第3章 数据仓库

自从1991年数据仓库之父Bill Inmon提出数据仓库的概念以来，数据仓库已从早期的探索阶段走向实用阶段，也进入了一个快速发展的阶段。在此期间，全球经济急速发展，激烈的竞争、企业间频繁的兼并重组，使企业对信息的需求大大加剧，这是数据仓库发展的根本原因。当越来越多的企业开始重视数据资产的价值时，数据仓库就成为必然的选择。

目前，企业面对经济增长减缓和市场竞争日益激烈的双重压力，为继续保持经济的高速稳定增长，大量的企业面临着减员增效、股份制改造等各种变革，准确、全面的信息是企业变革制胜的法宝。随着经营策略从以产品为中心转变为以顾客为中心，数据的潜在价值正在得到越来越多的关注，企业已经认识到充分地利用信息是应对挑战的关键，数据仓库正在成为IT领域中被关注的热点技术。

信息技术的广泛应用使企业的运营更加高效、灵活，但同时也带来了“数据爆炸”的问题，许多遗留下来的历史数据被束之高阁，人们面对浩如烟海的数据显得手足无措，如何有效地组织和存储数据，把其内部隐藏的信息转化为商业价值，为企业效益提供服务成为决策者们迫切关心的问题。数据仓库作为高效集成、管理数据的技术，为各级决策者洞察企业的经营管理状况，及时发现问题，提高决策水平提供了基础。目前数据仓库逐渐被越来越多的企业应用。

3.1 从数据库到数据仓库

企业的数据处理大致分为两类：一类是操作型处理，也称为联机事务处理(On-Line Transaction Processing，OLTP)，它是针对具体业务在数据库联机的日常操作，通常对少数记录进行查询、修改。用户较为关心操作的响应时间、数据的安全性、完整性和并发支持的用户数等问题。传统的数据库系统作为数据管理的主要手段，主要用于操作型处理。另一类是分析型处理，一般针对某些主题的历史数据进行分析，支持管理决策。

企业经过数年的信息化建设，数据库中都会积累大量的日常业务数据，传统的决策支持系统(DSS)直接建立在这种事务处理环境上。然而传统的数据库对分析处理的支持一直都不能令人满意，这是因为操作型处理和分析型处理具有不同的特征，主要体现在以下几个方面。

(1) 处理性能。日常业务涉及频繁、简单的数据存取，因此对操作型处理的性能要求较高，需要数据库能够在很短时间内做出响应。与操作型处理不同，分析型处理对系统响应的

要求并不那么苛刻。有的分析甚至可能需要几个小时，耗费了大量的系统资源。

(2) 数据集成。企业的操作型处理通常较为分散，传统数据库面向应用的特性使数据集成困难。数据分散，缺乏一致性，外部数据和非结构化数据的存在使得企业很难得到全面、准确的数据。而分析型处理是面向主题的，经过加工和集成后的数据全面、准确，可以有效支持分析。

(3) 数据更新。操作型处理主要由原子事务组成，数据更新频繁，需要并行控制和恢复机制。但分析型处理包含复杂的查询，大部分是只读操作。过时的数据往往会导致错误的决策，因此对分析型处理数据需要定期刷新。

(4) 数据时限。操作型处理主要服务于日常的业务操作，因此只关注当前的数据。而对于决策分析而言，对历史数据的分析处理则是必要的，这样才能准确把握企业的发展趋势，从而制定出正确的决策。

(5) 数据综合。操作型处理系统通常只具有简单的统计功能。操作型处理积累了大量的细节数据，对这些数据进行不同程度的汇总和聚集有助于以后的分析处理。

总的来说，操作型处理与分析型处理系统中数据的结构、内容和处理都不相同。Bill Inmon 归纳了操作型处理与分析型处理的区别，如表 3.1 所示[1]。

表 3.1 操作型处理与分析型处理的比较

操作型处理	分析型处理
细节的	综合的或提炼的
实体-关系(E-R)模型	星形或雪花模型
存取瞬间数据	存储历史数据，不包含最近的数据
可更新的	只读，只追加
一次操作一个单元	一次操作一个集合
性能要求高，响应时间短	性能要求宽松
面向事务	面向分析
一次操作数据量小	一次操作数据量大
支持日常操作	支持决策需求
数据量小	数据量大
客户订单、库存水平和银行账户查询等	客户收益分析、市场细分等

从上面的分析可见，操作型数据库面向企业日常的数据处理，其用户是企业的业务人员，难以支持复杂的数据分析。而分析型处理是面向分析，支持决策的，用户多是企业的各级管理人员，通过对企业运营的历史数据进行分析，得到信息、知识以辅助决策。

3.2 数据仓库的概念

从本质上讲，设计数据仓库的初衷是为操作型系统过渡到决策支持系统提供一种工具或整个企业范围内的数据集成环境，并尝试解决数据流相关的各种问题。这些问题包括如何从传统的操作型处理系统中提取与决策主题相关的数据，如何经过转换把分散的、不一致的业务数据转换成集成的、低噪声的数据等。

Bill Inmon 认为数据仓库就是面向主题的(subject-oriented)、集成的(integrated)、非易

失的(non-volatile)和时变的(time-variant)的数据集合,用以支持管理决策[1]。数据仓库不是可以买到的产品,而是一种面向分析的数据存储方案。对于数据仓库的概念,可以从两个层次进行理解:首先,数据仓库用于支持决策,面向分析型数据处理,不同于提高业务效率的操作型数据库;其次,数据仓库对分布在企业中的多个异构数据源集成,按照决策主题选择数据并以新的数据模型存储。此外,存储在数据仓库中的数据一般不能修改。数据仓库主要有以下特征。

1. 面向主题

在操作型数据库中,各个业务系统可能是相互分离的。而数据仓库是面向主题的。逻辑意义上,每一个商业主题都对应于企业决策包含的分析对象。如图 3.1 所示,一个保险公司的数据仓库的主题可能有顾客、政策、保险金和索赔等。

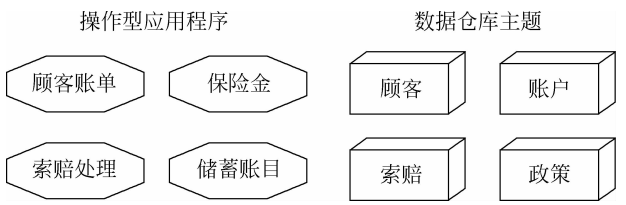

图 3.1 数据仓库的主题

操作型数据库对数据的划分并不适用于决策分析。而基于主题组织的数据则不同,它们被划分为各自独立的领域,每个领域有各自的逻辑内涵但互不交叉,在抽象层次上对数据进行完整、一致和准确的描述[2]。一些主题相关的数据通常分布在多个操作型系统中。

2. 集成性

不同操作型系统之间的数据一般是相互独立、异构的。而数据仓库中的数据是对分散的数据进行抽取、清理、转换和汇总后得到的,这样就保证了数据仓库内的数据关于整个企业的一致性。图 3.2 说明了一个保险公司综合数据的简单处理过程,其中数据仓库中与“保险”主题有关的数据来自多个不同的操作型系统。这些系统内部数据的命名可能不同,数据格式也可能不同,把不同来源的数据存储到数据仓库之前,需要去除这些不一致。

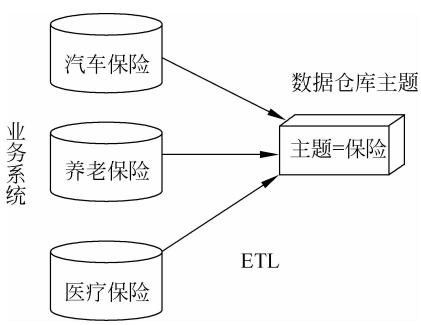

图 3.2 数据仓库的数据集成

3. 数据的非易失性

操作型数据库主要服务于日常的业务操作,使得数据库需要不断地对数据进行实时更新,以便迅速获得当前的最新数据,不致影响正常的业务运作。在数据仓库中只需要保存过去的业务数据,不需要每一笔业务都实时更新数据仓库,而是根据商业需要每隔一段时间把一批较新的数据导入数据仓库。事实上,在一个典型的数据仓库中,通常不同类型数据的更新发生的频率是不同的。例如,产品属性的变化通常每个星期更新一次,地理位置上的变化

通常每个月更新一次，销售数据则每天更新一次。

数据的非易失性主要是针对应用而言的。数据仓库的用户对数据的操作大多是数据查询或比较复杂的挖掘，一旦数据进入数据仓库以后，一般情况下都被较长时间地保留。数据仓库中一般有大量的查询操作，但修改和删除操作很少。因此，数据经加工和集成进入数据仓库后是极少更新的，通常只需要定期加载和更新。

4. 数据的时变性

数据仓库包含各种粒度的历史数据。数据仓库中的数据可能与某个特定日期、星期、月份、季度或者年份有关。数据仓库的目的是通过分析企业过去一段时间内业务的经营状况，挖掘其中隐藏的模式。虽然数据仓库的用户不能修改数据，但并不是说数据仓库的数据是永远不变的。分析的结果只能反映过去的情况，当业务发生变化后，挖掘出的模式就会失去时效性。因此数据仓库的数据需要更新，以适应决策的需要。从这个角度来讲，数据仓库建设是一个过程[3]。数据仓库的数据随时间的变化表现在以下几个方面。

