第3章 数据仓库

3.1 前 言

在第2章学习了数据预处理的相关知识，数据在进行数据集成、清洗、规约等数据预处理操作后，需要将数据存入挖掘信息的存储载体——数据仓库中。目前企业中存在越来越多的历史数据，企业处理历史数据的主要方式有删除历史数据、备份历史数据、数据预处理后存入数据仓库三种方式，其中存入数据仓库中的数据可以作为决策支持的依据。日益重要的数据挖掘平台可以提供联机分析处理(OLAP)、在不同的数据粒度层面实现多维度的数据分析等，同时数据仓库也可以为后期的、联机分析挖掘(OLAM)提供支持。本章将系统地介绍数据仓库的基本知识。

本章内容安排如下，3.2节将回顾数据库的基本概念，该部分内容有助于深入理解数据仓库的概念及数据库与数据仓库之间的区别。3.3节将介绍数据仓库的基础知识，其中包括数据仓库的概念、特点、作用，同时会将数据仓库与数据库管理系统(DBMS)进行对比，以进一步的加深对数据仓库的理解。数据仓库基于多维的数据模型，这种模型把数据看成一种数据立方体的形式。3.4节将讨论如何进行 N 维数据建模，以及数据立方体的概念和模型。在此基础上进一步介绍概念分层，以及在多个抽象层次上进行数据的分析与挖掘。本节最后还将对数据仓库上重要的数据操作——典型的OLAP操作进行探讨，并说明其操作数据立方体的方式。3.5节首先介绍数据仓库的视图概念、数据仓库设计的方法和步骤，然后介绍数据仓库的体系结构以及每一层的原理，为后续设计数据仓库做准备。3.6节将讨论实现数据仓库的方法，实现是指有效计算数据立方体，包括采用多路数组聚集算法、构建数据索引、高效的OLAP查询步骤等内容。3.7节讨论从数据仓库如何过渡到数据挖掘，这节还将介绍数据仓库的应用，然后将OLAP与联机分析挖掘进行对比，最后将介绍OLAM典型的四层体系结构。

3.2 数据库基本概念回顾

本节主要回顾数据库的基本概念，由此可以更加深入地理解数据仓库的概念及数据库与数据仓库之间的区别。

在3.2.1节将简单回顾一下数据库的相关概念；3.2.2节简单回顾数据库中的基本概念，包括库表、记录、域等；最后3.2.3节介绍数据库管理系统的概念。在这里只是对数据库的基本概念进行简单的回顾，如果读者想深入了解数据库方面的相关知识，可查阅有关教材。

3.2.1 数据库简介

为了更好地理解数据库，首先需要了解一下关于数据的概念。

1. 数据

数据是数据库中存储的基本对象，它是用来描述事物的符号记录，数据可以为数字、字符串、日期等类型，它们都可以转换成相应的格式从而存入计算机。一个典型的数据包括数据对象及其属性。

2. 数据库

所谓数据库是指以一种结构化的方式存储数据的文件系统。

数据库具有较小的冗余度，较高的独立性和易扩展性，同时可以被多个用户访问的并发性，数据库中的数据可以长期存储在计算机内，因此可以把数据库理解为具有相互关联关系的数据组成的集合。

例如，日常生活中为了记录顾客在超市里的购买信息，通常会把商品购买信息记录在一个账本上，此时顾客购买商品信息便相当于数据库中记录的数据，而账本自身便相当于数据库。

3.2.2 表、记录和域

数据库、表、记录和域之间的关系是，数据库可以包含多张表，一张表中包含多条记录，一条记录中有多个域。

表是描述事物的数据组织成的二维表；记录是指数据表中每一行数据；域是指数据表中的每一列字段；当某个字段值在表中具有唯一性时，称此字段为主键，主键可以用来唯一地标识记录。记录和域在一张表中的表现形式如图 3.1 所示。

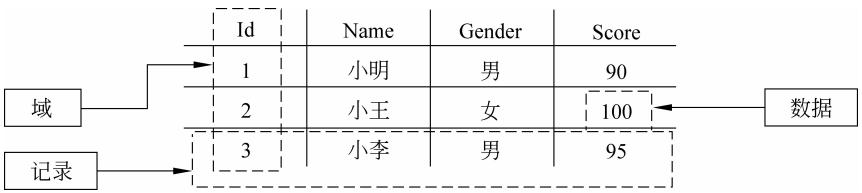

图 3.1 数据库中的域、记录和数据

3.2.3 数据库管理系统

1. 数据库管理系统简介

DBMS 是为用户提供定义、建立、维护数据库服务的软件，同时 DBMS 也为用户提供使用、操作数据库的功能。DBMS 对数据库进行统一管理和控制，其目的是为了保证数据库的安全性、一致性和完整性。

