第6章　分布式数据仓库

大部分企业建立和维护单一中央数据仓库环境。为什么单一中央数据仓库环境比较流行？这有许多原因：

- 数据仓库中的数据是全企业集成的数据，仅在总部使用集成视图。
- 数据仓库中的大量数据使数据的单一的集中式存储具有意义。
- 即使数据能被集成，但是若将它们分布于多个局部站点，则存取这些数据也是很麻烦的。

总之，政策、经济和技术等诸多因素都更倾向于建立和维护单一中央数据仓库环境。但是在某些特定场合，需要建立分布式数据仓库环境。

6.1　引言

为了便于理解分布式数据仓库何时有意义，我们先看一些处理的基本拓扑结构。图 6-1表明了一种常见的处理拓扑结构：

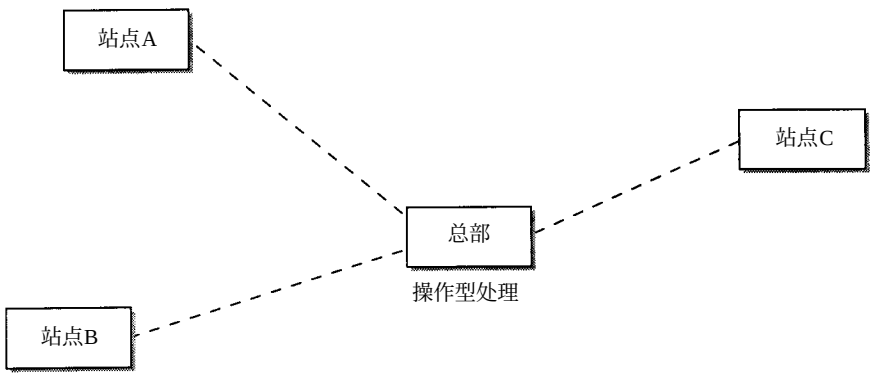


图6-1　许多企业处理的典型拓扑图

如图6-1所示，某企业设有一个总部，负责处理所有的业务。若在局部层上存在某些业务处理，这些处理也是非常基本的。局部层上可能拥有一系列的哑终端，但是所作的处理工作都是不太重要的。在这种拓扑结构中，不可能需要建立分布式数据仓库环境。

当局部层出现基本的捕获信息活动时，局部处理的复杂性将有所提高，如图 6-2所示。

在图6-2中，局部层有少量的捕获信息活动。一旦承揽了某业务，即将它传送到总部去处理。在这种简单的拓扑结构中，也不需要建立分布式数据仓库环境。

现在，看一下如图 6-3所示的拓扑结构。同前两种处理拓扑结构相比 ,在图6-3中，局部层有相对较多的处理过程。就拿操作型处理来说，局部站点是自主的。仅偶然或某些特定的处理需要将数据和业务活动发送到总部处理。对于这类企业来说，采用某种形式的分布式数据仓库就是必要的。

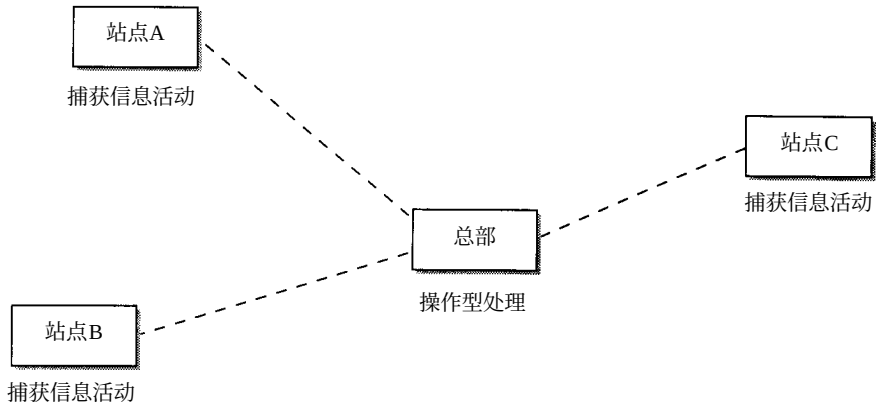


图6-2 某些场合，在站点层处理一些基本业务活动

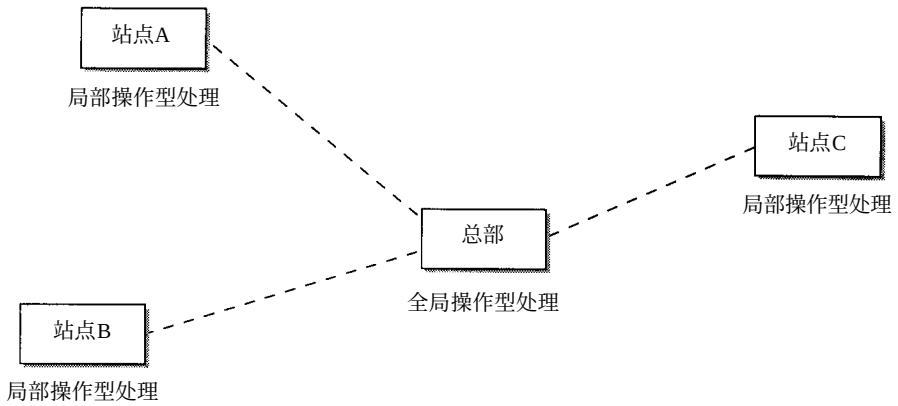


图6-3 在分布式数据仓库谱系的另一端——在局部层要做许多操作型处理

正如即将讨论的，分布式数据仓库的种类很多。认为分布式数据仓库仅是一种两级模式的思想是错误的。分布式数据仓库有许多层次(级别)。

局部自主性和处理过程较少的大多数企业一般拥有一个中央数据仓库，如图 6-4所示。

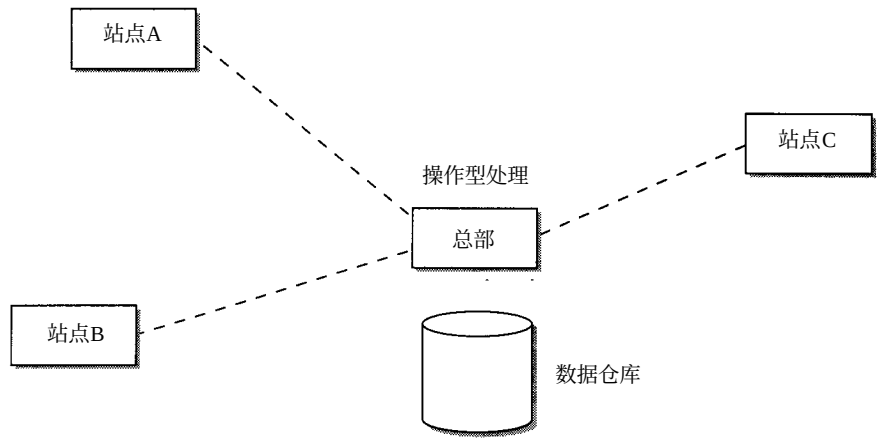


图6-4 大部分企业具有一个中央控制与存储的中央数据仓库

6.2 局部数据仓库

数据仓库的一种形式是局部数据仓库。局部数据仓库仅包含对局部层有意义的数据。图6-5表明了一系列局部数据仓库的一个简单实例。

在图6-5中，位于不同地区的分部或不同的技术联营组织各拥有一个局部数据仓库。局部

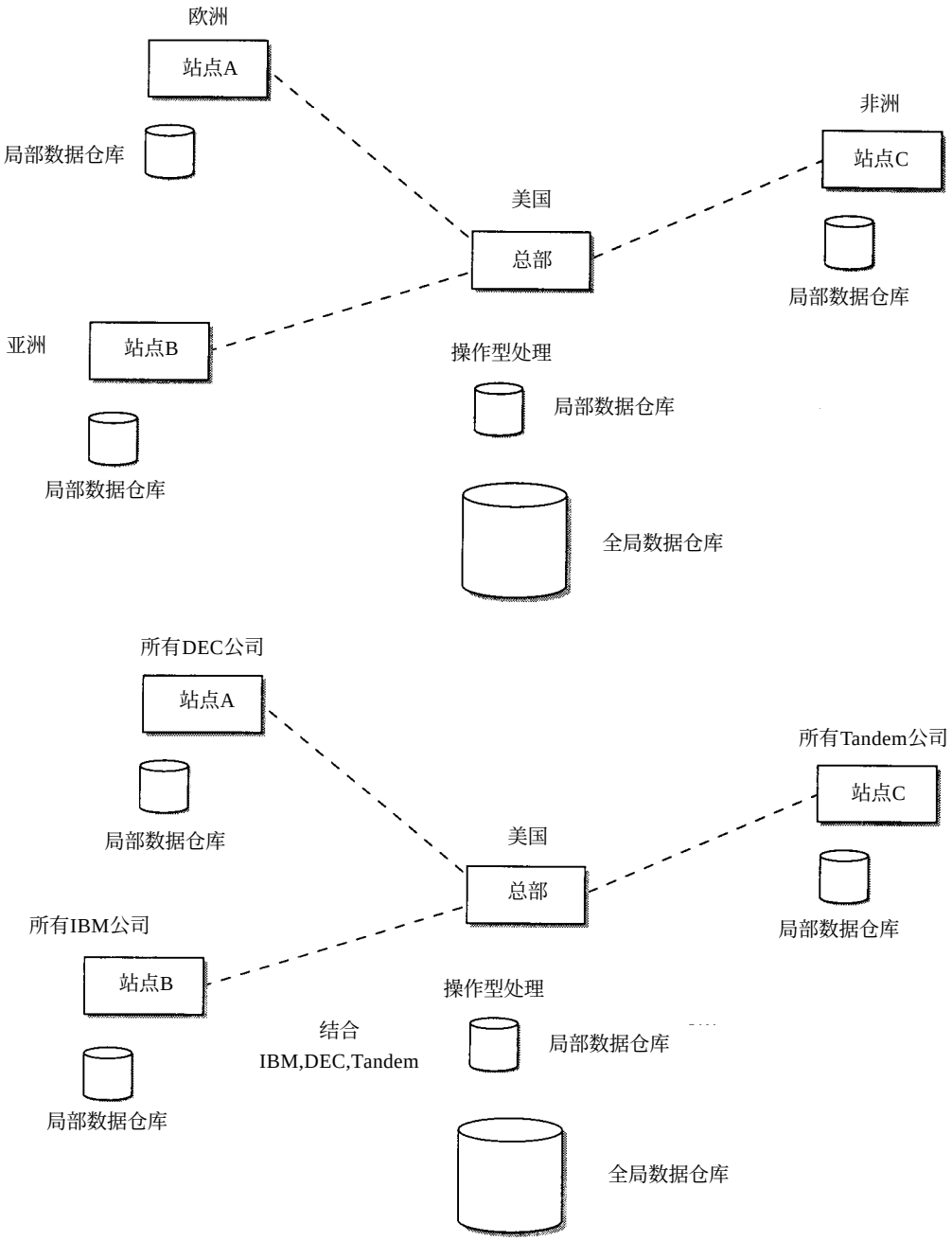


图6-5 需创建两级数据仓库的一些情形

数据仓库除了存储的数据是局部的外，具有其他任何数据仓库的相同功能。换句话说，局部数据仓库包含的是在局部站点上的历史的和集成的数据。局部仓库间的数据或数据结构不必要协调一致。

6.3 全局数据仓库

当然，也可以是全局数据仓库，如图6-6所示。全局数据仓库的范围涉及整个企业或组织。它内部的每个局部数据仓库也都有各自服务的局部站点范围，全局数据仓库的范围是该企业。同局部数据仓库一样全局数据仓库也包含历史数据。局部数据仓库的数据源如图 6-7所示，可看出它们的数据来源于相应的操作型系统。

研究不同的局部数据仓库间的公用数据是一个很有意义的问题。图 6-8表明每个局部数据仓库都有自己特定的数据和结构。

局部数据仓库间数据的重叠部分或公用部分是完全一致的，图 6-8所示的局部数据仓库之间的无论什么数据或处理过程不用协调。

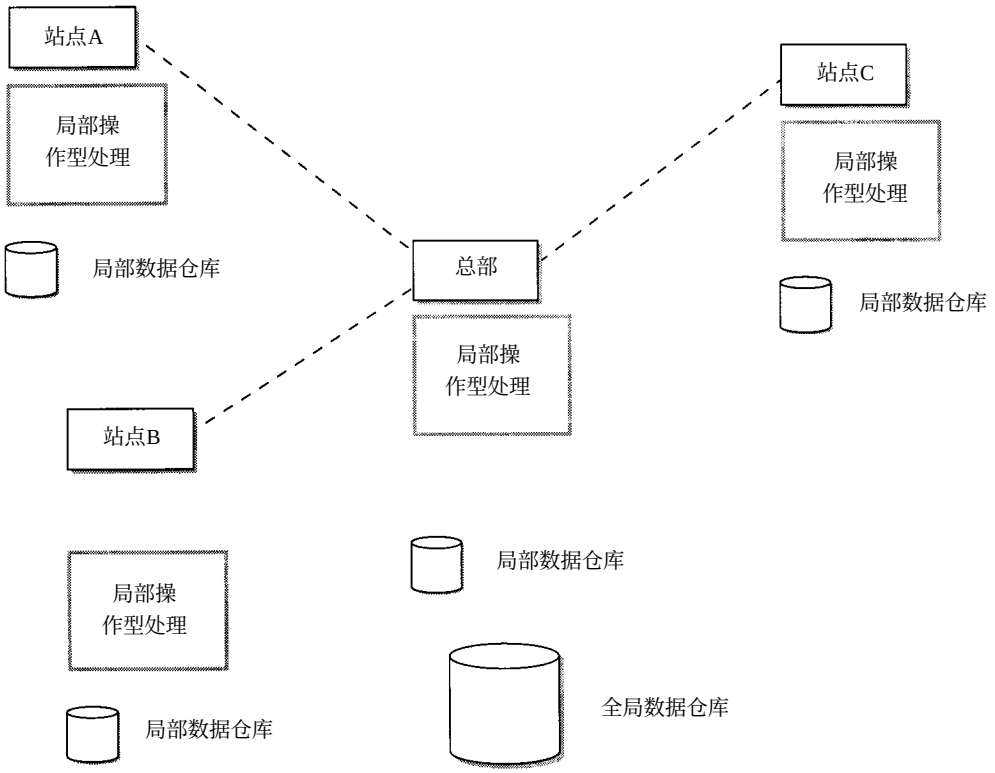


图6-6 典型分布式数据仓库的可能形式

但是，假定某企业内一个站点和另一个站点间的数据存在自然重叠是合理的。如果存在这样的交叉部分，那么最好将这些数据存放在全局数据仓库中。图 6-9表明全局数据仓库中数据来自于现有的局部操作型系统的情形。