(1) 数据仓库的数据时限一般要远远长于操作型数据的数据时限。

(2) 操作型系统存储的是当前数据，而数据仓库中的数据是历史数据。

(3) 数据仓库中的数据是按照时间顺序追加的，它们都带有时间属性。

数据仓库主要包括数据的提取、转换与装载(ETL)、元数据、数据集市和操作数据存储等部分，常用的数据仓库结构如图 3.3 所示[4]。

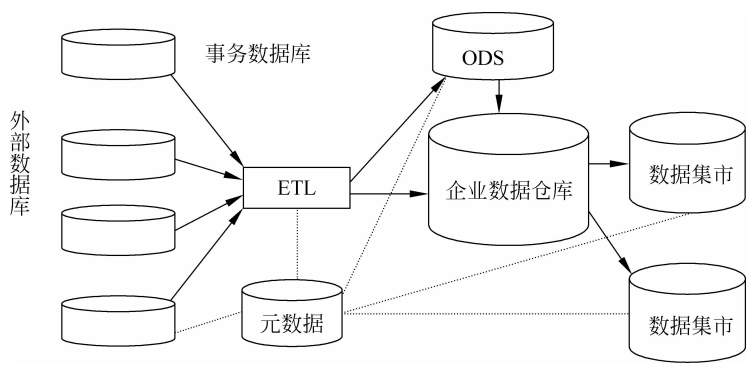

图 3.3　数据仓库的结构

3.3　数据集市

人们在早期开发企业级数据仓库时，一般是先建立一个全局的数据仓库，然后在此基础上建立各种应用，即采用“自顶向下”的方法。但在开发的过程中会出现以下问题。

(1) 如果按“自顶向下”的方法建立企业级数据仓库，建设规模往往较大，建设周期长，投资大。

(2) 在数据仓库建好后，随着使用数据仓库的部门增多，对数据仓库资源的竞争将成为企业面临的一个难题。

(3) 各个部门都希望能定制数据仓库中的数据，但数据仓库是面向企业的。

为解决上述问题,人们提出了数据集市的概念,如图 3.4 所示。数据集市支持某一业务单元或部门特定的商业需求,其中的数据可以来自数据仓库。数据集市可以在数据仓库的基础上进行设计,如图 3.4(a)所示,也可类似数据仓库直接设计,如图 3.4(b)所示。数据集市中的数据具有数据仓库中数据的特点,只不过数据集市是专为某一部门或某个特定的商业需求定制的,而不是根据数据容量命名的。

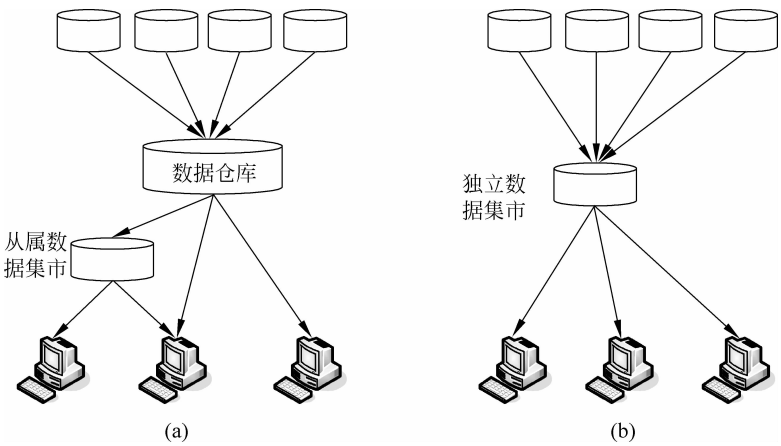

图 3.4 两种类型的数据集市

数据集市面向部门、业务单元或特定应用,因而规模较小,便于快速实现,且成本低廉,短期内即可获得明显效果。数据集市的应用不仅满足了部门的数据处理需求,而且作为数据仓库的子集有助于构建完整的企业级数据仓库。

3.4 元数据

当需要了解某地企业及其提供的服务时,电话黄页的重要性就体现出来了。元数据(metadata)就类似于这样的电话黄页。

1. 元数据的定义

数据仓库中的元数据是关于数据仓库中数据的数据。它的作用类似数据库管理系统的数据字典,用于保存逻辑数据结构、文件、地址和索引等信息。从广义上讲,在数据仓库中,元数据是描述数据仓库内数据的结构和建立方法的数据。

元数据是数据仓库管理系统的重要组成部分,元数据管理器是企业级数据仓库中的关键组件,贯穿于数据仓库构建的整个过程,直接影响着数据仓库的构建、使用和维护。

(1) 构建数据仓库的主要步骤之一是 ETL。这时元数据发挥了重要的作用,它定义了源数据到数据仓库的映射、数据转换的规则、数据仓库的逻辑结构、数据更新的规则、数据导入的历史记录以及装载周期等相关内容。数据抽取和转换的专家以及数据仓库管理员正是通过元数据高效地构建数据仓库的。

(2) 用户在使用数据仓库时,通过元数据访问数据,明确数据项的含义以及定制报表。

（3）数据仓库的规模及其复杂性离不开正确的元数据管理，包括增加或移除外部数据源，改变数据清洗方法，控制出错的查询以及安排备份等。

元数据可分为技术元数据和业务元数据。技术元数据为开发和管理数据仓库的IT人员使用，它描述了与数据仓库开发、管理和维护相关的数据，包括数据源信息、数据转换描述、数据仓库模型、数据清洗与更新规则、数据映射和访问权限等。而业务元数据为管理层和业务分析人员服务，从业务角度描述数据，包括商务术语、数据仓库中有什么数据、数据的位置和数据的可用性等，使业务人员更好地理解数据仓库中哪些数据是可用的以及如何使用它们。

上述表明，元数据不仅定义了数据仓库中数据的模式、来源、抽取和转换规则等，而且整个数据仓库系统的运行都是基于元数据的，元数据把数据仓库系统中各个松散的组件联系起来，组成了一个有机的整体，如图3.5所示[5]。

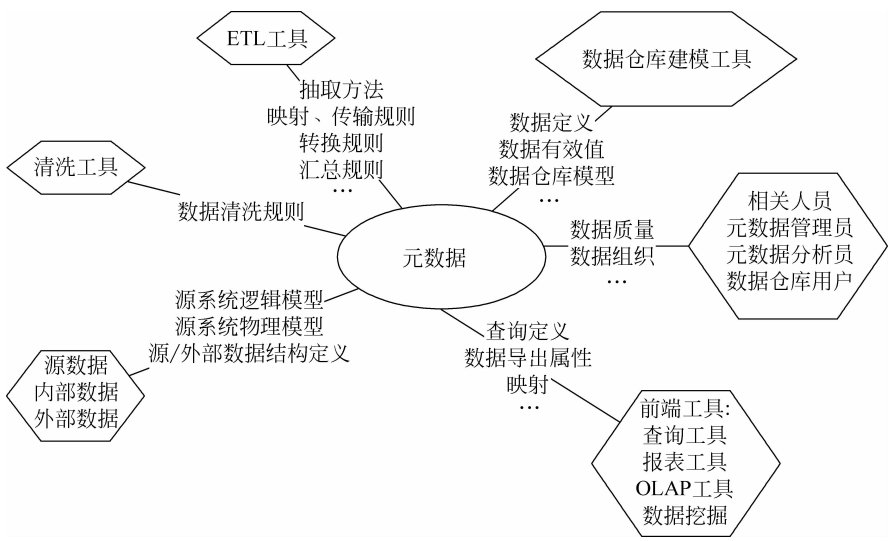

图3.5　元数据

2. 元数据的存储方式

元数据有两种常见的存储方式：一种是以数据集为基础，每一个数据集都有对应的元数据文件，每一个元数据文件包含对应数据集的元数据内容；另一种存储方式是以数据库为基础的，即元数据库。其中元数据文件由若干项组成，每一项表示元数据的一个要素，每条记录为数据集的元数据内容。上述存储方式各有优缺点，第一种存储方式的优点是调用数据时相应的元数据也作为一个独立的文件被传输，相对数据库有较强的独立性，在对元数据进行检索时可以利用数据库的功能来实现，也可以把元数据文件调到其他数据库系统中进行操作。不足是如果每一个数据集都对应一个元数据文档，在规模巨大的数据库中则会有大量的元数据文件，管理起来不方便。第二种存储方式元数据库中只有一个元数据文件，管理比较方便，添加或删除数据集时只要在该文件中添加或删除相应的记录项就可以了。在获取某数据集的元数据时，因为实际得到的只是关系表格数据的一条记录，所以要求用户系统可以接受这种特定形式的数据。因此，推荐使用元数据库的方式。

元数据库用于存储元数据,因此元数据库最好选用主流的关系数据库管理系统。元数据库还包含用于操作和查询元数据的机制。建立元数据库的主要好处是提供统一的数据结构和业务规则,易于把企业内部的多个数据集市有机地集成起来。目前,一些企业倾向于建立多个数据集市,而不是一个集中的数据仓库,这时可以考虑在建立数据仓库(或数据集市)之前,先建立一个用于描述数据、服务应用集成的元数据库,做好数据仓库实施的初期支持工作,对后续开发和维护有很大的帮助。元数据库保证了数据仓库数据的一致性和准确性,为企业进行数据质量管理奠定了基础。