目前市面上流行的商业化 DBMS 主要有 DB2、Oracle、MS SQL Server、MySQL 和 MS

Access 等。

虽然 Sybase、Informix 等著名的 DBMS 已经退出了历史舞台，但是这些曾经辉煌的 DBMS 都曾广泛应用在很多大中型企业，为这些企业提供数据管理的支持。目前在数据挖掘中为了解决数据集成的问题，对于这些历史上曾经使用过的多源异构类型的数据也要加以考虑。

2. DBMS 主要功能

(1) 数据的存储、检索和更新。

该功能可以完成对数据的存储、查询、修改等操作，操作数据库过程中使用最频繁的便是此类功能。此类操作可以通过使用结构化查询语言(SQL 语言)加以实现，如创建增加(Create)、插入(Insert)、更新(Update)、删除(Delete)功能的语句。

(2) 事务支持。

事务可以理解为把数据库的一系列操作看成一个整体，此类整体要么整体有效，要么整体失效。事务的主要功能是为了保证存在于一个事务内的所有操作被执行或不被执行。

为了达到上述事务的功能，事务需具有原子性、一致性、隔离性、持久性的特点。

简而言之，所谓原子性是指多个对数据库的操作可以看作一个不可分割的原子；一致性是指数据库的状态从一个一致的状态转变到另一个一致的状态；隔离性是指不同的事务操作不会相互影响；持久性是指事务对数据库的改变是永久性的。

(3) 并发控制。

当多个事务并发地执行时，在数据库中就会产生同时读取或修改同一数据的情况。若 DBMS 不支持并发控制，则会导致数据不一致性问题。如多个用户对同一个数据进行更新操作，并发控制功能则会保证此数据的一致性。

以上是对 DBMS 的三个主要功能的简要介绍，其他 DBMS 功能请读者详见专门的数据库教材。

3.3 数据仓库简介

在介绍完数据库的基本概念后，引入本章的主题——数据仓库。在第 2 章介绍了数据预处理的相关知识，数据在进行数据集成、清洗、规约等数据预处理操作后，需要把数据存放到数据仓库中。

3.3.1 节将简述数据仓库的特点，3.3.2 节将介绍什么是数据仓库，3.3.3 节介绍数据仓库的作用，3.3.4 节将对数据仓库与 DBMS 系统进行对比，最后 3.3.5 节介绍分离数据仓库的原因。

3.3.1 数据仓库特点

William H. Inmon 曾给出了数据仓库的概括性定义："数据仓库是一个面向主题的、集成的、时变的、非易失的数据集合，支持管理部门的决策过程"。

数据仓库具有面向主题的(Subject-Oriented)、集成的(Integrated)、时变的(Time Variant)、非易失的(Non-Volatile)4 个关键特点。下面详细说明这 4 个关键特征。

1. 面向主题的

(1) 围绕重要的课题或主题，如顾客、产品和销售。

(2) 着眼于决策者的数据建模和分析，而不是日常对数据的操作或事务处理。

(3) 通过删除一些对分析决策支持没有价值的数据，针对一个特定的主题为分析决策者提供简明扼要的信息呈现方式。

2. 集成的

(1) 数据仓库的建立是通过集成和整合多个不同的异构数据源，数据源包括关系型数据库、数据文件和联机事务记录等。

(2) 在数据仓库的建立过程中，数据清洗和数据集成技术得到应用。其目的是为了保证在集成不同数据源时，保证数据在命名规则、编码结构和属性度量等方面的一致性。此外，当数据被放入数据仓库时，数据往往经过了一定的转换。

3. 时变的

(1) 在时间层面上数据仓库中的数据明显地比操作性数据库中的数据存储时间要长，其表现为操作性数据库中的数据往往存储的是当前的数据，而数据仓库是从历史数据的角度提供数据。例如，数据仓库中存储的是 5～10 年之间的数据，而操作性数据库中存储的是当前时间段的数据。

