物理设计包含索引和物化视图的合理组合。*分区。*数据分区可以用来提高性能（稍后讨论）和可管理性。分区允许将表和索引划分为更小、更易管理的单元。加载和备份等数据库维护操作可以在分区上执行，而不是整个表或索引。目前支持的常见分区类型是哈希和范围。先按范围分区，然后在每个范围分区内进行哈希分区的混合方案也很常见。*面向Column的存储方式。*传统的关系型商业数据库引擎是以面向行的方式存储数据的，即表中某行的所有列的值都是连续存储的。30率先使用了面向列的存储方式，其中特定*列*的所有值都是连续存储的。这种方法优化了"大部分读"的临时查询工作负载。面向列的表示方法有两个优点。首先，与面向行的存储相比，数据压缩的程度要大得多，因为一列内的数据值通常比跨列的数据值重复得多。其次，只需要扫描查询中访问的列。在**con92 communications of the acm** | august 2011 | vol. 54 | no.8 review articles trast，在面向行的存储中，不容易跳过查询中没有访问的列。这两点加在一起，会导致扫描大表的时间减少。最后，我们注意到，在过去的十年中，主要的商业数据库系统都增加了自动化的物理设计工具，这些工具可以帮助数据库管理员（DBA）根据*工作负载*信息，如系统上执行的查询和更新，以及约束条件，如分配给访问结构的总存储量，选择合适的访问结构（参见Chaudhuri和Narasayya7的概述）。数据压缩对大型数据仓库有显著的好处。压缩可以减少需要扫描的数据量，从而降低查询的I/O成本。第二，由于压缩可以减少数据库所需的存储量，因此也可以降低存储和备份成本。第三个好处是，由于页面可以以压缩的形式保存，并且只在需求时才进行解压，因此压缩有效地增加了可以缓存在内存中的数据量。第四，某些常用的查询操作(例如，平等条件、重复消除)往往可以在压缩后的数据本身上进行，而不必对数据进行解压。最后，压缩在网络上传输的数据，可以有效地增加可用的网络带宽。这对于数据必须跨节点移动的并行DBMS来说非常重要。数据压缩不仅在关系型DBMS中起着关键作用，而且在其他专业引擎中也起着关键作用，例如在OLAP中。关系型DBMS中使用了不同的压缩技术。*空值抑制*利用了DBMS中几种常用的数据类型是*固定*长度的（例如*int*、*bigint*、*datetime*、*money*），如果为了存储的目的将它们作为可变长度处理，就可以进行显著的压缩。只有值的非空部分和值的实际长度一起存储。*字典压缩*识别数据中的重复值，并构建一个字典，将这种值映射到更紧凑的表示方式。例如，存储订单运输模式的列可能包含字符串值，如"AIR"、"SHIP"、"TRUCK"。每个值都可以通过分别映射到值0,1,2来使用两个比特来表示。最后，与面向行的存储中的压缩方案不同的是，在面向列的存储中，每个值的实例都需要一个条目(可能使用较少的比特)，而在面向列的存储中，其他压缩技术，如*运行长度编码*(RLE)可以变得更加有效。在RLE压缩中，值*v的k个*实例的序列由对（*v*,*k*）编码。当同一值的长运行发生时，RLE特别有吸引力；这可能发生在具有相对较少的不同值的列中，或者当列值被排序时。在数据压缩方面有几个有趣的技术挑战。首先，适合于大型数据仓库的新压缩技术，并与解压和更新成本发生可接受的权衡是很重要的。其次，即使对于已知的压缩技术，仍然存在重要的开放性问题--例如，对于RLE--表的排序顺序的选择会显著影响可能的压缩量。确定使用的最佳排序顺序是一个非平凡的优化问题。最后，是否压缩访问结构的决定是依赖于工作负载的。因此，需要自动化的物理设计工具也能根据工作负载信息推荐哪些访问结构应该被压缩以及如何压缩。**查询处理 在**BI任务中使用的一个流行的概念模型是数据的*多维*视图，如图2所示。