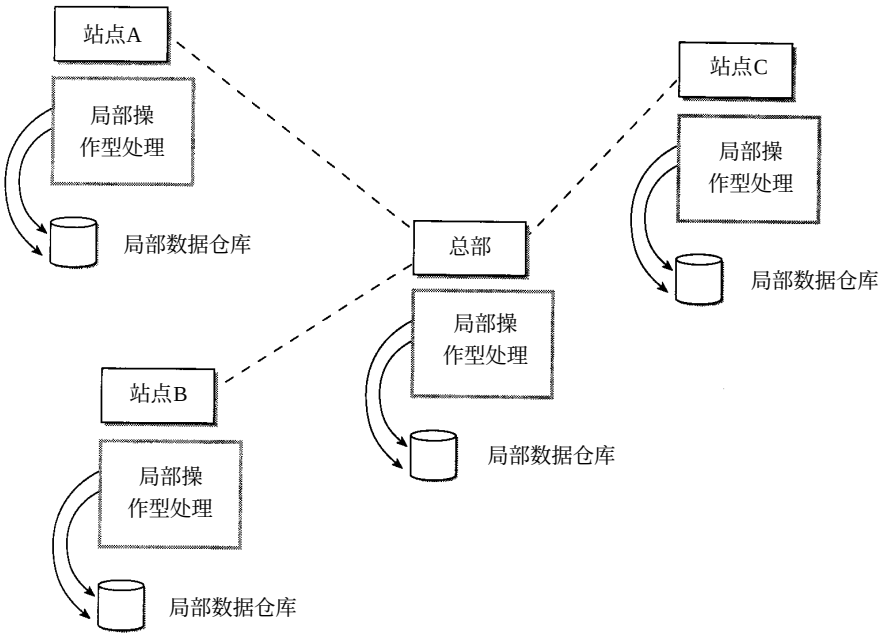


图6-7 从局部操作型环境到局部数据仓库的数据流

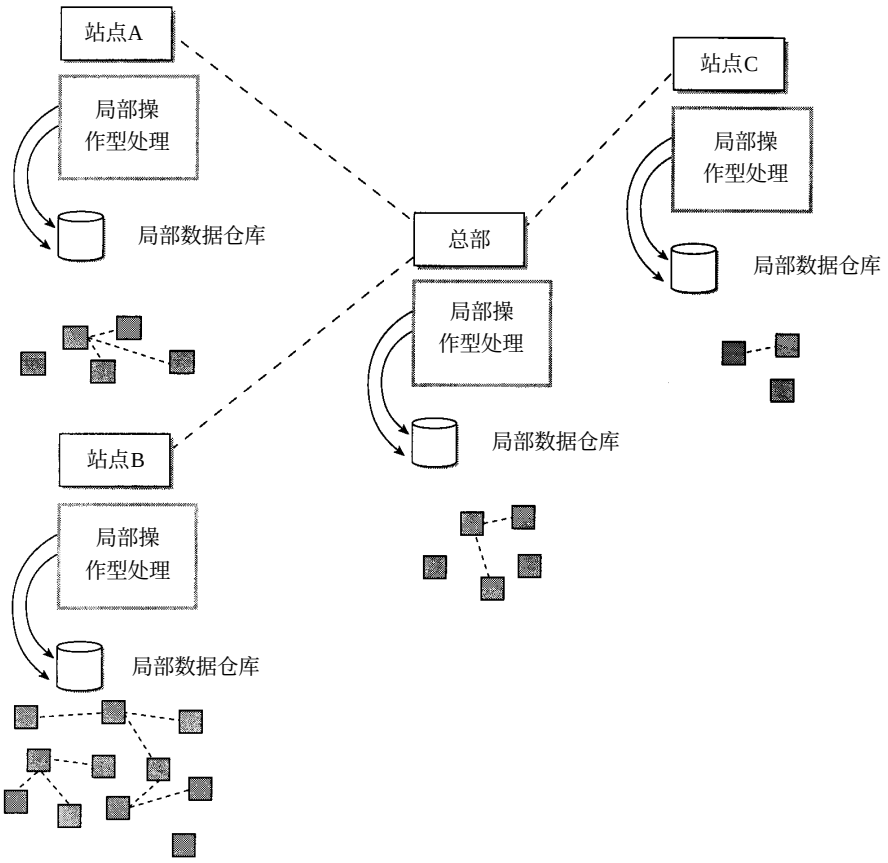


图6-8 局部数据仓库间的数据及结构是非常不同的

6.4 互斥数据

图6-9中隐含着未显示出来的假定，即假定局部数据仓库中的数据并不一定都出现在全局数据仓库中，全局数据仓库中的数据也并不都来自于局部数据仓库。所以，全局数据仓库中的数据不一定都从局部数据仓库中导入。

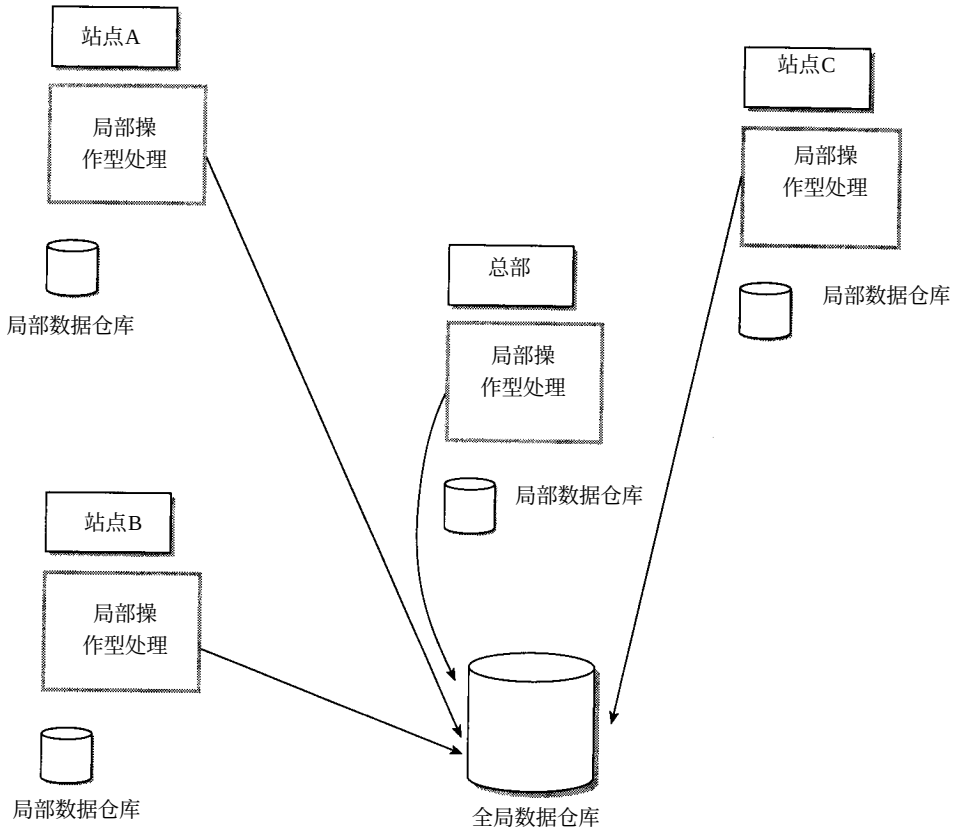


图6-9 全局数据仓库中数据来自于无关的操作型系统

全局数据仓库中包含的是企业内部公共的和集成的数据。分布式数据仓库环境成功的关键就是如何将局部操作型系统中的数据映射到全局数据仓库中的数据结构，如图 6-10所示。

图6-10表明全局数据仓库有一个公共的数据结构，包含企业内所有的公有数据。但是从每个局部站点到全局数据仓库的数据映射不同。换句话说，全局数据仓库是集中定义和设计的，而已存在的局部操作型系统的数据映射是由局部设计者和开发者选择的。

从局部操作型系统到全局数据仓库系统的数据映射，刚开始设计时很可能不完全准确，但是随着时间的推移，用户反馈信息的积累，它们将会逐步完善。

数据仓库的一种变化形式是在局部层保存全局数据仓库的数据“登台”区域。图 6-11表明这种情况，局部层在将数据传送到中心位置前先在每个局部区域登台这些数据。在许多场合，这种方法可能在技术上是必需的。另外它可能会带来的一个重要问题是：当局部数据仓库中保存的登台数据传送到全局数据仓库后应该腾空吗？如果局部层不删除这些信息，那么将导致数据冗余。但是，在某些情况下，一定量的冗余数据也是需要的。对此问题必须作出

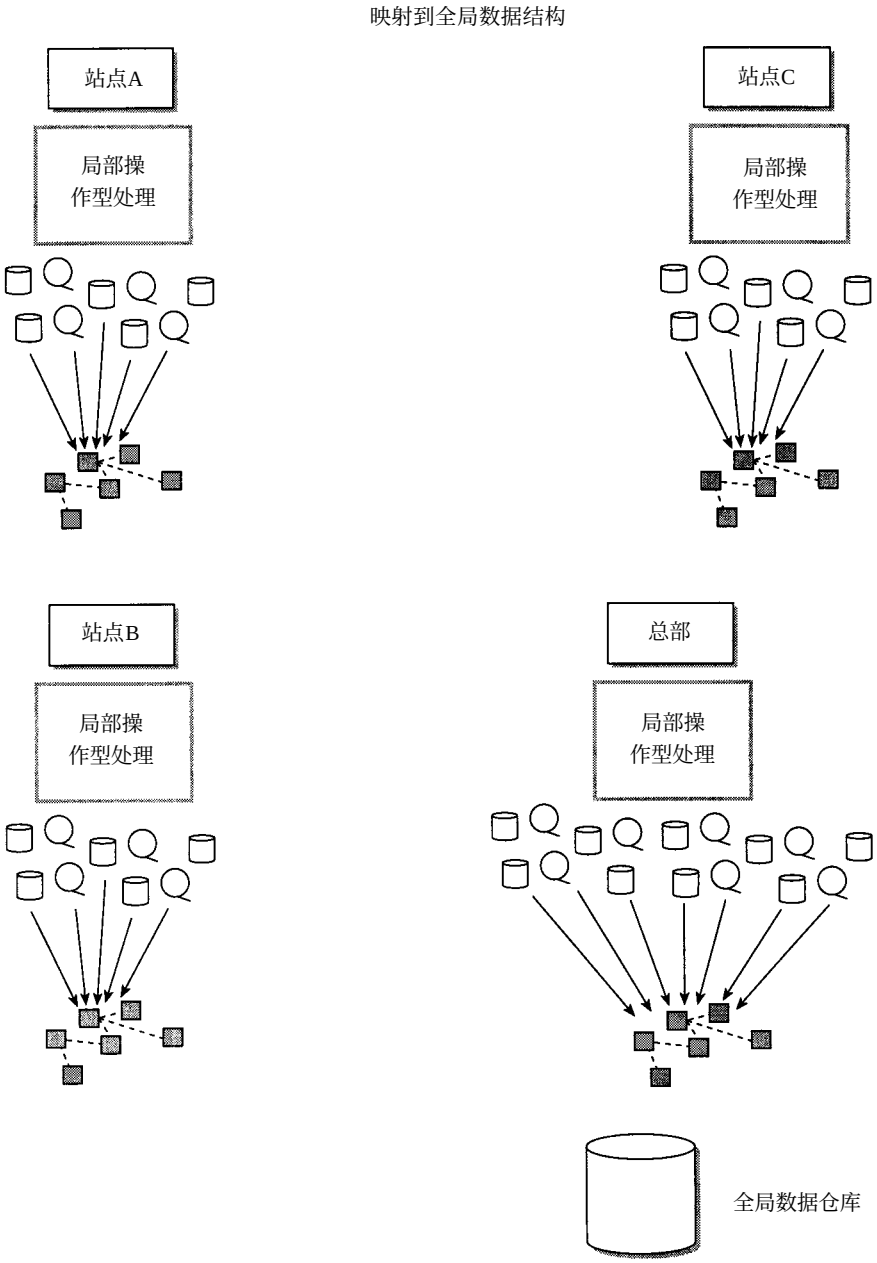


图6-10 全局数据仓库有一个公共的数据结构，每个局部站点以不同的方式映射到公共结构

决定，且应提出处理策略与过程。

起初的全局数据仓库建造好后可能还存在着许多候选的主题域。许多企业以企业财务作为主题域。财务之所以是一个好的起点，原因如下：

- 它是相对稳定的。
- 具有高的可视性。
- 仅是企业业务的一部分(当然除了金融机构)。
- 仅需处理少量数据。

建造全局数据仓库时，必须处理一些特殊问题。就数据层来说，全局数据仓库并不符合典型的数据仓库结构。其中一点是细节数据存在于局部层，而轻度综合的数据存在于中央全局层。例如，假定一个公司的总部在纽约，在远离总部的德克萨斯、加利福尼亚和依利诺州设有分部，这些分部单独管理销售和财务细节数据。总部将数据模型传送到各分部，各分部将必要的企业数据转换为公司内部可集成的数据形式。数据在局部层进行转换后，传送到纽约总部。原始的、未转换的细节数据仍然保存在局部层。仅将转换过的、轻度综合的数据传送到总部。这是典型的数据仓库结构的一种变化形式。