(2) 在数据仓库中，关键结构都显式或者隐式地包含时间元素。与之不同的是，在操作性数据库中，关键结构不一定包含时间元素。

4. 非易失的

(1) 数据仓库物理地分开存放数据，而这些数据都来源于操作性数据库，最极端的情况下，如果数据仓库中的数据被损坏了，还可以通过操作性数据库中的数据信息进行恢复。

(2) 在数据仓库中，通常的操作行为如更新数据不会发生。此外，数据仓库并不需要事务处理、恢复、并发控制机制等操作。数据仓库中只有两种类型的数据操作方式：初始化装载数据和访问数据。

3.3.2 数据仓库概念

数据仓库由数据仓库之父比尔·恩门在 1991 年《建立数据仓库》一书中提出，并被广泛接受。数据仓库是一个环境，通常数据仓库把来源不同的数据进行集成，为用户提供决策和分析的平台，同时提供用户对信息处理的支持，通常而言数据仓库中对数据的操作不易在传统的数据库中实现。

目前，关于数据仓库的定义已有多个版本，很难给定严格的定义。简单来说，数据仓库是一种语义一致性的数据存储，数据仓库是决策支持数据模型的物理实现，此外它也存储了企业用于决策的数据。

可以把数据仓库看作一种体系结构，数据仓库的建立是通过集成多个异构数据源进行构建的。其实数据仓库也是一种数据库，其与 3.2 节介绍的数据库有很大的相似性，只不过

建立数据仓库的目的是通过结构化或者专门的查询得到数据分析的结果并且为企业决策提供支持。

3.3.3 数据仓库作用

通过以上对数据仓库的简单介绍，数据仓库的主要功能和作用究竟是什么呢？在商业决策中，数据仓库的作用主要表现在如下几个方面：

(1) 提高客户的关注度。

通过分析客户的购买行为信息，可以获得客户购买商品的模式和购买的喜好倾向等信息。

(2) 微调生产策略。

通过分析历史产品的销售情况，进而重新配置产品和管理产品的组合，最大程度地提升利润。

(3) 查找利润来源。

通过对历史产品销售数据的分析，确定利润的来源，进而对产品的销售进行指导，提升利润。

(4) 管理客户之间的关系。

通过管理客户之间的关系，进而对公司的管理和运行提供指导。

此外，存放在数据仓库中的数据是集成多个异构数据源中的数据信息，同时企业中往往存在各种各样不同的数据源。通过建立数据仓库，企业可以有效方便地对上述异构数据源进行统一管理。

3.3.4 数据仓库与 DBMS 对比

通过数据仓库与 DBMS 的对比，可以更加深刻地理解数据仓库的作用和特点。为了进行深入的对比，首先介绍与 DBMS 和数据仓库相关的 OLTP 和 OLAP 操作。

1. OLTP 与 OLAP

(1) OLTP。典型的关系型数据库的主要任务是联机事务处理和查询处理，其中联机处理也就是常说的 OLTP (On-Line Transaction Processing)，OLTP 操作包含大部分日常操作，例如购买、库存、银行、生产、工资、登记、注册和记账等操作。

(2) OLAP。数据仓库的主要功能是实现联机分析处理 OLAP (On-Line Analytical Processing)，联机分析处理的主要目的是为了数据的分析和决策。

2. OLTP 和 OLAP 的主要区别

(1) 处理对象：OLTP 是面向顾客的，为顾客提供事务处理和查询处理等操作；OLAP 是面向市场的，为数据分析人员提供数据分析的支持。

(2) 数据内容：OLTP 处理的数据是当前详细的数据；而 OLAP 处理的数据是历史的数据，合并集成统一后的数据。

(3) 数据库的设计：OLTP 系统是采用“实体-关系”模型，也就是 ER 图的数据模型和面向应用的数据库设计；而 OLAP 往往采用星型模式和面向主题的数据库设计，其中星型

模式将在后续章节中进一步说明。

(4) 视图：OLTP 关注的是当前和本地的数据，而不去关注历史的数据信息；与之不同的是，OLAP 关注的数据是不同演变和不同数据源集成过来的数据信息。

(5) 访问模式：OLTP 中访问模式包括对数据的更新、查询等操作，这种操作需要并行化的控制和恢复机制，与在 3.2 节中提到的 DBMS 功能一致；而 OLAP 的数据访问模式主要是只读操作，而且这种读操作大部分是比较复杂的查询操作。