在多维数据模型中，有一组*数字度量*是分析的对象。这种度量的例子有销售、预算、收入和库存。每一个数字计量都与一组*维度*相关联，这些*维度*为计量提供了背景。例如，与销售金额相关联的维度可以是产品、城市和销售日期。因此，一个测量可以被看作是多维度空间中的一个值。每个维度由一组属性描述，例如，产品维度可以由以下属性组成：类别、行业、型号、推出年份。一个维度的属性可以通过*层次*关系进行关联。例如，一个产品与其类别和行业属性通过层次关系进行关联（图2）。概念模型的另一个显著特点是强调按一个或多个维度对计量指标进行*汇总*；例如，计算各县每年的销售总额并进行排序。联机分析处理（OLAP）服务器。在线分析处理(OLAP)支持对数据的多维视图进行过滤、聚合、中枢、滚动和钻取等操作。OLAP服务器采用多维存储引擎(MOLAP)；关系型DBMS引擎(ROLAP)作为后端；或称为HOLAP的混合组合来实现。*多维存储引擎（MOLAP）服务器。*MOLAP服务器通过存储引擎直接支持数据的多维视图，使用多维阵列**图2.多维数据。**... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ....**在这里**，我想说的是，我的工作是在一个很好的环境中进行的，我的工作是在一个很好**的环境中进行的。**他们通常会预先计算大型*数据立方体*，以加快查询处理速度。这样的方法具有索引性能好、查询响应时间快的优点，但提供的存储利用率相对较差，尤其是当数据集是稀疏的时候。为了更好地适应稀疏数据集，MOLAP服务器识别数据的密集区域和稀疏区域，并对这些区域进行不同的存储/索引。例如立方体的密集子数组被识别并以数组格式存储，而稀疏区域则被压缩并单独存储。*ROLAP服务器。*在ROLAP中，多维模型及其操作必须被映射成关系和SQL查询。它们依靠前面描述的数据存储技术来加快关系查询处理的速度。ROLAP服务器可能还需要实现SQL不支持的功能，例如，扩展的集合函数，*如中值*、*模式*和基于时间窗口的*移动平均*。ROLAP中使用的数据库设计是为了提高查询和加载数据的效率而优化的。大多数ROLAP系统使用*星型模式*来表示多维数据模型。数据库由一个事实表和每个维度的单表组成。事实表中的每一行都由一个指向每个维度的指针（也就是外键）组成，提供它的多维坐标，并存储这些坐标的数值计量。每个维度表由对应于维度属性的列组成。星型模式没有明确提供对属性层次结构的支持。*雪花模式*(如图3所示)提供了星型模式的细化，其中维度层次结构通过规范化维度表来明确表示。这导致了在维护维度表方面的优势。*HOLAP服务器。*HOLAP架构结合了ROLAP和MOLAP，将数据的存储拆分为MOLAP和关系存储。拆分数据可以用不同的方法。一种方法是像ROLAP服务器一样，将详细的数据存储在RDBMS中，并在MOLAP中预先计算汇总数据。另一种方法是将较近的数据存储在MOLAP中以提供更快的访问速度，而将较老的数据存储在ROLAP中。由于MOLAP在数据密度合理的情况下表现更好，而ROLAP服务器针对稀疏数据表现更好，所以和MOLAP服务器一样，HOLAP服务器也会进行密度分析，以识别多维空间的稀疏和密集子区域。如今所有主要的数据仓库供应商都提供OLAP服务器（例如IBM Cognos,15 Microsoft SQL,17和Oracle Hyperion23）。*内存BI引擎。*技术趋势为一类新的OLAP引擎提供了机会，这类引擎专注于利用大型主内存，以使*特设查询*的响应时间具有互动性。首先，访问磁盘上的数据与内存中的数据的时间比例正在增加。第二，随着64位操作系统的普及，非常大的可寻址内存大小（例如1TB）成为可能。第三，内存的成本大幅下降，这使得拥有大量主内存的服务器能够负担得起。