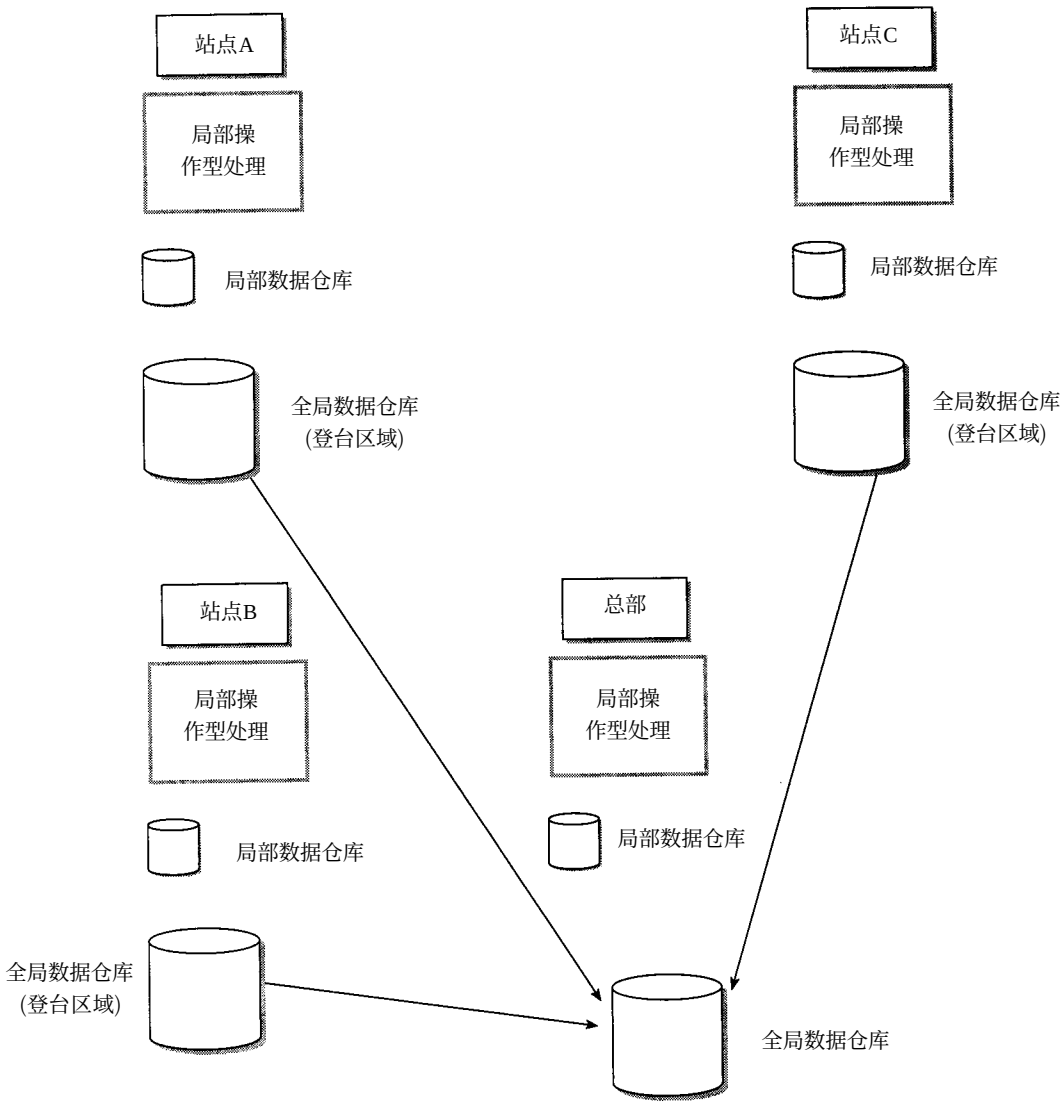


图6-11 在局部层上有可能使全局数据登台，然后将其传送到总部层的全局数据仓库

6.5 冗余

从数据冗余的角度考虑，局部数据仓库和全局数据仓库间存在着冗余问题。图 6-12显示了

一种策略，可避免局部层和全局层间的数据冗余(就此而言，全局数据是存放在局部登台区还是存放在全局层并不重要)。当局部数据仓库和全局数据仓库间的数据出现冗余时，即表明没有正确定义不同级别的数据仓库所辖的范围。当出现局部层和全局层之间的范围差别时，出现蜘蛛网问题将是迟早的事。为此，采用局部数据和全局数据相互排斥的机制将是一种重要策略。

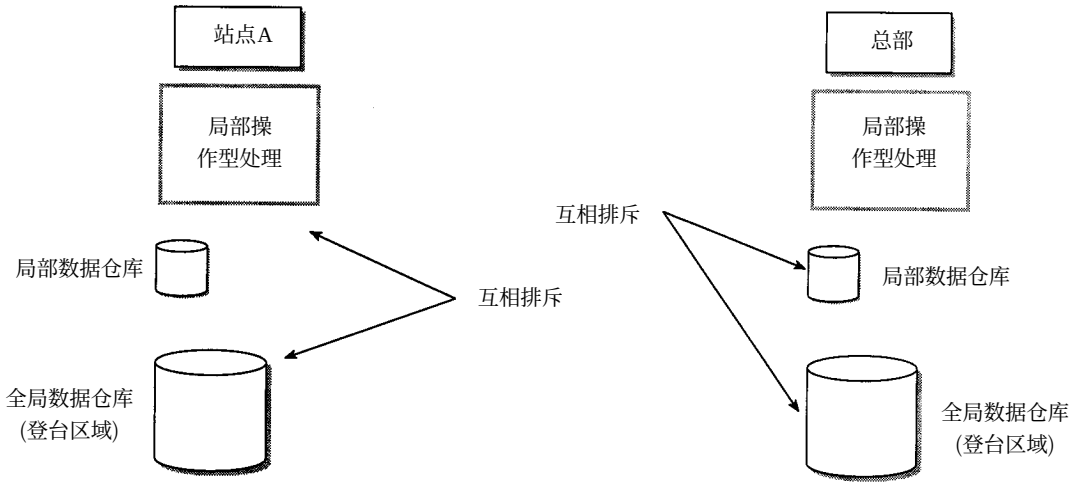


图6-12 数据可以存放在局部数据仓库或全局数据仓库，但不能两者都存放

6.6 全局数据存取

与管理不同的数据仓库所需要的策略相一致，还有一个数据存取问题。初看起来，这个问题好象微不足道，但实际却存在重要分歧。

图6-13表明了一些局部站点正在存取全局数据的情形。这些存取方式正确与否是与查询有关的，它们可能是或不是数据仓库的正确使用方法。如果在存取过程中，全局数据被作为信息使用并且仅被访问一次，那么在局部层上这种存取方式就可能是正确的。

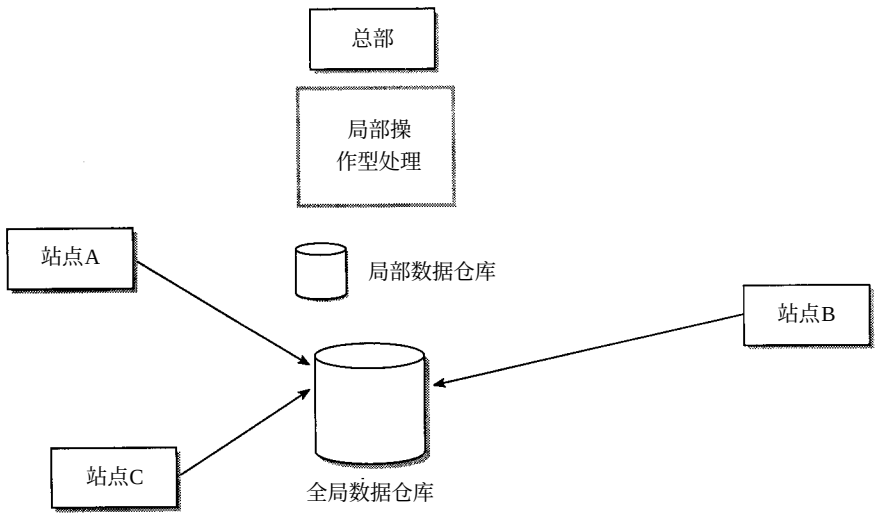


图6-13 需要解决的一个重要问题是局部站点是否应访问全局数据仓库

原则上，局部数据应局部使用，全局数据应全局使用。但这又会引发另一个问题：为什么全局分析还要局部处理呢？通常没有好的理由来作出解释。

另一个问题是在体系化信息环境中信息请求的路径选择问题。当仅存在一个单一的中央数据仓库时，此问题关系不大。但是当数据分布在一种复杂环境中时，例如图 6-14所示的分布式数据仓库中，就需要一定数量的处理来确保能在正确的位置存取数据。

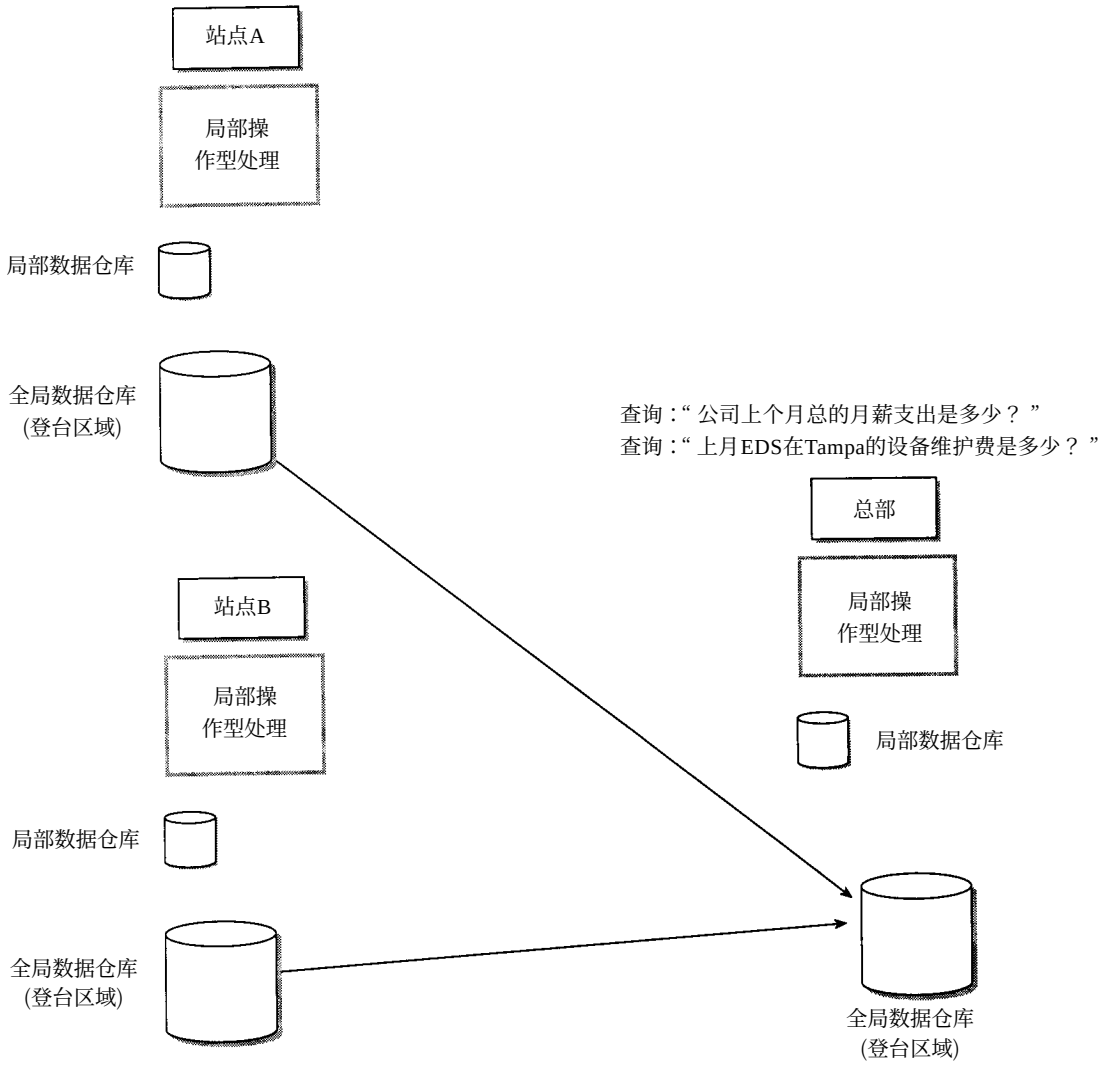


图6-14 正确响应查询需要引向体系结构的不同位置

例如，通过查询局部站点来确定整个公司的薪水是多少，这是不正确的。还有，在全局数据仓库中查询上月在某一特定站点上某一承包人的费用支付也是不正确的。

本章没有论述有关全局操作型数据这一相对独立的问题，仅举了一些每个局部站点具有自己的特有数据和处理的实例。然而局部站点的操作型系统间存在某些共同性是完全可能的。

分布式数据仓库的整个问题是比较复杂的。在简单的单一中央数据仓库环境下，其作用与功能是相当明了的。但是在分布式数据仓库环境下，范围、协调、元数据、响应性、数据传输以及局部数据映射等问题确实使得环境复杂化了。