3. OLTP 与 OLAP 的其他区别

OLTP 和 OLAP 的主要区别如上所述，其他的区别如表 3.1 所示。

表 3.1 OLTP 与 OLAP 区别

区 分 点	OLTP	OLAP
用户	IT 专业人员	数据分析人员
功能	日常的操作	决策支持
数据库设计	ER 图和面向应用	星型模式和面向主题
使用方式	重复	专门的分析处理
访问模式	读/写	主要为读
工作单元	短的，简单的事务	复杂的查询
记录数量	大规模	海量大数据
用户数	海量	小规模
数据库大小	100MB～1GB	100GB～1TB
度量	事务吞吐量	查询吞吐量，相应时间

3.3.5 分离数据仓库的原因

数据库里面可以放大量数据，那么为什么还需要把数据重新放入数据仓库中呢？原因主要有以下两个方面：

1. 提高二者的性能

DBMS 主要设计用来进行 OLTP，如建立索引，进行并发访问的控制，建立恢复机制等。而数据仓库主要设计用来进行 OLAP，例如复杂的 OLAP 查询，多维视图的数据组织方式，数据的集成。如果使用 DBMS 进行 OLAP 操作，可能会大大降低操作的效率和性能。

2. 不同的功能和数据

决策支持需要查询历史数据，而事务型数据库不维护历史的数据，决策支持需要整合异构数据源中的数据。此外，存在数据仓库里面的数据都是高质量的数据，如在整合不同的异构数据源时，存在不同介质中的数据经常出现不同的编码方式、数据格式，在把这些数据放入数据仓库之前，需要进行数据“清洗”等数据预处理工作，才能把数据放入数据仓库中。相

比之下，事务型数据库需要维护的只是原始数据，而且在对数据进行操作的时候也无须进行数据的预处理。

分开的另外一个原因是为了分别提高数据库和数据仓库的性能。由于数据库和数据仓库分别具有不同的功能，存储的数据内容也有差异，因此在实际应用中需要将两者分离，区别对待。但是需要注意的是，目前越来越多的关系型数据库直接支持 OLAP 操作，也许随着数据库的发展，OLAP 和 OLTP 系统之间的差异会越来越小。

3.4 多维数据模型

3.3 节介绍了有关数据仓库的基本知识，然而数据仓库基于高维的数据模型，这种模型把数据仓库中所有数据抽象成一种数据立方体的形式。

本节将介绍如何对 N 维数据进行建模。3.4.1 节将讨论数据立方体的概念。3.4.2 节将介绍概念模型，也就是多维数据模型，包括星型模式、雪花模式和事实星座模式等。为了在多个抽象层上进行数据的分析与挖掘，3.4.3 节介绍概念分层的基本概念。3.4.4 节将介绍典型的 OLAP 操作，OLAP 操作是指操纵数据立方体的方式。最后 3.4.5 节将介绍数据仓库的设计与实现方法。

3.4.1 数据立方体

数据立方体 是指从多维的角度对数据进行观察和建模。

为了更加容易理解数据立方体的概念，首先引入一个例子，如在电子商品销售(All Electonics)的数据仓库中，可以从多个角度看待和建立数据模型。对于电子商品销售数据仓库，可以从商品信息、销售时间等维度来分析数据。

1. 维表和事实表

所谓维是分析和看待数据的角度，而每一个维度都可以有一个与之对应关联的表，这样的表称为维表，维表中是一系列属性集合，而维表是为了进一步描述维。在商品销售数据仓库中可能有很多维度的表，如商品信息维表可以包含商品名称、商品品牌和商品类型等属性，时间维表可以包含天、星期和月份等属性信息。

这里要引入一个“事实”的概念。所谓事实是数据度量的，如在上述商品销售的数据仓库中，事实可以是销售量、销售额等信息，事实是分析维之间关系的关键。事实表中包含事实的名称或者度量信息，以及相关维度的编码。3.4.2 节将介绍概念模型，届时可清楚地理解维表、事实表以及二者之间的关系。

2. 数据立方体维度

一个数据仓库中，所谓维的数量是指从多少个角度来分析看待其所存储的数据。一个包含所有维的方体被称为基础方体，它是组成整个数据立方体的单元，不包含维的基础方体存放在最高层，称作顶点立方体，它包含所有数据的汇总信息。而数据立方体是指多维数据模型方体的集合。

3. 多维分析

为了更加具体地理解以上抽象的概念，以电子商品(All Electonics)销售数据仓库为例说明数据立方体的概念。这个数据仓库的维度、维表和事实表的信息如表 3.2 所示。