3. 元数据的作用

在数据仓库中,元数据的主要作用如下。

(1) 描述哪些数据在数据仓库中,帮助决策分析者对数据仓库中的数据定位。

(2) 定义数据进入数据仓库的方式,作为数据汇总、映射和清洗等的指南。

(3) 记录业务事件的发生和随之进行的数据抽取工作的时间安排。

(4) 记录并检测系统数据一致性的要求和执行情况。

(5) 评估数据质量。

4. 粒度

粒度反映了数据仓库按照不同的层次组织数据,根据不同的查询需要存储不同细节的数据。在数据仓库中,粒度越小,数据越细,查询范围就越广泛。相反,粒度级别越高,表示细节程度越低,查询范围就越小。例如,当信用卡发行商查询数据仓库时,首先要了解某个地区信用卡的总体使用情况,然后检查不同类别用户的信用卡消费记录,这就涉及不同细节的数据。数据仓库中包含的数据冗余程度较高,批量载入和查询会影响数据管理和查询效率,因此数据仓库采用数据分区存储技术以改善数据仓库的可维护性,提升查询速度和加载性能,解决从数据仓库中删除旧数据时造成的数据修剪等问题,把数据划分成多个小的单元。

根据粒度的不同,可以把数据划分为早期细节级、当前细节级、轻度综合级和高度综合级等[6]。ETL 后的源数据首先进入当前细节级,并根据需要进一步进入轻度综合级乃至高度综合级。一旦数据过期,当前数据粒度的具体划分会直接影响到数据仓库中的数据量以及查询质量。图 3.6 显示了典型数据仓库的粒度结构。

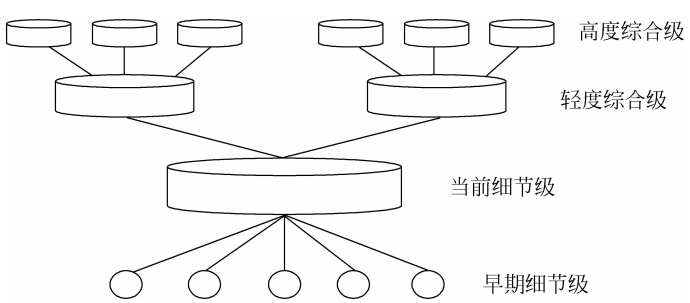

图 3.6 典型数据仓库的粒度

5. 通用数据仓库元模型

数据仓库中数据的多粒度化为用户使用数据提供了一定的灵活性，例如，家用电器销售数据可以同时满足市场、财务和销售等部门的需要，财务部门若要了解某地区的销售收入，则只需改变相关数据的粒度即可。

目前，大部分商务智能产品都有不同的元数据模型，这就可能给集成不同产品带来麻烦。针对这个问题，元数据模型的标准化管理是有必要的，通用数据仓库元模型（Common Warehouse Meta-model，CWM）就是不同元数据的存储和管理标准，这种标准是 OMG 为解决元数据交换而采用的一个标准，得到了 IBM、Oracle、NCR、Sun 和 HP 等公司的支持。这个标准提供了基于 XML 的元数据交换模型，可能大量应用于新一代的数据仓库系统。

通用数据仓库元模型通过不断完善的模型标准管理元数据，其中 Package 通过 OMG 提供的标准交互式数据语言（Interactive Data Language，IDL）转换为关系数据库管理系统中的 SQL 或存储过程实现。这里的 Package 利用统一的接口方便应用存储元数据。这样，元数据就可以在不同商务智能系统之间交换和共享，从而减少商务智能构建的费用。

3.5 ETL

数据仓库并非只是数据的简单累积，而是要经过一系列的抽取、转换和装载的过程，即 ETL。ETL 是构建数据仓库的重要环节，也是企业数据管理的核心，对数据仓库的后续环节影响比较大。目前市场上主流的 ETL 工具主要有 Informatica 的 PowerCenter、IBM 的 DataStage、Oracle 的 Warehouse Builder 以及 Microsoft 的 SQL Server IS 等。下面简要介绍 ETL 的主要功能。

1. 数据抽取

数据仓库是面向主题的，并非源数据库的所有数据都是有用的，所以在把源数据库中的相关数据导入到数据仓库之前，需要先确定该数据库中哪些数据是与决策相关的，数据抽取的过程大致如下。

（1）确认数据源的数据及其含义。

（2）抽取。确定访问源数据库中的哪些文件或表，需要提取其中哪些字段。

（3）抽取频率。需要定期更新数据仓库的数据，因此对于不同的数据源需要确定数据抽取的频率，如每天、每星期、每月或每季度等。

（4）输出。数据输出的目的地和输出的格式。

（5）异常处理。当需要的数据无法抽取时如何处理。

2. 数据转换

数据仓库的数据来自多种数据源。不同的数据源可能由不同的平台开发，使用不同的数据库管理系统，因此数据格式也可能不同。源数据在被装载到数据仓库之前，需要进行一定的数据转换。数据转换的主要任务是对数据粒度以及不一致的数据进行转换。

(1) 不一致数据的转换。数据不一致包括数据源内部的不一致和多个数据源之间的数据不一致等,例如,在一个应用系统中 BJ 表示北京、SH 表示上海、GZ 表示广州。而在另一个应用系统中,对应的代码分别为 1、2、3。此外,不同业务系统的数量单位、编码、值域或语义等都需要统一,例如,某供应商在结算系统的编码是 990001,而在 CRM 中编码是 YY0001,这时就需要抽取后统一转换编码。

(2) 数据粒度的转换。业务系统一般存储细粒度的事务型数据,而数据仓库中的数据是用于查询、分析,因此需要多种不同粒度的数据。这些不同粒度的数据可以通过对细粒度的事务型数据进行聚合(aggregation)而产生。

3. 数据清洗

数据源中数据的质量是非常重要的,低劣的"脏"数据容易导致低质量的决策甚至是错误的决策。此外,这些"脏"数据或不可用数据也可能造成报表的不一致等问题。因此有必要全面校验数据源的数据质量,尽量减少差错,此过程就是数据清洗(data cleaning)。目前一些商务智能企业提供数据质量防火墙,如 Business Objects(SAP)的 Firstlogic,它能够自动减少数据的噪声。清洗后的数据经过业务主管确认并修正后再进行抽取。数据清洗能处理数据源的各种噪音数据,主要的数据质量问题有以下几种。

(1) 缺失(missing)数据,即数据值的缺失,这在顾客相关的数据中经常出现,例如,顾客输入个人信息时遗漏了所在区域。

(2) 错误数据。常见的错误数据包括字段的虚假值、异常取值等。例如,在教学选课系统中,选修某门课程的人数不能超过该课程所在教室的座位数。这些错误数据产生的主要原因是由于业务系统在数据输入后不能进行正确性判断而被录入数据库。错误数据需要被及时找出并限期修正。

(3) 数据重复。数据重复是反复录入同样的数据记录,这类数据会增加数据分析的开销。

(4) 数据冲突。源数据中一些相关字段的值必须是兼容的。例如,一个顾客记录中省份字段使用 SH(上海),而此顾客的邮政编码字段使用 100000(北京地区的邮政编码)。数据冲突包括同一数据源内部的数据冲突和多个数据源之间的数据冲突。冲突的数据也需要及时地修正。

4. 数据装载

数据转换、清洗结束后需要把数据装载到数据仓库中,通常分为以下几种方式。

(1) 初始装载。一次对整个数据仓库进行装载。

(2) 增量装载。在数据仓库中,增量装载可以保证数据仓库与源数据变化的同期性。

(3) 完全刷新。周期性地重写整个数据仓库,有时也可能只对一些特定的数据进行刷新。

在初始装载完成后,为维护和保持数据的有效性,可以采用更新和刷新的方式:更新是对数据源的变化进行记录,而刷新则是指对特定周期数据进行重新装载。

3.6 操作数据存储

数据仓库实现了操作型数据与分析型数据的分离，从而为企业建立了数据库-数据仓库(DB-DW)两层体系结构。然而 DB-DW 并不能完全满足企业所有的数据处理需求。在实际应用中，有时会遇到企业日常管理和战术决策的问题，尤其是最近发展起来的操作型商务智能，需要对面临的问题和机遇做出快速反应。这类问题不是简单的日常事务处理，对此类问题的解决需要企业全局一致的、细粒度的、当前或接近当前(current)的数据，这些数据又是面向主题的、集成的和时变的，具有数据仓库的特征。这里的当前数据可根据业务分析确定。为处理上述问题，需要在 DB-DW 之间增加一个新的层次——操作数据存储*，它属于操作型处理和分析型处理之间的一个中间层次，这样就形成了数据仓库的 DB-ODS-DW 结构，满足实时或近实时的查询要求和报表需求。例如，电信公司渠道支撑分析人员需要及时地了解顾客消费行为的变化，尽快为顾客提供主动的服务，从而挽留高价值顾客，提高收入，这就需要对近实时的数据进行分析。

与数据仓库相似，操作数据存储中的数据组织方式也是面向主题的、集成的，因此数据在进入操作数据存储前也需要进行集成处理。然而操作数据存储又有类似操作型数据库的特点：操作型数据库是联机可变的，它会根据需要对这些数据执行增删和更新等操作。操作数据存储只存放当前或接近当前的数据，存取的数据是最近一段时间产生的，而不是历史数据。操作数据存储的数据可作为数据仓库的数据源成批导入。表 3.2 对操作数据存储、数据仓库和操作型数据库进行了比较。