6.7 分布式环境下其他考虑因素

分布式数据仓库的出现并不仅是由于公司向多个地区扩展而引起的，也有其他一些因素。例如，一种因素是把数据仓库置于销售商的分布式技术基础上，Client/Server技术非常适合这种需求。

第一个问题是，数据仓库能采用分布式技术吗？答案是肯定的。第二个问题是，数据仓库采用分布式技术的优缺点是什么？分布式数据仓库的第一个优点是引入代价低。换句话说，对于一个数据仓库，当最初采用分布式技术时，软硬件代价要比最初采用传统的大型集中式技术代价低。第二个优点是存放在数据仓库中的数据量理论上无限制。如果数据仓库中的数据量开始超过分布式处理器的处理能力，那么可在网络中加入另一个处理器。所以可实现持续增加数据。

图6-15中所示的进程描述了一种可能出现数据仓库中数据量无限增加的情况。这是具有吸引力的，因为一般数据仓库将包含越来越多的数据(但并不是无限量)。

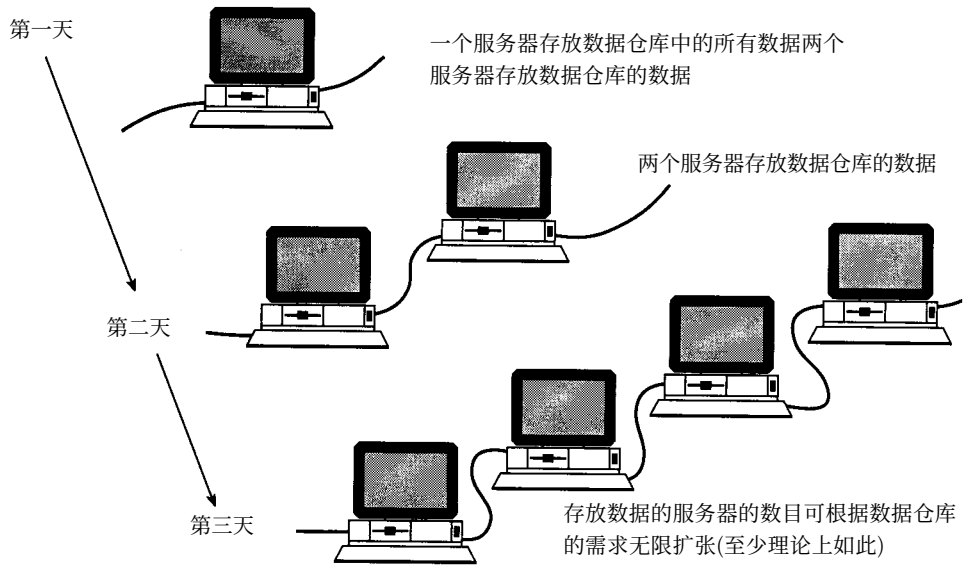


图6-15 添加服务器来保存数据仓库中的数据的进程

但是随之又带来另一些问题，当数据仓库中的处理器(即服务器)扩展到一定数量时，网上将出现过量的传输负载。图 6-16表明这种结果。

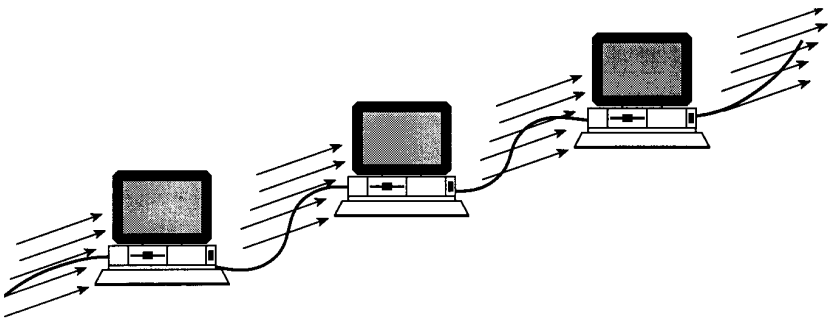


图6-16 增加服务器时网络中可能出现过量负载

另外，当数据分散在多个服务器上时，会出现一个查询需要访问多个服务器上的大量数据的问题。图6-17描述了一个查询希望调用来自多个服务器的大量数据。

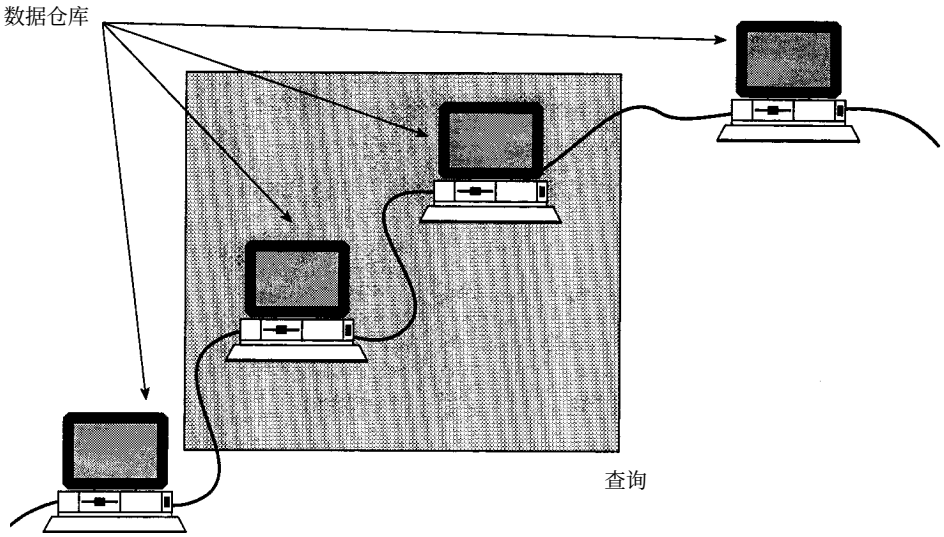


图6-17 一个查询访问多个数据仓库服务器上的大量数据

当然存在着一些技术和方法来处理分布在多个服务器上的数据仓库问题。确定无疑的是，随着时间的推移，数据仓库变得非常庞大。在分布式数据仓库的早期，这里讨论的问题还不明显，但是随着数据仓库越来越成熟，数据已变得很难管理了。

6.8 管理多个开发项目

许多企业采用数据仓库技术时，首先是为财务或市场管理部门建立数据仓库。一旦获得成功，企业内其他部门就希望在此基础上建立相应的数据仓库。总之，数据仓库的体系结构设计者就需要管理和协调企业内的多个数据仓库项目。

6.9 开发项目的性质

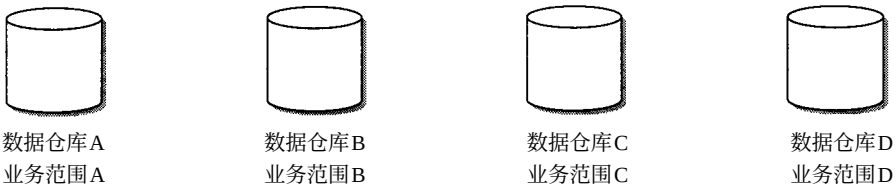
数据体系结构设计者管理多个数据仓库开发项目时，所面临的首要问题是开发项目本身的性质。除非他们知道这些数据仓库开发项目的类型以及它们同整个体系结构的关系，否则将很难管理和协调这些开发项目。由于不同方法所涉及的开发问题是非常不同的，所以不同类型的数据仓库项目需要采用完全不同的管理方案。

多个数据仓库开发项目可以分为四种典型情况，这些情况大体如图 6-18所示。

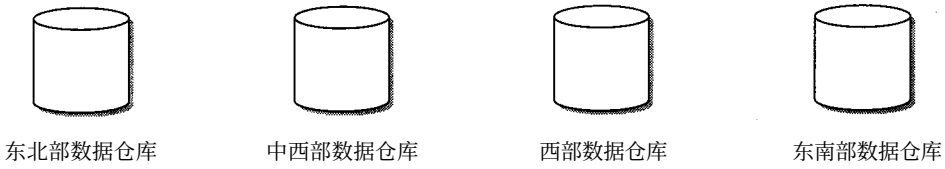
首先，图6-18中给出一种较少出现的情况，即一公司内的业务范围之间是完全分离的、非集成的，对应的数据仓库是由不同的开发小组独立创建的。不同的业务范围独立向公司汇报，但是与共享公司名称不同，在公司内没有业务集成和数据共享。这种公司的结构不是未知的，但不是常见的。在这种没有任何业务集成的罕见情况下，一项数据仓库开发项目与另一项数据仓库开发项目间发生冲突的危险几乎没有。相应地，数据仓库开发项目间很少或不需要管理和协调。

第二种情况是当多个数据仓库项目同时出现时，各个开发小组负责创建同一个数据仓库

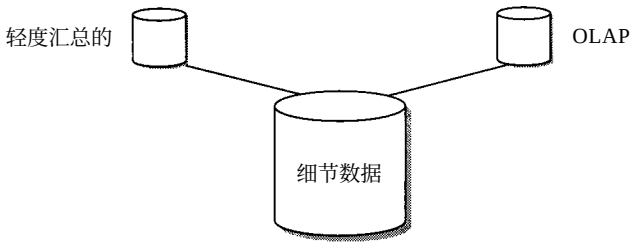
的不同部分，而导致多个数据仓库开发项目同时出现。在这种情况下，同一级数据是由不同开发小组创建的，但是它们分散在不同的地理位置。前一种情况很少出现，而这种情况却是常见的。为了从总体上获得满意的集成效果，要求开发小组间进行密切协作。若开发项目不协调，则大量数据的冗余存储和处理将可能导致较大的浪费。如果数据存在冗余，那么建立的数据仓库的效率可能很低，因为 DSS环境中将存在典型的蜘蛛网问题。由于这种情况较常见，所以值得充分关注。



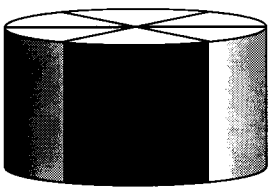
完全非集成的业务范围各拥有自己的数据仓库



同一数据仓库具有一些分布式部分



同一数据仓库内不同级的数据



数据仓库的细节级的不同的非分布式部分

图6-18 多个小组建造数据仓库的四种可能方式

第三种情况是，当多个数据仓库项目开发同时出现时，不同小组负责建立数据仓库环境中的不同级的数据(即汇总数据和细节数据)。这种情况也很常见。由于许多原因这种情况比前

面提到的两种情况较易管理。

第四种情况是，多个小组试图以非分布式的方式建立数据仓库环境中数据当前细节级的不同部分。这种情形很少出现，但是一旦发生，就必须特别注意。最后这种情况非常关键，数据体系结构设计者必须知道问题是什么以及如何协调它们。

对于每种情况，下面我们将就所涉及的问题和各自的优缺点分别进行讨论。本节先讨论一下完全无关的数据仓库。

完全无关的数据仓库的建立和运作如图 6-19所示。某公司有四个业务范围：高尔夫球场管理、炼钢厂、小额银行业务和快餐联营。业务间没有任何集成，因此将来的数据仓库间也不需要协调。从建模到基本技术的选择等所有机制(即平台、 DBMS、存取工具、开发工具等)，每个不同的业务范围均可完全独立进行。

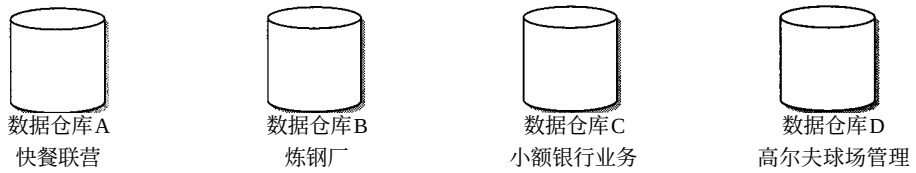


图6-19 四个完全独立的业务部门在业务级没有或很少有数据集成

对于所有自主的业务范围，在某一层上可能也是必须集成的，例如财务平衡表。如果各个业务范围对一个单财务实体负责，那么在财务平衡表层上就必须集成。可能需要建立一个企业数据仓库来反映企业财务。图 6-20表明了一个针对上述不同业务部门的企业财务数据仓库。

