序

我非常高兴能够与信息技术的专家们一起分享 Paulraj Ponniah 的这本书——数据仓库基础。1998 年春天，Raritan Valley 社区学院决定开一门关于数据仓库的课程。这缘于长期担任数据库设计和开发课程的 Ponniah 博士的想法。但是找到一本在数据仓库知识方面适合学院课程的好的教科书，是一件非常困难的事情。我们使用了当时的一本书，虽然并不十分适合我们的课程。为了开好这门课程，Ponniah 博士不得不使用自己在数据仓库方面的研究资料对这本书进行了补充。事实证明，这门课程对我们的学生帮助很大，他们学的也很不错。现在这本非常棒的数据仓库教科书问世了，这要感谢 Ponniah 博士的卓识远见，以及富有洞察力的出版商——John Wiley 和 Sons 的支持。

这本书的独特优点包括：

- 结构安排非常具有逻辑性。
- 内容的安排非常适合于介绍性的书籍，而且内容涵盖面很大。
- 主题经过了精心安排，章节内容包括了数据仓库知识领域的所有基础方面，并且从设计到应用的过渡非常自然而不牵强。
- 在每一章节中，主题的连续性非常棒。
- 本书中所包含的内容都是基础知识方面所必需的。
- 内容准确而紧跟潮流，图表内容准确地增强和说明了主题。
- 每一章节结束的时候，都有丰富的复习问题和练习。这在很多的数据仓库书中是找不到的。这些复习题对于教师教学和学生学习非常有益。

Ponniah 博士的写作风格十分的清晰和流畅。正是由于这本书的简洁和完备，我相信它一定得到大家的欢迎，特别是在高校的学生，信息技术领域的从业者，以及数据仓库专家们一定会对本书的出现感到非常高兴。

虽然数据仓库知识领域已经出版了很多的书籍，但是这本书满足了一个特定群体的需要，而且信息技术的从业者们一定会从此书中受益匪浅。此外，本书也一定会得到大学教授以及学习数据仓库课程的学生们的欢迎。这本书填补了这个领域的空白。

总之，Ponniah 博士为初学者和有经验的 IT 从业人员都提供了很好的学习资料。作为一

个长期工作在信息技术教学领域的学者，我向所有教授和学习数据仓库课程的大学老师和教研室负责人们，推荐此书。

PRATAP P.REDDY, Ph.D.

前言

谁需要阅读本书

你是一直关注数据仓库领域发展变化的信息技术领域的专业人员吗？你在关注一个正在充满机会的新领域吗？你是渴望掌握数据仓库基础知识的系统分析员、程序员、数据分析员、数据库管理员、项目经理或者是软件工程师吗？你想知道你需要阅读多少本书来学习数据仓库基础知识吗？你是否在数据仓库方面浩如烟海的资料和产品中迷失？你希望得到一本简单明了、专门为信息技术领域专业人员设计的数据仓库知识的书吗？你需要一本教科书能够帮助你学习一定深度的数据仓库知识（不能太深，也不能太浅）吗？如果你对上面大多数问题的回答是“是的”，那么这本书就是为你写的。

这是一本专门为信息技术领域专业人员而写的权威的书籍。本书的组织和编写工作都是专门为信息技术领域专业人员设计，在设计之初没有预先为对该领域感兴趣的所有人准备。在内容上，没有特别的着重于某些特定的方面而忽略其它方面。这本书将把你带入一个完整的数据仓库的世界。

这本书是如何做到适应信息技术领域专业人员的需要呢？作为具有丰富 IT 行业实际经验的信息技术领域专家，一个成功的多年从事数据库和数据仓库咨询者，以及在大学课堂和公开的研究会里教授数据仓库基础的老师，我非常了解信息技术领域专业人员的需要，在本书的每一章中我将一一谈到这些信息技术领域中的需求。

阅读背景

为什么这么多的公司都争相引入数据仓库技术？数据仓库已经不再是单纯的仅供科研和实验的概念，它已经成为一种主流的现象。超过半数以上的美国公司和很大一部分的国际贸易都在使用数据仓库。

现在，从零售业商埠到金融机构，从制造业企业到政府部门，从航空公司到公益事业，数据仓库已经成给一种人们从事商业活动分析和进行战略决策的革命性方法。这些拥有数据仓库并基于网络技术的公司，正在利用对重要信息的更快更方便的传递来提高公司的潜力和竞争力。

在过去的五年中，数以百计的供应商涌入了数据仓库产品这个市场，他们提供的解决方案和产品已经包含了数据仓库的整个产品线——数据建模，数据获得，数据质量，数据分析，元

数据等等。这一市场已经很大，并且还在继续增长。

角色的改变

在所有成长型公司中，信息技术部门感受到了自己角色的根本性变化。IT 不再是需要创造每一份报告，给最终用户提供信息。现在的 IT 部门的职责是建造信息传递系统，使最终用户自己通过革命性的方法来得到分析和决策用的信息。数据仓库正在被证明是成功的信息传递系统的类型。

负责建造数据仓库的 IT 专业人员需要转变他们关于开发应用的观念。他们不得不明白，一个数据仓库不是一个设计就可以适合全部的东西，他们必须对于如何从原数据系统里抽取、转换、展现数据，以及数据仓库的体系、结构，甚至很多信息传递的方法等等很多问题有一个清晰的认识。

总之，IT 专业人员必须对于数据仓库的基本原理有相当的了解。

这本书可以给你带来什么

这本书可以说是全面和详细的，你可以学到有关数据仓库每一个重要部分的内容，包括计划、需求、体系、结构、设计、数据准备、信息传递、配置和维护等等。这本书是专为 IT 行业的专业人员所设计的：你可以很容易的跟上进度，因为它是根据 IT 行业的专业人员的背景、知识上设计的，你对那些基础数据非常熟悉。它在结构上非常有逻辑，从对概念的总的介绍入手，依次介绍了计划和需求，然后是结构和体系，再到数据设计，信息传递，最后以数据仓库的配置和维护结束。这一进步的过程使你更加了解你的实验和每天的工作。

本书还提供了互动的学习实验。它不是单纯的讲授，你可以通过参与每一章结束时的复习思考题和练习来参与这种互动。在每一章中，你可以将每一个概念和技术同数据仓库的实践和市场充分的结合起来。你将会发现很多实际应用中的例子。虽然本书是作为你踏入数据仓库领域里的第一本基础书籍，但是本书通过涵盖几乎这个领域所有方面的知识，因此将为你在数据仓库领域成为专家迈出非常坚实的一步。

另外，本书还非常适合作为教科书，供自学人员，大学课程或者科研机构使用。它将为你提供一个成为数据仓库领域专家的机会。

在这里，我要感谢我在本书末尾提到的作者们，你们的工作给了我很大的帮助。我必须也向我的学生和同事们表示我的谢意，大家亲密无间的合作帮助我使这本书更好的适合 IT 行业的专业人员的需要。

Paulraj Ponniah, Ph.D.

2001 年 6 月

第一章 对数据仓库的迫切需求

本章目标：

- 理解目前对战略性信息的迫切需求
- 认识每个企业的信息危机
- 能够区分操作型和信息型系统
- 学会为什么过去提供战略信息的尝试都以失败告终
- 搞懂为什么数据仓库是可行的解决方案

作为一个信息技术的专业人员，你一定在电脑应用行业做过系统分析员、程序员、设计人员、开发人员、数据库管理员或者项目经理等等。你一定曾经做过设计、应用或维护系统的工作，来支撑每天的商业运作。依靠你所工作的企业，你一定也做过例如订单处理、会计帐务、盘点存货、核对帐目、保险申请等等工作。

这些应用都是企业在经营中的重要系统。如果没有这些系统，日常工作中的订单处理、盘点存货、客户服务、接收付款以及处理赔偿等现代商业行为都无法正常开展。这些日常系统从 1960 年开始建立和使用这些系统，并且已经长期以来一直依赖它们。企业成长得越大，各种的商业处理工作越需要这些数以百计的计算机应用系统。这些应用系统工作起来非常有效，但是仅限于它们当初被设计的范围。收集，存储和处理所有这些数据需要成功的完成日常的操作工作。他们提供在线信息并且向领导提供大量的报告。

在二十世纪九十年代，商业活动变得越来越复杂，公司企业迅速的全球化，竞争也越来越激烈，商业经理们渴望得到更多的信息来提高在商业活动中的竞争力。日常的操作型计算机系统提供大量的信息来支持每天的工作，但是与这些信息不同，经理人员需要的是可以用来进行战略决策的信息。他们想知道应该在哪里建立下一个仓库，需要扩大哪一条生产线，应该加强哪一个市场。这些曾经非常重要的操作型系统并不能提供有战略意义的信息。因此，商业活动迫切的需要能够带来战略信息的新的方法。

数据仓库是一种能够提供重要战略信息的新的范例。二十世纪九十年代，一些组织开始从建立数据仓库系统中得到富有竞争力的优势。图 1-1 给出了一个战略领域的例子，我们可以看到数据仓库已经正在不同的行业发挥作用。

获得优势的行业

*零售业

客户忠诚度

市场策划

*金融行业

风险管理

欺诈防止

*航空业

航线收益率

区域管理

*制造业

成本控制

后勤管理

*公益事业

资产管理

资源管理

*政府机关

人力计划

成本控制

图 1-1　使用数据仓库的行业

我们将会提出这样一个问题：为什么企业真的需要数据仓库？这一讨论非常重要，因为除非我们掌握了这种需求的重要性，否则我们对于数据仓库的学习就会缺乏动力。所以，请对于这个问题保持足够的注意。

对战略性信息的迫切需求

在我们讨论企业的战略信息的时候，我们可能需要看一看目前非常普遍的信息危机，这种危机就像几年前的技术趋势一样，曾经给我们带来阻力，现在能够帮助我们提供战略性的信息。只有当我们明白了战略信息所能够提供的机会和没有这些信息我们将会遇到的风险

后，我们关于战略信息的讨论才有可能停止。

企业中究竟是谁需要战略信息？我们对战略信息的定义究竟是什么？企业的经理们和管理者需要适当的信息来保持企业的竞争实力。他们需要这些信息来规划企业的战略，设定并建立目标，以及监控结果。

这里有几个关于企业目标的例子：

- 保住现有的客户资源；
- 在 5 年的时间里提高 15%的客户数；
- 在 3 年的时间里提高 10%的市场占有率；
- 将产品质量水平提高到前五名；
- 在出货环节提高客户服务水平；
- 在 2 年的时间里推出 3 种新产品；
- 在东北市场提高 15%的销售额；

对于这些目标的决策制定，经理和管理者们需要这样的一些信息：对他们公司的运营有一定深度的了解；了解公司运营的关键因素和它们之间的关系；掌握这些因素是如何随时间变化；将公司的运营状况和市场竞争联系起来进行比较。经理和管理者们需要将他们的注意力集中在客户的需要和喜好上，而从具体的技术、销售、市场结果、质量和服务水平等事务中脱离出来。这种在制定商业战略和目标时需要信息的类型，需要整个企业内部的广泛的基础。我们可以将这些信息归入一个统一的概念，称它们为战略信息。

战略信息不是企业日常操作工作中的信息，不是那些订货，发货，处理投诉或者从银行账户提款的信息。战略信息比这些信息重要很多，对于企业的持续生存和发展有非常关键的意义。企业中重要的商业决策依赖于正确的战略信息的提供。下图 1-2 列出了战略信息的一般特性。

综合的	必须有一个独立的、从企业整体来看的视角
数据完整性	信息必须是正确的，必须符合商业规则
容易取得	必须是通过普通方法容易获得的，对于分析工作是有用的
可信的	每一个商业因素必须都有一个而且只有一个值
适时的	信息必须是在一个规定时间范围内的

图 1-2　战略信息的特征

信息危机

可能你是一个在信息技术部门工作很长时间的专业人员，或者你是一个中等公司的一部分。无论你的公司有多大的规模，考虑你的公司中的所有的计算机应用系统，考虑所有的数据库和公司中支撑运营的所有数据。你们储存了多少年的客户数据？存储了多少年的金融数据？十年？十五年？所有这些数据都在哪里？在一个平台中吗？在旧系统中吗？在客户机/服务器结构的应用中吗？

我们现在面临两个事实：（1）公司有很多很多的数据；（2）信息技术资源和系统不能将所有这些数据转化成有用的战略信息。在过去的两个年代中，很多公司集聚了大量的关于运营的数据，据说信息在 18 个月内就会翻一番。

如果我们拥有这么巨大的数据量，存在着这么多的信息，为什么我们的经理和管理者们还不能利用它们进行战略决策？为什么我们还谈到信息危机的问题？很多面临信息危机的企业并不是因为缺乏足够的信息，而是因为能够得到的信息并不是为战略决策制定准备的。这些大量的数据对于企业的运作非常有用，但是对于商业战略和目标的制定就没有什么太大用处了。

为什么是这样呢？首先，企业的数据包括了很多种类型的结构和系统。你的订单处理系统可能是在 20 年前开发的，现在依然在大型机上运行。其中的有些数据可能还是 VSAM 文件格式的。你最近的信用分配和认证系统可能是在一个客户机/服务器平台上运行的，里面的数据可能是在关系表中。一个公司的各种数据存在各种相互独立的系统、平台中，而且它们的结构也各不相同。你的公司在过去使用的技术越多，公司里的数据就越不相同。但是，为了在公司中进行正确的战略决策制定，我们需要整合所有系统中的数据。

进行战略决策所需要的信息必须是适合进行分析的工作格式。经理和管理者们需要长期地观察企业的发展趋势，并保持公司在一个正确的轨道上运行。大量的可以得到的操作型数据是不能直接用来反映趋势的，因为我们说操作数据是事件驱动的。你可以得到特定时间发生的事情的信息，可以得到在某个订单下向某个特定客户销售的特定产品的信息。在操作型系统中，你不可能得到某个产品在一个月、一个季度或一年内的销售趋势数据。

对于战略决策的制定来说，经理和管理者们必须能够看到从不同商业观点和角度观察的数据。举个例子来说，他们必须能够回顾产品的销售数量，销售对象，地区和客户群。你能够设想操作型的数据可以进行这种分析吗？操作型的数据不能直接适合不同角度的回顾查询。

技术趋势

在信息技术领域工作了二三十年的专业人员已经目睹了这些年来发生的巨大变化。首先，企业中的计算机部门的名称从“数据处理”转变成“管理信息系统”，然后到“信息型系统”，一直到最近的“信息技术”。计算机发展的全过程经历了巨大的变化。过去的应用已经不能适应新的需求了。预先定好格式的报表不再能够满足使用者的要求。

这些年，MIP 的价格正在持续的下跌，数字存储的开销越来越小，网络的带宽随着价格的下跌而增长。特别是，我们已经看到了这些重要领域的革命性变化：

- 计算机技术
- 人/机接口
- 处理选择

下图 1-3 表现了这种变化和增长。

计算机技术

大型机　　小型机　　个人电脑/网络　客户机/服务器

人/机接口

打孔机卡　视频播放　　图形用户界面　语音

处理选择

分批处理　联机　　网络

图 1-3　信息技术的迅猛增长

我们现在处于技术革命的什么位置？随着硬件的越来越便宜和小型化，现在每个桌面上都有一个工作站，价格越来越低，作用却越来越大。新的软件提供了易于使用的系统，开放的系统体系结构创造了合作的机会，使更多的软件产品能够参与其中。提高的连接能力，网络性能以及 Internet 的普及使得大量系统和数据库的交流成为可能。

技术方面的所有进步都是值得称赞的，这使得计算的速度更快，更便宜而且更方便的普及。接下来，让我们了解现在的技术阶段是如何帮助我们得到战略信息的。

提供战略信息需要大量的数据，并且这些数据需要存储为一个合适的格式。数据存储方面技术的发展和存储成本的减少，为战略决策支持系统需要的数据存储提供了条件。分析者，管理人员和经理们可以直接使用战略信息来进行分析并预测商业的发展趋势。使用者可以提出一个问题并得到一个答案，然后问另一个问题，检查结果，再问另一个问题。这种互动的

处理可以一直继续。接口软件技术的发展使这种互动的分析成为可能。处理大规模的数据和提供互动的分析操作要求特别的计算能力。这种计算能力的迅猛发展和它的成本的迅速降低，将使系统能够提供战略信息。我们几年前不能做到的提供战略信息的工作，现在使用先进的信息技术后就可以完成了。

机遇和风险

我们已经看到了每一个企业都存在着信息危机，已经掌握了虽然企业中有很多的操作型数据，但是并没有适合战略决策制定的数据。然而，现在的技术发展已经使提供战略信息成为可能。当我们在讨论公司对于战略信息的需求必要性的时候，让我们问一些基础的问题。企业使用这些战略信息得到结果后究竟可以得到哪些机会？企业如果没有利用这些战略信息做出决策将会遇到哪些挑战和风险？

这里有几个例子，说明企业通过利用这些战略信息可以得到哪些机会：

- 一个经营长途电话的公司授权它的销售人员去制定更好的商业决策，在充满竞争的巨额利润市场中捕捉商业机会。一个通过网络得到的解决方案利用了内部和外部的数据来提供战略信息。
- 拥有 2500 亿美元资产的美国最大的商业银行使用战略信息可以允许决策者做出最快的决策来保留他们的有价值客户。
- 在大型健康管理组织的例子中，健康照顾项目得到了显著的提高，减少了 22%的紧急出诊，减少了 29%对于患哮喘症儿童的住院治疗，减少了糖尿病患者的发病可能性，提高了接种疫苗率，每年为医生和药剂师创造了超过 100,000 的情况报告。
- 在全美前五位的零售商中，与网络技术结合的战略信息分析工具使贸易商对于客户的了解更加深入，对存货的管理更加紧密，而且能够使正确的商品在正确的时间出现在正确的顾客面前。
- 一家要在本地区与 800 家授权的药剂店竞争的社区药剂店，想要对客户购买什么药进行深入的了解，以此帮助降低存货，提高竞争实力，最终提高公司的利润。

从另一方面看，让我们看看下面的几个例子，在使用战略信息进行分析和决策之前存在很多风险和失败的威胁：

- 一个拥有平均 150,000 辆汽车的汽车租赁公司，如果不能对车队进行有效的管理，那么公司就很容易出现赤字。在激烈的竞争中，只有有效的管理好车队，才能避免

失败的危险。汽车的空闲时间必须被严格控制在一个小范围内。如果不能在规定时间内将已经清洁好的车子送到规定的地点，就可能导致严重的损失。

- 对于一个全球范围的汽车和轻型卡车零件制造商的部件供应系统来说，有大约 100 个零件与统计数据不一致，不能达到预先的质量要求，或者耗费大量时间的手工数据收集，都将会带来很大的麻烦。可能会需要几个星期才能将报表完成以供决策使用，而且取得很难取得整个公司范围的综合数据。
- 一家大型公用事业公司，向 25,000,000 个用户提供电力，为了保持竞争力甚至说为了在激烈的竞争中生存，必须依靠来自于不同的来源的统一的战略信息、流线型的数据存取、以及企业分析用数据的。

过去决策支持系统的失败

公司的市场部门已经从当月的月报表发现东海岸区域的表现和销售额非常的低。市场部副经理很不安，想从 IT 部门得到一些报表来分析过去两年的每一项产品的情况。他想要做出快速的战略决策来挽回局势。首席信息执行官希望你的上司尽快发布报表。你的上司找到你并要求你停下手上的任何事情，专门做报表。在系统里面没有任何符合市场部需要的已经存在的报表形式。你必须从这么多的系统应用中自己提取数据从零做起。这样的情况是不是听上去很熟悉？

在你的职业生涯中，你肯定遇到过这样的事情。有的时候，你可能可以直接从数据库或者别的什么文件系统中得到这种特别查询所需要的信息。但事情往往不是这样的。你可能需要找到好几个系统应用程序中，甚至需要查询公司中的几个不同的平台，才能得到想要的信息。接下来将会发生什么？市场部门喜欢你提交的特别查询报表。但是现在他们想要另外一种形式的表格，包含更多的他们事先没有想到的信息。在第二轮的工作中，他们发现报表的内容还不是他们真正想要的。他们可能也发现了从不同的系统中取出的数据之间存在着矛盾的地方。

事实是在二十年或者更多的时间里，IT 部门一直在努力为公司提供用于制定战略决策的信息。有时候一个 IT 部门可以从一个系统中提取数据制作特别查询报表。而在更多的时候，需要从很多的系统中寻找制作报表所需要的数据，甚至需要写一些额外的代码来创造临时文件，从中得到特别查询报表所需要的数据。

过去 IT 部门在这方面所作的大部分工作都是以失败告终的。最终用户在开始的时候不

能十分明确的表明他们到底要什么。一旦他们看了产生的报告，他们就会想得到不同格式的更多的数据。这种情况会一直循环下去。因此，用来进行制定战略决策的信息应该通过一种互动的方式得到。最终用户应该可以在线查询，得到结果，并且再进行查询。信息必须采用一种适合分析的格式。

为了找到过去决策支持系统失败的原因，我们需要考虑这些年来信息技术是如何完成人们需求的。因此，接下来让我们了解一下决策支持系统的历史。

决策支持系统的历史

根据公司的规模和特点，大多数公司已经走过了下面这些阶段来为决策提供战略信息。

特别查询。这是最早的阶段。最终用户，特别是市场和金融部门的用户，会向 IT 部门提出特别的报告要求。IT 部门需要针对每一个需求，编写特别的程序，来提供特别查询。

特殊提取程序。这个阶段 IT 部门希望能够制作某种报表格式来满足一般的查询需求。IT 部门一般会写一些程序，然后定时性的运行这些程序，从不同的系统中提取数据。IT 部门可以通过创建和保存这种已经提取好的文件，来满足特别报表的需求。如果遇到了一些不能使用这些已经提取好的文件得到的报表，IT 部门就要写一些单独的程序了。

小应用程序。在这个阶段，IT 部门将这些提取处理进行了格式化。IT 部门基于这些已经提取好的文件开发了一些小的应用程序。报表打印程序可以使用用户定义的参数来打印信息。一些高级程序可以允许用户通过联机的屏幕来查看信息。

信息中心。在二十世纪七十年代早期，一些大公司创建了信息中心。用户可以向信息中心提出特别查询的请求或者在屏幕上查看信息。这些是预先定制好的报表。IT 专业人员会在这些信息中心帮助用户得到希望得到的信息。

决策支持系统。在这个阶段，一些公司开始建立更加复杂的系统来提供战略信息。与最初的尝试一样，这些系统是由提取好的文件组成，使用菜单形式，提供在线信息，也有能力打印特别的报表。大多数这种决策支持系统是为市场工作人员服务的。

管理信息系统。这是一个试图将战略信息展现到经理们的桌面的尝试。主要标准是简单而易于使用的。该系统提供每天的关键信息，也提供简单直接的报表查询。但是，这些报表是预先设定好的。在看完了整个地区的销售情况后，如果经理想看看关于地区、产品或者其它维度的分析时，除非这种细目分类是已经定制好的，否则就没有可能提供这种服务。这种局限管理信息系统在很多公司都没有能够存在很长时间。

不能提供信息

过去的每一个试图提供战略决策信息的系统都不能令人满意。图 1-4 表现了 IT 部门的这些尝试的过程。作为一个 IT 专业人员，我们对于这些情况非常熟悉。

这里有一些与不能提供战略信息相联系的因素：

- IT 部门接到了太多的特别查询要求，使负荷过重。在有限的资源里，IT 部门不能及时地响应这么多的请求。
- 这些请求不仅是大量，而且它们还老是在变化。用户需要更多的报表来扩大和理解前面的报表。
- 用户发现他们总是在寻求越来越多的额外的报表，所以有时他们试图寻找每一个可能的组合，这样只会增加 IT 部门的负担。
- 用户不得不总是依赖 IT 部门来提供信息。他们自己并不能直接的接触信息。
- 适合制定战略决策信息的信息环境必须是非常的灵活和有益于分析的。但是 IT 部门并不能提供这样的环境。

操作型系统和决策支持系统

为什么这些试图提供战略信息的尝试最后都失败了？我们不能提供战略信息的根本原因是我们一直试图从操作型系统中提供战略信息。这些操作型系统，例如订单处理、存货控制、呼叫处理、门诊病人次序安排等等，都不是为提供战略信息而设计的。如果我们需要提供战略信息的能力，我们必须从所有这些类型的系统中得到信息。只有专门设计的决策支持系统或者信息型系统才能提供战略信息。让我们来看看为什么。

使商业运转起来

操作型系统是联机处理系统（OLTP），这些系统过去用来运行公司每天的核心业务。因此它们也被称为面包黄油系统。操作型系统使得商业运转起来。（见图 1-5）它们支撑着公司基本的商业运作。这种系统将数据存入数据库。每一项处理都将信息作为一个独立的实体，例如一个订单，一张发货单，或者一个客户。

监视商业的运转

另一方面，专门为决策支持设计和建造的系统都不能支撑核心商业处理。它们过去被用来监视商业的运转，然后制定战略决策来促进商业的进程。（见图 1-6）。

决策支持系统用来从数据库中取得战略信息，相反 OLTP 系统是将数据放入数据库中。决策支持系统是为提供战略决策所开发的。

不同的范围，不同的目的

因而，我们发现为了提供战略信息，我们需要建造与操作型系统不同的信息型系统。像我们过去那样，希望能从操作型系统中得到战略信息是没有价值的。目前企业的竞争面对更加严峻的竞争，商业变得越来越复杂，继续我们过去的想法只会将我们带入失败。

导入数据

使商业运转起来

- 下订单
- 处理呼叫
- 出货
- 发货单
- 收取现金
- 预订一个航空座位

图 1-5 操作型系统

导出数据

监视商业的运转

给我销售量最好的产品名单

告诉我那些地区出现了问题

告诉我为什么（向下钻取）

让我看看其它的数据

展示最大的利润

当一个地区的销售低于某个值时，提醒我

图 1-6 决策支持系统

我们需要设计和建立这样的信息型系统

- 为不同的目的服务
- 它的范围是不同的
- 数据内容不同
- 数据使用格式不同
- 数据获取类型不同

在图 1-7 中总结了传统的操作系统和目前需要建设的信息型系统之间的不同。

	操作型	信息型
数据内容	当前值	存档的，导出的，总结的
数据结构	适于事务处理的	适于复杂查询的
连接频率	高	中下
连接类型	读取、更新、删除	读取
用途	可预知的，反复性的	特别查询，偶尔的，启发式的
响应时间	低于秒	几秒到几分钟
用户	大量	相关的少数

图 1-7 操作型和信息型系统

数据仓库——唯一可行的解决方案

通过上面的讨论，我们现在认识到我们确实需要多种类型的决策支持系统来提供决策信息。制定战略决策需要的这种类型的信息不同于我们能从操作型系统中得到信息。我们需要一个新的类型的信息环境，来提供分析、了解趋势和监控性能的战略信息。

让我们来检查这种新类型系统环境的特点和处理需求。同时，让我们来看看这种为战略信息设计的系统环境的优点。

一种新类型的系统环境

这种新类型的系统环境的特点如下：

- 为分析任务设计的数据库
- 从多种应用程序中的得到的数据
- 方便使用，有益于用户的长时间互动操作
- 深入读取的数据使用
- 不需要 IT 顾问，用户可以与系统直接互动
- 同时包含当前和历史数据
- 用户可以运行查询并在线得到结果
- 用户可以创建报表

新环境的需求处理

新环境的大多数处理是分析性的。这里有关于需求处理分析的四个层次：

1. 运行简单查询和报表
2. 可以用很多不同的方法运行“what if”分析
3. 可以查询、后退、分析，然后继续进行任意长度的处理
4. 发现历史趋势，并将它们应用到未来的结果中

数据仓库的商业智能

这种新的系统环境就是数据仓库的新范例。建造数据仓库的企业实际上就正是在建造这种新的系统环境。这种新的环境与以前支持每天操作的环境是完全不同的。数据仓库给企业带来了商业智能，帮助企业进行战略决策。数据仓库是唯一可行的解决方案。我们可以清楚地看到以前基于操作型系统的提取数据的解决方案是多么的令人不满。图 1-8 表现了数据仓库中的商业智能。

操作型系统　　抽取，清洗，聚合　　关键度量，商业维度

基本商业处理　　数据转换

图 1-8 数据仓库的商业智能

从一个高的角度来看，数据仓库通过商业维度的储存包含了商业处理的很多关键过程和度量。举一个例子来说，一个数据仓库可能包含了销售的部分，分别以产品、日期、顾客群体、销售地区和业务增长等为分析维度。

数据仓库从哪里取得这些数据？事实上，这些数据是从支持基本商业处理的操作型系统中抽取的。在操作型系统和数据仓库之间，存在一个数据准备区域。在这个区域中，操作型数据被清洗，并转化成为适合数据仓库使用的格式。

数据仓库的定义

我们已经了解了为什么说数据仓库系统是提供战略信息的唯一可行方案。所以，接下来让我们来学习数据仓库的功能定义。

数据仓库是一种信息化环境，它有以下特点：

- 提供对企业的一个综合而且完整的观察
- 使企业决策所需要的无论是当前数据还是历史数据都方便易得
- 使决策支持的互动工作不再需要借助操作型系统
- 使企业的信息保持一致性
- 提供了一个灵活的和互动的战略信息来源

一个关于信息传递的简单定义

在最终分析中，数据仓库是一个简单的概念。它源于对战略信息的需求，并且对一种新方法来提供这种信息的寻找结果。在过去的二十年中，使用操作型计算机环境的方法一直不能使人满意。这种新的概念并不是创造新的数据，而是通过使用大量的已有数据，将其转变成适合提供战略信息的格式。

数据仓库可以回答用户提出的问题，比如关于经营情况、不同方法的表现情况、商业趋势，以及可以采取什么措施来促进经营等等。数据仓库可以向用户提供对数据的直接接触，提供一个单独的关于经营情况的指标，还可以准确地记录发生的情况，可以使客户能够从多种不同的视角观察这些数据。简单地说，数据仓库可以支持决策型的处理工作。

给数据仓库下一个定义：使用所有已经存在的数据，通过清洗和转化，提供有用的决策信息。

一个环境，而不是产品

一个数据仓库不是一个你购买来提供战略信息的简单的软件或者是硬件产品，而是一个用户可以发现战略信息的计算机系统。在这个环境里，用户可以通过与数据的直接接触来做出更好的决策。它是一个以用户为中心的环境。

让我们来总结一下这种叫做数据仓库的新型计算机环境的特点：

- 数据分析和决策支持的完美环境
- 不固定，灵活而且交互式
- 百分之百用户驱动
- 非常适合提问-回答-再提问的模式
- 提供回答复杂、不可预测的问题的能力

多种技术的混合

让我们来重新看看关于数据仓库的定义。数据仓库的基本概念如下：

- 从操作型系统中提取所有数据
- 在需要的时候，可以将外部相关数据包含其中，例如工业标准指标
- 将多种数据源的数据进行整合
- 清洗并转换数据
- 用适合决策的格式存储数据

在这个简单的概念中，包含了几个不同的功能：数据抽取，数据装载的功能，数据转换，数据存储，以及提供用户接口。因而，需要不同的技术来支持这些功能。从图 1-9 可以看到，数据仓库是如何将提供这些功能需要的多种技术混合起来的。

虽然使用了很多的技术，但是它们全都是在一个数据仓库里共同工作的。最终结果是这个为每个需要的企业提供战略信息的计算机系统创造的。在这里提到的每一项技术中，都提供有几个工具包，你不需要从零做起建设你的数据仓库。

本章小结

- 企业不顾一切的寻找战略信息来应对越来越激烈的竞争环境，扩大市场份额，并且提高企业利润。

- 虽然企业在过去的几十年里积累的大量的数据，但是还是遇到了所谓的信息危机。不能提供进行决策分析所需要的数据。
- 过去的所有试图提供战略信息的尝试都失败了。这主要是因为 IT 一直试图从操作型系统中提供战略信息。
- 信息型系统与传统的操作型系统不同，操作型系统不是为战略数据设计的。
- 我们需要一种新的计算机环境来提供战略信息。数据仓库就是这种新型计算环境。
- 数据仓库是唯一可行方案。每一个企业都有对数据仓库迫切的需要。

思考题

1. 我们提到的战略信息是什么意思？试举出一个商业银行的五种战略目标。
2. 一家典型的零售店通过操作型系统收集了大量的数据，试举出三种交易数据。
3. 考虑一家医务中心利用战略信息提供的机会，你可以列出五种吗？
4. 为什么过去的所有试图提供战略信息的尝试都失败了呢？列出三种原因，并作解释。
5. 描述操作型系统和信息型系统的五种不同点。
6. 为什么操作型系统不适合提供战略信息？给出三个理由，并作解释。
7. 列出提供战略信息需要的计算环境的六个特点。
8. 在数据仓库中有哪些处理？试着描述它们。
9. 讨论一个数据仓库是一个环境，而不是产品。
10. 数据仓库是解决信息危机和提供战略信息的唯一可行的方法。列出四个理由来支持这个观点，并且解释它们。

复习题

1. 将下面的词组配对：

1.信息危机	A．OLTP 应用程序
2.战略信息	B．产生特别查询报表

3.操作型系统	C．爆炸式成长
4.信息中心	D．尽管大量的数据
5.数据仓库	E．数据清洗和转换
6.订单处理	F．用户取得信息
7.经理信息系统	G．用于决策
8.数据准备区域	H．环境，而不是产品
9.抽取程序	I．为了每天的操作
10.信息技术	J．简单，易用

2. 当前的硬件/软件技术的发展使数据仓库变得可行。举几个例子来解释技术的发展是如何帮助数据仓库的。
3. 假如你是一家保险公司的 IT 部门经理。给执行副总裁写一个备忘录，解释利用战略信息可以带来哪些机会。
4. 对于一家航空公司，如何利用战略信息来提高常见的飞行员数量？讨论并给出一定的详细解释。
5. 你是一家汽车制造公司 IT 部门的高级分析员。市场部副总经理总是在抱怨 IT 部门不能及时提供战略信息。起草一份建议给他，解释一下这个问题的原因，并解释为什么建立一个数据仓库是唯一可行的方案。

第二章 数据仓库的组成部分

本章目标

- 复习关于数据仓库的定义
- 讨论定义的特点
- 区别数据仓库和数据集市
- 学习数据仓库的每一个组成部分
- 介绍元数据的概念和它的重要性

我们在上一章中已经学到了，数据仓库是一个信息传递系统。在这个系统中，你将企业数据整合并转化为适合制定战略决策的信息。你从各种各样的系统中提取历史数据，然后将相关外部数据和内部数据综合起来。你还需要处理不同系统中的数据不一致的矛盾，然后将这些整合的数据转化成一个统一的格式，适合向各种层次的用户提供信息。最后，你将实现信息的传递过程。

为了建立这个信息传递系统，你需要不同的组成部分。这些组成部分通过最佳的方法排列在一起，有一个合适的体系结构。在我们了解这些独立的组成部分及其排列方式的完整结构之前，让我们先看看数据仓库的一些基础特点。

数据仓库的创始者 Bill Inmon 认为：“数据仓库是为支持管理决策建立的，面向主题的，完整的，长久的，随时间变化的数据集合。”

另一位数据仓库的著名专家 Sean Kelly 用这样的方法定义数据仓库，他认为数据仓库中的数据是

独立的

可利用的

完整的

包含时间变量的

面向主题的

长久的

容易取得的

定义的特点

让我们基于上面的这些定义，解释数据仓库的一些关键特点。数据仓库中的数据的性质是什么？这些数据与其它操作型系统的数据有什么不同？为什么必须是不同的？数据仓库中的数据内容是如何应用的？

面向主题的数据

在操作型系统中，我们使用独立的应用程序来存储数据。例如在一个订单处理应用程序里，我们为这个程序而保存数据。我们需要提供的数据包括输入订单的功能，检查存货，检验客户的信用，以及签署出货单。但是这些数据集合只是包含了与这种特定应用相关的功能数据。我们将会有一组数据集合，分别包含了独立的订单，客户，存货状态和详细交易信息等，但是所有这些都是围绕着订单处理来组织的。

同样，对于一个银行系统来说，关于客户贷款应用程序的数据集合包含了这一特定应用的数据。其它应用的数据，例如检查账目和储蓄的数据，都是与那些特别的应用相关的。同样，对于一家保险公司，不同的数据集合支持独立的应用，例如汽车保险，生命保险以及工人的赔偿保险等。

在每一个行业中，数据集合都是围绕独立的应用进行组织，来支持这些特定的操作型系统。这些数据集合不得不为每个应用提供数据，才能使得这些应用程序能够有效的运行。因而，每个应用的数据集合需要为特定的应用专门组织。

形成鲜明对比地是，在数据仓库中，数据是为主题而不是为应用而存储的。如果数据是根据商业主题进行存储，那么什么是商业主题？商业主题会随着企业的不同有所不同。对于一家制造企业来说，销售、发货和存货都是非常重要的商业主题。对于一家零售商来说，在付款处柜台的销售就是一个非常重要的主题。

从图 2-1 中，我们可以看到数据在操作型系统中的存储与在数据仓库中的存储有什么不同。在操作型系统中，每一个应用的数据根据应用的不同单独组织：订单处理、客户贷款、顾客账单、可接收账款，索赔处理以及储蓄账目等。例如，索赔对于一家保险公司来说就是非常重要的主题。关于汽车保险政策的索赔在自动保险应用中处理。汽车保险的索赔数据在这个应用中组织。同样，工人赔偿保险的索赔数据也在工人赔偿保险应用中组织。但是，在保险公司的数据仓库中，索赔数据按照索赔的主题进行组织，而不是根据像汽车保险或是工

人赔偿保险这样的单独应用来进行组织。

在数据仓库中，数据不是根据操作型应用而是根据商业主题来存储。

操作型应用　　数据仓库主题

订单处理，客户贷款、顾客账单、　　销售，产品，客户，账户，

可接收账款，索赔处理、储蓄账目　　索赔，政策

图 2-1　数据仓库是面向主题的

完整的数据

对于正确的决策，你需要将所有不同应用的相关数据组合在一起。数据仓库中的数据是从多个操作型系统中得到的。源数据基于不同的数据库，文件和数据部分。有很多独立的应用，所以操作型的平台和操作型系统都是不同的。文件版面，特征值编码，区域命名习惯等等都是不同的。

对于很多企业来说，除了从操作型系统中抽取的内部数据，外部数据也同样是非常重要的。一些外部公司可以提供某个专题的非常专业的数据，例如 Metro Mail，或者 A.C.Nielsen，以及 IRI 等等。你的数据仓库需要这样的外部数据。对于一个数据仓库来说，这又是一个重要的变量。

图 2-2 说明了一个关于银行部门整合数据的简单例子。存入数据仓库主题中的数据来自于三个不同的操作型应用。在这三个应用中，就有很多的不同。命名习惯就很不相同，数据值的属性也不相同。在储蓄账户应用中，账号是八位的，而在支票账户中就只有六位。

在将不同来源的数据存储到数据仓库中之前，你不得不先解决这些矛盾。你不得不将这些数据标准化，搞清楚每一个来源中数据名称的含义。在将这些数据放入数据仓库之前，你需要进行转换、合并和整合源数据的处理工作。

解决数据中存在的矛盾；整合来自不同操作型应用中的数据。

储蓄账户　　数据仓库主题

支票账户　　主题=账户

贷款账户

图 2-2 数据仓库是完整的

这里有一些需要标准化的项目：

- 命名规则
- 编码
- 数据属性
- 度量单位

有时间特性的数据

对于一个操作型系统来说，存储的数据包含了当前的值。在一个账单接收系统中，客户账户的平衡是当前未偿还的贷款的平衡。在订单系统中，一个订单的状态是当前这个订单的状态。在客户贷款系统中，客户拥有的结余帐目是当前账户的数额。当然，我们在操作型系统中存储一些过去的交易数据，但是，因为这些系统是支持每天操作工作的系统所以操作型系统反映的是当前的信息。

从另一方面来看，数据仓库中的数据是供分析和决策所用的。如果一个系统使用者希望看到某个客户的消费模式，他不仅需要当前交易的数据，而且还需要过去的交易数据。当系统使用者想要知道东部地区销售额下降的原因，他需要这一段时间的所有交易数据。一个食品杂货连锁店的分析人员想要同时提高两种或更多的产品销量，他必须要了解在过去几个季度里这几样产品的销售情况。

一个数据仓库，除了包括当前数据之外，还必须包括很多历史数据。数据仓库中的每一个数据结构都包含了时间要素。在数据仓库中，你将会发现过去的操作数据被存储为历史数据。数据仓库的这一点特性对于设计和应用都非常重要。

举一个例子，数据仓库中包含了销售的数据，它们被存储在与一个特定的时间要素相关的每一个文件记录或者数据表中。根据数据仓库的细节层次的不同，记录中的销售数量可能与一个特定日期有关，某个星期、月份或者季度。

数据仓库中的数据是和时间变化有关的数据：

- 可以对过去的数据进行分析
- 与当前的信息相关
- 可以对未来进行预测

数据的不变性

从操作型系统中提取的数据和从外部数据源中取得的数据，在数据仓库中被转换、整合并且存储。数据仓库中的数据不是用来进行每天的商业交易的。当你想要处理一个客户的下一张订单，你不能从数据仓库中得到当前存货的状态。操作型订单处理程序可以帮助你。而在数据仓库中，你只保存过去的存货状态信息。你不需要根据你的每一笔订单来实时地更新数据仓库。

操作型系统的数据每隔一段时间被存储到数据仓库中。根据商业交易的需要，这种行为一般来说一天两次，一天一次，一个星期一次，或者两个星期一次都是可以的。事实上，在一个典型数据仓库中，不同类型数据的转移通常的发生频率都是不同的。例如产品属性的变化通常每个星期更新一次。地理上的变化通常一个月更新一次。销售的数据每天更新一次。一般根据使用者的要求来决定这种数据转移或数据更新的频率。

如图 2-3 所示，每一个商业交易行为并不是直接在数据仓库中更新，而是在操作型系统中进行实时的更新。我们在每次交易发生的时候，从操作型系统中增加、改变或者删除数据，而并不是直接在数据仓库中进行更新。你不能在数据仓库中实时的删除数据。一旦数据存入了数据仓库，你就不能对这个数据进行修改。数据仓库中的数据不像操作型系统中的数据那样，可以随时修改。数据仓库中的数据是用来查询和分析的。

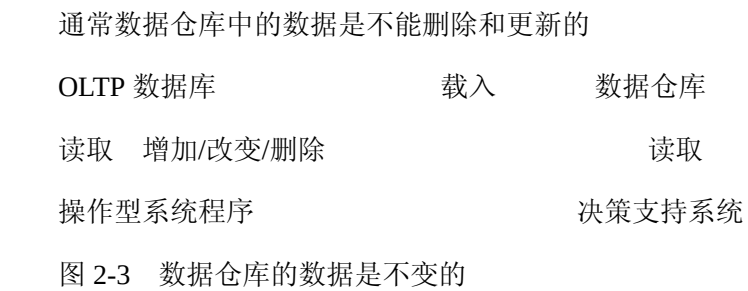


图 2-3　数据仓库的数据是不变的

数据粒度

在一个操作型系统中，数据存储通常非常的详细。例如一个食品店的销售系统中，销售数据记录了每一个结账柜台的每一笔交易。在一个订单处理系统中，每一个产品和每一笔订单都被详细记录。一旦你需要汇总数据，你需要将这些单独的交易数据加起来。如果你需要

这个月的某项产品订单情况，你要将这个月的所有关于这项产品的订单都调出来，然后相加。你不能在操作型系统中保留这一加和数据。

当用户需要查询数据仓库来进行分析工作的时候，他或者她通常开始看总和数据。然后可能需要看某个地区分类的数据。下一步通常是检查每个商店的销售情况。一般说来，用户是从一个高的层次向低层次的细节过渡。

因而，在数据仓库中，你会发现可以很有效地进行不同层次的数据求和。根据查询的需要，你能够得到不同程度的细节情况。数据仓库中的数据粒度就是指这种细节的程度。低层次的细节程度，数据粒度越细。当然，如果你想得到更加细节的数据，必须要在数据仓库中存储大量的数据。所以，必须根据数据类型和希望达到的系统查询性能的要求，决定数据粒度级别。图 2-4 给出了一个典型数据仓库的数据粒度的例子。

银行数据仓库系统的三个数据层次

每日数据	月汇总	季度汇总
账户	账户	账户
交易日期	月份	月份
数额	交易数	交易数
存/取款	取款	取款
	存款	存款
	期初结余	期初结余
	期末结余	期末结余

数据粒度是数据的细节程度。根据需求的不同，需要不同层次的数据细节。数据仓库一般含有至少两层的数据粒度。

图 2-4　数据粒度

数据仓库和数据集市

如果你读过最近几年的关于数据仓库的著作，那么你一定会遇到过这两个词“数据仓库”和“数据集市”。很多人对于这些词汇感到困惑。一些作者和产品提供商也会将这两个词混淆使用。在这里，我们来看看这两个词的区别。

Bill Inmon 于 1998 年在一个著名杂志上说：“今年 IT 经理们面对的最重要的问题就是到底是先建立数据仓库还是先建立数据集市。”这一论述就是今天也非常的正确。

在开始决定建立数据仓库之前，你需要考虑以下这些问题：

- 采取自上而下还是自下而上的方法？
- 企业范围还是部门范围？
- 先建立数据仓库还是数据集市？
- 建立向导还是直接实施
- 是否依赖数据集市？

这些都需要认真的考虑和计划。

你是选择先看看组织的整体情况，采取自上而下的方法，然后建立一个巨大的数据仓库？还是采用自下而上的方法，根据具体部门的要求，建立一个部门级的数据集市？

你可以建立一个大型的数据仓库，然后将知识库数据存储在部门的数据集市中。或者，你可以建立独立的局部数据集市，然后将它们组合起来建立你的数据仓库。这些数据集市和其它的相互独立吗？或者，通过整个数据仓库来提供数据的？你是否需要建立一个向导型数据集市？这些都是重要的问题。

它们有什么不同？

让我们来看图 2-5。这里有两个不同的基本方法:（1）数据仓库依靠数据集市提供数据；（2）几个部门的数据集市组成一个数据仓库。对于第一种方法，你需要从操作型系统中提取数据，然后转换、清洗、整合并将数据放入数据仓库中。那么，在你的项目中适合哪一种方法呢？是自上而下呢？还是自下而上？让我们仔细来看看这两种方法。

完整的/从企业整体考虑的	部门的
所有数据集市的集合	一个单独的商业处理过程
从阶段区域得到的数据	星型结构（事实表和维度表）
通过展示的方式进行查询	适合数据连接和分析的技术
基于公司数据视角的结构	基于适合部门数据视角的结构
通过实体-关系模式进行组织	

数据仓库　　数据集市

图 2-5　数据仓库和数据集市

自上而下和自下而上的方法

自上而下这种方法的优点是：

- 可以从整个企业的角度来看数据
- 体系结构完整——不是由不同的数据集市组成的
- 对数据内容的唯一、集中的存储
- 中央控制和集中的规则
- 对反复的查询能够做出快速的反应

这种方法的缺点是：

- 需要花更多的时间来建造
- 失败的风险很高
- 需要高水平的综合技能
- 费用很高

这是你建设一个大型企业级数据仓库使用的大致方法。在这里没有信息孤岛，数据仓库是大型而完整的。但是，这种方法将会花费更多的时间，并且承担失败的高风险。如果你的团队没有有经验的专业知识，将会是危险的。而且，向高级主管和领导推销这种方法也是困难的。他们可能不能很快地看到结果。

而另一种方法，自下而上的方法的优点是：

- 实施快速而方便
- 良好的投资回报
- 失败的风险较小
- 渐进的，可以先建立重要的数据集市
- 项目团队可以从中学习和成长

而这种方法的缺点是：

- 每一个数据集市对数据的视角都比较窄
- 每个数据集市都有多余数据

- 总是有矛盾和不一致的数据
- 增加无法管理的接口

在自下而上的方法中，你一个又一个的建立部门数据集市。你需要建立一个计划来决定究竟先建立哪个数据集市。这种方法最大的问题在于数据的不完整性。每一个独立的数据集市都不了解整个组织的需求。

一个实用方法

为了设计一个适合的方法，你需要检查你的组织到底需要什么。你的组织是需要长期结果，还是关于几个主题的快速数据集市？或者，你想要找一些其它的实用方法？

虽然无论是自上而下还是自下而上的方法，都有其优点和缺点，但是一个妥协的方法看上去是可行的。这种实用方法的主要推崇者是 Ralph Kimball，一个著名的数据仓库专家。这种实用方法的步骤是这样的：

1. 从整个公司的角度来计划和定义需求
2. 为完整的仓库创造一个体系结构
3. 使数据内容一致而且标准化
4. 将数据仓库作为一组数据集市来实施，每次一个

在这种实用方法中，你需要考虑什么是你的公司在很长一段时间内最想要得到的。这种方法的关键是你首先要站在整个企业的高度来进行计划。你为整个仓库系统建立结构体系。然后，决定每一个数据集市的数据内容。这种集市是经过认真设计的数据集市。你执行这些数据集市，每次一个。在实施之前，确定这些不同的数据集市中的数据内容有统一的数据类型，字段长度，精度和语义。在每个数据集市中，一个确定的数据元素必须表达一个相同的意思。这样就会避免几个数据集市之间的数据不统一。

在这种实用方法中，一个数据集市就是整个数据仓库系统的一个逻辑子集。因而，数据仓库就是所有数据集市的集合。企业中单独的数据集市是以一个特定的商业行为为目的，但是所有这些数据集市的集合就组成了企业数据仓库。

在这里，当我们提到数据仓库和数据集市的时候，我们使用这种实用方法中的含义。对于我们来说，数据仓库就是一组数据集市的集合。

数据仓库的组成部分

我们现在了解了数据集市和数据仓库的基本定义和特点。现在我们来看看它的组成部分。

当我们建立一个操作型系统，例如订单系统，索赔处理或者储蓄账户，我们将几个组成部分组合起来构成了这个系统。其中接口部分有 GUI（图形用户接口）是用来连接用户和数据输入的接口。数据存储部分包括数据库管理系统，例如 Oracle，Informix，或者 Microsoft SQL Server。展示部分是一组用户屏幕和报表。数据接口和网络软件组成了连通部分。根据信息需求和组织的结构，我们用最优的方法将这些组成部分设计好。

体系结构是这些组成部分的合理安排。使用软件和硬件的组成部分建立一个数据仓库时，为了适合组织的需要，用一个特定的方法将这些部分安排好来获得最大的效益。

图 2-6 显示了一个典型数据仓库的基本组成部分。左边是源数据部分，下一个是数据准备部分。中间的部分是数据存储部分，管理数据仓库的全部数据。这个部分不仅存储和管理数据，而且还保存了元数据的信息。而图的右边是信息传递部分，我们看到有很多的方法来使人们看到数据仓库提供的信息。

无论你是否为世界财富 500 强的大型制造公司建立数据仓库系统，还是大型全国连锁食品店，抑或是全球银行部门，数据仓库的基本组成部分都是相同的。每一个数据仓库都是由相同的组成部分构成的。所不同的是在不同的数据仓库中对不同组成部分的侧重点不同。

接下来，让我们来仔细了解这些部分。我们想知道这些组成部分都是什么，它们是怎么组合在一起的。我们将要了解到每一个特定部分的特点和用途。

源数据部分

数据仓库的源数据一般来自于四个主要的类别。

生产数据

这个种类的数据来源于企业的各种操作型系统。基于数据仓库的信息要求，你要从不同的操作型系统中选择数据部分。在处理这些数据的时候，你会遇到很多不同的数据格式，会发现数据被存储在很多不同的硬件平台上。此外，支持这些数据的是不同的数据库系统和操作系统。这就是大量来自垂直应用程序的数据。

在操作型系统中，信息查询的范围很窄。你在一个操作型系统中根据特定的商业主题来查询信息。你可能想要某个单独顾客的姓名和地址。或者是，你可能需要了解某个星期的某个顾客的所有订单。你还可能需要看一个发票以及这张发票上记载的信息。在操作型系统中，你不能进行宽范围的查询。你不能在操作型系统中进行没有预先安排的查询。所有的查询都是可以预测的。也就是说，你不能期望同时在不同的操作型系统中运行一个特定的查询。所有这些都意味着什么？简单的说，企业中的不同操作型系统中的数据没有一致性。在不同的系统中，词组的含义，例如一个账户信息，都是不同的。

生产数据的重要性和特性是不同的。你最大的挑战是将这些从不同生产系统得到的数据进行标准化，转换数据，并且将它们转换成数据仓库可以存储的有用数据。

内部数据

每一个组织中的用户都有自己的电子表格、文档、客户信息，有的时候甚至有部门数据库。这就是内部数据，也是数据仓库中的有用部分。

在企业中，单独用户的资料越来越受到重视。当你的销售代理想同这些顾客谈谈或者当你的市场部门想要对这个单独顾客进行特别销售的时候，这些细节信息就非常重要。虽然这些数据很多是从生产系统中得到的，但是大部分都是存储在客户资料中。

你不能忽略这些存在于私人资料中的内部数据。IT 部门必须与用户部门一起收集这些内部数据。

内部数据给数据转换和整合的过程增加了很多的复杂性。你不得不事先决定战略，然后才能从这些电子表格中收集数据，想办法从文档资料中将数据取出来，从部门数据库中收集有关数据。也就是说，你可能需要事先计划好对内部数据的获取。

存档数据

操作型系统是用来运行当前商业交易的。在每一个操作型系统中，你都要阶段性的将旧数据存储到存档文件中。根据实际情况的不同，决定多长时间存储一次，以及究竟存储哪些数据到存档中。一些数据一年以后才存档，有些时候数据在操作型系统中保存 5 年之久。

目前存在许多不同的存档方法。数据存档有几个层次：在第一个层次中，新的数据可能通过在线的方式存档到一个独立的存档数据库中。在第二个层次，旧一点的数据被存入平面文件或者磁盘存储中。在下一个层次，最旧的数据存入磁带机甚至丢弃。

我们刚才学过，数据仓库存储了历史数据的简短描述。而我们需要使用历史数据来进行分析工作。为了得到这些历史信息，我们需要检查我们的存档文件。根据数据仓库的需求不同，要得到足够多的历史数据。这种数据对于识别模式和分析趋势非常有用。

外部数据

外部数据的使用占大部分的经理们所使用的信息中的绝大部分。例如，一家汽车租赁公司的数据仓库包含了领先汽车制造商的当前生产计划数据。这种数据仓库中的外部数据可以帮助该公司随时检查自己的经营状况。

通常这种类型的数据无法从企业内部直接获得。企业内部的生产数据和存档数据是有一定局限的。它们可以告诉你企业过去的生产和经营情况。为了了解经营发展趋势和与其它公司进行比较，你需要从外部数据源得到数据。

通常从外部得到的信息与内部数据的格式是不同的。你需要对数据进行处理，使它们在数据格式和类型上与内部信息保持一致。

数据准备部分

在从不同的操作型系统和外部数据源得到数据之后，你需要为数据仓库的存储做准备。这些从不同数据源得到的数据需要进行清洗和转换，才能存储为适合查询和分析的格式。

为了准备这些数据，有三个主要的工作程序。你要抽取这些数据，转换它们，然后将这些数据载入数据仓库存储。这三个主要的工作程序就是抽取，转换和载入。数据准备部分就是由这三个工作程序组成的。数据准备提供了对这些数据进行清洗、转换、组合、复制和准备工作的空间，使它们可以供数据仓库存储和使用。

对于数据仓库的数据准备来说，一个独立的数据准备阶段是非常必需的。为什么呢？因为在数据仓库中需要从大量的数据源得到数据，记住数据仓库中的数据是面向主题的，而且交叉了很多的操作型应用程序。

现在我们已经了解了一个独立数据准备部分的重要性，让我们来看看数据准备部分到底做了哪些工作。我们将主要讨论数据准备部分的三个主要的工作程序。

数据抽取

这一工作程序是针对多个数据源的。你要对每一个数据源使用合适的技术。源数据可能

使用不同的数据格式，它们其中的一部分可能存于关系数据库系统，一些数据可能基于其它的历史网络系统和分级数据模型。很多数据源可能仍然是文本文件的格式。数据抽取的工作变得非常复杂。

市场上有很多的关于数据抽取的工具。你可以根据特定的数据源使用合适的外部工具。对于其它的数据源，你可以自己开发程序来进行数据的抽取工作。购买外部工具可能会导致高成本。

在抽取数据之后，你在哪里进行进一步的数据准备工作？你可以在自己的平台上进行数据抽取，只要这种方法适合你的结构框架。更常见的是，数据仓库执行人员将数据源抽取到一个独立的物理环境，这样可以更容易的将数据转移到数据仓库中。在这个独立的环境中，你可以从许多文本文件、关系数据库或者两者的结合中抽取数据。

数据转换

在每个系统中，数据转化是非常重要的工作程序。例如，当你执行一个例如杂志订阅应用程序的操作型系统，你必须从很多记录数据中得到建立数据库的运行。你可能需要将一个手工系统转化成你需要的系统。或者，你要将一个面向文件的系统转化成一个关系数据库表支持的现代系统。在上面这些例子中，你要将这些早期的数据进行转换。所以，与数据仓库有什么不同呢？

也就是说，如同你知道的那样，数据仓库的数据来自许多不同的数据源。如果说数据仓库的数据抽取遇到了很大的问题，数据转换将会遇到更大的挑战。数据仓库的另一个要素是数据载入并不是最初的载入。你将会发现源数据系统一直是在变化中的。你建立的任何一个转换任务都要适应这种一直进行着的变化。

在数据转换过程中，有几个单独的步骤。首先，要对每一个不同来源的数据进行清洗。清洗的过程中可以检查错误的拼写，检查多个数据源之间编码或者压缩格式的矛盾，或者补充数据的错误值，也可以排除从多个数据源系统中取同一个数值时出现的重复问题。

对数据元素的标准化也是数据转换过程的一个很重要的组成部分。要对数据类型进行标准化，并且对不同数据源的相同数值的长度进行补充。语义的标准化也是一个重要的任务。你要解决同义和同音异义的问题。当相同的字段名在不同的数据源系统中代表不同的意义的时候，你就必须要解决这个同音异义的问题。

数据转换过程解决了从不同数据源提取数据的解决方法。你要组合一个源记录中提取的

数据，或者对很多源记录中提取的数据进行组合。另一方面，数据转换还包括了清洗没有用的源数据，并将它们进行新的组合。在数据准备阶段，对数据的分类和聚类是很重要的部分。

在实际中，操作型系统选择的键值是有生产意义的值。例如，生产键值可能是一组生产单元、存货仓库编码和生产批次信息的组合。数据仓库中原来的键值可能就没有这些实际意义。我们将在第十章中详细解释这个问题。

一家食品连锁店的操作型系统存储了每一个结账柜台发生的每一笔交易信息。但是在数据仓库中，就没有必要将数据存储到这种细节程度。你可以在数据仓库中存储你需要了解的汇总信息。在这样的例子中，数据转换的工作就要包含适当的综合。

当数据转换工作程序结束后，我们已经得到了经过清洁、标准化和汇总后的完整数据了。你现在可以将数据载入到数据仓库中。

数据载入

数据载入的工作程序由两个不同的任务组成。当你结束了设计数据仓库结构的工作后，就必须要将数据载入到数据仓库中。最初的载入工作是要使用大量的时间将大量的数据载入。当这个数据仓库开始工作了，你需要继续提取源数据的变化，将这些数据变化按照数据仓库的要求进行转换后，存入正在工作的数据仓库中。图 2-7 中描述了这个过程。

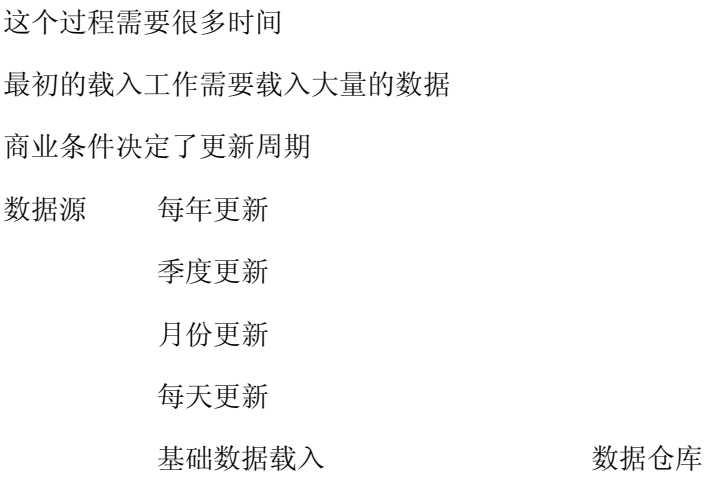


图 2-7　向数据仓库载入数据

数据存储部分

数据仓库的数据存储是一个独立的部分。企业的操作型系统支持每天的操作。这里有很多在线的交易处理应用程序。操作型系统的数据存储通常只包含当前的数据。而且，这些数据使用适合快速和高效处理的数据格式进行储存。而在数据仓库中，你需要存储分析用的大量历史数据。而且必须使这些数据的结构和格式适合分析工作，而不是适合快速检索的独立操作。因而，数据仓库的数据存储与操作型系统的数据存储是不同的。

在操作型系统的数据库中，当交易发生时就进行数据的更新。交易何时和如何改变数据库中的数据，并不完全在人的控制中。操作型系统中的数据可能随时改变。当分析人员使用数据仓库中的数据进行分析时，他们需要稳定的数据，并且代表了过去某个特定时间段的特征。在这一过程中，数据存储不能进行连续的更新。正是因为这个原因，数据仓库是“只读”的数据存储库。

通常来说，数据仓库中的数据库必须是开放的。根据需要的不同，你可以使用很多外部工具。数据仓库必须对外部工具是开放的。很多数据仓库都有关系数据库管理系统。

很多数据仓库都有多维的数据库管理系统。可以使用很多方法对数据仓库中抽取的数据进行加和，加和后的数据存储在多维数据库里。这种多维数据库系统通常是外部公司提供的产品。

信息传递部分

需要数据仓库中数据的究竟是什么样的人？这个范围是非常广泛的。没有经过培训、刚刚接触数据仓库的新手需要预制的报表和查询。有些用户偶尔需要使用一下，而不是经常。这种类型的使用者也需要预先定制好的信息。商业分析专业人员使用数据仓库中的信息进行复杂的分析。高级用户需要操纵整个数据仓库，得到感兴趣的数据，定制自己的查询，钻取数据层，并且创造用户报表和特别查询。

为了向广泛的数据仓库使用者提供信息，信息传递部分就包含了多种信息传递的方式。图 2-8 表现了这些不同的信息传递方式。为新手和偶尔使用者定制了报表和特别查询。向商业分析专业人员和高级用户提供了复杂查询、多维分析和统计分析等功能。将数据导入经理信息系统（EIS），供高级经理和管理者使用。一些数据仓库还提供数据挖掘工作需要的数据。数据挖掘应用是知识发现系统，这种挖掘机制可以帮助你发现商业趋势和数据的模式。

在数据仓库中，可以包括几个信息传递部分。一般来说，可以提供在线查询和报表。用户在线输入请求，在线得到结果。你可以通过 E-mail 建立发送定期的报表传递方法，或者可以使用局域网的杂志或网页发布报表。最近，通过 Internet 发布信息也流行起来。

元数据部分

数据仓库的元数据与数据库管理系统中的数据字典的概念比较相似。在数据字典中，保存了逻辑数据结构信息、文件和地址的信息，索引的信息等等。数据字典包含了数据库中数据的数据。

同样的，元数据部分是数据仓库数据的数据。这个定义是一个广泛使用的定义。详细的说就是，数据仓库中的元数据与数据字典很相似，但是又比数据字典更进一步。在本章我们将使用一个单独的部分来讨论元数据。这里，为了完整起见，我们将元数据作为数据仓库体系结构的一个单独部分列出。

管理和控制部分

这个部分在数据仓库中是位于其它组成部分之上的。管理和控制部分对数据仓库中的服务和动作起协调作用。这个部分控制了数据转换和数据载入的工作。另一方面，它也协调信息向用户的传递。它与数据库管理系统一起工作，使数据能够准确的存储。它控制数据准备部分的数据和数据仓库数据存储。

管理和控制部分与元数据部分一起工作，执行管理和控制的功能。因为元数据部分包含了数据仓库自身的数据，元数据也就成了管理模块的数据来源。

数据仓库中的元数据

将元数据看作是当地的电话黄页。你需要当地的商店的信息吗？它们在哪里，它们的名字是什么，它们的产品是什么？去查电话黄页吧。电话黄页是当地部门信息的一本字典。元数据部分与电话黄页的作用差不多，可以作为数据仓库内容的一本字典。

正是因为元数据在数据仓库中的重要性，我们将在第九章中详细地对它进行介绍。在本章中，我们先对元数据进行简单的介绍，它是数据仓库中重要的结构体系。

元数据的类型

数据仓库中的元数据主要分为三类：

- 操作型元数据
- 抽取和转换元数据
- 最终用户元数据

操作型元数据。我们知道，数据仓库中的数据来自于企业中多个操作型系统。这些数据源系统包含了很多不同的数据结构，字段长度和数据类型也有不同。在这些数据中，你要对不同来源的文件进行合并，解决编码和字段长度不同的问题。当你将这些信息传递给最终用户的时候，你必须能将这些数据与最初的数据源联系起来。操作型元数据包含了所有操作型数据源的信息。

抽取和转换元数据。抽取和转换元数据包含了源数据系统的信息，例如，抽取频率、抽取方法和数据抽取的商业规则。而且，这一类的元数据包含了数据准备阶段数据转换的所有信息。

最终用户元数据。最终用户元数据是数据仓库的向导。它使最终用户可以从数据仓库中找到自己需要的信息。最终用户元数据允许最终用户使用自己商业终端和自己的习惯来查询数据。

特别指出的意义

为什么元数据对于一个数据仓库是如此重要呢？

- 首先，它连接了数据仓库的所有部分。
- 其次，它为开发者提供了数据仓库内容和结构的所有信息。
- 最后，它向最终用户开放，使他们能够得到需要的信息。

本章小结

- 数据仓库的特点是：独立的，面向主题的，完整的，包含时间变量的，永久的。
- 你可以使用自上而下的方法来建造一个大型、全面的企业数据仓库；或者，你可以使用自下而上的方法来建立一个小型、独立的部门数据集市。每一种方法都有自己的优点和缺点。

- 有一种可行的方法可以用来建造数据集市，然后组成企业完整的数据仓库。
- 数据仓库的组成部分包括：数据源、数据准备、数据存储、信息传递、元数据和管理控制部分。
- 在数据仓库中，元数据非常重要，因为它负责连接数据仓库的所有部分并为最终用户提供向导。

思考题

1. 举出至少六种数据仓库的特点。
2. 为什么数据仓库比操作型系统更需要完整的数据？
3. 数据仓库中的每个数据结构都有时间元素。为什么？
4. 解释数据粒度，以及数据仓库是如何应用的。
5. 自上而下和自下而上的数据仓库建立方法有什么不同？讨论每一种方法的优点和缺点。
6. 数据仓库的数据来源有哪些？
7. 为什么需要一个独立的数据准备部分？
8. 在数据转换阶段，举出五种不同的工作程序。
9. 举出六种不同的信息传递的方法。
10. 数据仓库中的三种主要元数据是什么？简单介绍每一种的功能。

复习题

1. 连线：

1.永久性数据	A．用户向导
2.双重数据粒度	B．面向主题
3.数据集市	C．知识发现
4.异类数据	D．个人的电子表格
5.决策支持	E．应用方法
6.数据准备	F．由于多种数据源

7.数据挖掘	G．细节和汇总
8.元数据	H．只读
9.操作型系统	I．数据整合的工作空间
10.内部数据	J．数据仓库中的数据

2. 数据仓库是面向主题的。下列公司的主要商业主题是什么？
3. 一家国际制造业公司
4. 本地社区银行
5. 国内连锁酒店

6. 你是一个为一家保险公司建立数据仓库项目组的数据分析人员。列出你可能用到的数据源。解释你的观点。
7. 对于一家航空公司来说，将三个操作型应用程序的数据导入数据仓库。数据导入和更新的周期是怎样的？
8. 准备一张表格，列出一家大型国内食品连锁店的数据仓库中，所有的潜在使用者和信息传递方法。

第三章 数据仓库的发展趋势

本章目标

- 数据仓库的发展历史
- 了解数据仓库是如何成为主流的
- 讨论几种主要的发展趋势
- 掌握标准的需要
- 理解实现 Web 技术的数据仓库

在前面的内容中，我们学习了为什么数据仓库成为所有行业中企业所必需的。我们也学习了数据仓库是如何给企业带了大量利润的，以及讨论了数据仓库的主要组成部分。你现在对于数据仓库的概念、特点和主要功能都应该有了一个比较全面的认识。它是一个基本的概念，同时也是许多技术的融合。几家大型公司和技术推进者们在几年前将数据仓库的概念推到人们的面前。

在我们学习下面的内容之前，我们需要先问几个问题。当前市场的阶段是什么？商业应用采用了数据仓库的什么方面？技术的发展是怎样的？也就是说，目前的发展趋势是怎样的？

现在讨论趋势的问题是否有点太早？对发展趋势问题的讨论通常在结束的时候，作为留作思考的内容。这一章的内容并不是对未来可能出现情况的预测，我们是想要讨论现在正在发生的重要趋势。

与当前知识的潮流趋势保持一致，对你来说非常重要，这将是你对数据仓库知识继续学习的知识背景。当你收集数据仓库需求的时候，你需要意识到当前的趋势。当你深入到设计的阶段时，需要了解当前的趋势。进入数据仓库实施阶段的时候，需要保证你的数据仓库是与潮流方向保持一致的。关于潮流的知识非常重要，甚至在你学习的初期就必须要了解。

在本章中，我们将一起了解几种主要的潮流趋势。你要了解数据仓库怎样保持成长并变得越来越普及，以及为什么会这样。我们要讨论提供商的解决方案和产品的趋势。我们要将数据仓库和其它的技术现象联系起来，例如 Internet 等。我们还要在以后的章节中重新看到我们本章学习的某些趋势的内容。

数据仓库的持续成长

数据仓库不再是科研和实验用的纯理论的概念，它已经成为了主流。虽然数据仓库还没有在每一个牙医的办公室里出现，但是它也不再局限于高端用户了。超过一半的美国公司已经计划建立数据仓库。大约 90%的跨国公司已经有了数据仓库，或者计划在 12 个月内建立数据仓库。

从连锁零售商店到金融机构，从制造类企业到政府部门，从航空公司到公用事业，数据仓库正在革命性的改造人们进行商业分析和战略决策的方式。每一个拥有数据仓库的公司都通过得到积极的结果而获得大量的利润。这些公司中的大部分现在正在将数据仓库与基于网络的技术进行融合，以便更加方便快捷地得到关键信息的传递。

在过去的五年中，大量的供应商涌入了这个市场。数据仓库的每个阶段都出现了大量的解决方案和产品：数据建模、数据获得、数据质量、数据分析、元数据等等。由数据仓库协会发行的使用者指南列出了不止 105 个主要产品。这个市场已经非常庞大，而且正在持续增长。

数据仓库正在成为主流

有四个主要的因素促使大量的公司使用数据仓库：

- 激烈的竞争环境
- 政府的减少管制
- 需要再造内部流程
- 急需为用户定制的市场销售

电信业、金融银行业和零售业是最早采用数据仓库的行业，那是因为政府对电信业和银行业的减少管制。零售业采用数据仓库是因为激烈的市场竞争环境。公用事业公司进入这个市场也是因为管制的减少。下一批采用数据仓库的行业应该是金融服务、医疗健康、保险行业、制造业、制药业、运输业和销售业。

今天，电信和银行业继续在数据仓库使用上保持领先地位。这两个行业中的大约 15%的预算都是用在了数据仓库的建设上。这些行业的公司收集了大量的交易数据，数据仓库可以将这些数据转化成为有助于决策的战略信息。

目前，数据仓库存在于可想象得出的所有行业。图 3-1 中，按照数据仓库专业人员的平

均收入，列出了主要的行业名称。公用事业行业排在第一位。

年平均收入（单位：1000 美元）

公用事业 92　　媒体/出版业 89　　航空业 88　咨询业 83　高科技 83　金融服务 82　制药业 81　半导体业 79　商业服务 78　制造业 74　电信业 75　保险业 74　运输业 69　政府部门 66　其它 65　银行业 65　法律业 61　教育 57　石油化工 54

资料来源：数据仓库协会 1999 年进行的数据仓库薪金调查

图 3-1 使用数据仓库的行业

在数据仓库发展初期，大部分的使用者是一些跨国公司。建造数据仓库非常昂贵，它的工具也不多。只有一些大公司有能力支付。现在我们开始看到中等规模的数据仓库和小一点数据仓库的出现，这些公司现在能够支付建造数据仓库或购买数据集市的费用。看一看过去使用的数据库管理系统，你会发现数据库的提供商也开始使用增加了数据仓库特征的数据库管理系统，来帮助你建立数据仓库。完整的解决方案也越来越便宜，操作系统也能够承担数据仓库的功能了。

数据仓库的扩张

虽然早期的数据仓库主要集中于为高层次分析人员提供汇总数据，但是现在我们已经看到了各种不同行业建造的越来越大的数据仓库。现在的公司已经有能力获得、清洗、存储和使用这些由商业交易行为产生的大量数据。存储在数据仓库中的数据量保持持续增长，达到千兆位的数量级。存储了几千兆数据的数据仓库，在零售业和电信业已经不再少见。

我们以电信行业为例。一家电信公司一年创造了几亿个呼叫行为交易。为了发展合适的产品和服务，公司需要分析这些呼叫行为的细节，公司的数据仓库不得不用最小的粒度来存储数据。

同样的，考虑一个有几百个网点的连锁零售店。每天，每一个网点都创造了几千个销售行为信息。另一个例子是一家制药行业的公司，要处理数以千计的试验和测量，来得到需要的产品。这些行业的数据仓库趋向于非常巨大。

最后，让我们来看看一家典型医疗补助欺诈控制单位的潜在数据仓库大小。这个组织负责调查和起诉这个州的医疗补助欺诈行为。还要起诉那些滥用疗养院的病人，以及控制由医

生、药剂师和其它医护人员进行的欺诈行为。这类簇织通常有几个地区级的办公室。在一个地区发现的欺诈案件必须也在其它的地区进行检查。你能想象支持这样一个欺诈控制单位的数据仓库的大小吗？这将是几个 TB（千兆）的数据量。

解决方案和产品

作为一个信息技术的专业人员，你对数据库的解决方案和产品一定非常熟悉。同样，你肯定也对大多数的操作系统及其产品非常熟悉。市场上有多少数据库产品提供商？有多少操作系统产品提供商？数据库和操作系统的数量与数据仓库的产品相比，就很少了。有数以百计的数据仓库供应商，以及数以千计的数据仓库产品和解决方案。

开始的时候，这个市场比较混乱。每一个公司，无论大小，只要有任何与数据仓库相关的产品都投入了这个市场。数据仓库的概念就是每一个厂商自己定义的概念。每一个公司都标榜自己的产品为正确的数据仓库工具。数据仓库对于很多厂商来说都是一个新的概念。

在过去的十年中，情况发生了很大的好转。市场慢慢的成熟起来，也逐渐的稳定起来。图 3-2 中为当前的数据仓库市场的主要情况。

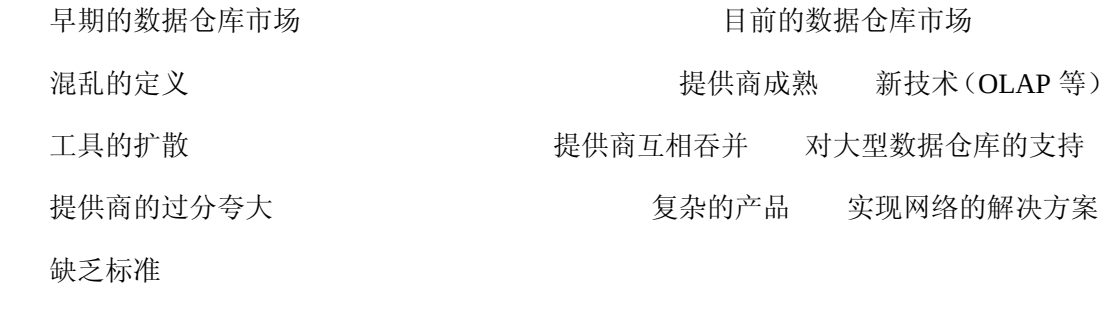


图 3-2　数据仓库市场的当前状况

我们通常在一个成熟的市场中能看到什么？我们希望能找到一个合并的方式，也就是数据仓库市场中到底会发生什么。现在越来越多的有实力的公司涌入了数据仓库的市场。一些主要的厂商通过收购其它公司来扩大自己的产品线。

现在传统的数据库公司也进入了数据仓库的市场。他们围绕原来的数据库产品来提供数据仓库解决方案。一方面，数据库管理系统中打包了数据抽取和转换工具。另一方面，查询和报表工具也增强了数据仓库的功能。一些数据库提供商通过提供更加复杂的工具，例如数据挖掘工具，进一步加强了其功能。

有这么多的提供商和产品，我们如何来对它们进行分类进而了解市场呢？这里我们将市场大致分为两块。第一大类是基于公司数据仓库的数据仓库提供商和产品，里面的数据是完整和已转换的。这个部分也就是战略型数据仓库的市场，大约占整个市场的四分之一。第二个部分比较松散和分散，包括部门数据集市、部分数据库市场系统，以及很大部分的决策支持系统。在每个部分中都有主要的供应商和产品。

我们还可以换一种方式来看这些产品。图 3-3 按照它们在数据仓库中的功能不同，列出了产品列表。

数据仓库产品（括号内为主要的产品的数量）

数据整合和清洗（12）

数据建模

抽取/转换

普通应用

特殊应用

数据转移

信息服务器

关系型数据库

专用索引数据库

多维数据库

决策支持

关系型 OLAP

桌面 OLAP

查询和报表

数据挖掘

应用开发

管理和控制

元数据管理

监控

工作计划

查询管理

系统管理

数据仓库支持的应用模块

金融

销售/市场/客户关系管理

平衡记分卡

专门部门

资料来源：数据仓库协会

图 3-3 按照功能分类的数据仓库产品

重要趋势

一些专家认为直到现在技术仍然是驱动数据仓库发展的动力。这些专家宣称我们已经能够在软件方面看到重要的进展。在接下来的几年中，数据仓库将在软件方面有长足的进步，特别是对于查询优化，对大型表格的缩影，提高 SQL 的性能，加强数据压缩能力，以及扩展多维建模的方面。

我们来看看这些重要的趋势，对每一个进行详细讨论，因为它们每一个都对数据仓库有非常重要的影响。在我们详细了解这些趋势的时候，试着掌握它的重要性并且在公司数据仓库中用到它们。我们先要回答这样一个问题：为什么必须在你的数据仓库中用到这些趋势具有优势？

多种数据类型

当你建设首个数据仓库的时候，你可能只是要包含数值数据。但是很快你就会意识到，只存入有结构的数值数据是不够的。需要同时考虑其它类型的数据。

传统意义上，公司的数据仓库中一般存储的是有结构的数据，大部分是数值数据。从这个观点来看，决策支持系统分为两类：处理结构数据的数据仓库和处理非结构化数据的知识管理系统。这种分类方法比较模糊。例如，大部分的市场是由结构化的数值数据组成的。一个决策者需要对销售量最好的产品进行分析，在分析过程中决策者找出了一个特定的产品类型。他或她现在要深入了解这个产品来进行进一步的分析。如何才能做到这一点呢？于是人们发现在数据仓库中不仅需要结构化的数据，而且也需要非结构化的数据。

什么样的数据我们可以称之为非结构化数据？下图 3-4 中，我们可以看到要在数据仓库

中整合不同类型的数据，来使得决策的制定工作更加有效。

结构化数据　图像　空间数据

结构化文本　　数据仓库知识库　　影像

非结构化文档　　音频

图 3-4 数据仓库：多种数据类型

现在让我们来看包含一些非结构化数据的行业中的进步。你将会了解为什么数据仓库中必须包含这些数据。

增加非结构化数据。一些提供商在他们的产品中包含了非结构化数据，特别是文本和图像，通过将这些多媒体数据作为另一种数据类型的方法来实现。这些数据被作为是关系数据的一部分，作为二进制大对象（BLOB）来保存，最大可以达到 2GB。使用用户定义功能（UDF）来将它们定义为用户定义类型（UDT）。

不是所有的二进制大对象都简单的当作另一种类型的数据类型进行存储。例如，一个视频剪辑需要一个支持多媒体信号流传递和保持音频同步的服务器。为了这个目的，就需要提供专门的服务器。

*查找非结构化数据。*你已经通过增加非结构化数据优化了数据仓库。你还需要做什么？当然，接下来就需要对这些非结构化数据进行查找了。现在提供商们提供了新的查找引擎，来查找非结构化数据。图像内容的查询就是一个例子。你可以利用这些产品来查找图像，可以通过不同的方式，例如大小、颜色和文本等。当不止一个图像符合要求时，这些选中的图像文件就会逐一的展现出来。

对于自由格式的文本数据，检索引擎可以完成通过单字、特征量、短语等的搜索工作。一些引擎还能够替换相应的文字和搜索。例如对一个单词“mouse（老鼠）”的搜索，还可以找到包含“mice（老鼠的复数）”的文档。

直接搜索音频和视频数据也属于搜索的范围。通常，它们被描述为自由格式文本，然后使用文本搜索的方法进行查找。

空间数据。数据仓库的一个重要使用者，可能是市场总监，需要利用数据仓库进行在线的分析。市场总监运行这样一个查询：给我看看 XYZ 商店头两个季度所有商品与去年同期对比的情况。在看完了这些数据后，他或她想到了两个问题：这个商店周围的居民的平均收入是怎样的？这些居民到这个商店的平均距离是多少？要回答这些问题，必须要在数据仓库

中包含了空间数据才行。

增加空间数据会大大提高数据仓库的价值。地址、街区、城市坐标、国家等都是空间数据的类型。目前数据仓库产品中已经包含了空间数据。一些数据库提供商在他们的产品中提供空间数据的扩展，使用 SQL 扩展将空间数据与商业数据结合起来。

数据可视化

当一个用户查询你的数据仓库并仅仅能看到输出列表或电子表格的结果时，你的数据仓库已经过时了。你需要使用图形和表格的形式来将结果展现出来。每一个使用者都希望看到图表类型的结果。结果数据的可视化强化了用户的分析处理过程，特别是当用户在察看长期历史趋势的时候。数据可视化帮助用户快速方便的阅读查询结果。

主要可视化趋势。 在过去的几年中，有三个主要趋势改变着数据可视化软件的方向。

*更多的图表类型。*更多的数据可视化使用一些标准的图表类型。大量的数据被转换成饼图、散点图或者其它类型的图表。现在数据可视化软件能够提供大量的图表类型。

*互动可视化。*可视化不再是静态的。动态的图表成为用户的接口。你的用户可以观看结果图，对它进行操作，然后在线观看新的结果。

复杂结果单元的可视化。用户可以使用饼图等图表形式观看简单的数值结果。但是新的可视化软件可以使大量的结果和复杂的数据结构形象化。

图 3-5 中总结了这些主要的趋势。你可以从中看到技术是如何走向成熟和进化的。

可视化类型。可视化软件现在可以支持大量的图表类型。简单线性图表的时代一去不复返了。用户的需求变化非常快。商业用户需要观看饼图或者直方图。技术用户需要散点图和星座图。观看空间数据的用户需要地图和其它三维的表现形式。经理们和管理者，需要监控企业的发展情况，就像需要数字仪表盘来监控速度、里程和交通信号灯一样。在过去的几年中，这三种主要的趋势影响着数据可视化软件的方向。

高级可视化技术。可视化技术方面的重要发展是从静态表格到动态互动展现的变化。

*可操作的表格。*用户可以旋转图表或者是改变图表的类型，来更清楚地了解结果。通过复杂的可视化类别，例如星座图和散点图等，用户可以用鼠标选择一个数据点然后改变观察数据的视角。

*向下钻取。*可视化的第一个步骤通常是将结果用总和的形式表现出来。然后用户可以通过对可视化进行钻取，得到更进一步的细节。

高级的互动。这种技术提供了一个最低限度的用户接口。用户可以简单地通过对其中一部分的双击，然后拖动和放下，对数据实体进行新的展示。或者，通过单击并从菜单中进行选择。视觉查询是最先进的用户互动特征。例如，用户想看一张散点图的某个偏僻的数据点，然后用鼠标选中它们，要求看一个关于这些数据新的可视化结果。数据可视化软件根据所选的这些数据创造合适的查询，向数据库提交这个查询请求，然后通过另一种展现方式展现结果。

并行处理

数据仓库是以用户为中心和集中查询的环境。用户经常会通过复杂的查询来进行多种类型的分析。每一个查询都需要读取大量的数据来得到结果。通常每一个用户进行互动的分析的时候都需要好几个查询，依次进行。如果数据仓库不能适合大型、复杂、同时发生的查询处理，数据仓库的价值就没有了。性能是很重要的。

另外一个对于性能非常重要的功能是装载数据和建立索引的功能。因为大量的装载数据一般比较慢。而因为有很多不同的方法来连接数据，索引通常在数据仓库中非常精细。正是因为大量索引需要建立，所以创建索引的过程一般也比较慢。

那么，如何提高索引处理、数据装载和创建索引的速度呢？一个非常有效的方法就是使用并行处理。通过硬件结构和软件技术共同来完成并行处理的过程。一项任务被分为小的单元，同时对这些小单元进行处理。

并行处理的硬件。在并行处理环境中，你会发现这样几个特点：多 CPU，内存模块，一个或更多的服务器点，内部节点间的高速通信连接等等。

你可以从三种结构方法中进行选择。图 3-6 表现了这三种选择和它们相应的优点。搞清楚它们的优点和缺点，便于为数据仓库选择合适的方案。

并行处理软件实现。你可以为你的数据仓库选择合适的并行处理硬件配置。如果操作系统和数据库软件不能和并行处理硬件配合起来，那是没有用的。你要保证软件能够合适的为硬件组成部分分配任务。

并行处理软件必须要能够完成下面这些任务：

- 分析并辨认可以并行处理的独立单元。
- 辨认需要按照次序处理的较小单元。
- 能够并行的执行独立单元，并按照次序执行非独立单元。

- 收集，比较和合并较小单元产生的结果。

数据库产品提供商通常提供两种并行处理的选择：并行服务器或者并行查询。你可以分开购买两者中的一个。根据数据库提供商的设备不同，这些软件可以供一个或更多的并行硬件配置使用。

并行服务器可以允许每一个硬件节点拥有自己独立的数据库实例，使所有的数据库实例都可以连接普通的数据库底层文件。

而并行查询支持一些关键操作，例如查询处理、数据装载、并行的索引创建等等。

在当前的技术条件下，不使用并行处理的数据仓库是不可想象的。总之，在数据仓库中采用并行处理时，应该要了解并行处理的以下优点：

- 提高查询处理、数据装载和创建索引的性能；
- 易扩缩性，在不改变现有应用程序的条件下，允许增加 CPU 和内存模块；
- 增加容错性，使数据库可以接受一些并行处理器的错误；
- 数据库的单逻辑视角，即使数据是存在于多节点的磁盘中。

查询工具

在数据仓库中，查询工具是非常重要的功能工具。数据仓库的成败可以说就是依赖于你的查询工具。正因为如此，数据仓库提供商们在过去的几年中都在致力于提高查询工具的性能。

我们将在第十四章中将详细地介绍查询工具。在本章中，我们需要了解以下功能。

- 灵活的展现方式——方便使用，能够用多种方式在线或者使用报表展现结果。
- 对汇总的识别——能够识别总和的存在或者汇总表格，并在需要汇总结果时自动对汇总表格进行查询。
- 交叉主题区域——能够自动地在不同主题间交叉。
- 整合性——整合不同的查询工具，包括在线查询、批报表、分析用的数据抽取，并且提供一种类型输出到另一种的接口。
- 克服 SQL 的局限性——提供 SQL 的扩展，解决一般标准 SQL 不能处理的请求。

浏览工具

在这里我们所提到的“浏览”是一个普遍的概念，而不仅仅指 Web 浏览器。用户可能

会在数据仓库中运行查询，从数据仓库中创建报表，会直接使用这些功能而不借助 IT 工作人员的帮助。这些功能被认为是数据仓库的主要优点之一。

如果用户不得不直接接触数据仓库，他们就要知道这里究竟能得到哪些信息。他们需要好的浏览工具来浏览元数据并寻找他们需要的信息。也就是说，你作为公司开发数据仓库项目的工作组成员，需要确定数据来源、数据结构和商业规则。你还需要出色的浏览工具来对数据源的信息进行浏览。这里有几个关于浏览工具发展的当前趋势：

- 允许对任何数据或信息对象的定义。
- 包括了开放的应用程序接口（API）
- 提供几种类型的浏览功能，包括对分等级用户的区别
- 允许用户浏览数据字典或元数据，查找感兴趣的信息对象，并能进一步地运行具有相关参数合适的查询工具
- 提供 Web 浏览和查找技术

数据融合

数据仓库是整合多种来源数据来提供完整企业视图的地方。数据从不同平台上运行的操作型系统中抽取出来，例如平面文件或不同数据库管理系统支持的数据库。除了这些内部数据，数据仓库也必须包含来自外部的数据。在数据仓库知识库中，你还可以发现不同类型的非结构化数据，例如文档、图像、音频、视频等格式。

在根本上，从不同数据源得到的多种类型的数据需要整合或融合后存储到数据仓库中。数据融合是一项针对不同数据源的数据合并的技术。它具有较宽的范围，需要提供对数据的实时融合。数据融合技术领域正在进行一些重要的研究。数据融合技术的规则和技术在数据仓库中有直接的应用。

数据融合不仅可以处理不同数据源的数据融合问题，还可以在数据仓库中有其它的应用。在现在的仓库中，我们要收集巨大比例的数据。信息存储的越多，在合适时间内找到准确信息的困难就越大。数据融合技术正是可以解决这样的问题。

大体上来说，数据融合仍然处于研究的阶段。提供商们还不能提供数据融合的工具。在这个阶段中，你需要做的就是随时关注它的发展。

多维分析

今天，每一个数据仓库环境都提供多维分析。它已经逐渐成为数据仓库的信息传递系统的一部分。向用户提供多维分析也就意味着用户可以使用很多不同的方法来对商业进行分析。多维分析也可以成为联机分析处理（OLAP）。

正是因为 OLAP 的重要性，我们将在第十五章中详细学习这方面的知识。在本章中，我们要需要了解提供商已经在 OLAP 工具上有了很大的发展。

代理技术

软件代理是能够运行代表用户一个预先定义的任务的程序。举个例子来说，在 Internet 上，软件代理可以用来根据用户定义的规则对 e-mail 进行分类和过滤。在数据仓库，软件代理可以用来对一些用户预先定好的商业条件进行提醒。也可以用来进行数据挖掘和预测模块技术的连接。一些提供商可以提供预警系统的工具。

因为数据仓库的规模持续在增长，代理技术应用的越来越多。例如市场分析用户需要使用数据仓库进行严格的风险识别来为企业找到商业优势。分析人员需要运行几个查询来进行多层次的分析。这些条件是额外的条件。所以分析人员需要通过反复的分析工作。一些威胁和机会只有通过反复的分析才能够发现。这需要花费分析人员大量的时间。

当一种威胁条件或机会通过复杂的分析得到发现后，就可以向一个软件代理系统进行描述了。当这样的条件在未来重新出现的时候，这个程序将会自动地提醒分析人员。这就是代理技术的实质。

软件代理也可以用在对商业经营的日常管理中。你的老板可能想要在全公司的销售额忽然低于目标值时，随时得到消息。这时软件代理程序就可以帮助他或她，在每次这种情况发生的时候进行提醒。

从外部信息提供企业获得的数据

数据内容的价值不仅可以从内部操作型系统中获得，还可以从适合的外部数据获得。在日益进步的数据仓库应用中，从外部信息提供企业获得的数据的市场也在迅速扩大。

成为企业组合的数据的传统提供者的例子是提供零售信息的 A.C.Nielsen 信息源公司和提供金融和经济数据的 Dun & Bradstreet 通信社。一些早期的数据仓库通过与这些公司合作

来丰富数据内容。

现在数据仓库的开发者们正在寻找新的提供商来提供其它类型的外部数据。现在数据仓库可以从新的提供者那里得到示例图片、市场调查数据和其它类型的有用数据。

数据仓库和 ERP

很多公司在使用由一些大公司，例如 SAP、Baan、JD Edwards 和 PeopleSoft 等，提供的 ERP（企业流程再造）应用包。ERP 市场非常大，超过 450 亿的市场规模。

为什么企业都在实施 ERP 项目？大多数的公司都对大量的独立应用而不能提供完整的从整个公司视角的分析而深深苦恼。这些旧系统大多数已经过时。从这些系统中获得数据来得到有价值的信息非常的困难，而且在一些大公司中几乎已经不可能。一些公司开始寻找替代办法来解决旧系统带来的问题，例如千年虫问题。ERP 提供商似乎可以帮助企业解决这类问题。

ERP 软件包中的数据。ERP 软件包的一个重要特征是支持企业每天的商业阶段，从存货控制到客户帐单，从人力资源到生产管理，从产品成本到预算控制等等。正是因为这个特征，ERP 软件包非常大而且复杂。ERP 应用程序收集和整合了大量的企业数据。因为这些专有的应用，大量的数据使用专有的格式进行存储，只用通过专门语言编写的程序才能对其进行访问。通常，需要当量的关系数据库表单来支持所有这些功能。

整合 ERP 和数据仓库。在九十年代早期，ERP 的概念刚刚兴起，这种解决方案声称可以得到企业一直寻找的企业整合数据。因为所有的数据都在一个地方进行清洗、转换、整合，所以它的吸引人的地方在于能够在一个整合的环境中进行决策和计划制定。后来，很多公司应用 ERP 意识到大量的为商业操作设计和规范的关系数据库表单，并不完全适合提供战略信息。而且，ERP 数据知识库缺乏从外部数据源和公司其它操作型系统得到的数据。如果你的公司已经拥有或正在计划 ERP 系统，你需要考虑 ERP 和数据仓库系统的整合。

整合选择。如图 3-7 所示，公司整合 ERP 和数据仓库通常采用下列三种方法中的一种。ERP 提供商已经开始补充数据仓库解决方案软件包。采用选择 1 的公司与当前可实现功能一起，同时实施 ERP 提供商的数据仓库解决方案。在选择 2 中，公司实施用户定制的数据仓库，使用第三方工具从 ERP 数据系统中抽取数据。从 ERP 数据系统中得到并载入数据并不容易。选择 3 是一个混合方法，将数据仓库提供商提供的功能和第三方工具提供的附加功能结合起来。

你需要仔细检查这三种方法，为你的企业选择一个最适合的方法。

数据仓库和知识管理

如果说 1998 年标志着 ERP 的苏醒，那么 1999 年就是知识管理系统在很多公司苏醒的标志。知识管理（KM）发展的速度非常快。操作型系统处理数据，信息化系统（例如数据仓库）帮助用户对数据进行获取、整合、存储和转换，使之成为对分析和决策有用的信息。知识管理将这种帮助提高到了一个新的层次。它完成了使用正确的信息，在正确的时间和地点，向用户提供知识的处理过程。

知识管理。什么是知识管理？它是一个用户获取、整合、组织和沟通知识的系统过程。它可以帮助用户分享企业知识，使工作人员更加有效的完成工作。知识存在于企业的什么地方？合作过程，文件，分析报表，对象，数学模型，what-if 案例，文本流，视频剪辑等等，所有这些都包含了知识。

一个知识管理系统必须存储这样的知识，有时我们也称它为知识仓库。如果说一个数据仓库包含了结构化的信息，那么一个知识仓库就包含了非结构化的信息。因而，一个知识管理结构必须有查找和找回非结构化信息的工具。

数据仓库和知识管理。作为一个数据仓库的开发者，你认为知识管理什么样的？举一个例子，比如说 A 地区的销售额已经发生了下降。公司的市场副总裁可以通过在数据仓库中运行一些查询和分析发现这个情况。但是这个副总裁不知道为什么会发生这样的下降，但是如果这时候可以看到一个分析人员提供的关于为什么销售额下降以及建议解决方案的文档，问题就清楚了。这份文件包含了有关的知识，虽然这只是一个简单的例子。这位时常副总裁需要大量的信息，但是还需要其它更多的东西。

存储在自由的非结构化格式中的知识，一定要与销售结果相联系，才能提供价值。利用组织、查找和对非结构化数据的找回技术，更多的知识体系可以进入数据仓库。如图 3-8 所示，你可以通过包含从知识库中得到的数据来扩展你的数据仓库，而知识库是公司知识管理系统的一部分。

现在，通过上面的场景，这位市场副总裁可以从数据仓库中得到关于销售额下降的信息，然后从知识库中得到相关的分析报告。从知识管理系统中得到的知识可以为从知识仓库中得到的信息提供隐含在数据背后的线索。

数据仓库和 CRM

激烈的市场竞争环境使得很多企业越来越关注如何保留客户和赢得新客户。客户忠诚度系统已经成为普遍的解决方法。企业正在从整体市场向细分市场营销转化。以客户为中心已经成为了一句口号。改善客户服务的关键点在于对客户体验和客户行为的关注。越来越多的公司正在实施客户关系管理系统（CRM）。很多大的提供商提供没有实际用处的 CRM 解决方案，却声称可以进行对客户一对一的服务。

你的公司什么时候才能有高水准的客户服务，作为一个数据仓库设计者，你能做些什么？如果你已经有一个数据仓库，你如何对它进行重新调整？如果你正在建设一个新的数据仓库，需要特别注意什么？你将不得不使数据仓库更多地考虑客户的内容，使数据仓库能够进行客户关系的管理，而不是一个简单的工具。虽然这会比较困难，但是一个能够进行客户关系管理的数据仓库将会带来更大的回报。

能够进行客户关系管理的数据仓库。你的数据仓库必须有每个客户的每笔交易信息。这就意味着数据仓库知识库中，必须收集每个客户对每个商品的每一笔交易信息。你不仅需要详细的销售数据，而且需要每一个客户的偶然浏览的每一类信息。在将数据汇总的基础上，还需要在数据仓库中保存每个客户的每一个浏览。详细的数据为能够进行客户关系管理的数据仓库提供最大可能的灵活性。要想使数据仓库能够进行客户关系管理将会使数据量飞速增长，而幸运的是，今天的技术使大量的细节数据方便地存储在很多存储管理设备中。

为了使你的数据仓库能够进行客户关系管理，你还需要提高其它的一些功能。对于与客户相关的数据，清洗和转换的功能更加复杂。在向数据仓库导入客户姓名和地址纪录的时候，你必须对这些非结构化数据进行分析，将重复的地方删除，将他们组合成为不同的类别，并利用外部统计学和心理学数据丰富这些记录。这些是主要的工作。传统的数据仓库工具并不适合这种以客户为中心的应用程序的特别要求。

动态数据仓库存储

到目前为止，我们讨论了很多重要的与建立数据仓库需求相关的重要趋势。为什么不现在结束我们对发展趋势的讨论呢？我们来看看什么是活跃数据仓库存储。

你能想象将数据仓库在允许一千五百万公共用户每天获取信息的基础上，还向全球的 3 万用户，包括雇员、客户和商业合作人开放吗？你能想象这样的一个系统要做到 24 ×7 的

服务，并保证 99.9%的可靠性吗？你的数据仓库会迅速承担重要任务，而不仅仅是提供战略性信息了。这种情况就是动态数据仓库存储。

一对一服务。这就是一个全球公司使用一个动态数据仓库完成的服务。这家公司在超过 60 个国家和地区有业务，在超过 40 个国家有制造业工厂，在大约 30 个国家有研发机构，并且 200 个国家销售超过 5 万种商品。不仅向雇员，而且向外部团体开放数据仓库所得到的优势是非常显著的。供货商提高了需求计划和供应链管理；公司和它的分销商在不同的营销策略的计划制定上进行合作；客户可以进行迅速的购买决策。动态数据仓库真的实现了对客户和商业合作伙伴的一对一的服务。

标准的出现

回想我们在第一章中讨论过的，数据仓库环境是多种技术的混合。建立一个数据仓库需要多种技术类型的混合。这个范围是很广的：数据建模，数据抽取，数据转换，数据库管理系统，控制模块，报警系统代理，查询工具，分析工具，报表工具等等。

现在在数据仓库行业，以及每一个支持数据仓库的技术中，都有大量的提供商和产品。这就意味着你在建立数据仓库的时候，可以有很多的选择。这是一个好消息。但是，有一点不好，就是当你试图使用多个提供商的产品时，结果通常会产生混乱。这些多厂商产品需要在数据仓库中进行合作。

不幸的是，这些不同的产品之间没有统一的标准来进行信息和功能的交互。当你使用一个厂商提供的数据库产品，另一个厂商的查询和报表工具，第三家厂商的 OLAP 产品的时候，这三种产品彼此之间没有标准的信息交换方法。因此，有两个领域迫切需要标准：元数据相互交换和 OLAP 功能。

元数据就像是数据仓库的信息地图。每一个产品都会增加元数据的内容；每一种产品需要使用其它产品创造的元数据。元数据就像胶水一样，将所有的功能模块连接在一起。

几乎所有的现代数据仓库都有 OLAP 功能模块。如果没有 OLAP，你就不能提供多维分析，不能提供多视角的信息和完整的计算。所以说 OLAP 是非常重要的。

在接下来的部分，我们将一起来学习在两个重要领域中创建标准所带来的进步。虽然已经取得了进步，但是我们还没有完全在其它领域采用标准。

元数据

元数据的标准有两个独立的方面：元数据联合，以及对象管理组。

元数据联盟。1995 年的 10 月，作为一个由厂商和感兴趣的团体建立的协会，该协会一直在进行被称为“开放式信息模型（OIM）”的元数据标准制订工作。微软在 1998 年 12 月加入了这个协会，与其它主要厂商一起成为坚定的支持者。在 1999 年的 7 月，元数据协会接受开放式信息模型作为标准，并开始继续开发它的扩展部分。在 1999 年 11 月，该协会正在努力推动新的关键标准。

对象管理组。另一个厂商的组织，包括 Oracle，IBM，HP，Sun 和 Unisys 等公司，通过对象管理组来寻找元数据的标准。对象管理组是一个解决对象技术中标准问题的大型组织。在 2000 年 6 月，对象管理组将“通用数据仓库元模型（CWM）”作为数据仓库元数据交换的标准。

虽然在 2000 年 4 月，元数据协会和对象管理组都表示他们将会一起合作得到一个唯一标准，但是到现在这还是一个难以达到的目标。因为大多数的企业数据都是通过 Oracle、IBM 和微软的工具进行管理的，所以两个阵营之间的合作也更加重要了。

OLAP

OLAP 协会建立于 1995 年 1 月。它的会员和成员向所有有兴趣的组织开放。到 2000 年，这个协会有 16 个一般成员，主要是 OLAP 产品的提供商。

在这些年中，该协会致力于多维应用程序接口（MDAPI）的 OLAP 标准的工作，并且提出了修正。图 3-9 中表现了该协会主要活动的时间线。

一些 OLAP 提供商、平台提供商、咨询顾问和系统集成商宣布他们支持 MDAPI 2.0。

1999 年 1 月　　宣布 1999 年的工作重点

1998 年 11 月　　发布增强的分析处理标准

1998 年 1 月　　厂商宣布支持 MDAPI 2.0

协会发布互通性开放标准

1997 年 5 月　　NCR 加入协会

1996 年 9 月　　发布 MDAPI 2.0 标准

1996 年 5 月　　IBM 加入协会

1996 年 4 月　　发布第一个标准

1996 年 3 月　　Business Objects 公司加入协会

1995 年 1 月　　OLAP 协会成立

图 3-9　OLAP 协会时间线

实现 WEB 技术的数据仓库

我们都知道在过去的几年中对计算机和通信领域影响最大的就是 Internet。每一个重要的行业会议和每一份商业杂志，都在讨论和 Internet 或者 Web 技术相关的问题。

从 1969 年的四个主机系统用户开始，到 2000 年已经有大约 9500 万个主机连接在互联网上，而且这个数字仍然保持快速的增长。截止到 2000 年，互联网上的网站数量已经接近 2600 万。大约有一亿五千万的全球用户在使用互联网。通过使用 Web 技术，大量的公司建立了 Intranet 和 Extranet 与员工、客户和商业伙伴保持联系。网络已经成为通用的信息传递系统。

在最近的几年中，Internet 的发展还带来了电子交易的迅猛发展。B to B 的电子交易年交易额已经超过了三千亿美元，电子交易的总量即将达到一万亿美元。如果没有 Web 技术，企业将无法生存和竞争。在 Internet 上从事商业活动的公司数量预计到 2003 年将会达到 400，000。

作为数据仓库的专业人员，你得到了什么启示？很显然，你必须利用 Internet 和 Web 技术的巨大潜力来提高你的数据仓库的价值。而且，你需要认识到电子商务的重要性，增强数据仓库的性能来支持和扩展公司的电子商务。

你必须将数据仓库转换成为一个能够实现 Web 技术的数据仓库。一方面，你需要将数据仓库放入 Web 中去，另一方面，你要将 Web 引入到数据仓库中。在接下来的内容中，我们来讨论一个能够实现 Web 技术的数据仓库的两个方面。

将数据仓库放入 Web 中

在早期的应用中，企业数据仓库是用来帮助管理者、经理、商业分析人员和一些高层职员进行分析和决策的工具。这些使用者在一个客户机/服务器的环境中传递数据仓库中的信

息。但是今天的数据仓库不再是定义为仅供一小部分内部人员使用。在当前的条件下，公司需要提高整个公司价值链上所有成员的生产力。从数据仓库中得到的有用信息不仅要提供给内部工作人员，而且也要提供给客户、供货商以及其它所有合作伙伴。