与传统的OLAP服务器不同，内存中的BI引擎（例如QlikView24）依靠一套不同的技术来实现良好的性能。首先，由于详细的数据是内存常驻的，它们避免了访问数据立方体、索引或实体化视图所需的昂贵的I/O。其次，它们使用的数据结构不适合基于磁盘的访问，但对于内存访问却非常有效。例如，考虑一个计算特定状态下每个客户的总销售额的查询。当数据最初加载到系统中时，引擎可以将每个状态的*指针*关联到该状态中的客户，同样，也可以将一个客户的指针关联到该客户的所有订单详情记录。这样就可以实现快速回答查询所需的快速关联访问，这让人联想到面向对象的数据库所使用的方法，以及传统DBMS中的优化，如*join索引*。第三，内存BI引擎通过使用面向列的存储和数据压缩等数据组织技术，可以显著提高在内存中运行的有效数据大小。内存BI引擎最适合于以读为主的数据，而不进行就地数据更新，由于数据解压成本的原因，新数据主要以增量批量插入的形式到达。关系型服务器。关系型数据库服务器（RDBMS）传统上一直作为大型数据仓库的后端。这种数据仓库需要能够针对*非常大的数据库尽可能*高效地执行*复杂的*SQL查询。实现这一目标所需要的关键技术是查询优化，它将一个复杂的查询，并将该查询编译成一个执行计划。为了保证执行计划能够很好地扩展到**图3.雪花模式。OrderNo** OrderDate **Order SalesPersonID** Name City Quota **SalesPerson DateKey** Date **Month Date** Year **Month State Year ProductNo** Name Description Category UnitPrice QOH **Product CategoryName** Description **Category CityName** State **City OrderNo SalesPersonID CustomerNo DateKey CityName ProdNo** Quantity TotalPrice **OrderDetails CustomerNo** Name Address City **Customer 94 Communications of the acm** | august 2011 | vol. 54 | no.8篇综述文章 大型数据库、数据分区和并行查询处理被广泛利用（查询处理技术概述见Graefe13）。因此，我们讨论两块关键技术--查询优化和并行查询处理。*查询优化*技术一直是BI的关键推动者。查询优化器负责选择一个*执行计划*来回答查询。执行计划是由物理运算符(如Index Scan、Hash Join、Sort)组成的，当执行计划被评估后，就会产生查询的结果。查询的性能关键取决于优化器从一个非常大的备选方案空间中选择一个好的计划的能力。对于这种复杂的查询，好的计划和坏的计划之间的执行时间的差异可能是几个数量级的（例如，几天而不是几分钟）。这个主题在数据库研究和工业界一直有着浓厚的兴趣（该领域的概述见Chaudhuri4）。继20世纪70年代末IBM研究公司在System R优化器方面所做的开创性工作之后，下一个重大的架构创新大约在十年之后出现：可扩展优化器。这使得系统设计者可以"插入"新的*规则*，扩展优化器的功能。例如，一个规则可以表示关系代数中的等价性（例如，将一个聚合推倒join以下）。应用这种规则有可能将执行计划转变为执行速度更快的计划。可扩展的优化器使得工业界和研究界多年来开发的许多重要的优化功能能够比较容易地被纳入，而不必反复修改优化器的搜索策略。尽管查询优化取得了成功，并且在BI中发挥了至关重要的作用，但仍然存在许多根本性的挑战。优化器需要解决一个固有的困难问题，即估计计划的*成本，即*计划所做的总工作（CPU、I/O等）。