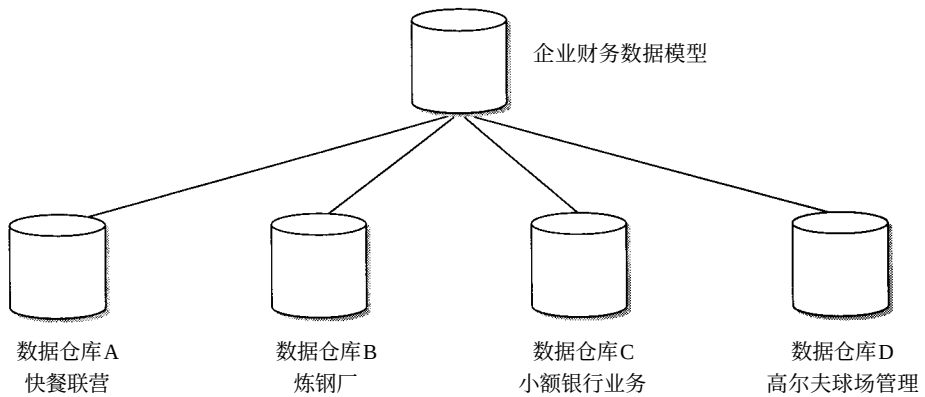


图6-20 独立的业务部门间具有共同的企业财务数据

企业的财务数据仓库包含简单(抽象)的一些实体，例如花费、收入、资金支出、折旧等信息，基本上都是在每个平衡表中出现的业务数据。(换句话说，在财务数据仓库中没有公用企业描述信息，诸如客户、产品、销售等。)当然，图 6-20描述的企业财务数据仓库中的数据可能来自局部数据仓库或在独立运作公司层中所出现的操作型系统。

局部层确实需要元数据。如果有企业财务数据仓库，那么企业财务层也需要元数据。但是在这种情况下，不必要把任何元数据都捆绑在一起。

6.10 分布式数据仓库

与无关的数据仓库模式不同，大部分企业内的部门间存在某种程度的集成。很少的企业是像图6-20所示的那样自主的。更常见的多个数据仓库项目的开发形式如图 6-21所示。

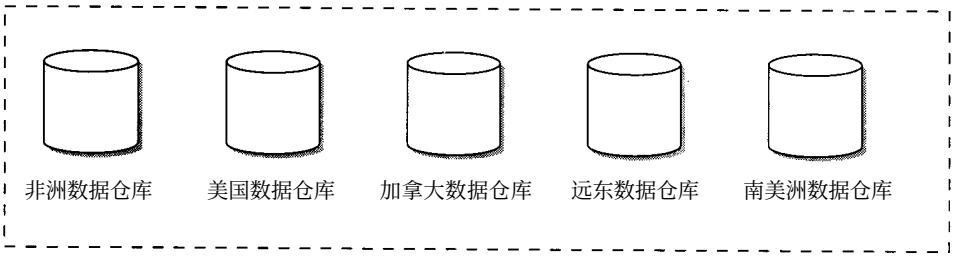


图6-21 逻辑上属于同一个数据仓库

在图6-21中，某公司在世界各地诸如美国、加拿大、南美、远东，非洲等地设有不同的分支机构(站点)。每个分支机构具有自己特有的数据。机构间不存在数据重叠，特别是对于细节事务数据。当第一个体系结构环境建立后，公司期望为它的每个分公司也创建一个数据仓库。在不同的分支机构间存在某种程度的业务集成，同时也假定在不同的区域，业务运作具有当地特色。这种企业组织模式在许多公司是很常见的。

许多企业建造数据仓库时，首先是在每个位于不同地域的部门内创建一个局部数据仓库。图6-22表明了一个局部数据仓库的构造情况。

每个分部根据自己的需要创建富有本地特色的自主数据仓库。值得注意的是，至少就事务数据而言，在不同的区域间不存在冗余的细节数据。换句话说，反映非洲事务的数据单元不可能出现在欧洲的局部数据仓库中。

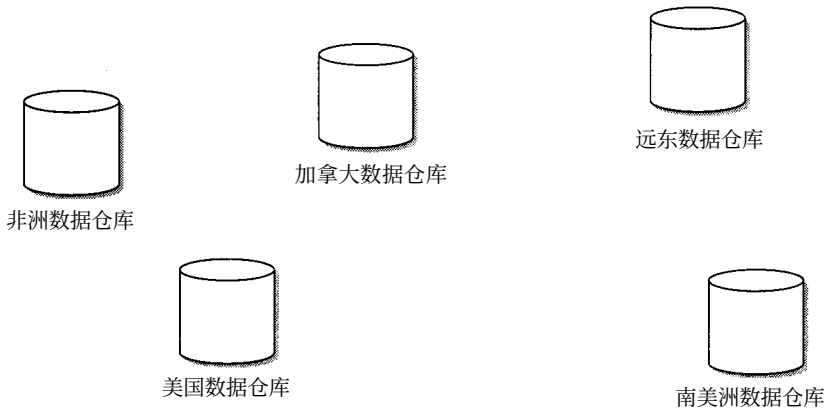


图6-22 在每个自主运作部门建立局部数据仓库

用这种方法创建分布式全局数据仓库有几个优缺点。优点之一是能很快完成。每个局部小组控制局部数据仓库的资源和设计。它们也乐于拥有这样的自主权和控制权。这样开发的数据仓库的优点能在整个企业内实时地表现出来。在 6个月内局部数据仓库就能建好、运行并使局部层分公司受益。不利之处是如果部门间的数据结构(不是内容)存在共同性的话，这种方法却不能识别或合理处理这样的共同性。

6.10.1 在分布的地理位置间协调开发

另一种方法就是尽量协调不同的局部组织间的局部数据仓库的开发项目。这种方法理论上听起来很合理，但真正贯彻起来不太有效。局部开发小组之间不可能完全同步，局部开发小组期待中央开发小组协调局部开发小组的项目，处理出现的各种矛盾。建立了分离的数据模型来支持每个分离地点的数据仓库的构建。

当数据仓库建造的价值在局部层表现出来后，公司就会决定建造一个企业数据仓库(图 6-23)。企业数据仓库反映不同地区、不同部门间的业务集成。它与局部数据仓库有关，但又不同。建立企业数据仓库第一步是在企业数据仓库中为相关的业务部门建立企业数据模型。一般来说，建立企业数据模型采用反复的方法。开始时，数据模型的规模较小、比较简单且限制于一个业务子集。图 6-24显示了企业数据模型的建立。在企业数据模型建立后，将形成企业数据仓库。

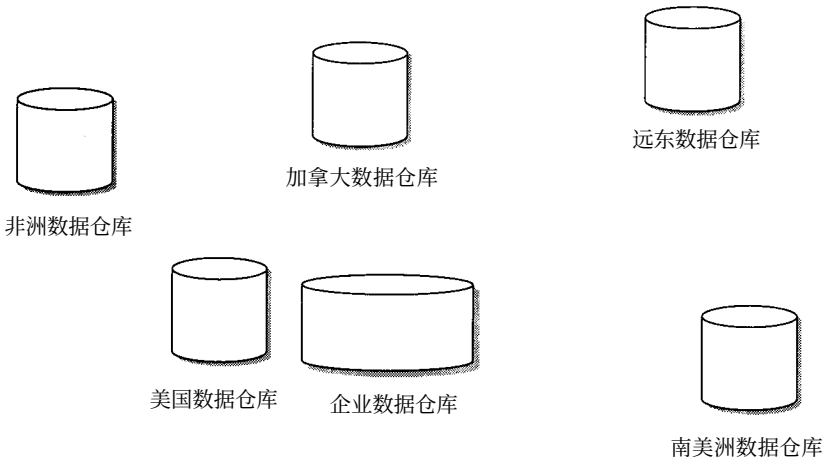


图6-23 某天决定建立企业数据仓库

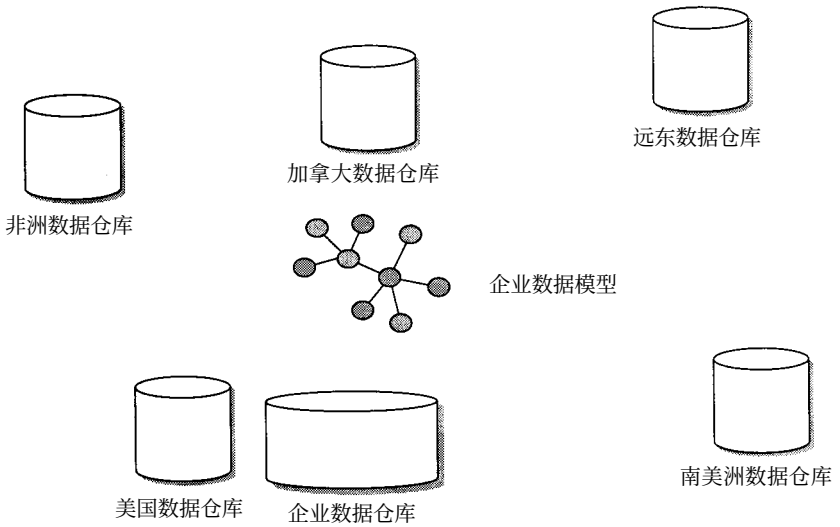


图6-24 建立企业数据模型

6.10.2 企业数据分布式模型

企业数据模型反映企业层的业务集成，因此可能与局部数据模型中的某些部分重叠。这是合理的也是正常的。而在其他情形下，企业数据模型与局部数据模型不同。不管什么情况，都由局部组织来决定如何使企业的数据需求和局部的数据提供能力相适应。因为局部组织比任何人都更了解自己的数据，它们做了最好的准备来表明应如何组织和重组织自己的数据以满足数据仓库中企业数据设计的规范。

当局部层间数据的重叠部分的结构设计得较好时，数据内容上不会有较大范围的重叠。图6-25显示出从局部层建立和装载企业数据仓库的情况。

企业数据仓库的数据源可能来自局部数据仓库，也可能来自局部操作型系统。记录系统完全应在局部层确定。记录系统的定义大多需要几次循环往复。

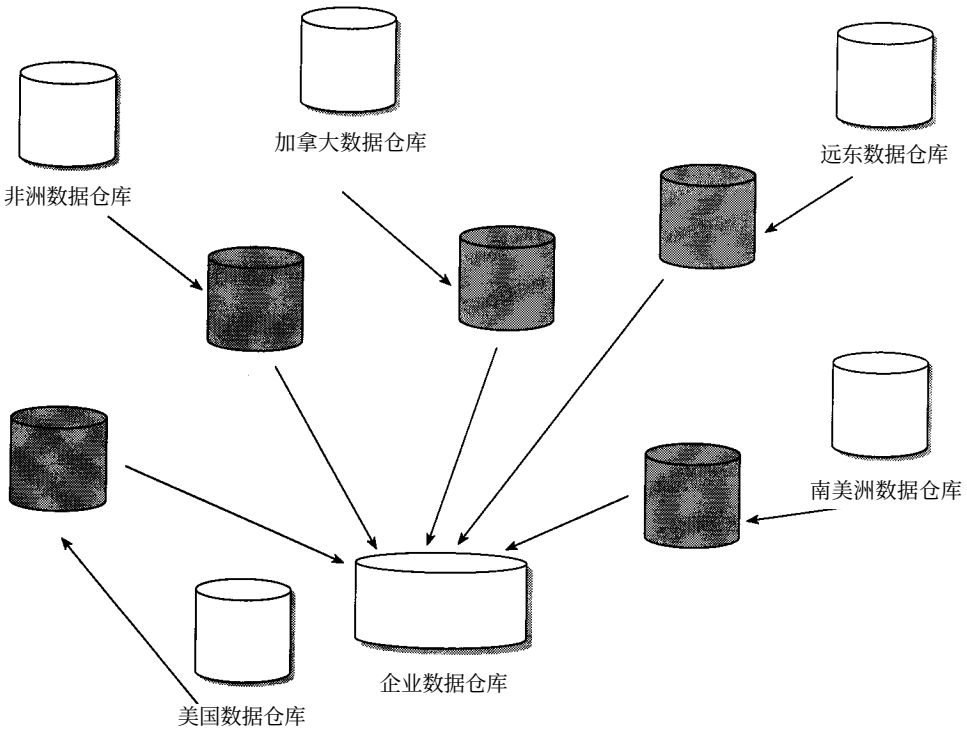


图6-25 从不同的自主运作部门装入的企业数据仓库

此外，一个重要的设计问题是从技术角度考虑如何将局部层的记录系统数据创建和传送到企业数据仓库。在某些情况，正式“登台的”数据保留在局部层。而另一些情况，它们被传送到企业环境，且在局部层不可存取。

通常，企业数据仓库中的数据在结构和概念上一般都是简单的。图 6-26表明企业数据仓库中的数据对企业层的DSS分析员来说是细节数据，同时对局部层的 DSS分析员来说却是汇总数据。这种表面上的矛盾同这样一个事实是一致的，即表现为汇总数据还是细节数据是由观察的角度决定的。