所以在今天的商业大环境下，你需要向整个价值链上的所有用户开放你的数据仓库系统，甚至也可能要向所有的普通用户开放。这是一个很高的目标。你怎么样才能使数以千计的用户能够 24×7 的得到信息服务呢？你如何能够做到这样的要求，而不必为此支付高额的成本呢？通过使用 Web 技术的 Internet 就可以达到这个目的。Web 将会是你主要的信息传递方式。

这种新型的传递方法将会改变用户从数据仓库获取、分析和共享信息的习惯。信息传递的组成要素将会发生变化。Internet 接口包括浏览器、搜索引擎、进栈技术、主页、信息内容、超文本链接和下载 Java 或 ActiveX 应用程序等。

当你将数据仓库放入 Web 环境中，也就是在数据仓库中引入 Web 技术，从用户的角度来看，其中主要的需求是：自服务数据获得、互动分析、高性能、零管理客户机（也就是瘦客户机技术，例如 Java 应用小程序），高安全性和统一的元数据。

将 Web 技术引入到数据仓库

将 Web 技术引入到数据仓库主要包括了收集所有访问公司站点的点击流信息，以及执行所有传统数据仓库的功能。你必须几乎实时的在一个目前称之为数据网络仓库的环境中完成这些工作，包括将这些点击流数据抽取、转换和装载到网络仓库的知识库中。你要为这些点击流数据建立多维图示，为网络仓库配置信息传递系统。

这些点击流数据记录了人们是如何使用公司的网站，购买了哪些商品，吸引客户的是什么，以及是什么吸引他们再次访问。点击流数据能够反映的信息包括：

- 客户需求
- 市场推广的效果
- 各产品之间的联系
- 与客户相关的统计数据的收集
- 客户购买模式
- 主页设计的反馈

一个点击流网络仓库可能是辨认、排序和保留电子交易客户的唯一重要的工具。这样的网络仓库可以带来这样的有价值信息：

- 站点统计数据
- 来访者转换
- 广告效果
- 提供合作伙伴的链接
- 根据定制的站点导航
- 不根据定制的站点导航
- 建立客户资料和网页浏览之间的联系
- 对最好客户和最差客户的分析

实现 Web 技术的配置

在图 3-10 中，我们可以看到一个实现 Web 技术的数据仓库的体系结构。请注意传统数据仓库所提供的基本功能模块。在此基础上，网络仓库的知识库包含了点击流的数据。

普通大众　　客户　　商业合作伙伴　　内部职员

通过 Extranet 得到的结果

Web

点击流数据，通过 Extranet 得到的结果

数据仓库知识库　　网络仓库知识库

图 3-10　实现 Web 技术的数据仓库简单示意图

网络和数据仓库技术的结合，对于 21 世纪的每一家企业都有非常重要的意义。正因为如此，我们将在第 16 章中对这个问题进行详细的讨论。

本章小结

- 随着大规模数据仓库的普及和产品提供商数量的增长，数据仓库已经成为主流。

- 现代数据仓库需要存储多种类型的大量数据：结构性的和非结构性的，包括文本、图像、视频和音频等等。
- 数据展现解决了用多种可视化形式展现信息的问题，这些形式包括：文本、数组、电子表格、图表等等。在数据展现形式上已经有了很长足的进步。
- 可以通过适当的硬件或者软件进行并行处理来提高数据仓库的性能。
- 使数据仓库能够与 ERP 系统、知识管理和客户关系管理系统共同工作，是一个非常重要的问题。
- 数据仓库行业非常需要统一的标准，特别是元数据和 OLAP 领域。
- 实现 Web 技术的数据仓库是指使用 Web 技术来进行信息传递，将从公司网站收集的点击流信息整合起来以供分析的系统。数据仓库和 Web 技术的结合对于 21 世纪的每一家企业都有非常重要的意义。

思考题

1. 指出能够反映数据仓库持续增长的三个特点。你能想到一些例子吗？
2. 为什么数据仓库在规模、存储数据量上都一直保持增长的势头？至少给出三种原因。
3. 为什么存储多种类型数据对于数据仓库来说非常重要？举出一个健康管理组织的数据仓库中存储的非结构型数据的例子。
4. 数据融合是什么意思？在数据仓库的哪个部分用到这个概念？
5. 描述你在一个财政部门数据集市中看到的用来传递信息的四种类型的图表。
6. 什么是 SMP（对称多处理）并行处理硬件？描述这种配置。
7. 解释代理技术的概念以及数据仓库中是如何使用这种技术的。
8. 描述任何一种将 ERP 与数据仓库整合的方法。
9. 什么是 CRM？如何让企业的数据仓库能够进行客户关系管理？
10. 什么是实现 Web 技术的数据仓库？描述其功能的三个特点。

复习题

1. 判断下列说法的正确或错误：

   A. 数据仓库对市场分类有帮助。

   B. 在数据仓库中非结构型数据比结构型数据更重要。

   C. 动态图表就是用户接口。

   D. MPP 是一个共享内存的并行硬件配置。

   E. ERP 系统可能会被数据仓库取代。

   F. 一个公司的知识基础中大部分是非结构型数据。

   G. 传统的数据转换工具能够满足实现 Web 技术的数据仓库的需要。

   H. 元数据标准促进了多种产品的共同合作。

   I. MDAPI 是一种数据融合的标准。

   J. 一个实现 Web 技术的数据仓库只存储从公司网站上获得的点击流数据。

2. 作为一家零售连锁店的数据仓库项目中的主要分析人员，你有责任提高输出结果的数据可视化的工作。列出你的建议。

3. 解释并行处理是怎样以及为什么会提高数据装载和索引创建的性能。

4. 讨论代理技术的三种典型方法，这种技术在大型制造型公司中能够提高数据仓库价值。

5. 你的公司正在进行 DVD 和录像带出租业务。最近公司开展了电子商务的业务，主管希望实现已有数据仓库的 Web 化。列出能够满足主管要求的三项主要任务，并简单解释。

第四章　规划和项目管理

本章目标

- 复习对数据仓库进行规划的要点
- 区分数据仓库项目与 OLTP 系统项目的不同
- 学习如何在数据仓库项目中应用生命周期方法
- 讨论项目团队组织、规则和责任
- 考虑需要注意的问题和成功的要素

在你看到本章标题的时候，你可能会认为这是为项目经理或者项目协调员写的章节。如果你还不是项目经理或在近期内成为项目经理，你可能会想要略过这一章。这会是一个错误。本章其实是为所有 IT 专业人员设计的，并没有考虑你在数据仓库项目中担任的角色。在本章中，你会了解你如何在一个项目中扮演好自己的角色。如果你希望成为一个成功的数据仓库项目团队的一员，你需要仔细阅读本章的细节。

首先让我们来看下面这一段对话：

咨询者：那么，你的公司已经有了数据仓库？你有多少个数据集市？

项目经理：11 个。

咨询者：很好。但是为什么有这么多？

项目经理：有10 个是错误的。

虽然这样的对话有一点夸张，但是据行业专家透露，超过 50%的数据仓库项目最后都是失败的。在很多案例中，一般情况是没有完成项目或者是系统不能运行。在一些例子中，项目是完成了，但是数据仓库成为了一个数据基地。没有对项目进行正确的规划和确立项目的体系。数据仓库没有和业务实际联系起来。项目在中间阶段就被放弃了。

造成这些失败是有一些原因的。当你的公司第一次进行数据仓库系统的建设时，这个项目将会带来很多组织结构上的变化。目前的重点是在企业范围内的信息分析上。直到现在为止，每一个部门和每一个用户都有他们自己的数据，而且自认为有他们自己的一套计算机系统。数据仓库将会完全改变这样的情况，使那些管理者、数据拥有者和最终用户非常担心。在建设数据仓库的时候你需要克服这些生产系统的问题。

规划你的数据仓库

相对于其它的因素，不合适的规划和不正确的项目管理更加容易导致项目的失败。首先，你必须决定你的企业究竟是否需要一个数据仓库。已经准备好建设一个了吗？你需要为数据仓库建立一些标准，你的公司必须决定要建立的数据仓库的类型。必须确定数据仓库的数据来源，甚至要确定你是否能够获得所有需要的数据。你必须确定谁将会使用这个数据仓库，如何使用，何时使用。

我们将会讨论与正确规划数据仓库相关的几个问题。在这里，读者将会了解一个数据仓库项目与以前的项目究竟有怎样的不同。我们将要学习如何成功的建立一个数据仓库。

关键问题

规划你的数据仓库应该从对关键问题的彻底了解开始。这些关键问题的答案，对于正确规划和成功完成数据仓库系统来说非常的重要。因而，让我们逐个的来看看这些关键问题。

价值和期望。一些公司在开始数据仓库建设的时候，没有考虑他们将从数据仓库中获得什么样的价值。当然，首先你必须要在公司的文化和当前需求的背景下，确定数据仓库是最可行的解决方案。在确定了这个解决方案的合适性之后，你要开始列举将会带来的收益和价值。你的数据仓库将会帮助管理人员更好地完成计划和决策的工作吗？能够提高财务指标？能够提高市场份额？如果是这样的，成本是多少？期望值是什么？作为全部规划处理的一部分，列出一个可能收益和期望的清单。这是规划数据仓库的起点。

风险评估。规划者一般会将项目风险与项目成本联系起来。如果这个项目失败了，会浪费多少钱？但是，对风险的评估不仅仅是对项目成本损失的计算。如果没有从数据仓库中得到的收益，企业将会面临什么样的风险？会有什么样的损失？会错失什么样的机会？对风险的评估范围是很广的，与每一项业务内容相关。要利用公司的文化和商业条件来评估风险，在规划文档中包含这个评估的部分。

自上而下还是自下而上。在第二章中，我们讨论了自上而下和自下而上的建立数据仓库的方法。自上而下方法是从建设企业范围的数据仓库开始，尽管可能要多次反复，然后将整个企业范围内的数据放入部门的和不同主题的数据集市中去。另一种方法，也就是自下而上的方法是从逐个建立单独的数据集市开始，将这些数据集市组合起来组成完整的企业数据仓库。

我们在第二章中了解了这两种方法各自的优缺点，也讨论了一种自下而上的实用方法，同时也确定了一个单独的数据集市必须与其它的数据集市相符合，保证可以从一个整体的角度对它们进行观察。为了成功的实施这个方法，你必须首先从整个公司的高度规划和定义好需求。

你必须要确定这些方法是否适合公司的需要。首先你是否有先建立企业数据仓库后建立独立数据集市所需要的大型数据源？这个选择会花费应用的大量时间，并推迟对潜在受益的估计。但是这个选择能够保证对企业数据的统一的和完整的视角。

你的公司可能会对一些数据集市的迅速运用非常满意。这个时候，对市场竞争和压力的迅速反应也许会非常重要。这也许不是建立一个完整数据仓库的好时机。或者，你可能想要采用那种实用的方法先来建立数据集市。企业究竟采用哪一种方法，都需要仔细考虑后做出选择。在规划文档中，要包含选择该方法的原因。

建造还是购买。这个问题是所有的组织都会遇到的主要问题。没有一个人会从零开始的建造数据仓库。市场上有很多第三方的工具和解决方案可供选用。真正的问题是你自己建设数据集市究竟要花多少钱？如果使用已有的解决方案要花多少钱？应该使用哪种方式呢？

数据仓库有很广泛的功能应用，你想要编写更多的数据抽取和数据转换的内部程序吗？你想使用内部程序来进行数据仓库数据的装载吗？你想完全使用外部工具来进行信息传递吗？你可以在使用内部软件的任何地方对存货进行控制吗？

你要做的就是在规划阶段寻找内部和外部软件之间合适的平衡。

单独厂商还是多厂商融合。提供商有很多的种类，对数据仓库的很多功能都有很多的厂商和产品。所以，怎么进行选择呢？怎么决定呢？有两个主要的选择：（1）全部使用一家厂商的产品；（2）使用不止一家厂商的产品，选择最合适的工具。只选择一家厂商的解决方案有下面这些优点：

- 工具之间的高度集成性
- 外观和风格上比较统一
- 各个组成部分之间的无缝配合
- 集中管理的信息交换
- 可通过谈判解决整体价格

这个方法可以较好的整合数据仓库的各个功能。但是，只有几家厂商，例如 IBM 和 NCR，能够提供完整的解决方案。

而采用多个厂商的产品进行融合的方法，其优点有：

- 可以建立适合企业组织的系统环境
- 在数据库和工具之间不需要进行妥协
- 可以选择最适合功能应用的产品

使用多厂商融合的方法，不同厂商之间的工具的融合问题成为一个非常棘手的问题。如果采用这个方法，必须保证所有这些工具都是兼容的，保持对单个厂商的权威非常重要。而且，你会大大减弱对单个产品的讨价还价能力，还可能最后得到较高的整体报价。采用这个方法需要注意一点：采用一个厂商提供数据库和信息传递的功能，选择其它厂商来提供剩下的功能。然而，如果你的环境不是非常的技术化，不建议采用这种多厂商融合的方法。

商业的需求，而不是技术上的

让商业上的需求成为数据仓库的驱动力量，而不是技术上的。虽然这个观点看上去非常明显，但是你很难相信有多少数据仓库项目违背了这个原则。有很多数据仓库开发者对于能给用户提供好看的图像更加感兴趣，而很少关注真正的需求。他们利用技术上的深度来展示系统的能力。

请记住，数据仓库不是关于技术的，而是用来解决用户关于战略信息的需求的。不要在了解需求之前就开始计划建设数据仓库。从关注究竟需求什么样的信息开始，而不是如何提供这些信息。不要将注意力完全集中在工具上，支持用户需求的基本结构体系才是更重要的。

所以在进行所有规划之前，进行一个初步的需求调查。怎么做呢？这个阶段不需要任何细节，也不需要非常深入的调查，而是要了解用户总体的需求。你的目的是对于整个商业过程有一个大体的了解。这种初步需求调查的结果会帮助你进行总体的规划，对于项目的范围设定也很有帮助。而且，它还会帮助你对单个数据集市实施次序等问题进行第一次的规划。举个例子，你可能需要先设计营销部门的数据集市，然后是财务部门的数据集市，然后再考虑人力资源部门的。

那么你需要在初步需求调查中收集什么类型的信息呢？最基本的，你必须收集下面这些用户群体的基本信息：

- 每一个用户群体的任务和功能
- 每个群体使用的计算机系统
- 关键性能的反应指标
- 影响每个用户群体成功的因素

- 谁是客户，他们是怎么分类的
- 记录客户信息的数据类型
- 生产和销售的产品
- 产品和服务的分类
- 产品生产的场所和位置
- 收益的衡量单位——按照每个客户、每个产品还是每个地区
- 成本和收入的衡量单位
- 对战略信息的当前查询和报表

作为初步需求调查的一部分，还包括一个对数据源系统的检查。即使在这个阶段里，你必须非常了解数据从什么地方抽取的问题。因此，需要检查数据源系统的体系结构，寻找数据结构之间的联系。数据的质量怎么样？能得到什么样的文档？采用什么样的方法从源系统中抽取数据最为合适？你的整体规划中必须要包括数据源系统的信息。

一把手原则

没有高层管理人员的支持，一个公司的数据仓库项目就不可能成功。建设数据仓库项目必须从开始的那一天起就得到最高层领导的完全支持。

没有任何项目像企业级数据仓库项目这样需要从整个公司的视角来整合各种信息。没有一个部门或群体可以在公司里独立完成一个数据仓库的项目。

你必须保证得到了公司最高层管理者的关注和支持。数据仓库必须要经常协调各种矛盾的需求，它的高层负责人必须通过施加其影响来调和这些矛盾。在大多数建设数据仓库的公司里，首席执行官也是直接关注这个项目的成功与否，他或她会指定一些高级管理人员关注数据仓库项目每天的进展情况。一旦该项目遇到阻碍，这个高层负责人就会来解决这些问题。

数据仓库的可行性分析

即使你的公司是一个中等规模的公司，对数据仓库的所有投资也会达到几百万美元。这些花费大体是这样的：硬件占 31%，软件（包括数据库管理系统）占 24%，人力和系统整合占 35%，管理占 10%。你如何通过风险和收益（包括有形的和无形的）之间的平衡，证明这些花费都是合理的？如何计算投资回报率？

这不是一件简单的事情。直到数据仓库建好并投入使用，很多收益是看不到的。你的数

据仓库要允许用户用多种方法进行查询和分析变量，可以运行 what-if 分析和制定决策，而且不会局限于已经预定好的方法。谁能预测查询和分析的运行结果，它们能够制定什么样的重要决策？这些决策是如何带来收益的？

很多公司在建设数据仓库的时候不需要事先进行完整的成本收益证明分析。所做的证明分析主要是基于直觉的判断和潜在的竞争压力。在这些公司中，最高层管理者自己能够看到从数据整合中得到的利益，提高数据质量，自行定制查询和分析的方式，以及降低信息获取的难度。如果你的公司是这样的公司，那么恭喜你的好运气。做一些简单的证明分析，就可以开始项目了。

但是不是每一个公司的领导都是这么容易被说服呢？在很多公司中，需要好几种类型的规范证明。我们要使用一些典型的方法来证明数据仓库项目的可行性。看看下面的这些例子，找出一个可以在你的公司中适用的例子：

1. 计算当前技术下进行决策信息制定工作所需要的成本。将这个预算成本与使用数据仓库后的可能成本相比较，找出两者之间的比率。看看高层管理人员能否接受这个比率。
2. 计算建设数据仓库后带来的商业价值，以及收益增加、红利增加、收入增加值和市场份额的增加值。将这些商业价值的成本与数据仓库建设成本进行比较，进行成本收益的证明分析。
3. 做完整的训练。判断数据仓库可能影响的所有部分，还有那些可能会影响数据仓库的部分。从成本项开始，一个接一个的，包括硬件部分的购买或租用、提供商提供的软件、内部软件、安装和调试费用、长期技术支持和维护成本等等。然后将这些有形的或者无形的收益用货币单位来进行衡量，包括缩减的成本、提高的收益和在商业中提高的影响力。将这些分析用现金流来表示，计算投资收益率。

全面的计划

可以使用很多方法来寻找建设数据仓库的理由。这种理由很容易找到是因为竞争对手有一个数据仓库，或者首席信息官向首席执行官提了一个建议，建议将数据仓库作为解决公司信息问题的解决方案。在很多例子中，一个高层管理者会在一个会议或是研讨会上了解这样的想法。无论你的公司建设数据仓库的理由是什么，真正的理由都是从一个考虑成熟的计划开始的。这个计划是一个十分规范的文件，包括规定的方向、语气和目标，要摒弃个人的意

见和倾向。它一定要考虑很多选择和理由，讨论数据仓库的类型并列举对数据仓库的期望。这不是一个详细的项目计划书，它是一个全面的计划，包括了基础的配置、分析需求以及对项目的正式授权。

图 4-1 列出了在正式可行性分析报告中所包括内容的类型。

建立数据仓库的理由：全面计划的内容列表

简介

任务陈述

范围

目标和对象

关键问题和选择

证明

一把手原则

执行策略

试验时间进度表

项目授权

图 4-1 数据仓库可行性分析报告的全面计划表

数据仓库项目

作为一个 IT 专业人员，你以前已经接触过应用程序的项目。你知道在这些项目中要做什么，以及从计划到应用需要什么样的方法。你曾经是分析、设计、编程或测试阶段的成员。如果你曾经做过项目经理或团队经理，你就会知道如何监控和控制一个项目。一个项目就是一个项目，如果你见过一个 IT 项目，你就没有见过所有的吗？

这个问题的答案不是一个简单的是或者不是。数据仓库项目不同于建立交易处理系统的项目。如果你第一次接触数据仓库，你的第一个数据仓库项目将会显示出主要的不同。我们将会讨论这些不同，也会找出解决这些问题的方法。我们还会问一个关于信息技术和用户部门建设数据仓库的基本的问题。使用传统的系统开发生命周期（SDLC）方法会怎么样？我们可以在数据仓库项目中也使用这个方法吗？如果是这样的话，在生命周期中的发展阶段都

是什么？

有什么不同？

让我们来看看为什么数据仓库项目是不同的。你对于 OLTP 系统的项目比较熟悉，下面我们将通过与 OLTP 应用项目的比较来了解这些不同。

让我们来看看数据仓库的几个主要功能点。首先，你要有数据获取模块，接下来是数据存储模块，最后是信息传递模块。一个数据仓库系统至少需要这三个主要部分。你会注意到一个数据仓库项目与传统的 OLTP 应用在这三个部分上就有不同。让我们来看看这些不同图 4-2 列出了主要的方面。

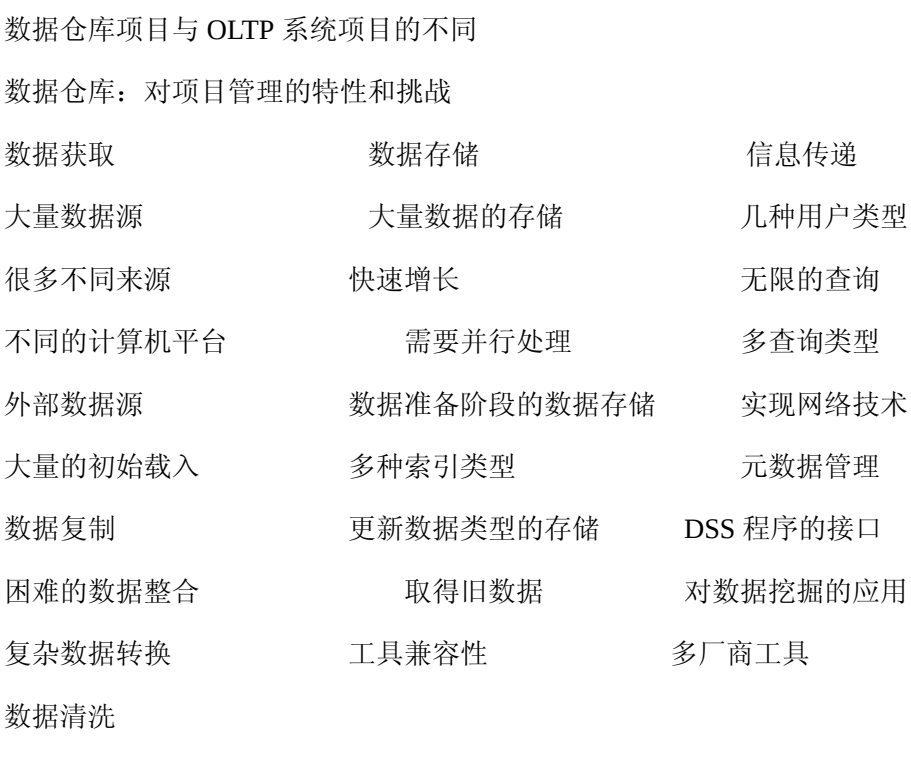


图 4-2　数据仓库项目的不同之处

数据仓库是一种新的范例，我们可以认为一个数据仓库项目与一个 OLTP 系统项目有很大不同。我们可以接受这种不同，但是更重要的是对这种不同带来结果的讨论。我们应该怎么做？如何改变项目实施的方法来适应新的项目？来看看下面的这些建议：

- 要意识到数据仓库项目有很广的范围，也会更加复杂，会用到很多不同的技术。
- 对新类型操作的要花费更多的时间和努力。

- 内部资源不能解决问题的时候，不要犹豫去找专家。一个数据仓库项目有很多不寻常的任务。
- 数据仓库中的元数据非常重要，需要整个项目的特别对待。要特别注意正确的建立元数据结构框架。
- 一般来说，你会在开发数据仓库的过程中使用一些第三方的工具，在项目时间表中，为评价和选择这些工具计划一定的时间。
- 为系统结构设计留出大量的时间。
- 在项目的每个阶段都让用户参与其中。数据仓库不仅对于公司中的 IT 人员，而且对于用户都是完全崭新的。这种共同的努力是必需的。
- 为培训用户使用查询和报表工具留出充足的时间。
- 因为数据仓库项目中有大量的任务，并行开发非常重要。在项目生命周期中做好并行开发的准备。

准备情况的评估

我们来看这个例子，你已经证明了数据仓库项目的可行性，也已经接到了来自最高层领导的支持，对数据仓库已经有全面的计划，也掌握了关键问题，并了解数据仓库项目的不同以及如何对待这些不同。现在你已经准备好立刻开始建设数据仓库了吗？

还没有。你需要做一个正式的准备情况的评估。通常，对于大多数的项目组成员和几乎所有的用户来说，数据仓库是一个全新的概念。一个关于准备情况的评估和定位非常重要。那么有谁来进行这样的评估呢？通常是项目经理在一个外部专家的帮助下完成这个工作。这时，项目经理已经有了数据仓库的培训或者已经有了初步的经验，参加与高层管理、用户和潜在团队成员的讨论。目标是对他们对于数据仓库的了解程度、准备程度和知识的盲区进行评估。在项目规划完成之前，准备一个正式的准备评估报告。

准备评估报告一般有以下的目的：

- 降低实施期间可能遇到的风险。
- 为问题的解决提供一个假设方法。
- 对公司承诺进行再评估。
- 对项目范围和规模进行再证明。
- 找出关键的成功因素。

- 重申用户期望
- 确定培训需求

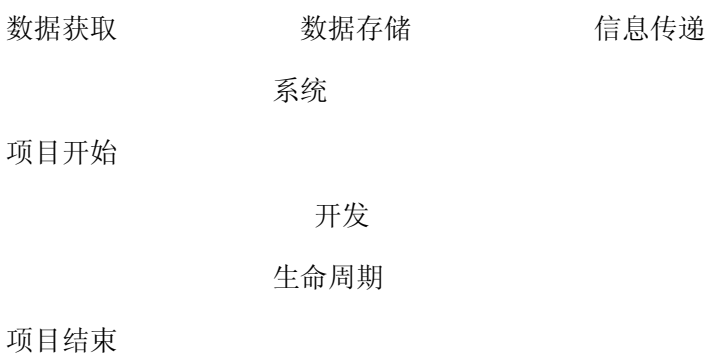
生命周期方法

作为一个 IT 专业人员，你对于传统的系统开发的生命周期（SDLC）一定非常熟悉。你知道如何开始一个项目规划，如何进入需求分析阶段，然后进入设计、建设和调试阶段，最后进入实施阶段。这个生命周期方法涵盖了系统开发过程的所有主要对象。它使整个过程秩序井然，用一个系统化的方法来建立计算机系统。生命周期方法论打破了项目的复杂性并且消除了关于项目组成员责任的任何不确定性。

生命周期方法论打破了项目的复杂性，使得这个方法完全可以在数据仓库项目中应用。一个数据仓库项目的复杂性表现在任务、技术和团队成员角色等方面。但是一个通用的生命周期方法不能直接搬到数据仓库项目中，通用的生命周期方法很难适应数据仓库项目的特别要求。注意数据仓库开发的生命周期方法不是一个瀑布方法，也就是说不是一个阶段结束后分级进入下一个阶段。

数据仓库项目的方法必须包括不断进行迭代的数据净化任务。例如，如果一个项目中的任务是对数据源进行识别，你就要开始对所有的源系统进行检查，并列出所有数据源的数据结构。这项任务的下一个迭代就是检查用户的数据元素。你要继续下一个迭代是检查数据库管理员和其它 IT 工作人员的数据元素。下一个任务是进一步对数据元素进行提纯。正是因为项目的复杂性，每一个任务都需要这种反复的过程。

记住一个数据仓库的主要功能组成部分包括数据获取、数据存储和信息传递。保证你的开发生命周期包括了所有这些功能。图 4-3 中表现了如何将这些功能组成部分与 SDLC 联系起来。



各阶段

图 4-3 数据仓库功能模块和系统开发生命周期

在任何一个系统开发周期中，数据仓库项目都从准备项目规划开始。项目规划描述了项目，识别特别的对象，考虑重要的成功因素，列出假说，并且突出关键问题。这个规划包括了项目时间表，列出任务及其分配，提供控制的进度。图 4-4 提供了一个数据仓库项目计划的概要。

简介

目的

准备情况的评估

目标和对象

假设

关键问题

成功因素

项目团队

项目时间表

开发细节

图 4-4　数据仓库项目计划：概要示例

开发阶段

在前面的章节中，我们提到了数据仓库的所有功能模块，包括数据获取、数据存储和信息传递。这三个功能模块组成了数据仓库的通用体系结构。必须要有正确的技术基本设施来支持这三个功能模块。因此，当我们在生命周期中规划开发阶段的时候，我们必须要保证这个阶段包括与这三个模块相关的任务。这个阶段还必须包括定义三个模块体系结构的任务，以及建立支持这个体系结构的底层结构。这三个模块的设计和实施阶段有时可以并行工作。

来看图 4-5，注意开发阶段的三个步骤。在每一个数据仓库的开发中，这些部分都由多个任务组成。你要改变这些任务使它们适合你的特殊要求。如果在你的公司中数据质量是一个问题，你需要特别注意这些相关的阶段。下面列出了将项目生命周期分为传统的几个主要

阶段：

- 项目规划
- 定义需求
- 设计
- 建设
- 部署
- 成长和维护

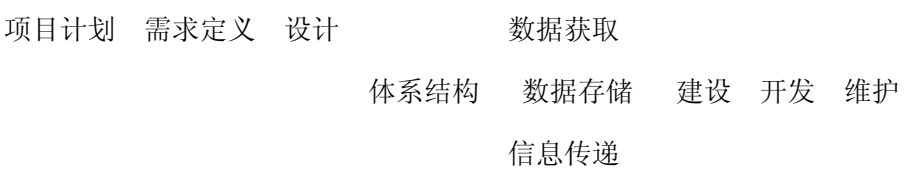


图 4-5　数据仓库开发阶段

图表中的这些矩阵都体现了独立的任务，并指定了合适的项目实施人员。使用这个表格作为一个向导，这里列出了数据仓库项目的主要任务。虽然大多数数据仓库的主要任务比较相似，但是每一个独立的任务会根据不同的项目有所不同。

本章接下来的章节，我们要进一步的讨论这些开发步骤。当你学到后面的时候，你就要回头看看上面这张表。

项目团队

在任何一种项目中，数据仓库项目的成败都依赖于项目开发的团队。好的团队就会带来成功。数据仓库项目在人员上与其它的软件项目比较类似，需要几个受过培训并有特别技能的人员来组成这个团队。为数据仓库项目组织的团队必须经过一定的培训，并且有一定的经验。

有两样东西会破坏一个项目：复杂性超负荷，以及责任不明确。在一个生命周期方法中，项目团队通过分担任务来减少任务的复杂性。当拥有合适类型技能和合适程度经验的团队成员独立完成一个任务的时候，这个人就会真正解决复杂性的问题。

在合理组成的项目团队中，每一个人都基于自己的能力和经验水平，承担自己的责任。在这样的团队中，就没有关于责任的不明确问题。

在接下来的章节中，我们将讨论如何赋予团队成员合适的角色。我们还要讨论这些角色相应的责任，以及需要的技能和经验。你将从中学到如何为你的数据仓库决定项目角色，试着为这些角色定义合适的责任和任务。

组织项目团队

组织一个项目团队包括让合适的人承担合适的工作。如果你正在为一个 OLTP 系统开发组织团队，你就会知道项目需要的技能是可以估计的，你需要的技能包括项目管理、需求分析、程序设计、数据库设计和程序测试等方面。但是一个数据仓库项目需要更多的角色。

一个好的开始是列出项目的所有挑战和需要的技能，你的清单可以包括这些内容：计划，定义数据需求，定义查询类型，数据建模，选择工具，设立物理数据库，对数据源进行抽取，数据确认，以及查询控制，建立元数据结构，等等。下一步是使用清单中列出的技能和挑战，准备一个需要支持开发工作的团队角色列表。

一旦你建立了角色列表，你就为分配人员做好了准备。不需要为每一个单独的角色都分配一个或者更多的人。如果你的数据仓库工作不是很大，或者你的公司资源不是很充裕，可以让同一个人负责几个角色。在这个人性化的分配过程中，记住使用者代表也必须是项目团队的成员。要让用户成为团队的一部分，并分配给他们合适的角色。

技能、经验和知识对于项目团队成员来说非常重要。此外，态度、团队精神、热情和强烈的责任感也同等重要。不要忽略这些重要的问题。

角色和责任

项目团队角色对应于一个或者多个相关任务。在很多数据仓库项目中，团队角色与团队成员的任务是同样的意思。如果你了解一个 OLTP 系统的开发项目，你就会发现项目成员的工作称谓或多或少是标准化的。在 OLTP 系统项目中，你会发现这些工作称谓，包括项目经理、商业分析人员、系统分析员、程序员、数据分析员、数据库管理员等等。但是，数据仓库项目的称谓定义还没有像这样的标准化，而且还有试验和研究的因素。

所以，什么是流行的工作称谓呢？首先让我们来看看图 4-6 中列出的称谓。不要被列表的长度吓住了，你不会需要这里所有的称谓。这个列表只是写出了所有可能的称谓。相同角色的责任可以在不同的项目中被表示成不同的称谓。在很多项目中，同一项目成员可以承担超过一个角色的责任。

数据仓库的从业者们一般使用不同的方法来对角色和工作进行分类。首先进行大体分类，然后将工作主题分别放入这些主要分类中。这里有一些角色的分类：

- 分类：早期开发人员、测试人员、后期维护人员、数据仓库管理人员
- 总体分类：IT 人员和最终用户，然后对这两种总体分类进行再分类，还可以继续进行分类。
- 分类：前台角色，后台角色
- 分类：培训人员，普通团队，特殊团队
- 分类：管理，开发，技术支持
- 分类：系统管理，数据获取，数据存储，信息传递

在你的数据仓库项目中，你可能想要首先进行最适合公司环境的总体分类。怎么做呢？你需要重新检查一下你的目标。你需要对在开发过程中需要特别注意的一些问题进行重新评估。例如，数据抽取是你的最大挑战吗？那么给这个工作一个特别的角色。你的信息传递功能会非常复杂吗？那么在信息传递部分使用特别的项目团队角色。一旦你决定了总体分类，那么下面的工作就在于每一种分类中的单独角色了。如果这是你的第一个数据仓库项目，你可能不能在一开始的时候就定义好所有的角色。不要担心，你可以在项目进行的过程中不断地增加新的团队成员角色。

你已经看过前面列出的可能的团队角色列表，以及他们的分类。这可能是你的第一个数据仓库项目，你可能就是那个决定团队角色的人。你会有一个问题：有没有一套标准的基本角色来使项目运转起来？答案是还没有。如果你比较倾向传统的方法论，你可以使用传统的分类方法：管理、开发和支持。如果你想突出数据仓库中三个主要功能模块，那么采用数据获取、数据存储和信息传递的分类方法。你还可以找到对这两种方法进行结合的新方法。

虽然没有一套标准的角色定义，但是在这里我们建议一套基本的角色定义：

- 执行负责人
- 项目经理
- 用户联络经理
- 体系结构总设计
- 基础设施专家
- 商业分析师
- 数据建模人员
- 数据仓库管理员

- 数据转换专家
- 质量保证分析师
- 测试协调员
- 用户应用程序专家
- 开发程序员
- 培训经理

图 4-7 列出了这些角色的相关责任。在你的数据仓库项目中，可以增加或修改对这些责任的描述，来适应项目的需要。

执行负责人

决定方向，支持并仲裁

项目经理

分配任务，检查进度并控制

用户联络经理

与用户进行合作

体系结构总设计

设计体系结构

基础设施专家

对基础设施进行设计和建设

商业分析师

定义需求

数据建模人员

相关性和维度建模

数据仓库管理员

数据库管理员的功能

数据转换专家

数据抽取、整合、转换

质量保证分析师

数据仓库中数据的质量控制

测试协调员

程序、系统和工具测试

用户应用程序专家

数据含义和关系的定义

开发程序员

内部程序和脚本的编写

培训经理

协调用户和团队的培训

图 4-7　数据仓库项目团队：角色和责任

技能和经验水平

我们已经讨论了决定项目角色的主要分类的方法。在你定义好数据仓库项目的相关分类后，你需要将这些团队角色应用到实际中。首先，你要写出这些已经建立好的角色的责任。接下来，你要为这些角色挑选合适的人选。在挑选之前，还有一个很重要的步骤。

为了适合角色及其责任，挑选出来的人必须有合适的能力。他们必须拥有合适的能力以及合适的工作经验。所以你必须要将这些不同角色所需要的技能和经验水平都列出来。图 4-8 给出了一个例子，描述了团队成员所需要的技能。你可以在你的数据仓库中使用图中列出的例子。

执行负责人

高级管理水平，对商业过程有深入的了解，有工作热情，有能力进行必要的仲裁

项目经理

与人交往的能力，项目管理经验，面向商业过程和用户，有实践和高效工作的能力

用户联络经理

与人交往的能力，在用户群体中有一定地位，组织能力，团队精神，从用户角度观察系统的能力

体系结构总设计

分析能力，总体规划的能力，接口方面的专业技能，有数据仓库概念的知识

基础设施专家

硬件、操作系统、计算机平台方面知识的专家，有数据仓库概念的知识

商业分析师

分析能力，与客户接触的能力，足够的行业经验，以及有分析工作经验

数据建模人员

在相关性和维度建模方面以及使用建模工具方面的专业知识和经验

数据仓库管理员

在物理数据库设计和实施方面的专家，有关系数据库、MDDBMS 的经验

数据转换专家

数据结构方面的知识，对数据源系统的深入了解，以及有分析工作经验

质量保证分析师

数据质量技术的知识，源数据系统数据的知识，以及有分析工作经验

测试协调员

对测试方法和标准、测试工具的使用非常熟悉，有一些数据仓库信息传递工具的知识，以及有程序员或者分析工作经验

用户应用程序专家

深入了解应用程序

开发程序员

编程和分析的技能，以及有专用语言编程和 DBMS 的经验

培训经理

培训技能，有 IT 人员或用户培训的经验，有协调和组织的技能

图 4-8　数据仓库项目团队：技能和经验水平

为数据仓库项目的团队选出所有的 IT 专业人员，不是一件容易的事情。目前，所有的 IT 专业人员都在或者曾经在 OLTP 系统项目中担任一定的角色。但是并没有很多的专业人员有数据仓库开发的经验。一些技能和经验是十分短缺的。

如果不能找到数据仓库项目需要的有经验的人员，那么你的资源是什么？你怎么样找到你的项目需要的人员？这就是培训的重要性。对合适的人员进行数据仓库概念和技术的相关培训。让他们学会对应于特定角色的基本原理和技能。除了培训内部人员，还要使用一些在内部无法找到的外部咨询人员。但是，作为一个普遍的规律，这些咨询人员不能担任领导的角色。项目经理或者主要管理者必须由组织内部人员担任。

用户参与

在一个典型的 OLTP 应用中，用户通过 GUI（图形用户接口）屏幕与系统进行交互。用户使用的数据输入和获取信息的屏幕，用户得到的任何信息都是从系统周期间隔创造的报表中得到的。如果用户需要特别的报表，他们不得不要求 IT 人员编写特别查询程序。

与之相对应的是，数据仓库的用户交互系统非常的直接和个性化。通常，没有或者只有少数固定的报表和查询。当项目建设完成的时候，用户可以直接使用数据仓库，而不需要 IT 人员的帮助。没有任何关于查询类型、报表类型或者分析过程类型的预先设定。如果说在 OLTP 系统和数据仓库系统之间有一点主要的不同的话，就是在用户使用这个系统的方法上。

那么，项目团队的组成与数据仓库开发之间有什么联系呢？如果用户使用事先没有预定的方法，直接使用数据仓库，他们必须在开发过程中有一定的影响力。他们必须在整个过程中是开发团队的一部分。与 OLTP 系统项目不同，一个数据仓库项目需要联合应用开发（JAD）技术。

只有团队成员中包括了合适的用户群体的代表，你的数据仓库项目才有可能成功。利用他们的专业技能和商业知识，以及在制定商业决策方面的经验，尤其是在信息传递工具的选择方面考虑他们的意见，在系统正式投入使用之前的测试工作中要得到用户的帮助。

图 4-9 中表示了用户如何参与到开发阶段中。回顾每一个开发阶段，决定用户如何以及在哪些阶段进行参与。这张图将用户参与和开发过程中的阶段联系起来。这里列出了在开发过程中用户可能参与的一些团队角色：

- 项目发起人——在整个过程中负责项目的执行
- 用户部门联络代表——帮助 IT 人员协调会议时间和进程；保证用户部门的积极参与
- 专题部分专家——提供用户在某些专门主题部分的需求向导；定义企业中商业团队使用的名词含义
- 数据检查专家——检查 IT 人员准备的数据模型；确认数据元素和数据关系
- 信息传递顾问——检查并测试信息传递工具；协助工具的选择
- 用户技术支持人员——为他们所代表的部门的用户提供第一步的技术支持

项目规划

在初步调查中，提供目标、期望、商业信息；保证高层管理的积极支持；作为项目发起人发起一个项目

需求定义

积极参与定义需求的会议；了解所有源系统；定义衡量商业成功的指标，以及进行分析的商业维度；定义数据仓库所需要的信息

设计

检查维度数据模型，数据抽取和转换的设计；提供数据库规模的预期；检查体系结构设计和元数据；参与工具选择；检查信息传递设计

建设

积极参与用户接受度的测试；测试信息传递工具；验证数据抽取和转换功能；保证数据质量；检验元数据的使用；检查查询功能；测试 OLAP 功能；参与制定应用文件。

实施

检验稽查步骤，保证最初数据的载入；保证供应能力与设计值一致；安排并参与用户培训

维护

提供进一步改进的输入；测试和接受改进

图 4-9　数据仓库开发：用户参与

项目管理需要考虑的事项

你已经组织好项目团队，完成了开发阶段，测试工作也结束，数据仓库已经实施，而且项目已经可以按期在预算内完成。所有的努力已经成功了？虽然这是项目团队的最好期望，但是最后实施的数据仓库可能是任何东西，而偏偏就不是一个数据仓库，如图 4-10 所示。

数据地下室

低质量数据，不能提供正确的连接

数据坟墓

一个昂贵的数据地下室，没有好的连接和性能

数据公寓

由一个早期系统提供商，或者一个不懂客户究竟要什么的无知顾问建立的系统

数据棚子

糟糕的数据堆放地，甚至在完成之前就崩溃了

数据茅屋

孤立的、支离破碎的数据集市孤岛

数据监狱

用户不能得到要求的数据

图 4-10　可能出现的失败

有效的项目管理对于数据仓库项目的成功非常重要。在这个部分中，我们要学习项目管理的内容，特别是在数据仓库项目中的应用，复习一些基本的项目管理原则，并列出成功的可能因素。我们将通过一个真实的成功案例来看看其成功的原因。但是，要指出的是，你不能在实际中完全照搬书中的内容。

项目管理的原则

如果你在 OLTP 系统项目中工作过，你就会了解一些项目管理的原则——不要陷入分析的陷阱，不要允许范围的扩大，监控整个过程，保证项目按照预定轨迹发展等等。虽然这些原则中的大部分也适用于数据仓库项目管理，我们不想在这里重复它们。但是，在这里我们要讨论一些适合数据仓库项目的原则。在项目的每个阶段，你必须将这些原则作为背景，使这些原则能够以每一个项目管理决策和行动为条件。主要的原则包括：

发起者的任务。如果没有强有力的执行发起者，数据仓库项目就无法成功。

项目经理。项目经理应该是面向用户和面向商业的，而绝对不是面向技术。

新范例。数据仓库对于大多数公司来说都是崭新的，创新的项目管理方法对于解决新的挑战非常重要。

团队角色。团队的角色不能随意的分配，这些角色必须反映出每一个独立数据仓库项目的需求。

数据质量。数据仓库的数据有三个主要方面包括：质量、质量和质量。

用户需求。用户需求组成了项目时间表的每一个任务的驱动力。

考虑增长的因素。在实施过后，会迅速出现大量的用户和查询；数据仓库的建设如果没

有考虑增长的因素，会很快的崩溃。

项目政治。公司实施的第一个数据仓库项目会遇到各种层次用户带来的挑战和威胁；试着处理项目政治就像在走钢丝，需要特别的谨慎。

现实的期望。在第一个数据仓库项目中很容易向所有人承诺；在正确和可以到达的阶段制定合理的期望，是最好的过程。

维度数据建模。一个设计精良的维度数据模型是一个必须的基础和蓝图。

外部数据。数据仓库不会完全依赖于内部数据；从相关外部数据源得到的数据是一个绝对需要的因素。

培训。数据仓库用户工具是不同的。如果用户不知道如何使用这些工具，他们就不会使用数据仓库。一个不能使用的数据仓库是一个失败的数据仓库。

警告征兆

当你的数据仓库项目的生命周期开始进行，你必须密切关注任何可能出现的会带来灾难的警告征兆。注意寻找任何可能会带来毁灭和失败的暗示。一些警告征兆可能是带来一些不方便，需要采取一点行动。但是可能会有一些警告征兆暗示了更广泛的问题，需要采取正确的行动来保证最好的成功。一些警告征兆可能会暗示严重的问题，需要采取立刻的补救措施。

无论这种警告征兆有多么的普通或自然，都必须谨慎对待。一旦你发现任何征兆，确认可能存在的问题，并采取行动来解决它。图 4-11 列出了一些典型的警告征兆，并有一些建议采取的措施。但是这些都仅仅是一些例子，在你的数据仓库项目中，你可能会发现其它类型的征兆。你需要根据情况的不同，采取正确的措施来解决这些问题。

警告征兆	暗示	行动
需求定义阶段超过了规定的日期	忍受“分析停顿”	停止对无用信息的收集。除去任何与用户相关的问题。严格保证最后的目标日期。
需要编写过多的内部程序	选择第三方工具	如果有时间和预算，选择不同的工具。否则增加编程人员。
用户在提供数据细节方面不合作	可能涉及到数据所有权的问题	非常棘手的问题。与执行发起人一起解决这个问题

用户对于查询工具不满意	没有对用户进行适当的培训	首先，确保使用的查询工具是合适的。然后提供另外的培训。
在数据的展现阶段持续出现问题	数据转换和映射阶段没有完成	重新访问所有的数据转换和整合路线。保证没有数据丢失。在这个确认过程中，一定要包括用户代表

图 4-11　数据仓库项目：预警征兆

成功的因素

你通过有效的项目管理，已经完成了数据仓库项目。你如何知道这个数据仓库项目是成功的呢？你需要三年或者五年来确定是否达到了预定的投资回收期（ROI）？在可以确定数据仓库项目成功之前，你需要等多久？或者，是否有一些征兆来帮助我们进行判断？

确实有一些成功的暗示，我们可以在实施后的很短时间内看到。下面就列出了几点通常情况下的成功因素：

- 查询和报表——用户直接从数据仓库中得到的查询和报表，其数量正在快速增长。
- 查询类型——查询变得越来越复杂
- 积极的用户——用户的数量稳步增长
- 用途——用户在数据仓库中寻找解决方案的时间越来越长
- 工作步骤——获取战略信息所需要的步骤越来越少

图 4-12 中列出了一个成功数据仓库项目的成功因素。这里列出的不可能是所有可能的成功因素，也不是一个在任何条件下都能成功的保证。你需要根据你的项目的情况，了解确保项目成功的因素，以及它的特殊的项目管理挑战。因此，将这张表作为普遍意义上的参照。

保证执行发起人的持续、长期支持。

从头开始，在数据仓库中建立定义明确的、真正的、大家认可的商业价值。实际地管理用户期望。

用户积极参与到整个项目过程中。

数据抽取、转换和装载功能是最耗时、耗力的。不要低估这个部分的时间和工作量。

首先确定体系结构，然后是技术，然后是工具。根据你的环境选择一个适合的体系结构。

为用户选择合适的查询和信息工具是非常重要的。要选择最有用和方便使用的工具，而不是看上去最好的。

计划好成长和发展。注意性能的考虑。

分配一个面向客户的项目经理。

在查询上集中设计，而不是在互动操作上。

定义正确的数据源，只载入需要的数据。

图 4-12　数据仓库项目：关键成功因素

一个成功项目的细审

无论你了解了多少成功的因素，也无论你学习了多少成功的向导，你都必须通过分析一个真实的成功案例来更好地掌握成功的原则。接下来我们就要来看一个真实的案例，这个案例中有一个非常成功的数据仓库项目。这个仓库完成了所有目标，并能得到期望的结果。图 4-13 中仔细介绍了这个数据仓库，包括成功的因素和收益。

商业背景

BigCom 公司，世界领先的数据、音像通信技术的提供者，有超过 3 亿的客户，而且近期有显著的增长。

技术和方法

通过实施大型公司级数据仓库，来为 1000 名用户提供战略信息；使用一个厂商提供的工具来进行数据抽取，并建立数据集市；使用其它厂商的查询和分析工具。

挑战

不能完全得到全球信息；缺乏通用的数据定义；在大量独立的应用系统中锁住了大量重要的商业数据；很多报表需要复杂的协调工作；每日的备份和更新所需要的重要系统停工。

成功因素

清晰的商业目标；强有力的上级支持；用户部门的积极参与；正确的工具选择；首先建立合适的体系结构；对于数据整合和转换有足够的重视；强调灵活性。

得到的收益

真实的企业决策支持；增强销售的评价体系；降低成本；简化商业过程；提高客户关系

管理；减少 IT 开发；可以得到公司网站上的点击数据。

图 4-13　一个成功数据仓库的分析

采用一个实用的方法

我们已经阐明了整个项目管理原则，描述了规划方法，并发现了一些理论上的细微差别，还需要一个实用的方法来达到最终的结果。不要受到这些原则、方法的限制，采用一个实用的方法来管理项目。重视最后的结果，只追逐理论上的原则永远不会得到期望的结果。

一个实用方法仅仅是一个通常意义上的方法，它是理论和实践的统一体。在使用一个实用方法的时候，你是完全面向结果的。你需要在衡量和平衡所采取行动的重要性，不是受到技术利益的驱动，应该是商业需求的驱动。

在数据仓库项目中，这里有一些采用实用方法的提示：

- 使用实际的方法来实施一个项目，意味着严格控制好偏离和滑动，在实施过程中进行纠正。在需要的时候重新分配优先级别。
- 将项目日期表作为工作流程和得到结果的向导，而不仅仅是控制和抑制创新。请不要试图控制每一个细节，然后你才能完成最新的日程表，用最少的时间做真正的工作。
- 持续的检查项目任务的从属性。将人物互相之间的依赖时间降低到最小。
- 有一种情况叫做“过度规划”。有时候，随时准备救火是一种有用的原则。
- 简单的说，“过度分析”可能造成“分析停顿”。
- 避免未经检验的技术。如果这个项目是公司的第一个数据仓库项目，这一点非常重要。
- 总是要将创造早期的供应能力作为项目的一部分。这些供应能力将会维持用户的兴趣。
- 首先是体系结构，然后是工具。不要在选择了工具之后，根据工具来建设数据仓库。首先根据商业需求建立体系结构，然后根据体系结构选择工具。

复习这些建议，然后在你的数据仓库项目正确地使用。特别是如果这是你们的第一个数据仓库项目，用户会对快速和容易见到的收益很感兴趣。你不久就会发现他们不再对你的项目进行日程感兴趣，而仅仅重视结果。他们只关心数据仓库是如何的有用并方便使用。

本章小结

- 在规划数据仓库项目的时候，需要注意的地方有：确立合适的期望，评估风险，决定自上而下还是自下而上的方法，选择厂商提供的解决方案。
- 驱动项目的是商业需求，而不是技术。
- 一个没有最高层领导支持、没有强有力的执行发起人的数据仓库项目，从第一天开始就注定是失败的。
- 只有在用户完全使用后，才能产生数据仓库的收益。计算项目的 ROI 不是容易的事情。在对潜在收益进行评估后，一些数据仓库项目才真正开始。
- 数据仓库项目与一个典型的 OLTP 系统项目有很大的不同。传统的系统开发生命周期方法需要进行改变才能够适应数据仓库项目的开发。
- 组织的标准和团队角色的分配在很多项目中还处于试验的阶段。根据你的项目情况分配团队角色。
- 用户的参与是数据仓库项目成功的重要因素。用户可以使用各种不同的方法参与项目。
- 考虑警告征兆和成功的因素；在最后的分析过程中，采用一个实用方法来完成一个成功的数据仓库。

思考题

1. 举出在对数据仓库规划过程中需要考虑的 4 个主要因素。
2. 解释自上而下和自下而上的两种建立数据仓库方法的不同。你有什么偏好吗？如果有，为什么？
3. 分别列出单独厂商和多厂商提供解决方案的三项优点。
4. 最初的需求调查是什么意思？列出在最初的调查中能收集到的六种信息类型。
5. 数据仓库项目与 OLTP 系统项目有什么不同？列出四个方面。
6. 举出并解释数据仓库项目的生命周期中的四个开发阶段。
7. 你是一个数据仓库项目团队角色中的核心小组吗？描述你的小组中三个角色的责任。
8. 列出任意三种数据仓库项目中可能遇到的警告征兆。为了解决这三种警告征兆所暗示的

潜在问题，你需要采取哪些行动？

9. 举出并解释在数据仓库项目中的任意五种成功因素。
10. 对于数据仓库项目管理来说，什么是“采用一个实用方法”？为什么一个实用方法可能会带来成功，给出两点理由。

复习题

1. 连线：

1．自上而下方法	A.走钢丝
2．单厂商解决方案	B.没有标准化
3．团队角色	C.成功的必需条件
4．团队组织	企业数据仓库
5．角色分类	E.持续的观察
6．用户支持技术	F.前端功能，后端功能
7．执行发起者	G. 完整规划中的部分
8．项目政治	H 正确的人担任正确的角色
9．积极的用户参与	I 前线支持
10．源系统结构	J。向导和支持项目

2. 作为项目经理，你需要和执行发起人一起为公司的第一个数据仓库项目，编写一个没有具体 ROI 计算的项目论证。编写这个文件，并包括规划文档。
3. 你是一家航空公司第一个数据仓库项目的数据转换专家。准备一个项目任务列表，包括所有数据抽取和转换需要的具体任务。
4. 为什么说用户参与对于最后的成功起着绝对重要的作用？作为一家银行最新成立的数据仓库团队的成员之一，你的工作是针对用户部门如何更好的参与开发过程，写一份报告。你将在你的报告中包含用户的哪些特别责任？
5. 作为一家大型零售连锁店数据仓库的体系结构的主要设计者，准备一个与体系结构设计相关的项目任务列表。这些任务主要体现在哪些开发过程中？

第五章　定义商业需求

本章目标

- 讨论为什么数据仓库有不同的需求定义，以及有怎样的不同
- 明确商业维度的概念
- 学习信息包的概念及其在需求定义中的用途
- 复习获取信息的方法
- 掌握一个常用需求定义文档的重要性

数据仓库是一个信息传递系统。这不是一个关于技术的系统，而是解决用户问题和向用户提供战略信息的系统。在需求定义阶段，你需要将重点放在用户需要哪些信息，而不是你如何提供这些需要的信息。提供信息的方法后来总会有的，但是不是在你收集需求的时候考虑这些问题。

大多数的数据仓库开发者有很强的开发操作性或是 OLTP 系统的背景。OLTP 系统主要是数据获取系统。另一方面，数据仓库系统是信息传递系统。当你开始为数据仓库收集数据的时候，你的思维应该是不一样的。你需要从一个数据获取的模式，转换成为一个信息传递的模式。这种不同将在数据仓库项目的整个阶段体现出来。

用户也有关于数据仓库系统的不同的视角。与需要进行每天商业操作的 OLTP 系统不同的是，在一个决策支持系统中不需要实时的输出。用户对决策支持系统的需求不会那么迫切，而不会像操作型系统那样，如果停止工作，业务就无法开展。

维度分析

建立一个数据仓库与建立一个操作型系统在很多方面有很大的不同，特别是在需求收集阶段。正因为这些不同，适合操作型系统的收集需求的传统方法将不再适合数据仓库系统。

不可预知信息的使用

假设你正在为公司的订单处理工作建立一个操作型系统。为了收集需求，你访问了订单处理部门的用户。这些用户会列出所有需要的功能。他们会告诉你如何接受订单、检查存货、

确认客户信用等级、定价、决定出货方式，以及将货发送到正确的仓库。他们会告诉你需要在应用程序的图形用户接口屏幕上显示哪些数据项，还会给你一个关于订单处理应用所需报表的清单。你可以了解他们是如何以及何时使用这个应用程序。

在向操作型系统提供需求信息的方面，用户可以告诉你关于需求功能、信息内容、使用类型等在内的准确细节。在这方面，数据仓库系统有很大的不同。用户不能清楚地定义他们的需求。他们不能准确定义他们真正想从数据仓库中得到哪些信息，也不能说明他们如何使用这些信息。

对于大多数的使用者，这可能是他们接触的第一个数据仓库。他们对于操作型系统非常熟悉，因为在日常工作中经常使用这些系统，所以他们可以为另一个操作型系统描述出需求。但是他们无法将数据仓库与以前接触过的任何系统联系起来。

因此，如果数据仓库定义需求的整个过程是如此的模糊，那么你怎么对数据仓库项目进行分析呢？你将处于一个进退两难的境地。为了安全起见，你将包含你认为用户能够使用的所有数据？你如何建立用户不能准确定义的那一部分内容呢？

最初，你会收集公司整个商业的数据。你会参考整个行业的实践经验，收集指导每天决策制定的一些商业规则，你会了解产品和市场是如何开发的。但是这些对于决定细节需求来说都是不够的。

商业数据的维度

幸运的是，情况并不像看上去那么糟糕。即使用户不能明确的描述他们究竟想在数据仓库中得到什么，他们也可以提供一些重要的关于商业的观察线索。他们可以告诉你什么是重要的衡量指标。每一个用户部门都可以帮助你了解他们如何衡量成功的指标，以及他们如何将各种信息综合成为战略决策。

管理者们从商业维度的方面来看待商业问题。图 5-1 中表示了管理者们在决策制定的时候可能问的各种问题，以典型的市场部副部长、市场经理和一个财务经理可能会问的问题为例。

市场部副部长

新产品创造了多少利润，按照月份，在南部地区，按照用户地理位置，按照销售部门，参照历史数据，并且与计划数据相比较？

市场经理

给我一些统计数据，按照产品种类、每天、每星期、每月进行汇总，按照销售地区，按照销售渠道进行统计。

财务经理

给我关于费用的列表，与预算比较，按照每月、每季度和每年，按照预算金额，按照地区，对全公司进行汇总统计。

图 5-1　管理者对商业维度的思考

让我们主要看看这些问题。市场部副部长所感兴趣的是每个月，某个特定地区，按照销售部门，参照历史数据和计划数据，所得到的新产品创造的利润。这些就是市场部副部长在分析数据时使用的商业维度。

对于市场经理来说，他的商业维度是产品、产品种类、时间（日、星期、月份）、销售地区和销售渠道。对于财务经理来说，商业维度是预算、时间（月、季度、年份）和地区。

如果数据仓库的用户使用商业维度进行决策制定，你也要在收集需求的时候考虑这些商业维度。虽然这个时候数据仓库的实际用处还不明显，但是制定决策的管理者使用的商业维度并不是模糊的。用户能够向你描述这些商业维度。你不会在需求定义的处理中迷失方向，你必须能发现这些商业维度。

接下来让我们试着具体掌握这些商业数据的维度。图 5-2 中展示了对于销售数据的分析，包括产品、时间和地理三个商业维度。这三种维度通过三个坐标轴来表示。你会看到一个由这三个维度组成的立方体（cube），会发现销售数据按照时间、产品和地理分类的切片。在这个例子中，销售部门的商业数据是三维的，因为我们在分析中只使用了三个维度。如果有超过三个的维度，我们要扩展成多维的立方体，也叫做超立方体（hypercube）。

商业维度的例子

商业维度的概念对于数据仓库的需求定义非常重要。因此，我们要了解更多的商业维度的例子。图 5-3 给出了商业维度的四个不同的例子。

让我们迅速的来看看这些例子。对于超级连锁商场来说，分析的标准是销售个体。我们来看这个超立方体，它的边包括时间、推销、产品和存储。如果你是这个超级连锁商场的市场经理，你会希望按照产品，在每个商场，按照时间顺序，与当地的促销行动相联系的，来

分析你的销售情况。

对于一个保险公司，商业维度与上面是不同的，包括代理商、个人索赔、时间、保险团体、个人政策和索赔状态。在航空公司的例子中，我们可以看到对频繁的飞行数据的分析维度，这些维度包括时间、客户、航线、费用等级、机场和飞行状态。

对于制造型企业的出货分析，提供了一些其它的商业维度，包括时间、出货/入货地点、运输方式、产品和其它相关细节。

在这些例子中，我们发现商业维度是根据行业和分析主题的不同有所不同。我们也可以看到时间维在所有的例子中是一个普遍的维度。几乎所有的商业分析都和时间有关。

超级市场

销售个体：时间　促销　产品　存储

制造型公司

出货：时间　发货客户　进货客户　运输方式、产品 交易

保险公司

索赔：时间　代理人　索赔　保险群体　政策　状态

航空公司

飞行：时间、客户、航线、费用等级、机场　飞行状态

图 5-3　商业维度的例子

信息包——一个新概念

我们现在要介绍一个新概念，收集和记录数据仓库的信息需求。这个概念将帮助我们综合在收集需求过程中产生的各种线索和模糊的想法。在收集需求过程中组成的信息包，对于数据仓库开发向下一个阶段推进非常有用。

不完全确定的需求

我们已经讨论过用户不能完全描述他们希望在数据仓库中得到什么。你也无法了解究竟数据仓库中需要哪些信息，不能确定用途类型，不能决定每一类的用户会如何使用新系统。所以，当需求不能完全确定的时候，我们需要一个新的概念来收集和记录需求。操作型系统

提供传统的方法并不能适应这个时候的需求。我们不能从功能、屏幕和报表开始我们的系统，也不能从数据结构开始。我们已经注意到用户开始考虑商业维度，并根据这些商业维度进行分析。这是一个重要的发现，能够构成收集信息的重要基础。

这个新的方法论——关于数据仓库的需求定义——是基于商业维度的。它将用户的需求放在商业维度的基础上进行分析。这个新概念结合了基本的衡量指标和用户用来分析这些基本指标的商业维度。使用这个新的方法论，你就会了解数据仓库中必须收集和存储的指标和相关的维度，以及特定主题下的信息包。

让我们来看看某行业的销售分析中的一个信息包，如图 5-4 所示。这里的主题是销售，分析的指标在图表的底部。在这个例子中，指标是当前销售、预测销售和预算销售。在图表的顶部表示了分析所用的商业指标。这些维度包括时间、地点、产品和人口群体年龄统计。每一个商业维度都包括了一个层次。例如，时间维度包括了从年份到每一天的不同层次。另一个时间维度的中间层次可能是季度、月份和星期。这些层次或者等级部分都在信息包图表中显示。

你在需求定义阶段的最初目标是按照数据仓库的所有主题编辑这些信息包。一旦你建立了这些信息包，你就能够进入下面的阶段。

信息包的用途可以归纳成下面几点：

- 定义普通主题范围
- 设计主要商业指标
- 决定数据展现形势
- 决定用户如何聚集和组合
- 决定用户分析或查询的数据数量
- 决定如何存取数据
- 建立数据粒度
- 估计数据仓库规模
- 决定数据更新频率
- 确定如何打包信息

商业维度

我们已经学过商业维度组成了需求定义新方法论的基础。数据必须被存储用来提供商业

维度，商业维度及其层次组成了所有未来阶段的基础。所以我们要更进一步地了解商业维度。我们要能够辨认商业维度和它们的等级层次。我们必须能够选择与商业指标相关联的合适的商业维度。

我们以一个汽车制造商的商业维度为例来看。例如我们的目标是分析销售数据。我们要建立一个数据仓库，允许用户使用各种方法来进行分析。第一个明显的维度是产品维度。对于制造者来说，销售的分析必须包括对经销商的销售的分析。因此，经销商是另一个重要的分析维度。作为一个制造者，你还想要知道谁购买了你的产品，以及购买了多少。客户人口统计就是另一个有用的商业分析维度。客户是如何支付的呢？交易中采用什么样的支付方式？这些问题的回答都要求将支付方法作为维度之一。几乎每一个查询或分析都包含了时间的因素。总之，从上面的缝隙中，我们有了下面这些维度：产品、销售商、客户人口统计、支付方式和时间。

让我们再看一个例子。在这个例子中，我们要看看一个连锁旅馆的信息包。这个例子的主题是客房使用量。我们要分析这个连锁旅馆的不同分支机构的客房使用情况。我们要分析每一个旅馆和每一种客房类型的使用情况。在这个例子中，我们需要包括时间维度，还有旅馆、客房类型的维度。

维度层次和范畴

当一个用户根据商业维度分析指标的时候，用户通常首先希望看到总数，然后是各个层次的细节数据。用户在这里所做的就是穿过商业维度的不同层次，来得到不同细节程度的数据。例如，用户首先看了全年的总体数据。然后看了季度的数据，看了每个季度的销售情况。在此之后，用户更进一步的看了每个月份的数据。这里我们注意到，时间维度的层次由年、季度和月份组成。这种维度层次就是我们在分析中向下钻取的路径。

在每一个主要的商业维度中，都有对分析有用的数据元素的分类。在时间维度中，你会有一个数据元素来表示一个特定的日期是否为假日。这个数据元素可以帮助你分析与其它日子相比，假期的销售情况有什么不同。同样，在产品维度中，你会想要对产品包的类型进行分析。这种产品包的类型就是产品维度中的数据元素。时间维度中的假日标志和产品维度中的产品包类型在这些维度中不是必需的维度层次。这样的数据元素也称为范畴。

每一个维度的信息包中都包含了层次和范畴。让我们回到前面谈过的两个例子，找出什么样的层次和范畴是维度中必需的。首先我们来看看产品维度。这里，产品是基本的汽车。

因此，我们将与产品相关的数据元素找出来，包括模型名称、年份、生产线、产品种类、外表颜色、内部颜色等等。来看另一个关于汽车销售分析的商业维度，我们将每一个维度的层次和范畴列出如下：

产品：名称、年份、型号、生产线、外部颜色、内部颜色

经销商：名称、城市、地区、商标、首次经销时间

客户统计：年龄、性别、收入范围、婚姻状况、拥有汽车情况、住所价值、

支付方法：金融类型、利率、代理人

时间：日、月、季度、年、星期、日期、季节、假日标示

让我们回到前面的那个旅馆分析的例子。我们已经讨论了三个商业维度。这里我们列出了可能的层次和范畴：

旅馆：旅馆、分支机构名称、分支编码、地区、地址、城市、邮政编码、管理者、建设年份、修缮年份

客房类型：客房类型、房间大小、床位数、床位类型、最大容量、套房家具、冰箱、厨房

时间：日期、月份、星期、季度、年份、假日标示

关键商业指标或事实

在上面两个例子中，我们已经讨论了这么多商业维度的问题。这些都是与这两个数据仓库的用户相关的商业维度。用户代表按照这些商业维度来考虑他们的商业主题，进行收集信息和分析。

但是使用这些商业维度的时候，用户究竟在分析什么？他们在分析什么数据？用户分析的这些数据就是衡量他所在部门的业绩的指标。这些事实向用户表明了他的部门是如何完成部门任务的。

在汽车制造商的例子中，这些指标是和销售相关的。这些数据告诉用户他们的销售情况，关于每一辆汽车的销售。这组有价值的分析指标如下所列：

当前销售价格

实际销售价格

可选价格

完全价格

经销商附加

经销商信用

经销商发货单

已支付数目

生产者收益

已提供资金数

在第二个旅馆的例子中，这些指标是不同的，依赖于分析的内容。对于旅馆客房来说，这些指标与每一个分支旅馆的指标相联系。下面有一个列表：

已租客房数

空闲客房数

不能使用客房数

居住者人数

收益

现在将它们放在一起，我们来讨论这两个例子中关于信息包的问题。在每一个例子里，指标或事实都表示在信息包图表的底部。商业维度作为柱状图的标题，每一个柱体都包含了商业维度的层次和范畴。

图 5-5 和图 5-6 是我们刚才讨论的两个例子的信息包示意图。

收集需求的方法

现在我们有了一个可以通过信息包图表定义需求的方法，让我们来讨论收集需求的方法。记住数据仓库是一个为战略决策制定提供信息的信息传递系统，而不是一个运行每日业务的系统。谁是能够在数据仓库中使用信息的用户？你从哪里得到用户的需求？

总的说来，我们可以将数据仓库的用户分为以下几类：

高层主管（包括发起人）

关键部门领导

商业分析师

操作型系统数据库管理员

上面这些人员指定的其它人员

经理主管人员会给你指定数据仓库的方向和范围。关键部门的管理者会向经理主管人员提交报表。商业分析师为主管和经理们准备这些报表。操作型系统数据库管理员和 IT 应用人员为你提供数据仓库源系统的信息。

你需要收集哪些需求？这里有一个简单的列表：

数据元素：事实类型、维度

根据时间的数据记录

从源系统进行数据抽取

商业规则：属性、范围、领域、操作记录

你将不得不在很多不同的部门来收集这些需求。有两个基本的方法：（1）采访，一对一或者在一个小团体内；（2）联合应用程序设计（JAD）会议。

采访

- 同时对两个或三个人
- 方便排定日期表
- 在细节非常复杂的时候是一个好方法
- 需要有效的准备
- 总是要进行采访之前的调查
- 也会激励用户为采访作准备

群体会议

- 同时进行 20 人左右或者更少的群体
- 只能在对需求有基本了解之后使用
- 不适合最初的数据收集
- 对确认需求很有用
- 需要很好的组织工作

采访技巧

采访会议会花费较大一部分的项目时间。因此，需要很好的组织和管理。在项目团队采用采访的方法之前，确认已经完成了下面的这些主要任务。

- 选择并培训项目团队成员来进行采访

- 为每个项目成员分配合适的角色（主要采访者/记录员）
- 准备一个要采访人员列表，并制定日程表
- 列出每次采访的目标
- 完成采访前的调查
- 准备采访的问卷表
- 准备采访的用户
- 为将要采访的所有用户开一个启动会议

你要采访的大多数用户一般分为三类：高层主管、部门管理者/分析师，IT 部门专业人员。采访这些类别的人员你期望有什么收获，如图 5-7 所示。

高层主管

组织目标

衡量成功的原则

主要商业因素，当前的和未来的

确认问题

组织的视点和方向

对数据仓库的预期用途

部门管理者/分析师

部门目标

成功要素

限制成功的要素

主要的商业要素

产品和服务

分析需要的有用商业维度

对数据仓库的预期用途

IT 部门的专业人员

关键操作型源系统

当前信息传递处理

分析的类型

当前支持信息请求的信息技术

关于数据仓库的期望

图 5-7 采访的期望

采访前的调查对于采访的成功非常重要。这里列出了一些关键调查话题：

- 商业个体的历史和当前结构
- 雇员数和他们的角色和责任
- 用户的位置
- 企业中商业个体的主要目的
- 商业个体与企业战略之间的关系
- 商业个体的次要目的
- 商业个体和其它个体以及外部组织的关系
- 商业实体的收益和成本
- 公司的市场
- 市场上的竞争

这里有一些关于采访中问题类型的提示。

当前信息来源

哪一个操作型系统创造了与重要商业主题相关的数据？

支持这些主题的计算机系统是什么类型的？

在已有报表和在线查询中传递的是什么样的信息？

现有信息传递系统的细节程度是怎样的？

主题领域

对于分析最有价值的主题领域是什么？

商业维度是什么？有哪些层次？

制定决策的商业分区是什么？

不同的地区需要的是全球信息还是只需要当地信息来制定决策？还是两者的混合？

只在特定的区域供应特定的商品和服务吗？

关键性能指标

最近商业个体的性能是如何衡量的？

什么是关键的成功因素？它们是如何被控制的？

这些关键指标是如何出现的？

所有市场都是用这个方式衡量吗？

信息频率

决策支持所需的数据更新频率是怎样的？时间间隔是多少？

数据仓库中的信息需求的时间线是什么？

作为需求定义的早期文档，准备采访提纲可以包括这些内容：

1. 用户资料
2. 背景和目标
3. 信息需求
4. 分析需求
5. 当前使用的工具
6. 成功关键因素
7. 有用的商业指标
8. 相关商业维度

采用联合应用程序设计方法

如果你能够收集从很多数据源得到的大量基础数据，群体会议可能会是单独采访的很好的替代品。在这种方法中，你可以让很多有兴趣的用户在群体会议中聚在一起。这种需求收集方法会大大缩短所花费的时间，因而缩短整个项目。而且，如果用户处于远程状态，群体会议可能会更加有效。

联合应用程序设计（JAD）技术在二十世纪九十年代曾经非常成功地使用在为操作型系统收集需求方面。计算机系统的用户大量增长，他们在开发阶段的直接参与非常重要。

我们从名字就可以看出，联合应用程序设计是一个联合处理的过程，所有群体为了一个既定的目标聚在一起。它是一个关于联合开发计算机应用的方法论，用户和 IT 专业人员通过一种有组织的方式组合在一起。在合适的条件下，联合应用程序设计方法可以用来建设一个数据仓库。

联合应用程序设计包括五个阶段的步骤：

项目定义

完成高水平采访

实施管理采访

准备管理定义向导

调查

熟悉商业领域和系统

证明用户信息需求

证明商业过程

收集早期信息

准备会议日程

准备

为早期阶段创建工作文档

培训记录人员

准备可视化工具

实施会前讨论

选择一个会议地点

准备主题列表

联合应用程序设计会议

从对日程和目的的回顾开始

回顾最初的设想

回顾数据需求

回顾商业指标和维度

讨论维度层次

解决所有开放问题

通过行动项目列表结束会议

后期文档

转变工作文档

规划收集到的信息

列出所有数据源

辨认所有商业指标

列出所有商业维度和层次

收集并编辑文档

实施回顾会议

进行最后论证

建立改变需求的工作程序

使用联合应用程序设计方法的成功与否很大程度上依赖于联合应用程序设计团队的组成。团队的大小和组成会根据数据仓库的环境和目标有所变化。但是，典型的组成一般可以归纳成下面这样：

执行发起人——控制资金、提供方向和授权团队成员

协助者——在整个联合应用程序设计过程中协助团队的人

记录员——指定的记录所有决定的人员

全职参与者——每一个参与数据仓库决策制定的人

临时参与者——受项目影响的人，但是只在特定领域

观察者——不参与决策过程，但是列席会议的人

回顾已有的文档

虽然大多数收集的需求可以通过采访和群体会议得到，但是你还可以从已有的文档中得到有用的信息。对已有文档的回顾可以通过没有太多了解用户商业个体的团队成员完成。确定对已有文档的回顾工作只需要项目团队成员的参与。

来自用户部门的文档。你能从已有的文档中得到什么？首先，让我们来看看数据仓库的用户使用的报表和屏幕展示。你需要找出商业个体功能的所有相关东西，收集的操作型信息和用户使用的信息，什么对他们来说非常重要，他们是否使用任何已有的分析报表。你需要看看所有操作型系统使用的用户文档，需要掌握什么对用户来说是重要的。

商业个体通常有关于操作和流程的文档。用户如何提供这些功能？具体回顾这些操作和程序的细节。你试图找出用户感兴趣的分析类型，回顾这些文件然后扩大这些你从为采访会议准备的文档中学到的东西。

从 IT 得到的文件。从用户得到的文件和用户的采访中，你可以得到关于分析指标的信息和这些分析采用的商业维度。但是从哪里你能得到商业维度的指标数据呢？这些必须从内部操作型系统中得到。你需要知道这些源系统中能得到什么。

你能在这些源系统的什么地方得到需要的数据？所以操作型系统的数据库管理员和 IT

部门专家对于收集数据非常重要。数据库管理员可以提供所有的数据结构、独立数据元素、属性、价值范围、以及区域和数据结构之间的关系。从用户那里收集到的信息中，你可以将用户信息与源数据联系起来。

与数据库管理员合作，得到数据字典或者数据目录的复本。学习这些数据结构、数据区域和关系。最后，你将从这些源系统中得到数据仓库，所以你需要完全理解这些源数据、数据平台和操作系统。

现在让我们求助于 IT 应用专家。这些专家会给你商业规则，并帮助你理解从源系统中得到的不同的数据元素。你要了解这些数据所有者，对数据质量负责的人，以及源系统中如何收集和处理数据。回顾这些组成源系统的程序和模型，学习并了解这些程序中的数据结构。

需求定义：范围和内容

计算机系统项目中经常忽略正式的文档。项目团队经过需求定义的阶段，实施了访问和群体会议。他们回顾了已有文档，收集了足够多的资料来支持系统开发生命周期中下一个阶段的工作。但是他们忽略了需求分析的详细文档。

为什么你需要提交需求分析的结果？这里有几个原因。首先，需求分析文档是下一个阶段的基础。如果项目团队成员因为某些原因不得不离开团队，这个项目也不会因为缺少了这个成员的知识而遭受损失。正式的文档还可以帮助后来的阅读人验证你的工作。

我们接下来要讨论一个正式需求分析文档的建议要点。在此之前，让我们来看看这个文档需要包括的信息类型。

数据源

这类信息是需求定义文档的基础。必须包含所有你从源系统中收集到的细节信息。你将会在数据仓库中使用源系统的数据，将从这些源系统中收集、合并和整合数据，正确地转换这些数据，并在此基础上建立数据仓库系统。

需求定义文档一般说来应该包括下面的信息：

- 可获得的数据源
- 数据源的数据结构
- 数据源的位置
- 操作系统、网络、协议和客户机体系结构

- 数据抽取过程
- 历史数据的获得

数据转换

仅仅列出可能的数据源是不够的。因为数据仓库中可能存在数据的数据结构之间的关系，你必须还要列出相关的数据结构。一旦你列出了数据源，你需要决定这些源数据如何正确地转换成适合数据仓库存储的数据类型。

在需求定义文档中，要包括数据转换的细节，需要包括将源数据转换成数据仓库数据的方法，指示你的指标和商业维度数据从什么地方得到。描述在将数据载入数据仓库之前发生的合并、转化和分割的过程。

数据存储

根据你对用户的采访，你已经发现了数据仓库所需要的数据详细程度，对于用户需要的数据集市的数量也有一定了解。而且，你还知道指标和商业维度的细节。

当你了解了用户通常分析的类型时，你就能够决定数据仓库中存储的集合类型。这将会给你关于数据仓库存储需求的信息。

需求定义文档必须包括足够多的关于存储需求的信息。估计数据仓库究竟需要多少历史和现有的数据。

信息传递

需求定义文档必须包括下面列出的这些向用户传递信息的需求：

- 向下钻取分析
- 向上钻取分析
- 横向钻取分析
- 切片分析
- 特别查询报表

信息包图表

需求定义文档中的信息包图表的展现是操作型系统和数据仓库系统之间的主要和重要的不同。记住信息包图表是决定数据仓库需求的最好方法。

信息包图表明确了数据仓库的需求信息。它们包括了重要的衡量商业性能的指标，分析时采用的商业维度和如何钻取分析的细节。

需要花费一定的时间来确保信息包图表能够正确地完成。数据仓库的数据设计将完全依赖于信息包图表的正确性和充分性。

需求定义文档大纲

1. 绪论。明确项目的目的和范围。包括主要的项目论证。提供文档内容的概要。
2. 总体需求描述。描述源系统，包括采访的主要内容概要，明确数据仓库中需要什么类型的数据。
3. 特别需求。包括需要的源数据的细节。列出数据转换和存储的需求。描述用户需要的信息传递方法的类型。
4. 信息包。提供尽可能的详细的信息包。
5. 其它需求。包括了各种各样的需求，例如数据抽取频率、数据载入方法、和信息传递的位置。
6. 用户期望。明确用户在问题和机会方面的期望，阐明用户期望如何使用数据仓库。
7. 用户参与。列出用户在开发生命周期中希望参与的任务和行动。
8. 综合实施计划。在这个阶段中，给出一个高水准的实际计划。

本章小结

- 与操作性系统的需求不同，数据仓库的需求非常的模糊。
- 商业数据是有维度的，数据仓库的用户是按照商业维度来思考问题。
- 数据仓库的需求定义可以基于商业维度，例如产品、时间或促销等等。
- 信息包——一个新概念——是需求定义的主要部分。一个信息包记录了重要的指标或事实，和分析这些事实使用的商业维度。
- 采访和群体会议是收集需求的标准方法。

- 采访的主要人物或群体会议中的人员是高层管理者（包括发起人）、部门经理、商业分析师和操作型数据库管理员。
- 回顾所有的与操作型系统相关的已有文档。
- 需求定义文档的范围和内容包括：数据源、数据转换、数据存储、信息传递和信息包图表。

思考题

1. 操作型系统和数据仓库的需求定义有什么不同？
2. 解释商业维度。为什么说商业维度对数据仓库的需求定义非常有用？
3. 信息包中包含了什么数据？
4. 什么是维度层次？给出三个例子。
5. 解释商业的指标或事实，举出五个例子。
6. 列出在收集需求中需要采访的人员类型。你希望从他们那里得到什么样的数据？
7. 在什么样的情况下，联合应用程序设计方法论能够帮助成功的收集需求？
8. 为什么要回顾已有的文档？你希望从中得到什么？
9. 举出任何五种正式需求定义文档的内容部分。描述每一个部分的主要内容。

复习题

1. 判断下面说法的对或错：
   A. 一个销售处理操作型系统和销售分析数据仓库的需求定义是非常相似的
   B. 管理者是从分析商业维度的方面来思考的。
   C. 个体销售和产品费用是一种商业维度的例子。
   D. 维度层次与钻取分析是相关的。
   E. 范畴是商业维度的属性。
   F. 联合应用程序设计是一对一采访的方法论。
   G. 实施采访前的调查并不是总是需要。
   H. 部门用户提供公司总体方向的信息。
   I. 部门管理者是提供操作型系统数据结构信息的好来源。
   J. 信息包图表是正式需求定义文档的重要部分。

2. 作为一个有三个产品工厂的全国范围应用制造公司的销售副总裁。描述任何三个不同的分析销售情况的方法。你进行分析的商业维度是什么？
3. BigBook 公司是一家大型图书销售商，有国内和国际的销售渠道。公司从出版商那里订货，然后向所有图书销售商分发。开始的时候，你想建立一个数据仓库来分析公司大量仓库的出货情况。确定指标和事实，以及商业维度。准备一个信息包图表。
4. AuctionsPlus.com 是一个互联网拍卖公司，销售高层次的艺术品。你正在这个公司的数据仓库项目中，你的责任是收集销售分析的需求。找出关键的指标、商业维度、层次和范畴。画出信息包图表。
5. 写出一个数据仓库正式需求定义文档的详细大纲，来分析一家大型连锁商店的产品收益率。

第六章　需求——数据仓库的驱动力

本章目标

- 了解为什么说商业需求是驱动力
- 讨论需求是如何驱动每一个开发阶段
- 学习需求是如何影响数据设计的
- 回顾体系结构方面的需求
- 注意 ETL 和元数据的特殊要求
- 检查信息传递是如何符合需求的要求的

在前面的章节中，我们详细讨论了需求定义阶段的内容。你已经学习了为一个数据仓库收集信息与操作型系统的需求定义是不同的。我们还了解了一个新的概念——信息包来表达需求。最后，我们将这些综合在一起建立了需求定义文档。

当你在设计和开发任何系统的时候，很明显，这些系统都一定要反映用户在商业处理上的需要。他们需要正确的 GUI 显示，系统需要有正确的逻辑来反映功能，用户必须能够得到需要的输出和报表。需求定义将会知道整个系统设计和开发的过程。

数据仓库的需求定义是怎样的呢？如果对于任何操作型系统来说，正确的需求定义非常重要，那么对于数据仓库来说就更加重要。为什么？数据仓库环境是一个信息传递的系统，用户可以自己连接数据仓库获得他们想要的输出结果。在操作型系统中，你为用户提供的是预先设计好的输出形式。

所以说数据仓库使用最合适的格式来包含正确的元素是非常重要的。你的用户必须能够找到他们所需要的所有统计信息，必须能够方便的连接数据仓库、运行查询、得到结果、并且毫无阻碍地进行结果的分析。

在数据仓库中，用户的商业需求组成了唯一的也是最重要的驱动力。数据仓库开发中的每一个阶段的每一项任务都是由需求决定的。在设计阶段进行的每一个决策，包括数据设计、体系结构设计、基本结构的配置、或者是信息传递方法的安排，都完全受到需求的影响。图 6-1 中表现了这一基本规律。

商业需求

规划和管理　　实施　　维护

设计（体系、结构　　数据获取、数据存储、信息传递）

建设（体系、结构　　数据获取、数据存储、信息传递）

图 6-1　商业需求是驱动力

因为需求是开发过程中每一个阶段的主要驱动力，你需要确保你的需求定义包括了支持每个阶段的所有细节信息。在这一章中，列出了一些重要的开发步骤，特别是需求如何引导、影响这些开发步骤的。为什么这些内容是必需的呢？当你在收集商业需求和编写需求定义文档的时候，你必须随时提醒自己这个阶段项目中哪些内容对其它阶段有重大作用。你的需求定义必须驱动项目的每一个阶段，所以你必须特别重视这一内容。

数据设计

在数据设计阶段，你将为下面这些内容提出数据模型：

- 转换、清洗和整合从源系统中得到的数据，准备将这些数据载入数据仓库。
- 数据仓库的存储本身

如果你采用建造数据仓库的那种实用方法，作为建立数据集市的综合方法，你的数据模型就要包括你第一次建立的数据集市的维度数据模型。另一方面，你的公司可能会决定建立第一个企业级数据仓库，其中要采用最初的数据集市。在这个例子中，你的数据模型必须既包括大型数据仓库的模型，也包括当初数据集市的模型。

这些数据模型将组成数据仓库的物理设计和执行的蓝图。你将使用这些模型来与团队成员进行交流，包括数据仓库可以得到什么数据元素，以及他们如何进行配合。你要在用户中推广这些数据模型，让他们了解数据内容和数据关系。单个数据集市的数据模型在用户交流中起到了强有力的作用。

需求定义中的哪个部分驱动了数据设计呢？为了了解这个问题，如图 6-2 所示，我们来设想一个金字塔结构的数据内容的模型。这个金字塔的下半部是企业级数据仓库的数据模型，金字塔的上半部分是数据集市的维度数据模型。你需要哪些需求定义来完成这个金字塔呢？需要两个基本的信息类型：源系统数据模型和信息包图表。

金字塔的下半部分需要当前源系统的数据模型。因此，确保你的需求定义文档包含有足够的关于组成部分和源系统数据关系的信息。在前一章中，我们已经讨论了信息包图表的内

容。请特别注意信息包图表，它是实际商业需求所反映的需求定义的一部分。否则，你的数据模型就不能真正反映用户想要在数据仓库中看到的内容。

商业维度的结构

在数据集市的数据模型中，必须突出反映用户分析商业指标时使用的商业维度。在上一章中，当我们讨论信息包图表的时候，我们看了一些例子。在一个信息包图表中，商业维度是作为列标题列出的。例如，如图 6-3 所示，这是一个汽车销售的商业维度，是我们在上一章图 5-5 的一部分。

如果你为这个数据集市创造了一个数据模型，图中列出的商业维度必须要包含在这个模型中。数据集市的有用程度与数据模型的正确程度直接相关。因此数据包图表中的合适的维度和正确的内容有非常重要的作用。

关键衡量指标的结构

关键衡量指标是商业分析和控制使用的指标或量度。用户通过关键衡量指标来衡量系统的性能。对于上面的汽车销售的例子，关键衡量指标包括实际销售价格、MSRP 销售价格、可选价格、完全价格等等。用户通过这些关键衡量指标来衡量他们的成功。他们可以对这样的指标进行计算和加和等操作。

除了基于这些维度可以得到查询结果，这些事实和维度可以用来进行分析。当你的用户通过产品、时间、位置等维度分析销售情况的时候，他们可以得到通过这些指标展现的结果，例如销售个体、收益、成本和盈利率。为了让用户通过合适的关键衡量指标回顾这些结果，你必须将信息包图表作为需求定义的一部分，其中包括了所有相关的关键衡量指标。

商业维度和关键衡量指标组成了维度数据模型的主要部分。数据模型的结构与商业维度的数量直接相关。每个商业维度的数据内容组成了数据模型的各个部分。例如，如果一个信息包图表将产品、客户、时间和位置作为商业维度，这四个维就是数据模型结构的四个独立部分。除了商业维度，关键衡量指标的集合也组成了数据模型的其它独立部分。

详细程度

数据模型中还需要反映什么内容？为了回答这个问题，让我们仔细看看你的用户计划如

何使用数据仓库来进行分析。首先来举一个例子。高级主管想分析不同区域的销售情况。首先他先看看今年全国范围某一产品的销售情况。然后下一步是看看这一年中各个不同地区这个产品的销售情况。接着，是按照季度进行分析。在此之后，用户可能想要将得到的结果与预算和上年情况进行比较。

我们可以看到，在这种类型的分析中，你需要提供向下钻取和向上钻取的功能。你还需要提供更低一个级别的数据吗？如果是这样，当你的用户想要看整年的全国数据的时候，系统必须在分析的同时进行聚合的计算。另一方面，你是否必须为在最低层次展现数据而保存细节，而且在展现高等级数据时进行聚合？

这种讨论为我们带来了需求定义与数据模型相关的另一个方面。如果你在数据仓库中需要总结，那么你的数据模型必须包括能够进行总结的结构。如果你可以在分析过程中的空闲时间让系统进行加和操作，那么你的数据模型就不需要有总结的结构。找出基本的钻取功能并包含足够的关于加和类型以及详细程度的细节。

体系结构规划

你知道数据仓库体系结构是指能够带来最大利益的体系部分的合理安排。那么如何来规划数据仓库的体系结构呢？基本上，每一个数据仓库都有很多相同的组成部分。因此，当你进行体系结构规划的时候，你不需要为你的数据仓库发明新的组成部分。你真正要做的是为你的环境设计每一个组成部分的规模大小，规划所有的部分如何组合在一起使他们能够作为一个整合的系统。

在我们进一步讨论之前，我们首先要看看一些主要的体系结构部分：

- 源数据
  - 生产数据
  - 内部数据
  - 存档数据
  - 外部数据
- 数据准备
  - 数据抽取
  - 数据转换
  - 数据载入

- 数据存储
- 信息传递
- 元数据
- 管理和控制

在为数据仓库规划整体体系结构的时候，你要确定每一个部分的范围和内容。例如，在公司中所有的源数据可能恰好在一个计算机平台上，而且在同一个关系数据库中。如果是这样，那么数据抽取部分就不需要很大的规模。此外，如果你的公司决定使用数据库管理系统为元数据存储提供的服务，例如别名定义和注释，那么你的元数据部分就会很简单。

因此，规划体系结构包括回顾每一个特定环境中的部分，并确定其参数。而且，还包括这些部分之间的接口。管理和控制模块如何控制这些不同的组成部分呢？进行这样的规划需要哪些信息？如何规划每一个部分并提供适当的基本设施结构来支持它们？当然，答案是商业需求。你在规划过程中需要的所有信息都必须来自需求定义。在接下来的章节中，我们来看看商业需求在结构规划中的重要性。我们要针对每一个部分回顾正确的需求是如何驱动数据仓库的规模大小和内容的。

组成部分的构成

让我们一起来回顾每一个组成部分，确定规划数据仓库体系结构究竟需要什么样的需求定义。记住规划体系结构包括决定每一个部分的规模大小和内容。在下面的列表中，列出了在需求定义中必须包含的信息类型。

源数据

- 操作型源系统
- 计算机平台，操作系统，数据库，文件
- 部门数据，例如文件、文档和电子表格
- 外部数据源

数据准备

- 数据源和准备阶段数据结构之间的数据规划
- 数据转换
- 数据清洗
- 数据整合

数据存储

- 抽取和整合数据的大小
- 数据库管理系统的特征
- 成长潜力
- 集中或分布

信息传递

- 用户的类型和数量
- 查询和报表的类型
- 分析的类型
- 决策支持系统的应用

元数据

- 操作型元数据
- ETL（数据抽取/转换/载入）元数据
- 用户端元数据
- 元数据存储

管理和控制

- 数据载入
- 外部源
- 预警系统
- 用户端信息传递

图 6-4 提供了一个需求驱动的体系结构组成部分的有用总结，指出了商业需求对数据仓库体系结构的影响。

特别考虑的问题

我们已经学习了需求对于体系结构组成部分的影响，现在我们来了解一些值得特别考虑的问题和功能。我们需要考虑这些问题，是因为如果在需求定义中忽略了这些问题，可能会发生很严重的后果。当你在需求定义阶段的时候，你必须特别注意这些问题。

数据抽取/转换/载入（ETL）。数据仓库中与 ETL 相关的工作是最耗时的，并且很受人为因素干扰。在建立体系结构的时候，对需求的关注以及这些工作都需要很长一段路要走。让

我们分别来看这三个步骤。

数据抽取。需要确认所有内部数据源。特别是所有计算机平台和数据抽取的所有源文件。如果你还有外部数据源，要通过这些外部数据源来决定数据结构的兼容性。而且要指出数据抽取的方法。

数据转换。在数据规划和为数据载入做准备之前，需要很多类型的转换功能。这些功能包括输入选择、输入结构分离、标准化和源结构的反向标准化、聚合、转换、缺失值的补充、名称和地址的转化。在实际中，这些将会是一个很长而且复杂的功能列表。检查每一个计划存入数据仓库的数据元素，确保映射和转换的过程。

数据载入。定义最初的载入。决定每一个主要数据集合在数据仓库中的更新频次。如果每天更新需要多少成本？你的环境需要每天超过一次以上的更新吗？从源系统中可以得到哪些变化？定义如何在每天、每星期和每月进行更新。

数据质量。坏的数据会导致坏的决策。无论你将数据仓库建设的如何好，也无论你提供多么完美的查询和分析功能，如果数据仓库的数据质量令人怀疑，用户就会很快丧失对数据仓库的信任。即使很小的差异，在决策制定过程中都会导致严重后果。数据仓库的数据质量是非常重要的。因此，在需求定义的早期就要辨认在源系统中可能存在的数据污染。而且，要注意所有可能存在的数据质量问题。请注意下面的提示。

数据污染源

- 系统转换和迁移
- 异类系统整合
- 源系统不正确的数据库设计
- 数据寿命
- 客户的不完全信息
- 输入错误
- 系统的国际化/本地化
- 缺乏数据管理制度或程序

数据质量问题的类型

- 源系统的虚值
- 源系统缺乏数据
- 多目标的字段
- 含糊的数据

- 互相矛盾的数据
- 名称和地址的不正确用法
- 违背商业规则
- 重复使用的主键
- 非唯一的标识符

元数据。你已经了解了数据仓库的元数据不仅仅是数据字典的条目。数据仓库的元数据不仅仅是数据字典或数据目录的细节，而是作为连接所有组成部分的粘合剂。当数据从一个组成部分迁移到另一个的时候，这种运动就受到一部分元数据的控制。当用户查询数据仓库时，元数据作为信息源将查询参数和数据库部件联系起来。

在前面的内容中，我们已经将数据仓库中的元数据分为了三类：操作型，数据抽取和转换，用户端。图 6-5 中表现了商业需求对于元数据体系结构组成部分的影响。

我们不需要再重申元数据部分的重要作用。将这些知识应用到你的数据仓库项目中。对于每一种类型的元数据，在你的需求定义中指出它们究竟需要多少细节。你必须有足够的细节来支持重要的决策，例如选择元数据仓库的类型，决定元数据存储采用集中式还是分布式。

工具和产品

当我们在数据仓库上下文中提到工具的时候，你可能只想到了用户端工具。很多人都是这样的。但是对于建造和维护数据仓库来说，你需要很多类型的工具来支持体系结构中的多种组成部分。

我们在讨论需求给数据仓库体系结构的影响后，下面来看看工具和产品的部分。首先，需求不会直接影响工具的选择。不要基于需求来选择工具，应该根据工具的要求来调整体系结构。设计数据仓库的体系结构，然后寻找合适的工具来支持这个体系结构。也许一种工具非常适合某一个数据仓库的功能，但是在另一个数据仓库中就完全不适合。这是因为它们的体系结构是不一样的。虽然在数据仓库体系结构中的组成部分大体上是相同的，但是它们的范围、大小、内容和结构都是不一样的。

其次，当为设计体系结构而收集需求的时候，有时候你会感觉很难使体系结构适应需求的要求。你可能会认为你无法设计出需求所需要的那种体系结构，这是因为可能很难找到支持那种体系结构的合适的工具。请注意市场上提供了大量的工具。我们需要指出一旦完成了体系结构设计的工作，你就可以找到最适合的第三方工具和产品。

总之，这些工具需要提供以下这些功能：

- 数据抽取和转换

中间键

数据抽取

数据转换

数据质量保证

创造载入映像

- 仓库存储

数据集市

元数据

- 信息获取/传递

报表生成

查询处理

OLAP

预警系统

决策支持应用

数据挖掘

数据存储规范

如果你的公司正在采用自上而下的方法来开发数据仓库，那么你必须为以下这些部分定义存储规范：

- 数据准备阶段
- 整体企业数据仓库
- 每一个独立的数据集市
- 任何 OLAP 需要的多维数据库

另一方面，如果你的公司采用的是自下而上的方法，你需要在这些部分定义存储规范：

- 数据准备阶段
- 每一个相关的数据集市
- 任何 OLAP 需要的多维数据库

一般来说，整体企业数据仓库会基于一个关系数据库管理系统（RDBMS）支持的关系模型。数据集市的结构一般根据这个关系数据库管理系统提供的维度模型。很多厂商提供了大量的多维数据库系统（MDDBs）。你的多维数据库系统的规范会基于你所选择的厂商。数据准备阶段的范围依赖于数据转换、清洗和转变过程的复杂程度和影响范围。需要准备的数据可能是一些文档，或是一个完整的已经开发完成的关系数据库。

无论你选择什么样的数据库管理系统，这个系统都必须和前、后端的其它工具互相影响。后端的工具是数据转换、清洗和载入过程的产品，前端工具是和用户的信息传递相关的工具。如果你想找到最适合你的环境的工具，这个选择可能会造成数据库产品来自于不同的厂商。那么，一个重要的数据库管理系统原则就是，系统必须是开放的。它必须和前、后端产品工具是相兼容的。

所以，我们谈到的商业需求对数据存储规范的影响是什么呢？商业需求决定了数据库系统的健壮性和开放性。在定义需求的时候，要注意需求对数据存储规范的影响，收集所有关于前、后端体系结构组成部分的有关信息。

我们接下来要看看商业需求对于选择数据库管理系统以及评估数据仓库存储的影响。

数据库管理系统的选择

在需求定义阶段，当你在采访用户并与他们开最后一次会议的时候，你一般不会特意讨论选择数据库管理系统的类型。但是，很多用户需求会影响对数据库管理系统的正确选择。市场上的关系型数据库管理系统产品通常是查询处理、报表产生和外部接口等很多部分的捆绑。对数据库管理系统的选择可能会受到产品中某个组成部分的影响。而且商业需求可能会决定需要这些工具中组成部分的类型。总之，接下来的这些关于商业需求的要素会影响数据库管理系统的选择：

用户的经验程度。如果用户对于数据库系统完全没有经验，数据库管理系统就必须有管理和控制失控查询的特性。另一方面，如果有很多的高级用户，那么他们就会要求自己定制查询方法。既然这样，数据库管理系统必须支持一个简单的 SQL 语言接口。

查询的类型。如果大部分的查询都很复杂而且需要得到大型的结果集合，那么数据库管理系统必须有一个强有力的优化功能。相反，如果是一些简单查询和复杂查询的混合体，那么数据库软件中必须有很多种的查询管理来平衡这些查询的执行。

开放性。开放性的程度取决于前、后端体系结构的组成部分，以及决定于商业需求。

数据载入。数据量和载入频率决定了数据载入和更新阶段的工作强度。

元数据管理。如果你的元数据部分不需要非常精细，那么一个拥有主动数据字典的数据库管理系统就够了。在你的需求定义中反映元数据结构的类型和范围。

数据存储位置。数据仓库准备采用集中式存储，还是分布式？这个问题的答案会决定选择数据库管理系统是否必须支持分布式数据库。

数据仓库的成长。你的商业需求定义必须包含对数据仓库未来用户数量、查询数量和复杂程度等问题的估计信息。这种成长估计将与所选数据库管理系统如何支持可扩缩性有直接的关系。

存储规模估计

你的数据仓库有多大？所有数据的存储需要多大的空间？这些问题的答案会影响存储介质的类型和大小。你如何能找到这些问题的答案呢？同样，需要从商业需求中寻找答案。在需求定义中，你必须有足够的信息来回答这些问题。

让我们来总结一下。你需要在需求定义阶段，估计下列这些部分的存储大小：

数据准备阶段。根据每一个商业主题的源系统数据结构的大小，估计并计算整个企业数据仓库的存储大小。对于数据集市，首先基于第一个数据集市的商业维度和指标估计数据准备阶段的存储。

整体企业数据仓库。根据每一个商业主题的数据结构，估计存储大小。你知道数据仓库中的数据是根据商业主题进行存储的。对于每一个商业主题，列出不同的属性，估计它们的域长度，并且计算每个主题需要的存储大小。

独立的或者相联系的数据集市。在定义需求的时候，我们创造了信息图表。一组这样的图表组成了一个数据集市。每一个信息图表包含了商业维度和它们的属性，还包括了分析需要的商业指标。使用这些信息图表中商业维度和指标的细节，估计数据集市的存储大小。从你的第一个数据集市开始。

多维数据库。这些数据库支持 OLAP 或多维分析。你的用户需要进行多少在线分析处理（OLAP）？企业级数据仓库或者数据集市提供了多维数据库的数据。根据用户的需要设计 OLAP 的细节，然后利用这些细节来估计这些多维数据库的大小。

信息传递策略

商业需求对于数据仓库信息传递机制的影响是非常直接的。在需求定义的过程中，用户告诉你他们希望从数据仓库中得到什么样的信息。你将这些需求记录在需求定义文档中。然后将所有这些特性和内容反映在信息传递组成部分中。这些工作听上去很简单而直接吗？虽然它们看上去直接而简单，但是还需要我们注意一些问题。需求中很多不同的部分使用不同的方法影响信息传递组成部分的不同元素。

用户使用数据仓库的方法影响了信息传递策略。大多数的用户都是高级用户和分析师吗？那么信息策略必须使之能够提供有效的分析工具。很多用户希望得到预制的报表和查询方式吗？那么就必须加强信息传递部分的查询和报表功能。

商业需求直接影响的信息传递部分包括：

- 查询和报表
- 分析类型
- 信息发布
- 决策支持应用
- 成长和扩大

图 6-6 中表现了信息传递中商业需求的影响。

数据仓库存在的唯一原因就是向用户提供战略信息。信息传递是整个体系结构组成部分的最上一次。其它的组成部分对用户都是透明的，但是他们看到和感觉到的就是信息传递部分给他们提供的内容。所以我们要非常重视与信息传递相关的商业需求的重要性。

接下来的部分中，包括了一些关于需求定义的有价值的提示，为了强调信息传递部分的有效和有用性。请仔细学习下面的内容。

查询和报表

什么样的用户会希望使用预先定制好的查询和报表呢？在规范中写入这部分的内容。而且，在规范中还要写入这些报表的产生和分发频率。究竟有多少用户会使用这些预制的查询？多久会使用一次？

第二种查询类型就是没有预先定制的查询。既然如此，用户定义他们自己的查询，并且自己在需要的时候运行这些查询。而且，用户自己提供报表需要的参数然后生成自己的复杂

报表。收集足够多的关于这种查询和报表的细节资料。

高级用户可能会进行一些复杂查询，大部分的时间会使用互动的分析。除了进行分析，你的高级用户需要进行独立复杂查询的能力吗？

分析的类型

大多数的数据仓库提供了一些关于运行互动操作和复杂数据分析的特性。使用向下钻取和向上钻取的分析方法是很常见的。回顾用户可能使用的所有分析类型，得到分析类型的可能复杂性的信息。

除了数据集市直接提供的分析，多数当前的数据仓库环境向用户提供了 OLAP 的环境。使用 OLAP 功能，用户可以进行多维分析，从多角度观察多维数据库中数据。这种分析的类型称为切片分析。估计需要提供的钻取功能的性能和范围。决定需要提供什么程度的钻取功能。

信息发布

有一些特定的程序来支持用户的特定要求。一个执行信息系统向高级经理们提供了决策支持。一个数据挖掘应用程序能够发现数据中隐藏的关系类型和预测可能性。我们将在第 17 章中详细讨论数据挖掘的有关知识。

数据仓库提供了这些决策支持应用的数据。有时候这些特别查询应用的设计和开发是在数据仓库项目的范围之外的。它们与数据仓库之间的唯一联系就是从数据仓库中读取数据。

无论你的公司关于特别决策支持应用的开发策略是什么，确保需求定义中已经包含了这些策略的细节。如果数据仓库只是用来提供数据，那么一定要定义数据源以及数据迁移的频率。

成长和扩展

让我们来看看数据仓库的实施过程。你已经向你的用户提供了运行查询、打印报表、执行分析、使用 OLAP 进行复杂查询和向复杂程序提供数据的功能。信息传递部分已经完成，而且工作情况良好。那么所有的工作是否都完成了呢？

信息传递部分会一直继续成长和扩展，查询和报表的数量和复杂程度方面都会不停的扩

张。体系结构的每个部分都需要增强。在最初的需求定义中，你需要加入这些成长和扩展的考虑。关于成长和扩展的足够多的细节能够影响信息传递部分的正确设计，所以收集足够多的细节来估计成长和扩展的情况。