表 3.2 操作型数据库、操作数据存储和数据仓库之间的比较

比较项目	操作型数据库	操作数据存储	数据仓库
数据内容	当前值	当前和最近的值	存档、归纳数据和经计算得出的数据
数据组织	面向应用程序	有主题域，数据集成	面向主题
数据性质	动态(经常变化)	动态(经常变化)	静态(刷新除外)
数据结构	适于操作型计算	较复杂	适于决策分析
访问频率	高	高到中	中到低
数据更新	按字段更新	联机可变，数据更新频率比较高	一般不修改，更新时间长
数据访问	少量记录/事务	少量记录/事务	大量记录/事务
用途	高度结构化、重复处理和事务处理	高度结构化、重复处理和事务处理	分析处理
对响应时间的要求	通常少于 5s	通常少于 5s	几秒钟到几分钟，有时几小时
对性能的要求	高	高到中	中

* 也有人把操作数据存储称为实时数据仓库。

3.7 数据仓库模型

模型是对现实世界进行抽象的工具。在信息管理的过程中需要把现实世界的事物及其相关特征转换为信息世界的数据后才能进行处理,这就需要数据模型作为转换的桥梁。

数据仓库的模型类似于仓储中的货架,模型设计也是数据仓库设计的主要内容。这些模型包括概念模型、逻辑模型和物理模型等。元数据模型自始至终伴随着数据仓库的设计、实施和使用。数据粒度和聚合模型也在数据仓库的设计中发挥着重要的作用,影响数据仓库的具体实现[7]。

1. 概念模型

信息世界是现实世界在人们头脑中的反映,概念模型用来表达信息世界中的信息结构,通常人们利用概念模型定义实际的数据需求。确定概念模型需要和用户一起完成。关系数据库一般采用实体-关系(E-R)图作为概念模型的表示方法。这是由于 E-R 图简单、易于理解,有良好的可操作性,对现实世界的描述能力也较强。目前的数据仓库实际上是通过主题分析表示概念模型的,每个主题用若干维(dimension)和度量(measure)表示。维度是人们观察世界的特定角度。例如,销售经理需要了解每个月、某个特定地区、不同销售部门销售新产品创造的利润,可以按照时间、地区、分销机构和产品型号等维度进行评价,这就是业务问题的分析维度。度量是确定与维度分析有关的数值信息,如销售量等。在银行卡业务分析中,银行卡业务的收入主要包括用户年费、透支的利息、向特约商户收取的刷卡消费手续费和其他收入。以顾客为中心,获取有价值的顾客是银行卡业务关注的重点,因此顾客、商户和卡业务是银行卡业务分析的主题。概念模型通过用信息包图表示,包括主题的多维特性、每个维度的层次以及分析指标(度量)。图 3.7 是销售分析的信息包图。

信息包:销售分析

维度 →

类别 ↓

日期	销售地点	销售产品	年龄组别	性别	
年	国家	产品类	年龄组(8)	性别组(2)	
季度	区域	产品组			
月	城市	产品			
	区				
	商店				
度量和事实:预测销售量、实际销售量、预测偏差					

图 3.7 销售分析的信息包图

2. 逻辑模型

在概念模型中确定了主题,定义了分析维度和度量后,就可以设计数据仓库的逻辑模型了。

数据仓库可采用多维数组实现，具有查询方便、快速的优点。但随着关系数据库的成熟，数据仓库也可以建立在关系数据库的基础上，在进行维度分析时利用原有的关系模型构建事实表和维表，事实表包括分析的主题相关维度 ID 和度量，维表包括维度的具体内容。例如，上面提到的银行卡业务分析中用户主题可以用用户基本信息、时间、渠道、产品、人口统计和用户细分信息等对银行卡用户进行多角度透视，全方位把握用户信息。目前主要的数据库厂商都扩充了数据仓库管理功能。数据仓库通常有以下两种基本的逻辑模型：星型模型和雪花模型。

1）星型模型

星型模型的核心是事实表，事实表把各种不同的维表连接起来[8]。图 3.8 所示为某公司销售数据仓库的星型模型，其中销售记录表是事实表，而日期表、商店表、产品表和顾客表都是维表。

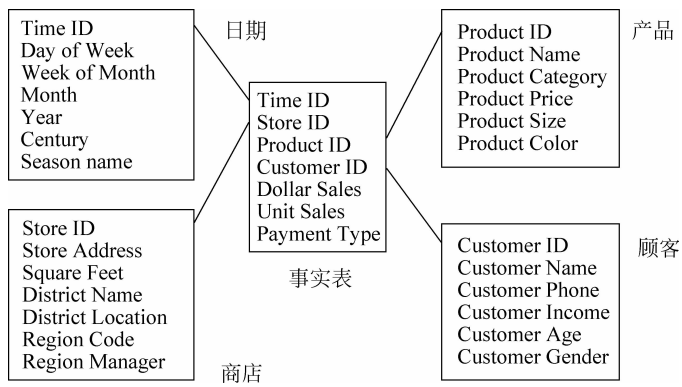

图 3.8　某公司销售数据仓库的星型模型

2）雪花模型

雪花模型是星型模型的扩展，某些维表中的数据可以进一步分解到附加的表中，以便减少冗余，节省存储空间。雪花模型对星型模型中的维表进行进一步标准化、规范化处理。图 3.9 所示为一个雪花模型，在星型模型的基础上做了进一步的扩展，在商店维表的基础上增加了店铺所在地区的细节维表。

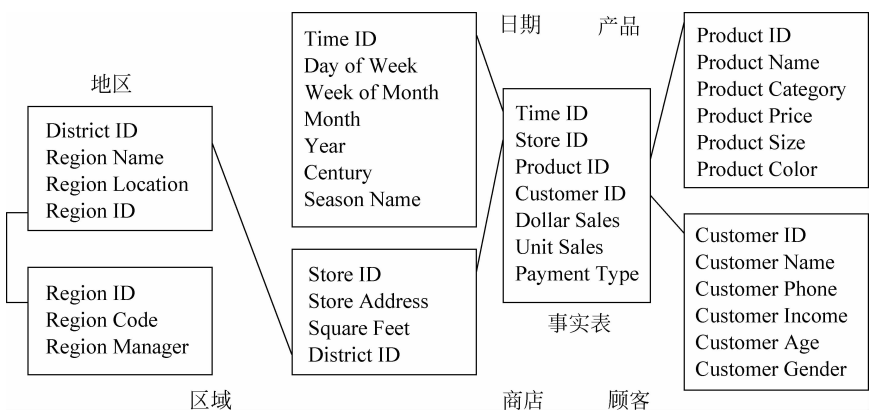

图 3.9　某公司销售数据仓库的雪花模型

除星型模型和雪花模型外,还有衍生模型,例如,星系模型描述了数据仓库中多个事实表共享一个或多个维表的情况。

总而言之,逻辑模型设计包括确定数据仓库中数据粒度、数据分割策略、关系模式以及记录系统定义等工作。逻辑模型对每个当前要装载主题的逻辑实现进行定义,并把相关内容记录在数据仓库的元数据中,包括适当的粒度划分、合理的数据分割策略、适当的表结构划分以及数据源等。

数据仓库逻辑模型设计中要解决的一个重要问题是决定数据仓库的粒度划分,粒度划分适当与否直接影响数据仓库中的数据量和所适合的查询类型。在划分数据仓库的粒度时,可以通过估算数据行数和所需的直接存储设备(Direct Access Storage Device,DASD)数,确定采用单一粒度还是多重粒度[9]。

在确定数据分割策略,选择适当的数据分割标准时,一般要考虑以下因素:数据量、数据分析处理的实际情况、简单易行以及粒度划分等。数据量的大小是影响是否进行数据分割以及如何进行分割的主要因素,而如何进行数据分析处理是选择数据分割标准的一个主要依据,因为数据分割与数据分析处理的对象是紧密联系的。此外,还要考虑选择的数据分割标准是自然的、易于实施的,确保数据分割的标准与粒度划分是互相适应的。

3. 物理模型

数据仓库的物理模型是逻辑模型在数据仓库中的实现,即数据仓库的物理分布模型,主要包含数据仓库的软硬件配置、数据的存储结构与索引、数据存放位置和存储分配等。

设计数据仓库的物理模型时,需要全面了解所选用的数据库管理系统,特别是存储结构和存取方法,并了解数据环境、数据的使用频度、使用方式、数据规模以及响应速度等,这些因素是平衡时间和空间效率的重要依据。此外,还应了解外部存储设备的特性,如分块原则、块大小和设备的I/O特性等。物理模型是在逻辑模型的基础上实现的,设计时应该权衡以下因素:存取时间、存储空间利用率和维护代价。当数据仓库的数据量很大时,需要对数据的存取路径进行仔细的设计,也可以建立索引以提高数据存取的效率。

同一个主题的数据并不一定要存储在相同的介质上。在物理设计时,常常按数据的重要程度、使用频率以及对响应速度的要求进行分类,并把不同类的数据分别存储在不同的存储设备上。重要程度高、经常存取并对响应时间要求高的数据存放在高速存储设备上,如硬盘或内存。而存取频率低或对存取响应时间要求低的数据则放在低速存储设备上,如磁盘或磁带。

确定数据存储位置时还要考虑一些其他方法,以设计相应的元数据,例如,是否对一些常见应用建立数据序列,是否对常用的、不常修改的表或属性冗余存储。有些数据库管理系统提供了存储分配的参数供设计者做物理优化处理,例如,块的大小、缓冲区的大小和个数等,这些参数都要在进行物理设计时确定。

设计数据仓库的物理模型需要考虑以下几个问题。

1) 确定项目资源

需要IT人员深入了解底层数据和业务需求。根据预算和业务需求,并参考以往的数据仓库项目经验,对该项目的成本、周期和资源进行估算。关于项目周期的估算,主要基于ETL功能点以及加权后的复杂度进行估算,因为ETL占据了整个数据仓库项目周期的70%。ETL不同的功能点具有不同的复杂度,通过以往的项目经验和专家评估,然后再根