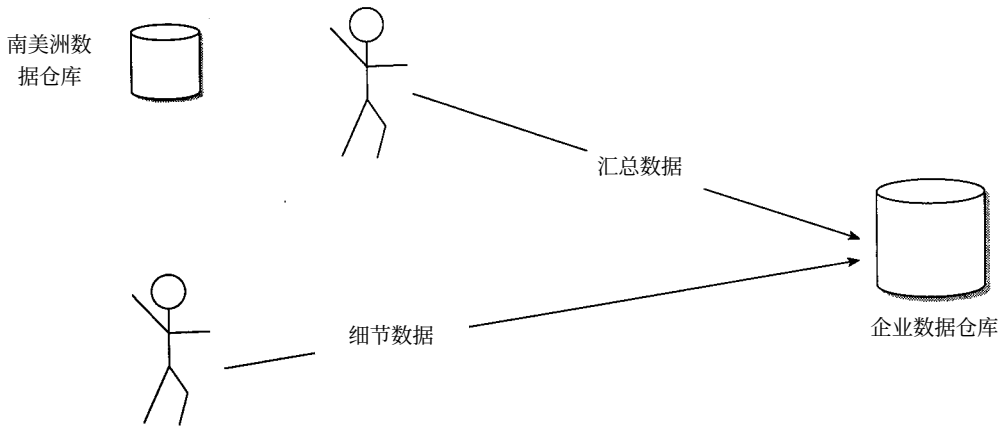


图6-26 在一个层次上是细节的而在另一个层次上是汇总的

分布式数据库的企业数据仓库与完全无关的各公司的企业财务数据仓库的对比，如图 6-27所示。

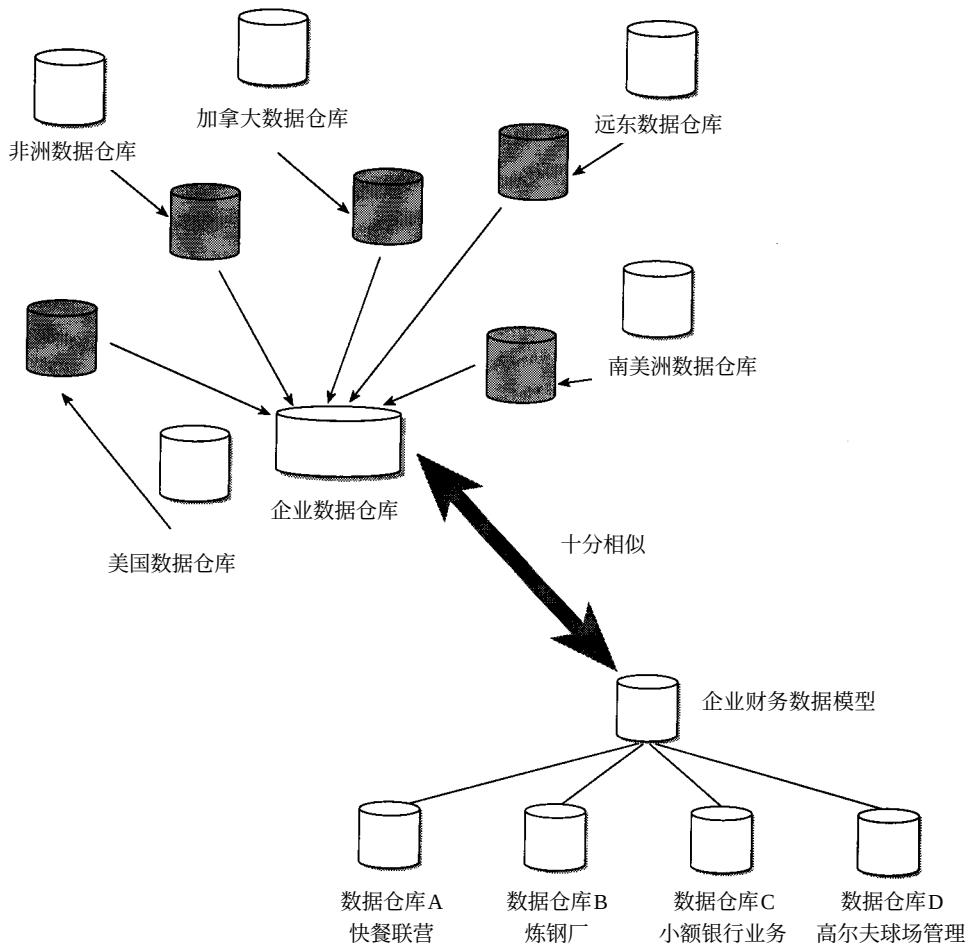


图6-27 分布式公司的数据仓库可以非常类似于一些无关公司的数据仓库

分布式公司的数据仓库在许多方面非常类似于无关公司的数据仓库，诸如在设计和运作方面。然而，它们之间存在一个主要区别。企业分布式数据仓库是对业务本身的扩展，反映客户、销售商、产品等集成信息。因此企业分布式数据仓库表示了业务本身的体系结构。但是，业务无关的公司的企业数据仓库是专门为财务服务的，希望将财务数据仓库应用到公司各部分的其他关系上是不可能的。两个数据仓库的不同是它们表达数据的深度不同。

6.10.3 分布式数据仓库中的元数据

在整个分布式的企业数据仓库中元数据起着非常重要的作用，通过它可以协调不同地域的数据仓库中的数据结构。毫无疑问，元数据为获得一致性和相容性提供了工具。

6.11 在多种层次上建造数据仓库

一个企业同时建造数据仓库的第三种模式是不同的开发小组负责建造数据仓库的不同层次，如图6-28所示。这种模式与分布式数据仓库开发模式区别很大。在图示的情况下， A组负责建造高层综合的数据， B组建造中层综合的数据， C组建造当前层细节数据。

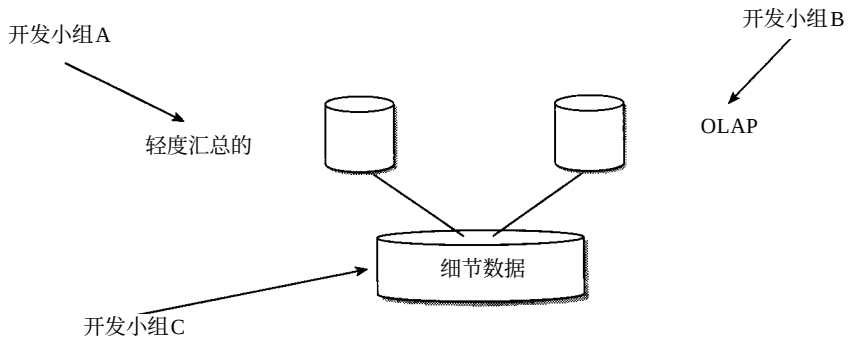


图6-28 不同的开发小组负责建造数据仓库不同层的不同部分

这种数据仓库的多层模式是很常见的。幸运的是，这种方式很易管理，且风险最小。数据体系结构设计者主要关心的问题是如何协调不同开发小组的工作，包括内容的规范说明和结构的描述以及确定开发时间等。例如，如果 A组的进展明显在B组和C组的前面时，那么将出现一个问题，即当A组在汇总级装载他们的数据时，要使用的细节数据可能还不存在。

不同的开发小组同时建造同一数据仓库的不同汇总级时，一个有趣的问题是，正是建造当前细节级的开发小组在使用数据仓库的数据模型。图 6-29显示了这个关系。

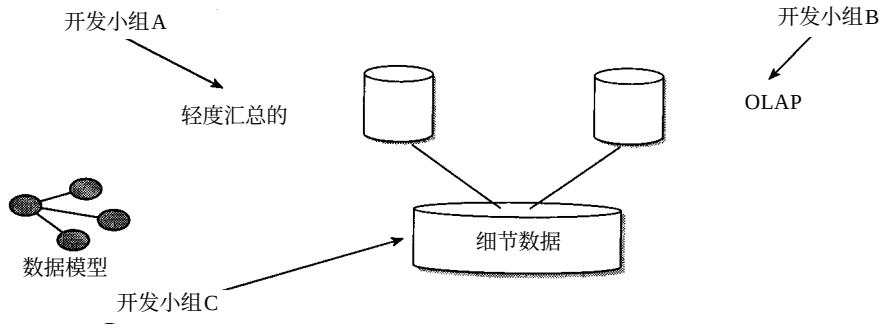


图6-29 正在开发最低细节级的开发组使用该数据模型

数据仓库的数据模型直接反映了负责当前细节级分析和设计的开发者的开发工作。当然，数据仓库模型是间接地反映了所有开发小组的需求。但是由于其他开发小组是对当前细节级数据进行汇总的，所以它们对各自的需求又有进一步的描述。在大多数场合，较高汇总级的开发者拥有自己的反映他们特定的需求的数据模型。

在数据仓库中管理建造不同汇总级的多个小组的问题之一，是数据仓库各层采用的技术平台的问题。一般来说，不同的开发小组选用的开发平台不同。事实上，不同的开发小组选取相同平台的情况确实也不常见。这有几个原因，而主要的是代价问题。数据的细节级，由于处理的数据量大，所以要求一个企业级的平台。不同汇总级，特别是在较高的汇总级，需处理的数据量相对较少，所以要求较高的汇总级同细节级采用同一平台(尽管这也是可以的)未免太过分(代价太高)。

数据仓库中各种汇总级常常被置于不同于细节级数据平台的其他技术平台上的另一个原因，是这些可选用的平台提供更多种多样的特殊软件，而这些软件中的许多是驻留细节数据的单一平台上所没有的。不管数据的不同层次是采用单一平台还是多种平台，都必须认真存储和管理元数据，以保证从一个细节级到下一层细节级的连续性。

由于数据仓库的不同开发小组在开发不同数据级时通常采用不同平台，这就出现了互连性问题。图6-30显示了级间的互连性需求。

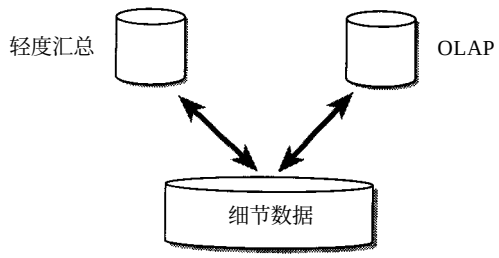


图6-30 数据仓库的不同级次之间的互连性是个重要问题

我们从几个方面来强调互连性问题。一是在调用(Call)级存取的兼容性。换句话说，在数据仓库的任何两级之间构成细节数据和汇总数据时所采用的技术间在调用语法上是否兼容？如果不存在一定程度的调用级语法的兼容性，那么接口的使用将会出现问题。互连性问题的另一个方面是有效带宽。如果数据仓库的每一级都有很重的负载，那么两种环境间的接口将会成为瓶颈。

但是，从事数据仓库开发的小组是相互协作的，需要保证管理低级细节数据的开发小组能为汇总数据和建立新层次数据的开发小组提供一个正确的数据基础。这种要求如图 6-31所示。

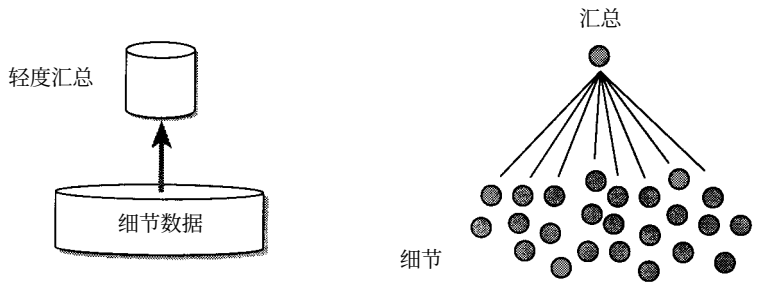


图6-31 细节级数据是建立数据综合级的基础

开发小组间的协作可以像满足各部门的数据需求的数据模型上的一个协议一样简单。如果条件许可，也可制定非常精细的协议。

开发项目本身的协调是另一个问题。不同的开发小组之间需要遵循一定的时间顺序安排，以使所有开发小组在需要数据之时都能获取所需的在较低级上收集到的数据。

6.12 多个小组建立当前细节级

当多个开发小组试图以非分布式的方式建立数据仓库中的当前细节级时，将出现某些特殊情形。图6-32显示了这种现象。

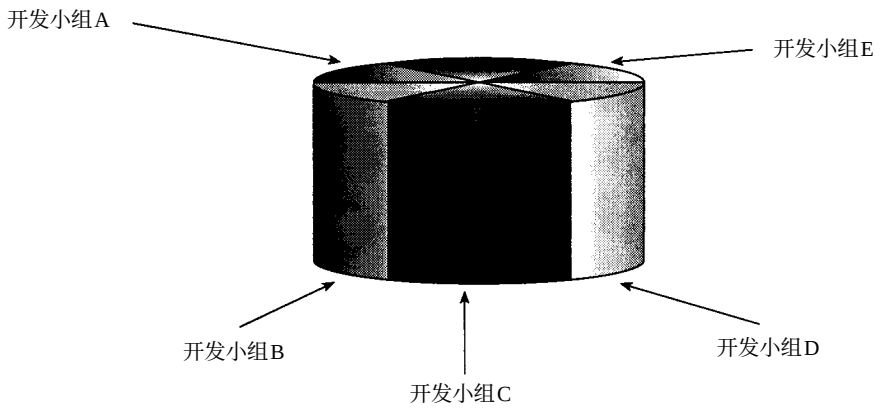


图6-32 不同的开发小组共同建立数据仓库中的当前细节级

只要开发当前细节级的开发小组开发的数据集是互斥的，就不会出现太多问题。在这种情况下，只要这些开发小组使用相同的数据模型，且不同开发者的技术平台间存在兼容性，那么就很少有风险。不幸的是，这种情况很少出现。更常见的是，多个开发小组设计和装载的是一些或全部相同的数据，如图 6-33所示。