然而，受限于只施加少量开销的要求，优化器通常使用有限的统计信息，如描述列的数据分布的直方图。这种对所有处理器的近似，并可能成为一个瓶颈。对于中小型数据仓库来说，共享磁盘系统的成本效益相对较高。在共享无系统中（例如，Teradata31），数据需要先验地分布在各个节点上。它们有可能扩展到比共享磁盘系统大得多的数据规模。然而，如何有效地将数据分布在各个节点上的决定对性能和可扩展性至关重要。从利用并行性的角度来看，这一点非常重要，同时也可以减少查询处理过程中需要通过网络传输的数据量。数据分配的两个关键技术是分区和克隆。例如考虑一个大型数据库，其模式如图3所示。两个大的事实表*Orders*和*OrderDetails*中的每一个表都可以分别在*OrderId*属性上，也就是在两个表连接的属性上，在所有节点上进行哈希分区。其他所有的维度表，相对较小，可以在每个节点上克隆（复制）。现在考虑一个连接*Customers*、*Orderers*和*OrderDetails的*查询。这个查询可以通过在每个节点上发出一个查询来处理，每个查询对事实数据的一个子集进行操作，并与整个维度表连接。作为最后一步，每个查询的结果通过网络发送到一个节点，该节点将它们结合起来，产生查询的最终答案。*数据仓库设备。*最近出现了新一代的并行DBMS，称为数据仓库设备（例如Netezza19）。设备是一套集成的服务器和存储硬件，操作系统和DBMS软件，专门为数据仓库而*预先安装*和*优化*。这些设备从以下趋势中获得了推动力。首先，由于DW设备供应商控制着完整的硬件/软件堆栈，他们可以提供更有吸引力的一次服务呼叫模式。其次，一些设备将部分查询处理推送到专门的硬件中，从而加快了查询速度。例如，Netezza使用FPGA(现场可编程门阵列)来评估存储层本身的表上的选择和投影运算符。因为有时会导致脆性，因为大的不准确会导致生成很差的计划。有人研究利用查询执行的反馈，通过观察*实际的*查询执行行为（例如，查询表达式的实际结果大小）来克服查询优化器所犯的错误，并在需要时调整执行计划。然而，以低开销收集和利用反馈也是一个挑战，要实现这种方法的优势还需要做更多的工作。*并行处理和设备。*并行性在处理海量数据库上的查询中起着重要作用。关系运算符，如选择、投影、连接和聚合，为并行性提供了许多机会。基本的范式是*数据并行*，即在不相干的数据子集（分区）上并行应用关系运算符，然后将结果合并。Dewitt和Gray10的文章对这一领域的工作进行了概述。几年来，各大数据库管理系统厂商都提供了数据分区和并行查询处理技术。并行有两种基本架构。*共享磁盘*，即每个处理器都有一个私有内存，但与其他所有处理器共享磁盘。*共享无*，每个处理器都有私有内存和磁盘，通常是低成本的商品机。有趣的是，虽然这些架构的历史可以追溯到大约20年前，但这两种架构在业界都还没有出现明显的赢家，而今天这两种架构都存在成功的实现。在共享磁盘系统中，所有的节点都可以通过共享存储来访问数据，因此不需要像共享无的方式那样先验地在各节点之间划分数据。在查询处理过程中，不需要将数据跨节点移动。此外，由于任何节点都可以为任何请求提供服务，因此负载均衡相对简单。但是，有几个问题会影响共享磁盘系统的可扩展性。首先，节点需要进行通信，以确保数据的一致性。通常情况下，这是通过*分布式锁管理器来*实现的，这可能会产生非平凡的开销。第二，网络必须支持组合的I/O带宽。决策支持查询，这可以显著减少需要在DBMS层处理的数据量。使用Map-Reduce范式的分布式系统。基于*Map-Reduce*范式9的大规模数据处理引擎最初是为了分析Web文档、查询日志和点击信息以生成索引和提高Web搜索质量而开发的。基于分布式文件系统并使用*MapReduce*运行时(或其变种如Dryad16)的平台已经成功地部署在集群上，其节点数比传统的并行DBMS多一个数量级。