据软件生命周期的估计，可以有效地获知项目的周期。关于人员费用的估算，不仅要考虑人员的工作经验、素养以及对新技术的掌握能力，而且还要兼顾人员流动等因素。当项目资源遇到瓶颈的时候，就可以考虑协作。

2）确定软硬件配置

数据仓库项目与其他应用系统的开发不同，尤其需要对数据容量进行估算，这是由数据仓库的特性所决定的。如果项目初期不加以考虑，就会造成灾难性的后果。

数据仓库的容量估算是可预见的，首先确定核心明细数据的存储年限、相关表的平均字段长度值、每年的记录数和每年预计的增长，然后再加上20%的冗余，就可以得到数据仓库的预计容量。

数据仓库的处理能力与容量密切相关，也与具体的关系数据库性能相关。因此，如何在Oracle、Microsoft SQL Server、DB2、Sybase以及MySQL之间寻找平衡，就需要考虑实际的预算和业务需求。

关于硬件的配置，既需要发挥软件的功能，满足实际的处理要求，又要为将来的系统扩展预留出一定的空间。

3）数据仓库存储设计

数据仓库一般采用分层设计，即ODS层、数据仓库层、数据仓库聚合层和数据集市等。数据仓库的分层是灵活的，没有固定的模式，一般视实际情况而定。

4）数据仓库ETL策略

（1）数据抽取策略

数据抽取不仅需要满足业务处理和决策分析的要求，而且不能影响业务系统的性能。

（2）数据转换策略

数据转换是根据决策分析主题的要求，对业务系统中抽取的源数据进行转换、清洗等处理，保证来自不同数据源的数据的一致性和完整性。

（3）数据加载策略

数据加载是指从业务系统中抽取转换后的数据装载到数据仓库中。

3.8 数据挖掘查询语言

下面介绍如何用数据挖掘语言描述数据仓库。数据挖掘原语可以用来定义数据挖掘任务。数据挖掘语言是由数据挖掘原语定义的，如数据挖掘查询语言（Data Mining Query Language，DMQL）。DMQL是一种基于SQL的数据挖掘查询语言，包括定义数据仓库与数据集市、挖掘概念/类描述、关联和分类等数据挖掘原语。数据挖掘语言提供了交互式数据挖掘工具，具有类似SQL的语法，易于与SQL集成。

数据仓库和数据集市可以使用两种数据挖掘原语定义：一种是立方体定义；另一种是维定义。立方体定义和维定义的语法如下。

```
define cube <cube_name>[<dimension_list>]: <measure_list>
define dimension <dimension_name> as (<attribute_or_subdimesion_list>)
```

【例 3.1】 定义雪花模型

图 3.10 所示为销售主题的雪花模型[10]。

```
define cube sales [product, time, branch, location]:
    dollars_sold = sum (sales_in_dollars), units_sold = count( * )
define dimension product as (product_key, product_name, brand, type, supplier(supplier_key,
supplier_name))
define dimension time as (time_key, day, day_of_week, month, quarter, year)
define dimension branch as (branch_key, branch_name, branch_type)
define dimension location as (location_key, street, city(city_key, city_name, province_or_
state, country))
```

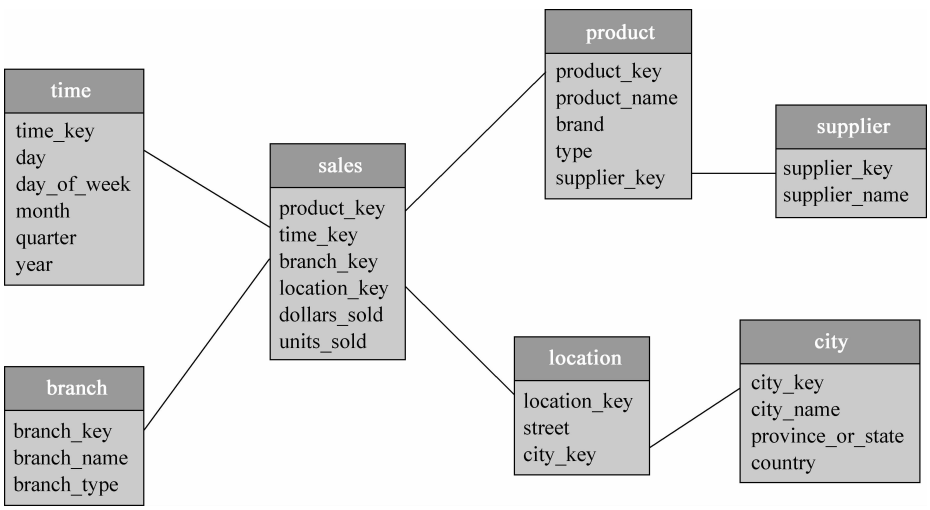

图 3.10 销售主题的雪花模型

【例 3.2】 定义星系模型

事实表共享维表的模式可以看作星型模型集,也称为星系模型或事实星座。数据仓库通常要对多个相关的主题建模,因此要经常使用事实星座。如图 3.11 所示,销售主题数据立方体与货运主题数据立方体共享时间维、产品维和区域维[10]。

```
define cube sales [product,time,branch,location]:
    dollars_sold = sum(sales_in_dollars),units_sold = count( * )
define dimension product as
    (product_key,product_name,brand,type,supplier_type)
define dimension time as (time_key,day,day_of_week,month,quarter,year)
define dimension branch as (branch_key,branch_name,branch_type)
define dimension location as
    (location_key,street,city,province_or_state,country)
define cube shipping [time,product,shipper,from_location,to_location]:
    dollars_cost = sum(cost_in_dollars),units_shipped = count( * )
define dimension time as time in cube sales
define dimension product as product in cube sales
define dimension shipper as (shipper_key,shipper_name,location as location in cube sales,
shipper_type)
```

```
define dimension from_location as location in cube sales
define dimension to_location as location in cube sales
```

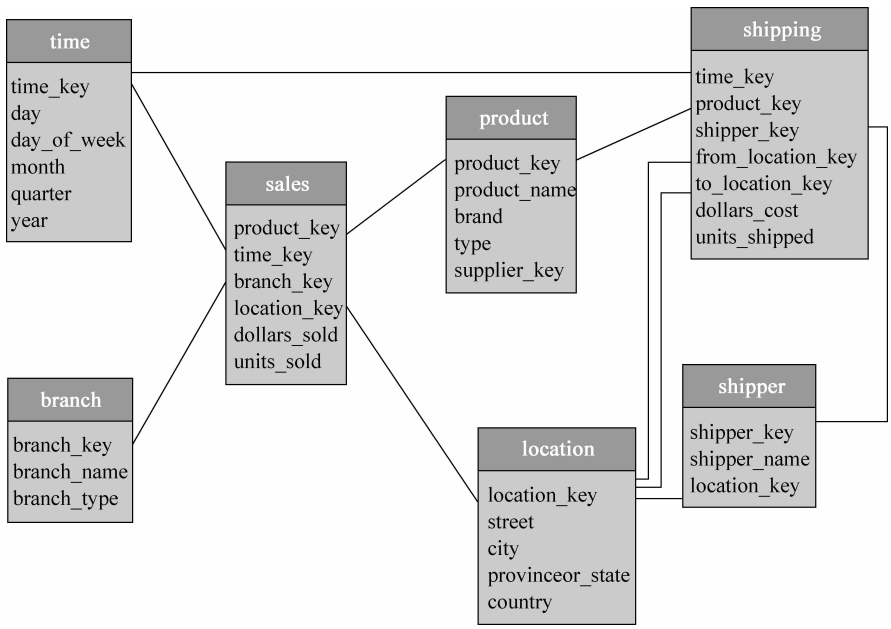

图 3.11 包含公共维的星系模型

数据立方体度量是一个数值函数，该函数可以对数据立方体求值。度量包括分布的和代数的：分布度量可以用分布聚集函数 count()、sum()、min()和 max()等计算，代数度量可以用代数聚集函数 average()和 standard_deviation()来进行计算。

3.9 医保数据仓库设计

下面以某市医疗保险（简称医保）数据仓库的设计为例，简要说明数据仓库的设计过程[11]。

1. 医保的业务分析

由于某市比较重视医保的发展，某市医保体系的信息化建设已涵盖了实时交易、个人账户管理、医保事务管理、医保服务点管理、统计查询、监督管理、审核结算管理、保障卡交换和顾客服务系统等多项业务。但建设的初期缺乏严格的整体规划，而且每个子系统都有局部的业务数据，形成数据孤岛，数据缺乏标准化和完整性。导致数据杂乱、冗余和数据交换繁杂，造成了数据的综合分析、利用能力不足，影响了医保管理的时效性，使决策者难以及时获得医疗状况的宏观情况。此外，从业务子系统中采集的各种基础数据量约有 500GB，并且随着医保数据采集力度加大，数据量按每月 50GB 左右的速度急速增长。按此预测，未来数据中心存储的数据量将达到 TB 级或 PB 级，医保信息系统将负载庞大的数据。