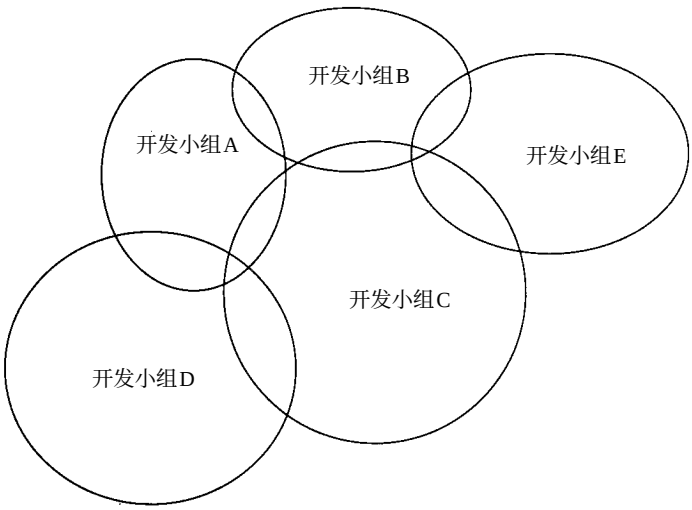


图6-33 不同开发小组开发的数据出现重迭

当数据出现重迭时，将会引起一系列问题。第一个问题是代价，特别是存储和处理的代价。必须考虑当前细节级数据中的所有冗余数据，尽可能减小任何数据冗余量。处理细节的代价也是一个主要问题。

第二个比较糟糕的问题是将蜘蛛网问题引入了 DSS环境。由于存在大量冗余的细节数据，所以自然会造成对数据的错误解释，且没有有效的解决方法。因此在数据仓库中的细节级创建大量冗余的细节数据是非常不期望的状况，这就违背了它的目的。如果多个开发小组并行地设计和装载当前细节级数据，那么一定要确保没有创建冗余数据。

为了确保在细节级不存在冗余数据，必须创建反映共同细节数据的数据模型。图6-34显示了多个开发小组根据他们的兴趣组合共同创建一个公用的数据模型的情形。除了当前活跃的开发小组，将来可能介入的其他开发小组也可能提供他们的需求。当然，如果开发小组知道将来的需求，但是又不能清楚地描述它们，那么这些需求也不会作为公用细节数据模型中的考虑因素。

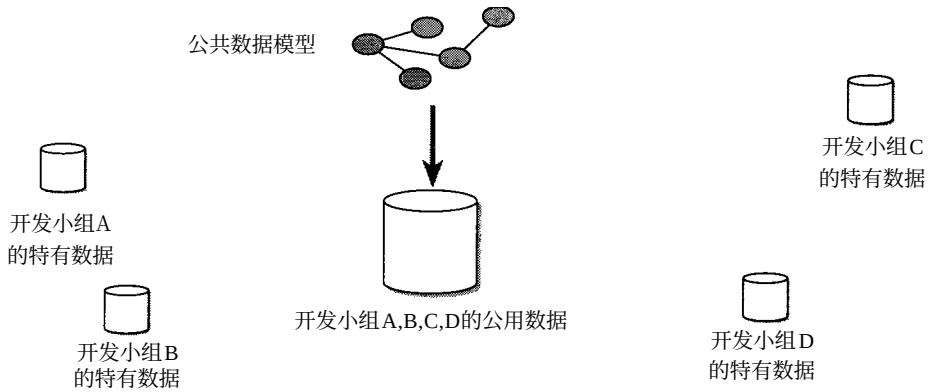


图6-34 对于所有开发小组，数据模型标识公用数据

公用细节数据模型反映了数据仓库中细节数据不同开发者的共同需求。

数据模型构成了数据仓库设计的基石。图 6-35表明在设计过程中数据模型将分割为多张表，它们中的每个均成为数据仓库物理上的一部分。

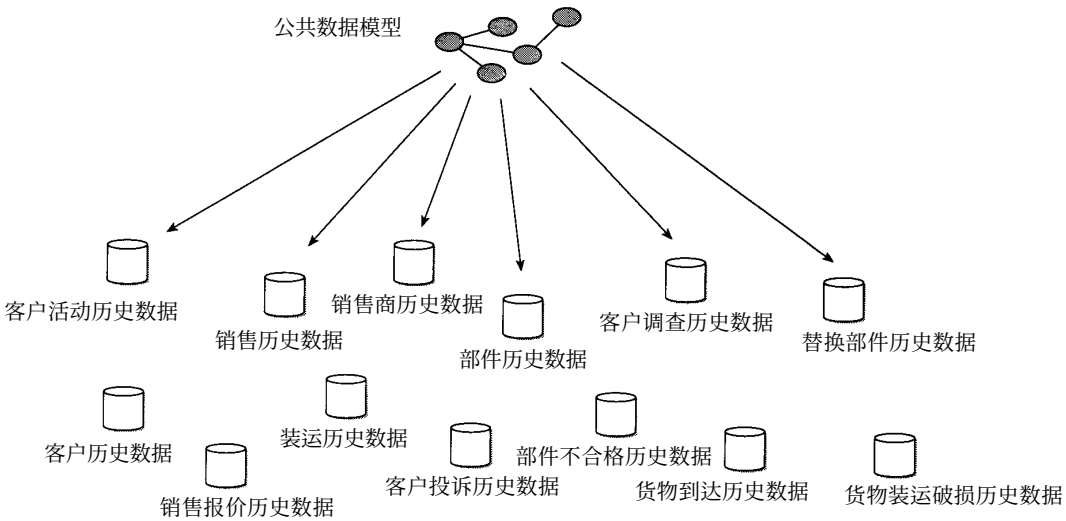


图6-35 数据仓库在物理上分布在多个物理表和数据库中

由于在实现时数据模型分割为多张物理表，所以数据仓库的开发过程采用反复的方法，不需要立即建立所有表。事实上，在一个时期建立几张表有诸多原因，这样处理，可在需要时使用最终用户的反馈信息来有条不紊地修改表。另外，由于公用数据模型分割为多张表，所以在以后的某个时刻增加新表来弥补目前未知的需求不致成为问题。

6.12.1 不同层不同需求

一般来说，不同开发小组的需求不同(见图 6-36)，这些特殊的需求就导致了所谓的“局部的”当前细节级。这些局部数据肯定是数据仓库的一部分。但是它与“公用”部分是截然不同的。局部数据有自己的数据模型，通常比公用细节数据模型更小更简单。

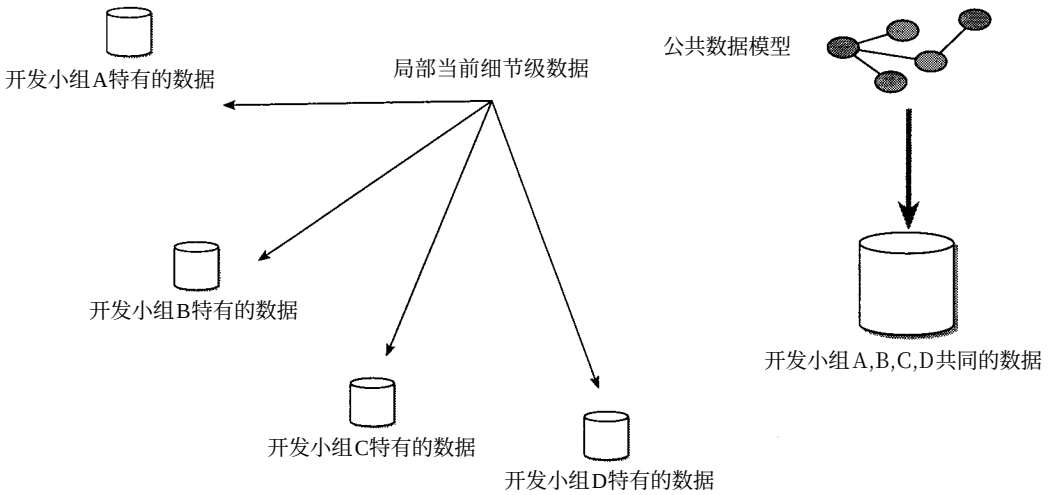


图6-36 数据仓库中的当前细节级包含各开发小组的特有数据

所有的细节数据肯定不存在冗余。图 6-37清楚地说明了这点。

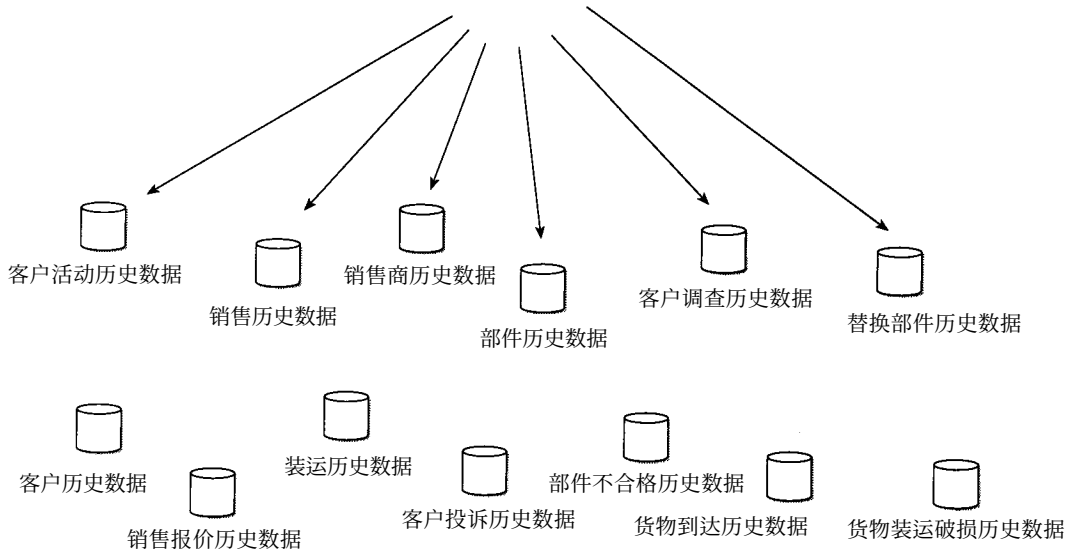


图6-37 构成数据仓库细节级的多张表中非键码数据的非冗余性

当然，数据非冗余性限制于非键码数据。在键码级数据肯定是冗余数据，因为外键码用于将不同的数据类型相关联。图6-38显示了使用外键码关联不同表的情况。

在图6-38的表中发现的外键码与受参照完整性所支配的典型的外键码关系不同。因为数据仓库中收集和存取的是快照数据，出现的外键码关系是以人工关系组织的。

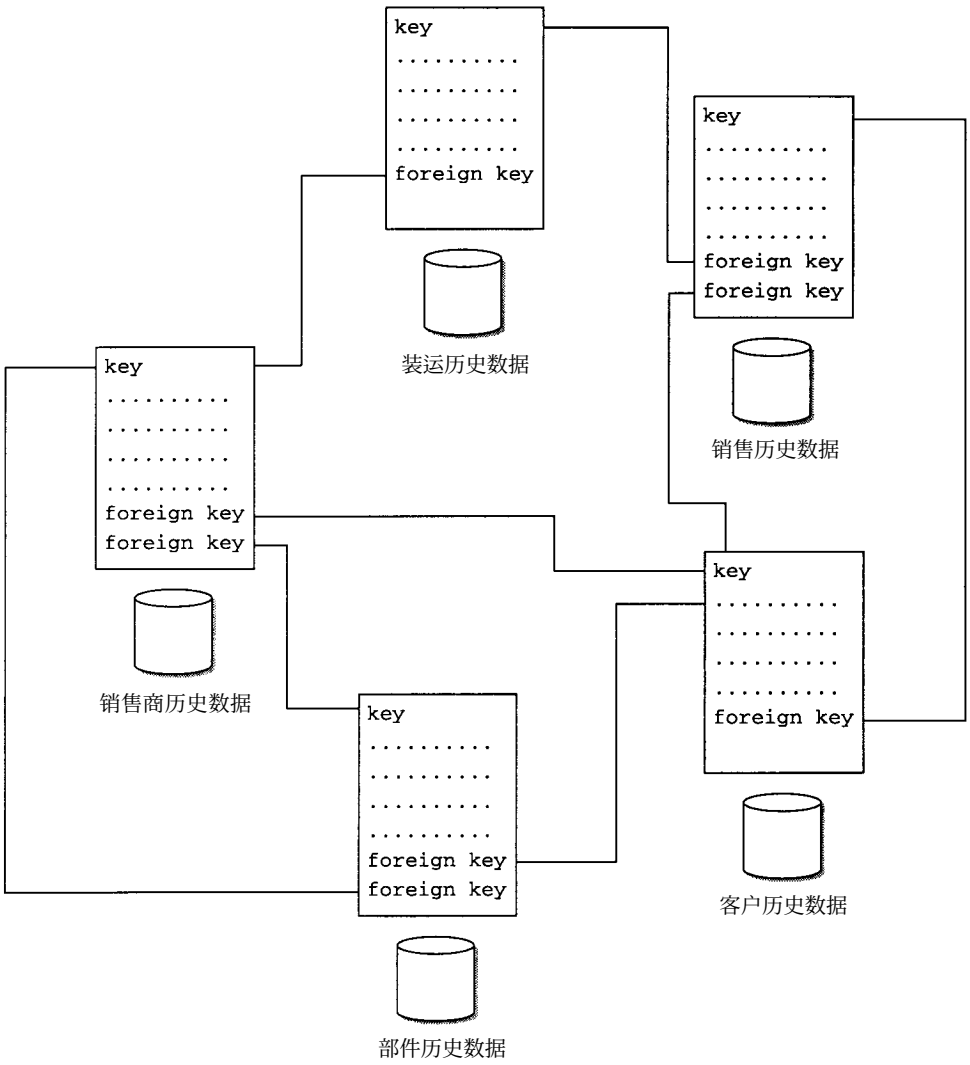


图6-38 数据仓库环境中的外键码

能否采用同样的技术来存放所有的细节表(公用的和局部的)？图 6-39显示了所有表以同样技术存放的情况。以单一技术平台存放所有细节表有许多正当理由。一是单一平台比多个平台代价低，二是维护和培训费用较低。细节数据采用多个平台唯一的原因是，使用多个平台可能将不需要单一的大平台，而多个小平台的代价可能比一个大平台的代价要低。不管怎么说，事实上许多机构为它们的所有细节数据仓库数据采用单一平台策略，并且运行效果很好。