本章小结

- 数据仓库项目中正确的需求定义非常重要，清晰的了解商业需求在每一个开发阶段的影响。
- 商业需求决定了数据设计阶段的输出。
- 数据仓库体系结构中的每一个部分都受到商业需求的影响。
- 为了提供数据质量，需要了解数据污染源、质量问题的主要类型，以及在需求定义阶段的早期排除数据污染的可能性。
- 数据存储规范，特别是数据库管理系统的选择，是由商业需求决定的。确保你在需求定义阶段收集到足够的相关细节。
- 商业需求在很大程度上影响信息传递机制。需求定义了用户从数据仓库中得到信息的方式、时间和位置。

思考题

1. “在数据仓库中，用户的商业需求是唯一重要的驱动力。”你同意这句话吗？如果你同意，举出四个理由。如果不同意，那么什么是数据仓库的驱动力？
2. 正确的信息图表是如何成为数据集市的数据模型的？简单解释。
3. 举出商业需求影响的五种体系结构组成部分。解释这些影响。
4. 需求对于厂商提供的工具和产品有怎样的影响？需求直接影响工具的选择吗？
5. 列出直接受到商业需求影响的信息传递的任意四个方面。对其中的两个方面进行解释。
6. 商业需求是如何影响数据库管理系统的选择的？描述任意三种方法。
7. 什么是 MDDBs？什么类型的商业需求决定了数据仓库中 MDDBs 的用法？
8. 需求是如何影响元数据框架的选择的？简单解释。
9. 什么类型的用户需求表示了数据仓库的粒度或细节水平？
10. 你如何估计存储大小？什么因素影响这种大小？

复习题

1. 连线：

1.信息包图表	A.决定数据抽取
2．钻取的需要	B.提供 OLAP
3．数据转换	C.提供数据
4．数据源	D.载入管理的影响
5．数据寿命	E. 数据库管理系统的查询管理
6．复杂查询	F. 数据的低层次
7．简单和复杂查询	G. 大型准备阶段
8．数据项	H. 影响数据设计
9．特别决策支持系统	I 可能的污染源
10. 企业级数据仓库	J. 数据准备设计

2. 公司中的源系统的数据质量非常糟糕。你在项目团队中，是数据质量保证专家。描述你将在需求定义文档中包括哪些细节来提出数据质量的问题。
3. 作为一个负责数据载入和更新的专家，描述你在需求定义阶段需要收集的所有的细节信息。
4. 你一家销售连锁公司的数据仓库项目的项目经理，这个数据仓库面对的是全国所有商店和所有用户。你如何保证在需求阶段中收集到决定数据库管理系统所需要的所有细节信息？给直接负责协调需求定义阶段的高级分析师写一个备忘录。
5. 你是一家制造型公司数据仓库项目中的查询工具专家，这个数据仓库面对的是一些主要办公室的主要用户。这些高级用户需要分析用的复杂工具。你如何决定需要使用的信息传递方法的类型？在需求定义阶段需要收集什么类型的细节？

第七章　体系结构及其组成部分

本章目标

- 了解数据仓库的体系结构
- 了解体系结构的组成部分
- 回顾数据仓库体系结构的有区别特性
- 检查体系结构如何支持数据的流动
- 理解技术性体系结构
- 学习体系结构的组成部分的功能和服务

了解数据仓库的体系结构

在第二章中，我们已经学习了数据仓库的组成部分，我们简单地了解了这些部分的名称。在第六章中，我们重新看了数据仓库的体系结构，并且了解商业需求是所有设计和开发的驱动力，包括体系结构的规划。

在本章中，我们要从另一个视角学习数据仓库的体系结构。你将会学到体系结构的各个组成部分是如何使数据从源系统向最终用户流动的。然后，你会看到体系结构的每一个部分，检查每个部分的功能、程序和特点。这个部分的讨论将会使你了解这些部分的技术性体系结构。

体系结构：定义

将数据仓库的所有部分结合在一起的结构，就是体系结构。例如，我们来看一个学校建筑的例子。这个建筑物的体系结构不仅仅是外表风格，也包括了不同的教室、办公室、图书馆、走廊、体育房、门、窗户、屋顶和大量的其它部分。当所有这些部分组合到一起，这种结构就是学校建筑的体系结构。将这个概念扩展到数据仓库中，数据仓库的不同部分组合在一起就组成了数据仓库的体系结构。

我们在建设这个学校建筑的时候，假设要求建筑者将教室做得大一点。结果他们就将教室的面积做大，但是会同时去掉所有的办公室，如果这样，这个学校建筑就是一个不完善的

体系结构。出了什么问题呢？首先，没有包括所有需要的部分。剩下部分的分配可能也是不对的。正确的体系结构对于数据仓库的成功是非常重要的。因此，在本章中，我们来进一步地看看数据仓库的体系结构。

在数据仓库中，体系结构包括大量的因素。简单来说，包括整合的数据，这是最核心的部分。体系结构包括准备和存储数据所需要的所有部分。另一方面，还包括所有数据仓库建设中的传递信息。体系结构还是支持数据仓库工作和满足商业需求的规则、程序和功能的组合。最后，体系结构是由数据仓库中的技术所组成的集合。

什么是数据仓库体系结构的主要目的？体系结构提供了开发和实施数据仓库的全部框架结构；它是一个全面的蓝图。体系结构定义了标准、衡量指标、通用设计和支持的技术。

三个主要区域的体系结构

我们已经了解了，数据仓库的三个主要区域是：

- 数据获取
- 数据存储
- 信息传递

在第二章中，我们已经了解了下面这些主要的数据仓库的组成部分：

- 源数据
- 数据准备
- 数据存储
- 信息传递
- 元数据
- 管理和控制

图 7-1 中是这三个区域的主要体系结构部分。在本章中，我们要来学习与这三个区域相关的体系结构。每一个部分都有明确的功能，提供特定的服务。我们来仔细看看这些功能和服务，以及这三个区域的底层技术体系结构。

因为这些体系结构组成部分的重要性，你还可以在后面的章节中学到更多的这方面知识的细节。现在，在这三个数据仓库区域中，让我们主要来看看这三个主要区域的功能、服务和技术体系结构，如图 7-1。

有区别的特性

作为一个 IT 专业人员，当你在开发一个 OLTP 系统的时候，例如订单处理、投资控制或销售报表等系统，你考虑了每一个系统的体系结构吗？虽然在这些操作型系统中通常不会总是提到体系结构，但是这些系统中都是存在一个基础的体系结构。例如，这样一个系统的体系结构可能包括文件转换、数据库的最初内容、数据输入的方法、在线展现的信息传递、以及在线和批处理报表。但是对于这样的系统，我们不会很详细地涉及到体系结构方面的问题。如果操作型系统是这样的，那么数据仓库有什么不同和区别吗？

数据仓库的体系结构是复杂、扩展和宽泛的。在数据仓库中，体系结构由截然不同的部分组成。体系结构有与众不同的特性，值得我们自己的探讨。在讨论这些体系结构之前，让我们先来回顾一下数据仓库体系结构的特性。

不同的目标和范围

体系结构必须能够满足提供战略信息的需求。战略信息与从操作型系统中获得的信息有很大的不同。当你通过操作型系统提供信息的时候，每一个用户群体的信息内容和数量都是有限的。举一个例子，在一个特定时刻，用户只对某一个用户和所有相关订单的信息感兴趣。但是，从数据仓库中，用户就会对获得大型的结果集合感兴趣。数据仓库中的结果集合可以是一年中根据季度、产品和销售区域划分的全年的所有销售情况。

因此，数据仓库体系结构必须包括能够向用户提供大量数据集合的部分。简单地说，与传统的操作型系统不同，数据仓库必须有一个不同的和更加精细的体系结构。

定义数据仓库的范围也是非常困难的。你如何定义一个操作型系统的范围？你需要考虑的问题可能是用户群体、功能范围、数据存储和输出形式。对于数据仓库的体系结构来说，在定义范围的时候需要考虑的因素有哪些呢？

首先，你必须考虑数据源的数量和范围。你要从几个源系统中抽取数据？有哪些外部源系统？你正在规划中包括部门文档、电子表格和私人数据库？包括存档数据吗？体系结构的范围可能通过数据转换和整合功能来衡量。在数据仓库中，数据粒度和数据容量都是很重要的考虑因素。

数据仓库对已有操作型系统的影响也需要进行重点考虑。因为数据转换、兼容和协调等问题，你必须决定数据仓库会对操作型系统造成多大的负面影响。什么时候运行你的批处理

抽取，它们将会如何影响生产系统？

数据内容

数据仓库中的只读数据在整个体系结构中作为主要的部分，处于中间位置。在一个操作型系统中，虽然数据库很重要，但是相对来说数据仓库的数据存储更加重要。在数据作为只读数据存入数据仓库之前，要运行大量的功能。这些重要的功能与操作型系统中的数据转换，不是一个层次上的概念。

在数据仓库中，你要整合大量数据源的数据。在抽取这些数据之后，你需要在数据准备区域中对这些数据进行转换、清洗和整合。然后你才能将这些整合的数据作为只读数据，载入到数据仓库中。操作型数据不是只读数据。

而且，数据仓库的体系结构必须支持根据商业主题，而不是根据应用进行分类的数据存储。数据仓库中的数据不能仅仅包含这些变量的当前值。这与大多数操作型系统有很大的不同。

当我们提到数据仓库中存储的历史数据时，我们是指大量的数据。很多公司选择在数据仓库中存储过去 10 年的数据。如果可能，一些公司甚至希望存储更早时期的数据。这是另一个关于为什么数据仓库体系结构必须支持大数据量的原因。

复杂分析和快速响应

你的数据仓库体系结构必须支持用户对战略信息的复杂分析。操作型系统中的信息检索操作相对于数据仓库来说，复杂性降低了很多。大多数在线信息检索是互动分析。用户不会运行一个单独的查询，通常用户会进行连续的查询，而且会持续一定的时间，因为用户通常首先在一个高层次运行一个查询，然后从数据仓库中得到反馈的一个结果，接着根据这个结果进行下一个层次的查询，一直继续下去。

因此，你的数据仓库体系结构必须能够支持提供分析的变化。用户必须可以进行钻取、切片和运行“what-if”应用。用户必须能够从不同的输出方式中得到结果。用户不再满足于文本或表格式的结果。每一个表格格式的结果都必须转化成图形图表。

提供战略信息是指制定快速决策和快速处理问题。例如，假如你是主管市场的副总裁，希望快速的调查某个地区连续三个星期销售额下降的原因，并且制定解决这个问题的方案。你的数据仓库必须提供分析工具和解决这个问题的足够信息。

数据仓库体系结构必须使快速制定战略决策更加容易。在体系结构中必须有适当的组成部分，通过数据仓库提供的信息，来帮助和支持用户要求的快速响应。

灵活性和动态性

在数据仓库的设计和开发阶段，还不知道所有的商业需求。使用信息包的技术，你可以了解大多数的需求，并对数据需求进行维度建模。然而，没有提到的需求会在用户开始使用数据仓库之后显现出来。你必须确保数据仓库的体系结构足够灵活来适应额外的需求。

额外的需求包括了商业需求中没有提到的部分。而且，商业条件本身是在变化着的。事实上，它们一直在变化。变化着的商业条件需要数据仓库中包括额外的商业需求。如果数据仓库的体系结构设计中考虑了灵活性和动态性，那么数据仓库就可以满足额外的需求。

元数据驱动

在数据作为有用的战略信息从源系统向最终用户流动过程中，元数据围绕了整个的活动过程。体系结构的元数据部分包括了关于活动每一个阶段的数据，并且事实上正是元数据使整个过程运动起来。

在操作型系统中，没有一个部分像元数据在数据仓库的位置这样重要。数据库管理系统的数据字典仅仅是数据仓库的元数据部分的微弱影子。所以，在数据仓库体系结构中，元数据部分与其它部分交织在一起，并将它们连接起来。数据仓库中的元数据是如此的重要，我们将在第九章中详细讨论。

体系结构框架

在本章前面的部分中，我们已经了解了数据仓库的三个主要的区域，它们分别是数据获取、数据存储和信息传递。在数据仓库的这些大的区域中，体系结构组成部分都有不同的用途。

支持数据流的体系结构

现在我们将这些部分组成能使数据从头流到尾的一个完整的框架。就像你知道的那样，最终到达用户端的数据，也就是有用的战略信息，是从不同的数据源的独立数据元素开始的。

从大量数据源收集的数据进入数据准备区域。接下来将会发生什么？抽取出来的数据在数据仓库完全存储之前，要通过数据准备阶段的详细准备处理。用户可以从数据仓库的存储得到已经转换成有用信息的数据。那么究竟什么是数据仓库呢？你同意数据仓库就是收集所有需要的源数据，然后转换并存储成合适的格式，然后向最终用户提供有用信息吗？

让我们来看看图 7-2，图中表现了数据流从开始到最后的过程，并且突出了使信息流动的体系结构。

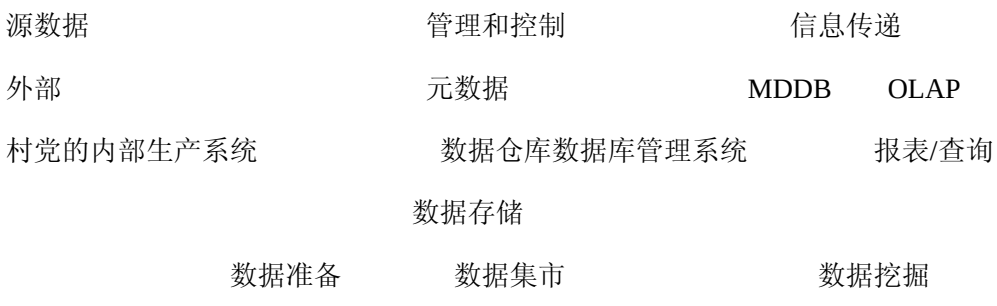


图 7-2　支持数据流的体系结构框架

让我们来看看数据的流动过程，并仔细看看这些体系结构部分。其中的一些部分控制了整个数据流动的过程。管理和控制模块就是这样部分，它几乎贯穿了整个数据运动的每一个步骤。

在数据源部分。很多内部和外部的数据源构成了源数据体系结构组成部分。源数据控制了为数据仓库准备和存储需要的数据抽取。数据准备体系结构组成部分控制了数据的转换、清洗和整合。

在数据仓库存储部分。数据存储体系结构部分包括从准备区域的数据装载，以及为信息传递部分将数据存储为合适的格式。元数据体系结构部分也是一个存储部分，包括了所有从开始到最后整个数据流过程的每一个阶段数据的信息。

在用户端。信息传递体系结构部分包括独立的数据集市、特定的多维数据库，以及各种查询和报表功能。

管理和控制模块

这个体系结构部分是一个完整的管理和控制整个数据仓库环境的模块。这是在多种层次工作并控制所有操作的一个保护部分。这个部分有两个主要的功能：第一个是持续控制所有

的操作，第二个是当出错的时候解决问题并恢复工作。图 7-3 中说明了管理模块是如何联系和管理数据仓库的所有操作。

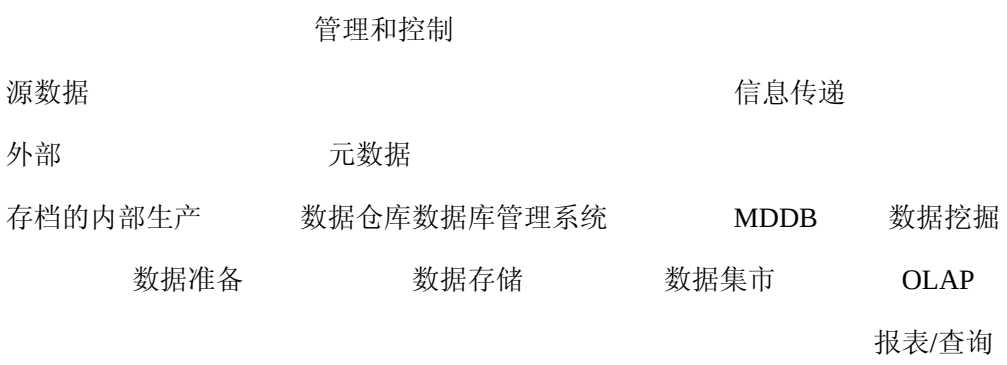


图 7-3　管理和控制模块

数据仓库的所有部分中都有与数据获取相关的操作，包括为了全部更新或者增加数据而从源系统中抽取数据等。将数据移入准备区域，以及执行数据转换功能也是数据获取的一部分。管理的体系结构部分管理和控制了这些数据获取功能，确保抽取和转换的工作正确进行。

目前市场上只有少数几种为数据仓库管理专门设计的工具。总之，数据仓库管理者通过使用数据仓库数据库管理系统的工具，执行了管理和控制模块的功能。

技术性体系结构

我们已经从不同方面，学习了数据仓库体系结构的不同组成部分。首先，我们将这些组成部分分成了三个主要区域，包括数据获取、数据存储和信息传递。然后，我们讨论了数据仓库体系结构的独特性质，突出强调了数据仓库与其它操作型系统相比的不同性质。我们还观察了数据仓库中数据流的过程，并是如何将各个组成部分联系在一起的。

你现在了解了体系结构的概念，以及数据仓库体系结构的构成。每一个体系结构的部分都执行了一定的功能，并提供特定的服务。当所有的部分执行它们预定的功能并提供需要的服务时，整个体系结构就支持数据仓库完成其目标以及商业需求。

因此，数据仓库的技术性体系结构就是其组成部分所提供的完整的功能和服务。技术性体系结构也包括了执行功能和提供服务所需要的程序和规则。技术性体系结构还包括了每一个部分提供服务所需要的数据存储。

让我们来看看另一个重要的特点。体系结构不仅仅是执行功能和提供服务所需要的工具

的集合。当我们提到体系结构组成部分之一的数据抽取功能时，我们仅仅是提到了功能本身以及与这个功能相关的多种任务。而且，我们将准备区域的数据存储与数据抽取功能联系起来，因为收取出来的数据是放在准备区域的。请注意这里我们没有提到执行这个功能的任何工具。这些工具都放在什么地方呢？抽取数据的工具是什么？与这个体系结构相关的工具是什么？工具是执行体系结构的手段。这就是为什么必须首先有体系结构，然后才是工具的原因。

接下来你将要为你的数据仓库体系结构选择合适的工具。让我们来看一个非常简单，也许不太现实的例子。假设你的数据仓库的唯一数据源是一个集中式关系数据库的四张表。如果是这样，数据源部分的范围是什么？数据抽取功能的等级是什么？它们都是有限的。那么你的数据抽取需要复杂的第三方工具吗？显然不是的。那么，站在另一个极端，假设你的数据源包括来自 50 个或者更多的、远程运行不同计算机平台的系统的数据库和文件，你的数据源体系结构组成部分和数据抽取功能就需要非常大而且复杂的范围。你必然需要从第三方产品提供商那里寻找合适的数据抽取工具。

在本章接下来的内容中，我们将会考虑这些组成部分的技术性体系结构。我们将会仔细讨论与每一个体系结构部分相关的各种功能、服务、程序和数据存储。你必须将这些方面都引用在数据仓库体系结构的建立过程中。当你为数据仓库建立体系结构的时候，你要准备一个包括所有组成部分的体系结构计划。这个计划还要详细地说明所有与每一个体系结构部分相关的功能、服务、程序和数据存储的范围和复杂程度。这个体系结构计划将会作为设计和开发的蓝本，以及工具选择的一个控制清单。

让我们来看看数据仓库三个主要区域的技术性体系结构。

数据获取

这个区域包括了从数据源抽取数据、将抽取出的数据移入数据准备区域、以及为向数据仓库载入数据所做准备的整个过程。作为这个区域的前面部分，有两个主要的体系结构组成部分，分别是源数据和数据准备。这个区域的功能和服务与这两个体系结构部分是相关的。数据源的变量对于这些功能和服务有直接的影响。

这个区域将会对于数据仓库的开发有什么样的重要作用？数据抽取、转换和载入的过程是非常耗费时间、人力的，同样也是非常重要的。因此，我们将在第十二章中深入的讨论这个问题。但是，这里我们要了解如何从正确的视角来安排这些体系结构的组成部分。图 7-4

中总结了数据获取的技术性体系结构。

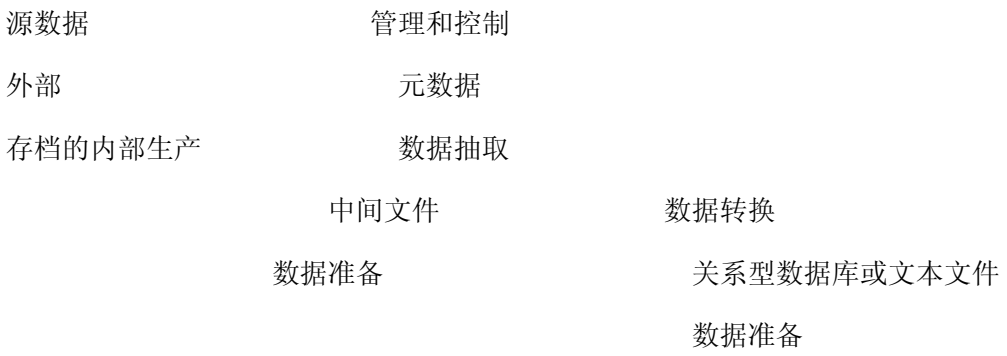


图 7-4　数据获取：技术性体系结构

数据流。在数据获取区域，数据流从数据源开始，然后暂停在准备区域。在转换和整合之后，为数据载入数据仓库做好了准备。

数据源。对于大多数数据仓库来说，主要的数据源一般是企业的操作型系统。一些企业中的多数操作型系统都是过去留下来的旧系统。这些旧数据一般是基于等级的或者网络数据库。你必须通过数据库管理系统中合适的第四代通用语言，从这些数据库中抽取数据。一些近期的操作型系统采用客户机/服务器的体系结构。通常，关系型数据库管理系统都支持这些系统。这里你可以使用 SQL 语言来抽取数据。

有很多公司已经采用了 ERP（企业资源计划）系统。ERP 数据源能够提供整合的数据。但是，使用 ERP 数据源会有一些困难。你不得不使用 ERP 提供商的数据抽取工具，而且大多数的 ERP 产品包括了非常大量的源数据表。

如果要加入外部数据源提供的数据，你就需要创造一些临时文件，来保存外部数据源中得到的数据。在对数据元素进行重定格式和重新分配的转换之后，你才能将这些数据载入准备区域。

中间数据存储。数据从数据源中抽取后，存入临时文件。有的时候，从一些数据源中抽取的同类数据会被存入不同的临时文件，然后在存入准备区域之前，合成另一个临时文件。

相对的处理也很普遍。从每一个应用中，创造一个或者两个大型平面文件，然后分成小一些的文件，在载入数据准备区域之前，正确的进行合成。典型的情况是使用平面文件从操作型系统中抽取数据。

准备区域。在这里，所有抽取的数据被放在一起，准备载入数据仓库。准备区域就像是一个集合车间或者一个建筑工地。在这个区域中，检查每一个抽取的文件，回顾商业规则，执行不同的数据转换功能，选择并合并数据，解决不确定性问题，并清洗数据。当数据最终

准备好后，数据就被临时的存储在准备阶段的库中，等待载入数据仓库。

在大量的数据仓库中，准备区域的数据被存储在连续的或者平面文件中。但是，这些平面文件中包含了采用合适的格式、整合和清洗后的数据。这些文件一般采用典型的格式，这种格式可以通过数据仓库关系型数据库管理系统的工具进行载入。现在越来越多的准备区域数据存储库成为了关系型数据库。这些准备区域的数据可以长时间的保存。虽然抽取过程可以很容易的通过正确的索引从关系数据库中获得，但是创造和维护这些关系数据库需要在源系统中采用大量的索引创造和数据移植。

准备区域可能含有粒度非常小的数据，含有包含商业指标的大量表。在准备区域中存储大量载入需要的合计数据的情况也很普遍。在准备区域中存有另一种类型的与商业维度相关的数据，例如产品、时间、销售区域、客户和促销计划等维度。

功能和服务

这里，我们来看看一个综合的功能和服务的列表。这个列表与数据获取区域相关，概括了这三个群体的功能和服务。这是一个综合的列表。没有指出每一个功能或服务的范围和复杂性。对于数据仓库中的技术性体系结构，你必须决定每一个功能或服务的内容和复杂程度。

功能和服务列表

数据抽取

- 选择数据源，决定应用在每个数据源上的筛选的类型。
- 使用复制或其它技术，创造自动从操作型系统中抽取的文件
- 创造存储数据的临时文件
- 从多种平台中转移已抽取文件
- 为创造抽取文件提供自动化的工作控制服务
- 重新定义外部数据源的输入格式
- 重新定义部门数据文件、数据库和电子表格输入的格式
- 创造数据抽取的通用编码
- 解决多数据源中通用数据元素的不一致性

数据转换

- 将输入数据转化成数据仓库需要的数据
- 清洗数据，复制和合并
- 根据数据仓库维度模型的需求，规范抽取数据的结构
- 转换数据类型

- 计算并得到属性值
- 检查参考完整性
- 根据需要合计数据
- 解决确实值问题
- 巩固并整合数据

数据准备

- 提供准备区域存储库的备份和恢复
- 分类和合并文件
- 创造改变维度表的输入文件
- 如果数据准备存储是一个关系数据库，创建数据库
- 保护数据仓库中每一个数据项相联系的源数据之间联系的检查线索
- 解决并创造载入表的主要外键
- 巩固数据集合，创造数据库管理系统功能载入所需要的平面文件
- 如果准备区域存储是一个关系数据库，抽取载入文件

数据存储

这个区域包括从准备区域向数据仓库存储库载入数据的过程。所有转换和整合数据的功能都在数据准备区域中完成。数据仓库中准备好的数据就像是即将要存储在工厂仓库中的已经完成的产品。

在向数据仓库载入数据之前，作为体系结构中另一个组成部分的元数据已经开始工作了。在数据抽取和数据转换区域中，元数据存储库中已经存有内容了。图 7-5 中总结了数据存储的技术性体系结构。

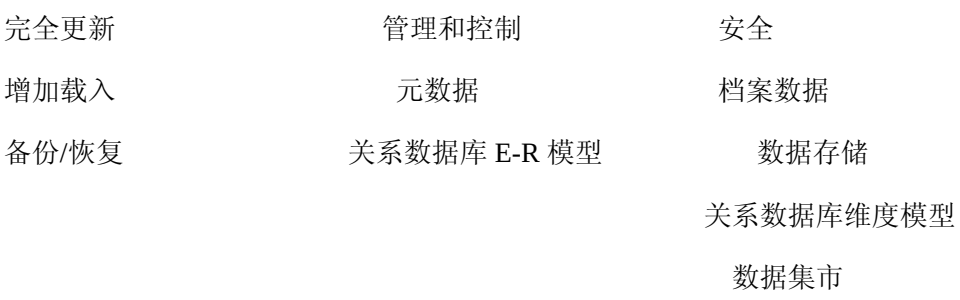


图 7-5　数据存储：技术性体系结构

数据流

流。对于数据存储，数据流从数据准备区域开始。转换和整合的数据从准备区域迁移到数据仓库储存库。

如果数据仓库是一个采用自上而下方法建立的企业级的数据仓库，那么就会有从企业级数据仓库存储库向独立的数据集市迁移的数据。相反，如果数据仓库是通过自下而上方法方法建立，已有数据集市的聚合，那么数据的运动就会暂停在适当的数据集市中。

数据群体。在数据准备区域中准备好的数据分为两类。第一类是为完全更新准备的文件和数据表的集合。这种数据通常是为数据仓库的最初载入准备的。一些数据仓库表也可能偶尔完全更新一次。

另一类数据时包含增加载入的文件和数据表的集合。其中的大多数与每晚的载入相关。一些维度数据的增加载入可能会用更小的频率执行。

数据存储库。几乎所有当前的数据仓库数据库都是关系型数据库。所有关系型数据库管理系统的能源、灵活性和用户易用性在数据处理的时候都可以使用。

功能和服务。

这里列出了一些综合的功能和服务的列表。这个列表与数据存储区域相关，包含了很多功能和服务。这是一个综合的列表，没有指出每一个功能或服务的范围和复杂性。对于数据仓库中的技术性体系结构，你必须决定每一个功能或服务的内容和复杂程度。

功能和服务列表

- 为完全更新数据仓库数据表而载入数据
- 在预定的时间间隔时执行增加载入
- 支持载入详细和汇总程度的维度表
- 优化载入过程
- 提供自动的载入数据仓库的工作控制服务
- 提供数据仓库数据库的备份和恢复
- 提供安全机制
- 控制和协调数据库
- 根据预定的条件，有间隙的从数据库中读取数据

信息传递

这个区域中提到了很多不同的向用户传递信息的方法。对于用户而言，信息传递部分就是数据仓库。他们不会直接接触其它的组成部分。对于用户而言，数据仓库体系结构的强度主要集中在信息传递部分的健壮性和灵活性上。

信息传递部分使用户更加方便的连接信息，直接从企业技术级数据仓库或者从独立的数据集市获取。大多数从数据仓库获取信息都是通过在线查询和互动分析的方法。然而，数据仓库也可以进行一些常用或者特别查询的报表。

几乎所有的现代数据仓库都提供了在线分析处理（OLAP）。最初的数据仓库向多维数据库（MDDBs）载入数据，这些汇总数据通过信息多维立方体的方式进行保存。用户通过这些多维数据库中的信息立方体，来执行复杂的多维查询。如图 7-6，这就是信息传递的技术性体系结构。

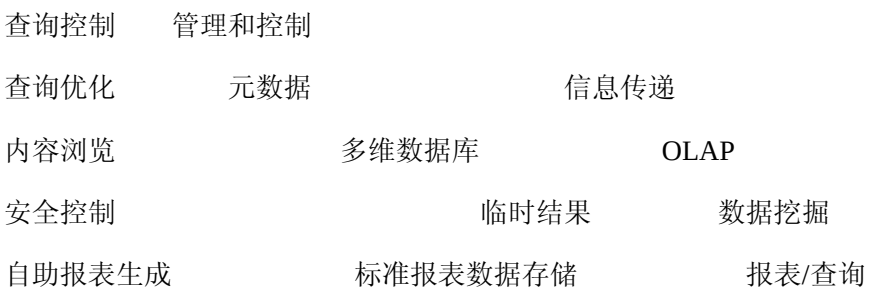


图 7-6　信息传递：技术性体系结构

数据流

流。对于信息传递，如果采用自上而下的方法，数据流从企业级数据仓库或者独立的数据集市开始。如果采用自下而上的设计方法，数据流就从相联系的数据集市开始。通常来说，用户在查询过程中得到转换成信息流的数据。而且，信息也可以采用打印或特别查询的方式传递给用户。有的时候，独立查询或报表的结果可以通过查询或报表工具提供商的数据存储进行保存。保存的信息可以在下次使用的时候，更快的呈现给用户。

在很多数据仓库中，数据也可以流向下游的决策支持系统，例如经理信息系统（EIS）和数据挖掘。另一个更加普遍的信息流是流向 OLAP 的多维数据库。

服务位置。在信息传递部分中，你可能会向用户桌面、应用服务器或数据库本身提供查询服务。这是在体系结构设计中非常重要的决策之一。

为了创建常用或特别查询的报表，你可能要包括一个全面的报表服务。这个服务可以允许用户创建和运行自己的报表，也可以定期的提供标准报表。

数据存储。在信息传递中，你可能会考虑下面这些中间数据存储：

- 关于独立查询和重复使用报表的结果的临时存储
- 标准报表的数据存储
- 多维数据库

功能和服务。首先让我们来回顾功能和服务的大体列表，并将它作为建立数据仓库体系结构的信息传递部分的向导。这个列表与信息传递联系起来，包括全面的功能和服务。这是一个综合的列表，没有指出每一个功能或服务的范围和复杂性。对于数据仓库中的技术性体系结构，你必须决定每一个功能或服务的内容和复杂程度。

- 提供控制信息获取的安全机制
- 监控用户连接，提高服务性能以及帮助未来增长
- 允许用户浏览数据仓库内容
- 通过向用户隐藏数据存储的复杂性，简化连接和使用
- 自动重新定义优化操作的查询格式
- 控制查询过程，特别是失控查询
- 向用户提供自助报表生成服务，包括创建、安排和运行报表的一系列灵活选择
- 为未来使用存储查询和报表的结果
- 提供数据粒度的多层次
- 提供监控数据装载的事件触发
- 向用户提供通过在线分析处理（OLAP）执行复杂查询的功能
- 向下游决策支持系统，例如 EIS 和数据挖掘，提供数据。

本章小结

- 体系结构是将数据仓库的所有部分结合在一起的结构。
- 数据仓库中的只读数据在整个体系结构中作为主要的部分，处于中间位置。
- 体系结构组成部分支持数据仓库在三个主要区域的功能，分别是数据获取、数据存储和信息传递。
- 数据仓库的体系结构是复杂、扩张和宽泛的，有一些与众不同的特性。

- 体系结构框架使数据从一端的数据源流向另一端的用户桌面。
- 数据仓库的技术性体系结构是完整的功能和服务体系。包括执行功能和提供服务的程序和规则，以及提供服务的每一个部分所需要的数据存储。

思考题

1. 你对数据仓库体系结构是如何理解的？用一两段话简单描述一下。
2. 什么是数据仓库的三个主要区域？这是逻辑划分吗？如果是这样，为什么？将体系结构组成部分与这三个主要区域联系起来。
3. 指出数据仓库体系结构的四个特性。
4. 从头至尾描述数据在数据仓库中的流动过程。
5. 对于信息传递而言，数据仓库实施中自上而下和自下而上的方法有什么不同？
6. OLAP 适合体系结构的哪个组成部分？OLAP 的功能是什么？
7. 定义数据仓库的技术性体系结构。它是如何与独立体系结构组成部分相联系的？
8. 列出数据存储区域的五种主要功能和服务。
9. 数据准备区域中存储的类型是什么？
10. 列出信息传递的四种主要功能和服务。简单描述一下。

复习题

1. 判断正误：
   A. 数据仓库体系结构就是完整的向导。而不是数据仓库的蓝本。
   B. 在数据仓库中，元数据部分是独特的，在操作型系统中没有相对应的部分。
   C. 一般说来，数据是从数据仓库存储库流向数据准备区域。
   D. 管理和控制部分与数据仓库中所有的操作都没有联系。
   E. 技术性体系结构就是指厂商提供的产品。
   F. SQL 语言用来从分等级的数据库中抽取数据。
   G. 文件的分类和合并在准备区域非常普遍。
   H. MDDBs 是普通的关系数据库。
   I. 有时，独立查询的结构存储在为再次使用准备的临时数据存储中。
   J. 下游特定应用直接从源数据部分得到。

2. 你最近成为国内汽车保险公司的数据仓库项目的管理者，需要准备一个关于选择正确厂商产品的列表，来帮助数据仓库的管理。列出你的数据仓库体系结构的组成部分的管理和控制功能。使用这个列表来得到工具选择的清单。

3. 作为负责数据准备的高级分析师，你需要设计数据准备区域。如果数据仓库是从多个维度平台上的源系统，以及两个外部系统中获得数据，你如何组织你的数据准备区域？

4. 你是一家著名连锁商店的数据仓库建筑师。数据仓库已经建立并运行大约一年了。现在决定向高级用户提供 OLAP 功能。你如何改变数据仓库体系结构的信息传递部分？

5. 你最近作为数据抽取专家加入了一个数据仓库项目团队，为一家本地但是增长中的药剂店开发数据集市。为数据抽取、数据转换和数据准备的功能和服务，做一个详细的列表。

第八章　数据仓库的基本构造

本章目标

- 了解体系结构和基本构造之间的区别
- 了解数据仓库基本构造如何支持体系结构
- 了解物理基本构造的组成部分
- 回顾数据仓库的硬件和操作系统
- 学习数据仓库可应用的并行处理
- 详细讨论服务器评价
- 学习如何选择数据库关系系统
- 回顾数据仓库需要的工具类型

什么是数据仓库的与体系结构相关的基本构造？两者有什么不同？在哪些方面有不同？为什么不得不分开来学习这两个部分？

在上一章中，我们详细讨论了数据仓库的体系结构。我们了解了不同体系结构的组成部分，并将它们分成了数据仓库的三个主要区域，分别是数据获取、数据存储和信息传递。我们还学习了组成每一个体系结构部分的技术性体系结构的构成元素。

在这一章中，让我们来了解什么是基本构造，以及它包括哪些方面。我们将讨论数据仓库基本构造的每一个部分，你会了解基本构造的重要性，并且掌握为你的数据仓库建立正确基本构造的技术。

支持体系结构的基本构造

考虑体系结构的组成部分。例如，让我们来看看数据准备部分的技术性体系结构。数据仓库技术性体系结构的这个部分有很多的功能。首先，体系结构中有一个部分称为数据准备。这个体系结构的部分包括数据载入数据仓库存储库之前数据准备的区域。其次，这个体系结构部分执行了确定的功能，并提供了数据仓库的特定服务。除此之外，功能和服务包括数据转换和数据清洗。

让我们来问几个问题。数据准备区域究竟在哪里？什么是特定的文件和数据库？这些功

能是如何执行的？如何提供这些功能？什么是下层基础？什么是基础结构？基本构造是支持这些体系结构的基础。图 8-1 中用简单的方式表现了这个事实。

数据仓库体系结构

数据获取- 数据存储- 信息获得

基本构造

图 8-1 支持体系结构的基本构造

什么是支持体系结构需要的多种因素？基本构造包括很多元素：首先，包括基本的计算平台。这个平台包括所有需要的硬件和操作系统。其次，数据库管理系统也是基本构造的重要元素。所有类型的软件和工具也都是基本构造的一部分。那么使这些体系结构运转的人员和流程呢？这些也是基本构造的一部分吗？从某种意义上说，是的。

数据仓库基本构造包括所有执行体系结构的基本元素。总之，基本构造包括一些元素，例如服务器硬件、操作系统、网络软件、数据库软件、局域网和广域网、每一个体系结构部分的厂商工具、人员、流程和培训。

数据仓库基本构造的元素可以分成两大类：操作型基本构造和物理基本构造。这个分类非常重要，因为每一个类别的元素与其它类别的元素相比，在性质和特征上都有很大不同。首先，我们来仔细看看操作型基本构造的元素。相对而言物理基本构造更加基础一些。在对于物理构造有一个基本的了解之后，我们要花一大部分的时间来仔细了解这些元素。

操作型基本结构

为了了解操作型基本构造，让我们再来看看数据准备的例子。基本构造的一部分是指计算硬件和相关的软件。你需要这些硬软件来执行数据准备的功能并提供服务。你需要软件工具来执行数据转换，需要软件来创建输出文件，需要磁盘硬件在准备区域文件中存放数据。但是执行这些功能的人是如何工作的？数据转换的商业规则和流程是如何安排的？管理软件是如何监控和管理数据转换任务的？

支持每一个体系结构组成部分的操作型基本构造包括：

- 人员
- 流程

- 培训
- 管理软件

这些不是开发数据仓库所需要的人员和流程，而是维护数据仓库运行所需要的人员和流程。这些元素与支持数据仓库运行的硬件和软件一样重要，它们支持数据仓库的管理工作并维护它的工作效率。

数据仓库开发者对于基本结构中的硬件和系统软件非常重视。这样做是正确的。但是操作型基本构造经常会被忽略。即使你已经有了正确的硬件和软件，你的数据仓库还需要操作型基本构造来执行相应的功能。没有合适的操作型基本构造，你的数据仓库就可能会勉强运行而工作效率越来越低。所以，注意数据仓库的操作型基本构造。

物理基本构造

让我们先来看看一个图表。图 8-2 中是物理基本构造的主要组成部分。你在这张图表中看到了什么？就像你知道的那样，每一个系统，包括你的数据仓库，都必须有一个完整的平台。基本上，这个平台的组成部分包括基本的硬件部分，软件操作系统，网络和网络软件等。连同这个完整平台一起的还有一系列的工具，运行这些平台来执行每个体系结构部分的不同的功能和服务。

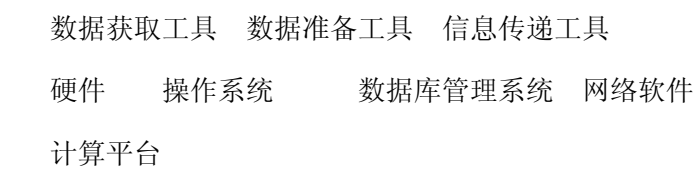


图 8-2　物理基本构造

我们将在下面的几个部分中仔细学习物理基本构造的元素。在数据仓库的基本构造中，最先需要定下来的就是关于硬件的决策。你要确保选中的硬件能够支持整个数据仓库体系结构。

可能我们可以回到大型机的年代，得到一些有用的线索。我们得到两个原则：首先，我们尽可能多的支持已有物理基本构造。其次我们尽可能的将基本结构作为标准组件。当需求提升而且我们可以使用更低的价格来使用更新的版本的时候，我们可以拔掉一个已有的组成部分并插入一个新的。

在数据仓库中，试着采用这两个原则。你已经有了支持当前操作的硬件和操作系统部分。你需要为此花费多少钱？如果提供额外的功能需要花多少钱？数据仓库存储需要的磁盘空间需要多少钱？你需要知道这些问题的答案。

在应用标准组件方法的时候，你可以在服务器硬件中添加程序吗？调查一下你是否可以通过增加更多的硬盘来扩展你的数据仓库。调查硬件组成部分，检查如果使用高版本替换这些部分需要什么。而且，列出需要获得和插入的添加部分的清单。

硬件和操作系统

硬件和操作系统构成了数据仓库的计算环境。所有的数据抽取、转换、整合和准备工作都在这些硬件和操作系统中运行。当你从准备区域向数据仓库存储中传送整合数据的时候，你就要使用服务器硬件和操作系统软件。当客户机进行查询的时候，服务器硬件和数据仓软件执行这些查询并计算结果。

这里有几个关于硬件选择的大体指导，但请注意不是特别针对数据仓库的硬件的。

可扩展性。当数据仓库随着用户数、查询的数量和复杂程度成长的时候，能够确保你选择的硬件可以相应增长。

支持性。厂商的支持对于硬件维护非常重要。确保来自硬件厂商的支持在一个相当高的可能水平上。

厂商鉴定。从其它用户那里得到厂商的鉴定是非常重要的。你可不希望你的数据仓库在首席执行官希望完成一些重要查询的时候，因为硬件的故障而停止工作。

厂商稳定性。检查提供商的稳定性和持久性。

接下来让我们来考虑关于操作系统选择的一些通用原则。首先，操作系统必须与硬件兼容。

可扩展性。可扩展性是第一位，因为这是每一个数据仓库的通用特征。数据仓库在飞快的成长，连同硬件和数据库软件一起，操作系统必须能够支持用户数和应用的快速增长。

安全性。当大量客户机工作站连接服务器的时候，操作系统必须能够保护每一个客户机和相关资源。操作系统必须向每一个客户机提供一个安全的环境。

可靠性。操作系统必须能够保护环境不遭受应用故障。

有效性。这是可靠性的必然结果。计算环境必须在非正常应用结束后能够继续工作。

优先多任务处理。服务器硬件必须能够平衡时间以及多任务的资源的分配。而且，操作

系统必须能够识别高优先级的任务，或者在需要的时候打断其它任务。

使用多线程方法。操作系统必须能够通过在一个多处理器硬件结构中，向多处理器分配线程，来完成同时发生的多请求。这个特性非常重要，因为多处理器结构在数据仓库环境中是可选的体系结构。

内存保护。在数据仓库环境中，大量的查询非常普遍。这就是说可以同时执行多查询。操作系统中的存储保护特性能够保护一项任务不会干扰其它任务的内存空间。

现在我们已经回顾了数据仓库中的硬件和操作系统需求，接下来让我们来试着缩小选择范围。什么是可能的选择呢？下面列出了三个通常的选择。

大型机

- 旧应用中的剩余硬件
- 主要为 OLTP 设计，而不是为决策支持应用设计
- 不适合数据仓库存储
- 不易于扩展
- 当数据集市可以获得过多空闲资源的时候，数据仓库存储却很少使用

开放系统服务器

- UNIX 服务器，大多数数据仓库的选择
- 通常健壮性比较好
- 适合并行处理

NT 服务器

- 支持中等规模的数据仓库
- 有限的并行处理能力
- 对于中等规模或者小型的数据仓库来说是划算的

平台选择

现在让我们来看看执行数据仓库体系结构中的多种功能所需要的计算平台。一个计算平台就是硬件部分、操作系统、网络和网络软件的集合。无论是 OLTP 系统或者是像数据仓库这样的决策支持系统的一个功能，都必须在一个计算环境中执行。

在我们深入讨论平台的选择之前，让我们先来回头看看体系结构部分的三个主要区域的功能和服务。这里是一个快速的扼要重述：

数据获取：数据抽取、数据转换、数据清洗、数据整合、数据准备

数据存储：数据装载、存档、数据管理

信息传递：报表生成、查询处理、复杂分析

现在我们通过这三个区域来讨论平台的选择。每一个功能是在哪里执行的？在什么平台上？你如何优化这些功能？

单一平台方案。这是数据仓库体系结构的实施中最直接和简单的方法。在这种选择中，所有的功能，包括前面的数据抽取到后面的查询处理，都在一个单一的计算平台上进行。这也许是最早期的方法，当开发者在已有的大型机、小型机或单一 UNIX 平台上开发数据仓库的时候。

因为数据抽取、数据存储和信息传递区域的所有操作都是在相同的平台上发生的，所以这种选择下几乎不会遇到任何兼容或接口的问题。数据从开始都结束都平滑的流动，没有任何平台间的转化问题。不需要任何中间件。所有的工具都在一个单一的计算平台上工作。

在很多企业中，旧系统仍然在大型机或者小型机上运行。其中的一些公司迁移到了基于 UNIX 的服务器上，另外一些迁移到了客户机/服务器环境中的 ERP 系统中，作为解决千年虫问题所作转化的一部分工作。无论如何，大多数的旧系统仍然存在于大型机、小型机或 UNIX 服务器中。这些旧系统与数据仓库之间的关系是怎样的？记住，旧系统提供了数据仓库中大部分的数据。如果这些公司想要采用单一平台的解决方案，那么选择的平台必须是大型机、小型机或基于 UNIX 的服务器。

如果公司的情况要求慎重考虑单一平台的选择问题，那么在决策之前要分析这些暗示的问题。单一平台解决方案是一个理想化的方案。如果是这样，为什么很多企业现在没有采取这个方案呢？

扩展容量的旧平台。在很多公司中，已有的旧计算环境可能已经使用二十多年了，已经完全扩展了容量。这种环境可能造成系统不能进一步更新来容纳新的数据仓库系统。

工具的未利用率。软件工具组成了数据仓库基本构造的一大部分。在本章的后面部分会仔细的讨论这个问题。大量数据仓库提供商提供的大部分的工具不支持大型机或小型机环境。在基本构造中没有合适的工具，你的数据仓库就会崩溃。

多种旧平台。虽然我们希望旧的大型机或小型机环境已经被扩展来容纳数据仓库，但是事实上却不是这样的。在大多数企业中，一些大型机系统、小型机应用和基于 PC 的系统是组合在一起共同存在的。大多数公司采用的方法是从大型机，到小型机再到 PC 机。图 8-3 中强调了这种典型的结构。

如果你的公司是这样的企业，你可以为单一平台解决方案做些什么呢？采用这种异类平台的混合形式，在数据仓库中采用单一平台的解决方案是不可能的。

公司的迁移政策。这也是需要考虑的重要问题。你非常了解客户机/服务器体系结构对于计算的各种好处，而且也意识到每个公司正在通过从大型机到小型机平台的迁移，建立新的计算范例。在大多数的公司中，使用信息技术的政策不允许一直存在旧平台。如果你的公司有类似的政策，那么你就不会被允许加入另外的重要系统，例如在旧平台上建立你的数据仓库。

混合方案。在检查公司的旧系统和新应用之后，很可能你会发现单一平台的方法不适合你的数据仓库。这是大多数公司都会得到的结论。另一方面，如果你的公司属于后一类，也就是说旧系统可以容纳你的数据仓库，就可以尽可能的采用单一平台解决方案的方法。如果可能，单一平台解决方案是更简单的解决方案。

对于前者来说，我们必须考虑其它的方案。让我们从第一个主要操作——数据抽取开始，跟随数据流的过程，直到准备区域。我们现在将要仔细来看看数据流并检查平台选择方案。

数据抽取。在任何数据仓库中，数据抽取功能的执行最好是采用每个源系统自己的计算平台。如果你的电话销售数据存放在一个小型机环境中，那就在小型机上为电话销售创建抽取文档。如果你的邮件订单程序在大型机上使用 IMS 数据库执行，那么就在大型机平台上为邮件订单创建抽取文件。如果你足够谨慎的话，还可以将所有的邮件订单数据库文件复制到另一个平台中，然后进行数据抽取。

最初重定格式和合并。在从多种数据源中抽取了未加工数据后，从每个数据源抽取的文件被重新定义格式，合并成新的抽取文件。每个步骤都有抽取数据的证明，保存输入和输出的记录。就像是抽取步骤一样，最好是在和原来一致的源平台上进行最初的合并工作。

初步数据清洗。在这个步骤中，你需要校验所有从数据源中抽取的数据，检查每个部分的缺失值，提供默认值，执行基本的编辑。这是源系统本身计算平台的另一个步骤。但是，在一些数据仓库中，这种类型的数据清洗通常是在所有源数据已经协调和统一之后进行。而且，从源系统中得到的数据特征和条件可以指示何时何地执行数据仓库的这个步骤。

转换和合并。这个步骤包含了所有主要的数据转换和整合的功能。通常来说，你会在这里使用转换软件工具。哪里是执行这个步骤最好的地方？很显然，不是任何一种已有的旧系统。你要在你的准备区域的平台上执行这个步骤。

确认和最后的质量检查。这个步骤是数据准备区域的重要部分。你要为这个步骤安排合适的系统。

创建装载映像。这个步骤为数据仓库中单个数据库文件，创建装载映像。这个步骤几乎总是发生在准备区域，而且因此也在准备区域的平台上运行。

图 8-4 中总结了数据获取步骤和相关的平台。你会注意到这些步骤的方案。将这个与你自己公司的环境联系起来，决定数据获取步骤在什么地方发生。

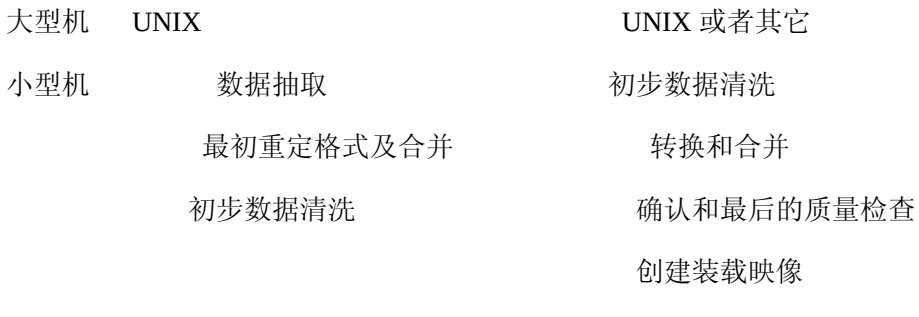


图 8-4　数据获取的平台

准备区域的选择方案。在数据获取步骤的讨论中，我们已经强调了每个步骤的最优计算平台。你会发现关键步骤发生在准备区域。所有数据仓库的数据都在这个区域中集合并准备。什么是准备区域的理想平台？让我们重复一点，最适合准备区域的平台依赖于你的源系统平台的状况。不过，接下来我们来看看准备区域的选择方案，以及主要的向导方法。这会对你的决策有帮助。图 8-5 中展现了准备区域方案的不同的选择。

旧系统之一。如果大部分的旧数据源是相同的平台，而且已经准备了额外的容量，那么考虑将你的数据准备区域放在旧系统上。在这个方案中，你会节省数据迁移方面的时间和精力。

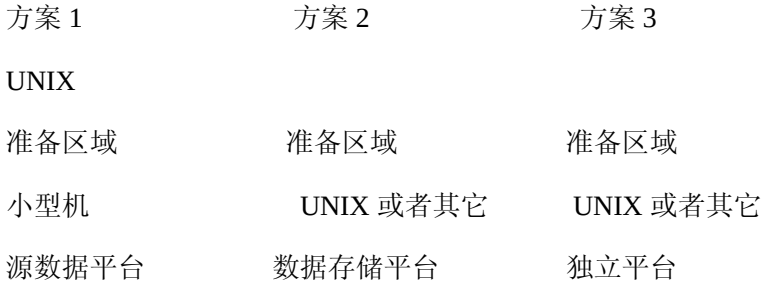


图 8-5　准备区域的选择方案

数据存储平台。数据仓库的数据库管理系统就是在这个平台上运行，数据库也存在于这

个平台上。当你在这个平台上放置数据准备区域时，你会发现在数据库中应用装载映像的所有好处。你可能能够排除一些中间件的子步，并直接将数据从一些准备区域的统一文件中应用在数据库中。

独立的最优平台。你可能会回顾数据源平台，检查数据仓库存储平台，然后你就会发现这些平台都不适合你的准备区域。很可能你的环境需要复杂的数据转换，也可能你需要清洗你的数据来为数据仓库做准备。在这样的条件下，你需要一个独立的平台来在向数据库载入之前准备你的数据。

这里有一些关于数据准备独立平台的优点：

- 你能够为复杂的数据转换和数据清洗而优化这些独立系统。这意味着什么？你可以使用所有必需的工具来促进这些不确定的系统，进行数据转换、数据清洗和数据格式化。
- 当在数据准备区域中转换和清洗这些抽取的数据的时候，你需要保存整个数据内容，并保证在这个过程中没有数据丢失。你可能会考虑一些包含跟踪信息的跟踪文件或者表。一个独立的环境对于管理运动的环境非常有效。
- 我们谈到了使用特定工具在准备区域操作数据的可能性。如果你有一个独立的计算环境，你就可以很容易的对特定人群进行这些工具的培训。

数据运动需要考虑的问题。数据获取和数据存储的独立步骤可以发生的任何计算环境中，数据都必须在各个平台之间运动。依靠公司中的源系统平台以及数据准备和存储的平台选择，你必须提供各个不同平台之间的数据转换。

来看看下面的这些方案。图 8-6 中总结了标准的方案。你会发现仅仅一个独立的方法是不够的。不要犹豫使用一个不同方法的平衡综合方案。在两个计算平台之间进行的每一个数据转移，选择最适合具体环境的方案。下面简单解释这些标准方案。

数据运动

方案 1-共享磁盘

方案 2-大规模数据传递

大型机　　大量数据

方案 3-实时连接

方案 4-手动方法　　UNIX 或者其它

小型机 UNIX

源系统平台　　目标平台

图 8-6　数据运动方案

共用磁盘。这个方法又回到了大型机的时代。在不同地区运行的程序允许通过在共用的磁盘的通用数据来共享数据。你可以使用这个方法在数据仓库中传递数据。你必须指明一个磁盘存储空间，使两个平台中的每一个都能将这个空间作为自己的磁盘。

大规模数据传递。在这个例子中，数据通过平台的传送发生在数据端口之间。数据端口只是接口装置，使大量的数据能够从一个平台转移到另一个平台。每一个平台都必须进行配置，能够在端口之间进行传递。这种方案要求特别的硬件、软件和网络部分。还必须有足够的网络带宽来支持大规模的数据量。

实时连接。在这个方案中，两个平台建立实时的连接，使一个平台上的程序可以使用另一个平台的资源。一个平台上的程序可以写入另一个平台的磁盘存储。而且，一个平台上执行的任务可以确定另一个的任务和事件的时间。通过广泛的采用 TCP/IP，这种方案对于你的数据仓库就是可行的。

手动方法。可能这是最后一个可用的方法。但是，这些方案都是直接和简单的。一个平台上的程序写入一个外部的媒介，例如磁带或磁盘。另一个平台上的程序从外部媒介中读取这些数据。

数据仓库的客户机/服务器体系结构。虽然大型机和小型机平台是数据仓库在早期应用中采用的平台，但是大体上来说，今天的数据仓库都是通过客户机/服务器体系结构建立的。它们中的大多数是多层结构、第二代客户机/服务器体系结构。图 8-7 中表现了数据仓库应用的一个典型的客户机/服务器体系结构。

桌面客户机　　服务类型

展现逻辑

展现服务

应用服务　　中间件/连通性/控制/元数据管理/网络间接/鉴定/查询-报表/管理/OLAP

数据库服务　　数据库管理系统

最初的数据存储库

图 8-7　数据仓库的客户机/服务器体系结构

数据仓库数据库管理系统完成数据服务器的组成部分。数据仓库的数据存储库就是建立在这个机器上的。服务器部分是一个主要的组成部分，我们将在下一部分中详细讨论。

因为数据仓库技术发展的非常迅速，现在你可以发现一个应用服务器部分的增殖。你会发现应用服务器的大量作用，这里有几个要点：

- 运行中间件，并建立连通性
- 执行管理和控制软件
- 从网页中获取数据
- 管理元数据
- 进行鉴定
- 管理和运行标准报表
- 进行复杂查询的管理
- 完成 OLAP 应用

通常来说，客户机工作站可以操作展现逻辑，并提供展现服务。让我们简要的来看看客户机工作站的重要问题。

客户机工作站需要考虑的问题。当你准备考虑工作站机器的配置问题的时候，你会很快的发现你需要满足很多类型用户的要求。我们仅仅考虑工作站关于数据仓库信息传递的需求。一个偶尔使用的用户可能会对通过网页浏览器来获得 HTML 报表的功能非常满意。而一个重要的分析人员，需要一个大型的工作站机器。在这两者中间的其它类型用户可能需要各种不同的服务。

那么你需要为每一种用户建立独特的配置吗？这是不现实的。我们可以在合适的平台上决定一种最小化的配置，支持数据仓库中信息传递工具的标准配置。大多数的用户可以应用这样的配置。在这种配置上增加一些需要的功能。对于高级用户，选择其它的配置，可以支持复杂分析的工具。通常来说，这种给高级用户的配置也会支持 OLAP。

这些为用户工作站选择配置需要考虑的因素，与任何操作型环境需要考虑的因素非常类似。但是，工作站连接数据仓库的主要考虑问题是所选工具的支持。这是进行选择的主要原因。

在考虑工作站时使用下面的列表：

- 工作站操作系统
- 处理能力

- 内存
- 磁盘存储
- 网络和数据传输
- 工具支持

数据仓库成熟的方案。在进行数据仓库计算环境的讨论之后，你会得出结论：一旦最初的方案定下来，关于平台的选择就是固定的。有趣的是即使是成熟的企业数据仓库，其平台的分配也是发展的。数据准备和数据存储可以从相同的计算平台开始。随着时间的推移，越来越多的用户开始依赖你的数据仓库来提供战略决策支持，你会发现平台的选择会不得不重新进行。图 8-8 中表现了数据仓库成熟所需要的东西。

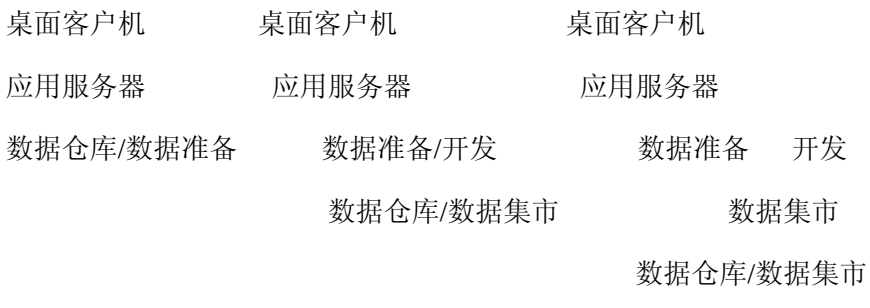


图 8-8　数据仓库成熟的平台方案

实用的方案。在我们离开这个部分之前，还需要了解数据源的类型和不同企业使用的目标平台。有一个独立的第三方调查可以给我们提供一些有趣的发现。图 8-9 中是关于主要数据源的调查结果，图中表现了它们的比例。在图 8-10 中，你可以看到关于这些问题答案的比例，关于数据仓库中数据存储部分的使用平台。

图 8-9 主要数据源

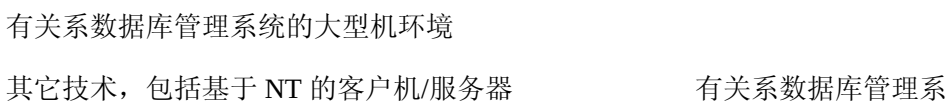


图 8-10　数据存储部分的目标平台

服务器硬件

选择服务器硬件是数据仓库项目团队所要面对的最重要问题之一。很可能，对于大多数数据仓库来说，服务器硬件的选择是类似最后的赌注的决定。可扩展性和优化查询性能是关键的概念。

你知道数据仓库存在的一个主要目的就是向用户提供信息。数据仓库中特别的、不可预测的复杂查询是信息传递最普遍的方法。如果你的服务器硬件不支持更快速查询处理，那么整个项目就是危险的。

扩展的需求是受一些因素驱动的。在你的数据仓库成熟以后，你会看到用户数量和查询数量有大幅度的增加。载入的工作也越来越多。一般说来，用户的数量会在六个月中增长一倍。而且，在你的数据仓库成熟以后，内容的增长也会越来越迅速，包括更多的商业主题并增加更多的数据集市。企业级数据仓库可能会达到 200GB，有些还可能在 18-24 个月内达到 1000GB。

针对可扩展性和复杂查询处理的硬件选择包括四种并行的体系结构。开始的时候，并行体系结构非常有用。如果你提高处理的数量，那么一个查询完成的时间就要更快一点。还可以将一个大的查询再分成一些独立的任务，分配到多个处理器中。使用多个计算引擎的并行处理可以提供很多好处，但是没有一种体系结构能够拥有所有的优点。

在第三章中，我们回顾了并行处理，它是数据仓库中重要的趋势之一。我们也简单看了三种主要的体系结构。在本章中，我们来总结一些当前并行处理硬件的方案。你会更进一步的了解每一个方案的特性、优点和局限。这样你就可以通过你的了解，帮助项目团队选择正确的服务器硬件。

SMP（对称多处理结构）。如图 8-11 所示。

处理器　处理器　　处理器　处理器

通用总线

共享磁盘　　共享内存

图 8-11　服务器硬件方案：SMP

特点：

- 这是一个共享型体系结构，最简单的并行处理机制。
- 每一个处理器都可以通过一个通用的总线连接共享内存。
- 处理器之间的通信可以通过通用的内存发生。
- 磁盘控制器可以使用所有的处理器

优点：

- 自从 20 世纪 70 年代开始使用，是已经得到证明的技术。
- 提供高并发性。你可以运行很多并发的查询。
- 能够很好的平衡工作量
- 提供可扩展的性能，只需要在系统总线中加入更多的处理器
- 作为一个简单设计，你可以很容易的管理服务器。

局限：

- 可以获得的内存是有限的
- 可能会受到处理器之间通信、输入输出和总线通信的带宽的限制。
- 可利用率是有限的；就像一个计算机拥有很多处理器。

如果数据仓库的规模一般在两百或三百 GB，并发性需求是合理的，那么你可以考虑这个方案。

群集。如图 8-12 所示。

处理器　处理器　　处理器　处理器

共享内存　　共享内存

通用高速总线

共享磁盘

图 8-12　服务器硬件方案：群集

特点：

- 每一个节点都有一个或更多的处理器和相关内存。
- 内存在节点之间不是共享的；只在每个节点内部共享。

- 每一个节点都有权使用一系列磁盘。
- 这个体系结构是节点的群集。

优点：

- 这个体系结构提供了高可利用性。所有的数据都是可利用的，即使有一个节点出现故障。
- 保护一个数据库的概念。
- 这个方案有利于不断的增长。

局限：

- 总线的带宽可能会限制系统的可扩展性。
- 这个方案可能会带来操作系统的高支出。
- 每一个节点都有一个数据高速缓冲存储器；这种体系结构需要为内部节点的同步而维护高速缓冲存储器的一致性。一个高速缓冲存储器就是一个“工作区域”，保存了当前使用数据；主要内存就像是一个装满整个屋子的大文件柜。

如果能够很好的定义你的数据仓库的增长，你就可以考虑这个方案。

MPP（大规模并行处理）如图 8-13 所示。

处理器　　处理器　处理器　处理器

内存　　内存　内存　内存

磁盘　　磁盘　　磁盘　　磁盘

图 8-13　服务器硬件方案：MPP

特点：

- 这种体系结构没有任何共享
- 更关注磁盘的存取，而不是内存的存取
- 与一个操作系统协作，支持透明的磁盘存取
- 如果一个数据库表存在于一个特定的磁盘中，那么对磁盘的存取完全依赖于它的处理器。
- 节点之间的通信通过处理者之间的通信完成。

优点：

- 这种体系结构是高度可扩展的。
- 这种方案提供了节点之间的快速存取。
- 任何失败都存在于失败的节点；提高系统的可利用性。
- 一般说来，每一个节点的成本是很低的。

局限：

- 这种体系结构需要严格的数据划分。
- 数据存取是受到限制的。
- 工作量的平衡是有限的。
- 必须维护高速缓冲存储的一致性

如果你在建设一个大约 400-500GB 的中等规模或者大型数据仓库，你可以考虑这个方案。对于 TB 级的大型数据仓库，需要特别的体系结构混合。

ccNUMA 或 NUMA（一致高速缓冲存储的不均匀内存体系结构）如图 8-14 所示。

处理器　　处理器　　处理器　　处理器

磁盘　　内存　　内存　　磁盘

图 8-14　服务器硬件方案：NUMA

特点：

- 这是最新的体系结构，在 20 世纪 90 年代早期开发。
- NUMA 体系结构，可以看成是一个完整的 SMP 分成了很多小一些的 SMP，易于建立。
- 在硬件中要将所有的内存模块看作一个大内存。系统中有一个真正的内存地址；内存地址在第一个节点处称为 1，然后接下去排列。每一个节点都包含了关于这个节点内存地址的目录。
- 在这种体系结构中，获取内存值所需要的时间是变化的，因为第一个节点可能需要存放在第三个节点的值。这就是为什么这种体系结构被称为不均匀内存存取体系结构的原因。

优点：

- 提供最大限度的灵活性
- 克服 SMP 的内存限制

- 比 SMP 有更好的可扩展性
- 如果你需要区分并使用集中的方法运行你的数据仓库数据库，你可以考虑这种体系结构。你也可以将你的 OLAP 数据放在相同的服务器上。

局限：

- NUMA 体系结构的编程要比 MPP 复杂的多
- 支持 NUMA 的软件非常有限
- 技术还不够成熟

这种方案对你来说更加的具有挑战。你可以使用一个 NUMA 的机器，包括一个或两个节点，但是如果你的公司对于硬件的经验不足，那么这种方案就不适合。

数据库软件

来看看主流商业关系数据库管理系统的特征。数据仓库越来越普遍，你可能想要了解数据仓库的特性，特别是它的软件产品。这也恰恰是数据库厂商们正在做的事情。数据仓库及其附件正在成为数据库产品的一部分。通过提高操作型 OLTP 系统数据库软件，来满足决策支持系统的要求。数据库管理系统也已经提高来支持大多数的数据库。

一些关系型数据库关系系统的产品现在可以支持数据仓库的数据获取区域。从其它数据库系统中大量的存取数据越来越简单。一些厂商还特别关注了数据转换的功能。复制特性已经得到加强，来帮助数据仓库进行大量的更新和载入工作。

比特映射索引在数据仓库中对于一些有很多独立值的域的索引工作非常有效。举个例子，在一个包括地理区域信息的数据库表中，区域编码的数量是很少的。但是，经常发生对地区选择的查询。在这个例子中，可以很快获得区域编码索引值的比特映射索引。厂商们已经加强了这个类型的索引。接下来，我们会在第十八章中详细讨论比特映射索引。

除了这些改进，更重要的方面是与装载平衡和查询性能相关的方面。这两点在数据仓库中非常重要。你的数据仓库是以查询为中心的。每一件可以提高查询性能的工作都是最值得做的。数据库管理系统厂商提供了并行处理的特性来增进查询性能。接下来，让我们简单看看数据库管理系统中的并行处理方案。

并行处理方案

数据库软件中的并行处理方案只是对于有多个处理器的机器而言的。目前大多数的数据库软件都可以并行处理大量的操作。这些操作包括：大规模数据载入、全部数据表的扫描、群组查询、特殊值选择、聚合、分类、创建使用子查询的表、创建索引、从其它表中插入列、产生约束条件、星型转换（根据星型模式处理查询的时候的一种最优技术）等等。请注意下面关于关系型数据库管理系统可以并行处理的操作的列表。

现在让我们来看看用户在工作站上创建查询的时候都发生了什么。每一个会话都通过一个服务器进程访问数据库。查询发送到数据库管理系统，并从数据库中取回数据。数据的取得以及结果返回都处于专用的服务器进程的控制之下。查询分配软件负责拆分工作，将需要处理的处理但原分配到目前可用的服务器进程池中，平衡负载。最后，查询进程的结果将被汇总，返回一个单一的综合的结果集。

查询间的并行

在这种方法中，几个服务器进程同时处理几个请求。多个查询被处理，处理方式基于你的服务器配置以及可用的处理器数目。你可以通过利用基于 SMP 系统的数据库管理系统的特性，增加服务器的吞吐量，支持更多的并发用户。

但是，查询间的并行有一些限制。让我们看看这些限制都在哪儿。多个查询被同时处理，但是每个查询依然被单个的服务器进程串行的处理。假设一个查询包括了读索引，读数据，排序，以及连接操作；这些操作都按照这种顺序进行。在下一个步骤进行之前，每一个操作必须完成。同一个查询中的不同部分不能被并行处理。为了消除这个限制，很多数据库管理系统的厂商都提供了提供查询内并行的产品版本。

查询内并行

我们用图 8-15 讨论查询内并行，让我们看看这幅图。在选择数据库硬件以及匹配的数据库管理软件的时候，它会给你很大的帮助。

我们假设一个用户有一个查询，对数据仓库的操作包括了一次读索引，一次读数据，一

次数据连接，以及一个数据排序。一个串行处理的数据库管理系统会按照上面所说的顺序串行的处理并得到结果。但是，如果这个查询运行在一个 SMP 系统的处理器上，其它查询也可以并行处理。这种方式就是上面所说的查询间并行处理。图 8-15 种的第一组操作演示了这种执行方式。

使用查询内并行技术，数据库管理系统将查询分解成更低层次的操作：读索引，读数据，数据连接以及数据排序。这样，每一个基本的操作都在一个处理器上并行执行。最后的结果有所有中间结果合并而成。让我们回顾一下数据库管理系统提供查询内并行的三种方式，也就是说，在同一条查询中的不同部分的并行方式。

水平并行

数据被分区，横跨多个磁盘。查询中的每一任务都被并行处理，例如，读数据，将会同时被多个处理器从多个磁盘中毒取。在第一个任务（从不同分区中读取相应数据部分）结束之后，下一个任务被执行，然后是再下一个任务，如此类推。这种方法的问题在于在所有需要的数据被读取之前，必须等待。请看图 8-15 的案例 1。

串行处理　　查询间并行

读索引　读数据　连接　　排序

查询内并行

案例 A：水平并行

案例 B：垂直并行

案例 C：混合并行

图 8-15　数据库管理系统的查询内并行

垂直并行

这种并行方式在不同的任务之间，而不像水平并行那样之发生在同一个任务中。所有查询中的操作都是并行执行的，但是是以一种管道的方式。这里假定数据库管理系统有能力将

查询分解成子任务；每一个子任务都有所有的操作，包括读索引，读数据，连接以及排序。然后子任务以串行方式之行。在这种方式下，数据库记录最理想的是被其中一个步骤处理完之后，接着又被另一个步骤处理，从而避免了等待时间。当然，在这种方式下，数据库管理系统在分解任务时，必须做很复杂的处理。现在，我们看看图 8-15 种的案例 B。

混合模式

在这种方式下，查询分解器将查询进行水平与垂直的分区。这种方法能够产生最好的结果。你会意识到它能够充分的利用资源，优化性能，具有高度的可扩展性。图 8-15 中的案例 C 演示了这种方式。

数据库管理系统的选择

我们对服务器硬件以及数据库管理系统并行处理选项的讨论肯定能够使你意识到数据库管理系统的选择是很关键的。你必须选择具有相应并行体系结构的服务器硬件。你选择的数据库管理系统必须与你选择的服务器硬件相匹配。这些都是数据仓库的关键决策。

在第 6 章讨论商业需求如何驱动数据仓库的设计与开发时，我们简要的提到了需求对数据库管理系统选择的影响。除了必须具有负载平衡以及并行处理选择之外，选择数据库管理系统的其它关键特性都列在下面。在选择数据库管理系统的时候，必须考虑这些因素：

- 查询管理器——干预或者取消运行的查询
- 查询优化器——分析并优化用户的查询
- 查询管理——平衡不同类型查询的执行
- 装载工具——高性能的数据装载，恢复以及重启动
- 元数据管理——一个活动的数据目录或者数据字典
- 规模的可扩展性——用户的数量以及数据的容量都需要考虑
- 扩展能力——能够有多种扩展到 OLAP 数据库的方式
- 查询工具应用编程接口——对领先厂商的工具开放
- 管理——提供所有的数据库管理员功能的支持

工具集合

考虑一个 OLTP 应用，例如一个商业银行的账户核查系统。如果你作为一个开发者，涉及并部署这个应用，你需要用到多少个第三方的软件工具呢？当然，编程语言以及数据库软件不算在内。我们指的是其它第三方厂商的工具，用于数据建模，图形用户接口设计软件，以及其它。你可能只会用到很少的一些。类似的，一个银行的出纳使用这个系统时，她或他可能不会使用任何的第三方工具。

但是一个数据仓库环境不同。当你作为一个项目组成员开发数据仓库时，你几乎要在开发周期中的每一个阶段都使用第三方工具。你可能会使用代码生成器构造用于数据抽取的内部软件。在数据仓库部署时，你的用户可能通过第三方工具访问信息，并使用报表生成器制作报表。软件工具是数据仓库环境基础架构的一个很重要的组成部分。

在数据仓库体系结构中的每一个部分，都会有相应的软件工具。图 8-16 显示了数据仓库中支持不同功能和服务的软件工具组。

软件工具在数据仓库中非常的重要。正如你在这幅图中看到的那样，工具涵盖了所有的功能。数据仓库项目组仅编写了一小部分内部使用的软件来执行这些功能。由于数据仓库中的工具是如此重要，我们将会在后面的章节中（在 12 章中讨论数据抽取和转换工具，在 13 章中讨论数据质量工具，以及在 14 章中讨论查询工具）继续讨论这些工具。附录 C 也提供了评价厂商解决方案的指南。在你的数据仓库项目到了需要选择工具的时候，该列表会成为一个方便的参考。

在这个阶段，我们将介绍数据仓库环境中需要的软件工具的类型。对每一种类型我们将简要的讨论目的以及功能。

在我们讨论软件工具的类型之前，让我们重申一下在前面章节所提到的格言。在那一章中，我们讨论了体系结构组件并研究了每个功能组件的功能与服务。看下面一段，再一次回顾一下这个重要的原则。

数据仓库管理

中间件以及连接部件

数据获取	数据存储	信息交付
源系统	数据建模	OLAP
抽取	数据仓库/数据集市	报表编辑器

转换	数据装载	决策支持应用	
质量保证	分段区域	预警系统	数据挖掘
	建立装载图像		

图 8-16　数据仓库的工具

体系结构先行，然后才是工具

这个标题的含义很简单：忽略工具，首先要设计好体系结构，然后，才是选择符合那些体系结构中部件的功能和服务的工具。首先要构造体系结构，然后才是选择工具。

为什么这条格言这么重要？为什么不提倡先购买工具，然后使用这些工具构造并部署你的数据仓库？这样做似乎会更简单。那些工具厂商的销售人员保证过一定会成功的。为什么这样做会不成功呢？让我们来看一个例子。

让我们开始设计你的信息传递体系结构部件。首先，商业需求是驱动的因素。你最大的一个用户群是一群高级用户。他们将自己构造报表。他们会运行它们自己的查询。这些用户总是会运行复杂的查询，包括了下钻，切片以及取数据立方体，然后将结果可视化。你知道这些用户是高级用户，他们需要最复杂的信息交付部件。信息交付部件的功能和服务必须是交互的、强大的。但是你不需要自己构造信息交付体系结构部件。

把它放在那吧。现在，让我们假设一个 XYZ 报表编辑器公司的销售人员说服了你，他们的报表生成工具能够满足你的数据仓库所有的信息交付需求。你的两个竞争对手都在他们的数据仓库中使用它。你购买了这个工具，并将它安装在系统中。你的那些高级用户会有什么样的处境呢？这个场景有什么不对吗？信息交付工具在体系结构功能部件建立起来之前就已经选定了。工具不能够满足体系结构的需求。

现在让我们看看数据仓库中有那些类型的软件工具。正如先前提到的那样，相关的细节问题将在后面的章节中讨论。这些章节会详细的介绍特定的工具类型。在下面的部分中，我们仅按照不同工具的类型，介绍它们的基本目标以及特性。

数据建模

- 使开发者能够为源系统以及数据仓库的目标数据库构建、维护数据模型。如果必要的话，数据模型可以构建在数据准备区域。
- 提供前向工程能力，生成数据库模式

- 提供反向工程的能力，从现有数据库的数据字典中生成数据模型
- 为数据设计师提供了构造星型模式的维度建模功能

数据抽取

- 有两种主要的抽取方法:用户完全刷新的批量抽取以及用于增量装载的基于修改的复制
- 工具的选择基于以下的因素：源系统的平台，数据库，以及源系统中的可用的内置抽取与复制工具

数据转换

- 将抽取后的数据，转换成适当的格式以及数据结构
- 提供指定的默认值
- 主要功能包括字段分解，合并，标准化以及反复制。

数据装载

- 将转换、合并之后的数据装载进数据仓库中
- 一些装载工具对装载的表格生成主键
- 对于那些使用同一个数据库管理系统引擎的数据仓库,数据库本身预编码的存储过程就可以用于装载。

数据质量

- 辅助定位、修改数据错误
- 可以用于缓存区中的数据或者直接用于源系统
- 帮助解决装载后数据的不一致问题

查询以及报表

- 允许用户构造图形话的复杂的报表
- 帮助用户构建规格化的查询、运行查询
- 有两个主要的分类：报表服务器，报表编辑器

联机分析处理（OLAP）

- 允许用户运行复杂的维度查询
- 使用户能够生成预先录制的查询
- 联机分析处理有两种类型：多维联机分析处理（MOLAP）以及关系型联机分析处理（ROLAP）。MOLAP 依赖于独立的多维数据库，该数据库从数据仓库中获得数据。ROLAP 直接依赖数据仓库本身，提供联机分析处理能力。

预警系统

- 对预先定义的例外进行提示，引起用户的注意
- 基于数局仓库提供预警，支持战略决策
- 有三种基本的预警类型：从单独的源系统，从集成的企业级数据仓库，从单独的数据集市。

中间件以及连接部件

- 在异构环境中对源系统透明的访问
- 对不同平台不同类型的数据库的透明的访问
- 工具的价格适中，但是事实证明它提供了不同数据仓库功能部件之间的互操作性，这是相当有价值的。

数据仓库管理

- 辅助数据仓库管理员进行日常的维护
- 一些工具专注于装载过程以及跟踪装载的历史
- 其它工具跟踪用户查询的类型以及数量

本章小结

- 基础设施是整个数据仓库体系结构的基础
- 数据仓库基础设施包括了操作基础设施以及物理基础设施
- 硬件以及操作系统构成了数据仓库的计算环境
- 实施不同的体系结构部件所需要的计算平台有几个选配部件

- 选择服务器硬件是一个关键的决策。应该在四种并行服务器架构中选出一种。
- 并行处理选项对数据库管理系统来说是关键的。当前的数据库软件产品可以支持查询内并行以及查询间并行。
- 在数据仓库中，软件工具用于数据建模，数据抽取 ，数据转换，数据装载，数据质量保证，查询以及报表，还有联机分析处理（OLAP）。工具也可以用于中间件，预警系统，以及数据仓库的管理工作。

思考题

1 操作型数据仓库基础架构有那些部分组成？为什么操作性基础架构和物理基础架构同样重要？

2 列出物理基础架构的主要部件。对每个部件用 2 到 3 个句子描述。

3 简要的描述你为数据仓库选择操作系统的原则（任意列出 6 条）。

4 什么是平台的缓冲区域选择？比较各种选择并列出优缺点。

5 在数据仓库内进行数据迁移有哪 4 种方法？解释其中的两种。

6 用两段话简要的描述选择客户端工作站的考虑因素。

7 服务器硬件可以有哪 4 种选择？列出特性，优点，以及每一种的限制。

8 数据库管理系统的厂商为数据仓库做了什么样的扩展？用 1 到两段话进行简要描述。

9 什么是数据库管理系统的查询内并行？有那三种方法？

10 列出 6 类数据仓库的软件工具。选择其中的 3 类，描述以下它们的特性以及目的。

复习题

1 匹配两列

1. 操作性基础设施	A. 非共享体系结构
2. 抢占式多任务	B. 提供高并发性能
3. 共享磁盘	C. 独立内存地址空间

4. MPP	D. 操作系统特性
5. SMP	E. 垂直并行
6. 查询间并行	F. 人，过程，培训
7. 查询内并行	G. 易于管理
8. NUMA	H. 数据仓库平台选择
9. 基于 Unix 的系统	I. 优化数据转换
10. 数据准备区域	J. 数据迁移选项

2 在你的公司中，处了有一个传统的系统在一台主机上面以外，所有的源系统都在一个独立的基于 Unix 的平台上。分析你的数据仓库的平台选择。你会考虑一种单平台的选择吗？如果是这样，为什么？如果不是，为什么不？

3 你是一个全国性的汽车租赁公司的数据仓库项目经理。你的数据仓库预计开始会有500GB。检查服务器的选择，写下评估的标准。

4 作为一个在东部领先的酒店连锁公司的数据仓库管理员，请您写一个报告，描述你为数据仓库选择数据库管理系统的准则。请清晰的描述你的前提假设。

5 你是一个大型本地银行（仅在一个州有分支机构）种负责数据仓库工具的高级分析员。列出在数据仓库种提供的工具类型。包括面向开发人员的工具以及面向用户的工具。描述你所需要的每一类工具的特性。

第九章　元数据的重要角色

本章目标

- 了解元数据的重要性
- 理解谁需要元数据，以及他们需要什么类型的元数据
- 复习元数据的三种功能区域类型
- 讨论商业元数据以及技术元数据的细节
- 检查所有的元数据需求必须被满足
- 理解元数据管理带来的挑战
- 研究提供元数据的选择

我们在前面的章节简要的讨论了元数据。在第 2 章，我们将元数据看作数据仓库的一个主要组成部分。我们将元数据分成 3 类，分别是操作型，抽取与转换，以及最终用户元数据。在第 3 章讨论主要的数据仓库趋势时，我们回顾了行业元数据标准化的开端。

在本章中，我们将深入的讨论元数据。我们会尝试消除关于内容，以及元数据特点方面的疑惑，获得它们的确实含义。我们也将理解为什么元数据如此重要。我们还会更进一步寻求在数据仓库环境种提供有效元数据的实际方法。

元数据的重要性

让我们以一个正面的假设开始。假设你的项目组成功的开发了第一个数据集市。每一件事都按照计划进行。你的管理很成功，项目组在规定的预算之内完成了项目，而且能够在最后期限之前轻松完成。所有的结果在一个全面的测试中都显示出来了。你的数据仓库就要准备部署了。这是一个重要的日子。

你的一个重要的用户正坐在工作站前泰然自若的编辑并运行第一个查询。在他或她敲击键盘之前，忽然想起了几个重要的问题。

- 我可以看那一些预先定义的查询吗？
- 在数据仓库中，哪一些是可变的数据元素？

- 有没有按照产品分类的关于单位销售与单位成本的数据？
- 我怎么浏览哪一些数据是可用的？
- 他们从什么地方为数据仓库取得数据？从哪一个源系统？
- 他们是怎么样合并电话订单系统和邮件订单系统的数据呢？
- 数据仓库中的数据是什么时候的？
- 最后一次新数据倒入是什么时候？
- 有没有按照月份或者产品的汇总？

这些问题是重要而中肯的。那么，答案是什么呢？答案在哪里？你的用户能够看到答案吗？你的用户能够容易的得到这些答案吗？

数据仓库中的元数据包含了这些关于数据的答案。你可以将答案保存在一个叫做元数据存储区的地方。如果你问一下数据仓库的实践者，或者读几本数据仓库方面的书，你会得到关于元数据的不同的定义。这里是一些定义的列表：

- 关于数据的数据
- 数据内容表
- 数据目录
- 数据仓库地图
- 数据仓库路标
- 数据仓库目录
- 将数据仓库内容结合到一起的粘合剂
- 处理数据的钳子
- 神经中枢

那么，什么是元数据呢？这些定义里面哪一个是最接近真实的呢？让我们看一个特定的例子。假设你的用户希望在运行查询之前，知道数据仓库中 Customer 表的信息。在你的元数据存储区里面与 Customer 相关的信息内容是什么？让我们回顾一下 Customer 实体的元数据元素，如图 9-1 所示。

实体名称：Customer

别名：Account, Client

定义：从公司购买产品或服务的一个人或者一个机构。

备注：客户实体包含了常规的，当前以及过去的客户。

源系统：已经完成的产品订单，维护合同，在线销售。

建立日期：1999 年 1 月 15 日

最后更新日期：2001 年 1 月 21 日

更新周期：每周

最后的完全刷新日期：2000 年 12 月 29 日

数据质量回顾：2001 年 1 月 25 日

最后的副本：2001 年 1 月 10 日

计划归档：每 6 个月

负责用户：Jane Brown

图 9-1　Customer 实体的元数据元素

在图中你看到什么呢？元数据元素描述了数据仓库中的 Customer 实体。它不仅仅是一个描述。它给出了更多关于语法和语义的解释。元数据完整、精确的描述了数据仓库中的数据的所有重要方面。对谁重要呢？对那些用户、开发人员以及项目组来说都是重要的。

在这一章中，我们会探索为什么元数据在数据仓库中的角色如此重要。我们会找出为什么元数据对用户和开发人员来说是重要的。没有了元数据，你的数据仓库就仅仅是一个分离的系统。如果元数据如此重要，你怎么才能提供呢？我们会讨论一些可能的选择，并给出一些建议。

数据仓库的关键需求

让我们以一种通用的方式考察源数据的需求。在后面的部分，我们做更具体的考虑。从更广义的角度来讲，正确的元数据对于使用、构建、管理数据仓库是绝对必要的。

使用数据仓库

数据仓库与操作型的系统（例如一个订单处理系统）由一个很大的区别。不同的地方在于使用方式——访问信息的方式。在一个订单处理应用中，你的用户是怎么获取信息的呢？你为他们提供图形界面以及预定义的报表。他们通过相关的界面获取订单的信息。他们通过特定每日报表获得订单汇总信息。你为你的用户构建界面以及定制报表。当然，这些都是根据用户特定的需求定制的。然而，用户在获取信息的时候，并不是自己每次都构建界面或者报表的格式。

形成鲜明对照的是，用户在使用数据仓库时，自己从数据仓库中获取信息。基本上，用户自己构建特定的查询并在数据仓库中运行它们。他们自己格式化报表。由于这些主要的差异，在他们构建并运行查询之前，用户需要了解数据仓库中的数据。所以，他们需要元数据。

但是，在我们的操作型系统中，我们没有灵活、简便的方法去了解数据库的内容。实际上，也没有使用友好的界面访问数据库内容的必要。数据字典或者数据目录只是为信息技术人员服务的。

对数据仓库来说，情况就完全不同了。你的数据仓库用户需要从数据仓库中获得最大的收益。他们需要复杂的方式浏览以及考察数据仓库的内容。他们要知道数据项的含义。你要防止他们忽略了数据的确切含义就去做分析，从而做出了错误的结论。

早期的数据集市相对局限，他们仅仅限于某一个主题。数据集市大多数为一个单独的部门的用户所使用。数据集市的用户能够获得有限的元数据。现在的数据仓库涵盖了更广阔的数据范围，并更大的规模。如果没有足够的元数据支撑，这些大型数据仓库的用户将会受到很大的限制。

构建数据仓库

我们假设你是项目组中负责数据抽取以及数据转换的专家。你对数据抽取方法非常了解，也会使用数据抽取工具。你也理解通用的数据转换技术。但是，要应用你的专业能力，你必须首先知道源系统以及它们的数据结构。你需要知道数据仓库中的结构以及数据内容。然后你需要决定数据映射以及数据转换。到目前为止，为了构建数据仓库中的数据抽取以及数据转换模块，你需要源系统的元数据，源系统到目标系统的映射，以及数据转换的规则。

现在假设我们换一种角色。你现在是数据仓库的数据库管理员。你负责数据库的物理设计以及初始的装载。还需要负责定期的增量装载。除此以外，你还有其它的职责。即使忽略棋它的职责，为了完成物理设计、装载，你需要了解相当多的元数据。你要知道缓存区域的布局。你需要知道数据仓库逻辑结构的元数据。你要知道数据刷新以及装载周期的元数据。这只是你所需要知道的信息中的最小的一部分。

如果你考虑构建数据仓库中的每一个活动以及每一个任务，你会意识到元数据是数据仓库中的一个相当重要的模块。元数据对于构建数据仓库是相当重要的。

管理数据仓库

由于现代数据仓库的复杂性以及庞大的规模，要管理数据仓库，没有坚实的元数据是不可能的。图 9-2 列出了一系列与管理数据仓库相关的问题。请仔细的看列表上的每一个问题。如果没有对这些问题作回答，就很难管理数据仓库。你的数据仓库元数据必须涵盖了这些问题。

数据抽取/转换/装载

如何处理数据变化？

如何包括新的源系统？

如何清洗数据？

如何改变数据清洗的方法？

如何转换成新的数据转换技术？

如何审计应用的改变？

数据仓库

如何增加新的汇总表？

如何控制运行的查询？

如何扩展存储？

何时规划平台升级？

如何为用户增加新的信息传递工具？

如何继续培训？

如何维护并扩展用户支持功能？

如何管理并提高即席查询的性能？

什么时候安排备份？

如何执行运行灾难恢复演习？

如何保持数据定义是最新的？

如何维护系统的安全？

如何监控系统的负载分布？

外部系统的数据

如何增加新的外部数据源？

如何去掉某些外部数据源？

如果合并或者获取数据，如何将新的数据加入到数据仓库中？

如何检验所有的外部数据？

图 9-2　数据仓库管理：问题列表

谁需要元数据？

我们先暂停一下，来考虑在数据仓库环境中那些人需要元数据。请看图 9-3 中的列。这个图告诉你谁需要元数据。我们将在后面的部分详细的讨论这个问题。

想象一个没有标签和文件夹的文件柜。如果没有元数据，那么数据仓库就像这样的一个文件柜。它可能装满了很多对你的用户、开发这以及管理者很有用的信息，但是，却没有任何简便的方法知道这些信息在哪里，数据仓库的价值是很有限的。

元数据就像一个神经中枢

在构建和管理数据仓库的过程中，不同的过程都会产生部分元数据。在数据仓库中，元数据处于一个关键的位置，它使不同的过程能够相互通信，是数据仓库的神经中枢。图 9-4 显示了数据仓库中元数据的位置。可以用这幅图确定应用在你的数据仓库中的元数据模块。

通过仔细的检查每一个元数据模块，你将认识到有两类人会用到元数据：（1）最终用户，以及（2）信息技术（IT）人员（包括开发人员以及管理员）。在下面的两个部分，我们将回顾为什么元数据对这两个群体来说是关键的。

	IT 专家	高级用户	一般用户
信息发现	数据库，表，列，服务器平台	数据库，表，列	预定义的查询以及报表以及商业视图的列表
数据的意义	数据结构，数据定义，数据映射，清洗功能，转换规则	商业术语，数据定义，数据映射，清洗功能，转换规则	商业术语，数据定义，筛选，数据源，转换，数据所有者
信息访问	用 SQL，第 3 代语言，第 4 代语言编写的程序代码，前端应用，安全性	查询工具集，复杂分析所需要的数据库访问	鉴权请求，信息获取倒入桌面应用（例如电子表格）

图 9-3　谁需要元数据？

（中央）数据仓库元数据

（周围顺时针）查询工具，报表工具，OLAP 工具，数据挖掘，应用，外部数据，数据装载功能，转换工具，清洗工具，抽取工具，源系统

图 9-4　元数据扮演一个神经中枢的角色

为什么元数据对最终用户是关键的

以下的内容将是一个数据仓库的关键用户——商业分析员的典型使用方式。你所在公司市场营销部门的领导要求这个商业分析员对一个最近出现的问题作一个深入的分析。

由于中西部以及东北部地区有很大的销售潜力，你的公司在每一个地区开了 5 个新的商

店。虽然在开了新的商店之后的两个月，全国的销售持续增长，但是在此之后，销售回落到先前的水平，并保持平稳。市场营销部门的领导希望知道原因，从而可以采取正确的行动。

作为一个用户，商业分析员希望从新的数据仓库中得到答案，但是他不了解数据仓库中的具体细节。他对以下问题的答案并不了解：

- 各种产品以及每个商店每天的销售数量以及金额是按照每一个单独的交易，还是按照汇总数据存储的呢？
- 销售情况能够按照产品，促销行为，商店以及月份进行分析吗？
- 当月的销售能够与去年同期的销售进行比较吗？
- 销售情况能够与预期目标进行比较吗？
- 边际利润是如何计算的呢？商业规则是什么？
- 一个销售区域的定义是什么？需要分析的两个区域包含哪一些州呢？
- 销售情况的数据从何而来？是从源系统吗？
- 销售数字是什么时候的？这些数据多久更新一次呢？

如果分析员不了解数据的情况，他有可能做出错误的分析。有可能新商店的销售额是从原有的商店中调拨过来的，因而总体的销售保持平缓。但是分析员由于错误的理解结果，不能找到正确的原因。

如果你能够提供一个适当的元数据作为引导，分析将会变得更有效率。如果有足够的正确的元数据，分析员不会在每次运行一个分析之前都需要求助于 IT 人员。能够容易的访问源数据对最终用户来说是很重要的。

让我们拿一个按目录保存货物销售条目的工业产品数据仓库。客户按照目录寻找被下订的产品。客户使用目录中的产品编号下订单。同样的，目录包括了颜色，大小，以及货物的形状。客户通过目录中的价格信息计算付款的总数。简单地说，目录包括了工业仓库中所有的条目，描述了产品情况，从而便于客户下订单。

类似的，数据仓库的用户就像一位顾客。一条来自用户的查询就像对一个工业仓库中货物条目的订单。就像一个顾客需要目录来下一个订单一样，你的用户也需要元数据来运行一条对数据仓库的查询。

图 9-5 汇总了最终用户需要的关键的元数据。这幅图显示了为最终用户提供的元数据的类型，以及他们需要这些类型信息的目的。

最终用户

元数据对于最终用户至关重要

数据内容 汇总数据 商业维度 商业指标 浏览路径 源系统 外部数据 数据转换规则 最后更新日期 数据装载/更新周期 查询莫版 报表格式 预定义查询/报表 OLAP 数据	元数据对 IT 人员至关重要

图 9-5　元数据对于最终用户至关重要

为什么元数据 IT 人员来说是关键的

数据仓库的开发以及部署需要用户代表以及 IT 人员的共同努力。然而，由于技术方面的原因，IT 人员主要负责数据仓库的设计以及随后的管理工作。要设计以及管理数据仓库，IT 必须能够访问正确的元数据。

在整个的开发过程中，元数据对 IT 来说都是关键的。从开始的数据抽取到最后的信息传递，元数据的相当的重要。同样，在数据抽取，数据转换，数据集成，数据清洗，数据缓存，数据存储，查询以及报表设计，OLAP 设计，到其它的前端系统，元数据对 IT 人员执行他们的开发活动都是不可或缺的。

以下是元数据发挥重要作用的过程汇总列表：

- 从源系统中进行数据抽取

- 数据转换
- 数据清洗
- 数据汇总
- 数据缓存
- 数据刷新
- 数据库设计
- 查询以及报表设计

图 9-6 汇总了 IT 人员对元数据的关键需求。这幅图显示了元数据为 IT 人员提供的信息类型以及它们需要这些类型信息的目的。

元数据 IT 人员来说是关键的

元数据对于最终用户至关重要	源数据结构 源平台 数据抽取方法 外部数据 数据转换规则 数据清洗规则 缓存区域结构 维度模型 初始装载 增量装载 数据汇总 OLAP 系统 WEB 访问 查询/报表设计

图 9-6　元数据 IT 人员来说是关键的

数据仓库任务自动化

维护元数据不再是一系列的文档。传统上讲，元数据是一堆对每一个过程的数据进行描述的文档。现在，元数据扮演了一种新的活跃的角色。现在我们看这一切是如何发生的。

正如你所知道的那样，在数据仓库环境中，工具执行了主要的功能。例如，使用工具可以从数据源中抽取数据。只要你提供映射算法，数据转换工具就可以将数据转换为目标数据结构。你可以指定数据元素的有效值，数据指定工具会使用这些值保证数据的有效性和一致性。前端工具可以是用户访问数据仓库，浏览数据内容。这些工具可以分成两类：IT 专家的开发工具，以及最终用户的信息访问工具。

作为一个开发者，你需要使用工具进行开发和设计。工具能让你建立并记录部分的数据仓库源数据。当你使用另外一个工具执行开发设计中的另外一个过程时，这个工具会用到第一个工具所建立的元数据。当你的最终用户使用访问信息的查询工具时，查询工具使用了某些后端工具构建的元数据。在这里元数据出现了什么变化呢？元数据不再是被动的文档，而是参与了整个过程，它协助了数据仓库任务的自动化。

让我们从定义数据源开始，考虑一下后端的过程。当数据通过数据缓存区域从数据源转移到数据仓库的过程中，会执行一些处理。在一个典型的数据仓库中，每一个处理过程都有合适的工具提供帮助。在数据转移的过程中，每一个工具都记录了自身的元数据。而后续的处理过程会使用这些源数据。元数据扮演了一个活跃的角色，辅助数据仓库处理过程的自动化。

以下是按照发生先后顺序排列的后端处理过程的列表：

1、 数据源结构定义；

2、 数据抽取；

3、 初始重格式化/合并

4、 初步数据清洗

5、 数据转换/合并

6、 有效性和质量检查

7、 数据仓库结构定义

8、 创建装载映像

图 9-7 显示了这 8 个处理过程。这幅图也指出了每一个处理过程中记录的元数据。而且，

该图指出每一个处理过程如何利用前面过程构建的元数据。

由于元数据驱动了整个过程，所以它在数据仓库中是相当重要的。但是，上面的讨论使我们意识到，每一个工具都使用它们自己的格式记录源数据。而且，每一个工具记录的元数据可能在相应处理过程的平台上。如果是这样的话，每一个工具用自己的格式记录的元数据怎么样驱动下一个工具所在的过程呢？因此，源数据必须标准化。我们将在本章的末尾讨论元数据的标准。

源数据结构定义|源系统平台，数据结构-> 数据抽取|抽取技术，初始文件以及结构->排序/合并规则，合并文件以及结构|初始重格式化/合并->数据清洗规则|初步数据清洗->数据转换/合并|数据转换规则，汇总->有效性和质量检查|质量检验规则->数据模型 逻辑/物理|数据仓库结构定义->关键结构规则，DBMS 的考虑|创建装载映像

解释： 处理过程； 相关的元数据

图 9-7 元数据驱动数据仓库的处理过程

建立信息上下文

设想这样一个场景。一个你的用户希望运行一个查询，得到南部地区 4 月份头 7 天三个产品的销售数据。这个用户的查询如下：

产品=Widget-1 或 Widget-2 或 Widget-3

地区=“南部”

时间=04-01-2000 到 04-07-2000

返回的结果：

	销售单位	数量
Widget-1	25,355	253,550
Widget-2	16,978	254,670
Widget-3	7,994	271,796

我们检查一下查询以及结果。在地区的设定中，“南部” 地区包括了那些地域呢？这些

地域是用户感兴趣的地域吗？在数据仓库中，“南部”这个数据项的上下文是什么？数据项 04-01-2000 指的是 2000 年 4 月 1 日还是 2000 年 1 月 4 日？数据仓库中日期的格式是什么？

看看结果集。销售单位显示的数字是按照产品物理的单位，还是按照一些指标，例如公斤或者是磅？结果集中显示的数量的单位又是什么呢？这些数量是美元还是其它货币？如果你的用户在欧洲访问数据仓库，这是一个很恰当的问题。

对于数据仓库中存储的日期，如果日期格式的头两个数字表示月份，接下来的两个数字表示日期，那么 04-01-2000 表示 2000 年 4 月 1 日。只有在这种上下文中，这种解释才是正确的。类似的，上下文对于其它数据元素的解释是相当重要的。

你的用户怎么样才能知道查询中每一个数据元素含义是什么，以及结果集的含义是什么？答案就是元数据。元数据告诉用户数据元素的确切含义。元数据建立了数据元素的上下文。数据仓库用户，开发者以及管理者可以通过元数据中存储的上下文解释每一个数据元素。

按功能区域划分的元数据类型

本章到目前为止，我们讨论了数据仓库环境中元数据的几个方面。我们已经了解为什么元数据对于最终用户以及负责开发、维护数据仓库的 IT 专家来说是很重要的。我们也理解元数据在数据仓库过程自动化中扮演了一个重要的角色。

在这个阶段，我们会深入讨论元数据的不同类型。通过分类，你将会对每一类的元数据有更深入的理解。

不同的作者以及数据仓库的实践者用不同的方法对元数据进行分类：有些按照用途分类，有些按照用户分类。我们来看看一些元数据分类的方法。以下列出了元数据不同的分类方法：

- 管理/最终用户/优化
- 开发/使用
- 在数据集市/在工作站
- 构建/维护/管理/使用
- 技术/商业
- 后端/前端
- 内部/外部

在前面的章节中，我们考虑了一种按照主要功能划分数据仓库环境的方法。我们可以将

数据仓库按照功能划分为 3 个区域：数据获取，数据存储，以及信息传递。所有数据仓库处理过程都可以归为这 3 个区域。作为一个开发者，你的设计和处理涉及了 3 个区域。与这些处理过程相关的每一个工具会生成、记录元数据，并可能会用到其它工具记录的元数据。

首先，我们按照这 3 个功能区域对元数据进行分类。为什么？因为每一个数据仓库处理过程都只出现在这 3 个区域中的其中一个。考虑每一个功能区域中的所有处理过程，然后将所有的处理过程合并。你会得到一个数据仓库处理过程的完全的集合，而不会漏掉任何一个。你也可以得到一个元数据类型的完全集合。

下面我方将讨论按照数据仓库功能区域划分的元数据类型：

- 数据获取
- 数据存储
- 信息传递

数据获取

在这个区域中，数据仓库过程与下面的功能相关：

- 数据抽取
- 数据转换
- 数据清洗
- 数据集成
- 数据缓存

在处理过程进行时，合适的工具记录了与这些处理相关的元数据元素。这些工具记录了数据仓库开发阶段以及部署之后的元数据元素。

作为一个 IT 专家，项目组中的一员，你会在这个区域中用到记录元数据的开发工具。这些工具（无论是在本区域中还是在其它区域）也可能会用到其它工具所记录的元数据。例如，在你使用一个查询工具构建标准查询时，你会用到数据获取区域中处理过程所记录的元数据。查询工具是在另外一个称为信息传递的区域中使用的。

在部署数据仓库之后，IT 专家会使用在数据获取区域中的处理过程所记录的元数据进行管理和监控正在运行的功能。你会使用这些元数据，监控数据抽取以及数据转换。你会引用这些元数据构建装载影像。

数据仓库用户也会使用数据获取区域中的元数据。当一个用户希望找到查询中的数据元

素的来源时，他或她会查询数据获取区域中的元数据。如果用户希望知道数据仓库中保存的边际利润，他或她会察看数据获取区域中元数据的来源描述规则。

对数据获取区域中记录和使用的元数据类型，请参考图 9-8。这幅图总结了元数据类型以及相关的数据仓库处理过程。试一下将这些元数据类型以及处理过程与数据仓库环境联系起来。

数据获取

处理

数据抽取，数据转换，数据清洗，数据集成，数据缓存

元数据类型

源系统平台	汇总规则
源系统逻辑模型	目标逻辑模型
源系统物理模型	目标物理模型
源结构定义	缓存区域中的数据结构
数据抽取方法	源系统到目标系统的联系
数据转换规则	外部数据结构
数据清洗规则	外部数据定义

图 9-8　数据获取：元数据类型

数据存储

在这个区域，数据仓库处理过程与下面的功能相关：

- 数据装载
- 数据存档
- 数据管理

就像在其它区域中，当处理过程在数据存储功能区域中进行时，相应的工具记录了和处理相关的元数据元素。这些工具记录了数据仓库开发阶段以及部署之后的元数据元素。

与数据获取区域中处理过程记录的元数据类似，数据存储区域过程中记录的元数据用于

开发，管理，以及最终用户。你将在完全数据刷新以及增量数据装载中使用到这个区域中的元数据。数据库管理员会在备份、恢复、性能调整的处理中，使用这些元数据。在数据仓库管理的清晰以及定期归档的过程中，也会用到该区域中的元数据。

那么，最终用户会用到数据存储功能区域的元数据吗？我们举一个例子，假如我们一个用户希望构建一个查询，按照销售区域将季度销售分开。在用户运行查询之前，她或他希望知道区域描述的最后更新时间。那么，用户可以从什么地方得到这个信息？存储在数据存储功能区域中的数据装载过程的元数据记录将为用户提供区域描述的最后装载日期。