某市医保管理部门决定采用数据仓库建立集成的数据平台，以整合各分布系统源的数

据,提高医保管理的综合分析能力,及时、有效地反映医保基金的运作情况。

2. 医保数据仓库系统架构

医保数据仓库的设计采用了典型的数据仓库三层架构。从数据源到最终呈现给用户,中间经过数据获取、数据管理和信息传递等过程。

(1) 数据获取。采用 ETL 工具从分布异构的数据源抽取相关的源数据,经过转化、清洗,装载到操作型数据存储(ODS)中。操作型数据存储提供当前操作型数据的统一视图,其中的数据以日为单位或以自定义的时间间隔进行定时更新,提供短期业务分析,例如日常报表等。此外,还可以使用 ETL 工具,把存储在外部数据源的数据或操作型数据批量加载到数据仓库。

(2) 数据管理。根据目前 500GB 的基础数据量和每月大约 50GB 的增长速度,可以预测未来某市医保数据仓库存储的数据量将达到 TB 或 PB 级。在数据仓库的设计中,考虑现有业务系统核心数据库的加载速度和转换复杂度,医保系统选用了 Oracle 数据仓库管理系统,利用其分区技术,把数据仓库划分为主仓库、元数据库和数据集市。Oracle 数据仓库管理系统提供了一个多维 OLAP (MOLAP)服务器 Oracle Express Server,使用多维模型存储和管理多维数据库或多维高速缓存,同时也能够访问多种关系数据库。在整个医保数据仓库解决方案实施过程中,通常把汇总数据存储在 Express 多维数据库中,而把详细数据存储在 Oracle 关系数据库中。

(3) 信息传递。某市医保数据仓库系统在信息传递设计中利用 Hyperion Essbase 套件设计了 OLAP 模型,先由业务人员制定需求,然后根据业务需求明确 OLAP 主题,并建立相应的映射数据。此外,根据业务人员习惯用 Microsoft Office 查看报表的特点,在展现层保留了原有 Word、Excel 等报表模式,利用 Hyperion Essbase 的 Office Addin 模块进行基于 Microsoft Office 的报表分析。

3. 医保数据 ETL

在医保数据仓库设计时 ETL 采用中间文件的间接加载和不含中间文件的直接加载等模式实现。

根据业务特征,医保数据仓库 ETL 过程大致如下。

(1) 预处理。预处理主要是为 ETL 做准备,包括清空工作区,定义数据源,检查目标数据仓库,定义 ETL 规则和数据映像等。

(2) 数据的批处理。数据的批处理主要包含数据的批处理加载。

(3) 维表的加载。维表的加载特别要注意维的对应和比较。

(4) 事实表加载。参照维表,查找相关主键进行事实表加载。

(5) 日志检测。日志检测可以监控数据加载的过程。

4. 建立数据集市

为了满足某些特定业务需求,提高分析的性能,在某市医保数据仓库系统中还设计了日常分析、监督审核和顾客关系等数据集市。这些数据集市在设计时采用了“自顶向下”的模式。此外,数据仓库的设计还采用了反馈模式:提出新需求后移交数据集市,数据集市的需

求也不断汇总到数据仓库。

5. 元数据的管理

为了避免元数据的分散管理，在医保数据仓库设计时，采用了 Sybase 数据仓库构建工具，从而建立统一的元数据库来实现元数据的集中管理。在医保数据仓库系统中，元数据有以下功能。

(1) 描述哪些医保数据在数据仓库中。

(2) 定义进入数据仓库的数据以及数据仓库中产生的数据。

(3) 业务发生而随之进行的数据抽取时间安排。

(4) 记录、检测系统数据一致性和执行情况。

(5) 评价数据质量。

医保数据仓库系统的建立，基本解决了医保信息化过程中的“信息孤岛”问题，为医保统计分析、查询、诊疗、检查等项目的费用预警和审核筛选提供技术支撑平台，强化了医保系统中的数据分析和数据挖掘功能，为业务部门的政策制定提供相应的数据信息和测算结果。此外，该系统对医保基金的支出、供需双方医疗行为的变化进行预测，并对参保职工个人和医药机构的违规行为进行分析，建立模型，探索违规和欺诈行为的规律，重点监控有此类倾向的人员和医药机构。

【例 3.3】 使用 SQL Server 2000 Analysis Services 创建数据仓库模型

Microsoft SQL Server 2000 Analysis Services 简称 Analysis Services，为商务智能应用提供了完整的分析平台，该平台具有关系存储、数据抽取、OLAP 分析和数据挖掘等功能。Analysis Services 允许开发人员设计、创建和管理包含从其他数据源聚合的多维数据结构，但仍然与 Microsoft SQL Server 绑定紧密。这里介绍使用 Analysis Services 创建数据仓库模型的方法。以 Analysis Services 附带的数据 FoodMart 为例，创建一个雪花形的数据仓库模型，具体的步骤如下。

图 3.12 连接数据源

(1) 首先在 ODBC 数据源管理器中添加用户数据源，并命名为 FoodMart 2000，数据库选用 Analysis Services 附带的 foodmart 2000.mdb。

(2) 打开 Analysis Services，连接到分析服务器后，新建 FoodMart 2000 数据库，右击“数据源”，选择“新数据源”，使用 Microsoft OLE DB Provider for ODBC Drivers 连接已建的 FoodMart 2000 数据源，如图 3.12 所示。

(3) 使用向导创建多维数据集。

① 从数据源 FoodMart 2000 中选择 sales_fact_1997 作为事实表，该表的详细情况可以通过单击“浏览数据”按钮进行查看，如图 3.13 所示。

单击“下一步”按钮，选择事实表中的 store_sales、store_cost 和 unit_sales 作为多维数据集的度量值，如图 3.14 所示。

② 继续单击“下一步”按钮，使用维度向导创建维表。为了构造雪花模型，这里创建的

图 3.13 sales_fact_1997 作为事实表

图 3.14 多维数据集的度量值

维表依次包括时间维表(time_by_day)、产品维表(product)、产品类别维表(product_class)、顾客维表(customer)、促销维表(promotion)和商店维表(store)。其中产品类别维表是在产品维表的基础上新增的,这样就构成了雪花模型的数据仓库模型。

- 首先创建时间维,选择星型架构,从“可用的表”中选取表 time_by_day,维度类型选择时间维度(其他维度均选择标准维度),日期列被自动指定为 the_date,然后定义维度层次为年、季度、月。最后用户输入维度名称 NewCubeTime 完成时间维度的创建,用户可以通过单击“预览”按钮查看,如图 3.15 所示。
- 继续创建产品维,把 product 和 product_class 连接起来:新建维度,选择“雪花架

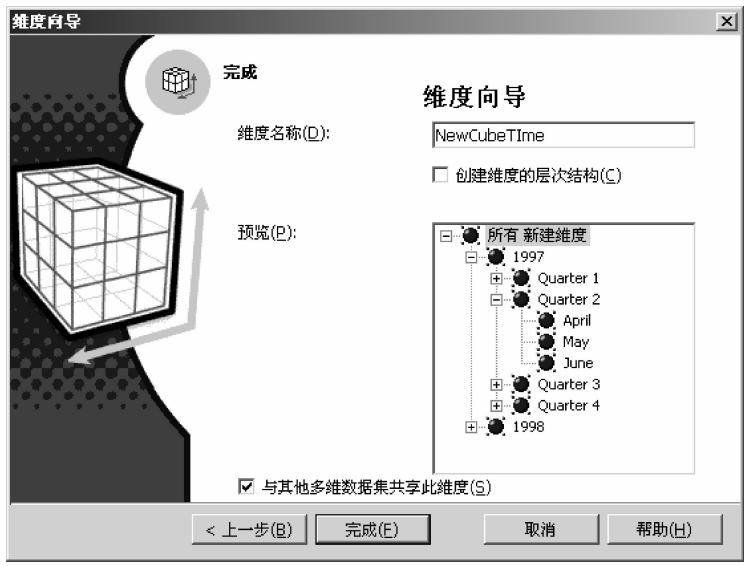

图 3.15 创建时间维

构”,从“可用的表”中选取表 product 和表 product_class。Analysis Services 允许用户拖放列指定连接,如图 3.16 所示。

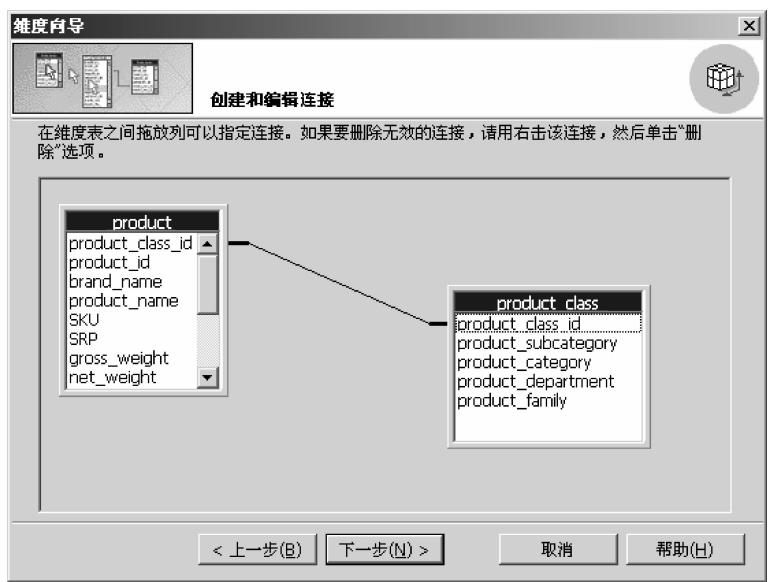

图 3.16 创建产品维

选择产品维的维度级别。默认情况下,Analysis Services 通过各个列中不同条目的计数自动分配级别,这里选择 product_category、product_subcategory 和 brand_name 定义。从图 3.17 中可以看到,“级别名称”前面的点数表明了不同列的级别。