另外，与并行DBMS不同的是，数据必须先加载到具有预定义模式的表中才能进行查询，*MapReduce*作业可以直接在无模式的输入文件上执行。此外，这些数据平台还能够自动处理数据分区、节点故障、管理数据在节点间的流动以及节点的异构性等重要问题。如前言所述，在企业分析中"大数据"挑战的背景下，基于*MapReduce*范式及其变体的数据平台引起了人们的强烈兴趣。使这类平台具有吸引力的另一个因素是，通过以可扩展的方式执行自定义的*Map*和*Reduce*函数，支持对非结构化数据（如文本文档（包括Web抓取）、图像和传感器数据）进行分析的能力。最近，这些引擎已经扩展到支持企业采用所需的功能（例如，Cloudera8）。虽然与成熟的并行 RDBMS 系统相比，严重的企业采用仍处于早期阶段，但在开源 Hadoop14 生态系统的帮助下，使用此类平台的探索正在迅速增长。在提高程序员生产力，同时仍能利用这里所提到的优势的目标的推动下，最近有人努力开发能够将类似SQL的查询，并自动编译成*MapReduce*引擎上的作业序列的引擎(例如，Thusoo等人32)。基于*MapReduce的*分析引擎的出现对并行DBMS产品和研究产生了影响。例如，一些并行DBMS厂商(如Aster Data2)允许在数据库中存储的数据上调用*MapReduce*函数，作为SQL查询的一部分。*MapReduce*函数以表的形式出现在查询中，允许其结果与查询中的其他SQL运算符组成。许多其他DBMS厂商提供了实用程序，用于在基于*MapReduce*的引擎和它们的关系数据引擎之间移动数据。这种桥的一个主要用途是便于将从*MapReduce*平台上的数据分析中提炼出来的结构化数据移动到SQL系统中。近乎实时的BI。当今企业的竞争压力导致了对近实时BI的需求增加。商业智能（也称为运营商业智能或justin- time BI）是减少从获取运营数据到可以对这些数据进行分析之间的延迟。考虑到一家航空公司跟踪其最有利可图的客户。如果一个高价值客户的航班出现长时间延误，主动提醒地面工作人员可以帮助航空公司确保该客户有可能改道。这种近乎实时的决策可以提高客户的忠诚度和收入。实现这种实时BI的一类系统是复杂事件处理（CEP）引擎（例如Streambase29）。企业可以指定他们希望在流式操作数据上检测的模式或时间趋势（称为事件），并在这些模式发生时采取适当的行动。CEP引擎的起源是在金融领域，它们被用于算法股票交易等应用，这需要在股票行情数据上检测模式。然而，它们现在也被用于其他领域，以实时做出决策，例如，点击流分析或制造过程监控（例如，通过RFID传感器数据）。CEP与传统的BI不同，因为操作数据在分析之前不需要先加载到仓库中（见图4）。应用程序定义了声明式查询，可以包含对流数据的操作，如过滤、窗口化、聚合、联合和连接。输入流中事件的到来会触发查询的处理。这些查询被称为"站立"或"连续"查询，因为只要事件继续到达输入流或查询被明确停止，计算就可能被连续执行。一般来说，在同一个流上可能定义了多个查询；因此，CEP引擎面临的挑战之一就是在可能的情况下有效地在各个查询之间共享计算。这些引擎还需要处理**流图4.复杂事件处理服务器架构的**情况。**复杂事件处理服务器架构。**设备 **事件源 CEP服务器 事件源** Web服务器 常态化查询 数据库服务器 数据库服务器 KPI仪表盘 寻呼机 交易站 股票行情 ABC GHI 25.50 50.75 DEF JKL 33.60 45.15 MNO PQR 15.30 25.50 **96 Communications of the acm** | august 2011 | vol. 54 | no.8 评论文章数据的延迟、缺失或失序，这对语义和效率都提出了挑战。在CEP中，有几个公开的技术问题，我们在这里只谈其中的几个。