对数据存储区域中记录和使用的元数据类型，请参考图 9-9。这幅图总结了元数据类型以及相关的数据仓库处理过程。请考虑如何将这些元数据类型以及处理过程与数据仓库环境联系起来。

（原书错误，图 9-9 与图 9-10 重复）

图 9-9　数据存储：元数据类型

***

信息传递

在这个区域，数据仓库处理过程与下面的功能相关：

- 报告生成
- 查询处理
- 复杂分析

这个区域的处理过程主要为最终用户服务。最终用户在处理这些过程时，会用到数据存储区域以及数据获取区域中记录的元数据。当一个用户在查询处理的辅助下，构建一条查询时，她或他可以引用数据存储区域以及数据获取区域中记录的元数据，并察看元数据配置，数据结构以及数据转换的元数据信息。同样的，通过存储在数据存储区域的元数据，用户可以找到完全更新以及增量装载的日期。

通常，记录在信息传递功能区域的元数据与预定义查询，预定义报表，以及对查询和报

表的输入参数定义。功能区域记录的元数据也包括 OLAP 相关的信息。开发者和管理员都参与这些处理过程。

对信息传递区域中记录和使用的元数据类型，请参考图 9-10。这幅图总结了元数据类型以及相关的数据仓库处理过程。请考虑如何将这些元数据类型以及处理过程与数据仓库环境联系起来。

元数据类型也可以分类成商业元数据以及技术元数据。这是另一种对元数据分类的有效方法。这是因为商业元数据中的内容和格式和技术元数据截然不同。下面两部分我们将讨论这种分类方法。

信息传递

处理

报表生成，查询处理，复杂分析

元数据类型

源系统	目标物理模型
源数据定义	用商业术语表达的目标数据定义
源结构定义	数据内容
数据抽取规则	数据浏览方法
数据清洗规则	查询模版
源系统到目标系统的映射	预格式化报表，预定义查询/报表
汇总数据	OLAP 内容

图 9-10　信息传递：元数据类型

商业元数据

商业元数据与数据仓库的用户有紧密的联系。商业用户需要知道数据仓库中什么数据是可用的，他们的角度与 IT 专家的角度有很大的不同。商业元数据就像一幅地图或者是一个易于使用的信息目录，它显示了信息所在的地方以及如何到达那个地方。对于经理层以及分析人员来说，它是一个重要的指南。

内容总揽

首先，商业元数据必须用商业术语和平直的语言描述内容。例如，数据表或者独立数据元素的名称不能够含糊，而应该是用户熟悉的有意义的术语。数据项名称 calc_pr_sle 是不能够接受的。你应该把名称修改为 calculated-prior-month-sale。

相对于技术元数据，商业元数据的结构化程度要差得多。相当一部分商业元数据来自于文本，电子表格，以及完全没有记录的商业规则和政策。虽然有不少的商业元数据来自于非正式的来源，它们和来自于正式来源的元数据（例如数据字典条目）是同样重要的。所有非正式的元数据必须用一种标准的格式获取，并存储在数据仓库中。

相当多的商业用户没有足够的技术背景构建查询或格式化报表。他们需要知道哪一些预定义查询是可用的，有哪一些预先格式化的报表。他们必须能够通过引用的商业名称识别数据表和列。因此，商业元数据应该用平直的语言表达所有这些信息。

商业元数据举例

商业元数据专注于为工作站的终端用户提供支持。它必须很容易的使最终用户理解数据仓库中什么数据是可用的，以及他们怎么使用这些数据。商业元数据完全从最终用户的角度描述数据仓库。它就像一个用用户易于理解的简单的商业术语描述的数据仓库外部视图。

以下例子能够使我们更好的理解商业元数据：

- 连接过程
- 安全性以及访问权限
- 用商业术语表述的数据的全面结构
- 源系统
- 从源数据到目标的映射
- 数据转换商业规则
- 汇总以及演化
- 表名称和商业定义
- 属性名称和商业定义
- 数据归属权
- 查询和报表工具

- 预定义查询
- 预定义报表
- 报表分布信息
- 访问路由通用信息
- 使用 OLAP 分析规则
- OLAP 数据货币单位
- 数据仓库刷新规则

以上列表并不是包括万象的，但是它为你给数据仓库建立一个类似的列表建立了一个很好的基础。你可以将这个列表作为一个指南，以保证向你的用户提供易于理解的商业元数据。

内容重点

通过以上例子列表，我们介绍一下商业元数据的重点。商业元数据能够回答什么样的问题呢？用户能够从商业元数据中得到什么样的信息呢？

下面我们列出一系列商业元数据能够为最终用户回答的问题。虽然下面的列表不能包括所有最终用户的问题，但是它可以作为一个有用的参考：

- 我怎么进入数据仓库？
- 我能够访问哪一部分的数据？
- 我能够看到一个特定表格中的属性吗？
- 我的查询中所需要的属性的定义是什么？
- 有没有一些预定义的查询或者报表能够给出我需要的结果？
- 我的数据来自于哪一个源系统？
- 我的查询所获取的数据有没有设置默认值，这些默认值是什么？
- 对于所需的指标，有什么样的汇总数据是可用的？
- 我所需要的数据项的值是从那些值推导而来的？
- 我的查询中设计的数据项最后更新的日期是什么时候？
- 在哪一些数据项上面我可以执行下钻的操作？
- OLAP 数据是什么时候的？我应该等到下一次更新吗？

谁会受益？

商业元数据对最终用户最有用。这是一个通常的说法。谁会从商业元数据中特别受益呢？商业元数据是如何为特定的最终用户群服务的呢？请看一下列表：

- 经理
- 商业分析员
- 高级用户
- 常规用户
- 一般用户
- 高级经理/高级管理层

技术元数据

技术元数据主要为负责开发、维护数据仓库的 IT 人员服务。数据仓库的每一个功能区域都有一些处理过程，技术人员在设计每个处理过程的时候都需要相关信息。你作为项目队伍中技术组的一员，必须知道数据仓库计划中的结构以及内容。项目组中不同的成员需要不同的技术元数据信息。如果说商业元数据对数据仓库的用户来说是一张路线图的话，技术元数据对构建、维护、管理数据仓库的 IT 人员来说，就像一个支持指南。

内容总揽

参与数据仓库项目的 IT 人员需要技术元数据，这是基于几个不同的目的。如果你是一个数据获取的专家，你所需要的元数据与信息访问开发人员是不一样的。总而言之，项目的技术人员需要了解数据抽取，数据转换，以及数据清洗的处理过程。他们需要从每一个出去作业中知道输出的布局，并理解数据转换规则。

处于 3 个不同的目的，IT 人员需要技术元数据。首先，IT 人员需要技术元数据进行数据仓库的初始开发。假设你负责数据仓库转换处理过程的设计以及开发。出于这个目的，早先数据抽取过程中的元数据将为你的开发提供帮助。

其次，技术元数据对与数据仓库的增长以及维护绝对是关键的。如果你负责对数据结构作修改，或者开发第二个版本的数据仓库，你怎么从众多的内容和过程中找到相关的信息呢？你需要技术元数据。

技术元数据对与对于数据仓库的管理也是关键的。作为一个管理员，你必须监控数据抽取。你必须必争增量装载及时并完全正确。你的责任可能也包括数据库备份和旧数据归档。数据仓库管理几乎离不开技术元数据。

技术元数据举例

技术元数据主要为开发、维护、管理数据仓库的 IT 人员提供支持。相对于商业元数据来说，技术元数据的结构化程度要高的多。技术元数据就像一个使用技术术语显示内部洗劫的数据仓库内部视图。以下列出了技术元数据的一些例子：

- 源系统的数据模型
- 外部源系统的记录布局
- 从数据源到缓存区的映射
- 缓存区域到数据仓库的映射
- 数据抽取规则以及计划
- 数据转换规则以及版本控制
- 数据汇总规则
- 数据清洗规则
- 汇总以及演化
- 数据装载以及刷新计划与控制
- 任务相关性
- 程序名称和描述
- 数据仓库数据模型
- 数据库命名
- 表/视图命名
- 列命名和描述
- 键属性
- 实体和关系的商业规则
- 逻辑和物理模型之间的映射
- 网络/服务器信息

- 连接数据
- 数据转移审计控制
- 数据净化与归档规则
- 访问权限
- 数据使用/时间选择
- 查询以及报表访问模式
- 查询报表工具

请回顾该列表，并针对你的数据仓库构造一份对比列表。

内容重点

例子的列表告诉你一个数据仓库环境中必须包括的技术元数据类型。就像在商业元数据中的例子一样，我们来看看技术元数据能为开发者和管理者回答的一系列问题，请看一下列表：

- 有哪些数据库和表？
- 每个表中有哪些数据列？
- 有哪些键和索引？
- 有哪些物理文件？
- 商业描述与技术描述是否匹配？
- 最后的一次更新是什么时候？
- 源系统以及它们的数据结构是怎么样的？
- 对每一个数据源的抽取规则是怎么样的？
- 你数据仓库中的每一个数据项从数据源到目标的映射是怎么样的？
- 有哪些数据转换规则？
- 如果除去缺失数据，数据项的缺省值是什么？
- 哪些汇总是可用的？
- 有哪一些衍生字段？衍生规则是什么？
- 在查询中的数据项最后更新时间是什么时候？
- 有什么装载与刷新计划？

- 数据净化与归档的频率，对那些数据项进行净化与归档？
- 构建 OLAP 数据的计划是什么？
- 有哪些查询和报表工具是可用的？

谁会受益？

以下列表列出了从技术元数据中获益的人群类型：

- 项目经理
- 数据仓库管理员
- 数据库管理员
- 元数据经理
- 数据仓库架构师
- 数据获取开发员
- 数据质量分析员
- 商业分析员
- 系统管理员
- 基础结构专家
- 数据建模人员
- 安全架构师

如何提供元数据

在设计和构建数据仓库时，需要收集并记录元数据。正如你所知道的那样，元数据从多个角度描述了数据仓库。你通过元数据从数据仓库中找到数据源，理解数据抽取和转换过程，并决定如何浏览数据内容，获取信息。大多数的数据仓库过程是通过软件工具的辅助完成的。每个工具必须能够访问同一个元数据集或是它的子集。

在近期由数据仓库研究所（Data Warehousing Institute）进行的一个研究中，86%的回复认可拥有一个元数据管理策略的重要性。然而，只有 9%实施了元数据解决方案。另外 16%有一个计划并开始实施。

如果大多数有数据仓库的公司都意识到元数据管理的重要性，为什么只有一小部分公司在这方面做工作呢？元数据管理是一个巨大的挑战。挑战不在于通过使用工具来获取元数据，而在于将不同工具生成、维护的元数据集成到一起。

我们将探究挑战在何处。你有什么选择去克服这些挑战，在数据仓库环境中建立有效的元数据管理呢？工业界对此如何处理？在工业界正在寻求标准的时候，你有什么暂时的选择？首先，让我们建立对元数据的基本需求。接下来，我们将在考察将要面临的挑战之前考虑元数据的来源。

元数据需求

元数据应该作为数据仓库用户的路线图。它也必须为开发和管理数据仓库的 IT 人员提供支持。让我们超越这些简单的陈述，看一下元数据管理需求的规范。

获取并存储数据

操作型系统中的数据字典存储了当前的结构以及商业规则。对操作型系统来说，保存数据字典条目的历史信息并不是必要的。然而，数据仓库中的数据历史跨度可能有几年之久（典型的是 5 到 10 年）。在这段时间中，源系统会发生修改，数据抽取方法，数据转换算法，以及数据仓库本身的结构以及内容都可能变化。因此，数据仓库环境中的元数据必须对修改进行跟踪。这样，元数据管理必须通过支持与时间相关特性的版本功能，提供获取和存储元数据的方法。

多种元数据来源

数据仓库元数据不能只有单一的来源。CASE 工具，源操作型系统，数据抽取工具，数据转换中局，数据字典定义，以及其它来源都可以形成数据仓库的元数据。因此，元数据管理必须足够的开放，能够从多个来源获取元数据。

元数据集成

我们考察了商业元数据和技术元数据的元素。你必须将所有这些元素用一种用户能够理解的统一的方式集成起来。来自源系统数据模型的元数据必须能够与来自数据仓库数据模型的元数据集成。最终用户使用的前端工具也需要集成起来。所有这些问题都是困难的，很有挑战性。

元数据标准化

如果你的抽取工具和转换工具存储数据结构，那么两个工具存储的格式必须是一致的。不同工具中存储的同样的元数据必须用同一种方式表示。

修订必须全局可见

如果数据或者商业规则修改的话，元数据也将发生修改。当一个数据仓库处理过程的元数据修改之后，这个修改必须反映到数据仓库的其它处理过程中。

保持元数据同步

关于数据结构，数据元素，事件，规则以及其它事项的元数据必须在所有时间内、在整个数据仓库中保持同步。

元数据交换

当你的最终用户使用前端工具访问信息，他们必须能够显示后端工具（如数据转换工具）记录的元数据。不同工具之间应该可以自由、容易的交换元数据。

支持最终用户

元数据管理必须为最终用户提供简单的图形化或者列表式的展现界面，是最终用户可以从商业的角度浏览元数据，并理解数据仓库中的数据内容。

以上列出的元数据管理需求都是有效的。元数据的集成和标准化是巨大的挑战。然而，在讨论这些问题之前，你首先需要知道元数据的来源。元数据来源列表将帮助你为数据仓库建立一个元数据管理的出发点。

元数据的来源

在不同的数据仓库处理过程中应用工具时，元数据作为一个副产品被记录下来。例如，当使用一个数据转换工具时，从来源到目标映射的元数据在转换工程中被记录下来。让我们看看所有常见的元数据来源。

源系统

- 操作型系统数据模型（手工制作或者来自于 CASE 工具）
- 来自于系统文档的数据元素定义
- COBOL 写字本以及控制块定义
- 物理文件结构以及字段定义
- 程序定义
- 外部数据来源的文件结构和字段定义
- 其它来源（例如电子表格和人工列表）

数据抽取

- 源平台的数据和连接
- 被选择数据源的结构和定义
- 被选择抽取的字段定义
- 在每一个平台上初始抽取文件的合并准则
- 字段类型与长度的标准化规则
- 数据抽取计划
- 增量修改的抽取方法
- 数据抽取任务流

数据转换和清洗

- 抽取文件到数据缓存文件的映射规范
- 独立文件的转换规则
- 字段缺省值
- 有效性检查的商业规则
- 排序安排
- 数据抽取到数据缓存的审计跟踪

数据装载

- 数据缓存文件到装载映像的映射规则
- 为每个文件指定键的规则
- 数据缓存到装载映像的审计跟踪
- 完全刷新的计划
- 增量装载的计划
- 数据装载任务流

数据存储

- 中央数据仓库和独立数据即时的数据模型
- 有多个表组成的主题区域
- 从属数据集市的数据模型
- 物理文件
- 表和列的定义
- 有效性检查的商业规则

信息传递

- 查询和报表工具列表

- 预定义查询和报表的列表
- OLAP 数据库的数据模型
- OLAP 数据的获取计划

元数据管理的挑战

虽然在数据仓库环境中元数据极为重要，但是要无缝的将所有部分的元数据集成起来却是一个相当困难的任务。工业标准现在还远没有形成。除非通过多重的转换工作，否则一个工具生成的元数据不能够被另一个工具所使用。这些挑战迫使很多数据仓库开发者放弃了建立适当的元数据管理的需求。

以下是提供元数据时面临的几个主要挑战：

- 每一个工具都有自己的元数据。如果你在数据仓库中使用了几个工具，你怎么样协调这些格式呢？
- 目前对元数据格式没有通用的工业标准
- 一个中央源数据存储区相对于多个分开的元数据存储区的集合来说要有优势，但是这些优势的描述却相互矛盾
- 在数据从源系统到缓存区域，再到数据仓库的过程中，元数据没有一个公认的传递方法
- 在数据仓库中保持元数据的版本控制的工作相当的繁重，而且很困难
- 在一个有很多源系统的大型的数据仓库中，将数据源相关的元数据统一起来是一项浩大的工作。你必须处理相互冲突的标准，格式，数据命名规范，数据定义，属性，值，商业规则，以及指标单位。你必须解决滥用别名以及使用不正确的数据有效性规则等问题。

元数据存储区

我们可以将元数据存储区看作一个用于分类、存储、管理元数据的，通用的信息目录。我们在前面提到过，商业元数据和技术元数据为不同的目标服务。最终用户需要商业元数据，数据仓库的开发者和管理者需要技术元数据。这两类元数据的结构是不同的。所以，元数据存储区可以看作是两个不同的信息目录，一个存储商业元数据，另一个存储技术元数据。这种分类比将所有元数据都存在一个物理存储区中要更合乎逻辑。

图 9-11 显示了元数据存储区中的典型内容。注意商业元数据和技术元数据的区分。你

注意到一个叫信息导航器的部件了吗？这个部件在商业软件中使用不同方式实现的。信息导航器的功能如下：

元数据存储区

信息导航器

导航数据仓库的内容，浏览数据仓库中的表、属性，查询构造，报表格式，下钻和上钻，报表生成与分发，临时存储结果。

商业元数据

源系统，源系统到目标系统的映射，数据转换商业规则，汇总数据集，用商业数据表达的数据仓库表和列，查询和报表工具，预定义报表，预定义查询，数据装载和刷新计划，支持列表，OLAP 数据，访问权限

技术元数据

源系统数据模型，外部数据源结构，缓存区域文件结构，目标数据仓库数据模型，源系统到缓存区域的映射，缓存区域到数据仓库的映射，数据抽取规则，数据转换规则，数据清洗规则，数据汇总规则，数据装载和刷新规则，源系统平台，数据仓库平台，净化/归档规则，备份/回复，安全

图 9-11　元数据存储区

查询工具界面

这个功能将数据仓库的数据与第 3 方工具连接起来，使这些工具能够从技术元数据中显示元数据定义。

下钻

元数据用户可以下钻，从一个层次到一个更低级别的层次。例如，你可以先得到一个数据表的定义，然后往下一层得到所有的属性，甚至得到某个属性的细节。

回顾预定义的查询和报表

用户可以回顾预定义的查询和报表，并使用适当的参数访问它们。

集中的元数据存储区域可以被最终用户、开发者以及管理员访问，这看起来是元数据管

理的理想方案。但是，集中的元数据存储区域要满足一些基本的需求。我们下面看看这些需求。要找到满足所有这些需求的存储区域工具并不容易。

灵活的组织

允许数据管理员将元数据分类整理，分成逻辑目录和子目录，并将元数据功能模块分配到各个分类中。

记录历史

使用版本控制维护元数据历史。

集成

使用对所有类型的用户都有意义的格式，存储商业和技术元数据。

良好的划分

能够区分并存储逻辑、物理数据模型。

分析和查询能力

能够浏览所有元数据以及关系。

可定制

能够为不同类型的用户生成定制的元数据视图，能够包含新的元数据对象。

维护描述和定义

使用商业和技术术语显示元数据。

命名规则标准化

灵活的应用任何类型的命名规则，在整个元数据存储区域实现标准化。

同步

保持整个数据仓库以及相关外部系统的元数据同步。

开放

通过工业标准界面，支持不同处理过程的元数据交换，兼容不同的工具。

选择一个合适的元数据存储工具，是一个项目团队必须要做的关键决策。你可以使用上面的准则列表作为评估元数据存储工具的指南。

元数据集成与标准

为了在数据仓库中不同的软件工具以及处理过程的元数据可以自由的交换，显然需要一

个标准。在第 3 章我们提到，元数据组织（Meta Data Coalition）和对象管理工作组（Object Management Group）正在做这方面的工作。元数据组织采用了一种称为开放信息模型（OIM）的标准。对象管理工作组则发布了通用仓库源数据模型（Common Warehouse Metamodel）作为它的标准。两个团体已经宣布了他们将会融合它们的标准，最终将只有一个工业标准。

你应该意识到这些努力最终将得到一个元数据的工业标准。同样的，请注意以下与元数据相关的要点：

- 标准模型提供了数据仓库环境中数据库模式管理、设计和重用的元数据概念。它包括逻辑和物理数据库概念。
- 模型包括了用于填充数据仓库的数据转换的细节。
- 模型可以扩展，从而包括获取数据立方体描述的 OLAP 特定元数据类型。
- 标准模型包括了指定元模式和目标模式以及它们间的数据转换。这些类型的元数据可以用于支持转换设计，影响分析（特定的模式变化将影响转换），以及数据体系（使用数据源和转换产生数据仓库中的特定数据）。
- 标准模型的转换模块获取关于数据转换脚本的信息。独立的转换与来源和目标的转换相关。一些转换的语义可以通过约束和对由表驱动映射的编码-解码来获取。

实施选项

我们对数据仓库环境中的元数据的重要性已经谈的够多了。我们也讨论了元数据的集成和标准。然而，现实是目前还缺少元数据的行业标准。那么，在一个使用多种不同厂商工具的典型的数据仓库环境中，有什么方法实施元数据管理呢？在这个部分，我们将讨论一些选项。我们希望通用的标准能尽快实现。

请看下面的选项并考虑哪一种最适合你的数据仓库。

- 选择并使用一个有商业信息目录模块的元数据存储产品。与此兼容的信息访问和数据获取工具与之无缝连接。对其它不兼容的工具，你将使用其它集成方法。
- 某些数据仓库顾问认为，一个集中的元数据存储区是一种有限制方法，它将危害到独立的处理过程的自治。虽然一个集中的元数据存储区是的元数据可以共享，但是在一个大型数据仓库中，它将难以管理。但是，在一个非集中化的情况下，元数据存储在体系结构的不同部分。元数据交换将会成为一个问题。

- 一些开发者有自己的解决方案。他们将数据仓库中每个工具的标准用法形成一组过程，并提供一个内容目录。
- 其它开发者生成它们自己的数据库，收集、存储元数据，在公司的内部网中发布。
- 一些开发者使用了一种聪明的方法，它们将信息访问和分析工具集成起来。他们将不同工具的元数据一同显示。有时候，可以使用从中央存储区导出的元数据作为查询工具的帮助文本。

目前的趋势是在报表和 OLAP 中使用 Web 技术。公司的内部网已经广泛的用于信息传递。图 9-12 显示了这个技术对访问元数据的方法的改变。商业用户可以使用它们的 Web 浏览器访问元数据并浏览数据仓库和数据集市。

特别注意数据仓库的元数据。请准备回答下列问题的元数据：

在你的企业中，元数据的目标是什么？

要达成这些目标，需要什么元数据？

元数据的来自何处？

谁来维护元数据？

他们怎么维护？

元数据的标准是什么？

怎么使用元数据？谁来使用？

需要什么元数据工具？

你应该在你的环境中设立元数据目标。

Web 客户端　Web 服务器　　数据仓库数据/元数据存储

Web 客户端　浏览器

图 9-12　元数据：基于 Web 的访问

本章总结

- 元数据对于使用、构建、维护数据仓库是关键的

- 对于最终用户来说，元数据就像一幅数据仓库内容的路线图。
- 对 IT 专家来说，元数据支持开发和管理功能。
- 元数据在数据仓库扮演一个活跃的角色，它辅助流程的自动化。
- 元数据类型可以按照数据仓库的 3 个功能区域分类，分别是数据获取，数据存储，以及信息传递。这几个类型与这 3 个区域中的处理流程相联系。
- 商业元数据与数据仓库商业用户相联系。技术元数据为负责开发和管理的 IT 专家服务。
- 有效的元数据必须满足一系列的需求。元数据管理是困难的，需要面对许多挑战。
- 目前还没有一个通用的元数据标准。缺乏标准阻碍了元数据的无缝传递。
- 元数据管理区就像一个包括了几个扩展功能的通用信息目录。
- 一个元数据实现选项包括商业元数据存储产品的使用。也有其它自创的解决方法。

思考题

1. 你为什么认为在数据仓库环境中元数据很重要？用一到两段文字解释。
2. 解释元数据为什么对数据仓库开发和管理是关键的？
3. 解释元数据是一个神经中枢的概念。描述概念如何应用到数据仓库环境中？
4. 列出 3 个元数据对最终用户相当关键的主要原因。
5. 为什么元数据对 IT 来说是关键的？列出 6 条原因并解释。
6. 选择 3 个元数据辅助自动化的处理过程。解释元数据是如何在这些过程中扮演重要角色的。
7. 建立信息上下文由什么意义？简要的用一个例子进行解释。
8. 在数据获取、数据存储、信息传递 3 个区域中，对每一个区域列出 4 种元数据类型。
9. 列出 10 个商业元数据的例子。
10. 列出 4 个元数据的主要需求。描述每一个需求的内容。

复习题

1 指出下面命题的对错。

A. 元数据在数据仓库中的重要性与在操作型系统中相同。

B. IT 人员进行数据仓库管理需要元数据。

C. 技术元数据的结构化程度不如商业元数据。

D. 维护现代数据仓库中的元数据只要文档即可。

E. 元数据提供了预定义查询的信息。

F. 商业元数据的来源比技术元数据的来源要多。

G. 商业用户和技术人员都使用技术元数据。

H. 元数据存储就像一个通用的目录工具。

I. 元数据标准方便了不同工具间的元数据交换。

J. 商业元数据只给商业用户服务；商业元数据不能被 IT 人员理解和使用。

2 作为一个国内软饮料生产商的数据仓库项目经理，你的任务是写一个编写元数据的计划。考虑你如何实施一个元数据策略，有什么可能的选择以及需要的内容。

3 作为一个数据仓库管理员，描述你工作中需要用到的元数据类型。解释这些不同类型的元数据是如何辅助你的工作的。

4 你负责训练你的数据仓库最终用户。写一个一般用户使用商业元数据执行查询的过程描述。请不要使用“元数据”这个词，而是用用户的术语描述。

5 作为一个数据获取专员，哪一些元数据可以帮助你呢？选择一个数据获取处理过程，解释元数据在其中的作用。

第十章　维度建模的原则

本章目标

- 清楚的理解需求定义是如何影响数据设计的
- 介绍多维模型，并将它与实体关系模型进行比较
- 回顾星形模式的基本知识
- 找出事实表以及维度表中的内容
- 理解在数据仓库中应用星形模式的好处

从需求到数据设计

需求的定义完全驱动着数据仓库的数据设计。数据设计包括将所有的数据结构进行集中。一个数据结构是由一组数据元素结合而成的。逻辑数据设计包括决定多种需要的数据元素，以及将这些数据元素组合成数据结构。逻辑数据设计也包括在数据结构之间建立关系。

让我们看看图 10-1。注意整个过程是由需求收集开始的。需求收集阶段的结果是记录的很详细的需求定义文档。这个文档的一个关键部分是一些信息包图的集合。记住，这些信息表格显示了表、商业维度和这些维度内部的层次结构。

信息包图形成了数据仓库逻辑数据设计的基础。数据设计流程最终的结果是一个多维的数据模型。

图 10-1　从需求到数据设计

需求收集　　需求定义文档　　信息包　数据设计　　多维模型

设计决策

在开始设计多维模型之前，让我们快速的回顾一些必须要作出的设计决策：

选择流程。要为初始的逻辑结构从信息包中选择主题。

选择粒度。决定数据结构中数据的详细程度。

识别维度，并整合维度。选择初始逻辑结构中的商业维度（例如产品，市场，时间等等），

确保每一个维度中的数据元素相互之间保持一致。

选择事实。选择初始逻辑结构中的计量单位。

选择数据库的持久度。决定你应该保存多长时间的历史数据。

维度建模基础

我们所需要的商业维度需要转变成逻辑数据模型，多维建模的名字由此而来。它是一种构造需要分析的商业维度和指标的逻辑设计技术。这种模型已经被证明在查询和分析方面又很高的性能。

我们所讨论的多维信息包图是维度建模的基础。因此，维度建模包括了表示商业维度所需要的数据结构。这些数据结构也包含了指标和事实。

在第 5 章，我们详细的讨论了信息包图。我们着重考察了一个汽车销售商的信息包图。请回到图 5-5，回顾一下该图的内容。在这张图中，你看到了什么？在图表的底部，你看到了那个汽车销售商所希望分析的指标的列表。接下来，看看列的名称。这些是汽车销售商所希望分析的指标所依赖的商业维度。在每一个列的标头，你可以看到商业维度中维度的层次结构和目录。

回顾汽车销售商的信息包图，我们注意到 3 类的数据实体：（1）指标与单位，（2）商业维度，以及（3）每一个商业维度的属性。所以，当我们将维度模型放到一起来表示汽车销售商的信息包图中多包含的信息是，我们需要相应的数据结构来表示这三种数据实体。下面我们讨论一下具体的做法。

首先，让我们处理在信息包图底部的指标与单位。这些是分析所需要的事实。在汽车销售商的信息包图中，事实包括以下这些：

实际的销售价格

MSRP 销售价格

选件价格

全部价格

经销商附加部件

经销商信用

经销商发票

付款数量

制造商收益

金额数量

这些数据中的每一项都是一个指标或事实。实际销售价格是关于销售中实际价格的事实；全部价格是关于与销售相关的所有物品的全部价格。在我们回顾这些事实项目时，我们发现可以将所有这些信息都归集到一种数据结构中。用关系数据库的术语来说，你可以将这种数据结构称为关系表。关系表由信息包图中的事实或指标构成。在这个场景中，这张事实表称为汽车销售事实表。

让我们来看看图 10-2，它显示了事实表是如何构成的。事实表的名称来自与分析的主题；在这个案例中，它称为汽车销售。每一个事实项目或者指标都作为汽车销售事实表的一个属性。

图 10-2　汽车销售事实表的构成

我们已经确定了汽车销售模型中的一种数据结构，并从信息包图中得到了事实表。现在，让我们讨论信息包图另外的部分，逐个的讨论商业维度。请看在图 5-5 中的产品商业维度。

如果我们需要按照产品来分析事实的话，就需要用到产品商业维度。有时候我们的分析可以分解到单独的产品模型。在另外的一些情况下，分析只需要在产品线这个层次上进行就可以了。而有时候分析甚至要在更高的层次（如产品目录）上进行。与产品维度相关的数据项如下：

模型名称

模型年份

部件风格

产品线

产品目录

外部颜色

内部颜色

第一个模型的年份

我们怎么样来处理维度模型中所有这些数据项呢？所有这些产品相关的数据都必须由某种方式来处理。所以，我们可以将所有这些数据项归集到一种数据结构或者说一个关系表

中。我们可以将这个表称为产品维度表。上面所列到的数据项是这个维度表的属性。

深入信息包图，我们将其它的商业维度用列标题来表示。在汽车销售信息包图中，这些其它的商业维度分别是经销商，客户的人口统计属性，付款方式，时间。就像我们构造产品商业维度的那样，我们可以构造出剩下的维度表——经销商，客户的人口统计属性，付款方式，和时间。这些在每一列中出现的数据项将会成为相应的每一个商业维度的属性。

图 10-3 将所有这些信息集成到一起。它显示了如何由信息包图构造不同的维度表。请仔细观察图 10-3，看看每一个维度表的构成。

图 10-3　汽车销售维度表的构成

到目前为止，我们已经构造了事实表以及维度表。那么这些表在维度模型中应该如何安排呢？它们在模型中的关系是怎么样的？我们又如何标记这种关系呢？维度模型主要用于满足查询与分析的需求。查询与分析有哪一些类型呢？通过多个维度表的维度属性，查询事实表中的指标就是典型的查询与分析。

让我们考察一下一个针对汽车销售数据的典型的查询。2000 年版的切诺基(Cherokee)吉普，在 2000 年一月份，通过 Big Sam 汽车经销商卖出的，客户是已婚，并通过 Daimler-Chrysler 金融机构提供 3 年贷款，符合以上条件的交易有多少销售收益？我们在分析实际的销售价格，MSRP 销售价格以及全部价格。我们需要通过多个维度表分析所有这些事实。维度表中的属性在我们的查询中扮演了约束和过滤的角色。我们也发现在一个查询中可以包含每个商业维度表的部分或者全部的属性。而且，每一个维度表都可能成为查询的一部分。

在我们决定如何在维度模型中安排事实和维度表并标记关系之前，让我们复习一下维度模型需要达到什么样的目标。以下是一些将关系表合成为维度模型的准则：

- 模型应该为数据访问最好的方式
- 整个模型必须是以查询为中心的
- 它必须为查询和分析进行优化
- 模型必须显示事实表与维度表之间的相互作用
- 整个结构必须使每个维度都能够有相等的与事实表交互的机会
- 模型应该允许沿着维度的层次结构下钻与上钻

通过检查这些需求，我们发现将事实表在中间，维表安排在事实表的四周能够满足以上

的条件。在这种安排方式下，每一个维表都与中间的事实表有直接的联系。这样做是必要的，因为每一个维表以及它的属性必须要又均等的机会参与到一个分析事实表的查询当中。

这样维度模型看起来像是一种星形，事实表位于星形的中央，而各个维度表分布在星形的各个角上。因此，我们将维度模型称为星形模式。

让我们看一下图 10-4 中汽车销售的星形模式。销售事实表在中央。各个维度表分布在星形模式的四周，他们分别是产品维表，经销商维表，客户人口统计特征维表，付款方式维表，以及时间维表。每一个维表，都与事实表有一个“一对多”的联系。换句话来说，在产品维表中的每一条记录，在事实表中都有一条或者多条记录相对应。

图 10-4　汽车销售的星形模式

E-R 建模与维度建模的对比

我们对操作性的，或者称为 OLTP 系统的数据建模都比较熟悉。对于这些系统，我们采用实体-关系（E-R）建模技术进行数据建模。图 10-5 列出了 OLTP 系统的特征，并显示了为什么 E-R 建模适用于 OLTP 系统。

- OLTP 系统捕捉事件或者交易的详细信息
- OLTP 系统关注独立的事件
- 一个 OLTP 系统是通向微观交易的窗口
- 反映运行业务所需要的细节信息
- 仅仅适用于回答交易级别的问题
- 数据一致性，非冗余性，以及高效的数据存储是最重要的

实体-关系建模

避免了数据冗余

确保数据一致

表达微观的关系

图 10-5　OLTP 系统中使用的 E-R 建模

到目前为止，我们讨论了维度建模的基础知识，并这种这种模型是最适于为数据仓库系统进行数据建模的。让我们概括一下数据仓库信息的特征，并回顾一下维度建模是如何适用于这些目标的。让我们看见图 10-6。

- 数据仓库需要回答全局问题
- 数据仓库关注经理如何管理业务问题
- 数据仓库反映商业趋势
- 信息是围绕商业流程来组织的
- 答案显示了业务是如何衡量流程的
- 通过几个商业维度，可以衡量业务情况

维度建模

捕捉关键指标

通过维度显示

面向商业用户

图 10-6　面向数据仓库的维度建模

使用 Case 工具

在数据建模方面，有许多的计算机辅助系统设计工具（CASE）。在第 8 章，我们介绍了这些工具以及他们的特性。你可以使用这些工具来建立面向特定数据库管理系统（DBMSs）的逻辑数据模型以及物理数据模型。

你可以使用一个计算机辅助系统设计工具来定义表，属性以及关系。你可以制定主键和外键。可以构造实体关系图。所有这些都可以通过图形用户界面以及强大的拖-拽（drag-and-drop）方式轻松完成。在建立了一个初始模型之后，你可以加入或者删除字段，改变字段的特征，建立新的关系，以及轻松的做出多次的修订。

在这些计算机辅助系统设计工具中，另一个很有用的功能就是“正向工程”，以及产生你所需要的目标数据库的模式。通过这些计算机辅助系统设计工具，你可以很轻松的完成“正向工程”。

在为数据仓库建模时，我们对维度建模相当关注。许多现有的厂商都对他们的建模工具进行扩展，包含了维度建模。你可以建立事实表，维度表，并构建每个维度表与事实表之间

的关系。结果就是星形模式的模型。然后，你可以将星形模式正向工程，产生你所选择的数据管理系统相应的关系模式。

星形模式

现在你已经学习了星形模式的基本知识，让我们通过一个简单的例子来看看这种模式的特征。构建星形模式是数据仓库建模的基础数据设计技术。对于数据仓库建设人员来说，很好的掌握这项技术是必须的。

一个简单的星形模式的回顾

我们将学习一个简单的星形模式，这个星形模式是为一个订单分析设计的。假设这是一个制造业公司的数据模式，市场部门关心他们应当如何处理公司所接到的订单。

图 10-7 显示了这个简单的星形模式。它包括了模视图中间的订单事实表，围绕事实表的是四个维度表——客户，销售人员，订单日期，以及产品。让我们开始分析这个星形模式。先从市场部门的观点来看这个结构。这个部门的用户会使用金额、成本、边际利润以及销售数量等指标进行分析。这些信息能够在事实表的结构中找到。用户会对总数按照客户、客户、销售人员、订单日期、以及产品进行分解，对这些指标进行分析。所有这些用户可能分析的维度都能够在维度结构中找到。这个星形模式能够容易的被用户理解并使用。这个结构是用户通过商业维度察看关键指标方式的一种映射。

如果你要察看订单金额，星形模式结构直观的回答了关于是什么、什么时候、谁出售、谁购买等问题。通过星形模式，用户可以轻松的将这些问题的答案形象化：对应一定数额的销售，究竟销售了什么产品？谁是它们的顾客？是由那个销售人员进行的？订单是什么时候发出的？

向数据仓库发出一条查询后，查询的结果是通过将一个或者多个维表连接到事实表之后产生的。连接是在事实表与一个或者多个为表之间产生的。事实表中的某一行于每个维表的多行建立关系。这些单独的关系可以清晰的表示为星形模式中的尖峰部分。

下面我们对星形模式进行一个简单的查询。假设营销部门希望得到 Maine 州的客户购买的，并由销售人员 Jane Doe 在 6 月份销售出去的，产品 bigpart-1 的销售数量以及订单金额。图 10-8 显示了这个查询是如何通过星形模式声长的。通过星形模式，查询的约束与过滤可以很容易的被理解。

图 10-8　通过星形模式，理解一条查询

钻取总数得到在更低的层次上的细节数据，是一种通用的分析类型。让我们假设营销部门有一个特别的分析需求，它可以用以下的查询表示：显示 1999 年在东北地区客户购买的 big parts 品牌产品的总数。在下一步的分析中，市场营销部门希望对相同品牌的产品钻取到东北地区在 1999 年每一个季度的级别。下一步分析需要深入到该品牌中独立的产品这一级别。最后，分析需要深入到东北地区每一个单独的州的细节级。通过回顾星形模式，我们可以洞悉所有这些钻取操作。我们可以参考图 10-9，看看如何从星形模式中进行钻取操作。

钻取步骤　　步骤 2　步骤 3　步骤 4

图 10-9　通过星形模式理解钻取分析操作

维表的内容

正如我们已经看到的，维表的集合是星形模式中的关键部分。这些维表表示了分析所需要依赖的商业维度。让我们观察一个维表，并分析它的特征。请看图 10-10 并回顾以下的观察。

--

大数值的数字属性

文本属性

非直接相关属性

非规范化的，

具有上钻/下钻的能力

多级层次结构

相对少的记录

图 10-10　　深入维表

--

维表键。维表的主键，可以维一的确定每一条记录

维表很宽。一般来说，一个维表会有相当多的属性/字段。有一些维表有 50 个以上的属性并非是不寻常的。所以，我们说维表很宽。如果你将表格的列排列一下，维表将会水平展开。

文本属性。在维表中，很难找到任何用于计算的数值数据。维表中的属性一般是文本格式的。这些属性表示商业维度中的部件的文本描述。用户可以采用这些描述构造它们的查询。

非直接相关属性。维表中的某一些属性经常不会与其中的其它属性直接相关。例如，包裹的大小与商品的品牌不是直接相关的；然而，包裹的大小与产品的品牌可能都是产品维度的属性。

非规范化。维表中的属性通常会被各种查询语句一次一次的使用。在一条查询中，一个属性作为一个约束，直接应用于事实表中的指标。为了使查询更有效率，如果查询直接从未表中获得一个属性，然后直接查询事实表，而不是通过其它中间表（那样会导致低效率），将会有最高的效率。如果对维表进行规范化，你将需要建立这一类的中键表，从而导致低效率的查询。因此，维表是扁平的，而不是规范化的。

上钻/下钻。维表中的属性提供了获取从高层次的汇总信息到低层次细节信息的能力。例如，一个层次结构中有三个属性，邮政编码，城市，以及州。你可以得到按州为单位销售总量，然后从下钻到以城市为单位的，以及以邮政编码为单位的销售总量。另一方面，你可以首先得到以邮政编码为单位的销售总量，然后上钻到城市，以及州。

多级层次结构。以客户维举例，该维表存在一个包括客户所在邮政编码号，城市以及州的单一的层次结构。但是，维表通常会有多级的层次结构，从而使钻取可以沿着这些层次结构中的任何一个进行。以商店的产品维表为例。在这种情况下，市场营销部门可能自己制定将产品分类到不同的产品目录以及产品部门的方法。另一方面，会计部门也许会用另外的方式将产品归类到产品目录以及产品部门中。所以，在这个案例中，产品维表会有多套属性，包括市场营销-产品-目录，市场营销-产品-部门，财务-产品-目录，财务-产品-部门。

相对少的记录。一个维表中的记录数通常会比事实表中的记录数量要少。一个汽车厂上的产品维表可能只有 500 行。另一方面，事实表可能会有上百万行记录。

事实表的内容

现在让我们关注一个事实表并考察其中的组成部分。记住，事实表是我们存储指标的地方。我们可以保存级别尽可能低的细节。在销售分析用的部门事实表中，我们可以通过按照每一条收款机的记录存储单独的交易，来保留所有销售信息。有一些事实表可能只包含了汇总数据。这些事实表称为聚集事实表。图 10-11 列出了一个事实表的特征。让我们来看看这些特征：

--

连接的事实表键。

数据颗粒。

完全的加和的指标。

半加和的指标。

大量的记录。

只有一些属性。

稀疏的数据。

退化的维度。

图 10-11　　事实表的内容

--

连接的事实表主键。

事实表中的一行记录与所有维表中的相应记录相关。在这个事实表例子中，你可以发现订单的数量作为一个属性。假如维表是产品，时间，客户，以及销售代表。对这些维表，假设维的层次结构中最低的层次分别是单独的产品，日历中的一天，一个特定的客户，以及一个单独的销售代表。那么事实表中的一行就必须与特种特定的产品，一个特定的日期，一个特定的客户，以及一个单独的销售代表相关。这意味着事实表中的记录必须可以由四个维表

中的主键所唯一确定。由此，事实表中的主键必须是所有维表主键连接起来的组合键。

数据颗粒。

这是事实表的一个重要的特征。正如我们所知道的，数据粒度是指标的细节程度。在这个例子中，指标是在最细节的层次的。订单数量与以下的特征相关：一个单独的订单中某种特定产品的数量，某个特定的日期，一个特定的客户，以及一位特定的销售代表。如果我们只保留每个月中某种特定产品的销售量，那么数据颗粒就会与现在不同，而处在一个更高的级别。

完全加和指标。

让我们看看这些属性：*order_dollars, extended_cost, quantiti_ordered*。每一个都与一个特定的产品，一个特定的日期，一位特定的客户以及一位特定的销售代表相关。在一个查询当中，我们假设用户希望得到某一个特定日期中某个州的客户对一个特定产品的订单总和。注意，不是一个特定的客户，而是一个特定州中的所有客户。然后，我们需要找到事实表中所有与这个州相关的所有客户，将 *order_dollars, extended_cost, quantiti_ordered* 这三个字段的数值加和。这些属性的数值可以简单的汇总。这样的指标称为完全加和指标。完全加和指标可以通过简单的加法进行汇总。当我们运行汇总事实表中指标的查询时，我们将不得不确认这些指标是完全加和的。否则，将可能的不到正确的汇总数据。

半加和的指标。

考虑事实表中的 margin_dollars 属性。举例来说，如果 order_dollars 时 120，extended_cost 时 100，那么 margin_percentage 就是 20。这是通过计算 order_dollars 和 extended_cost 之后得到的指标。如果你汇总事实表中某个州中所有客户的数值，你不能够将这些记录中的 margin_percentages 加起来就得到汇总数据。类似于 margin_percentage 这种衍生指标是非加和的。它们称为半加和的指标。在执行汇总查询的时候，应当将半加和的指标和完全加和指标区分开来。

表很长，但是不宽。

通常一个事实表的属性要比一个维表少。一般来说只有 10 个甚至更少的属性。但是比

较之下，事实表中的记录的数量是相当巨大的。我们举一个简单的例子，如果维表中有 3 种产品，5 个客户，30 天，10 个销售代表。那么，在这么少的维表记录数量下，事实表中的记录数量将会有 4500 条。相对于维表的记录数量来说，这个数字是相当大的。如果你将事实表展开成 2 位的表格，你会发现，维表只有很少的几列，但是有相当多的记录。

稀疏的数据。

我们讨论过事实表中的每一行都与一个特定的产品，一个特定的日期，一位特定的客户以及一个独立的销售代表相关。换句话说，对一个特定的产品，一个特定的日期，一位特定的客户以及一个独立的销售代表，事实表中都由一个相应的记录。如果在一个临近假日的日子里没有接到订单或者没有订单被处理，哪会发生什么呢？在这种日子中，事实表不会有相关的记录。而且，其它维表也可能会导致事实表中出现空的指标值。那么，我们是否需要将这些空的指标值放在事实表中呢？没有必要。因此，很重的一点就是要意识到这一类的稀疏数据，并理解事实表中可能会存在数据隔断。

退化的维度。

仔细的观察例子中的事实表。你会发现有 order_number 和 order_line 属性。这些属性不是指标，也不是事实。那么为什么这些属性会在事实表中呢？当你从操作形系统中获取维表以及事实表的属性时，你会发现操作型系统中某些数据元素既不是事实也不是严格的维属性。例如，这些属性是参考数字，例如订单数量，发票金额，订单流水号，等等。这些属性在某些类型的分析中是有用的。例如，你可能在寻找每个订单中某种产品的平均数量。然后你就需要将产品与订单中的数量联系起来，计算平均值。诸如例子中 order_number 和 order_line 的属性称为退化的维度，它们依然作为属性保留在事实表中。

不含事实的事实表

除了连接主键之外，一个事实表还包含事实或者指标。假设我们正在构建一个跟踪学生出勤情况的事实表。分析学生出勤情况，所要用到的维度可能是学生，课程，日期，课室，以及教授。出勤可能会被这些维度中的任何一个所影响。在需要标记出学生与一个特定课程，日期，课室，以及教授相关的出勤情况时，你需要什么样的指标来记录出勤事件呢？在事实表的记录中，出勤可以用数字 1 来表示。每一个事实表记录会包含数字 1 以表示出勤。如果

是这样的话，为什么还要在每一行中都记录数字 1 呢？没有必要这样做！事实表中相应的每一行都可以表示出勤的事实。当事实表表示事件的时候，就会出现这种情况。这种事实表不需要包含任何事实。它们是“不含事实”的事实表。图 10-12 显示了一个典型的布汗事实的事实表。

--

事实表中含有指标或者事实。然而，即便在事实表中没有任何指标或者事实的情况下，一些商业事件或内容也可以被事实表所表示。

图 10-12 不含事实的事实表

--

数据粒度

到目前为止，我们知道粒度表示事实表中的详细程度。如果事实表有最低粒度的颗粒，那么事实或者指标就处在从操作型系统中所能够得到的最低级别。事实表保存最低粒度的颗粒有什么好处吗？有没有需要折衷的呢？

如果将事实表的数据保持在最低粒度，用户通过数据仓库可以下钻到最低级别的细节数据，而不需要访问操作型系统。事实表的基本级别必须是所有相应维度的最低级别。这样，上钻与下钻的查询就可以执行得很有效率。

那么，什么是相应维度的自然最低级别呢？在产品、日期、抗税与销售代表四个维的例子中，自然的最低级别分别就是一个单独的产品，一个特定的日期，一个特定的客户以及一个特定的销售代表。所以，在这个例子中，事实表中的一个记录应该包含于某个单独的产品，某个特定的日期，某个特定的客户以及某个特定的销售代表相关的最低粒度的指标。

假设我们希望在销售代表维加入一个新的地区属性。由于事实表中已经是最低粒度的销售代表数据，所以这个修改不需要事实表的任何修改。这是一个“优美的”改动，因为所有其它的查询都不需要任何的修改，依然能够正常的运行。类似的，假设我们希望加入一个新的促销维度。这样你就必须彻底改动事实表，以包括这个新的维度。这样，事实表的数据颗粒将依然保持最低的级别。即使是这样，以前的查询语句依然不需要做任何的改动。这也是

一个“优美的”改动。最低粒度的事实表会便于“优美的”扩展。

但是，我们为了保持事实表有最低级别的数据颗粒，必须付出存储和维护方面的代价。最低级别的数据颗粒意味着大量的事实表记录。然而，在实际情况中，我们会构建汇总事实表来支持汇总数据的查询。

使用低粒度事实表有两个好处。低粒度事实表作为当前的操作性数据的自然的目的地，可以频繁的从操作型系统中抽取出来。而且，有很多的数据挖掘应用需要最低级别的数据颗粒。而数据挖掘应用的数据来自于数据仓库。

星形模式的键

图 10-13 显示了维表以及事实表中键的构成。

--

事实表：组合的主键，每一部分对应一个维

维表：生成主键

图 10-13 星形模式的键

--

主键

一个维表的每行都可以维表中的主键所唯一的识别。在一个产品维表中，主键唯一的标识一个产品。在一个客户维表中，客户号码唯一的标识每一个顾客。类似的，在销售代表维表中，身份证号码唯一的标识每一个销售代表。

我们将这些可能的维表候选键提取出来。现在，我们将考虑这些维表候选键的实现时要注意的地方。假设在操作型系统中产品代码时 8 位代码，其中两位代表了产品所在仓库的代码，另外两位表示产品的类别。如果我们使用操作型系统的产品代码作为产品维表的主键会出现什么情况呢？

数据仓库包含了历史数据。假设由于产品存放到了另外一个仓库，因而产品代码在某一年的中间发生了改变。这样，我们就不得不改变数据仓库中的产品代码。如果产品代码是产品维表中的主键，那么相同产品的新数据就必须以不同的键值存放在数据仓库中。这样，如果我们要将代码变化之前以及代码变化之后相同产品的数据进行汇总的话，就会出问题。是什么造成了这个问题呢？问题在于使用操作型系统的主键作为维表的键。

替代键

我们怎么样才能够解决上面所提到的问题呢？我们能不能使用操作型系统中的主键作为维表的键呢？如果不能，又有什么候选键呢？

在选择维表主键的时候，应该遵循两个原则：第一个原则来自于产品被存放到另外一个仓库的情况。换句话来说，操作型系统中的产品键是有内在含义的。产品键中的一些位标识了所在的仓库，另外一些键则标识了产品目录。这些键是有含义的。所以必须第一条原则就是：避免维表中的键有内在的含义。

在一些公司，一些顾客并不在公司的客户名单中。他们可能早在几年之前就离开了相应的公司。那么，这些离去的顾客的代码很可能很被分配给了一些新的顾客。现在，假设我们使用操作型系统的顾客键作为顾客维表的主键。那么，我们就会碰到问题——同样的顾客代码可能既和离去的顾客相关，也和新的顾客相关。离去顾客的数据仍然可能会用于按照城市和州进行的汇总。所以，必须第二条原则就是：不要适用生产系统中的主键作为维表的主键。

那么，我们应该用什么来作为维表的主键呢？答案就是替代键。替代键就是简单的系统生成的序列号码。它们没有任何的内在含义。当然，替代键可以映射为生产系统的键。但是，它们之间是不同的。通常的做法是将生产系统的键作为维表的附加属性。请回到图 10-13，STORE KEY 是商店维表的替代主键。操作型系统中的主键可能作为一个附加的非键属性存放在维表当中。

外键

每一个维表都于中央的事实表有一个“一对多”的关系。所以，每个维表中的主键必须是事实表中的外键。如果有四个维表，产品、日期、客户以及销售代表，那么这四个表中的主键必须作为订单事实表中的外键。

让我们重新检查一下事实表的主键。一共有 3 种选择：

1 一个单独的复合主键，其长度为四个维表键的总和。在这种情况下，除了复合主键之外，外键必须作为附加的属性保存在事实表中。这种情况增加了事实表的大小。

2 连接的主键，有四个维表的主键连接而成。这样，你就不需要将维表的主键作为附加的属性存放在事实表中了。主键中的每一个独立的部分都可以充当外键。

3 一个生成的主键，与维表的键无关。除了生成的主键之外，维表中的所有主键都必须作为

外键，存放在事实表中。这种方式同样增加了事实表的大小。

在实际情况中，大多数的事实表采用第 2 种方式。这种选择使你可以很容易的将事实表中的记录于维表联系起来。

星形模式的优势

当你考察星形模式是，你会发现每个维表与事实表之间都有一个“一对多”的关系。为什么星形模式会如此特别呢？为什么说这种模型特别适合数据仓库呢？这种模式被广泛而成功的应用于查询处理的优化方面，这又是什么原因呢？

虽然星形模式是一个关系模型，但是它不是一个规范化模型。在星形模式中，维表被故意的非范式化了。这是星形模式与 OLTP 系统中的关系模式的基本区别。

在讨论星形模式的一些显著的优势之前，我们需要注意的是，严格的坚持使用这种模式并不总是最好的选择。例如，如果客户是一个维度，而这个企业很多的客户，那么，一个非规范化的客户维就不是一个好的选择。一个很大的维表可能会相应的增加事实表的大小。

然而，星形模式的优势远远大于它的不足。让我们回顾一下星形模式的优势：

用户容易理解

OLTP 系统的用户使用预先定义好的用户界面或者查询语句模版，和应用系统进行交互。他们没有必要理解后台的数据结构。这些数据结构和数据库模式将留给 IT 专家来处理。

决策支持系统（例如数据仓库）的用户就不一样了。他们需要自己构造查询语句。在使用第三方工具与数据仓库进行交互的过程中，他们要知道自己究竟需要什么。他们必须对数据仓库中的有什么数据非常的熟悉，必须理解数据的结构，以及在整个模式中这些结构之间的相互关系。

星形模式确切的反映了用户是如何想的，他们在查询和分析时需要什么数据。他们从商业的角度来考虑问题。维表包含了用户经常查询和分析的属性。在你向用户解释产品 A 的数量存放在事实表中，并指出这部分数据与维表的关系时，用户已经能够理解这种关系了。这是因为星形模式完全按照与用户相同的理解关系的方式定义了连接的路径。星形模式很容易被用户理解。

假如要一位用户理解 OLTP 系统中的关系模式，在这种情况下，你必须解释一堆的规范化的表，有时需要面对几个表，一个一个的解释，才能够得到一个很小的数据集。星形模式

在简单性方面，显然要胜出一筹。用户可以很容易的理解结构以及关系。

星形模式在实施后有显然的优势。然而，即使在开发阶段，这种优势也不能忽视。由于用于可以很容易的理解星形模式，在数据仓库的开发过程中与用户交流就变得很容易了。

优化浏览

在一个数据库模式中，在数据实体之间建立关系是出于什么目的呢？关系用于连接不同的表，以得到你所需要的信息。关系提供了操作数据库的能力。使用连接路径，可以将表与表之间的数据联系起来。

如果连接路径很复杂、很费解，那么你要在数据库中浏览将会是缓慢而艰难的。另一方面，如果连接路径简单、清晰的话，浏览数据库将会变得更快速，而且是优化的。

星形模式的一个主要的优势在于它优化了对数据库的浏览。即使你要找到查询结果看上去很复杂，但是查询的过程是简单而且清晰的。让我们看一个例子，理解它是如何工作的。请看图 10-14，它显示了一个分析汽车缺陷的星形模式。假设你的汽车经销商服务经理在卖通用公司的汽车。你会注意到 2000 年一月份的白色 Corvettes 汽车会有较高的事故率。你需要一个工具来分析这些缺陷，查出潜在的原因，并解决这些问题。

在星形模式中，缺陷的数字作为指标保存在中央的事实表中。时间维包含了模型的年份。部件维中有一部分信息；例如，喷涂成乳白色。问题维包含了问题的类型；例如，喷涂层脱落。产品维包含了汽车的型号、组成、以及装饰情况。供货商维包含了部件供货商的数据。

现在我们看看怎么样确定导致喷漆为乳白色的 Corvettes 汽车喷涂层脱落的供货商是谁。看看 4 个从维表指向事实表的箭头。这些箭头显示了如何通过产品维得到与 Corvette 相关，从问题维得到与喷涂层脱落缺陷相关，从功能维得到与乳白色喷涂相关，从时间维得到 2000 年 1 月相关的记录。浏览从事实表直接转到供应商维，得到出问题的供应商。

图 10-14　　星形模式优化浏览

最适于查询处理

我们已经多次讨论了星形模式是一种以查询为中心的结构。这意味着星形模式最适于查询处理。下面，我们看看究竟是为什么。

我们可以构造一个基于图 10-7 中订单分析的星形模式的简单查询。2000 年 1 月卖给旧

金山(San Francisco)客户的产品 A 的总体成本是多少？这是一个涉及 3 个维度的查询。最适于处理这条查询的数据结构或者数据模式应该由什么特征呢？最终的结果（总成本）将从事实表中的记录得到。但是，究竟应该是那一些记录呢？答案是那些与产品 A，与旧金山，于 2000 年 1 月都相关的纪录。

让我们看看查询语句是如何执行的。首先，查询从客户维选择那些在旧金山的客户。然后，从事实表中选择那些与顾客维相关的记录。这是从事实表中得到的第一个结果集。接下来，从时间维中选择那些月份维 2000 年 1 月的记录。在从第一个结果集选择那些与被选出的时间维相关的记录。这就是从事实表中得到的第二个结果集。从产品维中选择产品为产品 A 的记录。再从第二个结果集中选择那些与选出的产品维相关的记录。现在就得到了事实表的最终结果集。将所有的扩展成本加起来就得到了总成本。

如果不考虑查询中涉及的维的数量，也不考虑查询的复杂程度，实际上每一个查询都只是简单的使用一些参数过滤维表，得到一些维表结果，然后从事实表中得到相应的结果集。这都得益于简单、清晰的连接路径以及星形模式。没有其它的方式能够达到这种效果了。

数据仓库查询的另一个重要方面就是上钻和下钻。让我们看一个下钻的场景。假设我们已经得到了所有加利福尼亚州的顾客的所有扩展成本。结果是通过事实表的记录得到的。接下来我们希望按照邮政编码来下钻这些结果。这可以通过选择在事实表中和相应邮政编码范围的相关的记录。相反的，上钻是通过扩展事实表的选择记录的过程。

星形连接和星形索引

星形模式允许查询处理软件使用更优化的执行计划。它使得查询有能够更高的效率。星形模式尤其适用于应用类似于星型连接或者星型索引这类提高性能的技术。

星型连接是一种高速，并型的，单独操作的多表连接。它可以通过一个单独的操作连接多个表。这种特别的模式能够显著的提高查询性能。

星型索引是一种独特的索引，它可以提高连接的性能。这些索引建立在事实表的一个或者多个外键上面。这些索引可以提高维表与事实表中连接速度。

我们将会在 18 章进行更深入的讨论，研究数据仓库的物理设计问题。

本章总结

- 维度模型的部件是从需求定义的信息包中演化而来。
- 实体-关系建模技术不适用于数据仓库；在数据仓库环境中，应该使用维度建模技术。
- 事实表包含了商业指标；维表包含了商业维度。每个维表中的层次结构用于下钻到更低级别的数据。
- 星形模式的优势包括：用户易于理解；优化浏览；提高查询处理速度；可以使用特别的提高性能的模式。

思考题

1 讨论在数据设计开始之前，需要考虑的设计问题。

2 为什么实体-关系建模技术不适合数据仓库？维度建模技术有又什么不同？

3 什么是星形模式？什么是功能部件表？

4 维表很宽，而事实表很长。清解释这种情况。

5 什么是适用于维表的层次结构以及目录？

6 完全加和指标以及半加和指标的区别？

7 解释事实表中数据稀疏的情况。

8 描述维表以及事实表中的主键。

9 讨论数据仓库中的数据粒度。

10 指出三个使用星形模式的优势。你可以思考一下星形模式的劣势。

复习题

1 将以下的两列配对起来：

1 信息包	A. 激活下钻功能
2 事实表	B. 引用数字
3 CASE 工具	C. 细节级别
4 维度的层次结构	D. 用户容易理解

5 维表	E. 半加和
6 退化的维表	F. 星形模式功能模块
7 边际利润百分比	G. 用于维度建模
8 数据粒度	H. 维度属性
9 星形模式	I. 包含指标
10 客户人口统计特征	J. 宽的

2 回到第 5 章的酒店连锁信息包（图 5-6）。使用信息包并设计一个星形模式。

3 什么是不含事实的事实表？设计一个简单的星形模式，其中要包含一个不含事实的事实表，该事实表按照诊断程序以及时间跟踪一个医院的病人。

4 假如你在一个制造公司的数据仓库项目团队担任数据设计专家。请设计一个星形模式跟踪生产数量。生产数量通常使用产品、时间、使用的部件、生产工具以及生产运作商业维度进行分析。请阐述你的设想。

5 在一个星形模式中跟踪一个地理上分布的公司的出货量，有以下的维表：（1）时间，（2）要发送到的客户，（3）发货的地点，（4）产品，（5）交易的类型，（6）发货的方式。考察这些维度，列出每一个维表可能用到的属性。为每一个表指定一个主键。

第十一章　维度建模：高级专题

本章目标

- 讨论并很好的理解慢速变化维度
- 理解大维度，并学会如何处理它们
- 细致的讨论雪花型模式
- 学习汇总表并决定在什么时候使用它
- 仔细的考察星形模式的各种类型以及它们的应用

通过上面的一章，你学习了维度建模的基础，知道了星形模式是由位于中央的事实表以及环绕在四周的维表组成的。虽然这种表示方式非常形象，但实际上它仍然是一种关系模型，因为每一个维表都和事实表又一个“父-子”连接。所以，维表中的主键就是事实表中的外键。

通过上一章的学习，你掌握了事实表和维表中属性的特点。你也理解了星型模式在决策支持系统中所具有的优势。星型模式易于被用户所理解；它优化了数据仓库内容的浏览方式，而且最适用于以查询为中心的环境。

但是，我们对维度建模的讨论还没有结束，下面，我们将讨论若干个专题。在星型模式中，维表使分析可以通过若干种方式进行。我们需要更详细的讨论维表。如果将一些指标进行汇总，并将一些汇总的数字保存在另外的事实表中，又会怎么样呢？星型模式是一种反规范化的设计。这种设计会导致很多的冗余并造成低效率吗？如果是这样的话，有没有其它的方式呢？

让我们在维度建模基础上，继续深入的讨论一些与维度建模相关的高级专题。

维表的更新

回到前一章的图 10-4，这是汽车销售商的星型模式。事实表“汽车销售”包含了一些

指标，例如“实际销售价格”，“配件价格”，等等。过了一段时间之后，实施表会出现什么变化呢？每天都会有越来越多的销售成交，事实表的行数会不断的增加。但是，事实表中的数据却很少会改变。即便对以前的数字要作修改，这些修改的数据也会作为另外的修改行添加到事实表中。

现在我们来看看维表。与事实表相比，维表要稳定的多。然而，与事实表只增加记录的行数不一样的是，维表的变化不仅包括增加记录的行数，而且属性本身也会改变。

看一下产品维，每一年，随着新模型的出现，维表的记录都会增加。但是产品维中的属性会有什么变化呢？如果一个特定的产品被划到另一个产品目录，那么产品维中相应的数值必须作修改。让我们讨论一下影响维表的变化类型以及如何处理这些不同类型的变化。

慢速变化维

在上面的例子中，我们提到了由于产品所属的产品目录发生变化导致的产品维表的变化。考虑一下客户人口属性维表。如果一个客户的状态有租房变为拥有住房，会发生什么情况？维表中相应的行必须作修改。接下来，看看付款方式维表。如果一种付款方式的经济类型发生变化，那么付款方式维表中也要作相应的修改。

通过考察修改维表的考虑因素，我们可以得出以下的原则：

- 绝大部分的维都是不变的
- 很多维度虽然会变化，但是这种改变是缓慢的
- 源记录的产品键不会改变
- 产品描述以及其它的属性的改变都是缓慢的
- 在源 OLTP 系统中，新的值会覆盖旧的值
- 在数据仓库中，覆盖维表的属性并不总是适当的做法
- 修改维表的方法依赖于改变的方式，以及数据仓库中什么信息必须被保存。

通常对维表的修改可以分成 3 种类型。我们将对这三种类型的修改作详细的讨论。你将理解为什么对不同类型的修改必须使用不同的技术。数据仓库的实践者对不同类型的修改采用不同的技术。他们还为这 3 种修改维表的方式起了名字，分别称为第一类修改，第 2 类修改，和第 3 类修改。

我们将使用一个为工业产品分发者服务的跟踪订单的简单的星型模式来学习这三类修

改，如图 11-1 所示。这个星型模式包含了事实表以及 4 个维表。我们假设维度需要做一些修改，然后讨论一下使用什么技术来进行修改。

第 1 类修改：改正错误

第 1 类修改的实质

这些修改通常与修正源系统中的错误相关。例如，假设一个原来系统中一个客户的名字是 Michel Romano，现在要修改成 Michael Romano。另外一个客户的名字要从原来的 Kristin Daniels 改成 Kristin Samuelson，她的婚姻状态要从单身该成已婚。

考虑一下这两个例子中的名字修改。这里没有必要保留旧的值。在 Michael Romano 的例子中，旧的名字是错误的，不应该保留。如果一个用户要找到所有 Michael Romano 的订单，这个用户应该用正确的名字。在 Kristin Samuelson 的例子中，应该使用同样的原则。

但是婚姻状态的修改有一点不同。只有在婚姻状态的修改是为了修正错误的情况下，才能够与修改名字采用相同的方法。否则，如果用户需要分析与婚姻状态相关的订单信息，这将会导致问题。

图 11-1　跟踪订单的星型模式

以下是对第 1 类修改的通用原则：

- 通常，这些修改与源系统中的改错有关；
- 这种修改在源系统中有时没有意义；
- 源系统中的旧的数值不能保留；
- 源系统中的修改不需要在数据仓库中保存。

对数据仓库应用第 1 类修改

请看图 11-2，它显示了对客户维表的第一类修改。应用第 1 类修改的方法是：

- 用新的值覆盖维表中的旧数值
- 属性的旧数值不需要保留
- 对维表没有其它修改

- 维表中的键值不受影响
- 这类修改是最容易实施的

图 11-2　应用第 1 类修改的方法

第 2 类修改：保存历史数据

第 2 类修改的实质

回到为 Kristin Samuelson 修改婚姻状态的例子。假设数据仓库的一个重要的需求就是通过婚姻状态跟踪订单。如果修改婚姻状态发生在 2000 年 10 月 1 日，那么在此以前 Kristin Samuelson 的所有订单相关的婚姻状态必须为“单身”，而所有在此之后的订单的婚姻状态必须为“已婚”。

在这个例子中，需要数据呢？在数据仓库中，你必须有隔离订单的方式，从而能够分开在改变发生之前以及发生之后的订单。

现在，我们要修改 Kristin Samuelson 的另一个信息。假设她原来的地址是纽约，在 2000 年 11 月 1 日，她搬到了加利福尼亚。如果数据仓库其中一个需求是能够根据州跟踪订单，那么这个修改可以采用修改婚姻状态的方法处理。在 2000 年 11 月 1 日之前的所有记录都应该与纽约州相关。

我们讨论的婚姻状态的修改以及客户地址的修改都属于第 2 类修改。以下是这一类修改的原则：

- 他们通常与源系统中的修改相关；
- 需要在数据仓库中保留历史数据
- 这一类修改将数据仓库中的历史数据分区
- 对属性的每一个修改都要保留

对数据仓库应用第 2 类修改

请看图 11-3，它显示了如何将第 2 类修改应用在客户维表重。应用第 2 类修改的方法是：

图 11-3　应用第 2 类修改的方法

- 在维表中增加一条新的记录，该记录存有修改后的数值
- 维表中可能会有一个有效日期字段
- 对原来维表中的原始记录没有作修改
- 原来的键不受影响
- 新的记录插入，该记录有一个新的替代键

第 3 类修改：暂时的（软性的）修改

第 3 类修改的实质

几乎所有对维表的修改都属于第 1 类或者第 2 类修改。在这两类修改中，第 1 类修改是最常见的。第 2 类修改保留了历史数据。如果在某个特定的日期应用第 2 类修改，该日期将成为分离点日期。在上面提到的在 2000 年 10 月 1 日修改婚姻状态的例子中，2000 年 10 月 1 日就是分离点日期。客户在该日期之前的所有订单都归类到较老订单组中；在该日期之后的记录则归到较新的一个订单组中。该客户一个订单一定可以归到两个组中的其中一个；而不可能同时属于这两个组。

如果在分离点日期之后，你需要计算某一段时期的订单，而这段时期跨越了两个组，哪有会怎样呢？使用第 2 类修改不能够处理这种情况。有时候（虽然可能性很小），你需要跟踪一段时期之内的订单，而这段时期包含了分离点日期之前与之后的记录。这种修改就属于第 3 类修改。

第 3 类修改是一种暂时的（软性的）修改。我们可以用一个例子来将这个概念解释清楚。假如你的营销部门希望对销售人员负责的地域重新划分。在作一个持久的划分之前，他们希望用两种方式计算订单：按照当前的地域划分，以及按照计划的地域划分。这种临时的修改就是第 3 类修改。

我们举一个例子，假设你希望将销售员 Robert Smith 从新英格兰州移到芝加哥州，要求跟踪他在两个州的订单情况。

以下是对第 3 类修改的一些通用的原则：

- 它们通常与源系统的临时修改相关
- 需要跟踪新旧两个属性值

- 新旧两个值用于比较改变所带来的效果
- 它们提供了前向和后向的跟踪能力

对数据仓库应用第 3 类修改

图 11-4 显示了对客户维表应用第 3 类修改。应用第 3 类修改的方法如下：

- 对受影响的属性，在维表中加入“旧的”字段
- 将“现有” 字段值赋值给“旧的”字段
- 将新的值赋给“现有”字段
- 加入一个“现有”有效日期
- 记录的键不受影响
- 不需要新的维表记录
- 现有的查询可以无缝的转换到“现有”的值
- 所有使用到“旧的”值的查询需要作相应的修改
- 这种技术对某段时期的临时修改最为适用
- 如果还有后续的修改，则需要使用更为复杂的技术

图 11-4　应用第 3 类修改

各式各样的维度

在讨论了对维度属性的修改类型以及相应的处理方式之后，让我们讨论一下关于维度表的更重要的专题，这个专题与特别宽又特别深的维表有关。

在先前的讨论中，我们假设维表的变化不会很快。如果是第 2 类的修改，你就必须用新的属性值构造一条新的记录。但是，如果属性值要多次改变，情况又会怎样？这样的一个维度不是慢速变化的维度。那么，我们怎么处理这个非慢速变化的维度呢？讨论这个问题需要考虑几个相关的专题。

大维度

你可能根据两个因素来判断一个大维度。一个大维度很深有很多的记录；也就是有很多的记录。一个大维度也可能很宽，也就是有很多的属性。在任何一种情况中，都可以说这是一个大的维度。大维度需要一些特别的考虑。你可能需要用特别的方法向大维度表中倒入数据。你也可能希望将大维度表与其它的小维度表分开来。我们还将以一个订单分析的星型式做例子。假设这个星型模式为一个制造公司服务，市场营销部门希望通过公司收到的订单，知道它们工作的成效。

在一个数据仓库中，客户维和产品维是典型的大维度。一个企业如果面向广大的公众的话，它的客户维度将是很大的。一个全国连锁店的客户维度可能有全美国所有家庭数目那么大。这样一个客户维表可能会包括了 1 亿条记录之多。电信公司或者旅游公司的客户维度表记录规模已相当的大，同样是上百万级别的，一千万或者两千万的客户记录也不是罕见的。大型零售店的产品维也相当的巨大。

以下是大型客户维和产品为的特写典型的特性：

客户维

- 巨大——大概有 2000 万条记录
- 可能会有 150 个维度属性
- 多层的层次结构

产品维

- 有时有 100,000 种产品之多
- 可能有超过 100 个维度属性
- 多层的层次结构

大维度需要特殊的处理。由于数据量大，很多涉及大维度数据仓库功能可能会很慢，效率很低。你需要采用高效率的设计方法、选择正确的索引、或者采用其它优化技术来处理以下问题，包括：

- 向大维度表填充数据
- 非限制维度的浏览性能，尤其是那些属性较少的维度
- 多限制的维度属性值的浏览时间

- 涉及大维度表的对事实表查询的低效率问题
- 为处理第二类修改所需要增加的额外的记录

多层次结构

大维度通常会有多层次结构。以一个大型零售商的产品维为例。市场营销部门的层次结构可能由一系列的属性组成。该部门的用户使用上钻和下钻来使用这一组属性。同样，财务部门使用属于它们的属性集上钻和下钻，访问同样的产品维。图 11-5 显示了一个大的产品维度中的多层结构。

产品维　财务层次结构　　营销层次结构

图 11-5　一个大的产品维度中的多层次结构

快速变化维

在处理第 2 类修改时，你要使用修改后的属性值建立一个附加的维度记录。这样，你就可以保留修改历史。如果这个属性再次修改，你要为最新的属性值增加一条新的记录。

大多数产品维度的修改并不频繁，一年大概 1 到 2 次。如果产品维度的记录在 100,000 行左右，那么对新属性值加入新纪录的方法管理起来还是比较容易的。即使记录的数目只有几千行，这种方法也是可行的。

然而，考虑其它的维度，例如客户维度。这些维度由很多行，有时候甚至超过 100 万行数据。对相当多记录的属性的不频繁的改变，第 2 类修改也不会很困难。但是一个客户维度的重要属性一年可能会修改很多次。快速变化的大维度对第 2 类修改来说问题就大了。在每一次总量装载之后产生大量新的记录，维表将会变得杂乱无章。

在考察处理快速变化的大维度的其它选择之前，我们单独的处理每个大维度。第 2 类方法在一个星型设计中依然是不错的选择。以下列出了一些第 2 类方法在许多情况下式和处理快速变化维度的理由：

- 如果维表保持扁平，它允许在维度中不同属性间进行对称的浏览。
- 即使加入了额外的维表行，基础的维表结构依然不变。事实表通过外键和所有维表

连接起来，保持了星型模式所具有的优势。

- 只有当查询涉及修改过的属性，结果才会显示出不同。对其它的查询，结果不会有变化。

如果维表特别大，变化又很快，情况会怎么样呢？这就要找一种第 2 类方法的替代方案。将大维度表分解成多个较为简单的小维度表，是其中一种有效的方法。怎么才能实现这种方法呢？

显然，你应该将快速变化的属性放到另一个维表中，而将缓慢变化的属性放在原来的表里面。图 11-6 显示了一个客户维度表是怎样被分解成两个维度表的。这幅图显示了将快速变化属性进行分解的通用技术。你将它作为一个指南，指导你对数据仓库中的大型、快速变化的维度进行分解。

任何事实表　客户维度（原始的）| 客户维度（新的） 行为维度（新的）任何事实表

图 11-6　分解一个大型的快速变化的维度表

废弃维度

检查一下你的源系统，看看其中的客户，产品，订单，销售区域以及推广战役等独立字段的数据结构。大多数的字段是在维表中经常使用的，但你会注意到一些字段，例如一些标志以及文本字段不在维表中出现。这些字段包括是/否标志，文本代码以及自由格式的文本。

这些标志或是文本数据中的一部分可能是过时了。但是，大部分字段可能有可能在查询中用到。它们不应作为主要维表中的重要字段，但是也不应该忽略。那么，怎么处理呢？以下是几个选择：

- 忽略掉所有的标志和文本。显然，这不是一个好的选择，因为这样做可能会丢掉一些有用的信息。
- 将标志和文本原封不动的放在事实表中。这种选择将增大事实表。
- 为每一个标志和文本建立一个独立的维表。这样做将大大的增加维表的数目。
- 只保留那些有意义的标志和文本；将所有有用的标志放在一个单独的“废弃”维中。

“废弃”维的属性对涉及标记/文本的查询来说是有用的。

雪花形结构

“雪花化”是一种规范化星型模式维表的方法。如果你将所有维表完全规范化，那么将得到一个以事实表为中心的雪花型结构。首先，我们从图 11-7 开始，该图显示了一个简单的制造公司销售情况的星型模式。

销售事实表包括了数量，价格以及其它相关的指标。有 4 个维表，包括销售代表，客户，产品以及时间。这是一个典型的星型模式，为了优化查询，维表都是反规范化的，不符合第 3 范式。

规范化选项

假设产品为中有 500,000 行记录。这些产品分成 500 个产品品牌，而产品品牌又分成 10 个产品目录。现在假设有一个用户运行一个查询，该查询只涉及一个产品目录。如果产品维表没有对产品目录字段做索引，那么，该查询就必须扫描 500,000 行。另一方面，如果产品维度只做了部分的分区，将产品品牌和产品目录分成两个不同的表，那么查询就只需要搜索产品目录表中的 10 行记录。图 11-8 显示了查询过程中搜索记录数目的减少。

图 11-8 中，我们没有完全的将产品维规范化。我们也可以将其它属性放到产品维表外，形成规范化的结构。“雪花化”或者说是维表规范化可以通过几种不同的方法实现。如果你想“雪花化”，就应该检查每一个维表中的内容以及用途。

客户，销售代表，事实表，时间，产品

图 11-7　销售：一个简单的星型模式

（顺时针）

客户，销售代表，事实表，事件，产品，品牌，目录

图 11-8　产品维：部分规范化

以下是将维表规范化的几种方法：

- 对一些维表进行部分规范化，对其它维表不作修改
- 对一些维表进行部分或者完全规范化，对其它维表不作修改
- 对每个维表部分规范化
- 对所有维表完全规范化

图 11-9 显示了对每一个维表进行部分或者完全规范化的星型模式的版本。

原来的销售星型模式（如图 11-7 所示）只含有 5 个表，而在规范化之后，则有 11 个表。注意在雪化型模式中，每个维表中级别较低的属性被移到一个单独的表中。这些新的表通过键与原来的维表连接。

销售事实

客户-国家

销售代表-地域-地区

时间

产品-包

-品牌-目录

图 11-9　销售：“雪花”模式

优势与劣势

使用雪花型结构的原因是明显的。通过减少维表中的所有长文本字段，你将节省存储空间。例如，如果你有一个目录叫“男性用品”，这个文本就会在每一行出现。如果维表很大的话，去除这些冗余字段将明显的减少存储空间。

假设产品维表有 500,000 行。通过雪花化，会减少 500,000 个 20 比特的目录名。同时，你必须在维表中假如一个 4 比特的目录键。节省的空间为 500,000 乘以 16，即 8MB。500000 行的产品维表大致要占用 200MB 的存储空间，对应的事实表有 20GB。空间只节省 4%。你会发现节省的空间不能够补偿雪花化的缺陷。

以下是雪花化优势和限制的一个总结：

优势

- 减少（很少的）存储空间
- 规范化的结构更容易更新和维护

劣势

- 模式比较复杂，用户不容易理解
- 浏览内容更为困难
- 额外的连接将使查询性能下降

在数据仓库中，通常不推荐雪花化。在数据仓库中，查询性能极为重要，而雪花化将降低性能。

什么时候使用雪花形结构

作为一个 IT 专家，你对第 3 范式很熟悉。你也知道非规范化结构所带来的问题。而且，浪费的空间也是考虑采用雪花化的因素之一。

除了明显的劣势，有没有适合使用雪花化的情况呢？雪花化的原则是将维表中的低级的属性移出，构建新的表。类似的，在一些情况下可以将一组属性分离，形成子维度。两个属性集的粒度是不一样的。这个过程与雪花化技术很相像。请看图 11-10，该图显示了从一个客户维度中分离出一个人口统计属性子维度。

虽然构造子维度可以解释为雪花化，但是将人口统计属性分离成另一个表还是很有意义的。通常，装载人口统计数据的辞书与装载其它数据的次数不同。两个属性集在粒度上是不同的。如果客户维很大，达到几百万行，节省的存储空间是很可观的。另一个将人口统计数据分离开来的原因与属性的浏览相关。相对于维表的其它属性来说，用户可能更多的浏览人口统计属性。

聚集事实表

聚集是从最低粒度的事实表中衍生出来的预先计算的汇总数据。这些汇总数据形成了一组独立的聚集事实表。你可以为一个特定的汇总构建一个聚集事实表。我们将从一个简单的

星型模式开始介绍。这个模式中事实表有最低的粒度级别。假设有 4 个维表围绕着这个事实表。图 11-11 显示了这个模式。

如果你在一个操作型系统中运行一个查询，它将返回一个关于单个客户，单个订单，单个发票，单个产品等等的结果集。但是，数据仓库环境中的查询将返回一个很大的结果集。这些查询会获取成百上千的记录，对事实表中的指标做处理，然后产生结果集。事实表的指标操作可能是一个简单的加总，或者计算平均值，甚至是复杂的算法运算。

销售事实

客户

销售区域

时间

产品

粒度：

每一天中每一个客户购买的每一个产品对应事实表中的一行

图 11-11　　事实表为最低粒度的星型模式

我们将考察几个典型的对图 11-11 中的星型模式例子的查询。

查询 1：客户代码为 12345678 的客户在 2000 年 12 月第 1 个星期中产品 Widget-1 的销售汇总情况。

查询 2：客户代码为 12345678 的客户在 2000 年头 3 个月中产品 Widget-1 的销售汇总情况。

查询 3：中南部地区的客户在 2000 年头 2 个季度中产品目录 Bigtools 的销售汇总情况。

仔细的考察这些查询，确定在每一个查询中是如何计算汇总的。汇总值是同时将事实表中合乎标准的记录的销售数量和销售收入加总得到的。每一个例子中合乎标准的结果集如下。

查询 1：事实表中所有客户键为客户代码 12345678，产品键为 Widget-1，以及时间属于 2000 年 12 月第一个星期的记录。假设一个用户一天最多购买一件产品，那么在汇总中最多只有 7 条记录。

查询 2：事实表中所有客户键为客户代码 12345678，产品键为 Widget-1，以及时间属于 2000 年第一个季度的记录。假如一个一个用户一天最多购买一件产品，那么在汇总中最多只有 90 条记录。

查询 3：事实表中所有属于中南部地区的客户，产品键属于 Bigtools 目录，以及时间属于 2000 年头两个季度的记录。在这个例子中，那么在汇总中将有很多记录。

显然，查询 3 由于要返回很多记录，所以运行的时间会很长。那么，我们怎样才能减少查询时间呢？汇总表能够做到这一点。在详细讨论汇总表之前，我们先看看在数据仓库中集中典型的事实表的大小。

事实表的大小

请看图 11-12，它表示了一个大型超级市场连锁销售情况的星型模式。基础事实表中大概有 20 亿条最低粒度的记录。请看以下的计算：

时间维：5 年×365 天=1825

商店维：每天有 300 个商店报告销售情况

产品维：每个商店有 40,000 种产品（大概每天每个商店售出 4000 件商品）

推广维：在一个特定的日期，在一个商店中一件售出的商品只属于一个推广

事实表最大的记录数目：1825×300×4000×1=200 亿

事实表　20 亿条记录

商店　　300 个

推广　　在一个特定的日期，在一个商店中一件售出的商品只属于一个推广

时间　　5 年，1825 天

产品　　40000 中产品（每天每个商店只售出 4000 种）

图 11-12　　星型模式：连锁商店

以下是另外一些典型情况下对事实表大小的估算：

电话呼叫监控

时间维：5 年=1825 天

每天跟踪的呼叫数量：1.5 亿

基础事实表记录的最大条数：2740 亿

信用卡交易跟踪

时间维：5 年=60 个月

信用卡招呼数量：1.5 亿

平均每个账号每月的交易数：20

基础事实表记录的最大条数：1800 亿

从以上的例子中，我们可以看到，在一些典型的情况下，最低粒度的事实表是相当大的。虽然没有查询会需要查找事实表中的单独一行，但是最低粒度的数据还是需要的。这是由于一个用户执行不同形式的分析时，她或他必须得到一系列有单独事实表记录组成的结果集。如果你不保留每个商店的细节，你就不会得到按照商店分类的产品结果集。另一方面，如果你不保留每个产品的细节，你就不会得到一个商店中按照产品分类的结果集。

所以，问题就在于此。如果你在事实表中需要最低粒度的数据，你怎么来处理庞大的事实表，生成查询的结果呢？考虑下列与连锁商店数据仓库相关的查询：

- 与全国平均水平相比，Wisconsin3 个新的商店在最近 3 个月表现如何？
- 最近一次节日的肉类和家禽销售战役效果如何？
- 和去年相比，6 月 4 号不同商品目录的销售情况如何？

这 3 个查询都需要从事实表种进行选择和汇总。对这些汇总，你需要基于有一个或多个维表的细节数据，但是对其它的维表，则只需要汇总数据。例如，对最后一个查询，你需要时间维表的细节数据，但是只需要产品目录的汇总数。如果你能够预先计算汇总数据，那么查询将运行的更快。如果有正确的汇总数据，那么每一个查询的性能都将大大提高。

聚集的需求

请参考图 11-12，它显示了连锁商店的星型模式。假设 300 个商店中，每一个品牌都有 500 个产品。在 40000 个产品种，假设每星期每个商店中的每个产品最少卖出。下面我们估计以下类型的查询所返回的记录数目：

查询涉及一个产品，一个商店，1 个星期——汇总一条记录

查询涉及一个产品，所有商店，1 个星期——汇总 300 条记录

查询涉及 1 个品牌，1 个商店，1 个星期——汇总 500 条记录

查询涉及 1 个品牌，所有商店，1 年——汇总 7800000 条记录

假设你预先计算得到一个聚集事实表，其中每一行汇总了一个品牌、一个商店、一个星期的总数。那么第 3 条查询只需要从这个汇总表中得到 1 条记录。类似的，最后以条查询要从聚集事实表种获取 15600 条记录，而不是从事实表中取回 7 百万条记录。

如果你预先计算得到另一个聚集事实表，其中每行汇总了一年、一个品牌、一个商店的总数，最后一条查询就只需要取回 300 条记录。

聚集表的记录数要远远少于事实表。所以，如果大多数查询在聚集表上进行操作，而不是在事实表上操作，数据仓库的性能将大大的提升。构造聚集事实表是提高查询性能的一种非常有效的方法。

对事实表进行聚集

聚集事实表将最低粒度的数据通过维度层次结构汇总成更高层次的数据。请看图 11-13，它显示了 3 个维度的层次。时间维度的层次结构最低的是天，最高的是年。商店维度的最低层次是城市，而产品维度的最低层次是产品。

销售事实（表内字段用英文）

商店　　所有商店　　层次结构

时间　　层次结构

产品　　所有产品　　层次结构

层次结构

最低层次

最高层次

图 11-13　　维度层次结构

在基础事实表中，记录对应维度层次结构中最低层次。例如，基础事实表中每一个行显示了与某个日期、某个商店和某个产品相关的销售单元和销售金额。通过每个维度的更高层次，你可以生成一系列的聚集表。下面我们来看聚集的几种方法。

多路聚集事实表

请看图 11-14，该图显示了构造聚集事实表的不同方法，并对每一种做了相应的描述。

单路聚集

如果从一个维度中的一个层次升到一个更高的层次，而在其它维度保持最低粒度，就生成了一个单路聚集表。请看以下的例子：

- 某个日期、某个商店的产品目录
- 某个日期、某个商店的产品部门
- 某个日期、某个商店的所有产品
- 某个日期、某个产品的所有地域
- 某个日期、某个商店的区域
- 某个日期、某个商店的所有商店
- 某个商店、某个产品的月份
- 某个日期、某个商店的季度
- 某个日期、某个商店的年度

商店	产品	时间	例子

商店*	产品	***日期****	单路聚集
地域	***目录****	月	
区域	部门	季度	
所有商店	所有产品	年	

商店	产品	时间	例子
商店	产品	***日期****	二路聚集
地域*	目录*	月	
区域	部门	季度	
所有商店	所有产品	年	

商店	产品	时间	例子
商店	产品	日期	三路聚集
地域*	***目录****	***月****	
区域	部门	季度	
所有商店	所有产品	年	

图 11-14　　构造聚集事实表

二路聚集

如果从两个维度中的一个层次升到一个更高的层次，而在其它维度保持最低粒度，就生成了一个单路聚集表。请看以下的例子：

- 按照日期和地域的产品目录
- 按照日期和地区的产品目录
- 按照日期和所有商店的产品目录
- 按照月份和商店的产品目录

- 按照季度和商店的产品目录
- 按照年度和商店的产品目录
- 按照日期和地域的产品部门
- 按照日期和地区的产品部门
- 按照日期和所有商店的产品部门
- 按照月份和商店的所有产品
- 按照季度和商店的所有产品
- 按照年度和商店的所有产品
- 按照日期和地域的所有产品
- 按照日期和地区的所有产品
- 按照日期和所有商店的所有产品
- 按照月份和商店的所有产品
- 按照季度和商店的所有产品
- 按照年度和商店的所有产品
- 按照月份和产品的大区
- 按照季度和产品的大区
- 按照年度和产品的大区
- 按照月份和产品的地域
- 按照季度和产品的地域
- 按照年度和产品的地域
- 按照月份和产品的地区
- 按照季度和产品的地区
- 按照年度和产品的地区
- 按照月份和产品的所有商店
- 按照季度和产品的所有商店
- 按照年度和产品的所有商店

三路聚集

如果从所有 3 个维度中的一个层次升到一个更高的层次，就生成了一个单路聚集表。请看以下的例子：

- 按照地域和月份的产品目录
- 按照地域和月份的产品部门
- 按照地域和月份的所有产品
- 按照地区和月份的产品目录
- 按照地区和月份的产品部门
- 按照地区和月份的所有产品
- 按照所有商店和月份的产品目录
- 按照所有商店和月份的产品部门
- 按照地域和季度的产品目录
- 按照地域和季度的产品部门
- 按照地域和季度的所有产品
- 按照地区和季度的产品目录
- 按照地区和季度的产品部门
- 按照地区和季度的所有产品
- 按照所有商店和季度的产品目录
- 按照所有商店和季度的产品部门
- 按照地域和年度的产品目录
- 按照地域和年度的产品部门
- 按照地域和年度的所有产品
- 按照地区和年度的产品目录
- 按照地区和年度的产品部门
- 按照地区和年度的所有产品
- 按照所有商店和年度的产品目录
- 按照所有商店和年度的产品部门
- 按照所有商店和年度的所有产品

每一个聚集事实表都是从基础事实表衍生而来的。衍生的聚集表与一个或者多个衍生的维表连接。图 11-15 显示了一个与衍生维表连接的衍生事实表。

基础表

销售事实

商店

从产品表衍生出来的维表

一路聚集

销售事实

时间

产品

图 11-15 聚集事实表和衍生维表

聚集稀疏性的影响

考虑一下，一个连锁商店有 300 家商场，40000 种产品，但是每个商店每天只卖出 4000 件商品。我们先前假设你有 5 年（1825 天）的记录，基础事实表最大的记录数计算如下：

产品=40000

商店=300

时间=1825

基础事实表记录的最大数目=220 亿

由于每个商店每天只有 4000 种产品售出，所以实际的记录数到不了 220 亿，大概只有最大数目的 10%。所以，基础事实表的实际记录数为 20 亿。

现在我们来看看在构建聚集的时候会发生什么情况。我们仔细的考察一个单路聚集：按天以及商店的品牌的销售总数。计算以下这个单路聚集的最大记录数。

品牌=80

商店=300

时间=1825

聚集表的最大记录数=43800000

在构建单路聚集的时候，你会注意到这个聚集表的记录数目并不是基础事实表的 10%。这是由于在按照品牌聚集时，更多的品牌代码会与商店代码和时间代码连接。这个单路聚集的稀疏程度大概是 50%，所以真实的估计大概有 21900000 条记录。如果稀疏程度依然是基础表的 10%的话，实际预计记录数量会更少。

如果你在更高的层次进行聚集，稀疏的程度可能会达到 100%。由于稀疏程度很难保持低水平，你会发现聚集并不能明显的提高性能。聚集真的能够减少记录的数目吗？

有经验的数据仓库工程人员会给你这样的建议：在构建聚集时，确保每一个聚集表能够会在聚集 10 条来自更低层次的表的记录。如果能够聚集 20 条以上，那就更好了。

聚集的选项

让我们回到刚才对只有 3 个维表的基础星型模式的单路、二路、三路聚集的讨论中，你可以有 50 种不同的构建聚集的方法。在实际情况中， 维表的数目不会只有 3 个，而是多得多。所以，可能的聚集表数目可能会扩展到几百个。

而且，从对聚集稀疏性失效的讨论中，我们知道聚集所能减少的记录数量是不成比例的。换一句话来说，如果基础事实表的稀疏程度为 10%，更高层次的聚集表并不能保持 10%。你的汇总层次越高，稀疏程度比例就越高。

聚集真的那么有效吗？在聚集过程中，有什么样的选项呢？你怎样进行聚集呢？首先，我们要搞清楚聚集的目标。

聚集策略的目标

除了提高数据仓库性能这个一般的目标之外，聚集还包括以下几个更实际的特定目标：

- 不要陷在过多的聚集之中。记住，要支持聚集，需要额外的衍生维表。
- 尝试满足大多数用户的需求。尤其是你的高级用户。
- 建立聚集不能过多的增加存储容量。对具有较低稀疏比例的那些大型聚集要注意。
- 保持聚集对最终用户是不可见的。就是说，聚集必须对用户的查询是透明的。查询工具必须能够意识到聚集的存在，并正确的使用它。
- 尽量减少对数据缓存处理的影响。

实用的建议

在讨论数据仓库需要哪一类的聚集计算之前，我们先花一点时间考察一下查询的一般特性。你的用户如何使用报表的结果？报表到达哪一个层次？是按照商店？按照月份？按照产品目录？看看各个维表，看一下它们的层次。要注意同一个维度里面，是否有多个层次结构。如果是这样，理清楚这些层次结构相当的重要。确定在每一个维度中哪一些属性是用于对事实表进行分组的。下一步，就是确定哪一些属性组合使用的，以及这些常用的组合是什么。

一旦你确定了属性以及他们可能的组合，看一下每一个属性可能取值的数量。例如，在一个酒店连锁星型模式中，假设在酒店维度中，酒店是最低的层次，城市比酒店高一个层次。假如酒店有 25000 个值而城市有 15000 个。显然，对城市进行聚集并没有什么明显的优势。另一方面，如果城市只有 500 个值，那么对城市进行聚集是值得考虑的。对维度中的每一个属性进行考察，对每一个属性的取值进行考察。将同一个层次结构中不同层次的属性值进行对比，以决定那些是聚集可能包括的属性。

列出一个对聚集有用的属性列表，然后就可以通过这些属性的组合构成你第一个多路聚集表的集合。要确定为了支持那些聚集事实表，你需要哪一些衍生的维表。之后，就可以开始实施这些聚集事实表了。

要记住的是，聚集是一种调整性能的手段。要提高查询性能就需要进行汇总，所以如果你的聚集表效果不是很好的话，也不要太在意。在必要的时候，你的聚集需要监控和修改。因为查询的特性可能会改变。随着你的用户对数据仓库越来越熟悉，他们将尝试使用新的方式归综合分析数据。那么，实用的建议是什么呢？那就是：做好准备工作，开始的时候构建一些合理的聚集表，并在需要的时候做一些调整。

星形模式族

如果看一个单独的有事实表和维表的星型模式，那么它并不在数据仓库范围之内。几乎所有的数据仓库都包含了多个星型模式结构。每一个星型模式都在事实表中保存了一些指标，为特定的目的服务。一组相关的星型模式称为星型模式族。构造星型模式族有几个原因。你可以通过增加聚集事实表和衍生的维表形成一个星型模式族。你也可以构造一个与所有用

户相关的核心事实表，以及为特定用户群服务的定制事实表。有很多因素导致了星型模式族的存在。首先，我们看一下图 11-16 的例子。

星型模式族中的事实表共享维表。通常，时间维表被族中的大多数事实表共享。在上面的例子中，所有的 3 个事实表都与时间维表相关。另一方面，多个星型模式中的维表也可能共享一个星型模式中的事实表。

如果你运营银行或者电话服务，那么获取每一个事务并按照特定间隔获取快照是很有意义的。然后，你就可以使用包含事务和快照模式的星型模式族。如果你在一个制造公司或者一个类似的产品类型的企业，你的公司需要监控价值链指标。而一些其它的机构，如医疗中心，其价值分布成点状，而不是链状。对这些企业，星型模式族支持价值链和价值环。我们将在下面的几个部分进行详细的讨论。

粗框——事实表

细框——维表

图 11-16

快照表和实务表

我们回顾一下一个电话公司的基本需求。一个电话客户账号由一些独立的事务组成。很多的事务在早上 6 点到晚上 10 点之间发生。住宅用户在节日和周末会有更多的事务。而机构用户更多的在工作时使用电话。公司积累了相当丰富的事务数据，这些数据可以做很多种有家值的分析。电话公司需要一个包含事务数据的数据模式，支持对业务扩张、新业务开发等问题的战略决策。这个事务模式回答诸如“周末和节假日忙时的利润与工作日梦时利润对比情况”一类的问题。

电话公司也要回答客户情况、账户平衡等问题。客户服务部门总是要回答关于单个客户账号情况的问题。而会计部门则对下个月中可能会收到的款项数目感兴趣。为了回答“这个月底会发出哪一些账单”一类的问题，电话公司需要一个定期获取快照的模式。请看图 11-17，该图显示了电话公司的快照和事务事实表。请看一下这两个事实表的属性。一个表跟踪单独的电话事务。另一个表定期保留单独账户的快照。还要注意的是，维表是如何被两个事实表所共享的。

快照和事务表在银行中也是很常用的。例如，一个 ATM 事务表存储了每一个 ATM 事务。这个事实表跟踪客户账户的每一个事物的数额。快照表保留了每天每个账号的余额信息。这两个表的功能是不同的。通过事务表，你可以对 ATM 事务做多种分析。快照表提供了一段时期的总额数，显示余额的变化。

财务数据仓库同样需要快照和事实表，这是由其分析特点决定的。这些数据仓库中，有一类问题与“单独事务在一段特定时期中，如何影响某个特定账户”有关。而另一类问题则关注特定时期期末特定账户的余额。事务表回答第一类问题；而快照则处理第 2 类问题。

区　　账号

电话事务事实表　　电话快照事实表

事务　　时间　　状态

图 11-17　　快照和事务表

核心表和定制表

考虑两种显然不同的业务。首先看看银行。一个银行提供一系列的金融业务。这些业务都不相同。账户核算服务和存款服务在很多方面类似。但是存款服务和信用卡服务是完全不同的。怎么样跟踪这些不同的服务呢？

接下来，考虑一个生产不同产品的制造公司。不同产品之间有一些元素是相同的，但是大多数元素是不同的。怎么样才能获得不同产品的信息呢？

不同类型的星型模式族可以满足这些公司的需求。在一种类型中，所有的产品和服务于一个核心事实表相连，而每一个产品或服务与一个独立的定制表相连。在图 11-18 中，你会看到一个银行的核心事实表和定制事实表。注意核心事实表保留了对所有类型账户都通用的指标。每一个定制事实表保留了特定服务的指标。注意被共享的维表以及这些表是如何构造星型模式族的。

时间

分支机构

家庭

账户审核定制事实表

账户

银行核心事实表

存款定制事实表

图 11-18

支持企业价值链或者价值环

在制造业务中，制造一个产品要通过很多的步骤，从原材料开始到仓库中的最终产品结束。这些步骤包括原材料，材料的装配，过程控制，包装，以及运送到仓库。一个最终产品从仓库中运到分销商的层库，然后转到零售商。在每一个步骤，产品都会增值。有几个操作型系统支持这些步骤。整个流程构成了供应链或者价值链。类似的，在一个保险公司中，价值链可能包含了从保险销售到条款拟定以及最后索赔过程这几个步骤。在这个例子中，价值链与服务相关。

如果你的业务是其中一种，你要跟踪价值链中不同不周的重要指标。你应该为每一个重要步骤构建星型模式，相关的星型模式就构成了一个星型模式族。你要为价值链中每一个步骤构造一个实施表以及相应的维表。如果你的公司有多条价值链，那么为了支持它们，你要为每一条价值链构造一个独立的星型模式族。

一个供应链或者价值链有许多步骤构成，这种组合是线性。在每个步骤中，都会有附加值。在其它类型有附加值的业务中，类似的线性步骤并不存在。例如，一个医疗机构中，为病人服务的价值是通过多个单元完成的，这些单元围绕服务呈环状。我们称之为价值环。一个大型医疗机构的价值环可能包括医院，诊所，医生办公室，配药室，实验室，政府代办处，以及保险公司。这些单元或者提供医疗服务，或者衡量医疗服务。每个单元提供的医疗服务可以用不同的指标进行衡量。但是，大多数的单元会使用同样的维，如时间，病人，医疗服务提供者，医疗方案，诊断，以及付款人。对一个价值环来说，星型模式族由多个事实表和一系列一致的维表组成。

使维度一致

在考察星型模式族的时候，你会注意到维表被多个事实表共享。维表构成的星型模式中的通用连接。对那些使之一致的维度，你需要确保公共维度可以被两个或者多个星型模式使用。如果产品维度被销售事实表和存货事实表共享，那么产品维度的属性与两个事实表联系的时候有共同的含义。图 11-19 显示了一组一致的维表。

一致维度

订单

产品

客户

销售人员

日期

出货

渠道

发送到

从…发送

图 11-19　　一致的维度

订单事实表和出货事实表共享一致的维度，这些维度包括产品，日期，客户，以及销售人员。一致的维度是一组从源系统中抽取的，解决了相互冲突问题的，完整的属性组合。例如，一致的产品维度反映了企业的产品情况，并包括了所有可能的产品层次结构。每一个属性的数据类型、长度和约束都必须是正确的。

使维度一致是数据仓库的一个基本需求。在构造数据仓库时，要特别注意这一点。这是项目组的一个主要责任。一致的维度可能在数据集市中出现。对任何类型的查询，用户界面都应该是一致的。查询的结果集也是一致的。当然，一个一致的维度可以用于多个事实表。

将事实标准化

除了使维度一致之外，将事实标准化也是一个基本需求。我们知道，事实表可以跨越多个星型模式。回顾一下的与事实表属性标准化相关的问题：

- 保证在数据集市中的定义和术语是相同的
- 解决同名问题
- 需要标准化的类型包括利润，价格，成本，边际利润
- 保证每个事实表中所有的衍生单元使用同样的算法
- 保证每一个事实都使用正确的指标单位

星形模式族小结

让我们通过一幅图来结束对星型模式组的讨论。这幅图显示了一系列标准化的事实表和一致的维表（图 11-20）。注意这里面的聚集事实表和相应的衍生维表。这些聚集表是什么类型的？单路还是二路？它们的基础事实表使哪一些？注意内些共享的维表。这些维度一致吗？看一下不同的事实表和维表是如何相互联系的。

账户

二路聚集

单路聚集

银行核心表

审核定制表

月

日期

分支机构

州

图 11-20

本章总结

- 慢速变化维可以按照变化的实质分成 3 种不同类型。第 1 类修改处理修订问题；第

2 类修改保存历史数据；第 3 类是暂时的修改。每一类修改在数据仓库中的应用方式是不同的。

- 大维度，例如客户或者产品，在应用优化技术时，需要特别的考虑。
- “雪花化”或者构建一个雪花型模式，使规范化星型模式的一种方法。虽然有一些情况下可以使用雪花模式，但是一般不推荐使用。
- 杂项的标记和文本数据放在一个“废弃”维表。
- 聚集或者汇总表提高性能。需要制定构建聚集表的策略。
- 一组相关的星型模式构成星型模式族。例如，快照和事务表，核心表和定制表，以及支持价值链或者价值环的表。一个星型模式族依赖于一致的维度表和标准化的事实表。

思考题

1. 描述慢速变化维。有哪 3 种类型？简要的解释每一种类型。
2. 将第 2 类、第 3 类变化进行比较。
3. 你能够对快速变化维使用慢速变化维的第 2 类修改方法吗？请讨论。
4. 什么是“废弃”维度？它们在数据仓库中是必要的吗？
5. 雪花型模式与星型模式有什么不同？说出雪花型模式的两个优势和两个劣势。
6. 慢速变化维和快速变化维有什么不同。
7. 什么是聚集表？为什么需要它们？举一个例子。
8. 用例子描述快照和事务事实表。它们之间有什么关系？
9. 举一个价值环的例子。届时星型模式族如何支持价值环。
10. 使维表一致是什么意思？为什么它在数据仓库中是重要的？

复习题

1. 指出对或错：

   A. 对慢速变化维的第 1 类修改与修订错误有关；

   B. 要对慢速变化维应用第 3 类修改，需要使用新的值覆盖维表中的属性值；

   C. 大维度通常有多个层次结构；

D. 星型模式使规范化形式的雪花型模式；

E. 聚集就是预先计算的汇总；

F. 基础表的稀疏百分比通常比聚集表更高；

G. 星型模式族中的事实表共享维表；

H. 多种服务的核心事实表和定制事实表对公司来说是很有用的；

I. 一致的维度在数据仓库中不是绝对必要的；

J. 一个价值环通常需要星型模式族支持。

2. 假设你运作的保险业务。找出与业务相关的两个第 2 类慢速变化维的例子。作为项目组中的分析师，写一份在数据仓库中应用第 2 类修改的规范。
3. 你是一个零售企业数据仓库项目组中的数据设计专家。设计一个有 3 个维表的星型模式，跟踪销售单位和销售数额。届时你如何构建 4 个二路聚集。
4. 作为一个国际银行的数据设计师，考虑可能的快照表和事务表的类型。使用一组快照表和事务表完成设计。
5. 为一个制造公司，涉及一个包含 3 个星型模式的星型模式组，以支持其价值环。