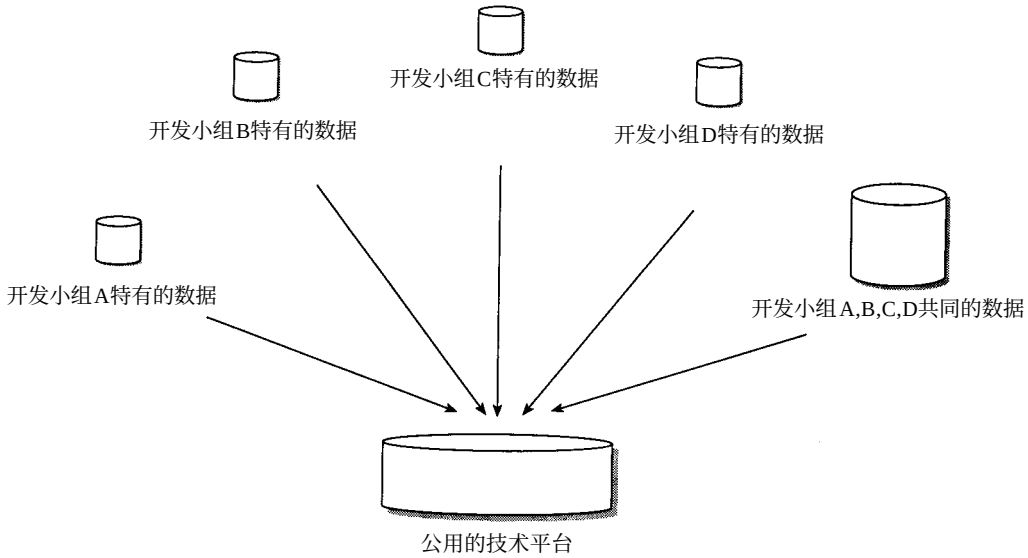


图6-39 数据仓库细节级不同类型的数据都在同一个技术平台上

6.12.2 其他类型的细节数据

另一种策略是对不同类型的细节级数据使用不同的平台。图 6-40显示了采用这种配置的一个实例。一些局部数据使用一种平台，公用数据使用另一种平台，而其他的局部数据采用一种其他的技术平台。这种选择具有一定的合理性，能很好地满足企业内不同的策略需求。采用这种选择，不同的开发小组至少能对自己特有的需求具有一定程度的控制能力。不幸的是，这种选择有几个主要的缺陷：必须购买和支持多个技术平台，最终用户必须接受多种技术培训，技术间的界限很难跨越等。图 6-41表明这种困境。

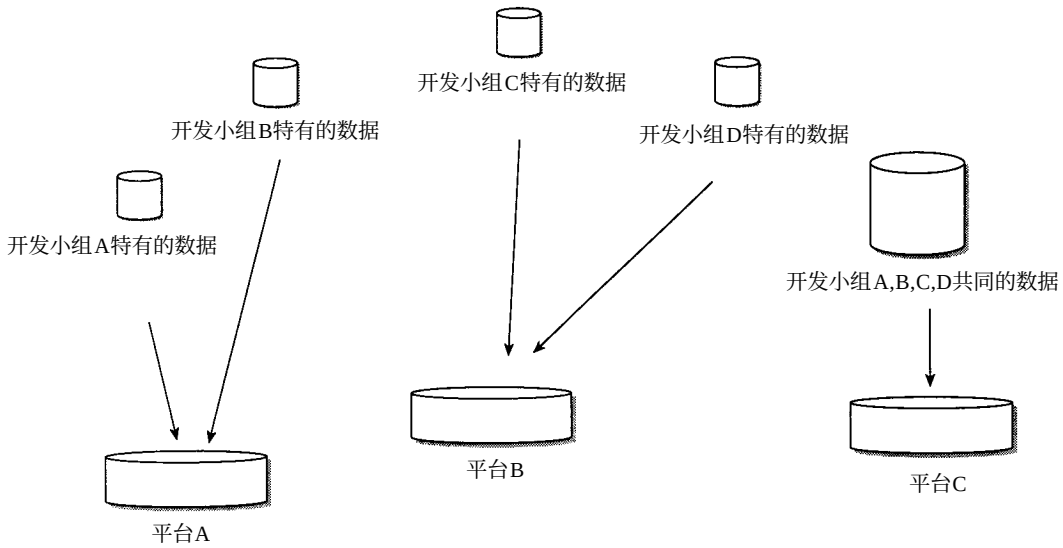


图6-40 数据仓库的当前细节级数据的不同部分分散在不同的技术平台上

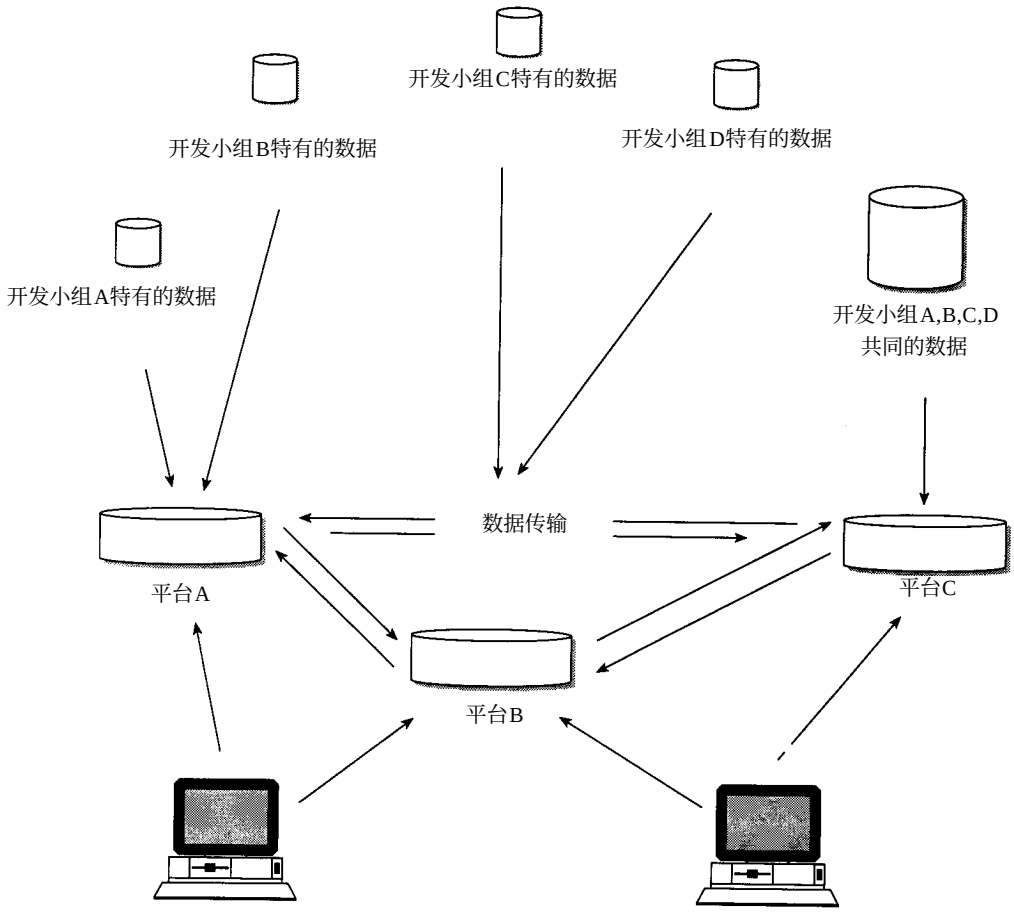


图6-41 数据传输和多表查询出现特殊技术问题

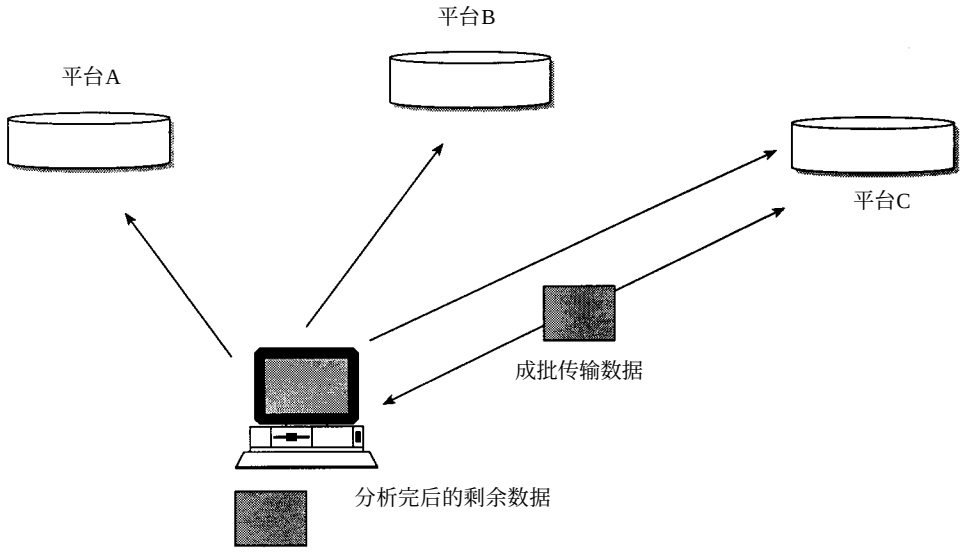


图6-42 不同平台间的接口问题

如果数据仓库中细节的不同级采用多种技术，那么就必须经常跨越技术间的边界。 IBI 的EDA/SQL最终用户技术能很好的适应这种需求。然而不幸的是，即使是像 EDA/SQL这样强有力的软件工具，仍然存在体系结构上的问题。存在的一些问题如图 6-42所示。

问题之一是数据传输。如果EDA/SQL接口用于少量的数据传输，那么效果还可以。但是，如果EDA/SQL用于大量的数据传输，那么 EDA/SQL将会成为性能的瓶颈。不幸的是，在 DSS环境中，对任一个请求都不可能预先知道它将访问多少数据。某些请求可能访问非常少的数据，而另一些可能访问大量数据。当细节数据位于多种平台上时，资源的利用和管理的问题将显现出来。

另一个相关的问题是“ 剩留 ”细节数据，即当细节数据从某地传送到另一个地方时将它们驻留在当地的数据仓库。这种随意的细节数据搬迁将会导致细节级数据的冗余，在一定程度上是不可接受的。

6.12.3 元数据

在任何情况下，无论采用多种技术还是单一技术管理细节数据，元数据扮演的角色都不可忽略。图6-43表明元数据需要位于数据仓库的细节数据的顶层。

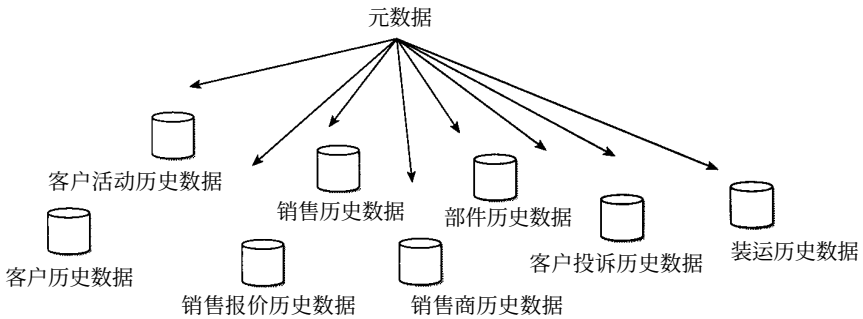


图6-43 元数据位于数据仓库的顶层

6.13 公用细节数据采用多种平台

另一种可能性也是值得考虑的，即公用细节数据采用多种技术平台。图 6-44概括了这种可能性。

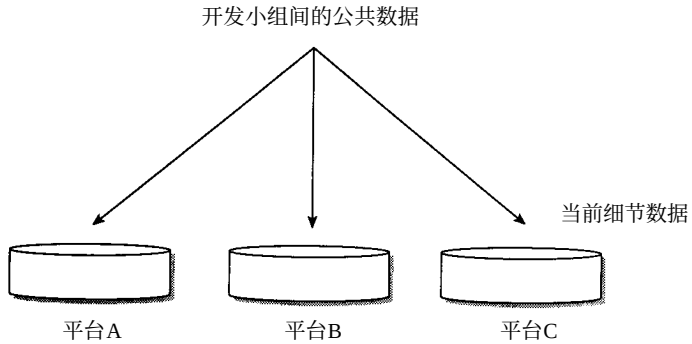


图6-44 公用的细节数据采用多种开发平台

但是这种可能性只是一种选择方案，决不是很好的选择。管理当前公用的细节数据是很困难的。在细节级别出现大量数据将会面临许多特殊管理问题。所以采用多种技术平台只能增加管理的复杂性。一般不推荐使用，除非有特殊的减轻策略。

6.14 小结

大部分企业建立和支持单一的中央数据仓库环境。但是在某些特定场合，建立分布式数据仓库环境可能会取得更大效益。局部数据仓库拥有局部操作站点感兴趣的数据；分布式全局数据仓库的数据及数据结构由全局层决定，而局部层到全局层的数据映射由局部处理。

总之，管理和协调分布式数据仓库环境比管理和协调单一场地的数据仓库环境要复杂得多。