一个重要的挑战是处理引用数据库中数据的连续查询（例如，查询引用数据库中存储的客户表），而不影响近乎实时的需求。在流数据上优化查询计划的问题有几个开放的挑战。原则上，由于查询是"永远"执行的，因此改进查询执行计划的好处是无限的。这为比传统DBMS中可行的更彻底的优化提供了可能。此外，能够在较长的时间内观察执行计划中操作符的执行情况，对于识别次优计划具有潜在的价值。最后，实时分析的重要性日益增加，这意味着许多传统的数据挖掘技术可能需要在流数据的背景下重新审视。例如，需要在数据上进行多次传递的算法对于流式数据不再可行。**企业搜索** BI 任务经常需要在企业内部搜索不同类型的数据。例如，准备与客户会面的销售人员希望在会面前了解相关的客户信息。这些信息如今被隔离在不同的来源中。CRM数据库、电子邮件、文档和电子表格，既存在于企业服务器中，也存在于用户的桌面上。越来越多的有价值的数据以文本的形式存在，例如，产品目录、客户邮件、数据库中销售代表的注释、调查回复、博客和评论。在这样的场景下，能够利用关键词搜索范式对所需信息进行检索和排序，对BI来说是非常有价值的。企业搜索的重点是在文本库和结构化企业数据上支持我们熟悉的关键词搜索范式。这些引擎通常利用结构化数据来实现*面状*搜索。例如，它们可能会在搜索结果中对文档的结构化属性进行过滤和排序，如作者、最后修改日期、文档类型、文档中引用的公司（或其他感兴趣的实体）。今天，一些供应商（例如，FAST Engine Search11和Google Search Appliance12）提供了企业搜索功能。企业搜索引擎的一个流行架构是*集成*模型，如图5所示。搜索引擎抓取每个数据源，并使用适合快速查询的内部表示法将数据存储到一个中央内容索引中。配置数据控制要索引哪些对象（例如，从数据库中返回对象的*抓取查询*），以及响应用户查询返回哪些对象（例如，当查询关键字与抓取对象匹配时，针对数据库运行的*服务查询*）。企业搜索引擎需要解决几个技术难题。首先，抓取依赖于为每个源提供合适的*适配器*。实现高度的数据新鲜度需要专门的适配器，能够有效地识别和提取源头的数据变化。其次，对不同数据源的结果进行排名是非平凡的，因为可能没有简单的方法来比较不同源的相关性。与Web搜索中的排名不同，企业中跨文档的链接要稀疏得多，因此不是一个可靠的信号。同样，查询日志和点击信息通常没有足够的规模，无法用于排名。最后，部署企业搜索可能涉及到手动调整相关性，例如，通过调整每个来源的权重。**提取-转换-加载工具** 报表、特别查询和预测分析的准确性和及时性取决于能否从操作数据库和外部数据源中高效地将高质量的数据导入数据仓库。提取-转换-加载(ETL)指的是一系列工具的集合，这些工具在帮助发现和纠正数据质量问题以及高效地将大量数据加载到仓库中发挥着至关重要的作用。*数据质量。*当来自一个或多个来源的数据被加载到仓库中时，可能会出现错误（例如，数据输入错误可能会导致记录中的州='加州'，国家='加拿大'），同一数值的表示不一致（例如，'CA'，'加州'），以及数据中的缺失信息。因此，帮助检测数据质量问题和恢复仓库中的数据完整性的工具对BI有很高的回报。*数据剖析*工具通过检测-**图5.企业搜索架构（集成模型）。应用查询引擎索引引擎**查询搜索结果内容索引配置数据网站业务数据电子邮件网络分享评论文章2011年8月第54卷第8期ACM 97的通信。8 | **acm 97** ing违反了数据中预期保持的属性。例如，考虑一个客户名称和地址的数据库。在一个干净的数据库中，我们可能期望（姓名，地址）的组合是唯一的。数据剖析工具可以*验证*这个唯一性属性是否成立，并可以量化该属性被违反的程度，例如，如果Name或Address信息缺失，就可能发生这种情况。数据剖析工具还可以*发现*在给定数据库中成立的规则或属性。