第十二章　数据抽取、转换和装载

本章目标

- 广泛的了解数据抽取、转换和装载（ETL）功能的所有不同方面。
- 检查数据抽取功能，包括其挑战、技术等，学习如何评估和应用这些技术。
- 讨论任务的范围和数据转换功能的类型。
- 理解数据整合和合并的含义。
- 认识数据装载功能的重要性，了解将数据应用到数据仓库中的主要方法。
- 了解为什么说 ETL 过程是非常重要、耗时和艰巨的任务。

你可能已经了解，组织中操作型系统中的数据完全不适合提供战略决策制定所需要的信息。作为信息技术的业内人士，我们完全了解在过去的二十年中，一直希望能够从操作型系统中得到战略决策所需要的信息，但事实证明是没有成效的。但是现在的数据仓库可以完成这项任务。

在大多数情况下，数据仓库中的信息是从操作型系统中得到的，这些操作型系统中的信息不能直接用来提供战略信息。那么究竟是什么造成了源操作型系统中的数据，与数据仓库中的信息之间的不同？就是数据抽取、转换和装载（ETL）功能所带来的结果。

ETL 功能改造了源系统中的相关数据，将它们变成有用的信息存储在数据仓库中。没有这些功能，就没有数据仓库中的这些战略信息。如果没有对源数据进行正确的抽取、清洗和用正确的格式进行整合，作为数据仓库中枢功能的查询处理，就不能进行。

在第二章中，我们讨论了数据仓库的主要组成部分，我们简要的在数据准备区域中了解了 ETL 功能。在第六章中，我们又看到有关 ETL 的内容，并了解了商业需求是如何驱动这些功能。而在第八章中，我们了解了数据迁移功能的硬软件基本结构。那么，为什么还需要再一次专门来了解 ETL 呢？

ETL 功能构成了数据仓库信息内容的前期必须工作。ETL 功能需要更多的仔细考虑和讨论。在本章中，我们将要进一步的研究有关 ETL 功能的内容。我们还要回顾 ETL 中的许多重要行动。在下一章中，我们还要继续讨论另一个有关 ETL 功能的重要内容——数据质量。现在让我们先对 ETL 有一个简单的概观了解。

ETL 概观

如果你还记得我们关于数据仓库技术结构的功能和服务的讨论，你将会发现我们将环境分成了三个功能区域。这些区域包括数据获取、数据存储和信息传递。数据抽取、转换和装载存在于数据获取和存储阶段。它们是后端处理过程，包括从源系统获取数据。接下来，它们包括了将源数据转换成特定格式和结构以供数据仓库存储需要的所有功能和过程。在数据转换的工作完成后，这些过程包括了所有关于将数据物理迁移到数据仓库内的所有功能。

当然，数据抽取的工作要先于所有其它的功能。但是从源系统抽取的数据的范围和程度是怎样的呢？为什么不将所有的操作型数据全都转存到数据仓库中呢？这看上去是一个直接的方法。不过，这个过程会受到用户需求的驱动。需求定义会指导你，从哪个源系统抽取什么样的数据。一定要避免这种情况：将所有数据都转存到新的系统中，然后等着看用户如何对它们进行操作。数据抽取就是一个预先选择的过程，根据用户的需求来选择需要的数据。

后端处理的范围和复杂程度，根据不同的数据仓库系统都有所不同。如果你的企业中有大量的操作型系统，运行在不同的计算机平台上，那么这里的后端处理工作就要相对复杂一点。所以，根据你的具体情况，数据抽取是一个很大的挑战。数据转换和装载的功能也会很困难。而且，如果源数据的质量较低，那么也会对后端处理有一定的挑战。除了这些挑战，如果在实际环境中只有一些装载方法可以使用，那么数据载入的工作也会相对困难。让我们分别来看看 ETL 功能的特别性质。

最重要和最具有挑战性

ETL 功能的每个部分都有一个特定的重要目的。当你要将源系统中的数据转换成信息存储在数据仓库中的时候，这些功能中的每一个都非常重要。为了将数据转换成为信息，你首先需要获取这些数据。在获取数据之后，你不能简单的将这些数据转存到数据仓库中，然后称它们为战略信息。你需要将这些抽取的数据根据转换的要求进行分类，使数据可以准确地转换成信息。数据装载也是一个很重要的过程。你必须完成 ETL 的所有三项功能，才能成功的将数据转换成信息。

举个例子，你的用户要进行一项分析工作。用户将根据商店、产品和月份对销售情况进行比较和分析。这些销售数字可以从公司的几个销售系统中得到。同时，你有一个产品主管的文件。而且，每一个销售交易都对应于一个特定的商店。所有这些都是源操作型系统中的

数据。为了完成这个分析，你必须在数据仓库中提供这方面的信息，包括在一张事实表中提供销售单位和金额，在产品维度表中提供产品目录，在商店维度表中提供商店名单，在时间维度表中提供月份。如何做到这些呢？从每一个操作型系统中抽取数据，在源系统中同意数据展现的变量名称，转换所有产品的销售情况。然后将销售情况载入事实表和维度表中。现在，在完成这三项功能以后，抽取的数据已经存在于数据仓库中，并且转换成信息，可以进行分析。请注意，每一个功能的执行及其执行的顺序都很重要。

ETL 功能很具有挑战性，在 ETL 中的大部分挑战都是因为源操作型系统的不一致所造成的。下面来回顾一下 ETL 功能中所遇到困难类型的原因。仔细考虑列表中的每一项，与你的实际环境联系起来，这样你就能找到正确的解决方案。

- 源系统彼此之间非常不同。
- 需要对多个平台和不同操作系统进行操作。
- 很多源系统运行在旧的数据库技术和应用系统上。
- 通常，取值不断变化的历史数据载源操作型系统中不能保存。历史信息对于数据仓库又非常重要。
- 很多旧系统中的数据质量各不相同，需要花很多时间进行处理。
- 源系统的结构随着时间会发生变化，因为新的商业条件的出现。ETL 功能也必须相应的调整。
- 源系统之间缺乏一致性，这是普遍存在的现象。在不同的源系统中，相同的数据可能会用不同的形式来表现。例如，有关工资的数据可能在不同的源系统中，表示成月工资、周工资或双月工资。
- 即使在全异的源系统中已经发现了不一致的数据，这种不一致的问题也缺乏一个解决问题的方法。
- 大多数源系统的数据格式各不相同，而且很多展现方式是模糊而隐藏的。

耗时而且费劲

项目团队在设计 ETL 功能、检查不同处理并实施的时候，你会发现这些工作会耗费整个项目工作量的很大一部分。一般说来，项目团队花上 50%到 70%的工作量在 ETL 部分，一点都不奇怪。在前面的内容中，你已经了解了有几个因素会增加 ETL 功能的复杂性。

数据抽取本身可以完全依赖于源系统的性质和复杂程度。源系统的元数据必须包含每一

个数据库和每一个数据结构的信息。你需要很细节的信息，包括数据库大小和数据的流失程度。在不接触操作型系统用途的条件下抽取数据的时候，你必须了解每一天的时间窗口。还要决定在每一个相关源系统中获取数据变化的工作机制。这些都是费劲而且耗时的工作。

数据转换功能中的工作可以包括全部的转换方法。你必须重新定义内部数据结构，对数据重新排序，在转换技术中应用多种形式，给缺失值增加新的默认值，而且你还必须设计性能优化所需要的整个集合。在很多情况下，你需要将 EBCDIC 转化成 ASCⅡ的格式。

现在，轮到数据装载功能了。最初的载入工作可能包括迁移数据仓库中数以万计的数据行。创建和管理这么多的装载映射不是一件轻松的工作。验证以及将这些映射关系应用到数据仓库的物理环境中，是更加困难的事情。有的时候，会花费两个或者更多的时间来完成最初的物理装载。

关于抽取和应用正在进行的逐渐增长的变化，有几个困难的地方。针对独立的源系统寻找合适的抽取方法可能就会很费劲。一旦你决定了抽取的方法，如果数据仓库不能经受很长的停机时间，那么寻找应用这些变化到数据仓库中的时间窗口，就可能会很棘手。

ETL 的需求和步骤

在我们进入与 ETL 相关的最关键的部分之前，让我们先来回顾一下功能步骤。对于最初的大量刷新和追加的数据装载来说，它的顺序应该是这样的：追加的变化的触发，更新和增长的载入的过滤，数据抽取、转换、整合、清洗，以及应用到数据仓库中去。

在 ETL 过程中什么是最主要的步骤？下面让我们来看看图 12-1 种列出的要点。这些主要步骤中的每一步都对应了一系列的行动和任务。使用这个图表，作为你的数据仓库 ETL 过程的参考。

事实表的 ETL

维度表的 ETL

为所有的数据装载编写过程

组织数据准备区域和检测工具

关于集合表的计划

决定数据转换和清洗规则

建立全面的数据抽取规则

准备从源系统到目标数据元素的数据映射关系

决定所有的数据源，包括内部和外部

决定数据仓库中需要的所有的目标数据

图 12-1　ETL 过程的主要步骤

下面列出了组成整个 ETL 过程的行动和任务的类型。这个列表决不是所有数据仓库都必须包括的内容，但是给出了完成 ETL 过程中需要考虑方面。

- 将几个源数据结构组合成数据仓库中目标数据库中的一个单独行。
- 将一个源数据结构分成几个结构后放入目标数据库中的几列。
- 从数据字典和源系统目录中读取数据。
- 从多种文件结构中读取数据，包括平面文件、索引文件（VSAM）、旧系统数据库（分等级的/网络化的）。
- 载入移植事实表的细节。
- 将数据从源系统平台上的一种格式转换成另一个目标平台上的格式。
- 得到输入字段（例如：从出生日期得到年龄）的目标值。
- 将隐藏的值改变成为对用户有价值的值（例如：用将 1 和 2 转变成男性和女性）。

关键因素

在我们继续下面的内容之前，让我们来看看一些关键的因素。第一个与数据抽取和转换功能的复杂性有关。第二个是关于数据装载的功能。

记住，数据抽取和转换功能的复杂性的主要原因是源系统中巨大的差异性。在大型企业中，我们可能会有很多种类的计算机平台、操作系统、数据管理系统、网络协议和旧系统。你需要特别注意这些数据源，并且仔细的对它们进行调查。将这种调查作为一个开始，找出数据抽取的所有细节。在数据转换功能中遇到的困难，也会与源系统的巨大差异相关。

现在，让我们来看看数据装载功能。通常，无论是最初的载入还是阶段性的更新，这种更新都会带来困难，不仅是因为复杂性，还包括载入工作本身需要过长时间的原因。你会必须找到合适的时间，来计划这些完整的刷新。追加的装载有几种类型的困难。首先，你必须决定从每一个源系统中获取数据的最好的方法。其次，你必须在不影响源系统的情况下执行获取的工作。然后，另一方面，你必须在不影响数据仓库用户使用的情况下，计划好装载的

工作。

在你的数据仓库项目中执行 ETL 功能的时候，要特别注意这些关键因素。现在，让我们来分别看看 ETL 功能的每一个部分。

数据抽取

作为一个 IT 专家，在实施操作型系统的时候，你一定参加过数据抽取和转换的工作。当你从一个面对 VSAM 文件的系统转换成一个新的使用关系数据库技术的订单处理系统的时候，你可能编写过数据抽取的程序来从 VSAM 文件中获取数据，将它们移植到关系数据库中。

这里有两个主要的因素来区别新操作型系统中的数据抽取和数据仓库中的数据抽取。首先，对于一个数据仓库来说，你必须从很多全异的系统中抽取数据。其次，对于数据仓库来说，你必须根据增长型装载工作以及最初一次性完全装载的变化来抽取数据。对于操作型系统来说，你所需要的就是一次性的抽取和数据转换。

这两个因素增加了数据仓库中数据抽取工作的复杂程度，而且，也促使在内部编写代码和脚本的基础上，第三方数据抽取工具的使用。使用第三方工具通常会比内部编程要贵一点，但是它们记录了它们自己的源数据。另一方面，内部的编程增加了维护的成本。如果你的公司处于商业条件不断变化的行业中，那么你就要尽量减少内部编程的使用。第三方的工具同能可以提供内在的灵活性。你需要做的就是改变它的输入变量。

有效的数据抽取对于数据仓库的成功非常重要。因此，你需要特别关注这个部分，阐明数据仓库的抽取策略。下面列出了数据抽取的一些要点：

- 数据源确认——确认数据源的系统和结构。
- 抽取方法——针对每个数据源，定义抽取过程是人工抽取还是基于工具。
- 抽取频率——对于每个数据源，建立数据抽取的频率，每天、每星期等等。
- 时间窗口——对于每个数据源，表示抽取过程的时间窗口。
- 工作顺序——决定抽取任务中的开始点是否必须等到前面的工作成功完成，才能开始。
- 意外处理——决定如何处理输入中遇到的问题。

数据源确认

让我们来看看这些要点中的第一个，数据源确认。我们接下来会了解其它的部分。当然，数据源的辨认覆盖了对全部正确数据源的辨认。对于数据源的辨认不仅仅是对数据源的简单辨认，还要包括检查和确定数据源是否可以提供数据仓库需要的值。

假设数据库中的一部分，也许是数据集市中的某一个，用来利用订单数据提供战略信息。根据这个目的，你需要存储关于订单状态的历史信息。如果有多种传递渠道来出货，你就需要获取这些渠道的数据。如果你的用户对分析订单的状态和处理过程很感兴趣，那么你就需要从这些订单状态中抽取数据。

在订单处理的事实表中，你需要关于整个订单数量、折扣、佣金、希望运输时间、实际运输时间和不同处理阶段的时间等等的属性。你需要关于产品、订单部署、运输渠道和客户的维度表。首先，你必须确定你是否有源系统来提供数据集市需要的数据。然后，从源系统中，你需要建立数据集市中每一个数据元素对应的正确的源数据。而且，你还要了解每一个过程，来保证确认的数据源是否真的是需要的。

图 12-2 中描述了关于订单履行的数据源确认的一个逐步方法。数据源确认不是一个简单的过程。它是数据抽取功能中十分重要的第一个步骤。你必须了解需要在数据仓库中存储的每一项信息的数据源确认过程。而且，数据源确认的过程需要大量的时间和复杂的分析工作。

数据源	数据源确认过程	目标
订单处理	列出对事实表进行分析所需要的每一个数据项或事实	产品数据
客户	从所有维度中列出每一个维度属性	客户
产品	对于每个目标数据项，找出源系统和源数据项	运输渠道数据
运输合同	如果一个数据元素有多个来源，选择需要的来源	部署数据
出货跟踪	确认一个目标字段的多个源字段，建立合并规则	时间数据
存货管理	确认一个目标字段的多个源字段，建立分离规则	订单度量
	确定默认值	
	检查缺失值的源数据	

图 12-2　数据源确认：一个逐步的方法

数据抽取技术

在检查不同数据抽取技术之前，你必须十分了解你所抽取的源数据的性质。而且，你还需要了解抽取的数据是如何使用的。

商业交易造成了源系统中数据的变化。在很多例子中，源系统中的属性值就是在那个时刻的属性值。如果你来看源操作系统中的每一个数据结构，每天的商业交易一直在改变着这些结构中的属性值。当一个客户改变地址的时候，有关这个客户的数据就会在源系统中的客户表中发生变化。当两个附加包类型增加到一个即将销售的产品中时，源系统中的产品数据就会发生变化。当对订单数量进行修改的时候，源系统中关于这个订单的数据也会随之发生变化。

源系统中的数据是依赖于时间的。这是因为源数据是根据时间变化的。一个单独变量的值根据时间变化的。而且，我们再来看看前面那个关于一个客户改变地址的例子。在操作型系统中，用户的当前地址是最重要的。当前的变化本身，也就是说从一个地址变化到另一个地址，是不需要保存的。但是需要考虑这些变化对于数据仓库中信息的影响。如果通过地址编码来分析一些问题，例如销售情况等，我们就需要了解地址变化前后的销售情况。换句话说，在数据仓库中不能忽略历史信息。这也给我们带来了问题：如何从源系统中获取数据？答案依赖于数据究竟如何在源系统中进行存储。所以让我们来了解数据是如何在源操作型系统中保存的。

操作型系统中的数据。这些源系统通常用两种方式来存放数据。源系统中的操作型数据一般来说分为两类。所使用数据抽取技术的类型依赖于这两类数据的性质。

当前值。源系统中的大多数数据属于这个类型。这里存储的属性值体现了这个时刻的属性值。这个值是暂时的。当商业交易发生的时候，这个值就会发生变化。没有办法来预测当前数值存在的时间，以及什么时候会发生变化。客户姓名和地址、银行账户余额和每一张订单的金额等，都是这个类型的一些例子。

那么对于这种类型数据的抽取如何实施呢？一个属性值在交易发生之前是不会发生变化的。当它发生的时候是没有预先通知的。数据仓库中变化的历史数据的抽取，对于这个类型的数据而言是很棘手的。

周期性的状态。这类数据与前面的类型不同。在这个类型中，属性值存储的是每次变化发生时的状态。在每一个时间点，状态值根据产生新值的时候进行存储。这类数据还包括关

于每一个事件发生时的事件。例如，关于保险政策的数据通常存储在保险公司的操作型系统中。操作型数据库存储了关于政策每一个变化的状态信息。简单来说，对于保险索赔来说，每一个事件，例如索赔开始、确认、评估和解决，都要存储关于时间点的数据。

对于这个类型的操作性数据，变化的历史存储在源系统本身中。因此，针对数据仓库中保存历史信息的目的的数据抽取工作是相对比较容易的。无论状态数据或者关于事件的数据，源系统中都在任何变化发生的时候，保存每一个时间点的数据。

让我们来看看图 12-3，确定你对于存储在操作型系统中的这两类数据的理解。请特别注意下面这些例子。

属性的例子　　不同日期操作型系统中存储的属性值

存储当前值

属性：客户的地址信息

存储周期性状态

属性：一家拍卖行委托财产的状态

图 12-3 操作型系统中的数据

在了解了数据是如何存储在操作型系统中后，我们现在来讨论数据抽取的技术。当你实施数据仓库的时候，从某一特定时间开始的最初的数据必须迁移到数据仓库中。这就是最初的装载。在最初的装载之后，你的数据仓库必须保持更新，是变化的历史和状态可以在数据仓库中反映出来。总的说来，从源操作型系统中，主要有两种数据抽取的类型，分别是静态数据和修正数据。

静态数据是在一个给定时刻获取的数据。就像是对相关源数据在某个特定时刻的快照。对于当前或者暂时的数据来说，这个获取回包括所有需要的暂时数据。而且，对于周期性数据来说，这一数据获取将包括每一个源操作型系统中可以获得到每个时间点的每一个状态或者事件。

一般在数据仓库的最初装载的时候使用静态的数据获取。有的时候，会需要一个对于一张维度表的完全的刷新。举个例子来说，假设源系统中的产品管理需要完全修改，你会发现做一个对目标数据仓库中产品维度的完全的刷新，更加容易一点。所以，根据这个目的，你会对产品数据进行静态的数据获取。

修正数据也称为追加的数据获取。严格说来，并不是增加的数据，而是最后一次获取数

据后的修正。如果源数据是暂时的，那么对修正的获取也不是容易的工作。对于周期性状态数据或者周期性事件来说，追加的数据获取包括特定时刻的属性值。

追加的数据获取可能是立刻进行的或者可以缓期的。在立刻的数据获取中，有三种不同的选择。可以缓期的数据获取有两种不同的选择。

立即型数据抽取。在这个选择中，数据抽取是实时的。当交易发生的时候，就会在源数据库和文件中发生。图 12-4 中就是立即型数据抽取的选择。

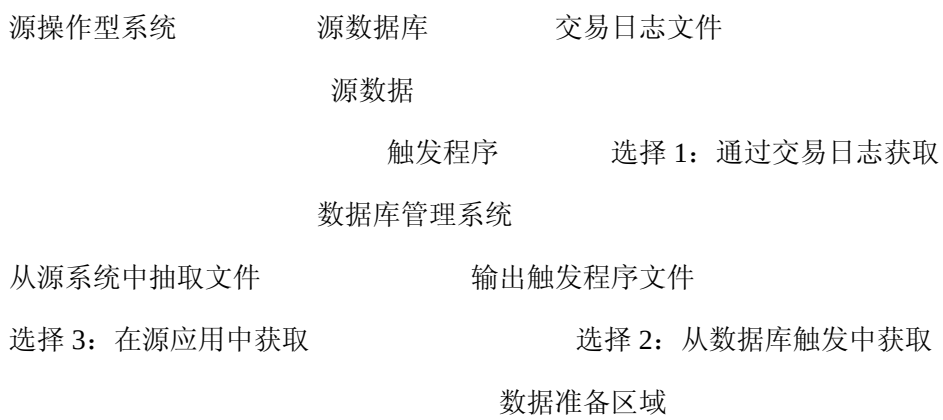


图 12-4　立即型数据抽取：可选方法

现在让我们来仔细看看关于立即型数据抽取的三种选择。

通过交易日志获取。这个选择使用了数据库管理系统中保存的应付各种可能发生灾难所备份的交易日志。对于每一个交易的增加、刷新或者对数据库表的删除，数据库管理系统都需要立刻将其记录在日志文件中。这种数据抽取技术通过读取交易日志，选择所有相关交易记录。在操作型系统中没有额外的管理费用，因为日志的记录本来就是交易处理的一部分。

你必须保证在日志文件更新之前已经抽取了所有的记录。因为在磁盘中存储的日志文件会装满，所以里面的内容会备份到其它的介质中重新使用。要确保根据数据仓库的更新抽取了所有的日志交易。

如果所有的源系统都是数据库应用系统，那么使用这个技术就没有问题。但是如果源系统中的一些数据是编入索引或者其它平面文件的，这个选择在很多情况下就不适用了。你必须应用其它的数据抽取技术。

当我们在学习通过交易日志获得数据的时候，让我们来看看复制的作用。数据复制是一种简单的方法，在一个分布式环境中创建数据副本。图 12-5 中表现了复制技术是如何在数据抽取中使用的。

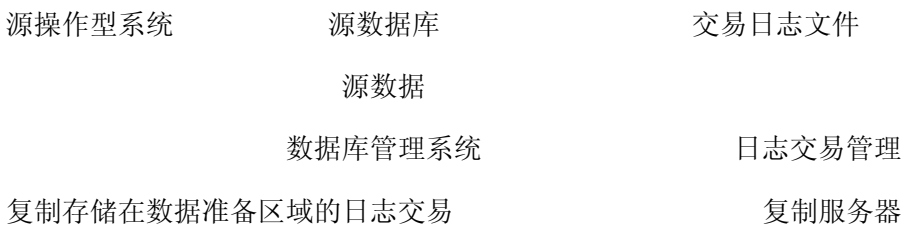


图 12-5　数据抽取：使用复制技术

正确的交易日志包含了所有不同源数据库表的变化。这里是使用复制技术获取源数据变化的主要步骤：

- 确认源系统的数据库表。
- 确认和定义准备区域的目标文件
- 在源表和目标文件之间创建映射
- 定义复制模式
- 计划复制的过程
- 从交易日志中获取变化
- 从日志向目标文件转换获取的数据
- 检验数据变化的转移
- 确定复制的成功或者失败
- 在元数据中证明复制的结果
- 维持数据源、目标和映射关系的定义

从数据库触发中获取。同样，这个选择可以应用于全都是数据库应用的源系统中。如同你知道的那样，触发器是存储在数据库中的特殊过程（程序），在特定的预定义事件发生的时候被触发。你可以为所有需要的事件创建触发器程序。触发器程序的输出被写入一个单独的文件，可以用来为数据仓库抽取数据。例如，如果你需要获取客户表中所有记录的变化，可以写一个触发器程序，来获取所有关于这张表的更新和删除信息。

通过数据库触发器获取的数据是在源系统中发生的，因此非常的可靠。你既可以在映射之前也可以在之后获取数据。但是，建立和维持触发器程序给开发工作增加了新的负担。而且，转换过程中触发器过程的执行给源系统增加了管理费用。此外，这个选择仅适用于数据库中的源数据。

从源应用程序中获取。这个技术也是针对应用程序的数据获取。换句话说，就是源应用程序用来帮助数据仓库中的数据获取。你必须修改相关的应用程序，写入源文件和数据库。

要对程序进行修改，写入所有对源文件和数据库表的增加、刷新和删除。然后另一个抽取程序可以使用独立的包含源数据变化的文件。

与上面的两个方法不同，这项技术可以用在所有类型的源数据中，无论是数据库文件、索引文件还是其它的平面文件。但是你必须修改源操作型系统中的程序，并对其进行维护。如果源系统程序的数量非常大的话，这会是一个困难的任务。而且，因为增加了获取独立文件的变化的处理过程，这项技术还会降低源应用程序的性能。

可以延缓的数据抽取。在上面讨论的例子中，数据抽取发生在源操作型系统中交易发生的时候，数据抽取是即时的。相反，可以延缓的数据抽取技术不会实时的抽取变化。图 12-6 中表现了可以延缓的数据抽取的方法。

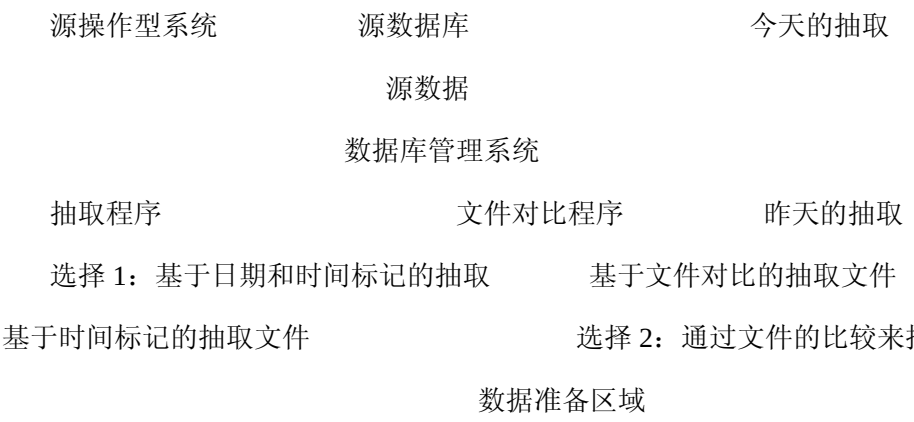


图 12-6 可以延缓的数据抽取：可选方法

现在让我们来讨论有关这种抽取的两种可选方法。

基于日期和时间标记的抽取。每次创建或者更新源记录的时候，都会有一个关于时间和日期的标识。时间标记提供了数据抽取中选择记录的基础。在这里数据的获取会在以后的时间中发生，而不是在每一个源记录创建或更新的时候。如果你在每天的午夜时间运行数据抽取程序，那么每天你都将抽取前一天午夜以后的日期和时间标记的数据。如果修改的工作量比较少，那么这项技术就是可行的。

当然，这项技术首先是假设所有的相关源记录中包括了日期和时间的标记。假设这一点是真实的，那么基于日期和时间标记的数据获取就可以执行在任何类型的源文件上。这项技术可以获取源数据的最新状态。两次数据抽取工作之间的任何中间状态都会丢失。

源记录的删除会带来问题。如果一个源记录在两次抽取之间删除了，关于删除的信息就不会被发现。你可以通过为首先的删除在源记录上进行标记，运行抽取程序，然后物理的删

除这项记录。这意味着你必须在源应用程序中增加更多的逻辑。

通过文件的比较来抽取。如果在你的环境中上面的这些方法都不适用，那么可以试试下面的方法。这项技术也称为快相微分技术，因为它是通过在源数据的两个快相之间的对比进行工作的。

假设你要将这项技术应用在获取产品数据变化上。在对产品数据的变化进行每天的数据抽取工作时，你对今天的产品数据和昨天的数据进行了完全的比较，也比较了记录并发现有插入和删除的操作。那么你就获取了这两个数据之间的变化。

这项技术使所有相关源数据的优先复制成为必需。虽然比较简单和直接，但是在一个大型文件中的完全比较的工作也可能会是比较困难的。但是，这对于一些没有交易日志或时间标记的旧数据源来说，可能是唯一可行的方法。

技术的评估

总结来说，下面的这些方法都是数据抽取可用的：

- 静态数据的获取
- 通过交易日志获取
- 通过数据库触发器获取
- 在源应用系统中获取
- 基于日期和时间标记的获取
- 通过文件的比较来获取

你会遇到一些大问题。你的环境中应该使用哪种方法呢？当你在实施中进行数据仓库最初的迁移的时候，会使用静态数据获取技术。然后，你通常会发现你需要这些技术方法的一个组合。如果你有旧的系统，你甚至会需要使用比较文件的方法。

图 12-7 中强调了这几种不同方法的优点和缺点。请仔细学习这个部分的内容，用来决定在你的实际环境中究竟使用哪一项技术。

静态数据获取

灵活性较好

源系统的性能不受影响

对已有的应用不需要修改

不能用在旧系统中

不能用在面向文件的系统中

需要使用产品提供商，没有内部成本

通过交易日志获取

不是那么灵活

源系统的性能不受影响

对已有的应用不需要修改

可以应用在大多数旧系统中

不能用在面向文件的系统中

需要使用产品提供商，没有内部成本

通过数据库触发器获取

不是那么灵活

对于源系统的性能有一点影响

对已有的应用不需要修改

不能用在大多数的旧系统中

不能用在面向文件的系统中

需要使用产品提供商，没有内部成本

在源应用系统中获取

很好的灵活性

对于源系统的性能有一点影响

对已有的应用有很大的修改

能用在大多数的旧系统中

能用在面向文件的系统中

因为内部的工作带来很高的内部成本

基于日期和时间标记的获取

很好的灵活性

源系统的性能不受影响

可能会对已有的应用有很大的修改

不能能用在大多数的旧系统中

能用在面向文件的系统中

需要使用产品提供商

通过文件的比较来获取

很好的灵活性

源系统的性能不受影响

对已有的应用不需要修改

可能可以用在大多数的旧系统中

可能可以用在面向文件的系统中

需要使用产品提供商，没有内部成本

图 12-7　数据获取技术：优缺点比较

让我们来看一些通常的评论。其中的哪一项技术在实施中比较容易而且便宜？考虑使用交易日志和数据库触发器的技术。这两种技术都可以通过已经存在的数据库产品来提供，都相对比较便宜和方便实施。基于交易日志的技术可能是最便宜的技术，在源操作型系统中不需要增加管理费用。在数据库触发器的例子中，需要创建和维护触发器程序。即使这样，对于源操作系统的维护工作和额外费用，相对于其它的技术也并不是很多。

在对源系统的数据获取中，开发和维护的工作是最昂贵的。这项技术需要对已有源系统的实质的修改。对于很多旧的应用来说，寻找原来的编码并且修改并不都是可能实现的事情。但是，如果源数据并不是存在于数据库文件中，或者没有日期和时间的标记，那么这就是唯一的选择。

会对源操作型系统的性能有什么影响？当然，可以延缓的数据抽取方法对操作型系统的影响是最小的。基于时间标记的数据抽取和基于文件比较的数据抽取都是在源系统的外部工作的。因此，当希望尽可能减小对操作型系统的影响的时候，这两种方法就是适用的。但是，这些可以延缓的获取方法会带来不准确的数据。它们记录了从源系统在当前抽取时刻的变化到前一次抽取时刻的变化，却不能获取任何中间的变化。因此，无论对暂时的源数据如何操作，你都只能得到历史的近似值。

所以我们的底线是什么呢？保守的在源系统中使用数据获取技术，因为它需要太多的开发和维护工作。对于数据库中的源数据，通过交易日志或数据库触发器获取数据是第一位的选择。在这两者之间，通过交易日志的获取因为其相对较好的性能，是相对好一点的选择。而且，这项技术可以应用在非关系型数据库中。文件比较方法是最费时间的方法，只有在所有方法都无法使用的时候，选择这个方法。

数据转换

通过利用上面我们讨论的这些技术，你已经设计好了数据抽取的功能。现在抽取得到的数据是没有经过加工的数据，不能直接应用于数据仓库。首先，所有抽取的数据必须要成为数据仓库可以使用的数据。拥有可以用来进行战略决策的信息，是数据仓库最根本的原则。你知道操作型系统的数据不能满足这个要求。然后，因为操作型数据是从很多旧系统中抽取的，这些系统中数据的质量可能不会像数据仓库要求的那样好。你必须在数据仓库使用之前提高数据的质量。

在将从源系统抽取的数据装载数据仓库之前，你不可避免的必须执行各种类型的数据转换工作。你必须根据标准对数据进行转换，必须保证虽然集合了很多的数据，数据的组合不能违反任何商业规则。

要考虑数据仓库中需要的数据结构和数据元素。现在考虑所有的从源系统中抽取相关数据。从源数据格式、数据取值和数据质量中，你可以知道你不得不进行多种类型的转换工作来使原始数据符合数据仓库的要求。对源数据的转换需要经过多种处理，将所有抽取的源数据转换成可以存储在数据仓库中的信息。

很多公司都低估了数据转换功能的范围和复杂程度。他们从一个简单的部门数据集市开始，几乎所有的数据都从唯一的源应用中获得。数据转换要求字段的转换和对数据结构进行重新定义。不要在进行数据转换的时候犯错误，准备好考虑所有的不同点，将足够的时间和工作量分配到设计转换的任务中去。

数据仓库方面的专家曾经试图将数据转换分成几种方法，例如大略的将数据转换分成简单转换和复杂转换。还有一些关于语义上的混淆情况。例如，有人将数据整合作为数据转换功能的一部分，实际上数据整合的工作是属于数据预处理的一种。还有人将数据整合作为源字段和目标字段之间的映射。排除这种分类的想法，我们来讨论转换功能的几种常见类型。你自己来决定在你自己的数据仓库环境中使用简单的或者是复杂的转换类型。

数据转换中的一个重要任务就是提高数据质量。简单说来，包括了对已抽取数据中的缺失值进行补充。数据质量对于数据仓库非常重要，因为急于不正确信息的战略决策的制定会带来破坏性的后果。因此，我们将在下一章中讨论数据质量的问题。

数据转换：基本任务

不考虑源操作型系统的变化和复杂性，也不考虑数据仓库的范围，你会发现你的数据转换功能中的大部分可以分成一些基本任务。让我们来看看这些基本任务，然后你可以从一个基本的视角来看数据转换功能。这里列出了一些基本任务：

选择。这发生在整个数据转换过程的开始部分。你选择整个记录或者是从源系统得到的部分记录。选择的任务通常构成了抽取功能本身的一部分。但是，在某些例子中，源结构可能不会服从数据抽取期间的必要部分的选择。在这种情况下，就要谨慎的抽取整个记录，然后进行选择的操作。

分离/合并。这项任务包括数据处理的类型。有的时候（并不是经常），你会在数据转换过程中对部分源记录进行分离的操作。在数据仓库环境中，对很多源系统中部分选中部分的合并操作，是更加普遍的现象。

转化。这是一项包括一切的任务。它包括多种对单独字段的基本转化，有两个主要原因：一是在数据转换中对不同源系统进行标准化，二是使这些字段对用户来说可用和可理解。

概括。有些时候你会发现，在数据仓库中无法用较低的粒度保存数据。可能是你的用户不需要分析或查询最低粒度的数据。比如说，对于一个零售连锁店来说，就不一定需要每一个收款机上的每一项交易的销售数据。在数据仓库中只需要存储每天每个商店关于每种商品的数据。所以，在这个例子中，数据转换功能就要包括对每天的销售数据进行产品和商店的汇总。

丰富。这项任务是对单个字段数据进行重新分配和简化的过程，使它们更容易被数据仓库环境所使用。你可以用一个或相同输入记录中的多个字段，创建一个更好的数据视图。当一个或更多的字段发源于多个系统的时候，这项原则就要扩展。

主要转换类型

你已经了解了转换的基本任务。当我们在考虑一些特定抽取数据结构的时候，你会发现你所需要的转换功能可以通过我们讨论过的一些基本任务的组合来完成。

现在让我们来看看转换功能的主要类型。下面列出了一些常见的转换类型：

格式转化。你经常会遇到这个问题。这些转化包括数据类型和单个字段长度的变化。在你的源系统中，产品类型通过代码和名称表示在数值型和文本数据类型的字段中。而且，这

些类型会根据不同的源系统有所不同。对这些数据类型进行标准化，将它们改变成文本格式而向用户提供有意义的值。

字段的解码。这也是数据转换的一个常见类型。当你在处理多个源系统的时候，你会遇到相同数据项用过多字段值描述的问题。一个典型的例子就是，对性别的解码，一个源系统使用 1 和 2 来表示男性和女性，而另一个系统使用 M 和 F。而且，很多旧系统使用的是隐藏编码来表示商业规则。在客户文件中 AC、IN、RE 和 SU 等都表示什么意思？你需要对所有这样的隐藏编码进行解码，将它们变成用户可以理解的值。

计算和导出数值。如果你想要知道连同销售和成本在一起的利润情况，数据仓库如何计算？从销售系统抽取的数据包括销售额、销售单位和每一个产品的成本估计。你必须在数据存入数据仓库之前，计算整体成本和利润率。平均每天的收支差额和操作比率就是导出字段的例子。

单个字段的分离。早期的旧系统将客户名称、地址和雇员数存储在大型文本文件中。名字的第一个字、中间的字和最后一个字都存在一个字段中。相同的是，一些早期系统将城市、地区和邮政编码也存在一个字段中。你需要在数据仓库中使用不同的字段，来存储姓名和地址的不同部分。这样做有两个原因：首先，你要通过对单个部分的索引来提高操作的性能；其次，用户需要通过使用单独的部分，例如城市、地区和邮政编码等，进行分析的操作。

信息的合并。这个数据转换的类型不是照字面意思那样意味着对几个字段进行合并，创建一个新的数据字段。例如，关于某一个产品的信息可能会从不同的数据源中获得。产品编码和描述从一个数据源得到，相关类型从另一个数据源中得到，成本数据从第三个数据源中得到。在这个例子中，信息的合并表示了产品编码、描述、类型和成本的组合，成为一个新的实体。

特征集合转化。这种数据转化类型与特征集合的转化是相关的，在数据仓库中有一种标准的文本数据特征集合。如果你的源系统中有大型机的旧系统，这些系统中的源数据就可能是 EBCDIC 的特征。如果你的数据仓库选择了基于微机的体系结构，那么你必须将大型机的 EBCDIC 格式转换成 ASCⅡ格式。当你的源数据是另一种硬件和操作系统的类型时，你就会遇到类似的特征集合转化。

度量单位的转化。现在的很多公司都有全球机构。在很多欧洲国家的度量单位都是公制单位。如果你的公司有海外机构，你就必须将公制的单位转化成标准的度量单位。

日期/时间转化。这个类型与日期和时间的标准格式有关。例如，美国和英国的日期格式可能是全球的标准格式。2000 年 10 月 11 日在美国写成 10/11/2000，而在英国就写成

11/10/2000。必须要对这些写法进行标准化统一。

概括。这个类型的转换是创建装载数据仓库的汇总。例如，对于一个信用卡公司来说，在分析销售模式的时候，就不需要在数据仓库中存储每一个信用卡用户的每一笔交易。相反，你要对每天的交易进行汇总，存储汇总数据。

键的重新构造。在从输入数据源中抽取数据的时候，看一看抽取记录的主键。你必须对这些关于事实和维度的索引键进行操作。请看图 12-8，在这个例子中，产品编码有固有的含义。如果你将产品编码作为主键，就会有问题。如果产品转移到另一个仓库中，仓库中的关于产品的键就要改变。这是一个旧系统中时常出现的问题。当在数据仓库的数据库表中选择索引键的时候，要避免选择这样已经有含义的键。将这样的键转换成系统自己生成的普通键。这就叫做键的重新构造。

生产系统键

产品编码

国家代码　仓库代码　　销售领域　产品编号　销售人员

数据仓库——产品键

图 12-8　数据转换：键的重新构造

数据整合和合并

ETL 功能的真正挑战是从很多不同的源系统中将所有的源数据放在一起。即使在今天，大多数的数据仓库都要从旧的大型机系统、小型机应用和一些较新的客户机/服务器系统的组合中抽取数据。这些源系统中的大多数都没有相同的商业规则。它们经常会有一些不同的命名规则和不同的数据表示标准。图 12-9 中表现了一个典型的数据源环境。

大型机　小型机　UNIX 系统

多种特征集合（EBCDIC 或者 ASCⅡ）

多种数据类型

缺失值

没有默认值

多种命名标准

商业规则不统一

不兼容的结构

矛盾的取值

图 12-9　典型的数据源环境

数据的整合是将所有相关的操作型数据组合成一致的数据结构，准备载入到数据仓库中去。在另一个主要转换规则应用之前，你可能会认为数据整合和合并是一个类型的预处理过程。你必须对命名和数据展现方式进行标准化，解决相同数据载不同源系统中表达不一致的问题。虽然很费时间，很多数据整合任务都是必需的。下面，让我们来看看一些可能会遇到的困难。

实体识别问题。如果有三个不同时期开发的旧系统，你就有可能有三套不同的客户资料。一个系统可能是旧的订单登记系统，另一个可能是客户服务支持系统，第三个可能是市场系统。这三个系统中的客户名单应该是差不多的。每一个文件中的相同客户可能都有一个唯一的识别码，这些唯一的识别码在这三个系统中就可能是不相同的了。

这就是识别的问题，你不知道哪些客户记录是与相同的客户相关的。但是在数据仓库中，你需要为每一个客户建立一个独立的记录。必须能够从多种源系统中得到单个客户的行动，然后将这些数据组合成一条单独的记录载入数据仓库中。这是一个很普遍但是也非常困难的问题。这种类型的问题在多个数据源中存在相同实体的地方非常普遍。厂商、供货商、雇员和产品都是容易发生这种问题的实体。

在上面的例子中，你必须设计复杂的算法来将所有三个文件中得到的记录进行匹配，建立匹配记录的集合。没有任何匹配算法可以完全解决这个问题。如果匹配的标准太紧，那么就会漏掉一些记录。另一方面，如果匹配标准太松，一个特定的集合就可能包含多个用户的数据。你需要让用户也参与这个过程，必须衡量与源系统相关的问题，决定如何操作这些实体识别问题。每一次执行数据抽取功能，都要在载入数据仓库之前解决实体识别问题吗？会对数据仓库有怎样的影响？一些公司，根据他们自己的条件，采用两个阶段来解决这个实体识别问题。在第一个阶段中，所有的记录，无论是否复制过，都作为唯一的一条记录。第二个阶段，通过周期性的自动算法和手工的识别，协调重复的记录。

多数据源问题。这是影响数据整合的另一种问题，虽然比较少见，相对于实体识别问题也没那么复杂。这个问题是由于一个数据元素有超过一个数据源造成的。举一个例子，假设

产品的单位成本可能从两个系统中得到。在标准的成本应用中，在一个特定周期内对成本值进行计算和刷新。你的订单处理系统也执行所有产品的单位成本。从两个系统中得到的成本项的存在一些细微变化。从哪个系统中你可以得到数据仓库存储的成本呢？

一个直接的解决方法就是分配一个高级别给这两个系统中的一个，从这个系统中得到产品的单位。有的时候，这样的直接解决方法对于数据仓库用户来说并不是很好。你可以根据最新的刷新日期来选择一个文件。或者，在有些情况下，你可以根据其它的相关字段来选择合适的源系统。

维度属性的转换

在第十一章中，我们讨论了维度表属性的变化，回顾了这些数据变化的类型。而且，我们还讨论了处理这三种变化维度的方法。第一种类型的变化是错误的纠正。这些变化应用到数据仓库中，不需要保存历史。第二种类型的变化是在数据仓库中保存历史。第三种类型的变化是试验性的变化，在用户需要使用两种方法（有变化和没有变换两种）分析单位的地方。

为了正确的应用这些变化，你需要对这些输入变化进行转换，准备将数据载入数据仓库中。图 12-10 中表现了源系统中数据抽取的变化是如何转换和准备数据装载的。这个图表列出了每一个维度表变化的类型，分别列出了类型 1、2 和 3。让我们仔细来看看这个图表，对解决方案更好的掌握。你一直会遇到关于维度表变化的问题。

源系统维度的数据变化	执行数据转换功能 决定维度变化的类型	执行数据清洗功能 合并和整合数据
类型 1	类型 2	类型 3
将生产键转化成已有代理键 创建装载映射	将生产键转化成新代理键 创建装载映射	将生产键转化成已有代理键 创建装载映射（包括有效日期）

图 12-10　维度变化的转换

如何实施转换

正是因为数据转换的复杂性和范围，仅仅是手工的操作方法显然是不够的。你不能沿用

在实施操作型系统的时候编写的转化程序。数据转换的类型要复杂和挑战的多。

你可能想要使用的方法取决于一些重要的因素。如果你考虑将大多数数据转换功能采用自动的方法，首先要考虑是否有时间来选择工具，以及工具的配置和安装，训练项目团队使用工具，将工具整合到数据仓库环境中。数据转换工具一般比较昂贵。如果数据仓库的范围比较适度，那么项目的预算也不会有数据转换的空间。

让我们来看看与使用手工技术和数据转换工具用法相关的一些要点。在很多例子中，将两种方法很好的结合起来是很有效的方法。要在可能的时间和预算额之间找到最好的平衡点。

使用转换工具。在最近几年中，转换工具在功能性和灵活性方面都有了长足的发展。虽然使用转换工具的理想目标是排除手工的方法，但是在实际中却不是可能实现的。即使你有最复杂转换工具组合，都要时刻准备好使用内部编码。

使用自动的工具当然会提高效率和准确性。作为一个数据转换专家，你必须详细说明各种参数、数据定义和转换工具的规则。如果你对工具的输入是正确的，那么剩下的工作也会比较有效。

使用转换工具会有一个主要的优点，那就是工具的元数据记录。当你确定转换参数和规则的时候，这些都会作为元数据存储起来。然后这些元数据会成为数据仓库整个元数据组成部分的一部分，可以被其它部分共享。当由于商业规则或者数据定义发生变化而带来的转换功能变化时，你必须将这些变化输入你的工具。转换的元数据会自动进行调整。

使用手工技术。直到最近，当转换工具开始在市场上出现的时候，手工技术都是主要使用的方法。手工技术很适用于小一点的数据仓库。在这里手工编写的程序和脚本执行每一项数据转换的工作。一般说来，这些程序都是在数据准备区域执行的。具有一定知识和经验的分析师和程序员都可以编写这样的程序和脚本。

当然，这种方法会带来精细的编码和测试。虽然最初的成本是可以预测的，但是后来的维护工作使成本越来越高。与自动的工具不同，手工的方法可能更容易发生错误。

一个主要的缺点是和元数据相关的。自动的工具记录它们自己的元数据，但是如果你需要存储和使用元数据的话，内部程序就必须有不同的设计。即使内部程序记录了最初的数据转换元数据，每次变化都会影响转换规则，必须对元数据进行维护。这会给手工编码转换程序的维护工作带来额外的负担。

数据装载

通常人们认为一旦创建了装载映射，转换功能就停止了。接下来的主要功能包括获得准备好的数据、将它们装载数据仓库并存储在数据仓库中。你要为数据仓库数据库中的目标文件创建装载映射。

向数据仓库中转移数据的整个过程可以有几种方法。你一定听说过应用数据、装载数据和刷新数据的阶段。为了看得更加清楚，我们使用了下面这些名词来表示这些阶段：

最初装载——第一次对所有的数据仓库表进行迁移

增量装载——根据需要用一种周期性的方式应用追加的变化

完全刷新——完全删除一个或多个表的内容，重新载入新的数据（最初装载是对所有表的刷新）

因为数据仓库的装载工作需要大量的时间，装载工作通常需要很多关注。在载入过程中，数据仓库必须是脱机的。在不影响数据仓库用户的前提下，你需要找到一个时间窗口。因此，考虑将整个载入过程分为小一些的部分，这将给你带来两个好处。你可以并行处理多个较小的载入。而且，你还可以在装载一部分数据仓库的同时，保持另一部分的正常运行和工作。估计装载的时间是一件困难的事情，特别是最初装载和完全刷新。一定要对装载进行测试，保证正确性并估计载入的时间。

当你开始装载的时候，不要希望每一个源装载映射文件中的每一项记录都可以成功的应用到数据仓库中。对于你希望载入的事实表，连接键可能会是错的或者与维度表不符合。对于不能顺利载入的映射要提供一定的过程来进行处理，而且要对载入记录的质量保证制定一个计划。

如果数据准备区域和数据仓库数据库在同一个服务器中，那么这会节省你的工作量。但是如果你必须将装载映射送到数据仓库服务器上，就要仔细考虑实施方案，选择一种最适合的方法。通过 Web、FTP 和数据库连接都是可以选择的方法。你必须考虑需要的带宽和对网络传输的影响。考虑数据压缩和对意外事件的计划。

什么是应用数据的常用方法？最直接的方法就是写一个专门的装载程序。根据你的数据仓库的大小，装载程序的数量可以很大。管理大量的装载程序是一件有挑战的事情。而且，维护这些程序也需要大量的时间和精力。和数据库管理系统一起工作的装载效用提供了一个装载的快速方法。将这个方法作为一个主要的选择。当准备区域文件和数据仓库在不同的服务器上时，数据库连接就是有用的方法。

你已经意识到数据装载的一些问题和困难。项目团队必须非常熟悉这些常见的挑战，才能够找到正确的解决方案。现在让我们继续来看数据装载技术和过程的知识。

应用数据：技术和过程

在本章前面的部分，我们定义了三种类型的数据仓库数据应用：最初装载、追加装载和完全刷新。考虑数据是如何应用在这三种类型中的。让我们来看看产品数据的例子。对于最初装载来说，你从很多源系统中抽取所有产品的数据，并对数据进行整合和转换，然后为项产品维度表中载入数据创建载入映射。对于增量装载，你要收集产品数据载前一次抽取后发生的变化，通过整合和转换过程运行变化，创建应用到产品维度表中的输出记录。完全刷新与最初装载类似。

在每一个例子中，你都创建了数据文件，应用到数据仓库的产品维度表中。你如何将数据应用到数据仓库中呢？采用什么样的方式？数据可以采用下面四种方式进行应用：装载、追加、破坏性合并、建设性合并。请仔细看看图 12-11，理解这四种方式的作用。让我们来具体解释每一个方式是如何工作的。

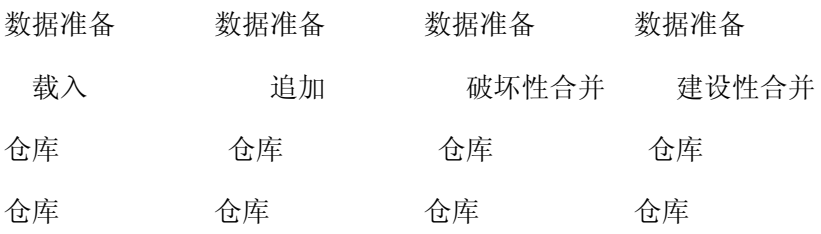


图 12-11 应用数据的方式

装载。如果要载入的目标表已经存在，而且也有数据存在于表中，载入过程就会抹去已有的数据，从输入文件中应用新的数据。如果在载入之前表已经是空的了，那么载入的过程就仅仅是从输入文件中应用数据。

追加。你可能会将追加的过程看成是装载的延伸。如果表中已经存在数据，那么追加的过程会无条件的增加输入数据，在目标表中保存已有的数据。当一个数据记录与已经存在纪录相同的时候，你要定义如何处理输入的副本。输入记录可能可以作为副本增加进去，也可以在追加的过程中进行丢弃。

破坏性合并。在这个方法中，你将输入数据应用到目标数据中。如果输入数据记录的主

键与一个已经存在的记录的键互相匹配，那么就对目标记录进行更新。如果输入记录是一个新的记录，那么就将这个输入记录增加到目标表中。

建设性合并。这个方法与破坏性合并有一点不同。如果输入记录的主键与已有记录的键相匹配，那么就保留已有的记录，加入输入的记录，并将增加的记录标记为旧记录的替代。

这些应用数据到数据仓库中的方法可以分为三类，下面我们分别来对它们进行讨论。

最初装载。例如你可以用一次运行对整个数据仓库进行装载。作为这个运行的变化，你能够将装载过程分成一些不同的子装载，作为独立的载入来运行这些子装载。换句话说，每一个载入过程都会从头开始创建数据库表。在这些例子中，你可以使用上面讨论过的装载方法。

如果你需要超过一个过程来创建一个表，而且按照计划你的载入过程必须要运行几天，那么这个方法就是不同的。对一个表的最初装载的第一次运行来说，可以装载的方法。下面的过程可以使用追加的方法来应用输入数据。

要特别注意最初装载的索引创建，或者完全刷新。大规模装载的索引创建可能会非常耗时。所以你可以在载入工作完成以后，重新建立索引。

增量装载。这些是源系统中正在进行的变化的应用。源系统的变化总是依赖于特定的时间，无论是否基于源系统的外在时间标记。因此，你需要一个方法在数据仓库中保存这些周期性的变化性质。

让我们来看看建设性合并。在这种方法中，如果输入记录的主键与一个已有记录的键相匹配，已有记录会在目标表中保留，加入输入的记录，并将增加的记录标记为旧记录的替代。如果时间标记也是主键的一部分，或者时间标记包括在输入和已有记录的之间，那么建设性合并可以用来保存周期性变化的性质。但是，关键是建设性合并方法是增量载入的合适的方法。必须根据不同目标表的性质来决定具体实施。

有没有适合破坏性合并的应用方法？类型 1 是如何改变维度的？在这个例子中，一个维度表记录的变化意味着纠正了已有记录的错误。已有记录必须被正确的数据记录取代，所以你可能会使用破坏性的合并方法。这种方法对于任何不重要的历史性观察的目标表也是适用的。

完全刷新。这个类型的数据应用会周期性的重写整个数据仓库。有的时候，你也可能会对一些特定的表进行刷新。部分的刷新很少见，因为每一个维度表都与事实表紧密相关。

就数据应用方法而言，完全刷新与最初装载比较相似。但是，在完全刷新中，数据在应用数据之前，就已经存在于目标表中。已经存在的数据必须在数据载入之前删除。与最初装

载一样，载入和追加的方法都可以应用在完全刷新中。

数据刷新和更新的对比

在最初装载之后，你可以通过两个方法对数据仓库进行维护：

更新——对数据源中增加的变化的应用

刷新——在特定周期中完全的重新载入

从技术上说，刷新比更新要简单很多。使用更新的方法，你必须想出正确的策略从每一个数据源中抽取变化。然后你必须选出最好的策略将这些变化应用到数据仓库中。而刷新的方法仅仅是周期性的对数据仓库表进行完全的代替。但是刷新工作需要大量的时间。如果你必须每天执行刷新工作，你可能会不得不将数据仓库停下来。如果你的数据库有比较大的表，那么情况就更麻烦。

刷新和更新这两种方法究竟在什么样的情况下孰优孰劣呢？图 12-12 中是更新和刷新之间的比较。刷新的成本不需要考虑源系统变化的数量。如果变化数量一直增长，完全刷新的时间和精力保持不变。另一方面，更新的成本会随着记录的数量保持增长。

在最初装载后，数据仓库可以通过两种方法维护

更新——对数据源中增加的变化的应用

刷新——在特定周期中完全的重新载入

成本　　更新　　刷新

变化的记录

图 12-12 更新和刷新的对比

如果需要更新的记录数在整个记录数的 15%和 25%之间，无论采用更新还是刷新的方法，每个记录的载入成本一般是相同的。这个范围是一个大概的分类。如果超过 25%源纪录每天都在发生变化，那么需要考虑完全刷新了。通常说来，数据仓库管理员一般使用更新的方法。有时候，当一些主要的结构发生变化或者发生大规模的改变，你可能也需要对数据仓库完全刷新。

维表的过程

在数据仓库总，维表包括了分析所需要的属性，例如销售额和成本。我们已经知道，客户、产品、时间和销售区域都是维表中的例子。对维表的维护过程包括两个功能：第一，对表的最初装载；其次，将变化应用到一个正在进行的基础上。让我们分别来看这两个功能。

第一个是关于源系统中记录的键，以及数据仓库中记录的键。对于前面已经讨论过的原因，我们在数据仓库的记录中不使用生产系统中的键。在数据仓库中，使用系统自己创建的键。源系统中的记录有它们自己的键。因此，在源数据应用到维表之前，生产系统的键必须转换成数据仓库系统自己创建的键。你可以将键的转换作为转换功能的一部分，或者你可以在真正的载入功能开始之前单独进行。独立的键转换系统相对较好。

下一部分的内容与类型 1、类型 2 和类型 3 的维度变化应用相关。图 12-13 中表现了这些不同的类型是如何操作的。

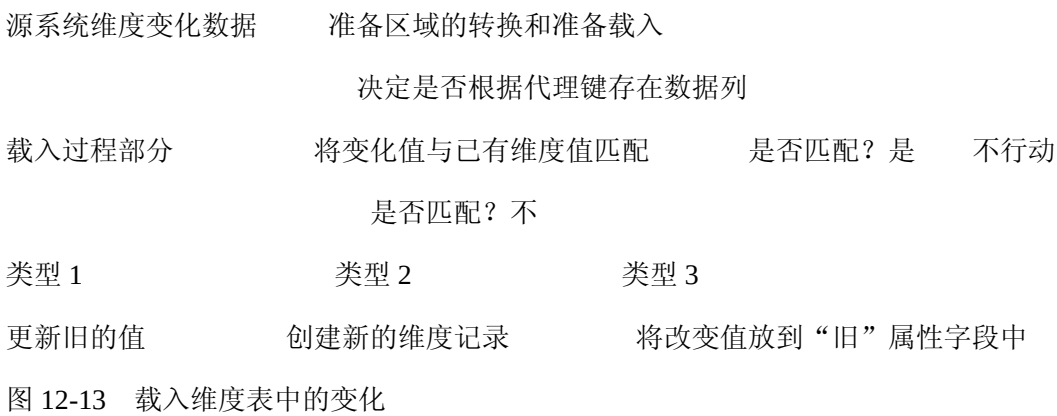


图 12-13　载入维度表中的变化

事实表：历史装载与增量装载

事实表的键是维度表的键的串联。因此，正是因为这个原因，维度记录首先要载入。然后，在载入每一个事实表记录之前，你必须为事实表记录创建串联键。

这里有几个关于事实表历史装载的提示：

- 确认历史数据对于数据仓库有效
- 定义和改进抽取商业规则
- 获取审计数据，再返回操作型系统

- 执行事实表代理键
- 改善事实表内容
- 重新结构化数据
- 准备装载文件

下面这些是关于事实表增量装载的一些有用的提示：

- 对事实表的增量抽取

由新交易构成

由更新交易构成

使用数据库交易日志

- 事实表的增量装载

尽量频繁的载入

使用分区的文件和索引

应用并行的处理技术

ETL 总结

到现在为止，你应该已经确信数据仓库的数据抽取、转换和装载功能包括了很广泛的范围。通常任何操作型系统中与开发联系在一起的转化功能的范围和复杂程度，都无法和数据仓库环境中的 ETL 功能相比。数据仓库中的数据抽取功能可能会包括几个不同的源系统。作为一个数据仓库开发者，你需要仔细检查这些不同的源系统，找到合适的数据抽取方法。我们已经在本章中对这些方法进行了深入的讨论。数据仓库的数据抽取不仅仅是一个一次性的事件，而是一个周期性的功能。

数据仓库中数据转换有很多的类型，不仅仅是一个字段到字段的转化。在我们的讨论中，我们考虑了很多常见的数据转换类型。我们能够考虑的这些类型不可能是完全彻底的。在你的数据仓库环境中，你可能还会碰到其它的数据转换类型。

数据仓库的数据载入与新操作型系统的载入相比有什么不同？针对一个新操作型系统的实施而言，你只需要对数据进行一次的转换和装载，就可以开始新的系统。向一个数据仓库装载数据的过程不会因为最初实施的完成而停止。就像抽取和转换一样，数据的装载也不仅仅是一个数据仓库开始时候的最初行为。除了最初的数据装载，你还有增量的数据装载和周期性的完全刷新。

幸运的是，很多厂商已经开发了关于数据抽取、转换和装载的强有力的工具。你不再是自己使用简单的手工方法来处理这些问题了。你可以有厂商提供灵活而合适的解决方案。第三方工具覆盖的功能范围很宽。你可以在 ETL 处理的每一个部分都得到有效的工具。工具可以从多种数据源中抽取数据、执行转换功能、进行大规模的载入。让我们来看看关于 ETL 工具的选择。

ETL 工具选项

厂商们提供的 ETL 工具一般主要分为下面三种：

1. 数据转换引擎。其中包括动态而复杂的操作算法。这些工具根据用户定义的时间间隔，从一组指定的源系统中抽取数据，执行复杂的数据转换，将结果导入一个目标环境中，将数据应用到目标文件。这些工具给你提供了最大可能的灵活性，选择最合适的数据转换方法，实施完全更新和增量载入。这些工具的功能包括了整个 ETL 过程。
2. 通过复制获取数据。这些工具中的大部分使用数据库管理系统维护的交易日志。在交易日志中获取的源系统的变化可以近乎实时的在数据准备区域被复制，等待进一步的处理。有些工具可以通过数据库触发器的使用来扶植数据。这些数据库中特定的存储过程用信号通知复制代理，获取并传送这些变化。
3. 代码生成。这些工具直接处理数据的抽取、转换和装载。通过创建执行这些功能的程序代码，完成这些过程。代码生成器会创建 3GL/4GL 的数据抽取和转换程序。你提供数据源的参数和目标输出及其商业规则。这些工具可以生成大部分常见程序语言的程序编码。当你想针对一些工具没有涉及部分，增加部分代码的时候，你可以自己进行编码。你可以在任何你想要的地方用自己的代码替代这些工具自动生成的代码。

更明确的说，ETL 工具能做什么？在下面列出的这几点中，请你仔细考虑，在你的数据仓库工具中是否需要这些特性：

- 从多种领先厂商的关系型数据库中抽取数据
- 从旧数据库、索引文件和平面文件中抽取数据
- 源字段和目标字段从一种格式向另一种格式，进行的数据转换
- 执行标准转化、重定义索引键和结构性变化

- 从数据源到目标的检查轨迹的提供
- 抽取和转换中商业规则的应用
- 将源系统中的几个记录组合成一个整合的目标记录
- 元数据的记录和管理

强调 ETL 中的元数据（Metadata）

第九章中详细介绍了数据仓库的元数据。我们在数据仓库的三个主要功能区域中讨论了元数据的规则和重要性。我们还对这三个主要功能区域中元数据的获取和使用进行了回顾。数据获取和数据存储中的元数据与 ETL 功能相关。

当你在执行部分或整个 ETL 功能时使用第三方工具的时候，这些工具的大部分对它们自己的元数据进行记录和管理。即使元数据是工具自己的格式，但是也是有用的。ETL 元数据包括了关于源系统、源数据和数据仓库目标数据结构之间的映射、数据转换和数据装载之间的信息。

但是就像你知道的那样，你所选择的工具可能不能完全执行所有的 ETL 功能。你必须增大内部编码的比重。对于数据抽取、转换和载入功能的每个部分，你都会有可能使用项目团队自己编写的代码。根据环境的条件，这些内部程序可能会在数量上有所不同。虽然内部程序可以给你更多的控制权和灵活性，但是有一个缺点。与使用 ETL 不同，自己编写的内部程序不会自己记录和管理元数据。你必须另外拿出精力和时间来处理元数据。虽然我们已经在前面的章节中广泛的讨论了元数据，我们还是要重申你必须保证在使用 ETL 功能的内部程序中不能忽略元数据。所有的商业规则、源数据信息、从来源到目标的映射、转换和装载信息必须在元数据字典中手工记录。这对于使元数据部分完整和正确来说非常重要。

ETL 的总结和方法

让我们来总结一下本章所谈到的功能点，如图 12-14 所示。

数据抽取

从不同源系统和外部源中进行抽取

数据整合

将从多种数据源中得到的所有相关数据组合起来

数据总结

基于预先定义的过程创建总计数据集合

元数据更新

维护和使用抽取、转换和装载功能的元数据

数据转换

根据转换规则的转换和重新结构化

数据清洗

根据清洗规则进行清洗和充实

最初数据装载

将最初的数据应用到数据仓库中

正在进行的装载

将正在进行的增量装载和阶段性刷新应用到数据仓库中

图 12-14　ETL 总结

你认为本章的长短及其主题覆盖的内容怎么样？你也可以认为这一章的内容就只是强调了数据抽取、转换和装载功能的重要性和复杂性。为什么呢？我们一遍又一遍的强调，源系统的种类和不同性质，都要求我们要特别注意 ETL 过程。首先，多种类和多性质增加了数据抽取的挑战。但是当你考虑不同源系统的数量的时候，数量越多，转换功能就会越复杂，会出现更多的不一致性。

但是，我们需要的是一个系统和实用的方法。任何时候你将一个任务分成两个，可以不需要任何犹豫。例如，寻找将最初装载分为几个子装载的方法。而且，详细的分析也是很重要的。每一个源系统都有自己的困难，你不能对任何源系统要求太高。在从数据源到目标的映射上，要计划足够的时间。建立一个数据转换的最初列表，经过分析后对列表的内容继续增加。

你每天都必须忍受数据的载入。频繁的增量装载对于保持数据仓库的更新来说非常重要。尽量多使用自动的方法来进行增量装载，将内部编码控制在一个合理的范围内。对于元数据的手工维护会带来沉重的负担。我们发现 ETL 功能是非常耗时、复杂和费劲的过程；但是，它们又是非常重要的。ETL 过程的任何缺陷都会在数据仓库中显现出来。你的用户

不会使用有错误的信息。你认为如果使用不正确或不完整的信息，他们会做出什么样的决策？

本章总结

- 数据仓库中的 ETL 功能是最重要、具有挑战性、耗时和耗力的。
- 数据抽取非常复杂，因为源系统的相异性；数据转换非常困难，因为任务的范围很宽；数据装载很有挑战性，因为数据量很大。
- 目前存在的数据抽取技术，每一种都有自己的优缺点。根据你的具体环境选择合适的技术。
- 数据转换功能包括了数据转化、清洗、合并和整合的部分。使用第三方工具和内部开发程序的组合，来实施转换功能。
- 数据装载功能分为最初装载、规律的阶段增量装载、完全更新。有四种应用数据的方法：装载、追加、破坏性合并、建设性合并。
- ETL 功能的工具主要分为三类：数据转换引擎、通过复制获取数据、代码生成器。

思考题

1. 列出你的三个理由，为什么说 ETL 功能是数据仓库环境中最有挑战的。
2. 列出 ETL 处理部分的任意五种类型的行为。其中哪些是耗时的？
3. 源系统的巨大差异是复杂性的主要原因。你同意这个说法吗？如果同意，简单解释一下。
4. 存储在源操作型系统中的数据主要分为哪两类？举例说明。
5. 举出 5 种主要的转换任务类型。针对每一种都举出例子。
6. 简单描述数据整合和合并中的实体识别问题。你是如何解决这些问题的？
7. 什么是重定义键？并解释需要的原因。
8. 定义最初装载、增量装载和完全刷新。
9. 解释应用数据过程中，破坏性合并和建设性合并之间的不同。在什么时候需要使用这些方法？
10. 在什么时候，使用完全数据刷新比增量载入要好？你可以想到一个例子吗？

复习题

1. 连线

1. 破坏性合并	A.实用静态的数据获取
2. 完全刷新	B.从 EBCDIC 到 ASCⅡ
3. 特征转化	C.最后分类的技术
4. 导出值	D.覆盖旧的值
5. 立刻数据抽取	E.新记录替代
6. 最初装载	F.平均每天余额
7. 文件比较方法	G.完全装载
8. 数据丰富	H.使数据更有用
9. 类型 1 维度变化	I.创建抽取程序
10. 代码生成	J.实时数据获取

2. 作为一个通信公司的数据仓库项目团队中的 ETL 专家，给你的项目经理写一个目录，描述一些可能遇到的挑战类型，并提出一些实际的步骤。

3. 你的项目团队决定使用系统日志来从源操作型系统中获取变化。你必须从四个操作型系统中获取增量装载需要的数据，而这些系统都在运行关系型数据库。有四种类型的销售应用。你需要数据来更新数据仓库中的销售数据。做一个假设，描述一下数据抽取的过程。

4. 在你的组织中，假设客户名称和地址是在三个客户文件中维护的，这三个文件支持三个不同的源操作型系统。描述一下在从三个文件合并客户记录时，你可能会遇到的实体识别问题。写一个过程，简单描述你如何解决这些问题。

5. 你是一家大型玩具制造商的数据仓库项目团队的准备区域的专家。讨论把数据应用到数据仓库中的四种方法。选择一种方法，应用在你的数据仓库中，并对你的选择进行简单的解释。

第十三章　数据质量：成功的关键

本章目标

- 清晰的理解为什么数据质量对数据仓库如此重要
- 观察差质量数据所遇到的挑战，学习如何处理的方法
- 了解高质量数据带来的好处
- 回顾数据质量工具的不同类型，了解它们的用途
- 学习数据质量的开端，学习数据质量的实用技巧

假设有一个小错误，看上去微不足道，存在于一个操作型系统中。在从这个操作型系统收集关于客户的数据时，让我们假设用户一直输入了错误的地区邮政编码。关于客户的销售区域代码陷入了困境，但是在操作型系统中，关于地区代码的正确性可能还不是那么重要，因为不需要根据地区邮政编码来向客户邮寄发票。这些邮政编码只是用来作为销售的目的。

现在来看看客户数据的下一个步骤，将它们导入数据仓库中。这个错误的结果是什么？所有数据仓库用户基于邮政编码所执行的分析都会出现严重的错误。一个在操作型系统中不明显的错误，会在数据仓库的结构中造成很大的问题。这个例子可能不会出现在很多实际的数据仓库，但是你会发现这个类型的问题却非常常见。源系统中质量较差的数据会造成数据仓库用户制定较差的决策。

“脏”数据是数据仓库失败的主要因素之一。用于一旦意识到数据存在不能接受的质量，他们就会立刻丧失对数据仓库的信心。他们会放弃使用数据仓库，那么所有项目团队的工作就都会白费，以后也不可能得到这些用户的信任。

大多数公司会过高的估计操作型系统的数据质量。很少有人会使用方法来验证各种操作型系统的数据质量问题。只要数据的质量可以达到完成操作型系统运行的标准，那么普遍的结论是所有企业数据就是合格的。对于有些正在建立数据仓库的公司来说，数据质量没有得到应有的重视。这些公司怀疑存在有问题，但是还没有严重到需要立刻注意的地步。

只有当公司致力于保证数据的质量的时候，他们才会惊奇于数据问题的范围和严重性。即使在他们发现一个很严重的数据问题时，他们通常也会低估对数据进行清洗的工作量，不会积极投入足够的时间和资源来解决这个问题。这个问题最多会被部分解决。

如果数据仓库的源系统中有一些不同的旧系统中，那么首先必须假设你的源数据就是

“脏”的。然后再来确定数据的质量等级。项目团队必须拿出足够的时间和精力，对解决数据质量问题进行计划。在本章中，我们要在数据仓库中定义数据的质量，要了解常见的数据质量的问题类型，在你分析源数据的时候，你可以确定问题的类型和解决方法。我们还将对数据清洗的方法及其可以帮助人们解决问题的工具进行探索。

为什么数据质量如此重要

数据仓库中的数据质量非常重要（这听起来非常明显），而且比在操作型系统中重要的多。根据从数据仓库中得到的信息所制定的战略决策，在范围和结果上的影响都更加深远。接下来我们列出了一些原因，解释为什么数据质量的问题如此重要。根据下面的这些观察，来提高数据质量：

- 增加决策制定的信心
- 更好的为客户服务
- 提高增加更高价值服务的可能性
- 减少错误决策的风险
- 减少成本，特别是营销工作的成本
- 增强战略决策的制定
- 提高生产的效率
- 避免数据污染的影响

什么是数据质量

作为一个 IT 工作者，你一定经常听到数据正确性的问题。正确性是与一个数据元素相关的。例如我们可以考虑一个客户的实体。客户实体有一些属性，例如客户姓名、客户地址、客户地区、客户生命周期等等。客户实体的每一个属性都针对于一个单独的客户。数据正确性，与客户实体的属性相联系，也就意味着一个独立事件的属性值可能正好是那个客户的姓名。数据质量也就是数据正确性，但是又不仅仅是这个。很多清洗步骤主要集中于数据正确性，你需要超出数据正确性的范围。

如果数据相对于它的位置而言非常适合，那么我们可以说这样的数据是有质量的数据。因为，数据质量是与用户所定义的数据项的用途相关的。一个实体中的数据项能够准确反映用户想要观察的东西吗？数据项能够适合用户所定义的目标吗？如果可以，数据项就可以反

映数据质量的标准。如图 13-1 所示，这里列出了数据正确性和数据质量之间的区别。

数据正确性

一个实体的特别例子正确的反映出实体的性质

根据数据库技术定义数据元素

数据元素符合确认的约束性

每一个数据项有正确的数据类型

通常与操作型系统相关

数据质量

数据项完全符合商业用户已经定义的原则

关于公司商业规则相对较宽的概念

不仅与独立的数据元素相关，而且与整个系统相关

整个系统的数据元素的形式和内容

商业用户需要整个公司范围的数据仓库

图 13-1 数据正确性和数据质量的比较

操作型系统中的数据质量是什么？如果数据库记录符合字段确认，那么我们通常说数据库记录的数据质量较好。但是这样单个的字段不能构成数据质量。

数据仓库的数据质量与单个数据项的质量是不同的，但应该是整个系统作为一个整体的质量。不仅是单个字段的数据。例如，在一个订单输入应用上录入关于客户的数据，你可能也会收集每一个客户的基本资料。客户基本资料对于订单输入应用来说是有密切关系的，但是它们没有得到足够的重视。但是当你输入到数据仓库中的时候，就会遇到问题。客户数据作为一个整体，缺乏数据质量。

这就是数据正确性和数据质量之间的区别之一。但是现在如何能够准确的定义数据质量呢？如何通过检查确定一个数据元素是一个高质量的元素？如果是这样的话，需要进行什么样的检查，如何进行检查呢？作为已经处理过大量数据的 IT 的专业人员，我们对于“脏”数据已经有一定的概念，而且知道如何判断一个数据元素是否拥有高质量。但是对于数据质量的模糊定义对于有效的处理数据质量问题来说并不合适。所以，让我们来看一些数据仓库中确认数据质量的方法。

下面的这个列表，是对高质量数据的特征的一个调查。我们将从数据正确性开始，了解

每一个数据质量的维度，用这个列表中的内容来辨认和衡量系统中的数据质量。

正确性。存储在系统中的关于一个数据元素的值是这个数据元素的正确值。如果有一个客户姓名和地址存在一个记录中，那么这个地址就是关于这个姓名的客户的正确的地址。

域完整性。一个属性的数据值在合理且预定义的范围之内。一个常见的例子是性别数据元素可以接受的是“男性”和“女性”的值。

数据类型。一个数据属性的值通常是根据这个属性所定义的数据类型来存储的。当一个姓名字段的数据类型定义为“文本”，所有这个字段包含的姓名都是文本格式，而不会是数字代码。

一致性。一个数据字段的形式和内容在多个源系统之间是相同的。如果在一个系统中产品 A 的产品代码是 1234，那么这个产品在每一个源系统中的代码都应该是 1234。

冗余性。相同的数据在一个系统中不能存在超过一个的地方。因为效率的原因，如果一个数据元素在一个系统中，有多个地方存储，那么系统的冗余性必须清楚的确定。

完整性。系统中的属性不应该有缺失的值。举一个例子，在客户文件中，有一个关于“州”的字段的有效值。在订单文件中，一个订单的每一个具体记录都要完整填写。

重复性。要很好的解决一个系统中记录的重复性的问题。如果产品文件中存在重复的记录，那么每一个产品的所有重复记录都要确认出来并创建一个交叉引用。

符合商业规则。每一个数据项的值都必须符合商业规则。在一个拍卖系统中，销售价格不能低于底价。在一个银行贷款系统中，贷款余额必须是正值或者零。

结构明确。无论一个数据项的结构可以分成不同的部分，这个数据项必须包含定义好的结构。举例说来，一个人的名字可以分为姓和名。姓名的值必须存储为姓和名。数据质量的这一特征可以使执行标准以及减少缺失值的过程简化。

数据异常。一个字段必须根据预先定义的目的来使用。如果地址的字段用来记录通信地址，那么这个字段就必须记录客户的通信地址而不是家庭住址，或者是电话号码等。

清晰。一个数据元素必须拥有所有数据质量的特征，但是如果用户不能清楚的了解它的含义，那么数据元素对于用户就毫无意义。正确的命名习惯可以帮助用户更好的理解数据元素。

时效性。用户决定了数据的时效性。如果用户希望客户维度数据不要超过一天，那么源系统中的客户数据的变化就必须每天都应用到数据仓库中。

有用性。数据仓库中的每一个数据元素必须满足用户的一些要求。一个数据元素必须正确、高质量，但是如果对于用户没有价值，那么数据仓库中的这个数据元素就是完全没用的。

符合数据完整性的规则。源系统中的关系数据库中存储的数据必须符合实体完整性以及参考完整性规则。允许使用空值作为主键的任何数据表都没有实体完整性。参考完整性可以正确的建立父子关系。在一个客户和订单的关系中，参考完整性保证了数据库中一个客户所有订单的存在。

提高数据质量的好处

每一个人通常都了解提高数据质量是一个很重要的目标，特别是对于数据仓库来说尤其重要。坏的数据会导致坏的结论。在这个部分，让我们来回顾一些数据质量所带来的好处。

对实时信息的分析。假设一个大型的零售企业，在全国超过 200 多家的商店中，运行多种类型的每日促销活动。这是一个主要的季节性活动。促销活动是数据仓库中存储的维度之一。市场部门要使用这些促销活动作为主要的维度进行多种多样的分析，控制和了解每个季节促销活动所带来的收益。这对于市场部门进行每天的分析工作来说非常重要。假设促销的细节每一个星期导入数据仓库一次。你认为这些促销数据对于市场部门来说是实时的吗？当然不是。数据仓库中的促销数据对于数据仓库用户来说是高质量的吗？根据前面我们讨论过的质量数据的特点来看，不是这样的。高质量的数据创造了实时的信息，这是为用户创造的一个重要收益。

更好的客户服务。客户服务中正确和完整的信息所带来的好处，不能被过分强调。例如，我们来看看一家大型银行中接电话的客户服务代表。电话线另一端的客户想要了解他的账户中的服务收费情况。银行客户服务代表注意到有 27.38 元在这个客户的账户里面。既然这个客户的账户余额几乎为零，那么为什么客户代表还要对服务费用小题大做呢？但是让我们来看看，假如这个客户服务代表点击这个客户的其它账户，发现他在其它账户上有 35000 元的存款，和超过 120,000 元的债券。那么你认为这个客户服务代表应该接这个电话吗？当然应该。完整而正确的信息能够大大提高客户服务的质量。

更多的机会。数据仓库中的质量数据是一个巨大的市场机会。它给产品和部门之间的交叉销售带来了巨大的市场机会。用户可以选择一个产品的买家，判断他可能感兴趣或者可能给购买的其它产品。市场部门可以进行更有针对性的销售。这是质量数据所能带来的众多机会之一。另一方面，如果数据的质量很差，那么在市场的战争就可能会失败。

减少成本和风险。不好的数据质量会带来哪些风险？明显的风险就是战略决策可能会导致灾难性的后果。其它的风险还包括浪费时间、生产和系统的故障，有甚至会带来客户和商

业伙伴的法律后果。例如在我们刚才的例子中，如果在客户资料中，地址不完整、不正确或者有重复，会浪费很多邮件。

提高生产率。用户应该从整个企业的角度来看待数据仓库的信息。这是数据仓库的一个主要目标。例如，如果从整个公司的角度来看待问题，一家大型商场的销售伙伴可以进行更好的销售过程和战略。

可靠的战略决策制订。这一点需要一再重复和强调。如果数据仓库中的数据是可靠而高质量的，那么基于这些信息进行的决策就是好的决策。没有数据仓库可以为商业增加价值，除非数据是清洁而具有高质量的。

数据质量问题的类型

我们在讨论数据质量对于数据仓库的重要性的时候，已经了解了数据质量的特征。这些特征本身就表明了高质量数据的重要性。我们对于提高数据质量的好处的讨论，加强了我们的认识。接下来，我们要讨论数据质量问题的类型，然后我们才能完整的了解数据质量。这些问题类型的描述，可以帮助你更加了解数据质量的重要性。

如果在一个 20 亿美元的公司的销售系统中有 4%的销售额发生了错误，那么会有多少损失？是 8000 万美元。当一个大的销售公司对于客户的邮件产品目录和内容说明书发生了错误，将会发生什么？如果客户文件中对于一个相同的客户有重复的记录，那么根据重复问题的影响程度，这个公司将会给相同的客户发送多个产品目录。

在最近的调查中显示，数据仓库遇到了这样的问题：数据仓库的开发和使用中最大的问题是什么？让我们来看看图 13-2 中的答案。差不多有将近一半的回答者认为数据质量是他们所遇到的最大挑战。数据质量是最大的挑战，不仅仅是因为数据污染问题的复杂程度，更多的是因为基于这样的数据会给战略决策带来的影响。

数据仓库中的挑战

数据库性能

管理者预期

商业规则

数据转换

用户预期

数据建模

数据质量

回答者比例

图 13-2 数据质量：最大的挑战

现在的大多数数据仓库都是从旧的系统中得到数据的。旧系统中的数据要经历一个衰减的过程。例如，让我们来看看一家零售连锁企业中的产品编码字段。在过去的二十年中，所售产品发生了很多变化。产品编码也一定发生了很多的变化。旧的编码已经不能继续使用，可能其中的一些已经分配给了新的产品。这在操作型系统中不是什么问题，因为这些系统都是对当前的数据进行操作。但是数据仓库中有很多历史数据，这些旧的编码通常会带来问题。

让我们来看看下面的这个列表，是关于数据质量问题的类型。这个列表决不是全部的数据质量问题，但是可以给你在数据质量方面提供参考。

字段中的虚假值。你有没有发现在社会安全号码输入框中有全部是 9 的值的现象？设置该字段的意图是输入正确的社会安全号码。但是，如果你全部填上 9 的话，系统也不会给你改正。有时候，你可能在邮政编码输入框中填上 88888（对亚洲客户）或者 77777（对欧洲客户）。

数据值缺失。这在客户数据中经常出现。在操作型系统中，用户只关心那些向客户邮寄信件所需要的客户数据。他们对那些在操作型系统中没有用的人口统计学数据不太注意。所以，你缺少哪些人口统计学数据，而这些数据对数据仓库的分析非常重要。数据值缺失也包括其它的数据类型。

对字段的非正规使用。你试过几次由于没有备注字段，而让你的用户将它们的建议放在客户联系字段里面呢？这是一种对字段的非正规的使用。

隐含的值。在传统系统中，这是一个普遍的问题。它们中有很多在设计的时候并不考虑最终用户。例如，客户状态代码可能用 R 代表 Regular，用 N 代表 New。然后，在某个时间，另外一个代码 D 代表 Deceased 加上去了。接下来，又加入了一下代码：A 代表 Archive，R 代表 Remove。虽然这个例子只是设计出来的，但是，这种隐含的属性值很容易导致混淆，在旧系统中很常见。

自相矛盾的值。源系统中有一些相关字段的值必须是一致的。例如，州和邮政编码的代码必须一致。你不能够在一个客户记录中对州字段使用 CA(加利福尼亚)，而在邮政编码字

段使用 08817（新泽西州的一个邮政编码）。

违反商业规则。在人力资源和薪酬系统中，有一个明显的商业规则：1 年中的工作日，加上假期，节日，以及病假不能够超过 365 天或者 366 天。任何员工记录的日期树木如果超过了 365 或者 366，就违反了这条基本的商业规则。在一个银行贷款系统中，最低的利率不能够超过贷款的最大利率。

主键重用。假设一个旧系统有 5 位的主键字段分配给客户记录。这个字段只能在客户数不超过 100000 时适用。如果客户数目增加，一些公司通过将旧客户的数据归档，然后对新客户从 1 开始分配主键来解决这个问题。这在操作型系统中是可以接受的，但是，在数据仓库中，你需要从目前的客户文件获取当前的数据，又要从归档的客户文件中获取以前的数据，这样就会遇到一个主键重用的问题。

标识不唯一。标识还可能出现另外一种容易混淆的情况。假设一个会计系统有它自己的产品代码作为标识，但是它和销售系统和库存系统的产品代码不一样。销售系统的产品代码 355 在会计系统中可能是 A226。一个唯一的标志在两个不同系统中代表的是不同的产品。

不一致的值。在一个不断扩展的保险公司中，不同旧系统中的代码规则可能是不一致的，例如在一个系统中 A=Auto，H=Home，F=Flood，W=Workers Comp，而在另一个系统中，对应的代码为 1，2，3，4；还有一个系统用 AU, HO, FL, 和 WO 表示这些信息。

不正确的值。产品代码：146，产品名称：水晶花瓶，高度：486 英寸。在这个记录中，数据是不准确的，产品名称和它的高度不匹配，可能产品代码也是错的。

一个字段多种用途。一个字段同一个数据值在不同部门可能有不同的含义。一个字段开始可能会作为一个存储区域字段，表示储藏室中的存储区域。后来，公司建起了自己的仓库，这个字段用来代表仓库号码。这类问题总是会存在的，因为储藏室代码和仓库代码在同一个字段中。这一类数据污染问题很难解决。

错误的集成。在一个拍卖公司中，买家在拍卖中投标，并在拍卖之后购买拍卖品。卖家通过拍卖公司出售它们的商品。在一个客户可能在一个拍卖系统中是一个买家，而在拍卖品收取系统中作为一个卖家。拍卖系统中客户号码 12345 可能与拍卖品收取系统中的 34567 集成到一起。数据集成的问题在于：拍卖系统中的客户代码 55555 和拍卖品收取系统中的客户代码 55555 代表不同的客户。出现这种问题的原因在于旧有系统过去都是独立开发的。

数据质量带来的挑战

在数据仓库的数据清洗过程中，有一个奇怪而有趣的方面。我们努力的保证数据仓库能够得到干净的数据。我们希望确定数据污染的范围。我们基于数据的情况制定数据清洗计划。这整个过程中奇怪的地方在于，数据污染是在数据仓库之外发生的。作为数据仓库项目组的一员，你正在努力减少哪些在你控制范围之外发生的破坏。

所有的数据仓库都需要历史数据。历史数据一个重要来源是那些陈旧的系统。最终用户经常会使用数据仓库中的旧数据去做战略决策，但是他们却不知道这些数据确切含义。在大多数情况下，旧系统没有具体的元数据。所以，你要在没有合适元数据帮助的情况下去修复旧的操作型系统中的数据污染问题。

数据污染的来源

为了得到一个良好的数据清洗的策略，有必要列出数据污染的一般来源。为什么源系统中的数据会被污染呢？我们来看以下数据污染源的列表。

系统转换。跟踪订单管理系统的演化情况。在 70 年代早期，公司开始使用基于文件的订单系统；订单数据输入到平面文件或者是索引文件中。在订单的输入过程中，并没有什么存货检查或者客户信用检查。在执行订单的过程中，需要使用报表或者打印稿。之后，系统转换成基于 VSAM 文件和 IBM CICS（作为在线处理监控器）的在线订单处理系统。接下来，就是层次数据库的版本了。可能你现在还有这样的订单处理系统。许多公司后来转移到关系数据库应用上面。在这种情况下，经过这么多的转换过程，订单数据会发生什么变化呢？系统转换和迁移是数据污染的重要原因。你应该去了解每一个源系统所经过的转换过程。

数据过时。在回顾产品代码是怎么样在多年之后失效的过程中，我们已经讨论过数据过时的问题了。旧的值在这个过程中丢失了它们的含义，变得不在重要。如果你的源系统中有很多是旧系统，那么请注意这些系统中的过时数据。

异构系统集成。你的源系统中异构系统越多，出现污染数据的可能性就越大。在这种情况下，数据不一致是一个常见的问题。考虑一下你的维表和事实表的来源。如果某一个表的来源是几个异构系统，那么就要注意那些倒入的数据。

拙劣的数据库设计。好的数据库设计会减少数据的错误。数据库管理系统提供了编辑字段的

功能。关系型数据库管理系统则通过触发器和存储过程保持商业规则的一致性。假如实体完整性和参照完整性规则可以防止一些数据污染。

数据条目的不完整信息。在输入数据条目时，如果没有输入所有的信息，通常会出现两类数据污染。首先，没有完全输入所有字段，将导致数据值缺失。其次，如果没有输入的字段是系统规定必须输入的话，那么输入者很可能随便输入一些通用的数据。例如，如果城市字段时必须输入的话，输入者可能会输入 N/A。类似的，输入者可能在社会安全号字段中全部输入 9。

输入错误。以前，数据输入员在将数据输入计算机系统之后，会有第 2 次的校验。在数据输入员完成了一批工作后，另一个人会检查这里面的数据条目。现在，由负责商业处理的用户输入数据。他们的主要工作不是数据输入。数据准确性主要由数据输入框以及用户自己的视力来保证。错误的数据输入是数据污染的一个主要来源。

国际化/本地化。由于商业条件的改变，业务的结构可能会扩展到国际领域。公司将转移到更广阔的地域，面对新的文化。如果一个公司是国际获得，那么源系统中的数据会有什么变化呢？现有的数据元素必须适应新的数据值。类似的，如果一个公司系统将其业务本地化，有一些数据元素的值就会被废弃。公司结构的改变以及由此引起的源系统的修改也是数据污染的一个来源。

欺诈。如果告诉你有些人会千方百计的往系统中输入错误的数据，你千万不要吃惊。错误的输入数据就是为了进行欺诈。小心那些涉及金额以及产品数量的字段。要确保源系统中这一类的编辑框是可靠的。

没有相关规定。在任何企业中，数据质量都不是自然而然就能够确保的。在源系统中防止输入错误数据，确保数据质量，是一项需要认真准备的工作。如果一个公司对数据质量没有相关的规定，它的数据质量就不可能得到保证。

姓名和地址的有效性

几乎每一个公司都会因为姓名和地址的重复而遇到麻烦。一个人可能在不同系统中有多条记录。即使在一个单一的源系统中，一个人也可能有多条记录。但是在数据仓库中，你必须将在很多源系统中存在的多种重复记录中关于一个人的所有行为合并起来。很多情况下都会出现这种类型的问题，无论是客户、雇员还是供应商。

下面来看一个关于拍卖公司的例子。假设有几种类型的客户，他们有各自不同的目的来到拍卖公司寻求服务。他们向公司要求的服务包括销售、通过拍卖购买、登记加入不同拍卖目录等等。其中，可能会有多个不同的旧系统来向这些不同地方的客户提供服务。一个客户可能需要所有这些服务，因此在所有这些旧系统中都有关于这个客户的资料。而另一个客户可能很多次的重复一种服务类型。有的时候，很可能在一个系统中就重复的创建了相同客户的多次资料。客户数据的输入可能发生在拍卖公司与这个客户发生的每一次接触。如果这是一个跨国的拍卖公司，那么客户数据的输入发生在世界范围内的很多拍卖站点上。你能够想象幅。这种重复客户记录的可能性，以及这种类型的数据污染所影响的范围和程度吗？

姓名和地址的数据有两个方法可以获得（见图 13-3）。如果数据输入有多种不同的字段格式，那么就很容易在数据输入的时候造成重复。这里列出了一些输入姓名和地址时可能遇到的问题：

- 没有独立键
- 一行中有很多姓名
- 一个姓名占了两行
- 姓名和地址在同一行中
- 个人和公司的姓名混在一起
- 同一个人有不同的地址
- 同一个客户有不同的名称和写法

单一字段格式　　姓名和地址：Dr. Jay A. Harreld, P.O.Box999
xx 大街 100 号
Anytown,NX12345，美国

多个字段格式　　头衔：Dr
姓
名
地址 1
地址 2
城市
州
邮政编码

国家代码

图 13-3　数据输入：名称和地址的格式

在对重复的客户记录进行处理之前，你需要考虑一些问题。首先，你需要将姓名和地址的数据用多个字段的格式表示出来。那不是一件简单的事情，因为很多姓名和地址都是用自由的格式存放在文本中的。其次，你要设计出寻找重复客户记录的匹配算法。然而幸运的是，有很多的工具可以帮助你对重复的客户信息进行处理。

数据质量低的代价

清洗数据、提高数据质量是需要代价的。虽然数据清洗非常重要，但是你需要估算一下使用低质量数据的代价。你的用户可以帮助你完成这个估算，因为如果数据质量低的话，他们会失去很多可能的机会并做出错误的决定。

以下是可能出现的损失的列表，你需要对这些风险和代价进行详细的估计：

- 根据常规分析方法所做出的错误决定
- 由于“脏”数据失去商业机会
- 由于错误的数据导致要重新运行数据清洗，造成源系统负载过重
- 由于数据不符合规则而导致政府机构的惩罚
- 审计的问题
- 不必要的冗余数据，而占用过多的资源
- 不一致的报表
- 每次修改数据错误所需要的时间和精力

数据质量工具

基于我们上面的讨论，你已经意识到数据仓库中数据质量的重要性。很多公司开始意识

到“脏”数据是数据仓库中的重要问题。

因此，你可想象这些公司会在数据清洗过程中进行投资。很多专家认为，数据清洗仍然没有很高的优先级。但是随着数据质量工具的出现，这种态度正在改变。你可以在源系统中或者在缓存区域中，或者在装载的映像中应用这些工具。

数据清洗工具的目录

通常数据清洗工具会在以下两个方面帮助数据仓库项目组。数据错误发现工具用于识别源数据的不准确和不一致；数据修正工具帮助修改错误的数据。这些工具使用了一系列的算法，对数据进行分析、转换、匹配、合并和修正。

虽然数据错误的发现和数据修订是数据清洗过程中两个不同的部分，但是目前市场上的大多数工具都具有两者的功能。这些工具可以识别和发现错误，也可以清洗并修订被污染的数据。在下面的内容中，我们将讨论这两方面的特性。

错误发现 特性

下面的列表描述了数据清洗工具所能遇到的一些典型的错误发现功能：

- 方便快捷的识别重复记录；
- 辨认出那些超出阈值范围的数据项；
- 找到不一致的数据；
- 检查可接受值的范围；
- 检查不同系统中数据项的不一致性；
- 允许用户辨认和检查数据质量问题；
- 控制数据质量在一段时间内的趋势；
- 向用户报告分析所用数据的质量；
- 解决关系数据库管理系统数据参考完整性的问题。

数据修正 特性

下面的列表描述了数据清洗工具所能遇到的一些典型错误修正功能：

- 规范不一致的数据；
- 改善不同数据源中数据的合并过程；
- 对属于同一个家庭的客户记录进行分群和关联
- 提供数据质量的衡量指标
- 检验可接受的数值

数据库管理系统的质量控制

数据库管理系统本身就可以作为一个工具用于数据质量的控制。关系型数据库管理系统有很多超出数据库引擎的特性（请看下面的列表）。新版本的关系型数据库管理系统可以避免数据仓库的几类错误。

域完整性。提供域数值的编辑功能。如果输入的数据超出了预定值的限制，就停止数据输入。你可以在建立数据字典输入的时候，定义编辑检验。

更新安全性。阻止对数据库的未授权的更新。这一特性将阻止未授权的用户使用不正确的方法更新数据。一些临时的未经训练的用户，如果有更新的授权的话，他们将会不正确的操作数据。

实体完整性检查。保证不会出现有相同主键值的重复记录。避免出现其它字段重复的记录。

缺失值最小化。保证在强制的字段中不出现空值；

参照完整性检查。确保基于外键的关系；不会删除相关父记录。

商业规则一致性。使用触发器和存储过程来确保商业规则的执行。数据库本身有很多特殊的脚本程序，当需要更新或者删除指定的数据项时，触发器就会被自动执行。存储过程可以用来保证输入的数据符合特定的商业规则；也可以通过应用程序调用。

确保数据质量的第一步

虽然数据质量很重要，然而很多公司还是没有意识到这一点。在很多情况下，缺失属性值不能够重新创建；在大量的案例中，数据值很费解，使得数据很难被清洗。这就出现了一些其它的问题。数据应该被清洗吗？如果是的话，真正的清洗需要的花费是多少？在应用数据清洗的过程中，哪一个部分数据的优先级更高？

以下几个因素导致了对数据清洗过程的不重视甚至是拒绝的态度：

- 数据清洗过程乏味而且很耗时间。清洗工作需要组合使用很多第三方工具、编写内部编码以及手工完成很多检查和验证的工作。很多公司不能够维持这样的努力。IT专家也不喜欢这类工作。
- 很多源系统的元数据根本就不存在。如果没有文档，根本不可能更正脏数据。
- 要求确保数据质量的用户通常会有很多商业上的责任。确保数据质量很少得到关注。
- 有时候，数据清洗的工作看上去非常艰巨，很多公司都不想开始确保数据质量的工作。

一旦你的企业决定开始数据清洗的工作，你可能考虑采用两种方法。你可以只让干净的数据进入你的数据仓库。任何受污染的数据在装载之前都必须进行清洗。这是一种很理想的方法，但是检测出错误的数据并进行清洗，需要很长一段时间。

第二种方法称为“需要的时候再清理”。如果用这种方法，你将所有数据原封不动的装载到数据仓库中，之后在进行数据清理。虽然你将数据装载进了数据仓库，但是在数据清洗之前，你不能运行任何查询。如果数据质量有问题，可能会影响用户的信心，而这对数据仓库成功至关重要。

数据清洗的决策

在数据清洗之前，项目组，包括用户，需要做出一系列的决定。数据清洗不仅仅是决定是现在清洗全部数据那么简单的。你要意识到在现实世界中，绝对的数据质量是不存在的。所以要采用“面向目标”的原则，先确定你的目标是什么，然后确定要使用哪一些数据。如果数据仓库的数据必须提供前 25 名顾客的销售额，那么对数据质量的要求是很高的。如果客户的人口统计属性用于挑选下一个营销战役的客户，那么数据的质量可以保持较低的级别。

对于数据清洗，你要回答几个基本的问题，并就这些问题做出决策。在下面的章节中，我们来看一些需要解决的问题：

需要清洗哪一些数据。这是一个基本的问题。它是你和你的用户首先需要回答的。而且，这个决定应该主要是由你的用户做出的，IT 专家只是起辅助的作用。要确定数据仓库要回答的问题的类型；然后找出回答问题所需要的数据。你应该确定每部分数据的价值，并估计对数据进行清洗，或者将脏数据留在数据仓库中对用户的分析所造成的影响。

清洗数据仓库中所有数据的成本是相当高的。用户通常也理解这一点。他们不期望 100% 的数据质量，而是通常只要清洗那些重要的数据，而忽略那些不重要的数据。但是，首先要确定用户对重要和不重要是如何定义的。

在什么地方清洗。数据仓库中的数据是从源系统来的，数据的错误也来自于源系统。然后，抽取出的数据进入缓存区域。装载映像从缓存区域装载进入数据仓库中。所以，理论上，你应该在这个过程中的每一个步骤对数据进行清洗。你也可以采用一些方法，将数据清洗的过程分解到其中的某几个步骤当中。

你会发现，在数据到达数据仓库之后再进行清洗是不现实的，这样会破坏转移和装载数据过程中得到的成果。通常，数据在被装载到数据仓库之前就应该进行清洗了。所以，你只能在两个区域清洗数据。

在缓存区域中清洗数据是相对容易的。在数据到达缓存区域之前，你就得到了关于数据结构和内容的信息。虽然这样做看上去是最好的方法，但是它也有一些缺点。在数据缓存区域将会出现数据污染。源系统也依然会有数据污染的问题。从源系统中产生的报表有可能与从数据仓库中得到的报表不匹配，从而导致混淆。

另一方面，如果你清洗源系统中的数据，这个工作将是复杂、昂贵而且艰巨的。很多就系统没有正确的文档。有一些旧系统甚至没有源代码，修改也就无从谈起。

怎么清洗。这就是如何应用厂商工具的问题了。你使用厂商的工具进行数据清洗吗？如果不是的话，你要编写多少内部代码呢？市场上有很多现成的工具可以用于数据清洗。

如果你决定清洗源系统中的数据，那么你必须找到适合源系统的清洗工具。如果你的系统太过陈旧的话，找这种工具可能不太容易，那就只好自己编写代码了。

如何发现数据污染的范围。在应用数据清洗技术之前，必须估计数据污染的范围。这是一个需要操作型系统、数据仓库潜在用户以及 IT 人员共同参与的工作。IT 人员在其中更是扮演了一个特殊的角色。他们负责安装数据清洗工具，并对用户进行培训。IT 人员必须编写内

部程序。

在前面的章节中，我们讨论了数据污染的来源。参照这个列表，列出在你的数据仓库环境中发现的污染源，然后确定数据污染的范围。例如，在你的环境中，过时的数据可能是一个污染源。如果是这样，那么列出所有作为数据仓库数据来源的旧系统。对那些抽取出来的属性，要检查它们的值，看这些值是否已经没有意义。类似的，应该对每一类污染来源作细致的分析。

请看图 13-4。在这幅图中，你可以找到几种发现数据污染范围的典型方法。你可以将它作为一个参考。

● 从旧版本转换而来的操作型系统通常会造成错误。 ● 从外包公司购买的系统转换而来的操作型系统可能会有缺失数据。 ● 未经检验和审计的外部来源数据可能会有潜在的问题。 ● 由于公司合并造成的应用合并会因为时间压力造成潜在的问题。 ● 如果旧系统的报表不再使用，可能是因为报表中有错误数据。 ● 如果用户不完全相信某个报表，那可能是因为坏数据。	● 如果用户觉得特定的数据元素或定义有含糊的地方，那就应该怀疑。 ● 如果每一个部门都有自己的标准数据（例如客户或者产品）拷贝，那么这些文件里面很可能有被破坏的数据。 ● 如果包含同样数据的不同格式的报表不匹配，那么数据质量可能有问题。 ● 如果用户做了过多的调整，那么可能是因为数据质量差的缘故。 ● 如果生产程序由于数据例外而终止，那么那个系统中很可能有被破坏的数据。 ● 如果用户不能够合并报表，那么可能是因为数据不完整造成的。

图 13-4　发现数据污染的范围

建立一个数据质量框架。你必须处理多种类型的数据污染。在数据清洗的过程中，你要做出一系列的决策。你必须发掘可能的数据污染来源。大多数关注数据质量的公司全面的考虑这

些因素并建立了一个数据质量的框架。这个框架是确保数据质量的基础。它包括了整个行动的系统计划。该框架指出其中参与的人员，他们的角色，以及责任。总之，框架将指导数据质量提升的过程。请参考图 13-5。注意框架中的几个主要功能。

<table>
<tr><td rowspan="4">初始的数据清洗工作</td><td>建立数据质量领导小组</td><td>得到一个一致的、合适的数据质量框架</td><td rowspan="4">接下来的数据清洗工作</td></tr>
<tr><td>识别坏数据所影响的商业功能</td><td>建立数据质量规定和标准</td></tr>
<tr><td>选择那些有较大影响力的数据元素，确定优先级</td><td>定义质量指标参数和基准</td></tr>
<tr><td>IT 专家</td><td>对有较大影响力的数据元素定制清洗计划，并执行数据清洗</td><td>对较少影响的数据元素指定清洗计划，并执行数据清洗</td><td>用户代表</td></tr>
</table>

图 13-5　数据质量框架

谁应该负责？

数据质量问题源自于源系统。所以，源系统数据的所有者是不是应该对数据质量负全部的责任呢？如果数据所有者应该对这些数据负责，那么他们是否也应该对源系统中的数据污染问题负责呢？如果源系统中的数据是高质量的，那么数据仓库中的数据也会是高质量的。但是，在操作型系统中，并没有维护数据质量的角色和职责。这是个严重的问题。操作型系统中的数据所有者通常不参与数据仓库。他们对在数据仓库中保持数据清洁并不关心。

为了构建数据质量框架，应该成立一个领导小组。所有关键人员都应该参与这个领导小组。这些人员应该包括源系统的代表，数据仓库用户，以及负责维护源系统和数据仓库的 IT 人员。领导小组负责分配角色、职责以及资源。领导小组还负责数据质量的审计。

图 13-6 显示了确保数据质量活动中的各个参与角色。这些人代表了用户所在的部门以及 IT 部门。他们是数据质量小组中的重要角色。下面是这些角色的职责：

数据消耗者。使用数据仓库，用于查询、报表和分析。建立可接受的数据质量等级。

数据生产者。对进入数据仓库的数据的质量负责。

数据专家。源系统的数据以及主体相关知识的专家。负责对源系统的数据污染进行识别。

数据规则管理者。负责解决数据转换并最终导入到数据仓库过程中的数据破坏问题。

数据完整性专家。负责源系统中的数据符合商业规则。

数据修正专家。负责使用工具或者编制内部程序，解决数据清洗的技术问题。

数据一致性专家。负责确保数据仓库（以及数据集市）中的所有数据都是同步的。

确保数据质量的活动

数据完整性专家（IT 部门）

数据规则管理者（IT 部门）

数据一致性专家（IT 部门）

数据修正专家（IT 部门）

数据专家（用户部门）

数据生产者（用户部门）

数据消耗者（用户部门）

图 13-6　数据质量：参与者和角色

净化过程

我们都知道要求数据质量达到 100%正确之后才进行装载是不现实的。这种等级的数据质量很少能够达到。那么，我们需要清洗多少数据呢？什么时候才应该停止净化的过程呢？

我们再一次的对谁使用数据以及为了什么目的使用数据这个问题进行讨论。估算以下每一部分数据的成本和风险。用户通常能够容忍某种程度的错误，只要这些错误不至于造成严重的后果。但是，用户需要找到数据错误可能涉及的范围，以及哪一部分数据应该被怀疑。