输入维度名称 NewCubeProduct,单击“完成”按钮结束对产品维的创建。

• 类似上述步骤,创建商店维 NewCubeStore、顾客维 NewCubeCustomer 和促销维

NewCubePromotion。

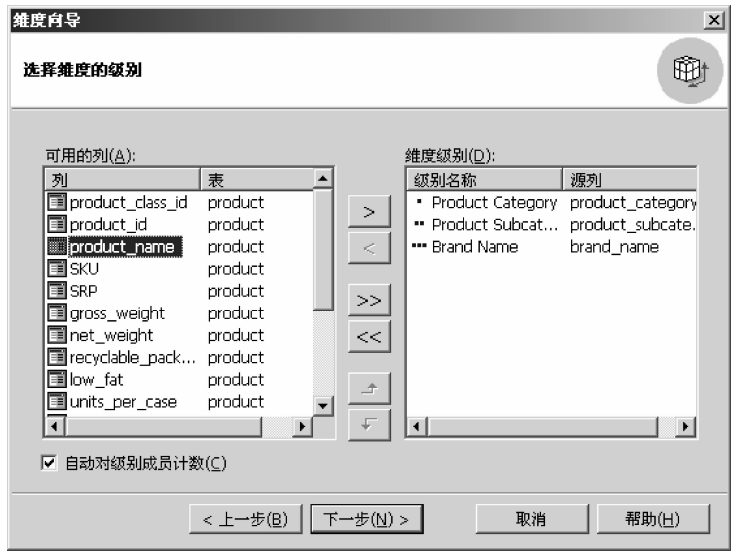

图 3.17 产品维的维度级别

③ 事实表和维表创建完成后,开始创建数据立方体。单击“下一步”按钮,输入生成的多维数据集的名称 NewCube,并单击“完成”按钮。最后得到的雪花形数据模型如图 3.18 所示,注意图 3.18 的左侧列出了多维数据集的维名称和度量,右侧为对应的雪花形数据集 NewCube。

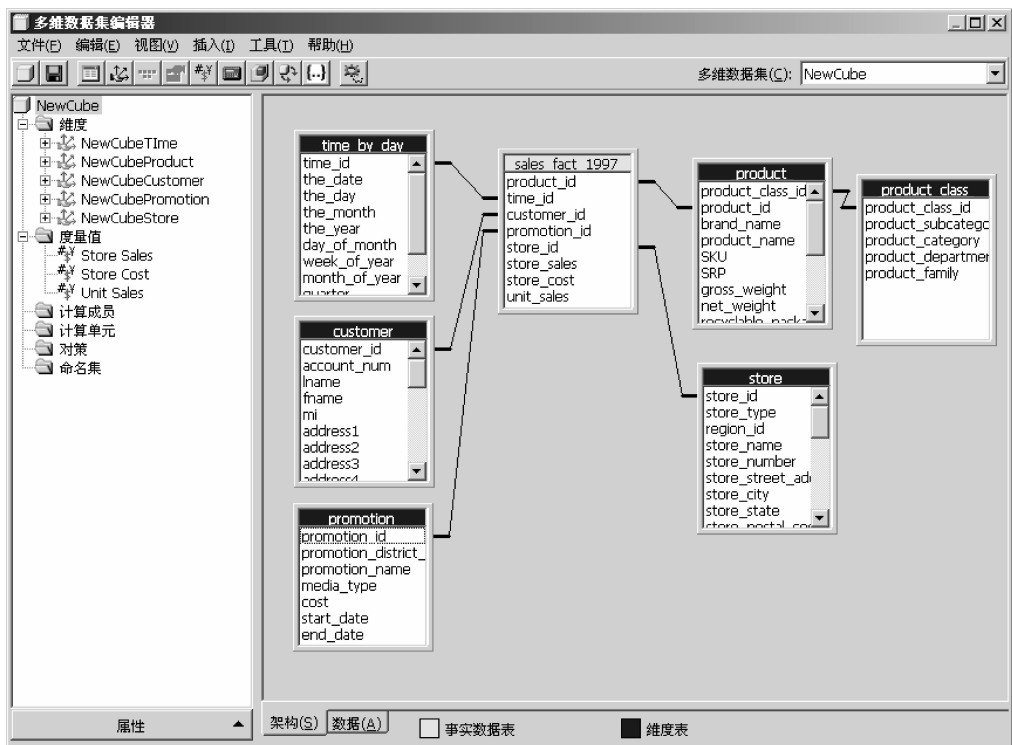

图 3.18 雪花形数据模型

【例 3.4】 使用 IBM DB2 Data Warehousing Design Studio 创建数据仓库模型

IBM DB2 Data Warehouse Edition 简称 DB2 DWE，它结合了 DB2 数据库的优点和 IBM 公司强大的商务智能基础设施。DB2 DWE 集成了数据转换、OLAP 分析以及数据挖掘的核心组件，使用 DB2 DWE 的 Cubing Services 可以方便地创建数据仓库模型。这里以 DB2 DWE 附带的数据库 GSDB 为例，创建一个雪花形的数据仓库模型，具体的步骤如下。

1）准备数据

首先解压附带的样本数据库文件 GSDB。在安装路径中打开文件夹...\IBM\ISWarehouse\samples\GSDB，解压文件 GSDB.zip 到当前目录。然后创建 GSDB 数据库："开始"→"所有程序"→IBM DB2→DB2COPY1（Default）→Command Line Tools→Command Window，进入路径...\IBM\ISWarehouse\samples\GSDB\win，运行脚本程序 setupGSDB-createDB-noprompt。脚本程序执行以后，用户可以通过 Control Center 检查 GSDB 是否正确地创建了，如图 3.19 所示。

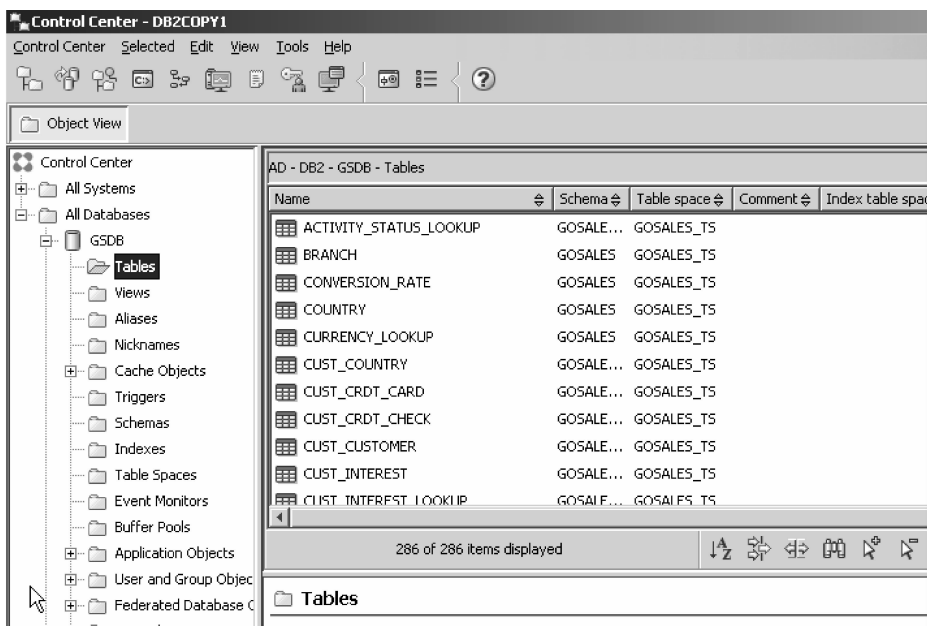

图 3.19 通过 Control Center 查看数据库 GSDB

2）使用 Design Studio 创建数据仓库模型

打开 Design Studio 客户端程序："开始"→"所有程序"→IBM InfoSphere Warehouse→ISWCOPY01→Design Studio。在 Data Source Explorer 窗口连接数据库 GSDB。若没有显示 Data Source Explorer，则可以通过 Window→Show View→Data Source Explorer 打开。在 Database Connection 找到数据库 GSDB 并右击选择"连接"，输入正确的用户名和密码即可创建连接，如图 3.20 所示。

然后创建 OLAP 数据设计工程。在 Design Studio 中，通过 File→New→Data Design Project（OLAP）创建一个新的 OLAP 数据设计工程，这里取名为 Cubing_Tutorial。在此基础上添加一个物理数据模型。

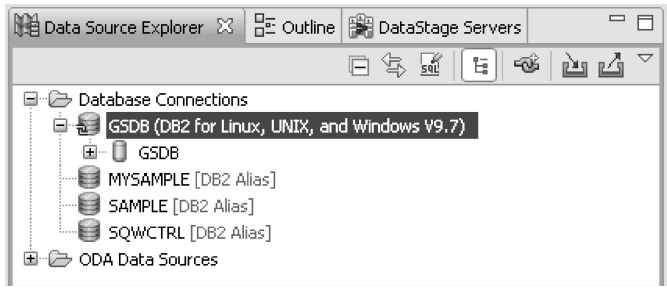

图 3.20 在 Design Studio 中连接数据源

(1) 在工程 Cubing_Tutorial 下,选中 Data Model,右击选择新建一个物理模型,并取名为 Database Model Tutorial,版本号则选择 V9.7,创建方式选择“反向工程”。单击“下一步”按钮。