例如，考虑一个需要导入数据仓库的外部数据源。知道哪些列（或列集）是源的*键*（唯一）是很重要的。这有助于将输入的数据与仓库中的现有数据进行匹配。为了提高效率，在剖析大型数据库时，这些工具通常会使用采样等技术。准确地从字符串中*提取结构*，对提高仓库中的数据质量具有重要作用。例如，考虑一个购物网站，该网站存储的MP3播放器产品数据具有制造商、品牌、型号、颜色、存储容量等属性，接收到的产品数据源为文本，例如，"Coby MP3 512MB MP-C756 - 蓝色"。能够健壮地将文本中存在的结构化信息解析为数据仓库中的适当属性是非常重要的，例如，对于回答Web站点上的查询。供应商已经为产品和地址等重要的垂直领域开发了大量的解析规则集。Sarawagi28的调查文章讨论了更广泛的信息提取领域的技术。另一个有助于提高数据质量的重要技术是*重复数据删除*：识别大约重复的实体（例如，客户）的组。这可以看作是一个图聚类问题，其中每个节点是一个实体，如果两个实体之间的相似度足够高，则两个节点之间存在一条边。定义两个实体之间相似度的函数通常基于字符串相似度函数，如编辑距离（例如，'Robert'和'Robet'的编辑距离为，以及特定领域规则（例如，'Bob'和'Robert'是同义词）。因此，能够在许多实体对之间有效地执行这种近似字符串匹配（也称为*模糊匹配*），对于去重复化来说非常重要。大多数主要供应商都支持模糊匹配和重复数据删除，作为其ETL工具套件的一部分。关于合并不同来源的数据的工具概述可以在Bernstein中找到。3 *数据加载和刷新。*数据加载和刷新实用程序负责将来自操作数据库和外部源的数据快速地移动到数据仓库中，并尽可能地减少两端的性能影响。这方面有两大挑战。首先，需要在数据源处高效地*捕获数据*，即识别和收集要移动到数据仓库的数据。*触发器*是SQL支持的通用结构，可以识别被插入/更新SQL语句修改的行。然而，触发器是一种相对较重的机制，会给运行OLTP查询的操作数据库带来非同小可的开销。一个更有效的捕捉变化数据的方法是嗅探数据库的事务日志。*事务日志*被数据库系统用来记录所有的变化，以便系统在崩溃时能够恢复。允许在处理交易日志记录时推送*过滤器*，以便只捕获相关的变更数据；例如，只捕获与组织内某一部门有关的变更数据。第二个方面涉及到将捕获的数据高效地*转移*到仓库中的技术。多年来，数据库引擎已经开发了专门的、性能优化的API，用于批量加载数据，而不是使用标准SQL。在仓库对数据进行分区有助于最大限度地减少数据仓库服务器上查询的中断。数据被加载到一个分区中，然后仅使用元数据操作将其切换进来。这样，引用该表的查询只在元数据操作所需的极短时间内被阻止，而不是在整个加载时间内。最后，加载实用程序通常也会检查点操作，以便在失败的情况下，整个工作不需要重做。利用上面讨论的捕捉变化的数据和高效加载的技术，现在的实用程序能够在几秒钟内接近刷新率（例如，Oracle GoldenGate22）。因此，就像前面讨论的那样，甚至有可能服务于一些接近实时的 BI 场景。**其他BI技术** 这里，我们讨论两个我们认为越来越重要的领域，在这两个领域中，研究起到了关键作用。数据挖掘和文本分析。数据挖掘能够对数据进行深入分析，包括建立预测模型的能力。数据挖掘提供的一套算法远远超出了关系型DBMS和OLAP服务器中的聚合函数。这种分析包括决策树、市场篮子分析、线性和逻辑回归、中性网络等（见调查6）。传统上，数据挖掘技术是由统计软件公司单独打包的，例如SAS，26和SPSS。27 其方法是从数据仓库中选择一个数据子集，对所选数据子集进行复杂的数据分析，以确定关键的统计特征，然后建立预测模型。最后，将这些预测模型部署在业务数据库中。例如，一旦确定了向客户提供房间升级的稳健模型，就必须将该模型（如决策树）集成回运营数据库中，以实现可操作性。