那么，怎么进行净化处理的过程呢？通过用户的参与，你可以将数据元素按照优先级分成 3 个类别：高优先级，中优先级和低优先级。对高优先级的数据，要达到 100%的数据质

量等级。中优先级的数据越准确越好。对这类数据，你要在数据修正的成本和坏数据可能造成的影响之间进行平衡。低优先级的数据可以在你有时间和资源的时候进行清洗。先从高优先级的数据开始清洗，然后是中优先级的数据。

一个常见的数据破坏问题和重复记录相关。我们前面讨论过，对同一个客户，在源系统中可能有多条记录。而客户活动的记录与这些重复的记录相关联。要确保数据净化过程包括了重复记录的修正技术。这些技术应该包括识别重复记录，然后将所有的活动数据与单一的客户相关联。重复记录通常出现在与人有关的实体中，例如客户，雇员以及商业伙伴。

到目前为止，我们还没有讨论从外部系统中获取的数据的质量问题。外部系统数据的错误数据可能会导致数据污染问题。当然，如果你为外部数据付费，而不是从公开领域获得这些数据，那么你自然可以要求这些数据有高质量。无论这些外部数据来自何处，你都要建立一些数据质量的审核标准。如果外部数据不能通过审核，那么你就要准备好清洗被破坏的数据，并要求一个干净的版本。

图 13-7 显示了整个数据净化过程。请观察途中的过程，看下面的总结：

- 建立数据质量的重要性。
- 建立一个数据质量领导小组。
- 建立数据质量框架。
- 分配角色和职责。
- 选择数据净化处理的辅助工具。
- 预备内部程序。
- 对参与数据清洗的人员进行培训。
- 回顾并确认数据标准。
- 将数据分成高、中、低 3 个优先级。
- 准备数据净化计划，从高优先级数据开始。
- 确保有修正重复记录和审核外部数据的技术。
- 根据制定好的计划进行净化处理。

源系统 数据仓库

数据缓存区域　　受污染的数据　　数据清洗功能　　清洗后的数据

厂商工具　　内部程序

数据质量框架　　IT 专家/用户代表

图 13-7　数据净化过程

对数据质量的实用建议

在实施一个完整的数据质量框架，并为数据质量投入时间和精力之前，我们先看一些实用的建议。记住，确保数据质量是一项需要平衡的工作。100%的数据质量等级是不现实的。另一方面，过多的错误从而影响商业运作也是不能接受的。你必须在数据净化和现有的资源之间取得平衡。以下是一些实用的建议：

- 识别出那些有较大影响的污染来源，从这里开始净化过程。
- 不要试图通过内部程序完成所有的净化工作。
- 工具相当有用，应该选择正确的工具。
- 对标准达成一致。
- 将数据质量与特定的商业目标联系起来。否则，确保质量本身并没有吸引力。
- 让数据仓库项目的高层领导参与到开始的数据净化工作当中。
- 让用户参与开发工作中。
- 如果需要的话，对特定的任务应向外部的专家求助。

本章总结

- 数据质量是关键的，这是因为它能够建立用户信心，提供更好的客户服务，支持战略决策的制定，并降低错误决策的风险。
- 数据质量包括准确性，域完整性，一致性，完整性，结构定义，清晰性，以及一系列问题。
- 数据质量问题包括假数据值，数据值缺失，隐含的值，数据值自相矛盾，违反商业规则，数据值不一致，等等。
- 数据污染是由于数据仓库的数据来源被污染造成的，而这些受污染的数据源是数据清洗

工作更加困难。

- 对公司机构来说，姓名和地址的数据质量低下，是一个严重问题，也是数据清洗要面临的最大的挑战之一。
- 数据清洗工具包括错误发现和错误修正的特性。应该学习这些特性，并在你的数据仓库环境中应用这些特性。
- 数据库管理系统本身可以用于数据清洗。
- 要开始在你的机构中进行确保数据质量方面的工作。在数据质量框架之内，要做出一些数据清洗的决策。

思考题

1. 为什么你认为数据质量对数据仓库是关键的？列出 5 个理由。
2. 解释数据质量除了数据准确性之外，还包括什么。据一个例子。
3. 列出 3 个数据仓库中高质量数据所带来的好处。
4. 举一个例子，说明数据质量问题的 4 种类型。
5. 什么是主键重用问题？这种问题通常在什么时候发生？
6. 描述数据清洗工具中的数据修正功能。
7. 说出 5 种数据污染源的名称。对每种污染源，举一个例子。
8. 列出数据清洗工具的 6 种错误发现特性的类型。
9. 什么是“需要的时候在清洗”策略？它对数据仓库来说是一种好的方法吗？
10. 说出数据质量小组中 3 种类型的参与这。他们的功能是什么？

复习题

1. 匹配下面两列：

1 域完整性	A 查找不一致性
2 数据过时	B 更好的客户服务
3 实体完整性	C 同步所有的数据
4 数据消费者	D 允许的值
5 低质量数据	E 用于跳过数据框

6 数据一致性专家	F 使用数据仓库的数据
7 错误发现	G 异构系统集成
8 数据污染源	H 失去商业机会
9 假数据	I 防止重复键值
10 数据高质量的好处	J 破坏字段值

2. 假设你是一个大型金融机构数据仓库项目组中的数据质量专家，该机构有很多 70 年代的旧系统。检查你可能会有的数据质量问题类型，并对如何处理这些问题给出建议。
3. 讨论常见的数据污染源，并举例子。
4. 你负责为数据仓库选择数据清洗工具。你用什么准则来选择呢？准备一个评估和选择这些工具的列表。
5. 作为一个数据仓库顾问，一个在全州拥有分支机构的大银行雇用你帮助公司开始确保数据质量的工作。列出主要的考虑因素。写出一份描述开始步骤、原则和过程的文档大纲。

第十四章　信息和用户类型之间的匹配

本章目标

- 理解数据仓库庞大的信息潜力
- 把所有将来会使用数据仓库的用户仔细地记录下来，并设计一个实用的方法对他们进行分类
- 深入地钻研信息传送机制的类型
- 使每个类型的用户匹配适当的信息传送机制
- 理解全部的信息传送框架，并研究它的组件

让我们设想，你的数据仓库项目研究小组已经成功地定义所有相关的源系统。你已经抽取和转换了该系统中的源数据，你已经为数据仓库存储库做了最好的数据设计，你已经采用了最有效的数据净化方法，除去了大部分脏数据。在使用了最佳的方法后，你已经把经过转换和净化的数据加载到了数据仓库中。那么，现在该做什么呢？

在最有效地执行完所有的这些任务后，如果你的小组没有提供良好的机制，使得信息能够很好的呈现给用户，那么从用户的角度来看你实际上是徒劳无益。正如你所知道的，数据仓库只为了一个理由而存在，就是为了给你的用户提供战略性的决策信息。对用户来说，信息传送机制就是数据仓库。用户的信息界面决定了你的数据仓库能否取得最终的成功。如果这种界面是直观的、易用的、新颖的和吸引人的，那么用户将会乐于使用数据仓库。如果这界面是难以使用的、麻烦的和令人费解的，这时项目研究小组最好能采取措施，避免出现这种情况。

谁是你的用户？他们想要什么？当然，项目研究小组知道这个答案，并基于这些用户的需求设计了数据仓库。那么你如何为用户提供所需的信息呢？这取决于你的用户是谁，他们

需要什么信息，他们何时何地需要这些信息和他们需要的信息的形式是什么。本章将考虑一个典型数据仓库的一般用户类型和为他们提供信息的方法。

数据仓库的成功很大程度依赖于用户可得到的信息传送工具。选择正确的工具是极为重要的。你必须确保该工具适合用户的实际工作环境。下面将详细讨论信息传送工具的选择问题。

数据仓库的信息

作为一个 IT 专业人员，你已经涉及到为各种用户群体提供不同类型的信息。而且你一定是工作在不同类型的操作型系统上，这些操作型系统将提供信息给用户。企业用户使用操作型系统提供的信息来完成日复一日的工作和交易。如果我们已经涉及操作型系统的信息，并且明白传送给用户的信息所蕴涵的意思，那么在数据仓库的信息传送机制方面，还需要什么呢？

让我们回顾一下数据仓库的信息传送和操作型系统的信息传送有什么不同。如果在数据仓库上可得到的战略性信息在源系统上也能很容易得到，那么我们将不需要数据仓库。数据仓库使用户能够通过获得源系统的数据，并以一种适于查询和分析的格式进行存放，从而支持其做出更好的战略性决策。

数据仓库 VS 操作型系统

操作型系统已经有数据库系统，可以用来查询和输出报表。假如这样的话，那么操作型系统上的数据库和数据仓库上的数据库有什么不同呢？这种不同和这些数据库包含的信息的两个方面有关。首先，它们在信息的用法上不同。其次，信息的价值也不同。表 14-1 显示了数据仓库和操作型系统在用法和价值上的不同之处。

用户从数据仓库中寻找他们所需的信息。他们通过内容来进行导航以寻找到想要的内容。用户写出自己的查询公式，然后运行它们。他们也可以设置自己的报表格式，然后运行

它们得到结果。一些用户可以使用预先定义的查询和预先设置格式的报表，但是大体上数据仓库是一个用户可以自由编辑自己的查询和报表的地方。他们围绕着内容，以很多不同的方式来执行自己的分析，查看自己感兴趣的数据。每一次用户使用数据仓库，他（或她）可能运行不同的查询和报表，而不重复以前的操作内容。这种信息传送方式是交互式的。

将这种数据仓库的使用方式和操作型系统的信息传送方式相比较。操型型作系统允许用户运行自己的查询和设置自己的报表格式的频率是多少呢？在一个存货控制应用系统中，用户经常运行自己的查询或编辑自己的报表吗？几乎从不。因为首先要考虑效率。操作型系统被设计成不让用户在系统中随意操作的系统，因为用户随意的查询可能和系统的执行发生冲突。其次，操作型系统的用户无法准确地知道数据库和元数据或者根本无法访问数据字典的入口，因此交互式分析是数据仓库信息传送机制的本质特征，却几乎没有体现在操作系统中。

对用户来说，数据仓库中信息的价值是什么？操作型系统中信息的价值和数据仓库中的比起来又怎么样呢？举个利用信息来进行营业分析的例子。操作型系统的信息向用户显示了企业是如何运作日复一日的营业活动。这些信息使用户能够监测和控制当前的运作。另一方面，数据仓库的信息给了用户分析收入、利润率、市场渗透和客户结构的增长模式的能力。基于这些分析，用户得以制定战略性决策来保持企业的竞争力和灵活性。看看企业的其它方面，例如销售方面。关于销售，数据仓库信息的价值是面向战略性事务的，诸如市场份额、分布战略、顾客购买模式的预测和市场渗透情况等。虽然这是数据仓库的信息用在销售方面的价值的一个体现，那么操作型系统中信息的价值又是什么呢？它们大部分是用来监测销售额以取消预算限额，以努力获得回头客等等。

我们看到，数据仓库中信息的使用和价值与操作型系统中是有所不同的。这种不同有什么含意吗？首先，因为有这些不同之处，作为一个 IT 专业人员，你不应该企图将操作型系统中信息传送原则应用到数据仓库中，因为数据仓库的信息传送是明显不同的，因而需要不同的方法，应当特别注意数据仓库信息传送中的‘交互式’本质特征。用户期望的是收集信息，并在没有 IT 的帮助下以自己的方式交互式地对数据仓库的数据执行分析工作。为数据仓库提供信息技术支持的所有 IT 专业员工没有为用户运行查询和报表，用户靠他们自己的操作就能完成工作，因此保证了用户可以轻松地以自己的方式使用数据仓库的信息。

信息潜力

在我们观察不同类型的用户和他们的信息需求之前，我们需要对数据仓库庞大的信息潜力有个了解。因为它具有这种巨大的潜力，所以我们必须非常注意数据仓库中的信息传送机制。只有我们完全意识到数据仓库在一个企业的全面管理中起到了关键作用的重要意义，我们才能对信息传送机制予以特别的重视。

全面的企业管理

在每一个企业，都是三种过程控制着企业的全面管理。首先，企业从事计划安排。接下来执行该计划。对计划执行结果的评估紧随其后。图 14-2 显示了这些计划－执行－评估的过程。

让我们看看在这封闭的循环里发生了什么。考虑一个企业计划扩展到一个明确的地域市场。假设你的公司想要提高西北地区的市场份额。现在这个计划依靠宣传活动，改善的服务和客户化的销售进行执行。在该计划被执行后，你的公司想要知道宣传活动和主动销售的结果。基于对该结果的评估，可以制定更多的计划来改变活动的构成或者发起更多的活动。计划，执行和评估的循环就这样持续着，确保企业的正常运转。

非常有趣的是，具有专门信息潜力的数据仓库很适合计划－执行－评估这个循环。数据仓库不仅可以报告过去发生过的事情，而且也可以帮助制定将来的计划。首先，数据仓库有助于制定计划。一旦计划被执行，数据仓库又可以被用来评估计划执行的效果。

让我们回到你的公司想要在西北地区扩展市场的例子。该计划由定义该地区恰当的客户群划分和定义要主推的产品两部分组成。为了实现这个目的，应该有效地利用你的数据仓库来区分和定义潜在的客户群和产品组。一旦这个计划随着宣传活动而执行，你的数据仓库可以帮助用户使用评估和分析活动的效果。数据仓库用户既可以按照产品，也可以按照西北地区的不同区域来分析结果。他们可以将实际销售额和为活动制定的销售目标进行比较，或者和去年的销售额进行比较，也可以和该行业的平均水平进行比较。用户可以估计宣传活动带来的收入增长。这个评估接着可以导致后续更深入的市场计划和执行活动。计划－执行－评估的这个循环对一个企业的成功是至关重要的。

在商业领域的信息潜力

我们举一个独立的例子：如何利用你的数据仓库的信息潜力来帮助企业制定市场扩张的计划和以及评估销售活动的效果。让我们考察该企业的一些方面，使数据仓库对管理环节中的计划和评估都有所帮助。

利润率的增长_为提高利润，管理层不但要懂得利润是怎么和生产线、市场和服务联系起来，而且管理层还必须要深入了解那些生产线和市场如何才能产生更大的利润率。从数据仓库中的信息，可以很理想的为利润率增长制定计划和评估该计划执行的结果。

战略性销售_战略性销售可以显著地推动业务的增长。当管理层研究向上销售和跨区销售给现有的客户和扩展客户的机遇时，他们可以为业务增长制定计划。数据仓库为战略性销售提供了极大的信息潜力。

客户关系管理_客户和企业的交互信息被各种各样的操作型系统所捕获。一个定单处理系统包括客户下的订单；产品出货系统，显示了出货信息；销售系统，获取了卖给客户的产品的清单；应收款系统，包括存款细节和欠款的平衡。这些数据从各种不同的源系统中提取出来，经过了转换和综合，使得数据仓库拥有了关于该客户的全部数据。这样，公司的管理层通过数据仓库的信息可以知道他们各个客户的各种情况。这些知识信息确保可以进行更好的客户关系管理。

公司采购_公司管理层从哪里可以全面了解企业范围内的采购模式呢？通过数据仓库。关于产品和供货商的数据在经过综合后，从各个源系统被汇总到数据仓库，然后数据仓库使公司管理机构能够为流线型的采购过程制定更好的计划。

实现信息潜力_前面我们提到了数据仓库的信息潜力，这种潜力有哪些潜在的意义？数据仓库使用户能在适当的交易历史记录中查看交易数据。各种操作型系统收集了大量的数据，这些数据来自不同类型的商业交易活动。但是这些操作型系统不能在计划的制定和结果的评估上提供直接的帮助。用户需要通过自己到合适的商业历史记录中查看数据来评估结果。例如，在查看西北地区的销售情况时，用户需要从地域，产品，营销和时间等不同角度，查看相应文件中的数据。数据仓库被设计用来沿着这些维对销售等活动进行分析。用户能够找到这些数据，将它们转换成有用的信息，并调整和利用这些信息来制定计划和评估商业活动的结果。

用户通过和数据仓库的交互作用来获得数据，将其转换成有用信息，并实现这些数据的全部价值。用户的交互作用通常通过图 14-3 所示的六个阶段来实现。概括如下：

1. 考虑商业需求并根据商业规则将它定义成可用到数据仓库中的数据。
2. 根据规定的商业规则来获得或选择合适的数据子集。
3. 用计算来丰富选中的子集，例如合计或平均。将代码转换成商业术语。
4. 用源数据将选中的数据和它的商业含义联系起来。
5. 把结果组成一种对用户有用的格式。
6. 以各种方式表现结构化的信息，包括表格、文本、图像和图表。

用户信息接口

为了完成这六个阶段和实现数据仓库的信息潜力，你不得不建立一个可靠的接口，通过它将信息传送给用户。将数据仓库放在一边，而把全部的用户放在另一边，该接口必须能让用户完全实现数据仓库的信息潜力。

这个接口逻辑上是放在中间，使数据仓库的信息能够传送给用户。该接口可以是一套具体的为你的环境定做的工具和程序。从这个意义角度，我们不是讨论一个接口的具体组成，而是只想详细地说明它的特征和性质。在没有深入地分析用户的类型和他们具体详细的信息需求的情况下，我们只能来定义用户信息接口的一般性质。

信息使用模式

当你考虑使用数据仓库的各种方法时，你发现所有的用法可以归结为两个基本的模式或方式。这两个模式都和获取战略性信息有关。记住，我们不是在考虑从操作型系统中获得的信息。

认证模式 在这种模式下，商业用户可以提出一种假设，并提出一系列的问题来肯定或者批判这种假设。让我们来看看在这种模式下如何使信息发挥作用。假设公司的销售部门计划在中南地区为两种产品进行宣传，而且接着执行了这个计划。现在该销售部门想要评估宣

传活动的结果。假设中南地区的销售额已经提高了，在这种情况下这个销售部门使用数据仓库中的信息来帮助他们确认这个假设的正确性。

发现模式 当商业分析员在发现模式下使用数据仓库时，他们不用预先做出某种假设。在这种情况下，商业分析人员希望能够发现客户行为和产品要求的新模式。数据仓库用户事先完全不清楚会从数据仓库中得到什么样的结果集，他们将利用数据挖掘的应用程序和数据仓库中的数据来进行知识发现活动。

我们已经看到，数据仓库用户和数据仓库中信息的交互作用不是在假设认证模式下发生，就是在知识发现模式发生。这个交互的方法是什么呢？换句话说，数据仓库用户和数据仓库的交互作用是通过信息方法来实现呢，还是通过分析方法或数据挖掘技术实现呢？

信息方法 数据仓库用户通过这种方法和查询、报表工具，获得历史数据或者当前数据，执行一些标准的统计分析过程。用户可以对数据进行轻微的或高度的概括，而且可以用报表和图表的格式来显示结果集。

分析方法 和分析方法的字面意思来理解一样，分析方法是数据仓库用户使用数据仓库来执行分析活动的一种方法。用户使用历史的概括数据或详细数据，根据商业维来进行分析活动。商业用户用他们自己的商业术语来引导分析，其中更复杂的分析活动包括下钻、概化和多层次/多视角查看。

数据挖掘方法 信息方法和分析方法都是在认证模式下工作的，而数据挖掘方法却是在知识发现的模式下工作的。

我们已经回顾了信息使用中的两种模式和三种方法。那么在这些模式和方法中使用数据的特征和结构是什么呢？数据仓库用户怎样通过用户－信息接口来获得这些数据呢？有代表性的，通过用户－信息接口获得的信息具有下面的一些特征：

预处理的信息 这些信息包括那些自动创建的、可以使用的日常信息，每月和每季的销售分析报表、汇总报表和日常图表也可以归入这个类型。这使得用户可以简单地拷贝这些预处理的信息。

预定义的查询和报表 这是一套为用户准备的查询模板和报表格式。在用户需要时，可以在这些查询和报表上输入适当的参数来运行它们。有时候，用户也可以对这些模板和格式进行细小的修改。

特别的结构 事实上预定义的查询和报表并不能顾及到数据仓库用户的所有需要，所以这些用户可以使用适当的工具来创建他们自己的查询和报表。通常只有一些超级用户和常规用户才有能力创建他们自己的查询和报表的结构。

最后，我们列出了用户－信息接口的一些必要的基本特征。这个接口必须：

- ●是容易使用的、直观的和对用户有吸引力的
- ●能够清楚地表示出商务中的需求
- ●能够将表示出的需求转化成一套正式的商业规则
- ●能够为了将来的使用而储存这些规则
- ●提供用户修改这些规则的能力
- ●能够根据这些商业规则来选择、操作和转换一些合适的数据
- ●具有一套数据操作和转换的工具
- ●能够正确地链接到数据存储器上，以获得这些选中的数据
- ●能够连接元数据
- ●能够以多种方式创建输出结果的格式，包括文本格式和图形格式
- ●能够建立过程来执行指定的步骤
- ●有一个过程管理工具

行业应用

在这一节中，我们已经清楚地认识到了数据仓库巨大的信息潜力，接着我们会深入地讨论更多的细节。在此之前，我们将停下来举一个产业部门的例子，使我们对如何实现数据仓库的信息潜力有一个新的了解。

制造业：它是授权和服务管理、产品质量控制、订单填写和发送、供应和后勤的一个综合体。

零售和生活消费品：它涉及仓库的布局、产品的捆绑，跨区销售活动和价值链分析活动等一些活动。

银行业和金融业：它主要涉及关系管理和信贷风险管理。

谁将使用这些信息？

你会发现，在配置数据仓库之后的六个月，当前用户的数量几乎增长了一倍。使用大部分的数据仓库都会发生这样的现象。访问数据仓库来获取信息的新用户会是谁呢？你只有知道谁将会来获取信息，你才能恰当地、充分地迎合他们的需要。

任何需要战略性信息的人，包括商业分析人员、商业计划人员、部门经理和高级主管人员，都有希望成为用户组的一部分。数据仓库用户可以建立数据集市来满足用户组中的一些人的特定需要，因此为了迎合这些人的需要你可以定义一个特殊的组。在这个阶段中，当我们讨论信息传送的机制时，我们不考虑信息的内容而只考虑信息传送的实际机制。

每一组用户都有特定的商业需求，而且他们希望能从数据仓库中得到这些需求的答案，因此当我们对用户组进行分类时，最好能从他们希望从数据仓库中得到什么这一角度来理解他们。这些用户在工作范围内怎样使用信息内容呢？每个用户都承担着一个特别的工作职能，他们需要数据仓库中的信息来支持他们的工作，因此我们可以根据这些用户的工作职能和组织级别对他们进行分类。

图 14-4 显示的是一个对用户组进行分类的方法。当你根据他们的工作职能、他们在组织层次上的位置和他们的工作效率对用户进行分类时，你能够更好地理解他们需要什么和如何向他们提供适当形式的信息等这些问题。如果你正在考虑的是一个会计和金融领域内的用户，可以肯定那位用户会比较习惯使用电子数据表和金融汇率。然而对一个客户服务行业内的用户来说，一个能够显示有关每个客户的综合信息的图形用户界面的屏幕是最有用处的。而对销售领域的用户来说，表格形式的信息可能更合适他们的需要。

用户的种类

为了让数据仓库的信息传送机制能够更好地适合用户的环境，你需要彻底地理解用户的种类。一开始我们可以根据用户的工作效率来对用户进行分类以了解每个这样的用户组如何和数据仓库进行交互，通过这种方法来理解用户的种类。

临时用户_这种类型的用户偶尔使用数据仓库，他们需要一个非常直观的信息界面和大的导航按钮，才能较好地使用数据仓库。

常规用户_这种类型的用户几乎每天都使用数据仓库，他们习惯各种选择项的操作，但不会从头开始创建自己的报表和查询，因而需要查询模板和预定义的报表。

超级用户_这种类型的用户非常精通数据仓库的使用技术，他们能够从头开始创建报表和查询。其中的一些人甚至会写自己的宏和脚本，能够把数据仓库中的数据导入到电子制表软件和其它应用程序中。

现在我们要改变一下角度，通过用户希望的和数据仓库交互以获取信息的方法来考察用户类型。

预处理的报表_使用日常报表，并且在有规律的时间间隔内运行和传送这些报表。

预定义的查询和模板_输入自己的一套参数来运行具有预定义模板的查询和具有预定义格式的报表。

有限的特别访问_从头开始创建并运行一些有限的简单类型的查询和分析。

复杂的特别访问_有规律地从头开始创建复杂的查询并运行分析会话过程，这为预处理和预定义查询和报表提供了基础。

现在让我们从用户的工作职能这个角度来考察用户组。

高级主管人员和经理_这种用户需要信息来帮助他们做出高级战略性决策。基于主要度量的标准报表对这些用户是有用的，但他们更喜欢使用定制的、个人化的信息。

技术分析员_这种用户希望具有复杂分析、汇总分析、下钻和多层次－多视角查看的能力和访问整个数据仓库的自由。

商业分析员_这种用户虽然对技术了解的比较多，但不懂得如何从头开始创建他们需要的查询和报表。预定义的导航按钮对他们是有帮助的。他们想要以多种不同的方式来查看结果，而且他们能在某种程度上修改和定制预定义的报表。

面向商业的用户_这些是喜欢用鼠标来操作的用户图形界面的知识工人，他们希望拥有标准的报表和一些方法来进行特别的查询。

刚才我们已经回顾了一些用来理解用户如何分组的方法。现在我们可以根据用户的访问

和他们对信息传送的实践和喜好，把这些方法归在一起并标上用户的种类。请看图 14-5，它显示了对用户进行分类的一种方法，而且这种方法已经被很多数据仓库专家和从业者所采用。该表显示了五种主要的用户类型，并且指出了每种用户类型的基本特征，同时也把在组织层次中处于不同位置的用户分配到特定的用户类型中。

虽然这种分类方法看起来显得新奇和有趣，但你将会发现它有助于我们更好地理解每组用户的特征。你可以把任何用户放入任何一种用户类型中，而且通过观察用户的工作效率、组织级别、信息需求或者使用数据仓库的频率时，你可以识别出该用户属于这些用户组中的哪一组，而这将有助于你更好地满足每个数据仓库用户的需求。总之，如果你能够为‘旅行者’、‘操作者’、‘农夫’、‘勘探者’和‘矿工’提供适当的信息传送机制，那么这相当于你已经照顾到了每个数据仓库用户的需求。

他们需要什么

到现在为止我们已经正式完成了对数据仓库用户的主要分类。现在我们要停下来并考虑一下我们是如何实现这一目标的。如果有两个用户的信息访问的特征、工作效率和需要信息的范围都很类似，你就有充分的理由把他们归入一个相同的主要类型中。例如，如果你考虑的是两个不同部门的高级管理人员，他们希望获取信息的方式和愿意获得信息的级别和范围都很类似，那么你可以把这两个主管人员归入‘旅行者’这一类型中。

一旦你把这两个主管人员归入‘旅行者’的类型中，你就可以容易地理解把信息传送给这两个主管人员的信息传送机制的要求，因此你就能够用公式来表示这个要求。如果某种类型的用户需要的信息类型和同一类型的另一用户需要的信息类型是类似的，那么了解这种类型用户的需求，在一定程度上也能帮助你明白如何才能最好地为这种类型的用户提供他们需要的信息。正规的用户分类使你更好地了解数据仓库用户的信息需求，而了解了数据仓库用户的信息需求反过来使你能够建立正确方法来提供信息。信息传送机制的根本目标是为每个类型的用户建立最好的信息传送方法和技术。

上面我们谈到了用户的分类，那么什么是‘旅行者’用户需要的呢？‘农夫’又需要什么呢？每个类型的用户需要什么呢？让我们一个一个地考察每个用户类型并回顾一下该用户类型的信息访问特征，从而最终了解这种用户类型的用户需要。

旅行者_想象一个旅行者要参观一个有趣的地方。首先，这位旅行者已经研究了那个地方的主要特征，知道该地方有着丰富的文化和多种多样的场所。虽然有很多有趣的场所可以参观，但是这位旅行者必须从中挑选一些最有价值的场所去参观。一旦他（或她）抵达了那个地方，旅行者一定能够轻松地选出一些场所来参观。如果旅行者在一些特别的场所发现了吸引人的东西，他（或她）会愿意在该场所多花些时间。

现在让我们把旅行者的故事应用到数据仓库中来。一个高级主管人员到数据仓库中索取需要的信息，就像一个旅行者去参观一个有趣的、有用的地方。假设这个管理人员具有广泛的行业知识，知道数据仓库全部的信息内容，但是他没有时间来详细浏览数据仓库的信息。每个主管人员都有一个特别的键码指示器，这好比是旅行者有特殊的地方要参观。如果这个主管人员查看键码指示器并发现了有趣的东西，他（或她）就会花更多的时间来深入地探察数据仓库的数据。旅行者对要参观的每个场所事先都有自己的一些期望，因此如果某个场所不符合他们的这些期望，旅行者就会想要探究发生这种情况的的原因。同样地，如果这个主管人员发现指示器不符合他的期望，那么他就有必要更加深入地调查这种现象。

因此，让我们总结一下划分为‘旅行者’的用户类型需要数据仓库中的什么内容：

- 每隔一定时间那个指示器的状态
- 没有任何困难地识别出用户感兴趣的项目的能力
- 能够轻松地选择所需要的东西，而无需在漫长的导航中浪费时间
- 从感兴趣的指示器很快到转移到另一个指示器的能力
- 无论什么地方有需要，用户都能轻松地得到和选中的键码指示器有关的附加信息，以便深入地探察这个指示器

操作者_以前我们曾经讨论过划分为‘操作者’的用户类型的一些特征。这一类型的用户对数据仓库感兴趣，主要是因为他们发现数据仓库是个信息的综合来源，那里不仅有各种历史数据，而且有他们感兴趣的各种详细的当前数据。‘操作者’实际监控着他们部门当前的表现情况。各个部门经理，生产线管理员和地区主管都可以划分为‘操作者’这一类型。

‘操作者’只对当前的性能和问题感兴趣，而对历史的数据一点也不感兴趣。作为联机事物处理（OLTP）系统的一个用户，‘操作者’希望能够缩短数据仓库反应的时间，更快地访问细节数据。那么‘操作者’该如何解决分布式系统产品中现有的瓶颈呢？可以替换当前

的出货方式的方法是什么，而且哪一个工业仓库的存货量比较低呢？操作者关心这样的问题，因为它们和当前的形势有关。因为数据仓库能够得到并储存从不同的源系统抽取来的各种数据，所以‘操作者’希望能在数据仓库中找到解决他们问题的答案。

下面总结了‘操作者’所需的内容。

●基于可靠的当前数据的直接答案

●性能度量的当前状态

●获取每天尽可能新的当前数据，或者更频繁地从源系统得到更新的当前数据

●对详细信息的快速访问

●对大部分的当前数据的快速分析

●简单的、直接的信息接口

农夫_如果说数据仓库用户中的一些用户和农夫有一些共同点，那么这些共同点是什么呢？考虑一下农夫的特性。农夫一般都对地形非常熟悉，而且他们准确地知道能从农作物上得到什么，因此他们的需求是一致的。农夫知道如何使用工具，如何在地里干活并且得到收获，而且他们也了解他们的农作物的价值。现在我们要用上述的这些特征来匹配划分为‘农夫’类型的数据仓库用户。

一般意义上，可以将一个企业中的不同类型的分析员划分为‘农夫’。这些数据仓库用户可以是各种技术分析员或者在市场、销售和金融方面的一些分析员。这些用户可能需要评估一些产品的利润率或者分析某种产品每月的销售额。因为他们的需要是几乎不变的，所以这些需要是可预测的也是常规的。

让我们总结一下划分为‘农夫’的用户类型的需要。

- 适当地综合从源系统抽取来的质量数据
- 能够容易地快速地运行可预测的查询程序
- 能够运行常规报表并产生标准的结果类型
- 在可预测的时间间隔内获得相同类型的信息
- 能够生成准确的较小的结果集

- 大部分是带有和历史数据的简单比较的当前数据

勘探者_这种类型的用户和普通类型的常规用户不同。‘勘探者’没有寻找信息的固定方法，他们倾向于到风险很少的地方去活动。‘勘探者’经常将不可预测的调查和随机探测结合起来。虽然有很多次调查都不会产生任何结果，但是有少数几次他们可能会挖出有用的模式和特殊的结果，获得小块的信息，这些信息的珍贵简直可以和固体的金子相比。因此‘勘探者’将使用非标准的过程和非正统的手段，继续他（她）的搜寻活动。

在一个企业中，研究员和熟练的技术分析员可以被划分为‘勘探者’。这些用户随机地使用数据仓库，而且使用的频率是不可预测的。为了密集地勘探数据他们可以连续几天一直使用数据仓库，接着在随后的几个月内都不使用它。实质上其它类型的用户是不会了解勘探者分析数据的方法的，因为勘探者运行的查询程序往往和大量的数据包有关。这些‘勘探者’用户利用大量的细节数据进行工作，以辨认出希望发现的模式。虽然工作产生的结果是比较难懂的，但是‘勘探者’会继续工作直到他们找到了希望发现的模式和关系。

和其它情况一样，现在我们要总结一下划分为‘勘探者’的用户类型的需要。

- 完全不可预测的和密集的特别查询
- 能够获得大容量的细节数据用来进行分析活动
- 能够执行复杂的分析过程
- 能够提供无结构的、全新的和创新的查询和分析
- 在紧急情况下的漫长的、延长的分析会话活动

矿工_挖金矿的人挖矿是为了发现有巨大价值的天然金块，而划分为‘矿工’类型的用户也以类似的方式来工作。在我们深入探讨矿工的特征和需要之前，我们要先拿矿工和勘探者相比，因为这两种用户都和大量的分析活动有关。有专家指出，‘勘探者’的作用是创造或者提出某种假设，而‘矿工’的作用却是证实或者反驳这种假设。从这个比较中我们可以看出矿工的一些作用。然而，‘矿工’是为了在数据中发现新的，未知的和未被怀疑的模式而进行工作。

‘矿工’是一个特殊的类型，因为在一个企业中他们是经过高度特殊训练和具有技能的特殊用途的一些分析员。许多企业里面没有可以划分为‘矿工’的数据仓库用户，它们雇用外面的顾问来做专门的数据挖掘项目。数据挖掘采用各种技术来执行专门的分析，而这些分

析能发现一些相关记录的聚合，未知变量值的估计和可能被顾客一起买走的产品的组合等等。

现在列出一些划分为‘矿工’类型的用户的需要：

- 通过对大量数据的访问来进行分析和挖掘活动
- 能够获得大容量的很多年之前的历史数据
- 能够从数据仓库中抽取数据并把这些数据以适合专门的挖掘技术的格式储存
- 能在两种模式下工作：一种是证实或反驳已提出的假设，另一种是在事先没有任何概念的条件下，发现假设

怎样为用户提供信息

上面所有的这些有关‘旅行者’、‘操作者’、‘农夫’、‘勘探者’和‘矿工’的讨论的意义是什么呢？我们讨论它们的目的是什么呢？作为数据仓库项目小组的一部分，你的目标是为每个数据仓库用户提供他们确实需要的东西，因此信息传送系统必须广泛地、适当地满足用户群体的完全需要。那么公司里的主管人员和经理需要什么技术和工具呢？公司里的商业分析员又该如何寻找他们需要的信息呢？公司里负责更深的、更密集的分析的技术分析员和负责监控日复一日的当前操作的知识工人又是一种什么样的情况呢？他们将怎样和数据仓库进行交互活动呢？

为了向用户提供最好的信息传送系统，你必须找到这些问题的答案。但是该如何寻找这些答案呢？你是不是要跑到每个用户哪里，确定他或她计划怎样使用你的数据仓库？接着你是不是要汇总所有的用户需求，然后提出信息传送系统的整体方案呢？这显然不是一个可行的方法。这也就是我们提出要划分用户类型的理由，因为如果你能够为所有这些类型的用户提供令他们满意的数据仓库，那么你的数据仓库就几乎已经覆盖了你的所有的用户群体。在你的企业里可能没有数据挖掘员，如果是这样的话，目前你就可以不用满足该组的需要。

我们已经回顾了每个用户类型的特征，也已经研究了每个用户类型的需要。我们研究用户类型的需要不是根据指定的信息内容，而是根据每个种类将怎样和以什么方法和数据仓库进行交互。现在让我们把注意力转移到一个最重要的问题上：怎样为数据仓库用户提供信

息？

请仔细地研究一下表 14-6，它描述了为上述五种用户类型提供信息的三个方面。‘结构含意’列出了和诸如元数据和用户信息接口等组件有关的要求，这些要求都是广泛的结构上的需要。表格为每个用户类型显示了对该类型用户最有用的工具的类型。当你要选择提供商和工具时，这个表格可以为你提供一些帮助。在表格中列出的“其它的考虑事项”包括数据仓库的设计问题、特殊的技术和任何不寻常的工艺要求。

信息传送机制

到目前为止从我们所有的讨论中，你会逐渐发现有四种根本的方法可以用来传送信息。你可能正在满足任何一种用户的需要，也可能正在构造一个信息传送系统以满足普通用户或超级用户的简单需要，但是在这些情况下信息传送机制的主要方法仍然是相同的。

第一种方法是通过报表来传送信息。当然，报表的格式和内容可能是深奥难懂的，但是这些本质上还是报表。通过报表来传送信息的方法是延续操作型系统的做法。你已经熟悉了成千上百的分布于操作型系统的各种报表。下一种方法也是一个来自操作型系统的不朽的技术。操作型系统允许用户在一个受约束的设置中进行查询操作，但是在一个数据仓库中，查询处理是一个最普通的信息传送方法，而且查询的类型从简单的到非常复杂的都有。正如你所知道的，操作型系统中的查询和数据仓库中的查询之间最主要的不同是数据仓库环境具有额外的能力和开放性，而操作型系统没有。

在数据仓库环境中，交互式分析的方法是特殊的，因为操作型系统很少为用户提供这样的交互式分析方法。最后，数据仓库是为下游的决策支持型应用系统提供综合数据的数据源。高级管理人员信息系统（EIS）是这些应用系统中的一个，但是更多的诸如数据挖掘等专门的应用系统，使人们有必要开发数据仓库。图 14-7 显示了在数据仓库和操作系统之间的信息传送方法之间的比较。

本节剩下的部分将致力于有关这四种方法的专门讨论。在设计这些信息传送机制的方法时，我们将突出报表和查询环境的一些基本特性，并提出了一些值得重视的细节。

查询

在一个数据仓库中查询管理对信息传送机制来说非常重要的，因为大部分的信息传送机

制是通过查询来实现的。数据仓库用户必须非常小心地管理这个完整的查询过程。现在首先考虑一个查询管理环境的特性：

- 在客户机上向用户提供查询的初始化、公式表示和结果显示等这些功能。
- 由元数据来引导查询过程。
- 有必要让用户能够轻松地通过数据结构来导航。
- 信息是用户自己主动索取的，而不是数据仓库强加给他们的。
- 查询环境必须要灵活地适应不同类型的用户。

让我们看看查询处理的过程。实质上这个过程可以分为三部分。第一部分涉及需要查询管理工具的用户，第二部分是关于查询本身的类型的，而在最后一部分，你已经拥有了位于数据仓库知识库中的数据，这些数据是用来进行查询的。图 14-8 显示了具有三个部分的查询处理过程。请注意每一部分的特性，而且当你建立查询管理环境时，你要重视这些特性并为你的查询管理环境提供这些特性。

现在让我们重点突出一些将在查询管理环境中用到的重要服务。

查询定义_确保数据仓库用户能够容易地将商业需要转换成适当的查询语法。

查询简化_要让用户能够了解数据和查询公式的复杂度，并能够简单地查看数据的结构和属性，而且用户也能容易地使用表格和结构的结合规则。

查询重建_甚至是看起来简单的查询也能导致密集的数据恢复操作，因此要使用户能够对输入的查询进行分解并重新塑造这些查询使其能更有效地工作。

导航的简单性_用户能够使用元数据在数据仓库中浏览数据，并能容易地用行业术语而不是技术术语来导航。

查询执行_使用户能够在没有任何 IT 人员的帮助下提交查询并执行这些查询。

结果显示_能够以各种方法显示查询到的结果。

对聚合的了解_查询处理机制必须要知道聚合的事实表，并且在必要的时候能够将查询重新定向到聚合的表格上以便更快地恢复查询。

查询管理_监控查询，并在失控的查询导致数据仓库的操作崩溃之前阻止它们。

报表

在这个子节中我们将考察报表环境中的一些有意义的特性。因为每个人都熟悉报表和它们的使用方式，所以我们就不重复大家都已经知道的东西，而是将报表和数据仓库联系在一起来讨论报表服务。在报表管理环境的全面定义方面上你都知道些什么呢？看看以下一些简短的条目：

●在报表中信息是数据仓库主动发给用户的，而不是象查询那样由用户主动地有选择地索取信息。报表被公布后，用户可以预订他或她需要的内容。

●和查询相比，报表是不灵活的、预先定义好的。

●大部分的报表是预先生成的，因此它们的格式是固定的。

●与用户主动生成的查询相比，用户对接收到的报表的控制更少。

●必须建立一个适当的分布式系统。

●报表通常是在服务器上生成的。

当你正在为数据仓库建构报表环境的时候，你可以使用下面列出的条目作为行动的方针：

预格式化的报表集_在预格式化的报表的仓库里要提供这些报表清晰的描述说明，因为这将使用户能够容易地浏览报表库的报表并选择他们需要的报表。

参数驱动的预定义报表_与预格式化的报表相比，参数驱动的预定义报表给了用户更多的灵活性。但是用户必须有能力来设置他们自己的参数，请求页面的暂停和进行部分统计等。

易于使用的报表环境_当用户除了预格式化报表或预定义报表外还需要新的报表时，他们必须能够轻松地利用简单的报表撰写工具来开发他们自己的报表。

在服务器的运行_在服务器上运行报表，以便客户端能够使用其它的信息传送模式。

报表时间安排_用户必须能在一个具体时刻或利用一个预先设计的事件来确定他们报表的运行时间。

公布和预订_数据仓库必须设置选项让用户来公布他们自己创建的报表，并允许其它用户预订这些报表或者接收它们的拷贝。

传送选项_提供各种选项诸如群发、电子邮件、网页和自动传真等让用户来传送报表，允许用户选择他们自己的方法来接收报表。

多数据操作选项_用户可以请求获得计算度量，通过交换行和列变量来实现结果的旋转，在结果中增加小计和最后的总计以及改变结果的排列顺序等操作。

多表达选项_提供了多种类型的选项，包括图表、表格、柱形格式、字体、风格、大小和地图等。

报表环境的管理_确保能够对进度表、监视器和待解决的问题进行简单管理。

分析

谁会是对分析很感兴趣的用户呢？这些用户是商业决策者，市场研究员，产品计划者，生产分析员－简而言之，就是所有的划分为‘勘探者’的用户。因为数据仓库具有丰富的历史数据等内容，所以它非常适合用来进行分析活动。数据仓库为这些用户提供了用来搜索各种商业趋势，寻找事务之间的相关性和辨别商业模式等操作的各种方法。

从某种意义上说，一个分析会话只是一系列相关查询之间的会话。数据仓库用户可以从一个初始的查询开始：今年第一季度每个产品线的销售额是多少？用户查看了一下数据，对其中两条生产线的销售额下降了这一情况感到好奇。用户接着继续下钻这两条生产线的各个产品的情况，因此下一个查询就是根据地区和州的来分析销售额下降的原因。用户接着比较两个以前的年份第一季度的销售额。在分析中没有任何固定的预定义的分析思路，用户可以从不同角度不同的先后顺序来进行分析，所以查询生成和执行的速度是和用户的思想保持一致的。

以前我们曾经涉及到查询处理的话题。任何查询都可以应用到作为分析会话的一部分来执行的查询中去。其中一个有意义的区别是分析会话中的每个查询被链接到前一个查询上，这样分析会话中的查询就形成了一个链接起来的系列。可见，分析过程是一个交互式的应用。

分析过程可以是极端的复杂，这取决于‘勘探者’追求的东西是什么。‘勘探者’可以在一个错综复杂的导航路径中分开采取几个步骤，而每个步骤可能需要大量的数据。然而这些步骤的联合可能要受到几个约束。‘勘探者’可能想要结果以多种不同的格式显示出来，以便他能够查看结果并领会结果所蕴涵的意思。这种复杂分析过程就是属于联机分析处理(OLAP)的领域了。下一章全部是用来介绍 OLAP 的，在那里我们将详细地讨论复杂的分析

过程。

应用系统

和数据仓库有关的决策支持系统是一些下游的系统，它们从数据仓库源头中获取数据。除了让用户可以直接访问数据仓库的数据内容外，一些公司还为一个特殊的用户组做了专门的应用系统。这是因为一些用户可能不太习惯浏览数据仓库并寻找特定的信息，所以如果将这些用户需要的数据在周期性的时间间隔内从数据仓库抽取出来，而且使用这些抽取到的数据来建立专门的应用系统，那么这就满足了这些用户的需要。

下游的应用系统和利用直接从操作型系统抽取的数据来驱动的应用系统有什么区别呢？建立一个依靠数据仓库来获取数据的应用系统有一个主要的优势，那就是数据仓库中的数据已经是合并的、综合的、转换过的和净化过的，而使用不同的操作型系统直接建立的任何决策支持应用系统可能不允许企业查看决策支持系统内的数据。

一个下游的决策支持应用系统开始建立时可以只是一个预格式化和预定义的报表集。你可以在系统内增加一个简单的菜单以让用户选择和运行报表，这样你就有了一个可能会对一些用户非常有用的应用系统。高级管理人员信息系统(EIS)是一个优秀的下游应用系统。利用数据仓库的数据建立的 EIS 被证实比十几年前的 EIS 更加高级，而那时的 EIS 是基于操作系统的数据。

最新的一项技术发展是数据挖掘，它是从数据仓库获取数据的一个主要的应用类型。随着越来越多的供货商在市场上提供支持数据挖掘技术的产品，这种应用变得越来越流行。数据挖掘是对知识发现进行处理。请参考 17 章，它阐述了数据挖掘的基础。

信息传送工具

正如我们以前指出的，你的数据仓库的成功取决于数据仓库中的信息传送工具的优劣。如果这个工具是有效的、有用的和诱人的，那么数据仓库用户就会经常使用数据仓库，因此你必须非常仔细地选择信息传送工具。下面我们将详细地讨论这个非常重要的事情。

信息传送工具能够以不同的方式来满足用户的各种需要，它的主要类型包括查询工具和数据访问工具等。这种类型的工具使用户能够定义查询，生成并执行查询从而获得需要的结果。其它类型的工具包括报表撰写器和用来格式化报表，安排时间表和运行报表的一些报表

工具等。另外有一些工具结合了不同的特性以便数据仓库用户可以使用单一的工具来查询和运行报表。更一般的，你将会发现很多能在单一的数据仓库环境中使用的信息传送工具。

信息传送工具一般要执行两种功能：它们把用户希望查询或运行报表的请求转换成SQL语句，送到数据库管理系统（DBMS）中；它们接收数据仓库 DBMS 返回的结果并在合适的输出上形成结果集显示给用户。通常，向 DBMS 发送的请求能够获得大容量的数据并对这些数据进行操作，因此和请求能够获得的数据量相比，结果集中包含的数据量就少多了。

桌上型电脑环境

在客户端－服务器的计算体系结构中，信息传送工具是在桌上型电脑环境中运行的。用户是在客户端计算机上发出他们的请求。当用户要选择某种查询工具作为信息传送的组件时，其实他（她）是在选择一种在客户端工作站上运行的软件。那么信息传送工具的基本类型是什么呢？将你要选择的工具划分到基本类型这一组，这有助于你更好地理解你可得到什么类型的工具和你实际需要什么类型的工具。

现在我们来考察这些信息传送工具，你可能需要从中选出一些来使用。请仔细地研究表14-9，它列出了能用在桌上型电脑环境中的信息传送工具的主要种类，并且概括了每个种类的使用方法和实现目标。注意每个种类的实现目标。每种工具的使用方法和功能有助于你为不同类型的用户找到适合他们使用的工具类型。

工具选择的方法学

因为信息传送工具在数据仓库环境中的特殊重要性，你必须谨慎地思考并形成一种方法学，用来选择你需要的适当工具。某个提供商提供的一套工具对一个特定环境来说可能是最好的，但同样的这套工具在另一个数据仓库环境中可能会是个完全的灾难。所以在工具的选择上没有一种类型的工具能够适应所有环境的这种说法。能够适应你的环境的工具也必须能够适合你的用户，而且必须最适合你的用户。因此在形成选择工具的方法学之前，请重新审议你的用户的要求。

你的数据仓库用户是谁？他们在什么组织级别上运行数据仓库？他们的工作能力怎么样?他们希望和数据仓库如何进行交互？他们对数据仓库的期待是什么？在用户的公司里有多少‘旅行者’？有任何‘勘探者’吗？你可以提出所有中肯的问题，并研究这些问题的答

案。

在设计和开发数据仓库的最好的实践中，一个正规工具的选择方法学的地位是很高的。一个好的工具选择方法学当然要包含你的用户代表，让他们成为设计开发过程的一部分，否则该方法学注定是要失败的。要让你的用户积极地参予到设置工具的标准和对选出的工具进行评估等活动中来。除了要考虑用户的喜好外，也要注意和其它数据仓库组件的技术兼容性，不能忽视技术方面的问题。

一个好的正规的方法学已经发展为一种分阶段的方法。将工具选择过程划分为几个定义好的步骤，接着说明每个步骤的目的和活动内容，估计完成每个步骤所需的时间。然后从一个步骤进行到下一个步骤，而每个步骤的活动开始的前提是要成功地完成前一个步骤的活动。图 14-10 显示了选择信息传送工具过程中的几个步骤。

用来选择适合你的环境的工具的正规的方法学，必须定义这个选择过程中的几个步骤的活动。下面是对选择过程中应该进行什么类型的活动的建议，你可以用它们来指导你的选择过程。

形成工具选择小组_在小组中包含大约四到五个人。因为信息传送工具是重要的，所以要确保工具选择的负责人是小组中的某个人。因为主要课题领域的用户代表能够提供用户观点，并起到课题专家的作用，所以一定要把他们吸收到该小组中来。必须要使小组中的某个人熟悉信息传送工具。如果数据仓库管理员在信息传送领域是有经验的，那么就可以让他领导该小组并推动整个工具选择的过程。

重新评价用户的要求_必须从信息传送的角度而不是用普通的方法来重新考虑一下用户的要求。列出用户的种类，把每个潜在的用户放在适当的类中，。描述每个种类的期望和需要，并将这些要求写成文档以便你能协调用户的要求和潜在工具的特性之间的关系。

定义选择的标准_对每个较大的工具组中的工具，例如查询工具或报表工具等，要详细地说明该工具的标准。可以参考一下‘工具选择标准’这一后面的子节。

研究可用的工具和提供商_在这个阶段你可能要花较长的时间，所以在该阶段最好能有个领先。你可以从提供商那里获得需要的产品文献。商业展览也有助于你得到潜在工具的第一印象。数据仓库协会是另外一个好的来源。在市场上有几百种信息传送工具，所以你必须将选择名单缩小到 25 个左右或者更少以便进行初步的研究。在这阶段，要把精力集中在名单上的工具的功能和特性。

准备一个要考虑的清单_这个阶段跟在研究阶段后面。你的研究将产生一个较长的潜在工具的初步清单。用文档记录这个初步清单上每个工具的功能和特性，同时也要注意这些功能和特性怎样和用户的要求相匹配。

获得额外的信息_在这阶段，你想要对初步清单上的工具做额外的、更密集的研究。你可以和提供商交谈，接触提供商为你建议的信息传送工具并把它作为一种参考。

选择前三个工具_选择初步清单上你最满意的三个工具作为可能的候选工具。如果你对最后的选择结果不是很有把握，可以选择更多的工具，但最好不要超过五个，因为如果你一个有更长的清单，它将会花掉你更长的时间来完成剩下的选择过程。

注意产品演示_现在你想要尽可能多地了解这个短清单上的工具。可以打电话给提供商向他们索取产品演示。如果你的环境配置恰好不适合选中的工具，你可以到提供商那里提出一些问题。在演示过程中，要保证工具的功能能够匹配用户的需要。

由 IT 完成评估_让 IT 对选中的工具与你的环境之间的兼容性做一个独立的评估。要测试工具的一些特性，例如该工具和你的数据库管理系统（DBMS）的连通性，并检验工具的可测量性。

由用户完成评估_这是一个关键的阶段。因为用户测试这个工具和能否接受它是非常重要的，所以不要跳过这个阶段。这阶段由一些足够数量的传送会话构成。如果能够在实际的使用中原型化，那么尽可能这么做，因为如果有两个产品在同等程度上匹配用户的需要，在这种情况下原型能够显示这两个产品本质的区别。

做出最后的选择_在这个阶段你将准备进行最后的选择，这也给了你重新评估这些信息传送工具的机会，这些工具非常接近用户的需要。这个阶段也同样要检验提供商。这些工具可能是优秀的，但如果当前的提供商是从另一家公司获得这些工具的，那么这些工具的技术支持可能会不足。或者提供商可能是不稳定的。因此必须从提供商那里核实所有的这些相关问题，然后做出最后的选择。记住，整个过程中都要有用户的参予。

你可能已经意识到，选择工具的过程将是紧张的并且它要花费相当长的时间，然而你一定不能轻视它。在截然不同的阶段之间进行活动而且从头到尾地让用户参予整个过程，这样做是明智的。

让我们用下面的一些实践技巧来结束这个子节：

- 让小组中的一个有经验的成员或者数据仓库管理员来领导工具选择小组并推动整个过程的发展。
- 让你的用户完整地参予该过程。
- 没有东西可以替代 IT 人员的评估，而且不要只满足于提供商的产品演示，你得自己试试这些工具的性能。
- 可以考虑从一些典型的信息传送交互机制中抽取原型，也要注意工具能否承受住多个用户的传送负载。
- 要把多个不同提供商的工具结合起来使用，这不是一件容易的事情。
- 要记住，信息传送工具必须能够和数据仓库的数据库管理系统（DBMS）兼容。
- 继续在最前沿考虑元数据。

选择工具的标准

再讨论了每类用户的需要之后，你一定已经理解了选择信息传送工具的标准。例如，我们讨论‘勘探者’和他们的需要时用到了复杂的分析。这告诉我们提供给‘勘探者’的工具必须得有适合复杂分析的特性。而对‘旅行者’来说，他们要用的工具必须是简单直观的。到现在为止，你已经深入地了解了选择信息传送工具的标准。在信息传送的三个主要领域，也就是报表、查询和分析上，你已经掌握了如何来选择适用这些方面的工具。

现在让我们把上述的想法放在一起，以提供一个关于选择信息传送工具的一般标准的清单。这个清单适用于上面提到的三个主要领域。你可以把这清单当作指南来使用，或者你自己也可以准备一个能很好地适应你的环境的清单。

易用性 这大概是能使你的用户感到满意的一个最重要的方法。在创建查询、建立报表和灵活地表示结果等方面上尤其需要易用性。

性能 虽然在数据仓库环境中系统执行和反应的时间不像它在联机事务处理(OLTP)系统中那么重要，但是它们在用户需要的清单中仍然排在较高的位置。不但信息传送系统而且整个环境都需要一个可接受的性能范围。

兼容性 信息传送工具的特性必须完全适合用户的需要。例如，联机分析处理(OLAP)和‘旅行者’这种类型的用户不兼容，同样的，预格式化的报表也不能准确地满足‘勘探者’

的要求。

功能性 这是兼容性的一个扩展。每个用户类型的属性要求工具中必须具有某项必不可少的功能。例如，‘矿工’就需要一个从捕获数据到发现未知模式的全部范围的功能。

综合 更一般的，可以在一个用户会话中混合进行查询、分析和生成报表等操作。用户可以从一个初始的查询开始，接着查询的结果将导致用户进行深入的查询或其它形式的分析等操作。在会话的结尾，用户可能获得以报表形式显示的最终的结果集。在你的环境中如果这种类型的使用是普遍的，那么你的信息传送工具必须能够综合具有这些不同的功能。

工具管理 集中的管理会使信息传送管理员的任务变得更轻松，所以信息传送工具必须要具有配置和控制信息传送环境的功能。

支持 Web 因特网已经成为我们通向世界的窗口。当今的数据仓库和在 Web 技术流行之前建立的数据仓库相比有一个很大的优势，就是它能够和 Web 进行交互用户。对信息传送工具来说，能够通过因特网或者你的公司的内部互联网来发布 Web 页面是很重要的。

数据安全 在大多数的企业里，保护数据仓库的数据和为操作型系统的数据提供安全保障是同样关键的。如果你的环境中有敏感的数据，那么你的信息传送工具必须具有安全的特性。

数据浏览能力 用户必须能够浏览元数据和查看这些数据的定义和意义。工具必须能够象图形用户界面(GUI)组件那样在屏幕上显示数据集，让用户通过点击图标来选择要查看的内容。

数据选择器能力 信息传送工具必须使用户无需使用技术上的术语和方法来执行表格的联合操作，就能够建立查询。

和数据库的连接 连接到任何主流的数据库产品的能力是信息传送工具的一个必不可少的特性。

显示特性 对信息传送工具来说，让结果集能够以各种不同的格式显示出来是很重要的，这些格式包括文本，列表格式，图表，图像和地图等。

可升级性 如果你的数据仓库是成功的，那么你就能确保在短时间内伴随着信息请求的复杂度的显著扩展，用户数量也会有持续的增长。为了处理更大容量的、具有额外的复杂度的请求，信息传送工具必须是可以升级的。

提供商的可靠性 随着数据仓库市场的成熟，你将发现有很多公司被合并或被兼并了，一些公司也可能会申请破产。因此即使你选择的工具可能是最适合你的环境的，但如果工具的提供商是不稳定的，那么恐怕你得重新考虑选择其它的信息传送工具。

信息传送框架

现在到了我们概括在本章中讨论过的所有东西的时候。我们对数据仓库用户进行分类并提出了标准的用户类型。你可以把每组用户都归入到这些用户类型中的某一种。在数据仓库的信息传送机制方面，每个用户类型都有具体的特征。对用户的分类帮助我们了解了每个用户类型的信息需要。一旦你明白了每个类型的用户需要什么，你就能够为这些类型的用户提供他们需要的信息。接着我们讨论了信息传送的标准方法和如何来选择信息传送工具。现在请参考表 14-11，它概括本章我们所有的讨论。该表显示了各种用户类型，指出了如何把企业中的各种用户划分到相应的用户类型中，而且表格也将每种类型的用户和适合这个用户类型的通用工具的种类相匹配。总之，这个表格包含了每样东西。

本章总结

●数据仓库中的信息内容和使用方法和操作型系统中的信息内容和使用方法有很大的不同。

●数据仓库具有庞大的信息潜力，这些信息潜力能够在全面的企业管理和不同的商业领域中得到应用。

●你可以通过一个有效的用户－信息接口来实现数据仓库的信息潜力。

●谁会使用数据仓库中的信息？要理解各种用户和他们的需要。各种数据仓库用户可以被划分到一个有趣的组中：‘旅行者’、‘操作者’、‘农夫’、‘勘探者’和‘矿工’。

●根据每个用户类型的各种需要、技能和行业背景，为该类型的用户提供他们需要的信息。

●查询、报表、分析和应用程序形成了信息传送机制的基础。每个用户类型都需要这些

信息传送方法。

●你的数据仓库的成功依赖于终端用户信息传送工具的有效性，所以要根据可靠的工具选择标准仔细地挑选需要的信息传送工具。

思考题

1. 数据仓库和操作型系统在使用方法和价值上有什么不同？
2. 简要地解释数据仓库的信息如何来促进客户关系的管理？
3. 数据仓库中的信息使用方法的两个基本模式是什么?为每个模式举个例子。
4. 列出用户－信息接口的任意五个必不可少的特性。
5. 什么是超级用户？超级用户希望如何来使用数据仓库？
6. 划分为‘农夫’类型的用户是什么人？说出这种类型的数据仓库用户的任意三个特征。
7. 列出查询管理环境的任意四个必不可少的特征。
8. 列出报表管理环境的任意四个必不可少的特征。
9. 列出信息传送工具的五个标准，并解释为什么你要根据这些标准来为你的数据仓库选择信息传送工具。
10. 用不超过四句话来描述你对信息传送框架的理解。

复习题

1. 匹配以下各列：

1．信息发现模式	A．EIS
2．数据仓库‘旅行者’	B．需要直观的界面
3．改写查询	C．计划，执行，评估

4．数据仓库‘勘探者’	D．确认假设
5．下游的应用系统	E．信息“拖”技术
6．全面的企业管理	F．分析大的数据容量
7．确认模式	G．需要指示器的状态
8．临时用户	H．高度随机的访问
9．数据仓库‘矿工’	I．分解和改善查询
10．查询	J．数据挖掘

2．比较数据仓库中的信息和操作型系统中的信息的使用和价值。解释它们之间的主要的区别，进行讨论并举一些例子。

3．考察你的数据仓库的潜在用户。你能把这些用户归入到临时用户、常规用户或超级用户中的某一种吗？如果你觉得这种简单的划分用户的方法是不足的，那么讨论一下你的数据仓库用户应该怎样进行分类。

4．在你的数据仓库的潜在用户中，你能把哪些用户划分为‘旅行者’呢？这些‘旅行者’的特征是什么？你怎样为你的组织中的这些‘旅行者’提供他们需要的信息呢？

5．你怎样理解数据仓库环境下的信息传送框架？作为一个终端用户信息传送专家，讨论一下你将如何为一个卫生维护组织（HMO）建立这样的一个信息传送框架？

第十五章　数据仓库中的联机分析处理(OLAP)

本章目标

●发现对联机分析处理不适当的要求并理解是什么导致了这种要求

●详细地回顾一下 OLAP 的主要特性和功能

●掌握维度分析（dimensional analysis）中的错综复杂的事物，学习超立方体、下钻和概括化和多层次/多视角查看的含意

●考察不同的 OLAP 模型并且确定哪个模型更适合你的环境

●通过研究 OLAP 的步骤和工具来考虑 OLAP 执行的情况

在前面的章节中我们提到了联机分析处理（OLAP）。当我们讨论信息传送方法时，曾提到过 OLAP。那么你已经对什么是 OLAP 和它怎样被用来进行复杂分析等这些问题有了一定的了解。跟联机分析处理的字面意思一样，用 OLAP 来进行分析时，它跟数据的处理过程有关。数据仓库为分析操作提供了最好的机会，而 OLAP 就是用来执行相关分析的工具。在实现了分析之后，数据仓库也有利于数据的访问。

现在我们有机会来深入地探究 OLAP。在当前的数据仓库环境中，各种提供商的分析处理工具有了如此巨大的进步，因此你不能拥有一个不带 OLAP 的数据仓库，那是不可想象的。因此贯穿本章的始终，你要留心一些重要的话题。

首先，你必须了解 OLAP 是什么和为什么它是绝对不可缺少的，因为这将有助于你更好地理解 OLAP 的特性和功能。接着我们将讨论 OLAP 的主要特性和功能以便你能牢固地掌握它。OLAP 有两个主要的模型，而你应该知道哪个模型更适合你的计算活动和用户环境。然后我们将突出每个模型的意义，学习如何在数据仓库环境中执行 OLAP，并且研究 OLAP 工具，并发现如何评估和获得这些 OLAP 工具。最后我们将讨论 OLAP 的执行步骤。

联机分析处理的要求

回忆我们在第 2 章中关于自上向下和自下向上建立数据仓库的方法的讨论。在自上向下的方法中，你使用实体－关系（E-R）模型技术建立了全面的企业范围的数据存储库，它将为部门数据集市提供数据。在自下向上方法中，你使用多维模型技术建立了若干个数据集市，这些数据集市的集合形成了你的公司的数据仓库环境。这两种方法的任何一种都有它各自的优点和缺点。

你也学习了一个实际的方法，用来建立一种具有一致的、标准化的数据内容的超市的混合体。在采用这个方法时，首先你在企业级上计划和定义需求，然后为一个完整的数据仓库建立它的基础结构，接着按照‘超市’的优先级的高低一次实现一个“超市”。这些超市是用多维模型技术设计的。

正如我们所知道的，数据仓库意味着用户可以使用获得的数据来执行必要的分析。分析能够帮助用户制定战略性决策，这是建立数据仓库的首要的主要理由。为了让数据仓库用户能够执行有意义的分析，数据必须以某种方法被映射，以便该用户能利用这些数据来分析键码指示器（key indicator）随着时间变化各个商业维的值。用多维技术设计的数据结构支持这种类型的分析。

在上面提到的所有的三种方法中，数据集市依赖于多维模型。因此这些数据集市必须能够支持对维的分析。在实践中这些数据集市似乎足够用来进行基础分析，然而我们发现，在当前的商业条件下用户的需要已经超出了这种基础的分析。有些用户必须有能力在更少的时间内执行更多的分析。因此我们将来考察数据仓库提供的传统的分析方法有什么不足之处，从而了解为了保持竞争力和扩展商务活动用户真正需要什么样的分析方法。

对多维分析的需要

让我们迅速地看一个大型零售业务的商业模型。如果你只是关注日常的销售，你很快就明白了销售其实是和许多商业维（business dimension）相互关连的。日常的销售只有在它们和销售日期、产品、分配渠道、商店、销售区域、宣传和更多的维度有关时才有意义。多维视图天生就适合用来表示任意的商业模型。很少有人限制某个模型只能有三个维或者更少的维。为了制定计划或战略性决策，经理或主管人员可以通过方案来探索商业数据。例如，他

们将实际销售额和预期的销售额相比，或者和前段时期的销售额相比。他们根据产品、商店、销售地域和营销等来考察销售额下降的原因。

决策制定者不再满足于诸如“我们在新泽西州爱迪逊的商店卖出了多少单位的 A 产品？”等这样的一维查询。考虑以下更多的有用的查询：新产品 X 在最近的三个月取得了多少收入，该产品在中南区域的不同商店不同月份取得多少收入，营销的对销售作用如何，和预期相比情况如何，和产品的前一个版本相比情况如何？用户接着要求和类似的产品进行更深入的比较，或者要求对该产品在不同区域之间的情况进行更深入的比较，也可以要求通过在行和列之间旋转表示方法来查看比较的结果。

为了有效地进行分析，你的用户必须要有简单的方法来沿着商业维来执行复杂的分析。他们需要一个能够表示数据的多维视图的环境，而且该环境必须能够通过简单的、灵活的信息访问来为分析处理提供基础。决策制订者能够用多种方法来查看结果，他们必须能够沿着任何维度在任何聚集水平上对数据进行分析。他们必须有能力沿着每个维的不同层次下钻和概括化数据。如果你的数据仓库中没有一个固定的系统来进行实际的多维分析，那么这个数据仓库是不完善的。

在任何分析系统中，时间都是个关键的维度。大多数的查询在执行时，都要沿着一些维度进行分析，而时间都会是那些维中的一个。此外，时间是一个独特的维，因为它具有连续的本质－10 月之后总是 11 月。用户可以随着时间的过去来监控各种性能，例如，将这月的表现和上月的相比，或者将这月的表现和去年同期的相比。

其它的关于时间维的独特性的要点是这个维在各层次上的工作方式。用户可以查找三月份的销售额，也可以查找该年前四个月份的销售额。在查找前四个月份的销售额的第二个查询中，暗指的下一个更高级的层次是一个考虑到时间的连续本质的聚集。因为商店维没有隐含的顺序，所以没有用户会查找前四个商店或最后三个商店的销售额。因此实际的分析系统必须认识到时间的连续本质。

快速的访问和强大的计算

无论用户是请求查询所有地域内所有产品的每个月的销售额，还是请求查询某个区域内某个产品每年的销售额，查询和分析系统都必须对这些查询做出一致的反应，不能因为用户要进行复杂的分析而让这些用户忍受系统漫长的反应时间。用户用公式来表示一个查询所耗

费的精力和等待接收查询的结果集所花的时间都必须和这个查询的类型无关。

让我们举一个例子来理解分析处理的速度和用户有什么关系。想象一个商业分析员正在寻找最近几个月来整个企业的利润率急速下滑的原因。分析员开始进行分析，他查询了这个公司最近五个月来各个月份的全面的销售情况。他发现虽然销售额没有出现下滑的迹象，但是最近三个月的利润率却迅速地下降了。于是这位分析员继续深入地进行分析，这时他想要找出这个企业在哪个国家的利润率下降了。分析员向数据仓库请求显示世界范围内利润率出现下滑迹象的主要区域，因此他发现是欧洲地区导致了整个企业的利润率下降了。现在分析员感到线索越来越清晰了，接着他查询欧洲地区中利润率下降的国家。分析员发现，一些国家的利润率上升了，一些国家的利润率急剧下降了，而剩下的那些国家的利润率是稳定的。此时分析员在分析中引进了另一个维。现在他要查询欧洲的不同国家、不同月份和不同产品的利润率，这个步骤使分析员更接近于找到利润率下滑的原因。分析员观察到最近两个月该企业在欧盟国家市场的利润率出现了急剧下滑的迹象，于是他进行了更深入的查询，从而发现这个企业在这些欧盟国家的生产成本和其它直接成本都保持在一般水平，但间接成本却上升了。分析员现在终于能够确定，这个企业的利润率下滑是由对该企业在欧盟的一些产品采取的额外征税所造成的，而且他也能够确定迄今为止征税对该企业利润率的准确影响。因此这个企业以这个分析员的分析为基础，制定了怎样解决该企业利润率下滑的问题的战略性决策。

现在请看图 15-1，它显示了一个单一分析会话中的所有步骤。在这个会话中有多少步骤呢？有很多步骤，但只有一个单一分析会话和思想队列。这个思想队列中的每个步骤组成了一个查询。分析员用公式表示每个查询，执行这些查询并等待查询的结果集显示在屏幕上，接着他开始研究该结果集。这些查询中的每个查询都是相互作用的，因为由一个查询得到的结果集形成了下一个查询的基础。用户只有保持这个要素，才能以这样的查询方式维持这个思想队列。快速访问对一个有效的分析处理环境来所是绝对有必要的。

你是否注意到，这个分析会话中的任何查询都进行任何繁重的计算？这是不实际的。在一个现实世界中的分析会话中，很多查询都要求进行计算，有时甚至是复杂的计算。这句话的含意是什么呢？它的意思是一个有效的分析处理环境不但必须是快速的、灵活的，而且也能支持复杂的、强大的计算。

接下来列出了包含在查询请求里的各种典型的计算：

●进行概括化操作以沿着维度的各种层次来进行汇总和聚集。

●沿着各个维的各个层次从最高层向最低层下钻。

●简单的计算，例如利润的计算（销售额减去成本）等。

●部分占整体的百分比的共享计算。

●包含键码性能指示器的代数方程。

●不固定的平均值和增长的百分比。

●使用统计方法来分析事物发展的趋势。

其它分析方法的局限性

现在你已经相当好地掌握了用户执行查询和分析的要求的类型。首先的也是最重要的，信息传送系统必须能够显示数据的多维视图。其次信息传送系统必须使用户能够使用各种方法沿着多维和这些维的层次来分析他们需要的数据和使用这些数据。这个工具必须是快速的，必须使用户能够进行复杂的计算。

让我们来考察传统的工具和方法不能胜任复杂分析和计算等这些任务的原因。我们已经熟悉什么信息方法呢？当然，我们最早熟悉的是报表这个方法，接着又出现了具有完整的功能和特性的电子数据表。现在人们已经接受了 SQL 这个用来从关系数据库中获取数据并操作数据的接口。上述的这些方法是在联机事务处理（OLTP）系统和数据仓库环境中使用的。现在，这些传统的方法还能适应我们正在讨论的多维分析和复杂计算吗？

首先，让我们看看 OLTP 和数据仓库环境的特征。这里当我们提到数据仓库环境时，我们不考虑繁重的多维分析和复杂的计算等操作。我们只考虑具有简单的查询和常规的报表的这种环境。请看表 15-2，它显示了 OLTP 和基本的数据仓库环境中的与信息传送的需要有关的一些特征。

现在我们要考虑如何在这两个环境中进行信息获取和信息操作。信息传送的标准方法是什么呢？它们是报表、电子数据表和联机显示。标准的数据访问接口是什么呢？SQL。让我们回顾一下这些内容，并确定它们是否足够用来进行多维分析和复杂计算。

报表编辑器提供了两个关键功能：能够通过移动鼠标并点击选项来生成和发送 SQL 命令，并且能够以各种格式来输出报表。然而，报表编辑器不支持多维报表，而且用基本的报表编辑器，你无法下钻到维的较低的层次，因此你不得不借助额外的报表。在基本的报表编辑器中，你无法通过交换行和列来旋转结果，而且报表编辑器不提供对集聚的导航。所以一旦你生成了某个报表并且执行它，你就无法改变结果数据集的显示形式。

如果报表编辑器不是我们理想中的工具或方法，那么电子数据表的计算和分析等特性又如何呢？电子数据表这项产品一开始发布的时候，就被定位为一种分析工具。你可以用电子数据表来进行“如果…就…”这种方式的分析操作。当你修改电子数据表中的一些单元格的值时，其它相关的单元格的值也会自动地发生改变。电子数据表中的聚集和计算的特性又怎么样呢？电子数据表中的一些附加工具能够进行一些形式的聚集，也能进行各种计算。第三方工具也增强了电子数据表产品的能力，使它能够用三维格式来显示数据。你也可以在电子数据表中查看行、列和页的内容。例如在电子数据表中行可以显示产品信息，列可以显示商店信息，而页可以显示月份的值。现代的电子数据表工具又向用户提供枢轴式表格和 n 个方向的交叉表。电子数据表使用起来非常麻烦，甚至在它利用附加工具加强了自己的功能之后也是如此。让我们举个对商店、产品、宣传和时间这四个维度进行分析的例子。我们假定每个维平均只有五个层次。现在我们要建立一个分析，获取数据，并且像电子数据表那样显示所有的聚集和多维的视图。你可以想象这个练习要花费多少精力。现在如果你的用户想要改变数据的导航并进行不同的概括化和向下查询等操作，你该怎么办呢？从中我们可以看到用电子数据表来进行多维分析和复杂计算的局限性。

现在让我们将注意力转到结构化查询语言 SQL 上。虽然 SQL 的最初目标是成为终端用户的查询语言，但是现在几乎每个人甚至是老练的用户都觉得这种语言太深奥了。有一些第三方产品企图扩充 SQL 的能力并且向用户隐藏 SQL 语法，使用户可以通过 GUI（图形用户界面）移动鼠标并点击选项的方法和自然语言的语法来生成他们的查询。但是，SQL 的专业用语不适合用来分析数据和探索关系，甚至利用 SQL 来进行基本的比较操作都是困难的。

诸如市场调查和金融预测等一些有意义的分析一般都要获取大量的数据，进行计算和汇总各种数据。或许用户可以使用 SQL 来获取结果并且使用电子数据表来显示 SQL 获取的结果，利用这种方法甚至可以实现详细的分析。但这里有个不利的消息：在现实世界中的一个分析会话里面，很多查询都是一个紧跟着另一个。每个查询可以转换成一些复杂的 SQL 语句，每个 SQL 语句可能会包括浏览整张表格、多表联合、聚集、分组和排列等操作。我们现在讨论的这种类型的分析，需要进行复杂的计算和对时间序列数据的处理。SQL 在这些方面表现得特别差，所以即使你能想象某个分析员能够准确地形成这些复杂的 SQL 语句，系统的开销仍将是巨大的而且这也严重地影响了系统的反应时间。

联机分析处理（OLAP）是用户需要的答案

用户当然需要能够进行复杂计算的操作的多维分析，但是我们不幸地发现传统的分析工具，例如报表编辑器、查询产品、电子数据表和语言接口等，都有一些不足之处。那么答案是什么呢？显然，在 OLTP 和基本的数据仓库环境中使用的这些工具不能胜任这个任务，因此我们需要的是为繁重的分析量身定做的一些不同的工具和产品。这就是在数据仓库中我们需要 OLAP 工具的原因。

在这一章里，我们将彻底地分析 OLAP 的各个方面，我们将提出 OLAP 正式的定义和详细的特征，我们将突出 OLAP 所有的特性和功能，而且我们将浏览不同的 OLAP 模型。但是既然你已经初步了解了 OLAP，现在就让我们列出 OLAP 的一些基本的优点，以此来证明我们的立场是正确的。

●使分析员、主管人员和经理能够更深入地了解数据的显示方法。

●能够沿着几个维重新构造度量（metrics）并允许用户从不同的角度来查看数据。

●支持多维分析。

●能够在每个维度里面进行下钻和概括化的操作。

●能够对分析做出快速的反应。

●对其它信息传送技术，例如数据挖掘等，进行补充

●通过使用图像和表格来实现可视化的表现形式，使用户能够更容易地理解结果集的含意。

●能够在 Web 上运行。

●能够实现交互式分析。

在这个阶段，你将通过学习一个典型的 OLAP 会话（看表 15-3）来理解 OLAP 的本质和能力。分析员请求获得各个产品线的情况的高级别的汇总，从而开始了查询。接着，这个分析员下钻各个年度的详细数据。分析员接下来要做的是旋转数据并查看各个年度的总计，而不是查看各个产品线的总计。在这样的一个简单例子中，你就能感觉到 OLAP 的能力和特性。

OLAP 的定义和规则

OLAP 这个术语源自哪里？我们已经知道多维是 OLAP 系统的核心，而且我们也提到过 OLAP 的一些其它的基本特性。OLAP 是一些繁重的分析的复杂因素的集合吗？我们能否有一个正式的定义和一套基本的方针来鉴别 OLAP 系统?

在一篇题为《为分析员用户提供联机分析处理》的论文中介绍过 OLAP 或联机分析处理这个术语，文章的作者 Dr.E.F.Codd 是公认的关系数据库模型之父。发表于 1993 年的这篇论文为 OLAP 系统定义了 12 条规则。之后在 1995 年，作者又为 OLAP 系统附加了六条规则。这里我们将讨论上述的这些规则。在此之前，让我们为 OLAP 寻找一个既简短又准确的定义。这样的一个简洁的定义是来自于 OLAP 委员会，该委员会的作用是为 OLAP 用户提供成员资格，主办研讨会并推广 OLPA 的使用。下面是该委员会关于 OLAP 的定义：

联机分析处理（OLAP）是一种软件技术，它使分析员、经理和主管人员能够通过快速的、一致的和交互式的访问来获取并理解各种可能的信息视图的数据，这些信息由原始数据转换而成用来反映一个企业实际的维度。

OLAP 委员会的这个定义包含了 OLAP 所有的关键因素，速度、一致性、交互式访问和多维视图－所有的这些都是首要的因素。一家商业杂志在 1995 年描述说，OLAP 是一个用来进行多维分析的有趣的术语。

Dr.Codd 提出的那些原则成了用来衡量任何一套 OLAP 工具和产品的准绳。一个实际的 OLAP 系统必须符合这些原则。当你的项目小组正在寻找 OLAP 工具时，它可以把这些原则按照重要性来划分它们的优先级，从而选出能够符合你的高优先级的那些原则的工具。现在我们要考虑有关 OLAP 系统的最初的 12 条原则：

多维概念的视图_能够提供一个直观的、易于使用的多维的数据模型。因为本质上用户是用多维的角度来看一个企业的，所以一个多维的数据模型符合用户认识行业问题的角度。

透明度_让技术、潜在的数据存储库、计算体系结构和元数据的不同本质全部对用户透明。这种支持一个实际的开放系统的方法的透明度，有助于通过一些用户熟悉的前台工具来提高用户的工作效率和生产率。

可访问性_只允许用户访问在执行一个特定的分析过程中实际需要的那些数据，并向用户显示一个单一的、一致的和连贯的视图。OLAP 系统必须将它自己的逻辑视图映射到各种种类的物理数据存储器，并且对视图进行任何有必要的转换。

一致的报表性能_当维的数量或数据库的容量增加了的时候，要确保用户的报表性能不会大

大地降低。每次在运行一个特定的查询的时候，必须让用户感觉到一致的运行时间、反应时间或机器的使用性能。

客户端/服务器体系结构_为了使系统具有合适的性能、灵活性、适应性和协同工作的能力，就必须让系统遵守客户端/服务器的原则。同时也要使服务器组件拥有足够的智能，以便各种客户端只需花费最小的精力和装上最小的综合程序就能连接上服务器。

等价的维_确保每个数据维在结构和操作性能上都是等价的。要使一个逻辑结构能够适应所有的维，而且这个基本数据结构或访问技术禁止偏向任何一个数据维。

动态稀疏矩阵的处理_创建和加载一个特定的分析模型用来最优化稀疏矩阵的处理，并使物理视图适应这个模型。当系统遇到一个稀疏矩阵时，它必须能够动态地演绎出该矩阵的数据的分布，调整存储器的结构和通过访问来取得和维持一致的性能。

支持多用户_支持终端用户和其它相同的分析模型中并发地工作，也支持他们根据相同的数据来并发地创建不同的模型。简而言之，为用户提供并发的数据访问、数据完整性和访问安全。

无限制的跨维操作_让系统能够辨别维的层次和在一个维中或跨维自动执行下钻和概括化的操作。在忽略了每个数据单元的通用数据属性的数目的情况下，使接口语言能够让用户跨越任何数据维来进行计算和数据操作，而且不限制数据单元之间的任何关系。

直观的数据操作_允许用户能够直观地完成合并路径的旋转、下钻和概括化和其它的操作，也允许用户在分析模型的数据单元上进行移动鼠标、点击选项和拖拉等动作。要避免在用户界面中使用菜单或者多个访问路径。

灵活的报表_使商业用户能够以一定方式安排列、行和单元，这样将有利于用户进行简单的操作、分析和信息的综合等操作。同样的，包含任何子集的每个维也要能被容易地显示出来。

无限的维和聚集层_在一个通用的分析模型里最少也要能容纳 15 个数据维，能够容纳 20 个那更好。任何维在任一给定的合并路径中都应允许用户自己定义无限多层的聚集。

除了这 12 条基本的原则外，用户也应考虑下面的这些要求。Dr.Code 并没有把所有的这些要求都列入他的原则清单里。

下钻到细节级别_允许多维的预聚集的数据库平滑地转换到处于细节记录级别的源数据仓库的存储库。

OLAP 分析模型_支持 Dr.Code 的四个分析模型：注释的（或叙述的），分类的（或说明的），计划的和公式的。

非标准化数据的处理_禁止在一个 OLAP 系统内的运行的计算影响到外部的数据服务。

存储 OLAP 结果_不要将具有写操作能力的 OLAP 工具配置在交易系统的顶端。

遗漏值_可以忽略遗漏值，不考虑它们的来源。

越来越多的数据库更新_对抽取来的、聚集的 OLAP 数据进行更多的刷新操作。

SQL 接口_使 OLAP 系统无缝地融入到现有的企业环境中。

OLAP 特征

让我们用简单的术语总结一下我们所谈到的东西。我们研究了商业用户绝对需要联机分析处理的原因。我们研究了为什么其它的信息传送方法不能满足多维分析在强大的计算和快速的访问这两个方面的要求。我们也讨论了 OLAP 能够怎样满足这些要求。我们也回顾了 OLAP 系统的定义和有权威性的原则。

在我们更加详细地讨论 OLAP 系统的特征之前，让我们用平实的语言来列出 OLAP 系统的最基本的特征：

●让商业用户有一个多维的逻辑视图，用来查看数据仓库的数据。

●使用户更方便地使用交互式的查询和复杂的分析。

●允许用户沿着单一的商业维或者跨越维度进行下钻操作，以获得更详细的数据或者进行概括化操作以获得度量的聚集。

●使用户能够进行复杂的计算和比较等操作。

●用许多有意义的方式来显示结果，这些方式包括图表和图象。

主要的特征和功能

你经常要面对一个问题：OLAP 是否不仅仅是个有着良好包装的数据仓库储存？难道联机分析处理只是一种信息传送技术吗？它除了是数据仓库中的另外一层用来提供数据和用户之间的接口之外，是否还有别的作用？OLAP 在某种意义上是数据仓库的一个信息传送系统。但 OLAP 不只是扮演这些角色。数据仓库储存数据并为用户提供对数据的简单访问，而 OLAP 系统将信息的传送能力提到新的高度，因而是数据仓库的一个补充。

一般的特征

在这一节，我们将特别注意 OLAP 系统的一些主要特征和功能。你将会更深入地接触多维分析，更深地体会在分析会话中进行下钻和概括化操作的必要性，更多地理解在分析中进行多层次/多视角查看操作的作用。在更详细地研究这些内容之前，我们要概括 OLAP 的一般特征。请看表 15-4，注意表中对 OLAP 的一般特性的总结，也应注意 OLAP 的基本特征和高级特征之间的差别。表格中显示的清单包括了你能在大多数实际的 OLAP 环境中观察到的一般特征。你可以使用这个清单作为 OLAP 特性的快速核对表，而且你的项目小组必须在 OLAP 系统考虑这些特性。

维度分析

到现在你大概已经厌倦了“维度分析”这个术语。迄今为止我们不得不多次地使用这个术语。你已经知道维度分析是 OLAP 武器库中的一个强大的装备，而且任何缺乏维度分析的 OLAP 系统绝对是没用的，所以要努力地了解 OLAP 系统提供的维度分析功能。

让我们从一个简单的星形视图开始。这个星形视图有三个商业维，即产品、时间和商店。事实表包含销售。请看表 15-5，它显示了这个星形视图和这个立方体模型的一个三维表示图，其中 X 轴表示的是产品，Y 轴表示的是月份，而 Z 轴表示的是商店。那么立方体上的各个轴表示的值是多少呢？例如，在这个星形视图中，时间是其中的一个维而月份是这个时间维的一个属性。月份这个属性的值显示在 Y 轴上。同样的，产品名和商店名属性的值显示在其它两个轴上。

这个视图只有三个商业维，看起来可能不大象是一颗星星。然而，它是个一维模型。在维表的属性中，从产品维中挑出产品名这一属性，从时间维中挑出月份这一属性，从商店维挑出商店名这一属性。现在看看这个立方体，它是利用物理立方体的几个主要边缘来显示这些属性的值。现在我们要更进一步，用立方体三条边的交叉点来表示纽约的一家商店 1 月份外套的销售额，其中产品是外套，月份是 1 月，而商店是纽约。

如果你要根据这三个维度将销售数据显示在一个电子数据表上，那么可以用表格的列来显示产品名，用行来显示月份，用页来显示商店名。请看一下表 15-6，它显示的是这些三维数据的其中一页。

屏幕上的这个页面显示了立方体的一个层次。现在看一下这个立方体，沿着一个层次或

平面移动并通过 Z 轴上的表示商店的那个点：纽约。这个层次或平面上的交叉点和不同产品不同时间的销售额有关。设法将这些销售数字和立方体上表示商店（纽约）的那个层次关联起来。

现在我们有了一个方法，用来在一个二维页面或一个三维立方体上描述三个商业维和一个事实。这个页面上每个单元格中的数字是销售额的数目。这个特殊的数据集的多维分析的类型会是什么呢？在这个分析会话的过程中可以运行什么类型的查询呢？沿着产品、商店和时间这三个维的结合体的各个层次，你可以得到这些销售额的数目。你也可以对销售情况进行各种类型的三维分析操作。分析会话中的查询结果将显示在屏幕上，而屏幕上列、行和页将分别表示这三个维。接下来我们要举一个关于简单的查询和多维分析会话过程中的结果集的例子。

查询

显示所有商店过去 5 年内所有产品的销售总计

结果显示

行：年份 2000，1999，1998，1997，1996

列：所有产品的销售总计

页：每页表示一个商店

查询

比较所有商店在 2000 和 1999 年间不同产品的销售总计

结果显示

行：年份 2000，1999；差别；增长（或下降）百分比

列：每个产品一列，显示所有产品

页：所有商店

查询

比较所有商店在 2000 和 1999 年间不同产品的销售总计，而且只针对那些销售额下降了的产品

结果显示

行：年份 2000，1999；差别；下降率

列：一个产品一列，而且只显示销售额下降了的那些产品

页：所有商店

查询

比较 2000 和 1999 年间不同商店不同产品的销售额，而且只针对那些销售额下降了的产品

结果显示

行：2000，1999；差别；下降率

列：一个产品一列，而且只显示销售额下降了的产品

页：每页一个商店

查询

显示上一个查询的结果，但要旋转和交换行和列

结果显示

行：一行一个产品，而且只显示销售额下降了的那些产品

列：年份 2000，1999；差别；下降率

页：每页一个商店

查询

显示上一个查询的结果，但要旋转和交换页和行

结果显示

行：每个商店一行

列：年份 2000，1999；差别；下降率

页：每页一个产品，而且只显示有销售额下降了的那些产品

这个多维分析可以继续下去，直到分析员确定了有多少产品的销售额下降了和哪个商店的损失最大。

在以上的例子中，我们只有三个商业维度，因此我们可以用一个立方体的边来表示每个维，也可以用行、列和页来显示结果。现在我们要增加另外一个商业维：营销，这将使得维的数目增加到四个。当你只有三个维时，你能够用一个立方体来表示这三个维，其中立方体的每一边表示一个维。同样的你也能够在电子数据表中显示三维的数据，用行和列表示其中的两个维，而用页表示第三个维。但是当你有四个维或者更多的维时，你该如何来表示数据呢？显然，一个三维的立方体是不起作用了。当你设法要在具有行、列和页的电子数据表上显示数据时，同样也遇到了一些问题。因此当有三个以上的维时，你该如何来进行多维分析的呢？这个问题将我们引到有关超立方体的讨论。

什么是超立方体？

让我们从产品和时间这两个维开始我们对超立方体的讨论。商业用户通常不但希望分析销售的情况，而且也希望分析其它的度量。我们假定他们要分析的度量是固定成本、可变成本、间接销售、直接销售和利润率。这是五个普通的度量。

这里描述的数据会显示在电子数据表上，列表示度量，行表示时间，而页表示产品。请看图 15-7，它是电子数据表显示的样本页面。在这个图形中，你要注意右边那三条直线，其中的两条直线显示了两个商业维而第三条直线显示了刚才我们提到的那些度量，你可以独立地沿着直线向上移或者向下移。一些专家将这种多维的表示法称为多维域结构。

这个图形也显示了一个利用边缘来表示数据点的立方体。将这三条直线和立方体的三条边联系起来。现在你在图形中看到的页面是单个产品的一个层次和根据其它两条直线进行的划分，而这两条直线也在页面中的列和行里显示出来的。有了这三组数据——两组关于商业维的数据和另一组关于度量的数据——我们就能够容易地通过立方体的三条边使这些数据变得更加形象生动。。

现在往模型中加入另外一个商业维。我们加入了商店这个维，结果导致了我们具有了三个商业维和一些度量数据。你如何将这四组数据表示为一个三维立方体的各条边呢？你如何通过一个三维立方体的边来表示一个具有一些数据点的四维模型呢？你又如何将这些数据切分成各个显示页呢？

这些 MDS 图表最拿手的事情。现在你不需要通过三维立方体的边缘来认识四维数据。你需要做的只是像 MDS 那样画出四条直线，用这四条直线来显示数据。请看图 15-8。通过这个图形你明白了用立方体来表示四个维的数据的这种做法是行不通的。但正如你所看到的，MDS 非常适合用来表示四个维的数据。你能想到用 MDS 的四条直线来直观地显示具有四个主要边缘的“立方体”吗？这种直观的显示就是一个超立方体，一个能够容纳三个以上的维的表示方法。一个超立方体是对多维数据的表示方法的一个普通的比喻。

你现在已经有了一个可以表示四个维的数据的方法，例如超立方体。接下来的一个问题是有关于如何在屏幕上显示四个维的数据的。你如何用行、列和页这三个显示组来显示四个维的数据呢？请将你的注意力转到图 15-9。关于这些显示组你注意到什么了吗？这个图形怎样来解决只用三个显示组就能显示四个维的数据这一问题？解决问题的方法就是在同一个显示组里结合多个逻辑维。你要注意产品和这些度量是怎样结合起来用列显示的。显示的这个页面是用来表示商店（纽约）的销售额。

让我们再看一个用 MDS 来显示一个超立方体的例子。我们将维增加到了六个。请研究图 15-10，它用六条直线来显示。这个图形中显示的维分别是产品，时间，商店，营销，客户人口统计和度量。

你有若干个方法可以在屏幕上显示六个维的数据。图 15-11 举例说明了一个这样的显示六个维的方法。请仔细地研究这个图形，注意产品和度量怎样结合起来用列显示，同时也要注意商店和时间怎样结合起来用行显示，注意人口统计和宣传怎样结合起来用页显示。

我们已经回顾了两个具体的问题。首先，我们提出了一个用 MDS 来显示三维以上的数据模型的特殊的方法。这个方法能够直观地显示一个超立方体。只有三个维的模型可以用一个实际的立方体来表示。但实际的立方体只局限在只有三个维或更少的维度的一些模型。其次，我们也讨论了当维数为三或者更多的时候我们应该如何在屏幕上显示这些数据。因为我们已经都解决了这两个问题，所以现在我们要转移到多维分析的两个非常有意义的方面。一个是下钻和概括化操作；另一个是多层次－多视角查看操作。

下钻和概括化的操作

回到图 15-5，看一下那个星形视图中的产品维表中的属性。要注意产品维的一些特定属性：产品名，子类，种类，产品线和部门。这些属性显示了从产品名到部门这样的一个上升的层次顺序。部门包含产品线，产品线包含种类，种类包含子类，而每个子类是由各自的产品名所组成。在一个 OLAP 系统中，这些属性被称为产品维的层次。

OLAP 系统为它的用户提供了下钻和概括化的能力。我们提到上面那个例子，也提到了 OLAP 具有下钻和概括化的能力，这有什么含意呢？请看图 15-12，它列出了和产品的维的层次有关的 OLAP 的能力。你要注意图中所给信息的不同类型。这个图显示了 OLAP 既能够概括化到较高的层次级别的聚集，也能够下钻到较低的层次级别的细节。你也要注意旁边所显示的销售数字，它是一个特定的商店在一个特定的月份里在这些聚集级别上的销售额。当你沿着图中的那些层次往向下走时，你会注意到那些销售数字，它们分别是用来说明一个单独的部门，一个单独的产品线和一个单独的种类等的销售额。你可以下钻以获得销售在较低层次的详细分类。这个图也显示了 OLAP 能够通过使用其它维的不同层次集，以跨越到其它的维并下钻另外的 OLAP 的汇总。你也要注意下钻并通过了一些储存在源数据仓库的存储库中的较低级别的粒度的这项操作。概括化，下钻，跨维下钻和下钻并通过一些粒度等

这些操作，是支持多维分析的 OLAP 系统的一些尤其有用的特性。

现在我们还剩下一个问题。当你正在进行概括化和下钻的操作时，你如何改变在电子数据表上显示的页面呢？例如，让我们回到图 15-6 并看一下显示在电子数据表上的页面。那个页面中的列显示了各种产品，行显示了一些月份，而页显示了一些商店。在这种情况上，如果你想要概括化到下一个较高级别的子类，那么该如何改变图 15-6 显示的页面呢？我们必须改变页面中的列，以用它来显示不同的子类而再不是各种产品。请看图 15-13，它显示出了这种变化。

在我们结束这个子节前，我们要再提出一个问题。当你已经概括化到产品维中的子类这个级别时，如果你也概括化到了商店维中的下一个更高的级别，这会出现什么情况呢？电子数据表上显示的页面会是什么样子的呢？现在电子数据表将用列来表示各种子类，用行来表示各种月份，用页表示各种地域，通过这种方法来显示销售的情况。

多层次/多视角查看或旋转的操作

让我们回到图 15-6。每个页面显示了某个商店的销售情况。通过用立方体的几个主要的边来显示这些数据元素的这个方法，我们可以把这个数据模型对应到一个自然的立方体上。显示的页面是立方体的一个层次或二维平面。特殊的，为纽约商店的显示这个页面是与产品、时间轴都平行的一个层次。现在请开始仔细地观察图 15-14。图表左边的第一部分显示该立方体的这种排列。为了简单起见，我们只列出了三种产品、三个月份和三个商店。

现在旋转这个立方体以便产品是由 Z 轴表示，月份是由 X 轴表示和商店是由 Y 轴表示。显然我们正在考虑的那个层次也会跟着旋转，那么显示那个层次的页面会发生什么情况呢？和刚才的页面不同的是，现在列是用来显示各种月份，行是用来显示各种商店了，而整个页面显示的是一种产品即帽子的销售情况。

你可以继续下一个旋转以便让月份由 Z 轴显示，商店由 X 轴显示，产品由 Y 轴显示。我们正在考虑的那个层次也跟着旋转，那么显示那个层次的页面会发生什么情况呢？和刚才的页面也不同的是，列是用来显示各种商店，行是用来显示各种产品，而整个页面显示的是某个月份即 1 月的销售情况。

所有的这些对用户有什么巨大的好处呢？你是否发现，通过每次旋转用户能够看到立方体中各种不同版本的层次的页面显示，而且用户能够从很多角度查看一些数据，更好地理解

这些数据和得出一些有意义的结论。

OLAP 的使用和从中获得的好处

在详细地研究了 OLAP 的一些特性之后，你一定已经从中推断出使用 OLAP 将获得一些巨大的好处。我们已经讨论过 OLAP 系统中的多维分析。执行一个含有一些复杂的查询的多维分析操作有时也会带来复杂的计算。

现在让我们总结一下 OLAP 系统能够带来的一些好处：

●能够提高业务经理、主管人员和分析员的生产率

●OLAP 固有的灵活性意味着无需 IT 的帮助用户就可以自己执行他们的分析

●使 IT 开发人员得到好处，因为使用专门为系统开发而设计的软件能够缩短应用系统的交付时间。

●使用户能够自给自足，这能够减少未交付的订货数量

●使一些应用系统能够更快地交付给用户

●通过减少执行查询操作和网络通信的时间，使 OLAP 用户的操作更有效

●能够利用一些度量和维来模拟现实世界中的一些问题

OLAP 模型

你听说过 ROLAP 或 MOLAP 这些术语吗？这还有另外一个变种，DOLAP。这些变种的一个非常简单的解释是和 OLAP 存储数据的方法有关的。这些仍然是联机分析处理，只是各自的存储方法不同而已。

ROLAP 表示的是关系型联机分析处理（relational online analytical processing），而 MOLAP 则表示多维联机分析处理（multidimensional online analytical processing）。无论是哪种情形，它们的信息接口仍然是 OLAP。另外 DOLAP 表示桌面联机分析处理（desktop online analytical processing）。DOLAP 是用来为联机分析处理的用户提供便携性的。根据 DOLAP 的方法学，只需要在一台桌上型电脑上装有 DOLAP 软件，就能够创建多维的数据集并将它传送到这台电脑上。DOLAP 是 ROLAP 的一个变种。

变种的概述

在 MOLAP 模型中，通过多维地储存数据从而最好地实现了联机分析处理，换句话说，也就是能够用多维的方式轻松地查看数据。这里的数据结构是固定的以便用来处理多维分析的逻辑能够基于建立数据存储坐标的方法。通常，多维数据库（MDDBs）是提供商的专有系统。另一方面，_ROLAP 模型只能依赖于现有的数据仓库的关系数据库管理系统，它的 OLAP 的特性是为关系数据库提供的

请看图 15-15，对比图中的那两个模型。注意图左边显示的 MOLAP 模型，它的 OLAP 引擎是放在一个特殊的服务器里。专有的多维数据库（MDDBS）是以多维的超立方体的形式来储存数据的，所以你必须执行一些特殊的抽取和聚集工作，以在多维数据库中创建这些多维的数据立方体。这个特殊的服务器能够像 OLAP 立方体那样来显示这些多维数据，以方便用户的处理。

在图的右边你能看到一个 ROLAP 模型。OLAP 的引擎是安置在一台桌上型电脑上。预先编制的多维的立方体并不是提前创建好的，而是储存在一个特殊的数据库里。关系数据就是用这些形象的多维的数据立方体来显示的。

MOLAP 模型

正如我们讨论过的，在 MOLAP 模型中，要分析的数据是储存在专门的多维的数据库中。大型的多维阵列形成了存储结构。例如，要储存 500 单位的 ProductA 产品的一些销售数字，其中月份是 2001/01，商店是 StoreS1，分配通道是 Channel05，销售数字 500 是储存在一个阵列中，用一些值（PrcductA，2001/01，StoreS1，Channel05）来表示。

阵列的这些值指出了单元的位置。这些单元是一些维属性值的交叉点。如果你注意单元是如何形成的，你就会明白不是所有的单元都有度量值。如果一个商店是在星期日关门，那么表示星期日的这些单元将全部是空的。

现在让我们考虑一下 MOLAP 的体系结构。请仔细观察图 15-16 的每个部分，注意多维的体系结构中的那三个层。经过预先计算和预先编制的多维的数据立方体是储存在多维的数据库中。在应用层的 MOLAP 引擎会主动将多维数据库中的数据的一个多维视图传送给用户。

正如我们以前提到过的，多维数据库管理系统是一些专有的软件系统。这些系统使用户在数据从主数据仓库加载到多维数据库的过程中，能够合并和制作汇总的立方体，因此那些需要汇总数据的用户将能够享受到预先合并的数据所带来的快速的反应时间。

ROLAP 模型

在 ROLAP 模型中，数据是作为行和列以关系形式来储存的。这个模型以商业维度的形式为用户显示数据。为了向用户隐藏该模型的存储结构和多维的显示数据，这个模型创建了一个元数据的语义层，该语义层支持从维到关系表的映射。一些附加的元数据支持概括和聚集，而且你可以在关系数据库中储存元数据。

现在让我们来看图 15-17，它显示的是 ROLAP 模型的一个三层体系结构。在中间层即应用层中的分析服务器在工作中创建了多维的视图。在表示层中的多维系统为用户提供一个多维的数据视图。当用户基于这个多维视图发出复杂查询的请求时，查询被转换成复杂的 SQL 语句传给关系数据库。不像 MOLAP 模型，ROLAP 模型不能创建和保持静态多维结构。

实际的 ROLAP 有三个截然不同的特征：

●支持所有的先前讨论过的基本的 OLAP 特性和功能

●能够以关系的形式储存数据

●支持某种形式的聚集

本地的超立方体是提供商提供的 ROLAP 的一个变种。下面是它工作的过程：

用户发出一个查询请求。

将查询结果储存在一个小的、本地的多维数据库。

用户对这个本地的数据库进行分析操作。

如果用户要求额外的数据来继续分析，他将发出另外的查询并继续分析下去。

ROLAP VS MOLAP

你是否应该使用关系方法或多维的方法来为你的用户提供联机分析处理？那取决于查询的性能对你的用户有多重要。此外，要在 ROLAP 和 MOLAP 之间的做出什么选择也取决于你的用户要执行的查询的复杂度。图 15-18 基于对查询的性能和复杂度的考虑，用图表显

示了这两种方案选项。选择 MOLAP 是为了快速反应和更密集的查询。这些只是两个主要的考虑。

作为项目小组的技术组成的一部分，你在模型选择上的看法和用户在这方面的看法是完全不同的。虽然用户能够从任何一个模型中获得多维的功能和益处，但他们会更加关注可用来进行分析的商业数据的范围、性能的能否被接受和成本的合理性等这些问题。

让我们用图 15-19 来总结一下有关 ROLAP 和 MOLAP 选择的讨论。图中的那个表格基于数据存储、技术和特性这些特殊方面来比较这两个模型。这张表是重要的，因为它集合了每件事物并显示了一个和谐的例子。

OLAP 执行的考虑事项

在考虑你的数据仓库的 OLAP 的执行之前，你不得不考虑有关在 MDDBMS 下运行的 MOLAP 模型的两个关键问题。第一个问题和缺少 OLAP 的标准化有关，使得每个厂商的工具有它们自己的客户端接口。另一个问题是可测量性。OLAP 通常都能有效地处理汇总数据，但却不擅长处理大量的细节数据。

另一方面，当你正在执行复杂的分析时，数据仓库中的高度标准化的数据可能会导致系统额外的开销。你可以通过使用一个星形视图的多维设计方法来减少这些开销。事实上，对一些 ROLAP 工具来说，能够多维地表示星形视图排列中的数据是它们必须具备的条件。

考虑一下我们该如何来选择 OLAP 的体系结构。看图 15-20，它显示了 OLAP 体系结构的四种选项。

现在你已经研究了为你的数据仓库提供 OLAP 功能的各种选项。这些都是重要的选择机会。记住，没有 OLAP 你的数据仓库用户就只有非常有限的分析数据的方法。现在让我们考察一些具体的设计考虑事项。

数据设计和准备

数据仓库可以为 OLAP 系统提供数据。在 MOLAP 模型中，单独的专有的多维数据库是以多维立方体的形式来储存从数据仓库中得到的所有数据。另一方面，在 ROLAP 模型中虽然不存在静态的中间的数据存储库，但数据仍然被送到 OLAP 系统中，而且这些 OLAP

系统会在工作中动态地创建一些立方体。这样，数据是从操作型源系统流向数据仓库，接着又从数据仓库流向 OLAP 系统。

有时侯，你可能希望能够缩短这些数据的流动路径，你可能想知道你不应该在操作型源系统之上建立 OLAP 系统的原因。为什么不将操作型源系统中的数据直接抽取到 OLAP 系统呢？为什么要费劲的先将这些数据转移到数据仓库，接着再由数据仓库转移到 OLAP 系统呢？下面是对这个问题的一些解释：

●OLAP 系统需要的是转换过的、综合的数据。这些系统事先假定数据在到达它之前就已经进行过合并和净化的操作。各种操作型系统之间的差异使得它们无法直接地进行数据的综合。

●操作型系统只保持一定范围的历史数据，但是 OLAP 系统需要广泛的历史数据，因此操作型系统的历史数据在送往 OLAP 系统之前，必须先结合档案中的历史数据。

●OLAP 系统要求数据必须是用多个维来表示的。这需要很多不同的方式的汇总。从各种不同的操作型系统中抽取数据并同时汇总这些数据，这是不可能的。因此在以不同的结合方式对一些数据进行各种级别的汇总操作之前，必须先合并这些数据。