(2) 选择反向工程的数据源类型为“数据库”,单击“下一步”按钮。

(3) 选择数据连接 GSDB,连接数据库 GSDB,单击“下一步”按钮。

(4) 选择 Schema 为 GOSALESDW,单击“下一步”按钮。

(5) 数据元素选择表和视图。

然后直接单击“完成”按钮,名为 Database Model Tutorial 的数据模型就创建完毕了,如图 3.21 所示。

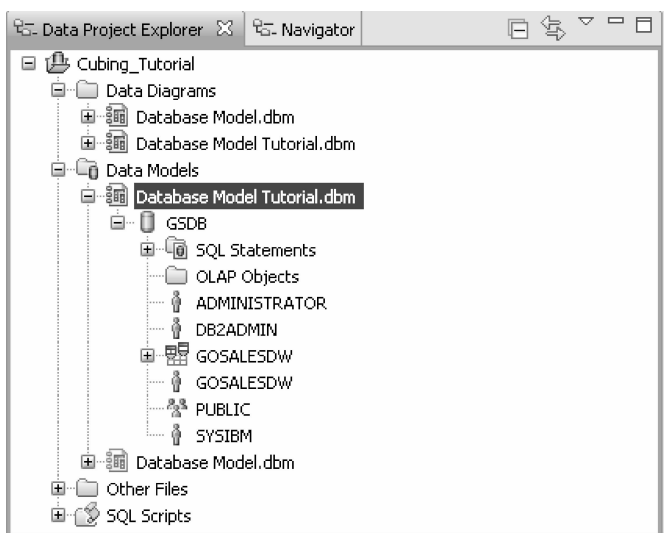

图 3.21 创建数据模型

接下来添加数据立方体模型。在上面创建的数据模型下选择 OLAP 对象并右击,选择添加数据立方体模型。选择 Schema GOSALESDW 下的数据库表 SLS_SALES_FACT 作为事实表。这张表里保存了大量的销售记录,它与多个数据库表有外键关联,如产品表 SLS_PRODUCT_DIM、时间表 GO_TIME_DIM 和雇员表 EMP_EMPLOYEE_DIM 等。这些关联表会被 Cubing Services 识别为数据立方体模型中的维表。而对于事实表 SLS_SALES_FACT 中的这些外键,Cubing Services 会自动把它们创建成维,除了主键和外键之外的其

他字段则会被创建为度量，如 UNIT_COST、UNIT_PRICE 和 QUANTITY 等字段。然后单击“下一步”按钮，在这里用户可以修改度量和维的名字，最后单击“完成”按钮，数据立方体模型就创建完成了，立方体模型包括度量、维、维级别和维层次等要素，如图 3.16 所示。同时，Cubing Services 还创建了一个缺省的数据立方体，这个缺省的数据立方体包括数据立方体模型中的所有度量和默认的维层次，如图 3.22 所示。

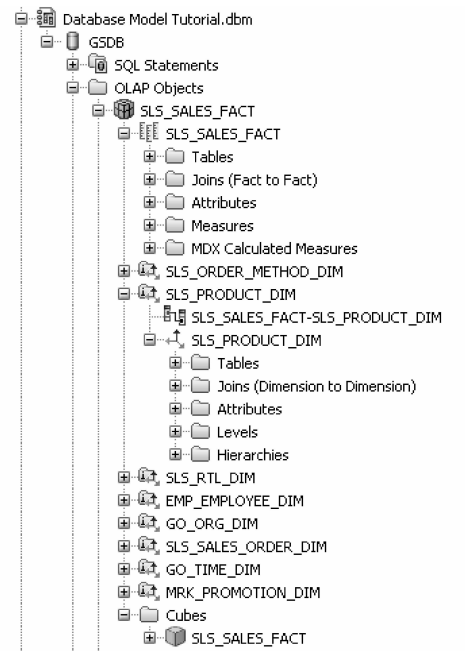

图 3.22　创建数据立方体

可以看出，Cubing Services 根据事实表的外键关联自动创建了维和度量。如果这些维表中还存在外键与其他数据库表关联，Cubing Services 会把这些表连接起来并创建雪花模型。这里产品维度表还与商标表 SLS_PRODUCT_BRAND_LOOKUP、颜色表 SLS_PRODUCT_COLOR_LOOKUP 和尺寸表 SLS_PRODUCT_SIZE_LOOKUP 等有外键关联关系，Cubing Services 则会把它们与产品维表相连接，创建雪花形数据模型，如图 3.23 所示。

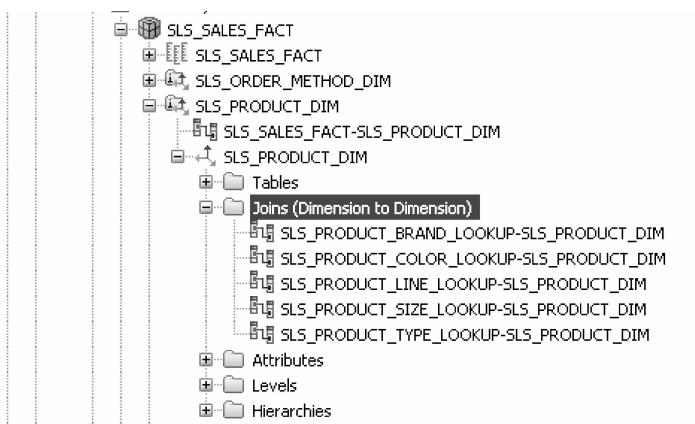

图 3.23　雪花形数据模型(一)

若选择产品维表与商标表连接，并忽略其他维表的二级关联关系，则相应的雪花形数据模型如图 3.24 所示。

在实际应用中，可根据需要把一个数据立方体模型实例化成多个不同的数据立方体。此外，Cubing Services 还允许修改数据立方体模型及其实例，并对数据立方体模型进行验证。

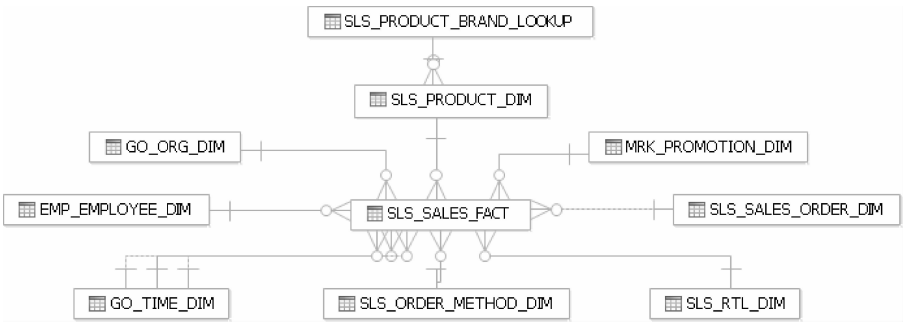

图 3.24 雪花形数据模型(二)

本章参考文献

[1] W H Inmon. Building the data warehouse[M]. New York: John Wiley & Sons,2005.

[2] Hammergren T. 数据仓库技术[M]. 北京: 中国水利水电出版社,1998.

[3] W H Inmon, J D Weleh, K L Glassey. Managing the data warehouse[M]. New York: John Wiley & Sons,1997.

[4] 池太蔚. 数据仓库结构设计与实施——建造信息系统的金字塔[M]. 北京: 电子工业出版社,2005.

[5] Claudia Imhoff, Nicholas Galemmo, Jonathan G Geiger. Mastering data warehouse design: relational and dimensional techniques[M]. New York: John Wiley, 2003.

[6] Joachim Hammer. The Stanford data warehousing project[R]. IEEE Data Engineering Bulletin,1995.

[7] Pool J 著. 公共仓库元模型开发指南[M]. 彭蓉,刘进译. 北京: 机械工业出版社,2004.

[8] [美]Ralph Kimball, Margy Ross 著. 数据仓库工具箱: 维度建模的完全指南[M]. 谭明金,译. 北京: 电子工业出版社,2003.

[9] Jiawei Han, Sonny H S Chee, Jenny Y Chiang. Issues for on-line analytical mining of data warehouse [C]. Proceedings of SIGMOD'98 Workshop on Research Issues on Data Mining and Knowledge Discovery, Seattle, Washington,1998: 1-5.

[10] Jiawei Han, Micheline Kamber. Data mining: concepts and techniques(Third Edition)[M]. Elsevier Inc.,2012.

[11] 陈琦. 上海医保数据仓库的设计[D]. 上海: 复旦大学硕士论文,2006.

思考题

1. 讨论数据仓库与操作型数据库、数据集市的区别。
2. 如何认识数据仓库的几个特点? 这些特点与企业管理决策有什么关系?
3. 什么是元数据? 元数据有哪些用途?
4. 讨论 ETL 的过程,其中数据质量对这个过程有什么影响?
5. 什么是操作数据存储(ODS)? 为什么要使用 ODS?
6. 数据仓库有哪些模型? 举例说明。
7. 举例说明数据挖掘查询语言(DMQL)的应用。

8. 以图 3.9 销售主题为例，给出数据仓库的概念模型和逻辑模型，并用 IBM DB2 Data Warehousing Design Studio 或 Microsoft SQL Server 2000 Analysis Services(或以上版本)实现。

9. 研讨题：阅读下面论文，讨论中药临床数据仓库系统的组成，分析数据仓库的相关主题。

Xuezhong Zhou，Shibo Chen，Baoyan Liu，et al. Development of traditional Chinese medicine clinical data warehouse for medical knowledge discovery and decision support[J]. Artificial Intelligence in Medicine，2010，48(2-3)：139-152.

10. 画出本章 3.9 节医保数据仓库的模型，并讨论在此基础上的决策分析。