这种方法会导致几个挑战：数据从仓库移动到数据挖掘引擎，以及挖掘引擎处潜在的性能和可扩展性问题（或隐含的用于建立模型的数据量的限制）。为了实用，这种模型需要在新数据到达时高效应用。越来越多的趋势是"独立数据库分析"，即在后端数据仓库架构中集成数据挖掘功能，以克服这些限制（例如，Netz等人20和Oracle数据挖掘21）。**98 acm通讯**｜2011年8月｜第54卷｜第8期评论文章 文本分析。8 评论文章 *文本分析。*考虑到一家生产便携式音乐播放器的公司对其产品进行调查。虽然许多调查问题是结构化的（例如，人口统计信息），但其他开放式的调查问题（例如，"在这里输入其他评论"）通常是自由文本。根据此类调查回复，公司希望回答以下问题。调查回复中提到了哪些产品？人们提到了哪些关于产品的话题？在这些场景中，面临的挑战是如何降低人力成本，即必须阅读大量的文本数据，如调查问卷、网络文档、博客和社交媒体网站，以提取回答这些疑问所需的结构化信息。这就是文本分析引擎的关键价值。当今的文本分析引擎（例如FAST11和SAS26）主要是提取结构化数据，这些数据大致可分为以下几类。*命名实体*是对已知对象的引用，如地点、人物、产品和组织。*概念/主题*是文档中的术语，在文档集合中经常被引用。例如，在上述便携式音乐播放器的情景中，"电池寿命"、"外观"和"附件"等术语可能是调查中出现的重要概念/主题。这样的信息有可能被用来作为对调查结果进行分类的依据。*情感分析会*对每个文本文档（或文档的一部分，如句子）产生"正面"、"中性"或"负面"等标签。这种分析可以帮助回答诸如哪种产品收到的负面反馈最多这样的问题。云数据服务。今天管理企业BI需要处理硬件供应、可用性和安全补丁等任务。云虚拟化技术(例如，Amazon EC21)允许将服务器托管在云端的虚拟机中，并通过更好地利用硬件资源实现服务器整合。托管服务器还通过卸载可管理性任务，利用现收现付的定价模式，只为实际使用的服务付费，从而有望降低成本。硬件虚拟化在云端的成功，促使数据库厂商对数据服务进行虚拟化，从而进一步提高资源利用率，降低成本。这些数据服务最初是作为简单的键值存储开始的，但现在已经开始支持作为托管服务的单节点关系型数据库的功能（例如，微软SQL Azure18）。虽然这类云数据库服务的主要初始用户是相对简单的部门应用（OLTP），但这一范式也正在扩展到BI（例如，Pentaho25）。在这些应用所收集的数据上需要全方位的BI服务，这对云数据库服务提出了新的挑战。首先，大型报告或特别查询的性能和规模要求将要求数据库服务提供商支持云中的大规模并行处理系统（并行DBMS和/或基于MapReduce的引擎），其次，这些服务是多租户的，复杂的SQL查询可能是资源密集型的。因此，向租户提供性能服务水平协议（SLA）并在租户查询之间明智地分配系统资源的能力变得非常重要。第三，传统的"内部"BI的许多技术挑战，如安全和精细的访问控制，在云数据服务的背景下变得更加重要。例如，在公共云中，处理加密数据查询的技术变得更加重要。由于这些原因，采用商业智能技术的中间步骤可能是在私有云中，私有云的前景与公有云相似，但对安全等方面有更多的控制。**结论** 如今，BI在研究和工业领域的前景是充满活力的。数据获取变得越来越容易，拥有10到100TB或更多关系型数据的大型数据仓库变得很普遍。文本数据也作为BI的宝贵来源被开发利用。硬件技术的变化，如主内存成本的下降，正在影响大型数据仓库的后端如何架构。此外，随着云数据服务的扎根，预计BI后端架构会发生更多变化。最后，在移动设备上为当今的知识工作者提供交互式BI体验的需求越来越大。在下一代移动设备上实现新颖、丰富和交互式的BI应用有很多机会。因此，商业智能软件在未来仍有许多令人兴奋的技术挑战和机会，将继续重塑其格局。