●假定你的数据仓库环境中有一些 OLAP 系统，也就是说，要用一个 OLAP 系统来支持市场部门，用另一个 OLAP 系统来支持存货控制部门，还有另外一个系统来支持金融部门等等。为了实现这个目标，你必须在操作型系统上为每个 OLAP 系统建立一个单独的抽取数据的接口。你能想象这样做会有多困难吗？

为了帮助用户来准备 OLAP 系统的数据，让我们首先来考察这个系统中的数据的一些有意义的特征。请浏览下面的条目：

●和数据仓库相比，OLAP 系统储存和使用更少的数据。

●OLAP 系统中的数据是汇总的，因此在这个系统中你几乎找不到任何最低级别的数据（即细节数据）。

●OLAP 数据处理和分析起来更加灵活，部分是因为这些操作涉及的数据太少了。

●你的环境中的 OLAP 系统的每个实例是为了服务的目的而定制的。按以前的话说，OLAP 数据是更加倾向于部门化，然而数据仓库的数据却是用来满足企业范围的需要。

一个最重要的原则是 OLAP 数据通常是定制的。当你建立一个 OLAP 系统并用系统的各个实例为不同的用户组服务时，你需要牢牢地记住这一点。例如，一个特殊的汇总结果集是用来满足一个用户组的需要，我们假设这个用户组是一个销售部门。让我们迅速地浏览一

下为一个特别的用户组或特殊的部门（例如销售部门）准备 OLAP 数据的一些技术。

定义子集_选择这个销售部门感兴趣的细节数据的子集。

概括_按照这个销售部门需要的汇总的方式来汇总和准备聚合的数据结构。例如，根据销售部门定义的产品种类来汇总产品。有时侯，销售部门和会计部门可能会以不同的方式来划分产品的种类。

非标准化_销售部门需要非标准化的数据，同样地也需要结合不同的关系表。如果销售部需要表 A 和表 B 进行联合，而金融部需要表 B 和表 C 进行联合，那么应该进行表 A 和表 B 的联合并使它成为销售 OLAP 的子集。

计算和推导_在你的公司中如果各个部门对一些度量的计算和推导都不一样，那么你应该使用销售部门的计算和推导。

索引_为销售选择那些适合建立索引的属性。

怎样为 OLAP 的数据结构建立数据模型呢？OLAP 结构包含了若干个级别的汇总和多种细节数据。那么你如何来为这些级别的汇总建立模型呢？

请看图 15-21，它指出了 OLAP 系统中的数据的一些类型和级别。在为 OLAP 系统进行数据建模时，你必须考虑到这些类型和级别，也要注意 OLAP 系统中的不同的数据类型。当你为你的 OLAP 系统建立数据结构的模型时，你需要提供这些类型的数据。

管理和性能

现在让我们将注意力转移到两个重要的但没有直接关系的问题。

管理_这些问题中的一个是 OLAP 环境的管理问题。OLAP 系统是全面的数据仓库环境的中一部分，但是为了管理 OLAP 系统我们必须了解一些关键的考虑事项。现在我们简要地指出了一些这样的考虑事项。

●对用户将访问什么数据和怎样访问这些数据的预期

●选择正确的商业维

●为从数据仓库导入数据选择正确的过滤器

●将数据移到 OLAP 系统（MOLAP 模型）的一些方法和技术

●对聚集，汇总和预计算的选择

●使用 OLAP 提供商的专有软件来开发应用系统

●多维数据库的大小

●多维结构的稀疏矩阵特性的处理

●下钻到最低级别的细节

●下钻并通过数据仓库或一些源系统

●跨越 OLAP 系统的那些实例并下钻

●访问和安全的特权

●备份和恢复的功能

性能_首先你必须认识到你的数据仓库环境中的 OLAP 改变了你的环境的工作量。一些查询通常是在数据仓库里运行，现在它们将被重新分配到 OLAP 系统。需要 OLAP 的那些查询的类型是复杂的并充满了相关的计算。漫长的、复杂的分析会话是由这样的一些复杂查询组成。因此可以将这样的查询装载到 OLAP 系统中，这将充分地减少主要的数据仓库的工作量。

将复杂的查询转移到 OLAP 系统必然使数据仓库的查询性能得到全面的改善。因为 OLAP 系统是专门为复杂的查询而设计的，所以如果在 OLAP 系统中运行这样的查询，查询的速度将会变得更快。随着数据仓库的容量的增长，OLAP 系统的容量仍将是可管理的、相对较小的。

多维数据库为每个复杂的查询提供了一个可预测的、快速的和一致的反应，这主要是因为 OLAP 系统预先聚集和预先计算了许多（如果不是全部的话）可能的超立方体，并储存了这些超立方体。这些查询将碰到最合适的超立方体。例如，假定只有三个维。这个 OLAP 系统将计算和储存下面的这些汇总：

●用来储存基本数据的一个三维的低级别的阵列

●用来储存维 1 和维 2 的数据的一个二维阵列

●用来储存维 2 和维 3 的数据的一个二维阵列

●一个高级别的维 1 的汇总阵列

●一个高级别的维 2 的汇总阵列

●一个高级别的维 3 的汇总阵列

所有的这些预先计算和预先聚集操作使得 OLAP 系统能够对查询在任何级别上的汇总做出更快速的反应。但是获得这种速度和性能不是没有代价的。你在装载的性能上付出了一定程度的代价。OLAP 系统不是每天都更新的，这是因为预先计算并装载所有可能的超立方体的装载时间是非常惊人的。企业使用更长的时间间隔，用来更新他们的 OLAP 系统。大部分的 OLAP 系统一个月更新一次。

OLAP 平台

OLAP 系统物理上是配置在什么地方呢？它应该和主要的数据仓库放在同一个平台上吗？是否应该重新计划把它放在一个单独的平台上？数据仓库和 OLAP 系统的容量的增长又是怎样的呢？它们的增长模式怎样影响企业的决策呢？当你提供 OLAP 性能为你的用户时，这些是你需要回答的问题。

数据仓库通常和 OLAP 系统在同一个平台上工作。当两者的容量都比较小时，基于成本的考虑而将它们放在同一个平台上是合理的。但是在一年时间内，用户通常会发现主要的数据仓库的容量有了快速的增长。这种趋势通常会持续下去。当出现了这种增长之后，你可能希望把 OLAP 系统转移到其它平台以减轻系统的拥塞。但你如何准确地知道是否应该分离这个平台和什么时候分离才是最好的？下面是一些指导原则：

●当主数据仓库的容量和使用逐步上升使得数据仓库需要该公共平台的所有资源时，你就可以开始分离这个平台。

●如果太多的部门需要这个 OLAP 系统，那么该 OLAP 系统应该运行在另外的平台上。

●用户希望 OLAP 系统是稳定的，并且能够很好地运行。OLAP 的数据更新不是很频繁，虽然对 OLAP 系统来说是个事实，但主要的数据仓库需要日常的应用程序来装载和全面更新某些表格。如果这个主要的数据仓库的这些日常事务干扰了 OLAP 系统的稳定性和性能，那么将 OLAP 系统移到其它平台上去。

●显然，在 OLAP 用户分散在不同的地域的这样的一个企业中，有必要设置一个或者更多的 OLAP 系统的平台。

●如果该 OLAP 系统的一个实例中的一些用户想要呆在远离其它一些用户的地方，那么你需要调查这个平台的分离情况。

●如果选择的 OLAP 工具需要的配置和主要的数据仓库的平台不同，那么这个 OLAP 系统

需要一个能够正确配置的单独的平台。

OLAP 工具和产品

OLAP 的市场正在变得越来越完善。现在已经出现了很多 OLAP 产品，而且大多数现有的产品取得了相当大的成功，产品的质量和灵活性也得到了显著的改善。

在我们提供一个用来评估 OLAP 产品的清单之前，我们将列出一些主要的指导原则：

●应该由你的应用系统和用户来引导你选择 OLAP 产品，而且不要被浮华的技术所迷倒而失去应有的判断力。

●要记住，你的 OLAP 系统的容量和当前用户的数量上都会增长。在选择 OLAP 工具之前要确定该产品是否具有可升级的能力。

●考虑管理 OLAP 产品是多么的容易。

●性能和灵活性是你的 OLAP 系统能否成功的关键因素。

●随着技术的进步，ROLAP 和 MOLAP 之间的度量差别变得有些模糊了。不要为这两个方法而烦恼，而要将精力集中在如何匹配提供商的产品和你的用户的分析需求。记住，浮华的技术不一定都是成功的。

现在我们要接触一些 OLAP 工具和产品的选择标准。当你评估这些产品的优劣时，请使用下面的核对清单并将每个产品和核对清单中的每一项进行一一对照：

●能够进行数据的多维显示

●能够实现聚集、汇总，预先计算和推导操作

●在一个广阔的库里有一些公式和复杂计算

●能够进行跨维的计算

●具有时间上的智能，例如由年初至今的（year-to-date）、当前的和过去的结帐期和变化中的平均数和总计

●能够根据单个或多个维进行旋转、交叉表格、向下查询和概括化操作

●具有和诸如电子数据表，私有的客户端工具，第三方工具和第四代语言环境等这些应用系统和软件的接口。

执行步骤

在这里，大概你的项目小组已经获得了建立并执行一个 OLAP 系统的命令。你已经了解了它的特性和功能，了解了它的意义，同时也了解一些重要的考虑事项。那么你将如何着手建立 OLAP 系统呢？让我们概括一下需要的一些关键步骤。这些步骤或活动是非常高级的。每个步骤是由若干个任务组成的而且这些任务将完成该步骤的目标。你将不得不基于你的环境的需求来提出一些任务。下面是一些主要的步骤：

●为各个维建立模型

●设计并建立 MDDB

●选择将要转移到 OLAP 系统的那些数据

●抽取数据并将这些数据送到 OLAP 系统中

●为 OLAP 系统装载数据

●对数据聚集和导出的数据的计算

●在桌上型电脑上执行某个应用程序

●对用户进行培训

本章总结

●OLAP 是至关重要的，因为它的多维分析、快速的访问和强大的计算超过了其它的分析方法。

●OLAP 的定义是基于 Codd 最初的 12 条指导原则。

●OLAP 的特征包括多维的数据视图、交互式的复杂的分析工具、执行复杂计算的能力和快速的反应时间。

●多维分析不受可以用一个立方体来表示的三维的限制。超立方体提供了一个表示更多的维的视图的方法。

●ROLAP 和 MOLAP 是两个主要的 OLAP 模型。它们之间的差别在于基本数据的存储方式。你要确定哪个模型更适合你的环境。

●OLAP 工具已经发展成熟了。一些 RDBMSs 包含了对 OLAP 的支持。

思考题

1. 简要地解释多维分析的概念。
2. 说出 OLAP 系统的任意四个关键的能力。
3. 陈述 Dr.Codd 关于 OLAP 系统的任意五条指导原则，并对每条原则进行简要的描述。
4. 什么是超立方体？如何在 OLAP 系统中应用超立方体？
5. 多层次/多视角查看是什么意思？请举出一个例子。
6. MOLAP 和 ROLAP 模型之间的本质区别是什么？列出它们的一些相同点。
7. 什么是多维数据库？它是怎样储存数据的？
8. 描述四个 OLAP 体系结构选项中的任意一个选项。
9. 为什么我们不推荐从源操作型系统直接把数据传送给 OLAP 系统的这种方法？讨论其中的两个原因。
10. 说出 OLAP 环境中的任意四个考虑因素。

练习题

1. 指出对或错：
    a) OLAP 使交互式的查询和复杂的查询运行起来更加方便。
    b) 超立方体能够由一个实际的立方体来表示。
    c) 多层次/多视角查看和通过旋转行和列来显示数据的这个方法是一样的。
    d) DOLAP 的意思是部门的（departmental）OLAP。
    e) ROLAP 系统将数据储存在一个多维的、专有的数据库中。
    f) ROLAP 和 MOLAP 之间的本质区别是数据的存储方式。
    g) OLAP 系统需要的是那些转换过的、综合的数据。
    h) OLAP 系统的数据几乎都不是汇总的。
    i) 多维域结构（MDS）最多只能显示六个维。
    j) OLAP 系统处理不了变化的平均数。
2. 假设你是一个出版社的项目小组中的一位高级的分析员，正在寻求对数据仓库的选择。现在请举个 OLAP 的例子，描述 OLAP 的各个度量和 OLAP 将如何在你的环境中发挥

必不可少的作用。

3. 挑出 Dr.Codd 最初的关于 OLAP 的任意六条指导原则。请解释说明为什么你选择的这六条原则对 OLAP 来说是重要的。
4. 现在你将要组建一个小组来评估 MOLAP 和 ROLAP 这两个模型并提出你的建议。这是为一个大型的重化工厂实施的数据仓库项目的一部分。描述你的小组将要使用的用来评估和选择 OLAP 工具的那些标准。
5. 假设你的公司是一家最大的鸡肉产品生产商，它的产品主要销售给超市、速食连锁店和餐馆，也出口到许多国家。来自世界各地的很多公司的分析员希望能够使用该 OLAP 系统。讨论一下项目小组必须如何为这个公司选择一个用来实现 OLAP 的平台。解释你的想法。

第十六章　数据仓库和 Web

本章目标

●理解支持 Web 的数据仓库的意思是什么，考察数据仓库必须支持 Web 的原因

●理解 Web 技术和数据仓库技术结合的含意

●深入了解基于 Web 的信息传送机制的所有方面的内容

●研究如何来连接 OLAP 系统和 Web，学会用各种不同的方法来连接它们

●考察建立一个支持 Web 的数据仓库所需要的步骤

从 20 世纪 90 年代开始，在计算机和通信领域占主导地位的现象是什么呢？毫无疑问，那就是万维网（Worldwide Web）和因特网。Web 对我们的生活和各种事情的影响只有过去的其它一些事物可以和它相比。

在 20 世纪 70 年代，人们取得了一个主要的突破，在那时引进了带有图形接口、指向设备（指向装置，给使用者输入方位的装置如鼠标等）和图标的个人计算机。而今天人们的突破是 Web，它是建立在早期技术革命的基础上。使个人计算机变成有用的和有效的是我们在 20 世纪 70 和 80 年代的目标，但是使 Web 变成有用的和有效的却是我们现在的目标。因特网的发展和 Web 的使用使早期的革命黯然失色。在 2000 年的一开始，估计在世界范围内大约有五千万的家庭在使用因特网。而到 2005 年末，估计这数字将增长 10 倍，到那时大约 5 亿的家庭将会浏览 Web。

人们都说，Web 几乎改变了人们的每件事情。而数据仓库也不例外。在 20 世纪 80 年代，数据仓库存储技术仍然处于定义和发展的阶段。在 90 年代，它开始发展成熟了。现在，经过了 90 年代的 Web 革命，数据仓库存储技术已经在 Web 运动中占据了一个显著的位置。那么 Web 为什么能够极大地影响数据仓库的发展呢？

Web 革命使人们得到的一个主要好处是什么呢？那就是能够显著地降低通信的费用。Web 已经大大地减少了传送信息所需要的费用。什么事情和这有关呢？数据仓库的一个主

要目标是什么呢？它就是战略性信息的传送机制。因此数据仓库和 Web 能够完美地结合。数据仓库用来传送信息，而因特网用来降低它传送信息的费用。我们已经明白了支持 Web 的数据仓库或“数据仓库”的概念。Web 迫使我们重新思考数据仓库的设计和发展。

在第 3 章，我们简要地考虑了一下支持 Web 的数据仓库。我们特别讨论了这个话题的两个方面。首先，我们考虑如何把 Web 当作一个信息传送渠道来使用。这将数据仓库带向了 Web，也使数据仓库向更多的用户开放。本章将关注 Web 和数据仓库之间的关系的这一方面。

在第 3 章讨论的另一方面，涉及到把 Web 带向数据仓库的问题。这一方面和你的公司的电子商务有关，你的公司网站上的点击流数据被送到数据仓库来进行分析。在本章，我们将绕过 Web－数据仓库关系的这个方面，因为几个作者和专业人员的许多文章和最近由 Dr.Ralph 和 Kimball 合作撰写的一本优秀的书已经充分地阐述了数据仓库这方面的情况。请看参考书目以获得更多信息。

支持 Web 的数据仓库

支持 Web 的数据仓库的含意是该数据仓库使用 Web 来进行信息传送和实现用户间的合作。随着时间的过去，越来越多的数据仓库连接到 Web 上。本质上这意味着对数据仓库信息的访问量正在逐步地增长。信息访问的增长，反过来导致了企业知识水平的增长。虽然在连接到 Web 之前已经能够为更多的用户提供信息的访问，但这种做法将导致很多的困难和成比例增长的通信费用，这是个事实。但是 Web 改变了所有的这一切。现在你能够更容易地增加更多的用户。通信基础设施已经在那儿。你的所有用户几乎都有 Web 浏览器，他们不需要安装额外的客户端软件。你可以发挥现有的 Web 的作用。Web 和它的网络、服务器、用户和页面成指数增长，使得人们采用因特网、企业内部互联网和外部互联网作为信息传输的媒介。支持 Web 的数据仓库占据了 Web 革命的中央舞台。让我们看看这是为什么。

为什么是 Web？

将数据仓库连接到 Web 上显得非常自然。为什么这么说呢？考虑一下你的用户是如何看待 Web 的。首先，他们将 Web 看作是一个庞大的信息来源，他们能够从中寻找有用的和

有趣的数据内容。你的企业内部互联网中的用户、顾客和商业伙伴已经在频繁地使用 Web。他们知道如何连接 Web。Web 是无处不在的，而且在 Web 上从来没有日落。用户唯一需要的客户端软件是一个 Web 浏览器，而且几乎每个人，年轻的和年老的，都已经学会了如何安装和使用一个 Web 浏览器。很多软件提供商也已经使他们的产品能够支持 Web。

现在从 Web 这个方面来考虑一下你的数据仓库。你的用户需要数据仓库来为他们提供信息。你的商业伙伴能够使用从数据仓库中得到的一些特殊信息。所有的这些有什么共同之处呢？熟悉 Web 并且能够轻松地访问它。这就是要发展支持 Web 数据仓库的强有力的理由。

你如何为你的数据仓库开发 Web 技术？你如何将数据仓库连接到 Web 上？让我们很快地回顾一下三种基于 Web 技术的信息传送机制，很多公司已经采纳这些机制。在每种机制中，用户都是用 Web 浏览器来访问信息。

因特网_第一种媒介当然是因特网了，它提供了低成本的信息传输。你可以和公司内外的任何人交换信息。因为信息是通过公共网来传送的，所以你必须要注意它的安全问题。

企业内部互联网_从 1995 年“企业内部互联网”这个术语产生开始，私有网络的概念已经已经在企业界得到了推广。企业内部互联网是一个私有的计算机网络，它是基于公共因特网的数据通信标准。在企业内部互联网内发送信息的所有应用程序都在防火墙的保护之内，因此和因特网相比它们是更加安全的，而且你还能够得到流行的 Web 技术的所有好处。此外，和因特网相比你能够更好地管理安全问题。

企业外部互联网_继因特网和企业内部互联网之后的是企业外部互联网。企业外部互联网不像因特网那样完全开放，也不像企业内部互联网那样只局限于内部的使用。企业外部互联网是对外部实体的有选择的访问进行开放的一个企业内部互联网。从你的企业内部互联网，除了向前和向下看外，你还能向外看到你的顾客、供应商和合作伙伴。

图 16-1 说明了如何通过这些信息传送机制来传送数据仓库的信息。要注意你的数据仓库怎样配置在 Web 上。只有当你要限制仅内部用户才能访问数据仓库的时侯，你才使用企业内部互联网。如果你需要对具有正确认证的外部实体开放数据仓库时，那么你就选用企业外部互联网。在这两种情况下，信息传送的技术和传输的协议都是一样的。

企业内部互联网和外部互联网具有几个优势。下面列出了它们的一些优势：

●使用一个通用的浏览器，因此你的用户将拥有一个单个的信息入口。

●只需要最小的训练就能够访问信息，因为用户已经知道如何使用一个浏览器。

●通用浏览器可以在任何系统下运行。

●Web 技术向用户开放了多种信息格式。用户能够接收文本、图象、图表、甚至视频和声

频。

●保持企业内部互联网/外部互联网的更新是很容易的，而且更新之后就能保证用户的信息是来自同一个信息源。

●通过企业外部互联网向你的合作伙伴开放数据仓库，以此来加强你们的伙伴关系。

●使你的数据仓库连接到 Web 上的配置和维护成本是比较低的，这主要是因为网络的费用减少了很多，而且基础设施的投资成本也比较低的。

技术的结合

Web 技术和数据仓库存储技术已经结合在一起而且这种结合只会变得更强，这是个事实。如果你不使你的数据仓库能够支持 Web，那你将会被你的竞争对手抛到后面。从 20 世纪 90 年代中期开始，提供商们竞相推出他们的支持 Web 的一些产品。自从 Web 产品开始出现以来，它们首次超过了客户端/服务器产品。这些产品间接地推动了 Web 和数据仓库进行更深的结合。

记住，和数据仓库相比 Web 的意义更加重大。因此 Web 和它的特性将起领导作用，而数据仓库不得不跟随着 Web。Web 已经使用户的期望达到了一个高的水平，所以用户会期待着数据仓库同样能够高水平地完成他们的任务。考虑一下由 Web 带来的一些用户期望，现在用户希望数据仓库也能采纳这些期望：

●快速的反应，虽然一些 Web 页面相对来说比较慢

●使用起来非常容易和直观

●能够一天 24 小时和一周 7 天地工作

●更时新的内容

●图形的、动态的和灵活的用户接口

●几乎个人化的显示

●希望能够连接到世界各地，能够进行下钻的操作

在最近几年，支持 Web 的数据仓库的数字已经在充分地增长。这些支持 Web 的数据仓库为什么能够发展得这么快呢？为了理解这两种技术的结合所产生的影响，我们必须考虑一下成本减少所产生的三种顺序的结果，它们是 Thomas W.Malone 和 John F.Rockart 在 20 世纪 90 年代早期提出的：

第一位结果：新技术简单地替代老技术

第二位结果：提高对新技术提供的功能的要求。

第三位结果：出现了新的技术密集型的结构

迄今为止，Web 技术和数据仓库存储技术的结合已经带来了什么结果呢？Web 数据仓库存储技术看来已经通过了前两个阶段。那些拥有支持 Web 数据仓库的公司通过信息传送的新方法已经降低了成本，而且紧跟在第一阶段后面已经提高了对信息的要求。大多数的拥有支持 Web 的数据仓库的公司，当他们到达第二阶段的末尾时，就不再往第三阶段发展了。

调整数据仓库使它能够支持 Web

人们对一个支持 Web 的数据仓库寄予了厚望，这意味着你必须彻底改造你的数据仓库。你必须做许多工作才使你的数据仓库能够支持 Web。现在让我们考虑一下一些产品或技术，它们用来使你的数据仓库能够支持 Web。

首先，让我们回到前面的关于三个阶段的讨论，然后再介绍一个新的技术。除了利用替代品来减少成本外，对数据仓库信息的要求也提高了。大多数公司似乎在第二阶段的末尾就裹足不前了。只有一些公司能够进行到下一个阶段，并且实现第三位的结果。这些结果是什么？这样的一些结果包括企业外部互联网和客户的数据集市、异常管理和自动的供应和价值链等。当你使你的数据仓库已经能够支持 Web 时，确信你不是停留在第二阶段。制定计划来开发 Web 的潜力并进行到能够找到实际的好处的第三阶段。

学习下面的使数据仓库能够支持 Web 的一些要求。

信息“推”技术_使用“推”技术来设计和实现数据仓库。信息传送系统基于用户的请求从数据仓库“拉”来信息，然后把这些信息提供给用户。Web 提供了另外一种技术。不需要用户每次的请求，Web 就能够将信息“推”给用户。你的数据仓库必须能够采纳这个“推”技术。

易于使用_通过得到的点击流的数据，你能够迅速地核对用户在网站上的行为。在其它方面，点击流的数据揭示了用户浏览这些页面有多简单或者有多困难。易于使用是用户要求比较高的一项内容。

迅速的反应_一些数据仓库允许一些任务长期运行以产生用户希望的结果。但是在 Web 模型里，用户希望获得较高的运行速度，而且这一点是不容磋商的或妥协的。

*没有停工时间*_设计 Web 模型时，要使系统在所有时间内都能运转。相同的，支持 Web 的数据仓库也不能有停止工作的时间。

*多媒体输出*_Web 页面具有多种数据类型－文本的、数字的、图形、声音、视频、动画、视频和地图。用户希望支持 Web 的数据仓库的信息传送系统能够以这些数据类型来显示输出的内容。

*个人的市场*_随着动态创建的 XML 正在逐渐地替代静态的 HTML 编码，Web 信息传送机制也变得更加的个人化。因此支持 Web 的数据仓库必须适应这种变化。

*可升级性*_更多的访问、更多的用户和更多的数据－这些是数据仓库支持 Web 后所产生的结果。因此，可升级性成为一个主要的关注事项。

作为数据源的 Web

当你正在讨论要使数据仓库支持 Web 时，第一个并且大概是唯一的一个想法是把 Web 技术当作一种信息传送机制来使用。具有讽刺意味的是，你很少想到 Web 上的内容会是你的数据仓库的一个有价值的、有效的数据源，因此在你把 Web 上的数据抽取到你支持的 Web 数据仓库之前，你可能会有些犹豫。

Web 上的信息内容是如此的不同和零碎，以至于你需要建立一个特殊的搜寻和抽取系统，用来筛选大量的信息并从中挑选出和你的数据仓库有关的内容。假定你的项目小组能够建立这样的一个抽取系统，那么这个系统的选择和抽取将由一些截然不同的步骤所构成的。在抽取需要的数据之前，你必须核对这些源数据的准确性。因为这些数据是在 Web 上找到的，所以你不能自动地假设它就是准确的。你可以从数据源的类型获得有关数据准确性的线索。请参考图 16-2，它显示了用来从 Web 上选择和抽取数据的一些组件的位置安排。

你如何使用 Web 上的内容来丰富你的数据仓库呢？以下是一些重要的使用方法：

●为商业维增加更多的描述性属性。

●要包含和某个维有关的名词性数据或序数性数据，以便将来可以获得更多的选项来进行旋转和交叉表的操作。

●增加链接性数据到某个维上，以便能够进行和其它维的相关分析。

●创建一个新的维表。

●创建一个新的事实表。

刚才我们讨论的内容大大超过了通常将 Web 当作一个信息传送媒介来使用的范围。从 Web 上选择数据和抽取数据是一个全新的范例。如果你的项目小组愿意在你的数据仓库环境中尝试它，那么结果将证明这样做是值得的。

基于 Web 的信息传送机制

我们已经明白了 Web 技术和数据仓库技术的结合为什么是必然的。这两种技术用来为用户提供各种需要的信息。Web 技术能够更轻松地昼夜不停地传送信息，因此许多公司想要使他们的数据仓库能够支持 Web，这一点也不奇怪。

当你将数据仓库连接到 Web 上时，又出现了其它的优势和可能性。其中的一个优势就是能够通过企业外部互联网的数据集市或类似的东西，提出更新的方法使数据仓库变得更有效率。我们也看到了把 Web 作为你的数据仓库的一个数据源来使用的这种可能性。

然而，更好的信息传送机制仍然是改变数据仓库使它能够支持 Web 的最有说服力的理由。Web 带来了信息传送机制的新前景，而且彻底改革了它的传送方法。因此我们要把更多的时间花在基于 Web 的信息传送机制上。Web 是如何增强数据仓库的使用呢？它带来的好处是什么，遇到的挑战又是什么呢？你将如何处理 Web 带来的在信息传送机制上的激动人心的变化呢？

扩展了数据仓库的使用

无论你怎样看待这件事情，将数据仓库连接到 Web 上所产生的优势看起来是非常令人惊奇的。用户能够轻松地在任何时候使用浏览器来执行他们的查询。在客户端/服务器环境下数据仓库同步地分配数据给各种用户不会遇到任何的麻烦。数据仓库的使用已经扩展到内部用户之外的其它用户。外部的感兴趣的团体现在也能够使用数据仓库的内容。随着使用的扩展，系统的可升级性不会再引起一系列的问题。培训这些用户的费用又是多少呢？当然，培训的费用是最小的，因为有了 Web 浏览器的使用。每件事情看起来是那么的美好，数据仓库的使用也得到了扩展。

让我们来了解在数据仓库使用时的增长是如何实现的？起初，支持 Web 的数据仓库每天只能得到 500 到 5000 的点击次数，但是依靠数据仓库的支持者，这个数字能在短时间内

暴涨。更多的用户意味着什么呢？非常简单，它意味着更多的数据。同样，因为 Web 从来不会关闭，所以它也意味着在一个 24 天×7 小时环境中将有更多数据。

我们来考察这种特别的增长现象。通用的访问产生了我们必须面对的一整套的挑战。这主要是使你支持的 Web 数据仓库强加了额外的疲劳。虽然升级系统以容纳更多用户和快速地扩展系统的使用要求不像在客户端/服务器环境下那样的紧迫，但它仍然是一个基本的挑战。

让我们来理解这种使用上的增长模式。有两种截然不同的因素在推动着这种增长：第一，一个完全开放而且从不关闭的窗口，其次，一个通过普遍存在的 Web 浏览器的简单直观的访问机制。结果，你要应付两种挑战。第一是用户数的增长。其次是这种增长的迅速的加速度。如果你已经通过一个企业外部互联网向你的客户和商业伙伴开放了你的数据仓库，你甚至会注意到这条陡峭的扩展曲线。

让我们将这种特别的增长称为“超增长”并调查这种现象。请参考图 16-3，它用图表显示了这一现象。超增长的显著特性本身证明了你无法及时地升级系统以容纳这种增长。用户量的增长速度超过了你升级数据仓库的速度，因而无法完全满足扩展中的使用要求。你不能只是简单地增加处理器、磁盘驱动器或存储器的数量来满足扩展使用的要求，那么你该怎样来处理超增长呢？

让我们提出一个最初的方法。它能否预料到这个问题并且完全避免这个问题的发生呢？换句话说，你能否控制这种增长并因此逐渐掌握这个问题呢？当你使数据仓库能够支持 web 时，你的第一个本能往往是过于热心地立刻向公众开放这个数据仓库。这就像是没有发出任何警告就打开了防洪水闸。因此为了控制增长，你必须自己抑制住你的热情，然后在精心定义的步骤内逐步开放你的数据仓库。如果数据仓库的使用模式不是非常有条理，那么这样做就是非常有必要的。最初不要一下子就向公众开放你的数据仓库，首先让你的一些内部用户能够访问它，接着增加更多的用户到这个组里，然后分段增加越来越来的用户。用这种方式，你能够持续地监控使用的增长模式。当你第二波向公众开放数据仓库时，可以采用相同的警戒的方法。

如果你的环境能保证向所有的用户，包括内部的和外部的用户，一下子开放你的支持 Web 的数据仓库，那又是什么情况呢？如果上述的分段方法不适合你的情况又该怎么办呢？如果你不能避免超增长你该怎么办呢？回答这些问题的关键在于这个增长曲线的一个明确特征。我们注意到超增长只在最初的阶段发生。在初始阶段之后，使用曲线似乎变平了或者至少增长率变得可以管理了。根据这个事实，我们来看看如何才能处理超增长的问题。

使用曲线增长到某一点然后就变平滑了，所以秘密就在于找到这个点。让我们回到显示超增长曲线的图 16-3。曲线图显示出超增长持续到十一月的前期，然后稳定在 750 用户左右。即使你从 100 用户开始，你也可能很快就会达到 750 用户的水平。因此，即使你一开始的目标只是 100 用户，你也要让数据仓库最初有足够的资源来容纳 750 用户。但你如何得出 750 用户这一数字和曲线开始变得平滑的那个点是十一月前期呢？没有任何具有工业标准的图表来预测超增长的模式，所以你的数据仓库的超增长模式完全依赖于你的环境的实际情况。你必须使用最好的评估技术来为你的环境制定超增长曲线图。

新的信息策略

当数据仓库和 Web 技术出现时，人们对数据仓库的期望是什么？数据仓库的用户接口如何在 Web 模型上进行改进和提高呢？在你的数据仓库能够支持 Web 之前，用户的期望一直是由一套定义好的标准来管理。现在，在能够支持 Web 之后，当用户使用相同的浏览器像他们获取其它因特网数据一样来访问数据仓库时，标准就不一样了。现在用户希望使用相同类型的信息接口，跟因特网会话中使用的接口一样。理解用户的这些期望将有助于你更好地为你的数据仓库开发新的信息传送策略。

信息传送原则_让我们总结一下用来阐述新的信息传送策略的一些原则。请研究下面的几点。

性能_行业内的一些专家达成了一个共识：当某个页面传送第一个屏幕的有用内容时，它的反应时间要小于 10 秒。在用户能在 10 秒之内看见页面提供的有用内容这一前提下，页面的装载时间可以更长一点。设计页面时，要以最低速度的调制解调器为标准的。要立刻地显示出导航按钮，并根据计划好的顺序来显示内容：立刻显示有用的那些内容，跟着是显示次一级的那些有用内容。要重新考虑慢的和多余的那些图形是否有必要，使用页面缓存技术，确保实际的数据库设计能够实现快速的反应时间。

用户选项_当用户访问一个页面时，他们习惯于看到一些标准的选项。这些选项包括导航的选择。支持 Web 的数据仓库的导航按钮包括下钻选项、主页、主题、网站地图、查找和网站负责人细节等等。同时也包括特别的数据仓库的选择、帮助菜单和该网站的联系方式的选择。更具体的，对一个数据仓库来说用户选项必须包括对报表库的导航、浏览商业维和选择每个维内的属性和浏览商业元数据的接口。

用户体验_每个 Web 设计者期望的最终目标是使每个页面都能令用户满意。用户都希望访问

和浏览一些精致的页面，但如果有太多事物和问题分散了用户的注意力，用户往往会关掉这个页面，因此要明智地使用字体、颜色、闪烁图形、粗体文本、下划线、声音素材和视频片断等选项。

处理_使商业系统能在 Web 上流畅地、无缝地工作是很重要的。对支持 Web 的数据仓库来说，这个要求变成了这个系统在一个分析会话中必须能够流畅地和 Web 进行交互，以便用户能够简单地、顺利地从一个步骤转移到下一个步骤。

用户支持_在一个冗长的处理中，用户必须重新确认在处理过程中没有丢失任何东西。用户必须知道他或她处在处理过程的什么位置，以便接下来的行动不会被打断。在对支持 Web 的数据仓库的访问中，要让用户知道中间过程的状态信息。例如在运行报表的时候，要为用户提供报表的状态。

解决问题_用户必须能从他们的错误中返回，纠正错误，接着继续进行。用户也必须能够报告出现的问题。

历史记录中的信息_由 Web 打开的数据仓库让人们看到了如何在更广阔的历史记录中查看各种信息。迄今为止，查询能够找到“多少钱”和“多么经常”这种直接问题的答案。这些问题的答案能够在公司框架的狭窄范围内找到。随着商业智能在 Web 上的开放，这个圈子拓宽了，已经包括完整的供应链和一些竞争的事项。现在用户可以在更大的战略框架内获得信息。

个性化的信息_我们能够为用户提供一些预定义的查询，它们将能够满足价值链上的每个人的要求，因此要努力地提供一些特别的性能，使用户能够提出和数据仓库中任何类型数据有关的任何类型的问题。

自助式访问_随着企业内部和外部更多的用户可以通过 Web 来访问数据仓库，工具必须使用户能够自助式地访问数据仓库中的信息。用户必须能够导航和探查到信息源。该信息源必须是一个带有自助式访问的实际环境，在那里用户没有别人的帮助就能够实现对信息的访问。

HTML 文件_标准的 HTML 文档或文件，也称为 Web 页面，是支持 Web 的数据仓库环境中的一种主要的通信方式。Web 页面是一种资源，可以用来在因特网或内部网络上显示信息。信息接口工具根据特别的用户查询或数据库储存过程来生成相应的 HTML 文件。

你可以在某个场合生成一个 HTML 文件，也可以使用一些触发器有规律地生成 HTML 文件。正如你所知道的，触发器是一种特殊的存储过程，当一个特殊的表中的某个具体的数据操作语句的条件得到满足时，触发器就会自动地执行。例如，当一些预定义的阈值被超过

时，触发器程序能够自动地以 HTML 文件的格式生成一个异常报表。接着通过调用相应的 Web 页面，用户就能够查看最新发生的异常事件。

数据库管理系统在它们的数据库引擎里提供了公布和预订的功能。无论何时触发器触发了一个特定的数据源，预订都使用户能够从零开始创建 HTML 页面或者刷新这些页面。一个公布的 HTML 页面能够包含直接来自一个或多个表的经过过滤的数据、由 SQL 语句查询得到的结果和存储过程的输出结果。然而，预订功能只局限于一些特殊的数据库中的数据。***作为一个战略性工具的报表***_接下来我们要讨论基于 Web 的信息传送机制的一种方法，它就是报表。在 Web 上，你能够通过 e-mail 来公布或分发文件。这些特性使报表有可能成为一个战略性工具。现在你能够将商业伙伴结合到你的供应链上。经理和主管人员可以引导被预订的报表自动地送到特定的客户和供应商那里。他们也可以设置库存水平面的阈值，并且只有在该水平面超过了限定值的时侯才让报表送出。

报表管理涉及到两种类型的报表。你可以制定常规报表和异常报表的发送时间表，也可以创建一些可以在 Web 上获得的参数驱动的报表。你甚至可以为报表贴上商业名字的标签，并根据用户的种类、等级和安全认证级别对这些报表进行分类。

几种报表管理技术是可以实现的：参数驱动的报表、可定制的报表、异常报表和预定义的报表等等。其中一个技术是 OLAP 的下钻技术，用户使用这个技术请求第一个报表以高度汇总的级别来显示结果。用户接着可以用这个报表来做进一步的分析。当第一个报表返回时，这个报表就成了下面进一步分析的基础。用户改变了请求参数，这样不用创建新的报表他就能在汇总数据上下钻额外的细节。另一个有用的技术和随需即取的（on-demand）页面有关。当报表返回若干个页面时，用户能够通过超链接而不是一次一页的页面浏览来导航到想要浏览的页面中去。

数据仓库的浏览器技术

Web 技术和浏览器技术的意思几乎是相同的。在一个支持 Web 的数据仓库中，浏览器是一个通用的客户端软件。你的用户将使用一个标准的浏览器来访问信息。让我们来回顾一下一些细节以便你能尽快地熟悉浏览器技术。一个基于浏览器的应用程序具有很多好处。用户接口——浏览器——实际上是免费的。你也不需要在客户端上配置或安装一个基于浏览器的应用系统，因为应用系统是运行在一个服务器上的。因此你立刻会发现应用系统的开发竟

然变得如此容易，甚至在该系统需要成百上千个桌上型电脑时。

目前，通常利用四种技术来建立支持 Web 的用户接口。这些技术就是 HTML、Java、ADO 和插件（Plug-ins）。请看图 16-4，它根据能力和缺点对这四种技术进行了一番比较。

下面是这四种技术的一些简要的描述：

HTML_HTML，它是管理起来最简单和最容易的技术，和平台无关可以在任何浏览器上工作。用户可以通过点击超链接来进行导航。HTML 支持图形和表格。它是“无状态的”，意思是应用系统之间并没有保存浏览器和应用系统之间的网络链接的历史记录。通过生成新的 HTML 页面，你可以实现诸如旋转和下钻等 OLAP 的一些特性操作，但你必须要等待结果页面的生成和被装载到桌上型电脑上。HTML 擅长处理静态报表，因此当你的应用系统不知道目标平台的特性时，HTML 技术是非常适合你的应用系统的。

Java_你是否需要高级的三维显示、下钻、拖放或者相似的高端功能呢？那么如果你需要这样功能的话，那么 Java 就是你需要的技术。你可以在所有主要的客户端平台上使用 Java。因为不允许 Java applet 在硬盘上进行写操作或在本地打印机上打印，这会使一些应用程序产生一些问题。因为 Java 是一种解释性的语言，所以执行起来它比编译性的语言稍微有点慢。桌上型电脑必须配置一个支持 Java 的浏览器。如果每次不得不从服务器上装载 Java applet 时，那么有时漫长的装载时间可能会令人无法接受。Java 适合作一个交互式的客户端，在那里漫长的装载时间不是一个考虑因素。

ADO_这是 Microsoft 为分布式的基于 Web 的系统而开发的解决方案。ADO，是作为 Microsoft 动态链接库（DLL）或数据链接库来执行的，用户能够通过浏览器从某个服务器上下载并安装这个链接库。正如人们所预料的，ADO 只能在 Windows 平台下运行，因此它不能配置在 UNIX 和 Mac 上 。ADO 是一个编译过的接口，因此运行速度比 Java 快。ADO/MD，作为旋转－表格服务（Pivot-Table Services）的一部分，Microsoft 对 ADO 的扩展可以用来在 Visual Basic 中创建 ActiveX 控件，通过这种控件用户就能够在一个 Web 页面上的 OLAP 服务中操作数据。ADO 被限制在 Windows 平台下使用，在那里你能够很好地对 DLL 进行操作。

插件_这是适合特定浏览器的一些程序，只能在某个浏览器上执行。插件能够安装在本地的磁盘上。因为每个浏览器都需要自己的插件，如果你选择了这个方法，你可能需要在你的环境中使用统一的浏览器。在多个平台上的 OLAP 客户端，尤其是那些使用 Java 的客户端，它们的带宽可能会受到限制。

安全问题

毫无疑问，当你通过企业内部互联网向企业内的全部用户开放你的支持 Web 的数据仓库或者通过外部网向合作伙伴开放数据仓库时，你都希望能够实现数据仓库的最大价值。当你采取下一个步骤向因特网上的大众开放你的数据仓库时，你甚至可以得到更多的价值。但这些行动导致了严重的安全问题，因此你必须采取不同级别的安全限制。

在网络级别上，你可以考虑一种支持数据加密和有限的传输机制的解决方案。网络级别上的安全只是安全计划的一部分。在应用程序级别上仔细地制定一个安全系统。在这个级别上，安全系统必须能够授权允许谁进入应用程序和允许每个用户访问什么内容，也必须能够对这些授权进行管理。

你听说过信息恐怖分子吗？授权那些不忠诚的或不可靠的雇员访问一些受保护的信息，是数据仓库安全上的一个最大威胁。要堵上这个漏洞是很困难的，但你必须从事这方面的工作。

OLAP 和 Web

你花了大量的时间和金钱来建立数据仓库，希望企业能够得到它们所需要的信息，帮助它们作出有价值的战略性决策。为了最大地实现价值的潜力，你需要尽可能地满足大多数用户的需要，这包括要扩展 OLAP 的容量使它适合更多的分析员同时使用。

企业 OLAP

早期的数据仓库是在开始时仅作为少数感兴趣的分析员所使用的小规模的决策支持系统。早期大型机上的决策支持系统具有强大的分析性能，虽然它无法和现在的 OLAP 系统相比。那些系统使用起来非常困难，因此几乎只有一小组的分析员在使用它们，这些分析员能够艰难地克服这些困难。

下一代的决策支持系统利用方便使用的图形用户界面和鼠标操作的接口取代了复杂的大型机计算系统。这些在客户端/服务器上运行的第二代系统除了支持简单的查询和报表外，现在也逐渐地能够支持 OLAP，但是开发和维护的成本仍然使它们难于扩展到更多的用户。

OLAP 和类似 OLAP 的性能仍然局限于少数的用户。

Web 为信息传送机制带来了新气象。支持 Web 的数据仓库向企业内外部的一大批用户开放，OLAP 服务的用户也超出了一组分析员的范围。但是一些问题出现了：OLAP 系统能否逐渐升级以支持一大批并发的用户执行复杂的查询和密集的计算？你的项目小组如何确保 OLAP 能在你支持的 Web 数据仓库上获得成功？

Web-OLAP 方法

一个潜在的能够成功实现的组合将是由 Web 技术、带有 OLAP 系统的数据仓库和一个瘦客户端体系结构来构成。你将如何在这样的一个环境中执行 OLAP 呢？OLAP 系统将如何在你的支持 Web 的数据仓库中运行呢？什么类型的客户端和 Web 体系结构将产生一个令人满意的结果呢？你可以用三个不同方法来回答这些问题。

1．***浏览器插件***_在第一个方法中，你使用一些插件或浏览器的扩展工具。这种方法除了在客户端的配置上像一个瘦客户端外，其余的地方完全像是在 Windows 中运行的一个稍微有点变化的胖客户端。但是这样做在系统的支持会出现问题，而且使用这种方法系统升级起来也比较困难。

2．***预创建的 HTML 文档***_在第二个方法中，你提供预创建的 HTML 文档和用来寻找这些文档的导航工具。这些文档是分析操作的结果集。这个方法利用了 Web 技术和瘦客户端节约资源的优势，但是用户只能使用预定义的报表。这个方法缺乏随需即取的分析，而且用户不能进行典型的联机分析处理的操作。

3．***服务器上的 OLAP***_最好的方法是用服务器来进行所有的联机分析处理的操作，并在一个实际的瘦客户端信息接口上显示分析的结果。这个方法实现了 Web 和瘦客户端体系结构的节约资源的优势。同时，它还提供了一个和客户端机器无关的综合服务器环境。因为应用系统和逻辑都集中在服务器上，需要进行的维护是最小的。版本控制也是一致的，每个人共享一个相同的组件：服务器、元数据和报表。这个方法在生产环境中运行得很好。

OLAP 引擎的设计

当数据仓库能够支持 Web 并且 OLAP 操作的级别也提高了的时侯，OLAP 引擎的设计

的好坏就决定了你能否顺利地扩充系统的容量。在你为支持 Web 的数据仓库选择的 OLAP 产品中，OLAP 引擎的设计是非常关键的。当并行操作的用户的数量增加时，一个经过适当设计的引擎能够产生一个线性的性能曲线。让我们考虑一些选项：

对 RDBMS（关系数据库管理系统）的依赖_OLAP 完全依靠 RDBMS 来进行多维的处理和生成复杂的多个通路的 SQL 来访问汇总数据。合并、聚集和计算都是在数据库内进行处理的，这样做给支持 Web 的系统带来了严重的问题。因此有必要使用大量的临时表。创建、插入、结束、分配磁盘空间、检查用户权限和为每个计算的操作修改系统的表格，这些操作的系统开销是巨大的。五个并行操作的用户就可能使 OLAP 系统陷于瘫痪。

对引擎的依赖_这里引擎通过生成的 SQL 语句来访问汇总数据并在中间层执行所有分析。使用这种方法你会观察到两个问题。繁重的网络传输和对大内存的需求使这个方法无法令人满意。你也可以得到一个线性的性能曲线，但是因为没有用到 DBMS 的潜力，所以这条曲线可能是很陡峭的。

智能的 OLAP 引擎_该引擎有足够的智能来确定请求的类型和该请求最适合的执行场所。因为它的智能，引擎能够在引擎组件和 RDBMS 之间分配合并、聚合和计算等任务。用这种模型，你能够有逻辑地、有规例地分离出结果的显示、逻辑和数据层。因此系统处理是均衡的，网络传输是也最优的。当前，当并行操作的用户的数量增加时，这似乎是一个最好的方法，这种方法所获得的性能曲线有一个渐进的倾斜，而且仍然是线性的。

建立一个支持 Web 的数据仓库

让我们概括迄今为止我们讨论过的内容。我们感觉到了 Web 是如何改变每件事情，这包括数据仓库的设计和开发。我们理解 Web 技术和数据仓库是如何结合起来并显示出强大的优势的。数据仓库的主要目标是为它的用户提供各种信息，而 Web 使信息的提供变得更容易了。这是多么好的一个技术结合啊！现在你的数据仓库已经能够扩展到一个更大的用户范围了。

当我们用数据仓库的特性来匹配 Web 的特征时，我们发现我们不得不调整数据仓库的设计和开发方法，以便使数据仓库能够支持 Web。我们完成了大部分的任务。Web 改变了数据仓库的信息传送方式。在信息传送中如果包含更多的基于 Web 的信息传送方法，它就更容易使用，但这也不同于传统的方法。我们也讨论过了基于 Web 的信息传送机制，我们

也接触过和 Web 有关系的 OLAP，因此我们现在该做什么呢？我们现在考虑着准备建立一个支持 Web 的数据仓库。

数据仓库的本质

在 1999 年的中期，Dr.Ralph Kimball 使一个新术语“数据仓库”普及了起来，这个术语包括一个支持 Web 的数据仓库的概念。他断言数据仓库正占据着 Web 革命的中心舞台。他接着宣称，这就要求人们重新开始和调整他们在数据仓库上的想法。他的话是多么正确啊！

为了形成建立一个支持 Web 的数据仓库需要遵循的一些原则，我们首先要回顾一下数据仓库的本质，接着将利用这个知识来定义建立数据仓库过程中需要考虑的一些事项。在复习数据仓库的主要特性之前，请看一下图 16-5，它对数据仓库进行了一次主要的回顾。现在我们来回顾数据仓库的特性。下面是数据仓库主要特性的一个清单：

●它是一个完全的分布式系统，由很多独立的节点组成了整个系统。就像 Dr.Ralph Kimball 所说的，数据仓库没有一个中心。

●它是一个支持 Web 的系统；它超出了一个客户端/服务器系统的范畴。数据仓库和客户端/服务器系统在任务的分布和组件的安排上是很不一样的。

●Web 浏览器是信息传送机制的一个关键。系统可以通过远程浏览器来传送信息请求的结果。

●因为它的开放性，需要认真地关注它的安全问题。

●Web 支持所有的数据类型，包括文本、数字、图形、照片、声频和视频等更多的类型，而且数据仓库也支持很多数据类型。

●系统能够在合理的反应时间内向信息请求返回它得到的结果。

●用户接口的设计是极为重要的，它关系到用户能否容易地使用接口和在 Web 上有效地发布信息。和其它配置中的接口不同的是，Web 有个明确的方法来衡量用户接口的有效性。对点击流的数据进行分析你就能够知道该用户接口好坏的程度。

●数据仓库生来就应该成为一个包含一些小规模的数据集市的分布式体系结构。

●因为组件的位置安排是基于一个连接数据集市的“总线”体系结构，所以具有完全一致的维和标准化事实是重要的。

●Web 从来都不休息，因此用户也期望着你的数据仓库在所有时间内都能工作。

●最后要记住，要向所有的用户组开放你的数据仓库，包括企业内部和外部的员工、客户、提供商和其它商业伙伴。

对如何实现数据仓库的考虑

现在我们来讨论在实现一个支持 Web 的数据仓库过程中你需要考虑哪些因素。上面列出的每个特性要求你重新调整原来的实现原则。通过浏览这个特性清单，你了解了实现一个数据仓库需要考虑哪些事项。现在我们要突出其中的一些至关重要的考虑事项。

如果数据仓库将会广泛地分布，你将如何来管理它呢？你如何使这个体系结构中的所有组件协调工作？你是否感觉到，要是中间层配置没有什么东西的话它似乎不可能正常地工作。在实际网络中，许多相互连接的用户组可以使用不同的技术和不同的平台。你如何把这些组结合在一起呢？从什么地方开始呢？

请仔细地研究下面的观察报告：

●为了使分布的各单元具有一致的基本体系结构，要积极地采用多维模型作为基本的建模技术。

●使用数据仓库的总线体系结构。这种体系结构具有完全一致的维和完全标准的事实，有助于正确的信息的流动。

●在一个分布式环境中，谁能够使维和事实都一致？在前面几章中，我们已经讨论了使维和事实都一致的含意。它主要是指某物始终都有一个相同的定义。一个好的建议是让用户将一致的维和一致的事实集中起来。这不需要物理上的集中；逻辑上的集中就可以了。这种集中使这个数据仓库表面上有了一个中心。

●现在仍然存在一些问题：实际上由谁来保持维和事实的一致呢？答案取决于在你的环境中起作用的是什么？如果可行的话，将保持维和事实一致的任务分配给本地的用户组。每个组有责任提出维度或事实表的定义。

●那么，所有单元如何了解全部的维和事实的完整的定义呢？这对 Web 来说是很容易的事情。你可以将这些定义公布在 Web 上；接着它们就成为一致的维和事实的标准。

●你如何在物理上实现一致的维表和一致的事实表呢？维表在物理上经常被复制。此外，要看你的环境中什么是可行的。要在物理上完全集中所有的维表是不实际的，但一致的事实表很少被复制，因为和维表相比，事实表通常都显得很大。

●最后一个考虑。现在我们明白了，数据仓库是基于有可能不同的数据库技术的维和事实表的一个分布集。你如何确保这样的一个分布集合像一个整体来工作呢？这就是查询工具或报表编辑器在这样的一个分布配置中要做的事情。让我们假定一个远程用户要执行一个特殊的查询。工具必须建立和每个事实表的供应商的连接，获得查询的结果集，这些结果集受到了一致的维的约束。接着工具必须使用一个单通路的种类合并（sort-merge），在应用系统服务器中对所有获得的结果集进行组合。因为维都是一致的，所以这将产生正确的最终结果集。

将组件放在一起

在这个子节中，我们将复习各种组件，这些组件需要放在一起才能够组成支持 Web 的数据仓库。考虑一下下面列出的事项：

●数据仓库配置已经超出了客户端/服务器计算的范畴，因此通常的两层或三层技术是不够的。随着用户数量的急剧增长，必须能够毫无困难地增加新的服务器。因此，考虑采用分布组件的体系结构。

●随着用户节点的分散，你必须努力实现在客户端上的最小管理。类似 Java 的这种实际的瘦客户端技术可以提供一个零管理的客户端设置。

●客户端技术将是瘦客户端和完全客户端的结合。要确保能够实现元数据的综合。IT 和各种用户类型将从统一的元数据中得到好处。

●选择合适的数据库来支持这个分布式环境。因为你可能会使用 Java，在数据库中带有 Java 引擎的 RDBMS 会是有用的。

●在很多 Web 应用系统中，当一个会话的所有数据通过 HTTP 服务器送往某个浏览器的时侯，这个 HTTP 服务器将成为一个拥塞点。因此只有你实现了 CORBA 模型，你才能容易地对系统进行升级的。CORBA 为分布对象提供了计算能力和升级能力，因为服务器和客户端都是通过 CORBA 来通讯的。

●确信你已经对管理和维护给予了足够的重视。这应该包括对维度、维内的层次、事实和汇总的确认。汇总管理实现起来可能比较困难。

●Web 接口由一个浏览器、搜索引擎、组件、“推”技术、主页、超文本链接和下载的 Java 和 ActiveX applet 所构成。

●对支持 HTML 的工具可以通用地配置。然而，HTML 处理复杂的分析是非常麻烦的。因

此你要尽可能多地使用 HTML，而用 Java 或一些插件来进行复杂的特别的分析。

Web 处理模型

首先让我们看一个 Web 体系结构的配置图。请看一下图 16-6，它显示了 Web 体系结构的全面的安排。我们发现这个体系结构看起来比一个二层或三层的客户端/服务器体系结构更复杂，而且你需要额外的层来满足 Web 计算的要求。最低限度你也需要在浏览器客户端和数据库之间增加一个 Web 服务器。你也要注意那个防火墙，它将保护你的企业的应用系统免遭外部的侵扰。

这覆盖了全面的体系结构。看图 16-7，它显示了一个信息传送机制的模型。该模型指出了 HTML 如何转换成 SQL 查询语句，接着如何使用 CGI（公共网关接口）脚本将这些语句传送到 DBMS 上。这个模型显示了一些通过 HTML 页面来传送信息的组件。这个模型能够用来阐述其它的技术。

本章总结

●Web 计算在 20 世纪 90 年代处于统治地位，该技术与数据仓库存储相结合，产生了激动人心的结果。

●支持 Web 的数据仓库通过 Web 可以在用户间传送信息和协同工作。

●使数据仓库支持 Web，意即数据仓库应包括如下特性：例如信息“推”（push）技术，易用性，快速反应能力，无停工时间（不停工），可以以多媒体形式输出和可升级性等。

●基于 Web 的信息传送扩展了数据仓库的使用范围，提出了新的信息策略。

●OLAP 和 Web 技术的结合为用户产生了巨大的利益。

●由于 Web 具有开放的特性，所以要使数据仓库能够支持 Web，需要在实施上认真地下功夫。

思考题

1. 简要地描述支持 Web 的数据仓库的两个主要特性。
2. 因特网、企业内部网（局域网）和企业外部网怎样应用到数据仓库？
3. 用户对支持 Web 的数据仓库的期望是什么？
4. 怎样使用 Web 作为你的数据仓库的数据源？你能从 Web 上得到什么类型的信息？
5. 说出一个 Web 页面上的任意四种标准选项，假设这个 Web 页面是从数据仓库获取信息。
6. 为数据仓库建立支持 Web 的用户接口的四种通用技术是什么？
7. 为什么对支持 Web 的数据仓库，数据安全是其主要的关注内容之一？
8. 列出数据仓库的任意四个特性。
9. 说出让 OLAP 在一个支持 Web 的数据仓库里运行的任意两个方法。
10. 数据仓库总线体系结构是什么？在支持 Web 的数据仓库中怎样配置它？

练习题

1. 指出对或错：

a) 企业外部网拒绝外部商业伙伴访问公司的数据仓库。

b) 在数据仓库中的 Web 技术以多种信息格式向用户显示信息。

c) 支持 Web 的数据仓库的唯一目的是向用户提供更好的信息传送机制。

d) 超增长在支持 Web 的数据仓库环境中是个罕见的现象。

e) Web 促进了“自助式”访问。

f) 数据仓库必需是一个分布体系结构。

g) Web 技术使在数据仓库中增加更多用户变得更容易了。

h) Web 和其特性与数据仓库不兼容。

i) 由于信息接口的问题，支持 Web 的数据仓库不能同时使用“推”和“拉”技术。

j) 在一个支持 Web 的数据仓库中，标准的 HTML 或 XML 文件是通信的主要手段。

2. 在数据仓库中，基于 Web 的信息导致了“超增长”。讨论这个增长现象并描述怎样提供超增长。

3. 你的项目小组要通过因特网生成和传送所有的报表。为了实现你的数据仓库的所有

报表将采用 Web 技术，为这个行动创建一个计划。讨论所有的相关内容。

4. 支持 Web 的数据仓库没有所谓的中心，而是一个分布的环境。分析以上陈述的含意。从这个角度来看，维护“维”和“事实表”都需要考虑什么？

5. 作为项目小组的 Web 专家，写一份文档来列出你的支持 Web 的数据仓库需要考虑的事项。只需列出考虑事项来，不需要实现技术。

第十七章　数据挖掘基础

本章目标

●了解数据挖掘的确切含义并研究它的特性

●将数据挖掘和 OLAP 做比较，理解它们的区别与联系

●注意在数据仓库环境中进行数据挖掘的位置

●认真学习重要的数据挖掘技术，理解每种技术的工作原理

●研究不同行业的一些数据挖掘应用系统，了解适用于你的行业的应用系统。

人们大都听说过数据挖掘。大部分人知道这个技术和发现知识有关。一些人可能知道数据挖掘用在一些诸如行销、零售、信用分析和欺诈检测之类的应用上。所有人都模模糊糊地知道数据挖掘与数据仓库相关，尽管不知其所以然。数据挖掘几乎在每个行业中都有应用，从零售和行销到新产品的开发，从存货管理到人力资源管理。

数据挖掘有着各种定义，也许和提供商和支持者的人数一样多。一些专家的定义很全面，包括了从工具到技术，从简单查询机制到统计学分析。其它人则将定义局限于知识发现技术。数据仓库的使用，虽然不是一个先决条件，但必将实际地推动数据挖掘的发展。

为什么数据挖掘在越来越多的事务中得到应用呢？基本的原因在于：

●在当今世界中，一个组织在一周内所生成的信息，大多数人穷尽一生也读不完。用人力根本不可能研究、破解和翻译完所有的数据来寻找有用的信息。

●数据仓库将数据适当地转换和净化后储存到精心组织的数据结构中以共享这些数据。然而，数据的透明卷（sheer volume）使得任何人都不可能使用分析和查询工具来辨别有用的模式。

●最近，市场上出现了很多适用范围广泛的数据挖掘工具，而且这些工具正在逐步地成熟起来。

●计算能力是数据挖掘需要具备的基本能力。而现在并行硬件、数据库和其它强大的组件的

价格也变得让人可以接受了。

●正如你所知道的，为了一劳永逸，各种组织正在花大力气建立合理的客户关系。一些公司想要知道怎样才能向现有的客户卖出更多的东西。而另一些组织想知道哪些客户是长期具有价值的。另外一些公司想要知道其客户的自然分类以便针对这些分类提供相应的产品和服务。数据挖掘使这些公司能够在他们的客户数据中找到答案和发现模式。

●最后，你的公司很可能为了提高竞争力而采用数据挖掘技术，而你的公司的竞争对手可能已经在使用数据挖掘了。

数据挖掘是什么？

在提出数据挖掘的一些正式定义之前，让我们设法在一个行业环境中理解这项技术。就像其它决策支持系统一样，数据挖掘传送信息，图 17-1 显示了决策支持系统的演进。记录基本的核算数据是最早的方法，这就是决策支持系统的原始类型。下一个方法是数据库，它提供了更多有用的决策支持信息。在 20 世纪 90 年代，用户通过数据仓库连同查询和报表工具来获得他们需要的决策支持信息，这些数据仓库开始成为决策支持信息主要的、有价值的来源。而更多老练的分析员则会使用 OLAP 工具。到此为止，获取信息的方法是由用户引导的。但数据的透明卷使得任何人不能使用分析和查询工具来辨别有用的模式。例如，在销售分析中，几乎不可能通过查询和下钻数据仓库的办法来考虑所有可能的联系并了解其内在的关系，因此你需要一项技术，它能学习过去的联系和结果并预测客户的行为。你需要一个自己能完成知识发现的工具。你想要的是一个数据引导的方法而不是用户引导的方法。这就是数据挖掘要插手并从用户那里接管过来的地方。

在这个领域领先的组织会以操作型系统为源头，从那里收集企业的数据，传输并净化数据，并将数据按某种格式存入数据仓库，以便进行多维分析。数据挖掘将会把这一过程再向前推进一大步。

定义数据挖掘

打个比方，想象在一个很深很宽的深渊里密密地放满了一些重要的原材料。你用一套复杂精密的钻探工具来挖掘和凿开这些东西。准确来说，你并不知道你通过努力想要获得什么。

你可能什么也发现不了，或者可能幸运地找到一些真正的天然金块。你可能会发现那里有一个你从来不知道的有价值的矿藏。你不是特意在寻找金块，你也不知道它们在那里或它们是否存在过。图 17-2 大致描述了这个设想。

现在，换一下场景，用你的数据仓库代替这个很深很宽的深渊。用你的数据仓库中的大块的数据内容代替深渊中的原材料，用数据挖掘工具代替钻探工具。天然金块是若干条有价值的信息，就像模式或关系，你从来不知道它们存在于数据中。事实上，你已经应用数据挖掘工具来发现有价值的东西，虽然先前你不知道这些东西的存在。这是数据挖掘的一个方面。数据挖掘和知识发现是同义的——发现知识的某个方面，你从未考虑过该方面的存在。

如果你不知道存在着某个模式或关系，你怎样引导数据挖掘工具来找到它呢？例如在某个抵押银行中，数据挖掘工具怎样知道是否存在着这样的信息金块能指出大部分拖欠抵押的户主属于某一确定种类的客户？

如果知识发现是数据挖掘的一个方面，那么预测就是另外一个方面。假设此时你正在寻找某一事件或情况的特定联系。你知道经过适当游说，你的一些客户可能会购买一些高档产品。你想预测客户购买高档产品的倾向。你的客户数据可能包含购买高档产品的倾向和年龄、收入水平和婚姻状况之间的有趣联系。你想找出导致客户购买高档产品的倾向的原因并预测你的哪些用户可能会提高购买模式。可见，预测是数据挖掘的另一方面。

那么，什么是数据挖掘呢？它是一个知识发现的过程。数据挖掘能以一种出人意料的特殊方式帮助你理解数据的本质。它从原始数据中挖掘出你所从不知道其存在着的一些模式和倾向。

Joseph P.Bigus 在他的书《数据挖掘和神经网络》书中写道，数据挖掘是从一个大的数据聚合中有效地发现不明显却有价值的信息。数据挖掘以新事实和数据关系的自动发现为中心。用传统的查询工具，你只能查找已知的信息。假定更多的有用信息是隐藏着的，而数据挖掘工具能够使你揭开这些隐藏的信息。

知识发现过程

在上述讨论中，我们描述说数据挖掘是知识发现的过程。数据挖掘通过你的数据揭示出你所从不知道的知识或信息。那么这些知识是什么?如何揭露这些知识呢？通常揭开的隐藏信息是以关系或模式的方式出现的。用户要设法去理解发现的关系或模式的种类。

关系举个例子，你要在回家的路上去附近的一家超市挑选面包、牛奶和其它一些东西。其它东西是什么呢？你不能确定。当你拿起牛奶罐时，你碰巧看到旁边的一包混合干酪。是的，你想要那个。你停下来，看身后的 5 位顾客。使你惊奇的是，你注意到那些顾客中也有三人要去买干酪。巧合吗？在五分钟的一段时间里，你已经看到了牛奶和干酪被顾客一起购买。现在来到了面包架。当你拿到面包时，一袋炸土豆片吸引了你的注意。为什么不要那包炸土豆片呢？而你身后的顾客也想要面包和炸土豆片。巧合吗？未必。可能这家超市是使用数据挖掘的全国连锁店的一间。数据挖掘工具已经发现了面包和炸土豆片，牛奶和干酪包之间的关系，尤其是在傍晚的高峰时间。因此这些东西被故意放在靠近的地方。

数据挖掘能够发现这种类型的关系。这种关系可以在两个或更多的不同对象之间，并能随时间的变化而一起发生。有时，关系可以发生在同一对象的不同属性之间。无论这些关系是什么，关系的发现都是数据挖掘的一个关键结果。

模式模式发现是数据挖掘的另外一个成果。以信用卡公司为例，该公司试图发现能够保证信贷限额增长或信用卡升级的使用模式。他们想知道哪些用户会被信用卡升级所吸引，和用户何时会升级信用卡。而数据挖掘工具能够挖掘出几千个信用卡持有者的使用模式，发现潜在的使用模式，这些模式将在未来的销售中产生巨大的效果。

在你从事数据挖掘之前，你必须清楚地确定你想要该工具完成什么任务。在这个阶段，我们不是试图预测你期待发现的知识，而是定义这项工作的商业目标。让我们来看看这些主要的阶段和步骤。首先看图 17-3，它指出了四个主要的阶段，接着请阅读以下的对这些细节步骤的简要描述。

*步骤 1：定义商业目标。*这个步骤和任何信息系统项目的步骤是相似的。首先，确定你是否真的需要一个数据挖掘的解决方案。接着陈述你的目标。你是否正期待着改善你的直接销售？你是否想要检测信用卡使用上的欺诈行为？你是否正在寻找产品销售之间的联系？在这个步骤中，详细说明你的期望。表达出在操作型系统中怎样表示和使用最终结果。

*步骤 2：准备数据。*这个步骤由数据选择、数据预处理和数据转换组成。从数据仓库选择将被抽取的数据，并且使用商业目标来决定选择什么数据。和选中的数据有关的适当的元数据也应该包含进来。现在，你也知道了你将使用什么类型的挖掘算法。挖掘算法影响数据的选择。为数据挖掘选择的变量也被称为活动变量（active variable）。

预处理意味着要改善选中的数据的质量。当你从数据仓库中选择数据时，我们假定这些数据

已经经过净化了。预处理也包括用外部数据来丰富选中的数据。在预处理的子步骤中，去除掉噪音数据，就是说，噪音数据不在使用范围之内。同时也要确保没有漏掉的值。

显然，如果被挖掘的数据都是从数据仓库中选来的，那么可以认为所有必要的数据转换都已经完成了。应该确保事实确实如此。

*步骤3：进行数据挖掘。*显然，这是一个关键的步骤。知识发现引擎将选中的算法应用到准备好的数据上。这个步骤的输出结果是一个关系集或模式集。然而，这个步骤和下个步骤的评估可以被重复执行。经过一个初始的评估，你可以调整数据，重复这个步骤。这个步骤的持续时间和重复密度取决于数据挖掘应用程序的类型。如果你正在分割数据库，就不需要太多次的重复。如果你正在创建一个可预测的模型，那么在用实际的数据库测试它之前，应该再三地建立这个模型并用样品数据测试它。

*步骤4：评估结果。*你实际上正在搜寻你感兴趣的模式或关系。这些有助于理解你的客户、产品、利润和市场。在这些选中的数据中，有许多隐藏的模式或关系。在这个步骤中，你会检查所有模式结果。你将应用一个过滤机制，只选择出希望要的模式来表示和应用。此外，这个步骤也依赖于所采用的数据挖掘算法。

*步骤5：表示发现。*知识发现的表示可以采用可视化导航、图表、图形或自由形态的文本等格式。表示也包含了为重复使用而储存在知识基础上的你感兴趣的发现。

*步骤6：发现的混合使用。*任何数据挖掘的目标是理解各种事务，辨别出新的模式和可能性，并将这些理解转化为行动。这个步骤是利用上一步所得的结果来创建有价值的商业项目。你要通过最好的方法来整合所发现的结果，以便能利用它们来改善经营情况。

这些主要阶段和它们的步骤细节显示在图 17-4 中。请仔细地研究该图，注意每个步骤。也要注意在步骤中使用的数据元素。该图阐述了知识发现的全过程。

OLAP VS 数据挖掘

在读过 OLAP 那一章之后，你现在一定是那个专题的专家了。正如你所知道的，有了 OLAP 查询和分析，用户就能够从复杂查询中获得结果，得到他感兴趣的模式。虽然数据挖掘也能使用户发现他感兴趣的模式，但在获得结果的方法上与 OLAP 有本质的区别，参考图 17-5。这个简单的图表显示了两者的基本的不同之处。

当分析员在一个分析会话中使用 OLAP 时，他或她对正在寻找的东西有一定的预先认

识。分析员是在经过深思熟虑后才带着设想开始进行分析的。但是在数据挖掘的情况下，分析员对可能会产生的结果没有预先的认识。在 OLAP 查询中，是用户起着引导的作用。每个查询可能导致另一个更复杂的查询，因此用户需要对结果有一定预先的了解。这个过程与数据挖掘完全不同。OLAP 有助于用户分析和了解过去的情况，然而数据挖掘有助于用户预测未来的情况。为了详细解释上述内容，图 17-6 列出了这两种方法所能够分别回答的一些问题。

注意 OLAP 是怎样基于过去表现而给出问题答案的。当然，从这些答案中你能很好地理解过去发生了什么。你可以从这些有关过去表现的答案中推测出未来的情况。与之对应，你要注意数据挖掘能做什么：它能发现特定的模式和关系，预测未来。

我们曾经说过 OLAP 是用来分析过去的，而数据挖掘是用来预测未来的。你一定在推测，一定有更多的关于数据挖掘的描述，而不仅仅是这个泛泛的描述。那么现在让我们来看看数据挖掘不同于 OLAP 的所有方面。为了得到 OLAP 和数据挖掘之间区别的完整清单，请参考图 17-7。

在另外一个意义上，OLAP 和数据挖掘是互补的。你可以说是数据挖掘捡起了 OLAP 不要的东西。在使用 OLAP 工具时，是由分析员来推动过程的发展。然而在数据挖掘中，分析员准备好数据后什么都不用做了，这时是由 OLAP 工具来推动过程的发展。

数据挖掘和数据仓库

企业数据仓库要么是作为一个集中的存储库为独立的数据集市提供数据，要么是作为相似的数据集市在总线结构上的联合，它为数据挖掘形成了一个非常有用的数据源。它包含了从各种源操作型系统抽取来的所有重要数据。这些数据已经经过净化和转换，并储存在你的数据仓库中。

数据挖掘非常适合于数据仓库环境，并在其中起着重大的作用。一个清洁和完整的数据仓库是数据挖掘的基础，没有数据仓库就不能进行数据挖掘的操作。这两种技术是互相支持的。以下是两者之间关系的一些要素。

●数据挖掘算法需要大量的数据，在细节一级上更是如此。而大多数数据仓库包含有最小粒度级别的数据。

●数据挖掘适用于经过综合和净化的数据。如果能正确地实现 ETL 功能，那么你的数据仓

库中的数据就是非常适合数据挖掘的。

●数据仓库的基础设施具有并行的处理技术和强有力的关系数据库系统，已经是非常健壮的了，由于其硬件可升级，所以不需要新的投资来支持数据挖掘。

我们来分析一下传统分析方法和数据挖掘方法在使用数据仓库数据时的不同之处。当分析员想要进行一个分析时，假定使用 OLAP 工具，他或她从高级别的概括数据开始，接着依靠下钻技术穿过一些较低的级别。许多情况下，分析员不需要向下到达细节所在的级别。这是因为他或她在较高的级别上就找到了合适的子集，得出了结论。但数据挖掘不同。当数据挖掘算法正在搜寻倾向和模式时，它涉及大量的细节数据。例如，如果数据挖掘算法正在寻找客户购买模式，它当然需要客户级别上的细节数据。

那么折中的方法是什么呢？在数据仓库中你需要提供的粒度级别是什么？除非在最低粒度级别上保存细节数据的负担很大，否则我们要尽量储存细节数据。再者，为了进行数据挖掘，你可能还要从操作型系统中直接抽取细节数据。这要求增加数据合并、净化和转换的额外步骤。你可能还要在数据仓库中为传统的查询保留轻度概括的数据。大部分各种维度的概括数据都可以放在 OLAP 系统中。

对数据挖掘来说，数据仓库是一个容易进入的有价值的数据源。数据挖掘工具所抽取的数据来自于数据仓库。图 17-8 阐述了怎样在数据仓库环境中配置数据挖掘工具。注意数据仓库环境是怎样支持数据挖掘的，注意数据仓库和 OLAP 系统内的数据级别，也要观察数据仓库的数据在知识发现过程中是如何流动的。

主要的数据挖掘技术

我们将要介绍数据挖掘技术了，很快我们意识到可以用很多方法来对这些技术分类。对数据挖掘不熟悉的人可能会被这些技术的名字和描述给搞糊涂了，甚至在从事数据挖掘的顾问中，似乎也没有一组统一的术语。即然没有统一的术语，那我们就用通俗些的词语吧。

很多从事数据挖掘的人似乎在用于特定应用领域的一套数据挖掘功能上达成了共识。每种功能可以应用各种数据挖掘技术。这些技术由可应用于各种功能的算法所构成。让我们再次回顾这些术语。但在此之前，请看表 17-9，它显示了数据挖掘的应用领域、挖掘功能的举例、挖掘过程和挖掘技术。请设法理解表中阐述内容的联系，并研究如下几点：

●数据挖掘算法是数据挖掘技术的一部分。

●数据挖掘技术被用来实现数据挖掘功能。当执行特定的数据挖掘功能时，你就是正在进行数据挖掘。

●一个特定的数据挖掘功能通常适合一个特定的应用领域。

●每个应用领域都是某行业的主要领域，数据挖掘在那里被积极地使用着。

我们将在这节的剩余部分讨论这些主要功能的重点部分、用于实现功能的过程和数据挖掘技术本身。

数据挖掘覆盖了范围广泛的多种技术。因为这不是一本关于数据挖掘的教科书，所以关于数据挖掘算法的细节描述不在它的讨论范围之内。在这个领域中有许多写得很好的书，你可以参考它们来满足你的兴趣。

在这里我们要探究一些较基础的东西。我们将选择六种主要技术来讨论。因为我们的意图是泛泛地理解这些技术，而不是深入地研究技术细节，所以你的主要目标是对数据挖掘技术有一个全面的理解。

聚类（cluster）

聚类（clustering）的意思是形成类簇。举个非常简单的例子——你怎样开洗衣店。你将衣服分成白的、深色的、浅色的、免烫的和要干洗的几组，这样你有了五个不同的类。每个类都有相同之处，你可以根据这个相同之处恰当地洗干净该类。聚类帮助你为不同的片（piece）采取特定的适当的行动，这些片组成了类。现在考虑在一个高级社区里的一家专业商店的业主，他想采购适当类型的产品来迎合附近的邻居。在经常光顾他的商店的人中，如果该业主有关于每个人的年龄和收入水平的数据，通过使用这两个变量，业主大概可以将顾客归入到四个类中。可以形成以下这些类：居住在高档社区中的富有的退休人员、度周末的中年的打高尔夫的人、富有的年轻的俱乐部成员和恰巧呆在社区里的低收入的顾客。和这些类有关的信息将有助于业主的销售。

聚类或者类检测是最早数据挖掘技术中的一个。这项技术被称为无指导的知识发现或无监督学习。通过上一句话我们想要说明什么呢？在类检测技术中，你不是去搜寻预先分类的数据，而且在类检测技术中，也没有自变量和因变量之分。例如，在商店顾客的情况中，有两个变量：年龄组和收入水平。两个变量都平等地参与了数据挖掘算法功能。

类检测算法搜索成组或成类的数据元素，这些元素相互之间是相似的。这样做的目的是

什么？你期望以相同的方法表现相似的顾客或相似的产品。接着你能选中一个类，用它做一些有用的事情。此外，在专业商店的例子中，商店业主能抓住富有退休人员这个类中的成员，瞄准他们感兴趣的产品。

注意聚类的一个重要方面：当挖掘算法产生一个类时，你必须理解那个类准确地表示什么意思。只有这样你才能用那个类做一些有用的事情。在上面的例子中那个商店业主必须能理解其中的一个类表示居住在高级社区里的富有的退休人员。只有这样商店业主才能够用那个类做些有用的事情。辨别数据挖掘算法形成的每个类的意思不总是那么简单。一个银行可能会有二十个类，也许只能解释其中两个的意思。但该银行只使用这两个类所得到的回报可能就已经足够大了，以至于可以忽略其它十八个类。

如果只有两个或三个的变量或维的话，即使是这个类涉及了很多条记录，辨别出这个类也是相当简单的。但如果正在处理来自 100，000 个记录的 500 个变量时，你就需要一个特殊的工具了。数据挖掘工具怎样执行聚类功能呢？不用深入太多的技术细节，让我们来研究这个过程。首先是一些基础。如果你有两个变量，那么在一个二维图形中，在表示这两个变量的坐标上画点。请参考图 17-10，它显示了这些点的分布。

让我们考虑一个例子。假设你想要数据挖掘算法形成你的客户的类，但你想要算法使用 50 个不同变量来对每个客户进行表示，而不只是用两个变量。现在我们正在讨论一个 50 维的空间。想象每个客户记录下的这 50 个维有不同的值，那么每个记录形成了一个 50 维的向量，这个向量也就定义了 50 维空间上的一个点。

让我们假定你想向这些客户销售产品，并且你准备分 15 个不同的组进行销售活动。因此你设定了类的数目为 15。这个数目是 k-means 方法聚类算法中的 k，用这个算法来进行类检测是非常有效的。经过精心的推测，选出十五个初始记录（称为“种子”），每个记录作为一个质心，形成一个集。每个种子代表从客户记录中选出的 50 个变量的一套值。下一步骤是，用算法将数据库中每个客户记录分配到与之最接近的种子所代表的类中。接近程度是指一个记录中 50 个变量的设定值和种子记录的值的接近程度。15 个类的第一个初始集形成了。接着算法计算质心或计算第一个集 15 个类中每个类内的平均值。在每个质心中 50 个变量的值用来表示相应的类。

接着开始重复的过程：每个客户记录重新和质心的新集匹配，组的分界线被重新划定。经过了若干次的重复过程之后，最终的类出现了。现在请参考图 17-11，它阐述了怎样决定质心和怎样重新划定组的分界线。

算法怎样重新划分组的分界线呢？是什么因素决定了一个客户记录靠近某一个质心而

不是其它质心呢？每种组检测算法都会采用不同方法，将不同记录的变量值和质心的变量值作比较，通过比较来计算各条客户记录与质心之间的距离。计算距离之后，算法就可以重新划定组的分界线了。

决策树

这项技术应用于分类和预测。决策树之所以吸引人，主要是因为它的简单。沿树而行，你就能破解规则，理解为什么一个记录用某个方法来分类。决策树表示规则。你可以使用这些规则来找到属于某个类的记录。请看图 17-12，它显示了一个决策树，用来表示男人和女人购买一台笔记本电脑的不同决策。

在某些数据挖掘的过程中，你并不关心算法是如何选择一个确定的记录的。例如，当你正在一个销售活动中选择销售的目标时，你不需要知道把他们作为目标的理由。你只需要能够预测出哪些人有可能对邮寄作出反应。但另一些情况中，预测的理由是重要的。如果你的公司是一家抵押放款公司并且想要评估一个项目时，你需要知道必须拒绝掉这个项目的理由，这样你的公司才能避免受到任何的歧视诉讼。无论什么情况下，只要需要给出理由并且需要跟踪决策路径，那么都适合使用决策树。

你已经看过了图 17-12，一个决策树表示一系列的问题。每个问题决定了继续下去的问题会是什么。好的问题会产生成为一个系列。决策树的根部在顶端而叶子在底部，这是一个不成文的规定。处于根部位置的问题必须是一个能最好区别目标种类的问题。当一个数据库记录进入树的根节点时，该记录将以它的方式向下运动直到遇到了某个叶子节点，而这个叶子节点就确定了这个记录的种类。

你怎样能衡量一棵树的效率呢？在笔记本购买者的决策例子中，你可以忽略已经知道分类的记录。接着你能计算已知记录的正确率。正确性高的树才比较有效。同时你也要注意分枝。一些路径由于其规则好而优于其它路径。通过修剪不合格的分枝，你能提高整棵树的预测有效性。

决策树算法是怎样建立这棵树的呢？首先，算法试图找到一种测试，它将以尽可能好的方式将记录分到各个分类中。从根开始，在每个较低级别的节点中，任何规则只要能最好地分离子集就将被采用。寻找树的每下一层的过程仍在继续着。树一直在增长，直到你找不出更好的方法来分离输入的记录。

基于记忆的推理

你愿意找一个有经验的医生还是找一个新手看病？当然，答案是明显的。为什么？因为有经验的医生是根据他或她的经验来治疗你的疾病。医生知道过去的几个与你相似的病例是怎样治好的。我们在根据经验制定决策时都做得很好。当前情况和过去经历事件有着相似之处，这是我们可以依靠的。我们该怎样使用经验来解决当前的问题呢？首先，我们确定有哪些过去相似的实例，接着我们利用那些实例，将有关那些实例的信息应用到现在的情况中。基于记忆的推理（MBR）算法也使用相同的方法。

MBR 使用一个模型的已知实例来预测未知的实例。这种数据挖掘技术有一个已知记录的数据集。算法知道这个训练数据集中的记录的特征。当一个新记录到达时，这个算法找到和新纪录相似的记录（称为“邻居”），接着使用邻居的特征来进行预测和分类。

当一个新记录到达这个数据挖掘工具时，首先该工具计算此记录和训练记录集的中各记录之间的距离。这个计算由数据挖掘工具中的距离函数来做。它的结果决定了训练数据集里的哪个数据记录有资格成为该数据记录的邻居。接着，算法使用一个组合函数来组合距离函数的各个结果，而获得最终的结果。距离函数和组合函数是基于记忆的推理技术的关键组件。

让我们通过一个简单的例子来观察 MBR 怎样工作。这个例子是根据已知测试者回答的数据集来预测新测试者最后读过的书。为了使这个例子变得简单些，假定当前有四本畅销书。被调查的学生已经读过这些书，而且也说出了他们最后读的是哪本书。四个调查的结果显示在图 17-13 中。看图的第一个部分。在那里你看见了已知测试者的分布图。表的第二个部分包含未知的测试者，他们在分布图中的某些位置上。从每个未知测试者在分布图的位置，你能确定他到已知测试者的距离，接着找到最近的邻居。最近的邻居预示着每个未知测试者读过的最后一本书。

为了用 MBR 解决一个数据挖掘问题，你需要关心三个关键的问题：

1. 选择最合适的历史记录来形成训练或基本数据集
2. 找出构成历史记录的最好方法
3. 确定两个必不可少的函数，也就是距离函数和组合函数

关联分析

这个算法对从关系中找到模式尤其有用。如果你近距离观察工商界，你会清楚地注意到所有类型的关系。定期航线将城市链接在一起，电话连接了人们，传真机与传真机相连，医师开处方与病人产生联系。在一个超市的零售交易中，顾客去一趟超市所购买的很多事务是联系在一起的。可见，在任何地方都存在着关系。

关联分析技术可以挖掘关系并发现知识。例如，如果你观察超市每天的零售交易的话，就会想到为什么脱脂乳和黑面包在一次交易中被同时购买的概率为 80%呢？超市篮子里的这两种商品之间是否有着强大的关系？假如这样的话，这两种产品能在一起推销吗？是否还有更多的这样的组合呢？我们怎样能找到这样的链接或密切的关系呢？

继续来看上文提到的另一个例子。对一个电话公司来说，它希望知道住宅客户是否已经拥有了传真机。为什么呢？因为如果一个住宅用户有了一台传真机，那么该用户可能想要再申请一根线路或者想要某种类型的升级服务。通过分析电话呼叫建立的两个电话号码之间的关系和其它一些条件，这个电话公司也能发现它想要的一些信息。关联分析算法可以发现这样的组合。根据不同的知识发现类型，关联分析技术可以分为三种类型的应用：关联发现、序列模式发现和类似的时序发现。让我们简要地讨论每种应用。

关联发现（associations discovery）_联系是指项与项之间的密切关系。关联发现算法能找出这样的组合，在这些组合中如果出现某一项，那么另一项也会出现。当你将这些算法应用到超市里的购物交易中时，它们将找出有可能被一起购买的那些产品之间的密切关系。关联规则表示了这样的密切关系，而算法能够系统地、有效地得到这些关联规则。请看图 17-14，它显示了一个关联规则和这个规则的注释部分。这两个部分－支持度因素和置信度因素－指出了联系的强度。具有较高支持度和置信度因素值的规则是更有效、相关和有用的。由于它的简单性，关联发现成为一个流行的数据挖掘算法。它只有两个因素需要解释，甚至这两个因素往往也是直观的。因为当每次读数据集时如果增加了新的维，这项技术都必须计算这两个因素的结合，因此测量成了一个主要问题。

序列模式（Sequential Pattern）发现_正如它的名字所显示的意思，这个算法将发现这样的模式：一个项目集跟着另一个特定的项目集。时间在这种模式中起了作用。当你选择记录来分析时，你必须有日期和时间这些数据项，来激活顺序模式的发现。

假定你想用这个算法了解产品的购买顺序。零售交易为数据挖掘操作形成了数据集。零售交易中的数据元素可以由交易的日期和时间、交易中所购买的产品和购买这些产品的顾客的标识组成。这些交易的一个范例集和应用这个算法的结果显示在表 17-15 中。注意顺序模式的发现，同时也要注意那个支持因素，它指出了联系的相关性。

人们可以发现以下这些典型的联系：

●购买了数码相机的人有 60%的概率会紧接着买彩色打印机

●购买了桌上型电脑的人有 65%的概率会紧接着买磁带备份驱动器

●购买了窗帘的人有 50%的概率会紧接着买客厅家具

类似的时序发现该技术依赖于时序的可用性。在前几项技术中，它们得到的结果显示出了一些时间上连续的事件。然而这项技术是先找到一个事件顺序，接着再提出其它类似的事件顺序。例如，在零售部门商店，这项数据挖掘技术分析了一个部门的零售流之后，能够找出其它的零售流与之类似的部门。这项技术还可以应用于找到价格连续变化情况相似的股票。

神经网络

神经网络可以模仿人的头脑，通过向一个训练数据集学习和应用所学知识来生成分类和预测的模式。在数据是不定形的并且没有任何可显示的模式的情况，这种算法是很有效的。人工的神经网络的基本单元模仿了人脑的神经元。这个单元被称为节点，它是神经网络模型的两个主要结构之一。另一个结构是链接（link），相当于人脑中神经元之间的连接。请看图 17-16，它描述了一个神经网络。

让我们通过一个简单的例子来理解一个神经网络是如何做出预测的。神经网络在输入节点处接收变量值或预测值。如果有 15 个不同的预测值，那么就有 15 个输入节点。预测值还要经过适当的加权（weight）。现在请看图 17-17，它显示了一个神经网络的工作情况。神经网络中可以有若干个内部层对预测值进行操作，从一个节点到另一个节点，直到发现的结果在输出节点上表示出来。内部层也被称为隐蔽层，因为随着输入的数据集经多次反复的处理，内部层一次又一次地改头换面地重复着这些预测值。

遗传算法

在某种程度上，遗传算法和神经网络有一些共同之处。这项技术也有一定的生物学基础。根据生物学，进化和自然选择决定了适者生存。随着一代又一代的演化，该过程将适者个体中的基因材料从一代遗传给下一代。遗传算法将相同的原则应用到数据挖掘中。这项技术使用一个“选择——交叉（cross-over）——突变运算”的多次反复的过程来使模型一代代地进化。在每次反复中，每个模型通过从前一个模型继承来的特性和其它的竞争，只有最好的预测模型能够保留下来。

现在我们要通过一个常见的例子来理解遗传算法中的进化过程。假设你想要解决如下问题：你的公司正在做一个有奖邮寄活动，并且想在邮寄中包含免费的优惠券。记住，这是个有奖的邮寄，目的是提高利润。同时，邮寄的回报率不能过低。问题是：在每个邮件中放置多少优惠券才能产生最大的利润？

乍一看，好像答案是在邮件中放尽可能多的优惠券。这样是否顾客就会用完所有的优惠券，而公司的利润达到最大化？然而，一些其它因素似乎使问题变得复杂了。首先，邮件中的优惠券越多，邮寄的成本将越高。增加的邮寄成本将耗掉一部分利润。第二，如果邮件中没有送出足够的优惠券，邮件中的每张优惠券就都不会被使用。这样就错过了机会并存在着潜在的收入损失。最后，如果一个邮件里放入太多的优惠券，那可能会使顾客感到厌烦，他或她根本不会使用任何优惠券。所有的这些因素都说明在邮件中放入的优惠券的数量要适中。现在看图 17-18，它显示了进化的前三代，该进化在这个问题上是应用遗传算法来表示的。

让我们来研究这个图。每个模拟的生物体都有一个基因，表示了生物体对邮件中优惠券数量的最好猜测。注意第一代中的四个生物体。对其中的两个生物体，基因或估计的优惠券数量是不正常的。因此，这两个生物体被删去了。现在剩下的两个存活的生物体重新产生了的具有类似它们基因的后代。再次提醒你：基因表示了一个邮件中放置优惠券的数量。在每一代中重新划定正常值，而进化的过程继续下去。在每一代中，合适的生物体存活下来，进化继续下去直到只有一个最终的生存者。这个最终生存者的基因就表示每个邮件中放置的优惠券的最适宜的数量。

当然，以上的例子是过分简单化的。我们没有解释在每一代中的数量是怎么产生的，也没有指出正常值是怎样划定的和你如何来排除不正常的生物体。执行这些功能需要复杂的计算。然而，这个例子使你对这项技术有了相当的了解。

进入数据挖掘

你现在有了足够的知识来朝着正确的方向学习，帮助你的公司进入数据挖掘并获得收益。那么初始步骤是什么？你公司应该怎样开始这项有吸引力的技术呢？首先，记住你的数据仓库将要为数据挖掘过程提供数据。无论你的公司计划用数据挖掘技术来做什么，数据源始终是你的数据仓库。在进入数据挖掘之前，一个合理可靠的数据仓库将使数据挖掘操作能在一个强大的基础上进行。

正如以前提到过的，只有在大量数据可用时，数据挖掘技术才能产生好的结果。几乎所有的算法都需要最低粒度上的数据。考虑在你的数据仓库中拥有细节级别的数据。另一个要点涉及到数据的质量。数据挖掘是从数据中发现模式和关系。挖掘‘肮脏’的数据会得到不准确的发现结果，而基于可疑的发现结果而采取的行动将产生重大的错误。数据挖掘项目会使项目成本升高，因此如果因为数据不够纯净而失败的话，你付不起这样昂贵的代价，所以要确保数据仓库里的数据具有高质量。

当你应用某项数据挖掘技术时，能够发现一些你感兴趣的模式和关系，这种情况是很好的。但你的公司将怎样来处理这些发现呢？如果发现的模式和关系不值得使用的话，那就白白浪费了精力，所以在着手一个数据挖掘项目之前，要清楚地了解你期待解决的问题类型和期待获得的收益类型。在确定了目标之后，接下来做什么呢？你需要一个方法来比较一些数据挖掘算法和选择最适合你的特定要求的工具。

在先前的章节中，我们谈到了主要的数据挖掘技术。你学习每个不同的技术，了解了这些技术是怎样工作和怎样发现知识的，但是你每次只涉及到一项技术。我们是否能有个框架来比较这些技术？是否能有个比较方法来帮助你选择数据挖掘工具？请看表 17-19。

这些模型结构涉及到怎样来认识技术，而没有涉及到实际上是如何实现的。例如，一个决策树实际上可以通过 SQL 语句来实现。在这个框架中，它的基本过程是某种数据挖掘技术的执行过程，例如决策树执行在决策点分离的过程。一项技术如何使这个模型有效是很重要的。在神经网络系统的技术中，没有一个有效的方法来确定发现的结束，但是这个模型要求通过不同层的节点来处理输入的记录，并在输出节点结束这个发现。

当你正在寻找工具时，支持多项技术的数据挖掘工具是值得考虑的。你的组织目前可能不需要具有多项技术的复合工具。那么一个多任务的工具更合适。此外，很多数据挖掘分析

员想要使用若干项技术来交叉验证发现的模式。现在市场上比较容易找到的工具包含了以下技术：

●聚类检测

●决策树

●关联分析

●数据可视化显示

在我们阅读一个选择数据挖掘工具的标准细节清单之前，让我们首先对工具选择有一个一般但是必不可少的了解。请仔细考虑这些提示：

●工具必须能够很好地和你的数据仓库环境结合，能接受仓库的数据并能与整个元数据框架兼容。

●发现的模式和关系必须尽可能的精确。发现不稳定的模式比根本没有发现任何模式更危险。

●在大多数情况下，你可能需要解释模型的工作，知道结果是怎样产生的。工具必须能够解释规则和如何发现模式。

让我们来看一个评估数据挖掘工具的标准清单。该清单可能有遗漏之处，但它覆盖了一些必不可少的要点。

数据访问_数据挖掘工具必须能够访问诸如数据仓库的数据源，并快速地将要求的数据集带到它的环境中。你可能经常需要其它数据源的数据来补充从数据仓库抽取出来的数据。所以工具必须能够读其它数据源和各种输入格式。

数据选择_在选择和抽取用来挖掘的数据时，工具必须能够根据各种标准来执行它的操作。选择功能必须包括对不需要数据的滤除和从现存的数据项中获得新数据项。

对数据质量敏感_因为其重要性，所以再次提及数据质量。数据挖掘工具必须对它挖掘的数据的质量敏感。工具必须能够认出遗漏的或不完整的数据，并弥补这个问题。工具也必须能产生错误报告。

数据可视显示_数据挖掘技术处理真实的数据卷，会大范围地产生结果。如果不能用图画和图表来显示结果的话，那么该工具的价值就大大减少了。所以要选择带有好的数据可视显示能力的工具。

扩展性_工具的体系结构必须能综合数据仓库管理和其它诸如数据抽取和元数据管理之类的功能。

性能_工具必须提供和要挖掘的数据量无关的一致性能，如应用的特定算法、指定变量的数

目和要求的精确级别。

可升级_数据挖掘需要用大量的数据来工作发现有意义的、有用的模式和关系。因此要确保该工具能够升级以处理巨大的数据量。

开放性_这是个吸引人的特性。开放性是指能与环境或其它类型的工具结合。寻找能够连接外部应用程序的工具，外部应用程序中的用户能从其它的应用程序中获得对数据挖掘算法的访问。工具必须能够和桌上型电脑工具共享输出，例如图形显示、电子数据表和数据库设备。开放的特性也必须包括主流的服务器平台上的工具的可用性。

算法组_选择一个能提供一些不同算法的工具，而不要选择只支持单一数据挖掘算法的工具。

数据挖掘应用程序

你将发现大范围的各种应用程序会从数据挖掘中受益。科技围绕着覆盖了商业和非商业领域上的大范围应用的众多改良技术而发展。某些情况下，结合使用多种技术会获得更大的利益。你可以应用一个聚类检测技术来识别客户类。接着你可以使用一个预测算法，将它应用到一些经过识别的类上，发现那些类中的客户的预期行为。

数据挖掘在非商业领域中的应用是功能强大，它在搜寻领域被广泛地应用着。在石油勘探中，数据挖掘技术可以发现适合钻探的位置——潜在的矿物和石油的沉积物所在之处。模式发现和匹配技术在军事应用上用来帮助鉴定目标。医学研究对数据挖掘来说是一个成熟的领域。数据挖掘技术有助于研究者发现疾病之间的相互关系和潜在特征。犯罪调查机构使用该技术来将罪犯的特征和罪行联系起来。在天文学和宇宙学上，数据挖掘帮助预测宇宙中发生的事件。

在这个科学主宰的社会中，数据挖掘的应用程度还不算太高，但在商业领域这项技术却是在非常普遍地应用着。大多数的工具是以商业部门为目标的。请看以下的清单，它列出了数据挖掘在商业领域的一些主要应用：

客户细分_这是最普遍的应用之一。商业部门使用数据挖掘来更好地理解顾客的行为。类检测算法可以找出具有相同特征的客户类。

市场篮子分析_对零售来说这是一个非常有用的应用。关联分析算法找出被一起购买的产品之间的密切关系。其它的诸如高档商品房拍卖的交易使用这些算法来寻找客户，对这些客户他们可以卖出更高价值的项目。

风险管理_保险公司和抵押交易使用数据挖掘来发现和潜在客户有关的风险。

欺诈检测_信用卡公司使用数据挖掘来发现不正常的客户花费模式。这样的模式能暴露信用卡的欺诈使用。

拖欠跟踪_贷款银行使用该技术来跟踪可能拖欠还款的客户。

需求预测_零售和其它商业使用数据挖掘来匹配需求和供应趋势，预测对特定产品的需求。

数据挖掘的收益

到现在你已经确信了数据挖掘技术的能力和有用性。没有数据挖掘，在很多组织中的大量数据里隐藏的有用知识就不会被发现出来了，也就不能通过找出的模式和关系而获得收益了。这些收益的类型是什么呢？我们已经接触了数据挖掘的应用，你也掌握了它潜在的益处。

让我们列举出一些收益的类型来感受一下数据挖掘所产生的巨大的效用。请浏览以下条目，它指出在现实世界情形下可实现的收益类型：

●在一个生产生活消费品的大公司里，如果成品部在订单中有规律地发生短货现象，或隐藏定购单和运货单之间的不同的话，那么只要采用数据挖掘，就可以通过发现订单模式和过早的存货丢失的模式，检测出犯罪行为。

●一个邮购公司通过选定更多的目标公司可以改善邮件推销的效果。

●一个超市连锁店找出了某些产品被一起售出的密切关系，重新整理货架，改善了收入。

●一家航空公司通过发现频繁搭乘飞机者的旅行模式，提高了对商业旅行者的销售额。

●一家百货公司预料到了会发生抢购，所以增加了专门的销售。

●一个国家的卫生保险提供者通过检测索赔欺诈，节省了大量的资金。

●一家主要的带有投资和商业服务的银行公司提高了直接销售活动的平均值。预测模型算法发现具有终身价值的客户类。

●一家柴油机制造厂通过汽车登记的历史数据发现了某种模式，从而预测柴油机的销售，提高了销售额。

●一家银行通过检测到它的常用帐户磨损的早期警告信号，避免了损失。

●一家目录销售公司通过预测哪些顾客将使用假期目录，使得假期的销售比去年相比翻了一番。

在零售业的应用

让我们简要地讨论一下零售业如何来使用数据挖掘技术并从中受益。激烈的竞争和有限的利润率已经使零售业倍受折磨了。在这些因素的压迫下，零售业比其它大部分产业更早地采用了数据仓库存储数据。经过了若干年，这些数据仓库已经收集了巨大容量的数据。在很多零售企业中的数据仓库已经成熟了。此外，通过扫描仪和收银机的使用，零售业已经能够捕获销售数据的细节点。

大容量的数据和低粒度的数据这两个特性的结合，对数据挖掘来说是很理想的。当其它产业只是在制定计划时，零售业已经开始使用数据挖掘了。零售业中所有类型的生意，包括食品杂货连锁店、顾客零售连锁店和商品目录销售公司，广泛地使用直接销售和推销。直接销售在这个行业中相当重要，所有的公司都严重地依赖直接销售。

直接销售包括对特定客户类有目标地销售和推销。聚类检测和其它预测性的数据挖掘算法将客户分类。因为这是零售业的一个关键领域，许多提供商为客户分类提供了数据挖掘工具。这些工具能在后端和数据仓库结合用来进行数据选择和抽取。在前端，这些工具和标准的显示软件协同工作得很好。客户分类工具能够发现类，并预测直接销售活动的成功率。

零售业的推销活动有必要了解需要推销哪个产品和以什么结合方式来推销。零售商使用关联分析算法来寻找经常被同时售出的产品之间的密切关系。正如你所知道的，这是市场篮子分析。基于该密切关系分组，零售商能够计划出特殊的销售项，也能安排货架上产品的摆放。

除了客户分类和市场篮子分析外，零售商还使用数据挖掘来进行存货管理。一个零售商的存货包含成千上万种产品。存货周转率和管理对这些企业来说是意义重大和至关重要的。数据挖掘在零售业的另一个应用是销售预测。零售销售常遭受季节性的波动。节假日和周末也有影响。因此，销售预测对这个产业是很关键的。零售商求助于数据挖掘技术的预测性算法来做出销售预测。

在零售业中，数据仓库的其它类型的使用是什么呢？这个行业感兴趣和的问题和利害关系是什么呢？这儿是一个经过删节的清单：

●顾客长期开销模式

●顾客采购频率

●最好的推销类型

●宣传展览的储备计划和安排

●附有优惠券的邮件的设计

●购买特殊商品的顾客类型

●季节性的和常规性的销售趋势

●繁忙时刻的人力计划

●最有利可图的客户分类

在通信行业上的应用

数据挖掘应用下一个要查看的行业是通信。这个行业在 20 世纪 90 年代解除管制。在美国，便携式电话极大地改变了前景，虽然早些时候这股热浪已经袭击了欧洲和亚洲的一些小国家。为了顺应这个极端竞争的市场背景，许多公司抢着寻找方法来了解他们的客户。保持客户和获得客户已经成为它们销售中的最优先考虑的事项。通信公司相互竞争着设计、提供最好的产品，吸引客户。自然而然地，这个竞争压力将通信公司引向了数据挖掘。所有的主要公司已经采用了这项技术，并获得了很大收益。几个数据挖掘提供商和咨询公司专门研究该行业的问题。

客户吸收（customer churn）有着重大的利害关系。你一周几次接到该行业中的销售人员突然打来的电话？很多数据挖掘的提供商提供的产品包含客户吸收。最新的便携式电话（cellular phone）市场取得了最高的吸收率。一些专家估计获得一个新客户的花费大概是 500 美元。

如果通信网络有问题则埋伏着潜在的危险。在现今竞争的市场中，客户一但遇到小问题，就很容易受诱惑而转移。在这样的环境下很难挽留住原有的客户。一些数据挖掘提供商专门研究适合该行业的数据可视化产品。这些产品在网络地图上闪烁报警符号，用来指出潜在的问题领域，这使得负责的员工能够采取预防性的措施。

在通信行业，数据挖掘的应用是一个有用的帮助。以下是这个行业的问题和关注点的一个总清单：

●保持用户，尤其是当用户面对别的竞争者的诱人条件时

●通过客户行为分析出将来使用是否会有所增加

●发现有利可图的服务包

●对电话欺诈的预测

●向现有客户推销附加的产品和服务

●找出促使客户使用电话的因素

●和竞争对手做比较的产品评估。

在银行和金融业的应用

这是另一个行业，你将发现在其中频繁地使用了数据挖掘。银行业在过去的几年里已经规范化了。合并和兼并在银行业更是经常发生，并且银行已经扩展了它们的服务范围。金融业是一个波动且不稳定的领域。银行和金融业是数据挖掘的肥沃土壤。银行和金融机构生成了大量的细节交易数据。这样的数据适合用于数据挖掘。

数据挖掘在银行的应用是各式各样的。欺诈检测、潜在客户的风险评估、趋势分析和直接销售是数据挖掘在银行的主要应用。

在金融领域，预测的要求占了支配地位。用高水平的近似来预测股价和物价意味着大量的利润。对潜在的金融风险的预测被证实是很有价值的。神经网络算法被用在预测，股票预购和债券买卖，证券管理和合并与兼并等领域。

本章总结

●决策支持系统已经发展到了数据挖掘的阶段。

●数据挖掘，也是知识发现的一种，是数据引导的，反之其它的诸如 OLAP 的分析技术是由用户引导的。

●数据挖掘的知识发现过程揭露了关系和模式，甚至原先可能都不知道存在着这些关系和模式。

●六个步骤组成了知识发现过程。

●在信息的获得和发现上，可以认为 OLAP 和数据挖掘是不同的和互补的。

●数据仓库是数据挖掘操作最好的一个数据源。

●一般来说主要的数据挖掘技术有类检测、决策树、基于记忆的推理、关联分析、神经网络和遗传算法。

思考题

1. 给出三个主要的理由，来解释数据挖掘应用于现今商业领域的原因。
2. 用二到三句话定义数据挖掘。
3. 说出数据挖掘操作的主要阶段。从中挑出两个，描述在这两个阶段的活动的类型。
4. 数据挖掘和 OLAP 有什么不同？简要地解释。
5. 数据仓库是数据挖掘的一个先决条件吗？数据仓库是否有助于数据挖掘？假如这样的话，在哪些方面对数据挖掘有所帮助？
6. 简短地描述类检测技术。
7. 基于记忆推理（MBR）技术是如何工作的？它的潜在原则是什么?
8. 说出关联分析技术的三个普通应用。
9. 神经网络和遗传算法是否有相同的地方？指出它们的一些区别。
10. 市场篮子分析是什么？举出这个商业应用的两个例子。

练习题

1. 匹配以下各列：

1．知识发现过程	A．揭示发现的理由
2．OLAP	B．神经网络
3．类检测	C．距离函数
4．决策树	D．为挖掘提供数据
5．关联分析	E．数据引导的
6．隐蔽层	F．欺诈检测
7．遗传算法	G．用户引导的
8．数据仓库	H．形成组
9．MBR	I．高度重复的
10．银行业应用	J．关联发现

2. 作为一个数据挖掘顾问，你被一家提供很多金融服务的大商业银行聘用了。该银行已经有了一个数据仓库，它在两年前开始使用。管理层想要找出在现在的客户中哪

些人对新服务的销售活动会作出反应。描出知识发现过程的大致情况，列出阶段，并指出各个阶段中的活动。

3. 叙述决策树是怎样工作的。选择一个例子并解释这个知识发现过程是怎样工作的。
4. 遗传算法的基本原则是什么？举出一个例子。用这个例子来描述这项技术是怎样工作的。
5. 在你们的项目中，你负责数据挖掘的需求分析和工具集的选择。列出你用来选择工具集的标准。简要地解释每个标准的必要性。

第十八章　物理设计过程

本章目标

- 明确数据仓库的物理和逻辑设计的区别
- 研究物理设计详细过程
- 明白物理设计需要考虑因素及其含义
- 掌握物理设计中存储条件的角色
- 检验数据仓库环境下索引技术
- 总结所有提高系统性能的方法

作为 IT 专业人员，你一定已对逻辑模型和物理模型非常熟悉。可能也有过将逻辑模型转化为物理模型的实际经验。知道物理模型与操作平台、数据库、硬件和第三方工具的细节紧密相关。

通常为完成 OLTP 系统的物理建模需要进行大量的准备工作，逻辑模型是物理模型的基础。然而作为补充，要完成物理建模，必须考虑更多的因素。你必须考虑数据库对象的物理存储，存储介质是什么，有什么特点？这些信息将用于设置存储参数。接下来的一个重要环节是建立索引。每个表的哪个字段必须建立索引？而且，为了提高性能你还要寻找其它一些办法。必须检查 DBMS 的初始化参数并决定如何适当地设置这些参数。同样地，在数据仓库环境下，为完成物理建模你还要考虑很多不同的因素。

我们已经很详尽地探讨了数据仓库的逻辑模型。你已经掌握了维建模技术，它可以帮助你完成逻辑建模。本章里，我们将使用数据仓库的逻辑模型完成其物理建模。物理设计使项目组更接近于最后的实现和部署。物理设计将逻辑设计推向更有意义的下一步。

物理设计步骤

图 18-1 描述了数据仓库物理设计过程的各个步骤。这些步骤在下面将要被经常提到。在下面各小节中，我们广泛地描述了这些步骤中包括的各项任务。你会明白你正处在完成物理建模的过程中的哪个环节。本节后的其它小节详细描述了物理设计的各主要方面。

建立规范

许多公司投入大量的时间和金钱为信息系统制订规范。这些规范包括从怎样定义数据库的字段到如何跟用户交流以获得详细的需求分析。一个 IT 小组专门负责保持标准的更新。一些公司，每次版本的更新必须由 CIO 授权。通过规范制订小组，CIO 知道规范是否被严格和正确地更新。接下来是在公司的内部网上公布这些规范。如果你的 IT 部门能够非常敬业的维护这些规范，那么请你在构建数据仓库时积极采纳和利用这些规范。

在数据仓库环境里，规范还包括很多其它领域。因此规范要保证在不同领域的一致性。如果你使用同样的方式定义数据库对象的名字，就不会模棱两可了。例如公司规范指定对象的名字是由横线连接的多个单词构成，并且第一个单词表示业务的种类。根据这个规范，当你看到对象的名字，就立即知道了它所属的业务类型。

规范在数据仓库中起到很大的作用。因为对象名称不一定仅限于由 IT 部门使用。用户在阐述和运行他们自己的查询时也可能用到对象的名称。规范是非常有意义的，在本章的后面我们还会提到规范，下面进行物理设计的下一个步骤。

建立聚集计划

假设在你的环境中有 80%的查询是关于汇总信息的，如果数据仓库只存储最小粒度的数据，每次查询将遍历所有的明细记录，然后生成汇总信息。例如对所有商店按产品查询一年的总销售额，如果你按产品和商店记录每日的销售记录，就需要读入大量的原始记录。如何提高这类查询的性能呢？如果有了按仓库建立的产品汇总表，查询就会很快。但是应该建立多少汇总表，汇总表有怎样的限制呢？

这一步，可以考虑建立聚集表。从需求分析上找到蛛丝马迹，对每个维表检查继承层次

特性。哪些层次对于聚集更重要？进行评估和折衷，你所需要的是聚集的综合计划。计划必须指出为各个层次建立汇总的聚集类型。很多聚集最后都将在 OLAP 系统中体现出来。如果 OLAP 系统不是给所有人使用的，则必需的聚集将在主数据仓库中提供。聚集数据表必须包括在物理模型中。后面的章节将对聚集级别作更多的说明。

确定数据分区方案

考虑数据仓库中的数据容量，事实表究竟有多少行呢？让我们作一个粗略的估算。假设有 4 个维表，平均每个表有 50 行。对于这些维表中的行，潜在的事实表将有超过六百万行记录。事实表非常巨大。大表非常难于管理。在加载过程中，整个表必须对用户锁定。而且，备份和恢复大表也因为它们的容量而非常复杂。分区可以将大表分解成易于管理的小表。

考虑事实表的分区方法，并不是简单地分解数量。根据你所处的环境，真正需要确定的是如何精确地进行事实表分区。你的数据仓库可能是一系列数据集市的聚集。你必须为所有的事实表确定分区选项。是需要垂直分区还是水平分区？有时，你会发现你的某些维表也需要分区。产品维表就特别大。检查你所有的维表以决定哪些维表也需要分区。

这一步，我们需要给出分区方案。方案必须包括：

- 选择需要分区的事实表和维表
- 每个表的分区类型——水平或者垂直
- 每个表的分区个数
- 表的分区条件（例如，按照产品分组）
- 描述分区中的查询方法

建立聚簇选项

在数据仓库中，很多的数据访问是基于对大量数据的顺序访问。当有这类访问和处理时，就可以从聚簇中获得很大的性能提高。这种技术是将存取和管理相关的数据放在存储介质的相邻物理块上。这种安排使相关联的数据能够在一次输入操作中全部取出。

完成物理设计前要建立正确的聚簇选项。逐个检查表，找出相关表。相关表的行在很多情况下是同时被处理的，可以安排存储相关的表在相同文件和相同介质的相邻地方。例如对两个相关的表，可以让两个文件交叉存放为一个文件，一个表的一条记录后面跟着另外一个表的相关行的所有记录。

准备索引策略

这是物理设计中的至关重要的一步。不像 OLTP 系统，数据仓库是以查询为中心的。众所周知，索引可能是最有效的提高检索性能的机制。一个可靠的索引策略会给我们带来巨大的回报。策略必须包括为每个表建立索引的计划，指定需要索引的列。索引列的顺序在提高性能方面有很大的影响。仔细分析每个表以决定哪几列适合于使用位图索引。

准备一个综合的索引计划。计划必须包括每个表的索引。而且对每个表，要指明建立索引的顺序。描述在数据库实例化初期希望建立的索引。许多索引可以等到对数据仓库监测一段时间后进行。要在索引计划上多花一点时间。

安排存储结构

你想把数据放在具体的物理存储介质哪个位置呢？什么是物理文件？计划好了安排每个表到指定文件吗？怎样将物理文件分成数据块？所有这些问题的答案是数据存储计划。

在 OLTP 系统中，所有数据存在于可操作的数据库中。当你安排 OLTP 系统的存储结构时，你的努力仅限于被用户应用访问的可操作表。但在数据仓库中，你不仅仅要关心与数据仓库表相关的物理文件。你的存储计划还要包括其它的存储类型，例如临时数据解析文件，缓冲区和前台应用的存储。计划应该包括所有不同存储领域中的存储结构。

完成物理建模

最后一步是评价和确认前面所做的工作和完成的任务。到了这步，你已经有了数据库对象命名标准。已经确定了需要哪些聚集表，如何对大表进行分区，完成了索引策略，计划好了其它性能选项。你也知道哪里存放物理文件。

所有来自前面那些步骤的信息就可以完成物理建模。最终的结果是建立物理方案。可以使用选定的 RDBMS 的数据定义语言（DDL）进行编码，建立数据字典中的物理结构。

物理设计要点

我们已经逐一探索了数据仓库物理设计的各个步骤。每一步包含特定的操作以完成最后的物理模型。再回头看看这些步骤，其中有一步是和物理存储结构有关，其它几步是为提高数据仓库性能相关。物理存储和性能都是关键因素。本章后面将深入地探讨这两个方面。

本节我们将深刻理解物理模型本身的意义。回顾从逻辑模型转换为物理模型过程中的各组成部分和步骤。首先，让我们从物理设计过程的全部目标开始。

物理设计目标

对数据库进行逻辑设计，目的是为反应现实情况的信息内容建立一个概念模型。逻辑建模代表所有的数据组成和相互关系。物理设计过程目的并不着重于结构。在物理设计中，更接近操作系统、数据库软件、硬件和操作平台。这时更注重的是这个模型将怎样地运行起来而不是模型是什么样子。

总结一下，物理设计过程的主要目的：一是提高性能，二是更好地管理好存储的数据。物理设计是基于数据的使用。访问的频率、数据容量、选择的 RDBMS 支持的特性和存储介质的配置影响物理设计的最终结果。必须注意这些因素并且逐个分析，建立一个高效的物理模型。下面指出物理设计的几个关键目标。

提高性能　对于在线响应时间，OLTP 环境的性能和数据仓库有差距。OLTP 系统的响应时间强制性地必须少于三秒，而数据仓库没有这么苛刻的时间要求。依赖于查询时处理数据的容量，响应时间从几秒钟到几分钟都是合理的。即使用户知道它们的不同，目前的数据仓库和 OLTP 系统响应时间超过几分钟也是不能接受的。在这个层次上要努力提高性能缩短响应时间。确保数据仓库性能受到日常监视，总能够保持在最优化状态。

监视性能和提高性能必须在不同层次上发生。在最基础层次上，重点是操作系统的性能，上面一层是 DBMS 的性能。在这一层监视和提高性能是数据仓库管理员的事情。在更高层次上，逻辑数据库设计、应用程序设计和查询格式也对整个数据仓库性能有影响。

保证可伸缩性　这是一个关键目标。在过去一段时间内，数据仓库的使用逐步增强并且增长迅速。前面我们已经详细说到了这类快速增长。在快速增长期内，几乎不可能跟上使用的显著增长。

我们已经看到，数据仓库应用正以两位数递增。用户数量激增而且查询更加复杂。随着用户数量的增长，数据仓库的并发用户也成比例增长。要有合适的方法来满足用户使用数据仓库的快速增长。

存储管理　为什么说存储管理是物理设计的主要目标？好的数据存储管理将大大提高数据仓库性能。可以将相关的表存放在一个文件中，也可以将大的表分成几块存储以便于管理，还可以通过设置 DBMS 存储空间管理参数以优化文件块的使用。

提供简便的管理　这个目标覆盖所有使管理更加简便的各项工作。例如，包括正确安排表中的行以避免经常性重组。另外一方面是备份和恢复数据库表的简便。纵观不同数据仓库的管理任务，只要涉及到存储或者 DBMS，一定要做到易于管理。

弹性设计　在物理设计阶段，弹性意味着使设计更加开放。转换到数据模型后，可以更少地更改物理模型。物理设计必须有内建的弹性，以满足未来的需要。

从逻辑模型到物理模型　逻辑模型里有表、字段、主键和关系。逻辑模型包括的结构和关系，在数据库中使用 DBMS 的数据定义语言（DDL）进行编码。将逻辑模型转换为物理模

型需要做些什么呢？参照图 18-2。图中可以看到沿着转换过程箭头进行的各项工作。右侧最后的框代表物理模型。这是按照箭头执行操作后的最终结果。参考这些操作，将它进行改编以适用于你自己的数据仓库环境。

物理模型的组成

已经讨论了物理模型的一些基本概念以及在物理设计步骤中实现，现在我们浏览一下这些细节。物理模型代表的信息内容更接近于硬件层。这就意味着你必须仔细描述各种细节，例如文件大小、字段长度、数据类型、主键和外键，必须全部在模型上反应出来。首先看一下图 18-3，描述了物理模型的主要组成部分。逐一分析每个组件。这些组件是通过方案和子方案在 DBMS 的数据字典中描述出来。使用 DBMS 的数据定义语言（DDL）写出方案定义。图 18-4 给出一个方案定义示例。注意不同类型方案描述语句的不同。注意数据库、表和列定义。观察数据类型和字段大小是如何定义的。数据库管理员都非常熟悉这些方案定义语句。

将它们组合到一起。将逻辑模型和物理模型组件联系起来。图 18-5 提供了它们之间的组合视图。注意与图 18-4 的方案定义之间的联系。

```
CREATE SCHEMA ORDER_ANALYSIS
    AUTHORIZATION SAMUEL_JOHNSON
……
CREATE TABLE PRODUCT (
    PRODUCT_KEY CHARACTER (8)
        PRIMARY KEY,
    PRODUCT_NAME CHARACTER (25),
    PRODUCT_SKU   CHARACTER (20),
    PRODUCT_BRAND CHARACTER (25))

CREATE TABLE SALESPERSON(
SALPERS_KEY CHARACTER(8) PRIMARY KEY,
```

```
SALPERS_NAME CHARACTER(30),
TERRITORY CHARACTER(20),
REGION CHARACTER(20))

CREATE TABLE ORDER_FACT(
PRODUCT_REF CHARACTER(8) PRIMARY KEY,
SALPERS_REF CHARACTER(8) PRIMARY KEY,
ORDER_AMOUNT NUMERIC(8,2),
OREDER_COST NUMERIC(8,2),
FOREIGN KEY PRODUCT_REF REFERENCES PRODUCT,
FOREIGN KEY SALPERS_REF REFERENCES SALESPERSION)
```

图 18-4 方案定义 SQL 示例

规范的意义

数据仓库中的规范涵盖了很广的范围，包括对象、进程和过程。对于物理模型，对象命名规范有特殊意义。规范为交流提供了一致性的意义。项目成员之间必须要有高效的交流。在一个数据仓库项目中，最好有用户代表存在。用户从数据仓库中访问数据比 OLTP 环境更直接。和用户之间无障碍的交流也变得更有意义。

下面是规范中的一些小技巧。

数据库对象的命名

对象组件命名　　使用一种清楚的方法组合对象的名称。名称本身必须能包括对象本身的描述。例如，一个字段的命名：customer_loan_balance。这个名称立刻能让人明白这个字段包含余额数量。什么余额？贷款余额。是全部的贷款余额吗？谁的贷款余额数量？第一个单词说明这是客户的贷款余额而不是全部的贷款余额。对象名称由多个单词组成，通常就能将意思表达得清楚一些。可以将每个单词在名称中的功能定义规范化。在我们的例子中，第一个

单词代表主题，第三个单词是对象的一般类，而第二个是一般类的限定词。许多公司采用了这种类型的命名规范。可以强化公司正在使用的规范，使之更加简明清晰。

单词分界符　规范化分界符也叫做 delineators。横线或者下划线是经常用到的。如果你使用的 DBMS 有特殊的要求，请按照要求选择分界符。

逻辑模型和物理模型的命名　　对象的命名，例如表和字段，对应于逻辑模型和物理模型需要包括两个版本。必须为这两个版本制订命名规范。相对于用户群体，IT 专业人员用的是逻辑模型命名。分析员和逻辑模型设计者之间交流使用的是逻辑对象命名。而用户为了查询数据用到的表名和列名使用的是物理模型命名。所以，必须让用户适应物理模型的命名规范。最好的方法是让同一个对象命名的逻辑模型和物理模型版本相同。为了更清楚的命名可以为对象加上更多的限定词。在商业术语中别忘了声明这些定义。

缓冲区文件和表名称定义　数据仓库环境中缓冲区是最繁忙的地方，大量的数据交换发生在这里。从数据源系统中取出数据并在这里建立了许多中间文件，在这里进行数据的转换和加工。在缓冲区内准备加载文件。由于这里有大量的文件，很容易就对这些文件失去控制。为了避免混乱，必须清楚哪些文件用作什么目的。需要为缓冲区内的数据结构采用有效的命名规范。考虑以下的一些建议。

标示进程　　标示和进程相关的文件。如果文件是从转换中生成的，用文件名表示清楚。如果这个文件是每日更新的一部分，文件名要清楚地体现出来。

表明目的　　假设你要为每日更新产品维表建立一个日程表。你要知道必需的输入加载文件。如果文件名表示了它被建立的目的，将为你建立的更新日志有很大的帮助。缓冲区文件命名规范应该包括文件建立的目的。

示例　　下面给出的是在缓冲区内几个文件的命名。看看下面的命名是否有意义的，是否符合规范：

sale_units_daily_stage

customer_daily_update

product_full_refresh

order_entry_initial_extract

all_sources_sales_extract

customer_nameaddr_daily_update

物理文件命名规范　　规范必须包括所有类型文件的命名转换。这些文件不仅限于数据仓库的数据和索引文件。还有其它的一些文件。需要为下面的文件建立规范：

- 保存源代码和脚本的文件
- 数据库文件
- 应用程序文档

物理存储

考虑一个查询过程。对查询语句的语法检验和数据字典授权验证之后，DBMS 对查询语句进行翻译确定需要哪些数据。从需要的表、行和列的数据入口，DBMS 将请求映射到物理存储数据访问发生的地方。查询涉及到了物理存储，这里就是输入操作开始的地方。数据查询效率与数据在物理存储介质上的存储位置以及以何种方式存储有密切关系。

存储区数据结构

纵观与数据仓库相关的所有数据。首先，有缓冲区内数据。虽然可以在存储和加载过程中寻求提高效率的方法，从用户角度来看，调整缓冲区内的数据对数据仓库性能影响不大。其次，其它与数据仓库数据内容相关的数据，就是数据和索引表。如何安排和存储这些表对性能有很大影响。接下来，你在 OLAP 系统中有多维的数据。很多情况下，这些支撑软件决定了存储以及如何从 OLAP 系统中查询数据。

图 18-6 显示了数据仓库的物理数据结构。观察数据的不同层次。注意细节和汇总的数据结构。想想这些数据结构如何在物理存储中以文件、块和记录实现。

优化存储

看完了物理存储结构。将每个数据结构分解到物理存储这一级，这些结构以文件形式存放在物理存储介质上。以客户维表和销售人员维表为例，存储这两个表有两种最基本的数据存储方式的选择。其一是将每个表的记录存储在各自的物理文件中。或者，如果这两个表的记录经常是一起被查询，可以将这些表的记录存储在同一个物理文件中。在两种方式下，记录都是存在文件中。一个记录集合组成一个块。也就是说，一个文件包括很多块，每个块包括若干条记录。

在本小节，让我们看几个优化存储的技巧。记住，物理层上的优化跟 DBMS 提供的功能和特性有很大的关系。你要将这里的相关技巧和你所使用的 DBMS 联系起来。下面是几个优化技巧。

设定正确的块大小　　你已经知道，一组记录存储在一个块中。块有什么特殊之处呢？文件中的块是数据库被操作的数据和内存之间 I/O 传输的基本单位。每个块有一个包含了控制信息的块头。块头不是用来保存数据的。太多的块头意味着更多的空间浪费。

假设客户信息文件的块大小是 2KB，平均 10 条客户记录填充一个块。每种 DBMS 有自己的缺省块大小设置。2KB 和 4KB 是常用的。如果一个查询的数据记录在第 10 个块，操作系统就将这整个块读入内存以获得需要的数据记录。

增大文件中的块大小会有什么效果？更多的记录和行可以放入一个块中。因为一次读操作可以读入更多的记录，大块减少了读操作的次数。另外一个优点是块头的空间占用减少。在较大的块中块头所占空间比例比较小。从而所有块头占用的空间将减少。但是大块也有它的劣势，即使需要的记录很少，操作系统也将读入很多不必要的信息到内存中，影响了内存管理。

考虑以上所有的因素设置合适的块大小。通常，增加块大小会提高性能，但是还要找到最合适的块大小。

设置合适的块使用参数　　大多数优秀的 DBMS 允许定义块的使用参数，设置一个合适值

以获得性能的提高。你会发现这些使用参数本身和设置方法依赖于特定数据库软件。通常，两个参数决定块的使用，块使用合理将提高性能。为了理解它们的含义，对这两个参数给出一个例子。

示例：

块空闲率（Block Percent Free）　　20

块使用率（Block Percent Used）　　40

块空闲率　　DBMS 为每个块预留一部分空间，使块中的记录能够扩展。记录只有在更新扩展时使用这里的预留空间。当记录被修改而且扩展了，预留空间就要被使用。这个示例的块空闲率参数设为 20。意思就是每个块的 20％为记录更新修改预留空间。在数据仓库中，基本没有多少更新。最大的负担是插入数据记录。累积的负载也大多数是插入。当然，较少改变的维表有很少的更新操作。所以，将这个参数设为一个很大的值将是巨大的空间浪费。一般做法是将这个值设的越小越好。

块使用率　　这个参数指定一个水平线，当块中已使用空间低于该水平线时才运行向该块中插入新的数据行。本例中此参数设为 40。当行从这个块中删除时，空闲空间是不能被立即使用的，除非最少有 60％的空闲空间才允许重新使用。当已经使用的空间降低到 40％以下，空闲空间才可以使用。数据仓库中是怎样的情形呢？大多数，是添加新的记录。很少有删除记录除非是将数据仓库中的数据归档。一般规律是将这个参数设的越大越好。

数据迁移管理　　当块中的记录被更新而在本块中没有足够的空间存储扩展后的记录时，大多数 DBMS 会将整个扩展后的块移到一个新块，然后建立一个到迁移块的指针。这种迁移影响了性能，需要读入多个块。这个问题可以通过调整空间空闲率参数加以解决。然而迁移不是数据仓库的主要问题，因为只有很少的更新产生。

块使用管理　当数据块中包含大量的空闲空间性能将会下降。如果查询需要全表扫描，由于需要读入太多的块而导致性能下降。这时就需要减少块空闲率增加块使用率。

解决动态扩展　　当磁盘上文件的当前扩展区间已满时，DBMS 需要找到一个新的扩展区间以允许插入新的记录。这种从空闲空间中寻找新的区间称为动态扩展。然而动态扩展带来了巨大的开支。分配较大的初始扩展区间可以减少动态扩展。

采用文件分带技术　　当将数据分成多个物理部分存储在多个物理设备上可以采用文件分带技术。文件分带使 I/O 操作并发进行，提高了文件访问性能。

使用 RAID 技术

冗余阵列（RAID）技术在今天的使用的非常普遍，数据仓库从中获益匪浅。在大型服务器上建立的这些磁盘阵列可以使服务器不间断工作，即使发生磁盘故障也可以恢复。从 RAID 获益的底层技术就是将数据分成几个部分，将各个部分按照条带的形式写入多个磁盘。这种技术可以从一个失效的磁盘中恢复并重建数据。RAID 是容错的。它的基本特征如下：

磁盘镜像——将相同的数据写入到连接到相同控制器的两个磁盘

双磁盘——和磁盘镜像类似，不同是每个驱动器有自己单独的控制器

奇偶校验——为数据加入校验位以保证数据传输可靠

磁盘分带——数据按扇区或者字节分布在多个磁盘上

RAID 分为六个不同的等级：从 RAID0 到 RAID5

参见图 18-7，给出了 RAID 的简单描述。注意其优缺点。最低级的配置是 RAID0，提供数据分带。最高级的是 RAID5，是一种高效的配置。

估计存储容量

讨论物理存储没有估计存储容量是不够的。物理模型工作在物理存储中。随着数据仓库的增大需要知道最初和后续需要多少存储空间。

下面是估计存储容量的几个技巧：

对每个数据库表，确定

- 最初的大约行数
- 行的平均长度
- 行的每月增长

- 表的初始大小，以兆字节（MB）计算
- 表 6 个月和 12 月的大小

对所有表，确定

- 索引的个数
- 索引需要的空间，最初、6 个月和 12 个月

估计

- 排序、合并需要的临时空间
- 缓冲区内的临时文件
- 缓冲区内的永久文件

数据仓库索引

在像数据仓库这样的以查询为中心的系统，最重要的是快速地处理查询。无缘无故极缓慢的查询是你的用户放弃数据仓库的主要原因。对一件需要快速连续复杂查询的分析事务，你的查询结果要跟得上思维的步伐。在所有提高性能的方法中，索引占的份量很高。

你的数据仓库应该建立怎样的索引？DBMS 供应商提供了不同的选择。选择不再局限于顺序索引文件。为提高数据操作效率，所有的供应商都提供了 B-Tree 索引。另外一个是位图索引。本节下面我们将看到，这种类型的索引技术非常适合于数据仓库。一些供应商为特定的需要扩展了索引的能力。包括分区表索引和索引组织表。

索引一览

让我们从数据仓库的角度看索引技术。数据表是“只读”的。这说明基本上没有记录的更新和删除。在加载后也没有数据插入到表中。当你增加、更新和删除记录将导致大量的对索引文件的操作。但是在数据仓库中不会这样。所以你可以为每个表建立很多的索引。

每个表要建立多少个索引呢？大多数的索引建立在维表上。通常，数据仓库比 OLTP 系统有更多的索引。随着表的增大，索引也增大，需要更多的存储空间。一般的规则是索引的最多个数和表的大小成反比。由于要为新记录建立索引，大量的索引会影响加载过程。需要

平衡多个因素以决定每个表建立几个索引。一个个表逐个确定。

索引和加载　当存在大量的索引时，向数据仓库中加载数据速度会非常慢。因为当每加入一条记录到数据表，每个相关的索引必须重新建立。这个问题在初始加载时尤其严重。为解决这个问题可以在加载工作前先删除索引。这样在加载过程中就不为每条记录建立索引入口。加载过程完成后，再建立索引。建立索引需要很长的时间，但是不会有在加载时建立索引那么久。

大表的索引　拥有上百万条记录的表不能建立太多的索引。当表太大了，建立超过一个索引是非常困难的。如果必须建立多个索引，建议将大表分成小表，然后再建立多个索引。

索引读　在查询数据过程中，索引记录是首先读入的，然后再读入对应的数据。DBMS 从多个索引中选择最优索引。例如，DBMS 使用建立在四个列上的索引而且许多用户从这四列和另外一列取数据。获得数据的过程是怎样的呢？DBMS 从索引记录中获取对应的数据记录。你至少需要两次输入输出操作。本例中，DBMS 需要再检索另外的一列数据记录。因此应该考虑将这一列也加入到索引中。这样 DBMS 读索引后发现所有需要的信息已全部包含在索引记录中，就不需要再读入数据记录。

选择索引的列　怎样在表中选择最适合的列作为索引列？如果采用索引，哪几列能够带来最优的性能？分析最常用的查询，观察哪几列经常用来限定查询。这几列就是索引的候选列。例如若查询是基于产品线的，就将产品线作为索引列的候选。

一种缓冲机制　很多数据仓库管理员对如何开始索引非常迷惑。每个表需要几个索引，初始配置的数据仓库的哪几列必须作为索引列？他们对表本身有初步的了解，但对实际需要的查询没有什么经验。这是关键的一点，经验不是准则。你需要用户在数据仓库上试用一段时间。一个临时的索引方法应该谨慎一些。只为每个表的主键和外键建立索引。密切监视系统性能。特别注意那些运行很长时间的查询。随着越来越多的用户的使用要增加索引。

B-Tree 索引

大多数数据库管理系统将 B-Tree 索引作为缺省的索引机制。当你使用数据库软件的 DDL 编写建立索引代码，系统建立的是 B-Tree 索引。RDBMS 自动对主键建立 B-Tree 索引。B-Tree 索引优于其它索引地方在于其获取数据的速度、易于管理和简便。如图 18-8，显示了一个 B-Tree 索引。注意树结构最上面有一个根节点，这个索引包括一个 B-Tree（平衡二叉树）结构。本例的索引列是姓名。B-Tree 使用存在的所有姓名作为索引列。上面的方块包含索引数据指向下面的方块。把 B-Tree 索引想象为包括分层结构的块。最下一层的块也就是叶子块指向数据表的一行。数据的地址在叶子块中。

若表的某一列有很多唯一的值，那么就说这列的可选择性比较强。在一个地域维表中，城市这一列包括很多唯一值，这一列的选择性比较好。B-Tree 索引比较适合于高选择性的列。因为叶子节点的值是唯一的，它们将指向一个唯一的数据行，而不是一个数据行链。如果单列不是高度可选择性的会怎样呢？在这种情况下能使用 B-Tree 索引吗？例如，雇员表的雇员名字没有很好的可选择性。因为有很多人重名。但是可以将姓和名连在一起。这样可选择性就增强了。可以为这两列建立 B-Tree 索引。

索引的增长和被索引的数据表成正比。包括多列的索引将大大增加索引的大小。因为数据仓库处理的是大量大容量的数据，索引文件的大小要引起重视。数据仓库中的选择性是怎样的呢？大多数类可选择性很好吗？不是这样的，检查每个维表中的列，大量的列包含的是低选择性的列。B-Tree 索引处理低选择列并不太好。有可选的方法吗？这将导致我们转向另外一种索引技术。

位图索引

位图索引非常适合于低选择性的数据。一个位图是一个按序排列的点阵，对应于索引列的每个不同的值。假设颜色字段有三个不同的选择，白、浅黄褐色和黑。用这三个值建立一个位图索引。每个位图入口点有三个点。假设第一个点代表白色，第二个是浅黄褐色，第三个是黑色。如果一种产品是白色的，这个产品的位图有三个点，第一个点被设为 1，第二和三个点设为 0。如果一种产品是浅黄褐色的，那这个产品的位图三个点表示，第一个设为 0，第二个为 1，第三个为 0。这样你得到了一张图。现在研究一下图 18-9 所示的位图索引示例。

此图代表销售表的一部分，为三个不同的列建立位图索引。注意索引中每个入口包含顺序的点如何代表列中不同的值。为基本表的每一行建立一个入口点。每个入口点包括基本表行的地址。

位图索引怎样获取查询的行呢？考虑在上面例子中销售表上的一个查询：

Select the rows from Sales table

Where Product is “Washer” and

Color is “Almond” and

Division is “East” or “South”

图 18-10 说明了使用布尔逻辑怎样基于图 18-9 所示位图索引获得结果集。

你可能已经观察到，位图索引支持低选择性的查询。这种技术的优点在于当在查询中的低选择性列上使用断言的高效性。对于低选择性的列，位图索引比 B-Tree 索引占用更少的空间。在数据仓库中，很多的数据访问都是基于低选择性的列。而且，如果使用“what-if”分析，需要查询几个条件。你会发现位图索引更适合数据仓库环境而不是 OLTP 系统。

另一方面，如果有新值加入到包含位图索引的列中，位图索引必须重新建立。还有一个缺点是在位图索引访问后总是要访问数据表。而如果请求的信息已经包含在索引当中，B-Tree 索引不需要访问数据表。

簇索引

一些 RDBMS 提供了一种新型的索引技术。在 B-Tree、位图或其它顺序索引方法中，有存储全部列的数据段，也有存储索引的索引段。索引字段包括被索引的列而且还包括数据段中的地址入口。簇表将数据段和索引段结合起来，合二为一。数据是索引，索引也是数据。

簇表极大地提高了性能，因为一次读既有索引段也有数据段。使用传统的索引技术，需要一次读获得索引段，第二次读获得数据段。如果查询匹配或者一段范围的值使用簇表将会非常快。如果你的 RDBMS 支持这种类型的索引，在你的环境中能用就用这种索引。

索引事实表

事实表中一般都有什么？这些列的本质是什么？重新回顾一下 STAR 方案。事实表的主

键包括所有相关维表的主键。如果你有四个维表，库存、产品、时间和促销，那么事实表的完整主键就是这四个表的主键的合成。其它的列呢？其它的列是度量，如销售单位、销售现金和支出现金等等。以上类型的列都可以考虑作为事实表的索引列。

- 如果 DBMS 不在主键上建立索引，手工为全部的主键建立一个 B-Tree 索引。
- 仔细设计组合键中的单个键的顺序。将查询中经常使用到的列作为组合键中级别高的键。
- 单独考虑组合键中的每一个键，根据查询过程需要建立组合索引。
- 如果 DBMS 支持访问智能组合索引，就可以为每个单独的键建立索引。
- 不要忽略为包括度量的列建立索引。例如，如果查询是基于指定范围的现金销售额，则字段“dollar sales”是索引的候选列。
- 不要对事实表使用位图索引。基本没有低选择性的列。

维表索引

维表中的列都可以用作查询的谓词。查询可能像这样：北方分公司三月的 A 产品销售量是多少？这里的列，产品、月份、分公司从不同的维表中获得，是建立索引的候选列。检查维表的每一列，为这些表计划索引。索引事实表可能得不到太大的性能上的提高，但是维表索引将较大地提高性能。

下面是建立维表索引的技巧：

- 在单一主键上建立唯一的 B-Tree 索引。
- 检查约束查询经常用到的列。这些列是位图索引的候选列。
- 在大的维表中查找经常被一起访问的列。确定如何在这几列上建立和安排多列索引。记住，经常访问的列或者在维表中的层次比较高的列应该在多列索引中具有较高的优先级。
- 为经常用于连接（Join）条件的每个列建立单独的索引。

提高性能的技术

除了前几节讨论的索引技术，还有一些其它的方法可以用来提高数据仓库性能。例如，存储时进行数据压缩以使更多的数据写入到一个块中。也就意味着一次读入可以获得更多的数据。另外一个方法是合并表。这种方法同样是让一次读可以获得更多的数据。经常从数据仓库中清除不需要的数据也可以提高整个系统的性能。

在本节的余下部分，我们看看其它提高性能的技术。很多是来自于 DBMS，大多数都适合于数据仓库环境。

数据分区

典型情况下，数据仓库包括一些非常巨大的数据表。事实表包括上百万条的记录。维表例如产品表和客户表也可能有很多行。当你拥有如此巨大的表，就面临一些特殊问题。首先，载入大容量表将花费很长的时间。其次，为这些大表建立索引也要花上几个小时。在大表上处理查询会怎样呢？从大量数据排序获得结果集的查询也需要很长时间。备份和恢复大表花费更长的时间。而且，当从大表中有选择性地清除和归档记录时，遍历所有的行需要很长的时间。

如果能够将大表分成可管理的块会怎样呢？会不会有性能的提高？在小表上的维护和操作比较简单和迅速。分区是一个重大的决定，必须在前面计划好。等数据仓库配置好，已经成为产品再做这件事是非常费时和困难的。

分区就是将表和它的索引分成可管理的几个部分。DBMS 支持和提供了这种分区机制。在定义表的时候也定义了分区。表的每个分区被看作不同的对象。当一个分区的容量增大，你还可以将这个分区再分区。将分区分布在不同的磁盘上以获得最优的性能。表的每个分区可能有不同的物理属性，但是有相同的逻辑属性。

表分区的条件是怎样的？可以将大表水平或者垂直分区。在垂直分区中，将选择的列编组分割为分区。每个分区和原始表具有相同的行数。通常，宽的维表适合于垂直分区。水平分区正好相反。将选择的行分组进行分区。在数据仓库中，基于时间的水平分区是非常好的选择。可以按照最近事件和历史事件将表分区。可以将最近的事件继续运行而将历史数据脱机进行维护。水平分区比较适用于事实表。

已经看到，分区是一个有效的存储管理技术，有利于提高性能。总结的优点如下：

- 查询只需要必需的分区。应用对分区透明的或者可以明确地指定访问单独的分区。访问

少量的数据使查询更快。

- 整个分区可以脱机维护。可以分开定时维护分区。分区后可以进行并行维护操作。
- 建立索引更快。
- 向数据仓库中加载数据更容易和易于管理。
- 数据损坏只影响一个分区。备份和恢复一个单独的分区减少停机时间。
- 将分区映射到不同的磁盘驱动器以平衡 I/O 负载。

数据聚簇

在数据仓库中，很多查询需要顺序访问大量的数据。数据聚簇技术解决了顺序访问的问题。聚簇能够预取顺序相关数据。

可以通过物理地将表放在一起以获得数据聚簇。当在 DBMS 中将表声明为簇表，这些表放在磁盘相邻地区域。如何实施数据聚簇依赖于 DBMS 的特征。请研究这些特征并用好数据聚簇。

并行查询

考虑一个访问大量数据的查询，执行统计然后在多个约束下执行一个选择。很明显，如果将这个过程分成小的部分然后并行地执行这些部分可以大大提高性能。并行执行将更快地获得结果。几个 DBMS 供应商提供了并行查询特征，并且并行查询对用户是透明的。查询的设计者，不需要指定为并行查询怎样分割查询语句。DBMS 为用户做这些事情。

并行处理技术可以用于数据加载和数据重组。并行处理技术和数据分区紧密联系。服务器硬件的并行架构也影响并行处理的方式。一些物理选项对高效的并行处理很重要。如果你需要并行处理就要评估一下那些将两个分区放在一个驱动器上的语句。并行处理和分区技术一起提供了提高性能的巨大潜力。然而设计者需要决定怎样更有效地使用这种技术。

汇总级别

我们已经讨论过几次，数据仓库需要包括明细和汇总数据。选择优化 I/O 操作的粒度层次。例如销售数据存储每日明细和每月汇总。如果用户经常查询每周的汇总信息，就要考虑在每周的级别下增加另外一个汇总。另一方面，如果仅保存了周和月汇总数据而没有每日的明细数据。就不能从数据仓库中查到每日的明细数据。按照用户需要确定汇总和明细的级别。

在数据仓库中建立滚动汇总结构是非常有用的。假设在你的数据仓库中需要保存每小时、每日、每星期和每月的汇总。建立滚动机制使数据能够自动按照时间汇总到下一个汇总级别。小时汇总自动到日汇总，日汇总到周汇总等等。

参考一致性检查

众所周知，参考一致性保证两个关联表的有效性。关系数据库的参考一致性规则将子表的外键和主表的主键连接起来。每当增加或删除一行，DBMS 会检查是否保持了参考一致性。这个检查保证当子表的行存在时主表的行不能删除。主表行不存在时，子表不能添加行。在 OLTP 系统中参考一致性检验非常重要，但是降低了系统性能。

考虑将数据加载到数据仓库中。在缓冲区内建立加载映像时，数据结构已经开始了释放、清除和转换的过程。只要涉及到主表和子表，将要加载的数据已经验证了正确性。所以，在加载数据时不需要检查参考一致性。这时关闭参考一致性检查将获得性能的提高。

初始化参数

DBMS 安装是性能优化的开始。在安装数据库系统时，需要仔细计划初始化参数。很多时候，你会意识到性能的下降是由于不合适的参数造成的。数据仓库管理员对选择正确的参数负有特别的责任。

例如，如果你将并发访问的用户最大数量设的过低，那么用户个数就会是瓶颈。很多用户必须等待访问数据库，即使数据库资源是足够的，仅仅是因为这个参数设的太小了。另外，将这个参数设的太大也会造成没必要的资源浪费。下面看看检查点频率。DBMS 多久写入一次检查点记录？如果两次连续检查点间隔太小，将占用太多的系统资源。如果间隔太大又

会影响恢复。这仅仅是两个例子。所有的初始化参数都要正确地设置。

数据阵列

什么是数据阵列？考虑在一个金融数据集市中，需要保存单个帐户的月结余。在一个规范化结构中，一年的月结余将在表中找到 12 条记录。假设在很多查询中用户需要查全部的月结余记录。怎样提高性能呢？可以建立一个数据阵列或者是有 12 个入口的重复组，每一组包含每月的结余。

虽然建立阵列是违反规范性原则的，但这种技术可以大大提高性能。在数据仓库中，时间跟数据关系密切。经常用户搜索一系列时间的数据。另外一个例子是，查询 24 个月每个销售员的月销售额。如果分析一般的查询，你会很奇怪地看到，可能有很多需要的数据已经存储在阵列中了。

本章总结

- 物理设计使数据仓库的实现更接近硬件。物理设计过程可以总结为七个不同的步骤。
- 规范的重要性不能被忽视。在物理设计过程中要使用好的规范。
- 物理设计中的优化存储分配占有很高的地位。要使用 RAID 技术。
- 数据仓库的性能很大依赖于好的索引机制。B-Tree 索引和位图索引都比较好。
- 其它的一些提高性能的方案也是物理设计的一部分：数据分区、数据聚簇、并行处理、建立汇总、调整参考一致性检验、合适的 DBMS 初始化参数调整和使用数据阵列。

思考题

1. 列出四个物理设计的步骤，并描述其中两个的任务。
2. 对物理设计对象命名。哪个对象你认为使最重要的？
3. 物理模型包括的组件是什么？这些和逻辑模型组件的联系？
4. 给出两个原因，为什么数据仓库环境中命名规范很重要
5. 列出三个优化存储的策略。简短描述它们。
6. 什么是索引读？它怎样提高性能的、
7. B-Tree 索引优于其它索引的两个原因

8. 物理表中的可选择性列是什么意思？哪种类型的索引技术适合于低选择性数据？为什么？
9. 什么是数据分区？给出两个理由，为什么在数据仓库环境中数据分区很有用？
10. 什么是数据聚簇？给出一个例子。

练习题

1. 连线题

1. 事实表	A、设为更高级别
2. 动态扩展	B、每个入口的数据地址
3. 文件分带	C、数据组的重复
4. 块使用率	D、数据和索引的结合
5. B-Tree 索引	E、DBMS 寻找新的扩展段
6. 参考一致性检查	F、设为低的级别
7. 簇索引	G、分区候选
8. 块空闲率	H、页节点里的数据地址
9. 位图索引	I、存储在不同的设备中
10. 数据阵列	J、加载时暂停

2. 为你的数据仓库制订规范的大纲。考虑所有类型的对象和它们的命名转换。指出为什么规范是重要的。建立一个详细的目录。
3. 回顾图 10-7 所示的 STAR 方案的订单分析。转换为物理模型。列出物理模型的所有组件。将物理模型和逻辑模型联系起来。
4. 最常用的块利用率的两个参数是什么？用适当的例子描述正确设置这两个参数怎样提高存储利用率。和 OLTP 系统的设置有哪些不一样？为什么？
5. 作为数据仓库管理员，提高性能是首要任务。着重你计划采用的技术。对每种技术，找出实现技术必须要做的工作。

第十九章　数据仓库部署

本章目标

- 研究部署阶段在数据仓库开发流程中的角色
- 部署的主要任务和如何完成任务
- 检查领航系统的需要，如何划分领航的类型
- 数据仓库环境下的数据安全
- 考察数据备份和恢复需求

现在到了可以进行开发数据仓库第一个版本的时候了。部署是构造数据仓库的下一个步骤。在部署阶段，只剩下了几个细节，启动数据仓库，用户收获的时候到了。到了部署阶段，大部分功能都已经实现了。部署主要涉及到用户培训、服务支持、访问数据仓库的硬件和工具。

为了认清我们目前在数据仓库开发循环中我们所处的位置，需要总结一下我们到现在为止完成的功能和操作。下面是在构造阶段我们已经完成的主要工作：

- 基本构造的组件进行了完整测试。
- 架构的有效性已被论证。
- 数据库已定义。各表的空间分配完成。
- 缓冲区的文件分配全部建立。
- 解析、转换和其它的缓冲区作业全部测试完成。
- 建立的加载映像在开发环境中已经测试完毕。初始加载和增量加载已经完成。
- 查询和报表工具已经在开发环境中测试完毕。
- OLAP 系统已经安装和测试过了。
- 完成了数据仓库的 Web 功能。

部署的主要任务

让我们从构造完成开始说起。大家已经看到，我们已经完成了大量的复杂工作。所有的组件已经测试过，已经各就各位了。如图 19-1 是部署阶段的各项任务。每个框中的主要活动是本阶段的主要工作。在部署阶段，为用户和项目组之间建立一个反馈机制，让用户知道部署是如何进行的。如果这是初次进行，大多数用户对这个过程不熟悉。虽然用户可能已经接受了培训，在这个时候充分的交流也是很重要的。准备提供支持。让我们看看部署各阶段的主要工作。随着我们的推进，请了解一些技巧并将它们运用到你的环境中去。

完成用户接受

用户正当的=接受在部署阶段不只是一个形式而是绝对必须的。在关键用户没有对数据仓库表示满意前不要强行进行部署。有些些公司有一个正式的项目完成过程。其它的采取一系列用户接受测试，使每个功能被接受。不管用户接受工作是怎样完成的，但是一定要完成，跟你日常工作环境中通常所做的一样。

从用户这方面看，谁应该接受用户接受测试呢？想想已经在项目组中的用户。如果你的项目组中有一个用户联系经理，这个人就应该负起责任。让最终用户应用专家在他们自己领域进行接受测试。除了对项目组中的用户代表进行测试，也包括其它的一些用户的最终测试。

怎样进行用户接受测试呢？在部署最后阶段哪些特殊用户需要测试呢？下面是一些技巧：

- 在每个领域或部门，让用户选择几个典型的查询和报表，一是为了可以简单地验证一下结果，另外是验证维表的约束。让用户执行查询产生报表。最后从操作型系统生成报表作为验证。比较操作型系统的报表和数据仓库产生的报表。找出并计数所有的不同。在和数据仓库比较之前，验证操作型系统的结果没有任何错误。
- 现在是测试预定义查询和报表的最好时候。让每个用户组选择一些这样的查询和报表测试它们的执行。
- 让用户测试 OLAP 系统。如果你采用 MOLAP 方式，为 OLAP 系统建立多维立方体。让用户选择大约五个典型分析案例进行测试。同样，与操作型系统的结果比较。
- 众所周知，几乎每个数据仓库，用户需要学习和习惯新的界面工具的功能。大多数用户

要很容易地使用这些工具。在结束前为工具的可用性设计接受测试。当然，大多数这类测试是在工具选择时做的。但在那时，是在提供商或者系统开发测试环境下测试的。现在是产品环境的测试。这是最大的不同。

- 如果你的数据仓库是基于 Web 的，让用户测试 Web 特性。如果 Web 技术用于信息交换，也让用户测试这一项
- 没有系统性能接受测试的用户接受测试是不完整的。项目必须将用户的性能期望放在一个可以接受的程度上。响应时间大约是三到五秒。实际上，单独的查询和平均值是不同的，这是很明显的。用户能够接受这些变化，只要是例外而不是常事。

执行初始加载

在第 12 章，我们已经深入地了解了数据仓库的数据加载。我们回顾一下初始加载如何进行，还有增量加载的方法。我们已经知道了将数据放入仓库中的四个模式。项目到了部署的阶段，项目必须测试初始数据加载和仿真增量加载。当前是完整初始化加载。现在还不能进行第一次增量加载，通常在部署后的 24 小时之后做。回忆在 12 章学到的背景知识，让我们回顾完整初始化加载的步骤。如果你需要回到第 12 章简单地温习一遍，现在就可以去做。特别注意怎样加载维表和事实表的映像。初始加载过程使用这些已经以数据记录形式存在的映像。

以下是完整的初始加载主要步骤：

- 删除数据仓库关系表中的索引。都知道，加载时建立索引耗费大量的时间。初始加载的数据量非常大，有几十上百万的数据行。不能承受任何使加载进程减慢的重负。
- 你已经知道，每个维表和对应的事实表是一对多的关系。也就是 DBMS 在关系上使用了关系一致性检验。但是我们假设加载映像是非常仔细地建立的。我们可以挂起这些限制，加速加载过程。这取决于项目组，取决于你对加载映像建立的信任程度。
- 在一些情况下，初始加载可能需要几天的时间。如果你的加载过程几天后因为系统失效而中断，灾难就产生了。如何解决这样的问题？需要从头开始重做一遍吗？不。确定你已经做了合适的检查点。从最后的检查点继续。
- 由于第 12 章的表述的原因，先加载维表。记得怎样为维表建立键。回忆事实表的

键怎样从维表的记录形成的。这就是为什么先要加载维表，然后是事实表。一些数据仓库组喜欢在加载大表之前先加载小的维表以验证加载过程。

- 下面加载事实表。事实表记录的键在缓冲区建立加载映像前已经解析出来。
- 基于已经为聚合和统计表建立的计划，建立基于维表和事实表的聚合表。有时候，加载映像已经在缓冲区建立了。如果这样，应用这些加载映像建立聚合表。
- 在加载时停止了索引建立，现在是建立索引的时候了。
- 如果没有挂起参考一致性约束，在加载过程中，所有的参考性错误记录在系统中，检查日志文件，找出所有加载异常。

准备用户桌面

为用户准备计算机在数据仓库项目从头到尾过程中只占相对很小的一部分。虽然在整个过程中只占不到 10%，但用户在他们桌面上看到和经历到的东西对他们才是最有价值的。桌面工具是数据仓库为用户准备的工具。所以，特别注意安装数据访问工具，用于用户计算机和服务器的网络连接，中间层的配置。依赖于配置方法，为准备桌面留够充足的时间。

开始前，为客户端给出一个配置列表，所有的信息传输软件的安装，桌面计算机需要的硬件，网络连接的整个需要。给出几条实际的建议。

- 客户端数据访问工具远程配置是一种快速的方式。数据仓库管理员能够从一个中间地方安装不同客户端的计算机，避免单独对每个客户端的上门服务。另外，如果计划逐一地安装测试连接工具，计划需要更长地时间。
- 不管部署方式是远程安装还是单独为每个用户安装，这是升级工作站和其它用户缺少的软件的唯一时机。
- 如果服务器和中间件没有做好，桌面工具就没有作用。计划安排一段时间，安装测试这些附加部件。
- 测试所有客户端，保证所有部件安装正确，连接在一起工作正常。
- 完成准备桌面工具意味着用户可以上机访问数据仓库信息。还需要建立用户名和密码，保证完成和测试通过。

完成初始用户培训

培训的重要性定位怎么强调都不过分。IT 专业人员可能认为数据组件、应用和工具是分散的东西。从 IT 部门的观点看，培训是这三个部件的培训。但是对用户，这是一个。他们是不会区分应用和工具。培训必须从用户的角度来考虑。操作型系统实现和数据仓库实现存在巨大的不同点。数据仓库提供的功能有很大的潜能。用户意识不到数据仓库工具真正能为他们做些什么。

开始前，按照下面几个方面培训用户：

- 简单数据库和数据存储概念
- 数据仓库基本特征
- 每个用户组的数据仓库内容
- 浏览数据仓库内容
- 使用数据访问和查询工具
- 信息获取的 Web 技术应用
- 预定义的查询和报表集
- 可以进行哪些类型分析
- 查询模板，如何使用它们
- 报表生成计划
- 数据加载计划和数据流通
- 用户支持机构，包括一线联系

制订最初用户支持

在部署后的最初几天，数据仓库项目组的每个成员通常都非常繁忙。用户有很多方面的问题，包括从如何登录到复杂的 drill-down 分析。许多问题可能仅仅与硬件相关。用户需要大量的支持，尤其在最初阶段。

图 19-2 是一个最初用户支持示意图。注意基本支持的中心。每个部门的用户代表是接触的第一点。这个人必须训练有素，能够回答应用和数据内容相关的大多数问题。用户代表也必须知道客户端工具的使用。如果用户代表不能解决问题就需要热线支持。至少在初始部

署时，技术支持工作要做好。并且，注意技术支持组提供的支持类型。

部署筹备

建造和部署数据仓库是许多组织的主要任务。这个项目需要很多不同的技术。数据仓库包括几种不同的技术，你面对着新的设计技术。维建模是一个很不一样的技术，以前不被设计者在可操作的系统中使用。数据解析、转换和加载是冗长的并且耗费劳力。用户以新的形式接受信息。付出是巨大的，对企业来说，比以前所有的信息项目都大。

在这种情况下，什么是部署你的数据仓库的最合理的方法。可以肯定的是，如果将部署分成几个可管理的部分，就可以将数据仓库以一个合适的步骤来引入。在筹备阶段进行部署。计划好筹备，从用户和项目组的角度建立最高效的时间表。

在第二章，我们讨论了三个建立数据仓库的方法。自顶向下的方法是从企业的标准化数据为部门的几个数据集市提供数据。自底向上的方法是建立一组数据集市不考虑将它们融合在一起。最实际的方法是通过建立一致的数据集市的聚合从而建立一个超级数据市场，。

不管你的项目组对你的环境似是而非的原因采取的方法，分解和筹备你的配置。对所有的方法，全部都需要需求分析、架构和基础构造是共同的观点。你为整个企业做出计划，但是你得将它分成几个定义好的阶段来进行配置。图 19-3 显示了配置的几个阶段。注意不同方式下的建议阶段。注意在自顶向下方式下如何建立整个企业级数据仓库。接着这些相关的数据集市以一个合适的顺序进行配置。自底向上方法的结构性不太好而且不好优化。在实际方法中，你一次只配置一个数据集市。

维的一致是实际方法成功的关键。让我们重新考虑一下这个重要的概念。主要是因为是能够保护组成企业数据仓库的所有数据集市内聚力。在基础阶段，维的一致的意思是：当我们在两个不同的数据集市中提到“产品”，它们有相同的意思。另一种说法是，在每个接下来配置的数据集市中的产品维表和第一个产品维表是相同的。表的所有数据和键都是相同的。

一个领航系统

大多数公司在配置整个数据仓库前都部署一个领航系统。这里不是将第一个完整的数据集市作为领航系统。它是无关独立的，有其特殊作用。先部署一个领航系统有几个原因，它使项目组获得更广泛的经验，获得新技术带来的特殊经验，为用户演示概念证明。

如果你的项目组选择一个领航系统，从一开始就将重点放在基础上。认清领航系统的目的和意义，为它选择一个合适的主题范围。记住很少领航系统部署是用后抛弃的项目。对待领航系统应该像对待一个正规的项目一样。

什么时候领航系统数据集市有用？

开始部署数据仓库充满了失败的危险。如果你第一次没有做好，你可能就没有第二次机会使你的客户相信你的新模式的价值。成功是主要的目标，不能冒失败的风险。必须能够在较短的合理时间内证明潜在的积极效果，而且能够将工作简单地管理好。一个给部分在有限封闭的环境里的用户部署领航系统，是非常吸引人的。

这不是说领航系统部署总是必需的。根据你的环境不同不一样。你的用户组可能只包括复杂性很高的分析，你的 IT 组是由很多有经验的人组成，对他们来说没有什么系统不简单。如果是这样你的数据仓库的配置就不需要一个领航系统。但是大多数公司不是这样的。按照下面的列表指出的条件确定领航系统部署在什么地方是有用的：

- 对用户成员来说，数据仓库概念都是全新的
- 必须给用户演示，说服他们，他们获得数据信息是非常简单的
- 用户需要从新工具和技术中获得经验
- 分析人员应该能够感觉到数据仓库中的分析特征的能力
- 赞助商和上级管理者在大量投入前必须看到数据仓库概念带来的好处
- IT 设计者和架构者需要在维建模技术和这个模型的数据工作上获得经验
- 项目组需要确保 ETL 功能完善
- 项目组需要确认所有基础组件是否能够很好地工作配合。像并行处理、复制、中间件连接、Web 技术和 OLAP 元素。

领航系统的类型

从领航系统的保证条件情况看，可以推断出领航系统部署有多种类型。当一个项目组考虑一个领航系统，有一系列的原因，所以领航系统要向需求倾斜。经常，领航系统并不只为一个原因而建立，而是一系列需求融合在一起而需要一个领航系统部署。在本小节，让我们简单看看六种领航系统。每个领航系统代表一种主要的原因，当然也包括其它一些小的目的。首先，如图 19-4，描述了六种领航系统，指出了每种类型的主要目的。

概念证明领航系统

数据仓库是决策支持的可行的解决方案。建立这个推断是概念证明领航系统的主要目的。你应该给很多的用户包括上层管理人员证明这个概念。必须为赞助商和高级经理明确概念以获得建立整个数据仓库的资金。定义领航系统的范围是基于用户的。不考虑范围，这类领航系统应该提供使用数据仓库的主要特征的示例，获得信息是如何地简单。你的重点在于用户与信息系统的交互。概念证明领航系统是基于有限数据的。

项目组认为这类领航系统在开发方案中比较简单。不需要很长的时间。主要目的是给用户的印象是觉得数据仓库是信息获取的一种非常有效的方式。通常，概念证明领航系统花费不长于半年的时间来建立和测试。将焦点放在有效地表述概念和快速地获得认同。

技术证明领航系统

这可能是最简单和容易建立的。主要为证明一两种技术，用户对这种领航系统没有太大的兴趣。你可能只是想测试和证明一种维表建模工具或者数据测试工具。或者，你可能只是想证明 ETL 工具的有效和有用性。利用这种领航系统，跳过产品说明和提供商的声明自己来探索一下工具本身。

这种技术证明的领航系统是在于你能够着重于一两种技术，证明它们能够满足需要。你可以检查特定类型的复制工具对你的数据仓库环境的可用性。然而，领航系统只限于证明你所有技术集合的一小部分，它不能说明所有的部分都能够工作正常。这个问题带出了下一种类型。

综合测试领航系统

用于开发和配置以检验所有的基础结构和架构组件能够很好地一起工作。这不是功能完全的完整的数据仓库和一个小的数据库系统，但是你可以验证数据流从通过源操作型系统缓冲区到信息获取组件的整个过程。

这个领航系统使专业 IT 人士和用户都认识到数据仓库的复杂性。项目组从新技术和工具中获得经验。领航系统不能在短时间内组合和配置到一起。领航系统的范围报告整个数据仓库的功能。对项目组比对用户更有用。

用户工具认定领航系统

推行这种领航系统是为了给用户提供他们看到和使用的工具。将重点放在最终用户信息的获取工具上。将数据内容和数据准确性置于后台。只注意工具的可用性，用户可以观察到给他们的最终用户工具的所有特征，使用工具，测试其特征和实用性。如果不同的用户提供了不同的工具，你要为这种领航系统提供不同的版本。

注意，这里没有关注数据的完整性和这类领航系统如何工作与数据仓库的整个数据内容联系。用户工具认定领航系统有极其有限的应用。一个常用的地方是 OLAP 系统。

广事务领航系统

和前一种类型比较，这类领航系统包括很广的业务范围。试着理解这类领航系统如何开始。管理者在特殊业务上对决策支持有相同的迫切需要。如果将这些放在一起以满足需求，那么潜在的成功机会将是巨大的。在企业中管理者是最先想从数据仓库中获得好处的。项目组的责任是提供高度可见的可简单获得的领航系统。

这类领航系统基于一些特定需求有一定的问题。首先，你有时间压力。根据需求，领航系统的范围太窄，而不能和后面的数据仓库集成。或者，这个领航系统可能会认为是太复杂了。太复杂的项目不能认为是一个领航系统。

扩展种子领航系统

首先，注意存在这类领航系统的动机。你想要有商业价值的东西。这个范围必须是可管理的。你想让它对用户技术上尽可能地简单。然而，你有一个合适的简单主题的选择。简单不是没用。选择一个简单，有用和可视化较强的商业领域，但是计划深入广泛地使用领航系统来了解数据仓库地特征。就像种下一颗好种子看着它发芽生长。

项目组从中获益是因为他们可以观察和测试不同部分的工作。用户获得对工具的认同和理解他们怎样和数据仓库交互。数据仓库的管理员功能也应该被测试。

选择领航系统

应该明白对领航系统类型没有工业标准的命名协议。一个数据仓库实践者可以调用一个基础领航系统，测试领航系统或者另外一个架构计划领航系统。实际的名字没有关系。范围、内容和积极性。而且注意这些组或者类型也是任意的。你也可以提出其它四种类型。然而，推动任何领航系统的相同的动机就是上面描述类型的一种。记住没有一种领航系统仅仅属于特定类型的一种。你会发现要跟踪你需要采用的很多种领航系统类型。当项目组建立数据仓库时，就在一个特定得技术和业务环境中介绍这种新的决策支持系统。企业的技术和环境影响领航系统的选择。而且，这种选择也依赖于这个数据仓库计划是 IT 驱动、用户驱动还是一个真正的联合小组。

让我们检查企业的条件决定我们的这种领航系统是否合适地匹配。研究下面介绍的指导思想。

如果数据仓库概念对你们企业是全新的，而且你的上级管理层需要直接证明和说服，采用概念证明领航系统。但是大多数公司不是这样的。很多的文献、讲座和开发商演示关于数据仓库的概念，实际上每个人都或多或少了解这个概念。对企业来说剩下的问题仅仅是对概念的适应性问题。

技术证明和综合测试证明是为 IT 服务的。用户通常不直接从这两种得到什么。如果你正在将当前的基础结构扩展到数据仓库，或者你采用了并行处理硬件和 MOLAP 技术，你就需要考虑这两种。

用户相关的数据仓库的培训的重要性怎么说也不过分。越多的用户适应数据仓库并从中获益，项目成功的几率越大。所以，用户工具认同领航系统和广业务领航系统有非常实际的用处。虽然用户工具认同领航系统有其范围和应用的局限性，但是确实有它存在的位置。通常它是用完就扔的领航系统。他不能和主数据仓库配置基础，但是可以用作培训的工具。关于广业务领航系统要说一句话：它有潜在的巨大成功的可能，会在管理者眼中提高数据仓库的位置，但是注意不要做的太复杂。如果范围太大和复杂，你就可能会失败。

最先开始，扩展种子领航系统可能是最好的选择。而且用户和开发组都可以从这种领航系统获益，用于其可控性和有限的范围，这种领航系统不能包括所有的功能和特征。但是一种领航系统不是详细的描述。只要涉及到所有重要的功能就达到了目的。

扩展和集成领航系统

这个问题出现在你已经将领航系统完成了其主要功能后。领航系统的真正目的和存在期限是什么呢？你一定要将领航系统扔掉吗？所有在领航系统上的努力都浪费了吗？不是的。每种领航系统都有其特定的目的。建立和部署一个领航系统是为了获得预定的结果。概念证明领航系统有一个主要目的，也是唯一目的——证明数据仓库对用户和高层管理的有效性。如果你能借助于领航系统证明它，这个领航系统就是成功的，达到了其目的。

明白领航系统在开发整个数据仓库过程中的位置。领航系统不是初始部署。它可以是初始安装的前奏。不需要太多改动，领航系统可以扩展和集成到整个数据仓库中。参看图 19-5，检查每种领航系统说明它们如何集成到数据仓库中。注意可扩展种子领航系统是集成的最好选择。在每种情况下，观察为了集成需要做哪些事情。

安全

数据仓库是真正的信息金矿。组织的所有关键信息都以简单格式化地存在，可以方便地获取和使用。在单操作型系统中，安全性只对企业地一小部分提供保护，但是数据仓库地安全性涉及到企业数据的绝大部分。而且，安全性必须覆盖从数据仓库中取出的数据和存储在其它数据区域的数据例如 OLAP 系统。

在操作系统中，安全性通过授权保证对数据库的访问。可以使用单个表或者数据视图授权用户访问。在数据仓库中建立访问限制非常难。数据仓库的分析可能从一个或者两个表获得信息开始分析。随着分析的继续，更多的表卷入其中。整个查询过程是非常特别的。哪些表需要限制，哪些表需要为分析开放呢？

安全策略

项目组必须为数据仓库建立安全策略。如果你为保证企业信息集合有一种安全策略，就将数据仓库的安全策略作为企业安全测量的补充。首先，安全策略必须承认数据仓库中的信息的巨大价值。策略必须提供授权指导，建立用户角色。

下面是数据仓库的一般安全策略规定：

- 策略覆盖信息的范围
- 物理安全性
- 工作站安全性
- 网络和连接安全性
- 数据库访问授权
- 数据加载的安全清除
- 统计级别的安全性
- 元数据安全性
- OLAP 安全性
- Web 安全性
- 安全违反的分辩

管理用户权限

你知道，用户被授权访问 OLTP 系统。访问权限和个人或者用户组相联系的有建立、读、更新和删除数据等操作。这些操作的访问限制可能设置在一个单独表的一列或者几列。

大多数的 RDBMS 提供基于角色的安全管理。一个角色就是一组用户访问数据库的共同需要。可以使用数据库管理系统的语言组件执行特定的语句来建立角色。建好角色之后，就可以指定用户特定的角色。访问权限在角色的层次上进行授权。完成之后，所有指定那种角色的用户拥有相同的访问权限。

怎样处理异常。例如，用户 JANE 的角色是 ORDERS。你被授予了角色 ORDERS 具有的特定访问权限。所有这些权限赋予 JANE，除了一个例外。JANE 允许多访问一个表，例如促销维表。怎样处理这样的异常呢？需要单独为 JANE 授权让她可以访问促销维表。其余的，JANE 从角色 ORDERS 获得访问权限。

表 19-6，提供了一些角色，操作和权限示例。观察和数据仓库环境相关的操作和注意权限如何和操作相匹配。

密码

数据仓库中的密码安全保护和操作系统类似。对数据仓库的更新只在数据加载时。用户密码和数据批加载关系不大。删除数据仓库记录也不经常发生。仅当你想归档旧的历史记录，才执行批删除工作。密码主要用来授权用户对数据的只读访问。用户需要密码进入数据仓库环境。

安全管理员应该为密码和密码有效期设置可接受的模式。安全系统将自动使到期的密码失效。用户从管理员获得初始密码后应该更改密码。而且在这个密码失效之前也需要更改密码。还有附加的安全步骤。

在你的公司里执行密码模式规范。密码必须是加密和任意的，不容易被识破。不要让用户使用自己的名字或者爱人的名字作密码。不让用户使用自己奇怪的模式。为密码设定一个规范。在密码中包含文字和数字。

安全机制必须能够记录和控制用户使用无效密码登录的未授权尝试的次数。当达到未授权尝试次数后，必须暂停用户访问数据仓库，直到管理员重新启用用户。成功登录后，非法尝试的次数应该显示出来。如果次数很高，必须汇报。说明有人在别人不在的时候在使用他的工作站。

安全工具

数据仓库环境中，数据库系统的安全组件是主要的安全工具。我们已经讨论了 DBMS 提供的基于角色的安全管理。大多数商业数据管理系统安全包含达到了数据列的层次。

一些组织使用第三方安全管理系统保证所有系统的安全性。如果你的公司是这样的话，利用已经安装的安全系统，将数据仓库置于大的安全伞下。这个安全系统赋予用户一个单一登录特征。用户只需要一个用户 ID 和密码就可以在公司的所有计算机系统中使用。用户不需要为不同系统记住不同的用户 ID 和密码。

一些客户端工具有自己的安全系统。大多数 OLAP 工具有其安全特征。基于工具的安全性组成了安全解决方案的几个部分。一旦你在工具集中的安全系统中建立了用户，你不需要在 DBMS 级再重复，然而一些数据仓库小组也提供 DBMS 安全特性来提供双重保护。

基于工具的安全性，是工具集的集成部分，不能被停用。为了访问数据使用工具集，需要从工具集软件中获得安全清理。如果已经计划使用 DBMS 本身作安全保护，那么基于工具的安全保护就被认为是冗余的。不同提供商的工具有自己不同的信息提示界面。信息以目录、文件夹和项目层次组织起来。必须为这三个级别提供安全认证。

备份和恢复

已经知道了 OLTP 系统的备份和恢复步骤。你们中的一些人，例如系统管理员，负责过系统备份并可能曾进行过一两次灾难恢复。

在任务优先 OLTP 系统中，数据丢失和停机是不能忍受的。数据丢失将导致严重的后果。在一个系统如机票预定或者在线订单处理中，停机很短时间都会导致上百万美元的损失。

数据仓库环境中这些因素有多么苛刻的呢？当订单处理系统停机进行恢复时，你可能有几个小时使用手工恢复。如果机票预定系统停机，就没有手工补回了。数据仓库和 OLTP 相比怎样呢？停机时间很苛刻吗？用户能否忍受少量的数据丢失？

为什么备份数据仓库？

数据仓库的大数据是多年积累的结果。历史记录可能包括 10 年或者 20 年前的数据。当数据进入数据仓库，你知道是经过了很多繁杂的清理和转换过程。数据仓库中的数据代表了企业浓缩和富的历史。这么困难组合在一起的数据，用户不能容忍一点的数据丢失。当灾难发生后，你必须能够重建数据。

当数据仓库停机很长时间，潜在的损失不像可操作的系统那样清楚。订单处理人员不会等待系统恢复。而且，如果分析人员处在主要销售旺季或者时间非常紧迫的时候需要执行重要的分析研究，影响就会非常明显。

观察数据仓库的使用。配置好后的很短时间后，用户数量大量增加。查询类型和分析过程的复杂性加强。用户需要越来越多的报告。通过 Web 技术访问增大。很快，数据仓库将成为核心任务状态。随着越来越多的用户对数据仓库中数据的依赖，备份数据内容和错误快速恢复能力的重要性达到一个新的高度。

在 OLTP 系统中，恢复需要数据各个版本的备份。从最后备份开始恢复到系统停止工作的时间点。但是你可能也知道数据仓库的情况可能和 OLTP 系统不同。数据仓库不是直接数据入口的数据积累。不是源操作型系统首先将数据导入进去的吗？为什么还要对数据仓库建立备份？难道不能从源系统中重新解析和加载数据吗？虽然这看起来是个很自然的解决方法，但是基本上是不实际的。从源系统中重新建立数据需要很长的时间，你的用户不可能忍受如此长的停机时间。

备份策略

现在你已经感觉到数据仓库备份的必需性，有几个问题和需要提出来。数据哪些部分需要备份？什么时候备份？怎样备份？清晰地计划一个定义好的备份和恢复策略。虽然这个策略数据仓库和 OLTP 系统很类似，你仍然需要另外一个策略。你可以基于你们公司已经有的 OLTP 系统的策略建立一个，但是要注意新系统的特殊需要。

一个好的备份策略包括几个重要因素。我们来看几个。下面是包括在你备份策略中一些有用技巧的集合：

- 决定你需要备份什么。为用户数据库，系统数据库和数据库日志列一个表。
- 数据仓库的巨大容量是一个很大的问题。容量问题在备份和恢复策略中占主导因素。高性能是关键因素。
- 寻求一个简单的可管理设置。
- 能够将当前数据和历史数据分离出来，对每个段有分离的步骤。当前段的活动数据随源操作型系统输入而增长。历史或者静态数据是过去的内容。历史数据没有必要备份那么频繁。
- 除了全备份，可以考虑日志备份和差异备份。你知道，日志文件存储了最后的全备份事务或者以前日志文件备份。和这个不同的是一个完整的差异备份。差异备份包括了所以自上次全备份以来的所有改变。
- 不要小看备份系统数据库。
- 介质的选择很重要。这里，数据仓库的容量决定了合适的选择。
- 商业 RDBMS 使用“容器”概念来支持单个的文件。容器是包括很多物理文件的大的存储区域。这些容器例如表空间、文件组等等。RDBMS 有特殊的方法更加高效地备份整个容器。使用这个 RDBMS 特征。
- 虽然 RDBMS 的备份功能为 OLTP 系统服务，数据仓库备份需要更快的速度。从第三方供应商获取备份和恢复工具。
- 从数据仓库中计划周期性地归档非常陈旧的数据。好的归档计划可以减少备份和恢复时间，而且还能提供检索性能。

建立一个实际的日程表

毫无疑问，你需要合适地备份数据仓库。很多用户最终将一直依靠数据仓库获得信息。但是在备份和恢复决策中巨大的容量是非常关键的因素。全备份数据仓库需要很长的时间。在灾难过程中，从源操作型系统中解析和加载数据不能作为一种选择。所有，怎样建立一个实际的备份日程呢？考虑下面几点再做决定：

- 你知道，OLTP 系统备份一般在晚上执行。但是在数据仓库环境中，晚上的时间用于每日的增量加载。备份和加载竞争系统时间。
- 如果你的用户群分布在不同的时区，找一个时间槽可能更麻烦。

- OLTP 系统需要经常备份。在前向恢复过程中，如果没有规则的全备份和经常性的日志文件备份，用户必须手工输入不能恢复的部分数据。和数据仓库比较，这里没有用户的重新输入数据。不能恢复的部分数据，如果可能的话只能从源系统中获取。数据解析和加载系统不支持这类恢复。
- 建立一个实际的日程会遇到下面这些问题。恢复完成前用户可以忍受多长的停机时间？在最坏的情况下用户可以忍受多大的数据损失？直到丢失的数据恢复，数据仓库能够高效地运行较长的时间吗？

一个实际的数据仓库备份日程依赖于你们公司的条件和情况。通常，一个实际的方法包括下面几个元素：

- 将数据仓库分为活动和静态数据。
- 对活动和静态数据建立不同的日程。
- 更经常性地备份活动数据，较少地备份静态数据。
- 包括差异备份和日程备份到备份方案中。
- 同步备份每日的增量加载。
- 如果可用，存储增量加载文件作为恢复的一部分。

恢复

让我们讨论一下恢复过程的几个问题。在这之前，参考图 19-7 说明数据仓库环境下的恢复过程。注意备份文件和在恢复过程中如何使用它们。而且，注意数据丢失的可能性是否存在。下面是一些实际的技巧：

- 有一个清楚的恢复计划。将不同的灾难情况列表，指出每种情况下如何进行恢复。
- 仔细地测试恢复过程。执行日常恢复练习。
- 考虑在你公司的条件，建立恢复步骤，估计恢复的期望停机时间。从用户得到停机时间的一般意见。当第一次灾难发生时，不要使用户惊讶。让他们知道这是整个方案中的一部分，他们需要为可能发生灾难做准备。
- 每次中断期，确定恢复需要多久。让用户快速合适地被通知。
- 通常，你的备份策略决定恢复将怎样被执行。如果计划包括从每日增量加载文件中恢复的可能性，保存这些文件的备份，使之随手可得。

本章总结

- 按照构造阶段，配置数据仓库的第一个版本。
- 配置阶段的主要活动，用户接受、初始加载、桌面准备、初始培训和初始用户支持。
- 领航系统在几种情况下是需要的。领航系统的一般类型是，概念证明、技术证明、综合测试、用户工具鉴别、广业务和可扩展种子。
- 虽然数据仓库的数据安全和 OLTP 系统类似，由于数据仓库数据访问的特点要求提供更加完善访问权限。
- 为什么备份数据仓库？虽然在数据仓库中很少有直接的数据更新，还是有几个需要备份的理由。备份日程和恢复步骤由于数据仓库的数据容量而更加困难。

思考题

1. 列出数据仓库配置的四个主要活动。对其中两个活动，描述其主要任务。
2. 简述用户接受过程，为什么很重要？
3. 什么是初始数据加载的重要考虑？
4. 委身在事实表前加载维表是一个好的惯例？
5. 准备用户桌面的两个一般方法？你喜欢哪种方法？为什么？
6. 初始阶段，用户应该培训哪些课题？
7. 提供领航系统系统的四个一般理由？
8. 什么是概念证明领航系统？什么情况下适合这种领航系统？
9. 列出一个好的安全策略的五个一般保证。
10. 为什么数据仓库应该备份。和 OLTP 系统有何不同？

练习题

1. 判断对错

   A. 在初始加载前删除索引是一个好的习惯。

   B. 事实表的键和维表的键无关。

C．远程配置桌面工具通常比较快。

D． 当用户对数据仓库非常熟悉，需要 pilot 数据集市。

E．因为可以从源系统中恢复数据，备份数据仓库任何情况下都是不必要的。

F． 密码必须是加密和任意的。

G． 总是检查加载作业。

H． 在加载维表前加载事实表是个好的惯例。

I． 初始用户培训必须包括基本数据库和存储概念。

J． 基于角色的安全提供方式不适合于数据仓库。

2. 的数据仓库初始配置准备一个用户桌面准备计划。潜在用户分布在全国 30 个主要中心。海外用户有四个中心点也要访问数据仓库。五个主要区域办公室的分析人员将使用 OLAP 系统。你的数据仓库是基于 WEB 的。作合适的假设，考虑所有条件，给出一个计划。
3. 缓冲数据仓库考虑什么？哪情况下推荐使用缓冲配置？描述你计划如何确定缓冲。
4. 领航系统作为广业务有什么特征？他的优点和缺点是什么？这种领航系统应该考虑吗？解释什么情况下建议使用这种领航系统。
5. 数据仓库管理员准备备份和恢复计划。指出备份方法和日程。浏览恢复选项。描述备份功能范围。你如何确信可以从灾难中恢复？

第二十章　升级和维护

本章目标

- 清楚掌握正在进行的维护和管理的需要
- 理解监视数据仓库的统计集合
- 认识统计怎样用于管理增长和提高性能
- 仔细讨论用户培训和支持功能
- 考虑其它经营和管理问题

现在你处在哪里呢？假设有下面一些似是而非的情况。所有的用户接受测试是成功的。有两种领航系统；一个是用于完成最终用户工具集测试，另外一个是一个可扩展种子领航系统，用于配置。项目组已经成功配置了数据仓库的初始版本。用户都非常高兴。部署后的第一周只有几个小问题。基本上所有的初始用户都已经完整培训过。没有多少 IT 技术支持，他们好像可以自己照顾自己。第一个 OLAP 立方体集合证明了它们的价值，分析人员非常高兴。用户从 Web 上获取报表。所有的艰苦工作都得到了回报。现在怎么样？

现在仅仅是开始。接下来有更多的数据集市和部署工作。项目组需要确信已经准备好了升级。要确信所有的监视功能都各就各位能够一直向项目组报告状态。培训和支持功能需要加强和流水线化。确认所有的管理功能已经完成并能够工作。数据库调节要按常规步骤进行。

初始部署后马上项目组需要做的是评价整个过程。下面是主要评价的几个任务：

- 评价测试过程和提供建议
- 评价领航系统的目标和完成情况
- 调查最初培训过程中使用的方法
- 开发过程中的文档着重点
- 验证初始部署的结果，和用户的期望值比较

评价过程和结果表格组成日后数据仓库性能提高的基础。当扩展和产生下一个版本，让业务需要、模型考察和基本结构因素成为升级的指导因素。和前一个版本紧跟的每个版本，

可以使用前面版本的数据模型。建立每一个版本作为一个逻辑的下一步。避免版本不相联系。在现在的基础上进行构造。

监视数据仓库

实现一个 OLTP 系统，没有随着部署而全部停止。数据库管理员继续监视系统性能。项目组继续监视新系统是否和需求相符，获得结果。监视数据仓库和 OLTP 系统比较类似，除了有一个较大的不同。OLTP 系统监视和数据库环境相比要小。可以很容易察觉到，数据仓库的监视活动范围包括很多特征和功能。除非数据仓库监视以一种正规形式进行，就不可能获得预期的结果。监视的结果为升级计划和性能提高提供了数据。

图 20-1 给出了数据仓库监视的活动和意义。可以观察到，统计是监视活动的生命血液。它将导致数据仓库的升级计划和性能优化。

统计采集

我们所说的监视统计是指标，代表了数据仓库的功能信息。这些指标提供硬件和软件资源的使用情况。从这些指标中，可以确定数据仓库工作的怎么样。指标指示增长趋势。明白服务器功能怎样。看到终端用户工具的使用情况。

怎样在工作中的数据仓库中进行统计信息采集。通常有两种针对采集过程的方法。采样方法和事件驱动方法。采样方法是定时测量系统活动的特定指标。可以设置采样时间。如果监视进程工具的采样时间为 10 分钟，统计记录每 10 分钟产生一次。采样方法对系统本身影响较小。

事件驱动方式不一样。统计记录不是按时进行，而是特定事件发生时才记录。例如，如果监视索引表，可以设置监视机制记录索引表更新事件。事件驱动方式增加系统负载但是比采样方法更彻底。

使用什么工具收集统计信息？数据库服务器和主机操作系统所带的工具通常用来收集监视统计数据。除了这些，很多第三方提供商提供了在数据仓库环境中很多特别有用的工具。大多数工具收集指标值并且解释结果。数据收集组件收集统计信息，分析组件进行解释。很

多系统的监视是实时的。

让我们注意一下这些有用的监视统计。以下无序列表包括不同用途的统计。你会发现大多数可以应用于你的环境中。

- 物理磁盘存储空间使用。
- DBMS 在块中寻找空间或者导致碎片的次数。
- 内存缓冲区活动。
- CACHE 缓存使用。
- 输入输出性能。
- 内存管理。
- 数据仓库内容，不同入口发生的次数（例如：用户个数，产品个数等）。
- 每个数据表的大小。
- 事实表记录的访问。
- 主体域相关的使用统计。
- 每日按时间槽完成的查询个数。
- 数据仓库每个用户在线的时间。
- 每日不同用户的总数。
- 每日时间槽内最大用户个数。
- 每日增量加载持续时间。
- 有效用户计数。
- 查询响应时间。
- 每日生成报告的个数。
- 数据库中活动表的个数。

为升级划使用统计

当你为数据仓库部署更多的版本，用户数量的增长和查询复杂性的加强，你不得不考虑这些明显的增长。但是你怎么知道哪里需要扩展？为什么查询变慢了？为什么响应时间变长？为什么数据仓库扩展表空间失败？监视统计为你提供数据仓库情况的线索，指导你如何为升级作好准备。

我们下面给出监视统计提示我们应该采取的行动：

- 为已存在的表分配更多的磁盘空间。
- 为附加的表计划更多的磁盘空间。
- 修改文件块参数最小化碎片。
- 建立更多的统计表处理大量对统计信息的查询。
- 重组缓冲区文件以处理更大的数据量。
- 为加强缓冲管理增加更多的缓存。
- 升级数据库服务器。
- 将报表生成使用其它中间件代替。
- 24 小时周期中消除高峰时间的使用。
- 通过对表进行分区，进行并行加载和备份

为优化使用统计

统计的另外一个最好的用途和性能相关。你会发现大量的监视统计被证明对数据仓库的性能优化非常有用。下面的章节我们将仔细讨论这些细节。现在，我们给出一些可以基于统计信息而使性能提高的数据仓库功能：

- 查询性能
- 查询格式
- 增量加载
- OLAP 加载频率
- OLAP 系统
- 数据仓库内容浏览
- 报表格式化
- 报表生成

为用户公布趋势

这是一个在 OLTP 系统中不常见的新概念。在数据仓库用户必须自己深入系统，获得信

息。他们必须知道内容。用户必须知道数据仓库中的数据趋势。最后一次增量加载是什么时候？什么是主题域？不同入口的次数多少？OLTP 系统完全不一样。这些系统为用户准备了模式和标准化的信息。OLTP 系统用户不需要内部视图。参见图 20-2，列出了必须为用户公布的统计类型。

如果你的数据仓库是基于 Web 的，使用公司内部网为用户公布统计信息。另外，提供深入查询查询统计所在的数据集能力。

用户培训和支持

你的项目组可能已经建立了最好的数据仓库。从源系统中解析数据并且非常仔细计划和设计好了。转换功能满足所有的需求。缓冲区设置的很好，支持任何工作在这里执行。向数据仓库中加载数据没有任何问题。你的最终用户有高效获取数据的工具，这些工具非常适合他们的需要。数据仓库的每个组件工作地非常好。什么事情都各就各位并且工作地非常好，但是如果用户没有好的培训和支持，项目组的努力将付诸东流。这将是巨大的失败。不能低估用户培训和服务支持的意义，初始时和正在进行时都不能。

确实，当项目组选择工具时，可能一些用户接受了这些工具的最初培训。这些不能代替正规的培训。你必须建立一个培训计划，考虑用户必须培训的所有领域。在初始阶段，和接下来部署第一个版本时，用户需要在支持下进行。不要忽略建立一个有意义且有用的支持系统。你知道在 OLTP 系统实现中的技术和应用支持。对于数据仓库，因为工作是不同并且全新的，好的支持更加重要。

用户培训内容

用户应该培训什么？什么是重要和必需的？使期望的应用和培训的内容相匹配。每个用户组怎样和数据仓库打交道？如果一组用户经常使用预定义查询和预格式化报表，那么这些用户的培训比较简单。然而如果另一组分析人员需要格式化他们自己的特别的查询来进行分析，那么对分析人员的培训计划就必须加强。

设计用户的教学内容，要使内容更广更深。记住，来培训的用户技术能力和知识层次不一样。通常，准备使用数据仓库的用户有基本的计算机技能知道计算机如何工作的。但是对

大多数用户，数据仓库简直是天方夜谭。

我们重复一下上一章提到的内容。在培训计划中，三个重要的部分必须需要提供。其一是用户必须掌握你们能从数据仓库里得到什么。他们必须清楚数据内容和如何获得数据。其二，你必须告诉用户关于应用的情况。预建立的应用是什么？他们可以使用预定义的查询和报表吗？怎样使用？最后，你必须训练用户获得数据的工具的使用。知道了这些，不要让用户理解成是培训项目分成了三个部分，数据内容、应用和工具。绝对不要将培训计划分成三个分离的不相关的三个部分，而只是作为整个培训计划的一个潜在主线。参见图 20-3 显示培训计划包括的主要课题。此图包括了三个课题，数据内容、应用和工具，要保证每一个都不能忽视。在准备培训计划的课程提纲时，让这三个课题覆盖课程内容的每个部分。

准备培训计划

一旦确定了课程内容，你就要准备培训计划本身。确定必需的准备。首先，项目组必须确定培训计划的类型，然后建立每种类型的课程内容。下一步，确定谁负责准备课程内容资料。组织课程资料的实际准备。然后是培训培训者。大量的精力用于组合成一个培训计划。不要小看准备一个好的培训计划。

让我们看看需要准备一个培训计划的各种任务。每个公司需要不同的培训计划。下面是组织一个优秀的用户培训计划的通常技巧。

- 成功的培训计划依赖于用户代表和 IT 技术人员的集体参与。本项目组的用户代理和用户部门的本领域专家都适合于和 IT 技术人员一起工作。
- 让 IT 技术人员和用户一起工作准备课程内容。
- 记住包括的课题，数据内容、应用和工具使用。
- 给出一个当前所有需要培训用户的列表。将用户按照知识和技术等级分成逻辑组。确定每个组需要培训什么。做了这些工作，可以让培训计划更与公司的需要相匹配。
- 确定多少不同的培训过程会实际对用户有用。一个好的课程包括介绍性课程、深入课程和工具使用的特别课程。
- 介绍性课程通常有一天。每个用户必需参加这个基础课程。
- 在深入性课程中有几种途径。每种途径满足特定用户组，着重于一到两个主题。
- 工具使用的特别培训也有一些变化，依赖于不同的工具集。OLAP 用户有自己的课

程。

- 保持课程文档简单和直接而且包括足够的图表。如果本课程包括维建模，STAR 方案示例可以帮助用户形象化其关系。不要进行一次没有任何课程资料的培训。
- 你可能已经知道，动手过程可能更有效。在介绍课程中只有一个演示。另外两种课程将进行动手训练。

课程是怎样组织起来的？每种课程的主要内容是什么？让我们看看几个课程大纲的示例。图 20-4 给出三个示例大纲，每种课程对应一个。将这些大纲作为指南。根据你们公司的需要修改大纲。

数据内容	应用	工具
数据仓库中的主题	预定义查询	终端用户工具特征和功能
数据仓库的事实表和维表	查询模板	与数据仓库元数据交互的工具
数据仓库导航	预格式化报表	登录工具软件的步骤
数据粒度和聚集表	性能报表	导航和浏览数据仓库工具使用
源系统和数据解析	报表打印选项	格式化查询和结果的工具使用
数据转换和清除规则	数据进入下层应用	报表工具使用
商业术语和意义	预开发应用	
	OLAP 统计和多维分析	
	执行信息系统	

执行培训计划

培训计划必须在部署数据仓库第一个版本前准备好。在部署前安排第一批用户的培训。用户从培训中学到的东西对他们来说是新鲜的。第一批用户认识到数据仓库的有用性可以有效保证其成功实现，所以要对第一批用户慎重对待。

接下来的培训继续为其它的用户服务。当实现数据仓库的下一个版本，修改培训资料并且继续提供这些课程。最初你会意识到你需要对课程进行完整的安排。一些用户可能需要更新课程。记住用户对他们的业务有责任。他们应该抽出时间配合培训时间表。

由于有关培训的紧张的工作，特别如果你有大量的用户群，就需要一个培训管理员的专

门服务。管理员安排课程，课程的讲师，确认培训资料已经准备好，安排培训地点，仔细计算动手训练需要的资源。

你对高级执行官和高级管理人员必须培训哪些东西呢？在 OLTP 系统，高级管理和执行人员很少需要坐在桌面电脑前深入系统。但在数据仓库系统中就不一样了。这种新环境支持各种决策制定，特别地对于高层人员。当然，高级官员不需要知道怎样执行每一个查询，生成每个报表。但是他们需要知道怎样寻找他们感兴趣的信息。大多数对此感兴趣的高级管理人员不愿意和其它员工一起接受这样的培训。你需要为这些执行者安排单独的培训课程，有时候是一对一的培训。你要修改介绍性课程，为这些执行者提供另外特别的培训。

虽然进行了培训，你会发现还有其它一些需要使用数据仓库的用户没有培训到。一些用户太忙了而不能离开工作岗位。一些分析人员和能力较强的用户可能觉得他们不需要进行正式的培训而可以自学。你们公司针对这些情况必须有明确的制度。当你允许一个没有经过一点培训的人进入数据仓库，有两种情况经常发生。一是，他们由于占用过多的注意力将破坏支持体系。二是，当他们不能执行一些功能或者解释一些结果，他们不说是自己缺乏培训而会责备系统。通常，“没有培训，不能使用数据仓库”的制度是有效的。

用户支持

用户支持开始于第一个用户进入数据仓库的鼠标的第一下点击。这不是笑话，而是强调用户支持的重要性。你知道，缺少好的用户支持，用户的挫折感慢慢增强。必须在部署第一个数据仓库版本前建立好支持体系。如果你有一个领航系统计划或者一个早的可交付时间表，确信用户可以向支持系统求援。

作为一个 IT 专业人士已经工作过其它的实现和正在进行的 OLTP 系统，你清楚支持功能是如何操作的。就让我们不要试图重复你们已经熟悉的东西。我们只看看支持系统的两个环节。让我们给出一个数据仓库环境中的分层的用户支持结构。参见图 20-5。本图说明用户支持功能的组织。注意不同的层次。注意每个用户组的用户代表如何作为接触的第一点。

现在一些与数据仓库环境支持相关的要点。注意一下的一些要点：

- 让每个用户知道支持途径。每个用户必须知道首先联系谁，硬件问题先要找谁，工具软件的问题找谁，等等。
- 在一个多层支持结构，清楚哪一层支持哪些功能。

- 如果可能，将数据仓库支持和整个公司的支持统一起来。
- 在数据仓库环境，你需要另外一种支持。经常，用户想将从数据仓库中获得的信息和源系统中获得的信息对应起来。你们支持体系的一部分能够解决这些数据对照问题。
- 经常，至少在刚开始，用户需要知道浏览整个数据仓库系统内容的整个线索。为这类支持作一个计划。
- 包括如何找到和执行预定义查询和预格式化报表的支持。
- 用户支持可以作为有效的渠道以鼓励在其它部门获得成功的用户，获得他们关心的特殊问题的反馈。确保通信和反馈渠道畅通。
- 大多数企业从为数据仓库支持提供一个公司网站获益。可以公布数据仓库的一般信息，预定义查询和报告，用户组，新版本，加载时间表和 FAQ。

管理数据仓库

在部署完数据仓库的第一个版本之后，该轮到管理功能了。直到现在，在整个数据仓库开发生存期中还在下面一些方面。设计、构造、测试、用户接受和部署。现在，数据仓库关心的是两个主要功能。第一个是支持管理。数据仓库管理员必须使所有的功能以最佳状态运行。第二是修改管理。随着新版本的部署，工具有了新的发布，ETL 功能的提高和自动化，管理组的焦点包括功能增强和修改。

本节，让我们看看数据仓库管理的几个重要方面。我们将指出主要因素。部署后管理包括下面一些方面：

- 性能监视和优化
- 数据增长管理
- 网络管理
- ETL 管理
- 未来数据集市发布管理
- 数据获取增强
- 安全管理
- 备份和恢复管理

- Web 管理
- 平台升级
- 继续培训
- 用户支持

平台升级

你的数据仓库部署平台包括基础结构，数据传输组件，终端用户信息获取，数据存储，元数据、数据组件和 OLAP 系统组件。经常地，数据仓库是综合的跨平台环境。组件按相关的路径进行，从底层的计算机硬件，到接下来的操作系统、通信系统、数据库、GUI 和应用支持软件。随着时间推移，提供商就会声明需要升级这些组件。

在初始部署之后，有一个恰当的计划采用平台组件的新的版本。你可能对 OLTP 比较有经验，升级对正常工作有潜在的严重中断，除非他们能正确地管理。好的计划最小化中断。提供商试图强迫你掉入他们升级计划的陷阱里。如果时间不太方便，拒绝提供商的主动。在你方便的时候计划升级，而且要在你的用户可以承受中断时进行。

管理数据增长

管理数据增长需要特殊的重视。在数据仓库中，除非你警惕数据增长，否则可能会很容易非常快地失去控制。数据仓库已经有巨大的数据量。起始于巨大容量的数据，即使很少比例的增加产都会导致大量数据的产生。

首先，数据仓库可能包括太多的历史数据。对很多公司来说由于业务条件的改变，10 年前的数据可能不会产生有意义的结果。终端用户倾向于选择最小的粒度来保存明细数据。至少在初始时期，用户继续对比数据仓库结果和操作型系统产生的结果。分析人员在分析过程中生成很多类型的统计。经常地，分析人员想存储这些中间数据集以用于将来相似的分析中。无计划的中间数据集加速了数据量的增长。

下面是一些管理数据增长的实际建议：

- 分散明细数据级别，而使用汇总表代替。
- 约束不必要的 drill-down 功能，去掉对应的明细级别数据。

- 限制历史数据的容量，定期归档旧数据。
- 不鼓励分析人员保存无计划的汇总。
- 如果真的需要，建立附加的汇总表。

存储管理

随着数据增长，存储的使用也增长。因为数据仓库的大容量数据，存储花费在整个费用中占用非常高。专家估计存储成本是软件成本的四到五倍，然而你会发现存储管理没有引起数据仓库开发和管理人员足够的重视。下面是存储管理的一些技巧可以作为指南：

- 每次数据仓库的新版本初始测试需要更多的存储空间。对增长要有计划。
- 确信存储配置是弹性和伸缩性的。必须能够增加更多的存储而最小地中断当前的用户。
- 使用模块化存储系统。如果没有使用，使用一个转换器。
- 如果在分布式环境中有多个服务器有各自的存储池，考虑将服务器连接到一个单一可以智能访问的存储池。
- 随着使用的增加，计划分布数据到多个卷以最小化访问瓶颈。
- 确信可以从坏的存储扇区里转移数据。
- 寻找可以优化的存储系统以避免存储损耗。

ETL 管理

这是一个主要的持续性的管理功能，所以要尽量自动化完成。安装一个报警系统当出现异常情况时能获得重视。下面的建议对 OEL（data extraction，transformation，loading）管理很有用：

- 定时运行每日的解析作业。如果在特殊情况下源系统不可获得，重新安排解析作业。
- 如果使用了数据复制技术，保证复制进程结果正确。
- 保证源系统记录个数和解析出的文件记录个数的所有的对比完成。
- 确定所有已定义的数据转换和清理工作可以正确地进行。
- 解决转换和清理功能出现的异常情况。

- 验证加载映像建立过程，包括为维表和事实表记录合适键值。

数据模型更新

当你在将来扩展了数据仓库，数据模型就会发生更改。如果下一个版本包括一个新的主题数据集市，那么模型就将扩展并包括新的事实表、维表和一些聚集表。同样，物理模型也改变了。必须进行新的存储分配。版本更新对整个数据模型有怎样的影响呢？下面是部分列表，可以根据你的数据仓库环境加以扩充：

- 元数据更新
- 物理设计更改
- 增加的存储分配
- ETL 功能更新
- 增加的预定义查询和预格式化报表
- OLAP 系统更新
- 增加的安全系统
- 增加的备份和恢复系统

信息发布加强

随着时间的推移，你的用户不再局限于刚开始使用的最终用户工具了。在这时候，用户定位和使用数据更加专业。他们已经准备好了越来越多的复杂查询。市场上新的终端用户工具层出不穷。如果这些工具确实对他们有用，为什么不让你的用户使用最新最好的工具呢？

增强终端用户工具和采用一种新的工具集有什么关系呢？不像对 ETL 的改变，这些改变和用户直接相关，所以要仔细计划改变，谨慎地执行。

请看下面一些技巧：

- 保证新工具集和数据仓库组件的兼容性。
- 如果新工具集对已存在的工具是新增的，给用户一段过渡时间。
- 保证终端用户元数据的集成。
- 安排新工具集的培训。

- 如果原始工具集有附加的存储数据，计划将这些数据转移到新工具集。

持续的优化

作为 IT 专业人员，你熟悉 OLTP 系统的优化技术。这些技术也可以用于数据仓库优化。这些技术基本是相同的，除了有一个最大的不同：数据仓库包括更多一些，实际上，是 OLTP 系统很多倍的数据量。这些技术将用于一个充满数据山峰的环境里。

没有必要重复索引和其它技术，这和 OLTP 环境没有什么不同。让我们看看几个实际的建议：

- 制定一个检查索引使用的日常安排。删除不再使用的索引。
- 每天监视查询性能。调查长时间运行的查询。和执行长时间查询的用户组一起工作。如果需要，就建立索引。
- 定期分析预定义查询的执行。RDBMS 有对应的查询分析器完成这项工作。
- 观测每日不同时间的加载分布。确定变化巨大的原因。
- 虽然你已经为优化建立了持续的日常安排，有时候，你会碰到导致突然不幸的一些查询。你会听到一些特别用户群的抱怨。准备好这些特别的优化需要。数据管理组必须有人抽出时间处理这些情况。

本章总结

- 紧跟着初始部署，项目组必须执行检查过程。
- 持续监视数据仓库需要不同指标的采集和统计。使用这些统计制定增长计划和优化。
- 用户培训包括确定需要培训的内容，准备培训计划，执行培训计划。
- 用户支持需要按层次划分以满足不同的支持需要，数据内容、应用和工具。
- 持续性运营管理包括一下几个方面：平台升级，数据增长管理，存储管理，ETL 管理，数据模型更新，信息获取加强和持续优化。

思考题

1. 列出为监视数据仓库功能需要的统计的类型。
2. 描述采集统计提出的对付增长计划的六种方式。
3. 数据仓库中统计怎样对优化有用的？
4. 你认为给用户公布统计和类似数据有用吗？如果是，为什么？
5. 用户培训内容的三个主题是什么？为什么这些重要？
6. 描述准备培训计划的任意四项任务。
7. 培训管理员的责任是什么？
8. 你认为多层用户支持体系适合于数据仓库吗？有可替换的方式吗？
9. 公司内部网在用户支持上可以扮演怎样的角色？
10. 列出 ETL 管理的任意五个因素。

练习题

1. 为部署后的数据仓库建立一个统计采集步骤。列出所有需要采集的统计。指明每种统计将怎样用于后来的支持和性能提供中。
2. 你被指派提高六个月前部署的数据仓库的性能。你对这次任务的计划是什么？你需要哪些类型的统计？建立详细的计划。
3. 你被聘请为一个全国汽车代理数据仓库的培训经理。数据仓库是基于 Web 的。总部的分析员使用基于 MOLAP 模型的 OLAP 系统工作。建立一个渐进的培训计划。描述计划的重点。
4. 回顾图 20-5 所示的用户支持体系。这个结构是否满足一个基于 Web 的数据仓库，有大约 250 名内部用户和 150 名业务伙伴用户使用本数据仓库。大约 20 名全职分析人员使用 OLAP 系统做复杂的分析和研究。你想对支持结构作哪些提高？指出重点。
5. 作为数据仓库项目经理，给 CIO 写一个项目完成计划和执行计划责任人。列出完成的主要工作。提出未来版本发布的过渡部署计划。指出持续支持计划。简要并着重于增长和维护两个主题。

附录 A

项目生存期步骤和列表

数据仓库项目生存期：

主要步骤和子步骤

注意：这里指出的子步骤是在高层次上的。扩展这些子步骤以适应于你们环境的需要。

项目计划

- 定义范围，目标，目的和期望
- 建立实施策略
- 项目资源的初步鉴定
- 组建项目组
- 估算项目安排

需求定义

- 定义需求搜集策略
- 与用户进行对话
- 回顾已有的文档
- 研究源系统
- 获得分析人员需要的业务规则和维

设计

- 设计逻辑数据模型
- 定义数据抽去、转换和加载功能
- 设计信息获取框架
- 建立存储需求
- 定义整体架构和

构造

选择和安装基本硬件和软件

选择和安装 DBMS

选择和安装 ETL 和信息发布工具

完成物理模型设计

完成元数据组件

部署

完成用户接受测试

初始数据加载性能

准备数据仓库的用户桌面

培训和支持最初用户群

保证数据仓库安全、备份和恢复

维护

持续监视数据仓库

继续性能调节

继续用户培训

为用户继续提供支持

持续的数据仓库管理

主要工作列表

注意：将下面所列工作细分。准备你自己的每个主要工作任务列表。

项目计划

完成项目范围和目标定义

准备项目初始文档

同赞助商和用户代表进行最初会谈

完成项目组成员选拔

准备项目安排

完成项目计划文档

需求定义

完成面谈和组会议汇总

从源系统数据中汇总可以获得的任何信息

用户组文档信息交付需求

与用户讨论有关系统性能问题

完成信息包图表显示度量和维

准备和公布需求定义文档

设计

完成逻辑模型的 E-R 图和维图表

记录数据解析和数据传输功能

估计数据存储需求

用户分类确定每类用户的信息获取规范

记录所有类型工具需要的功能

为内部程序，例如传输，加载和信息获取，创建设计规范

构造

安装和测试基础硬件和软件

安装测试选择的 DBMS

安装测试选择的工具

完成物理数据模型和存储分配

测试初始和持续的数据加载

完成整个系统测试

部署

完成用户接受文档

加载初始化数据

安装测试客户端需要的所有软件

培训最初用户

为不同等级的用户分配数据仓库授权

建立备份和恢复步骤

维护

为持续的监视建立步骤获得统计

需要优化数据仓库

完成用户培训资料建立培训计划

建立用户支持机制

部署数据库管理步骤

不是按照需要升级基础设施

附录 B

成功的关键因素

- 除非你们公司已经准备好了，否则不要上马数据仓库。
- 选择最好的执行发起人。保证持续、长期和忠诚的支持。
- 强调项目的业务方面，而不是技术方面。选择熟悉业务的项目经理。
- 从整个企业角度看需求。
- 有一个实际的，分段实施计划。
- 和用户交流实际的期望，信守承诺。
- 不要超出成本预算和预期投资汇报率。
- 建立合适有效的交流手段。
- 整个项目生存期，将项目作为 IT 人员和用户之间的结合点。
- 采用被证明过的技术；避免过分超前的技术
- 知道数据质量的重要性。
- 不要忽视从外部源获得的潜在数据。
- 不要低估花在数据解析、转换和加载（ETL）功能上的时间和精力。
- 选择适合于你环境的架构；数据仓库不是一个万能的提议。
- 架构第一，技术其次，然后才是工具。
- 确定一个清晰的培训策略。
- 警醒“分析麻痹”
- 开始一个合适可见的 pilot 部署以获得早期回报。
- 不要忽视伸缩性的重要性。为增长和进化做好计划。
- 着重查询而非事务处理。数据仓库是以查询为中心的，不是基于事务的。
- 清楚定义和管理数据所有权的考虑。

附录 C

评价厂商解决方案指南

下面是几个实际的指导。这些是关于所有类型产品，可以执行多种功能，从数据抽取析到信息发布。采用这些建议到你自己特定的环境中。

- 首先，确定你的数据仓库功能确实需要厂商提供工具和解决方案。
- 对你需要的每种产品，仔细列出需要的各种功能特性。将这些功能按照重要性高、中和低分类。这些类型功能来评分你考虑的产品。
- 分配充足的时间来彻底研究可能的解决方案和厂商。
- 如果你想从很多不同的厂商找到解决方案，你要准备不兼容和集成限制的严重挑战。选择两到三个比较适合你需要的厂商的产品。
- 元数据是数据仓库的关键组件。保证你选择的厂商产品可以很好地处理元数据。
- 提供商的持久和稳定性和产品本身的有效性一样重要。甚至当这些产品适合于你的环境，如果你关心的是厂商的持久性，选择这些产品的适合就需要再考虑考虑。
- 不要仅仅依靠厂商的演示作为你选择的基础，也不要仅仅检查和厂商自己提供的参考信息。
- 在你的环境中测试工具和产品，使用你自己数据的子集。
- 安排用户代表和项目组内 IT 专业人员测试这些产品，独立又互相联系。
- 建立明确的方式比较和打分竞争的产品。你可能需要一个打分系统在一类产品中为你寻找的各种功能打分。
- 你的数据仓库的成功依靠最终用户工具。特别注意最终用户工具的选择，你可以在其它工具上稍微马虎一点，但绝对不是最终用户工具。
- 你的大多数最终用户对数据仓库都是新手。好的、直观和易于使用的工具将赢得他们。
- 用户喜欢能够完美集成在线查询、批处理报表和数据抽取以供分析的工具。