表 3.2　电子商品销售数据仓库的维表和事实表

维　　度	time,item,branch,location
维　　表	time(time_key day day_of_week month quarter year)item(item_key item_name brand type supplier_type)branch(branch_key branch_name branch_type)location(location_key street city province_or_state country)
事实表	(time_key item_key branch_key location_key dollars_sold units_sold location_key)

(1) 首先从二维的角度观察温哥华(Vancouver)的电子商品销售数据。如表 3.3 所示，从时间维度和商品类型维度两个维度进行观察。表 3.3 中所显示的事实度量是指销售金额(单位:美元)。

表 3.3　从时间维度和商品类型维度观察温哥华的电子商品销售数据

Location="Vancouver"				
Time(季度)	Item(类型)			
	家庭娱乐	计算机	安全产品	电话
Q1	605	825	400	302
Q2	680	920	512	401
Q3	781	1026	501	350
Q4	824	1120	580	420

(2) 从时间、商品类型、供应方和销售地 4 个维度来多维度地观察电子商品销售的数据仓库，其中事实度量为销售额(单位：美元)。因为显示四维的数据比较困难，所以把 4 个维度的立方体映射成三维立方体的序列来显示，如图 3.2 所示。

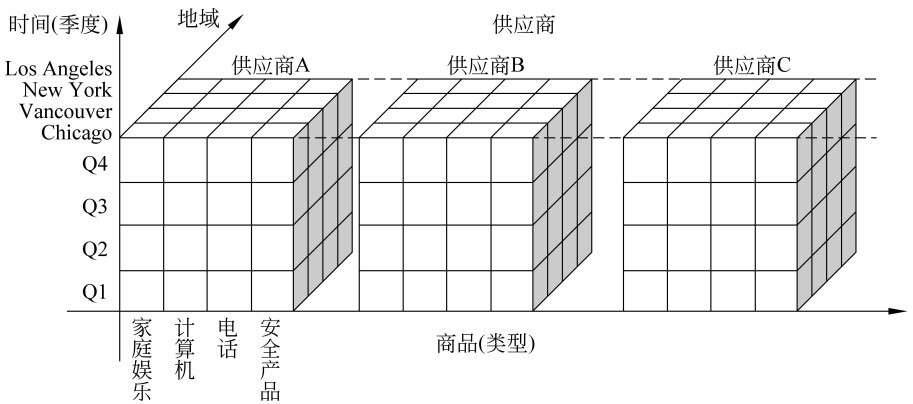

图 3.2　从时间、商品类型、销售地和供应方 4 个维度进行多维度观察的电子商品销售的数据仓库

(3) 时间(time)、商品(item)、地域(location)和供应方(supplier)形成的数据立方体方格如图 3.3 所示,可以从每一个维度对上述数据进行汇总和分析。放在底层 4-D 的方体为基础方体,0-D 方体为所有维度数据信息的汇总为顶点方体,可以在图 3.3 中直观地得出结果。

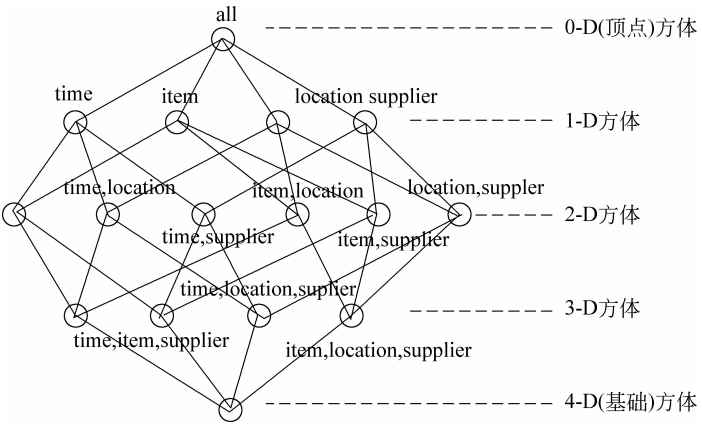

图 3.3 时间、商品、地域和供应方维度形成的数据立方体的方格

通过上述分析,数据立方体不仅可以进行二维、三维的数据分析,并且可以进行高维的数据分析。其中数据立方体中每个单元存放一个聚集值,该值对应于多维数据空间中的一个数据点。

此外,每个属性都可能存在层次化的概念分布,允许在多个抽象层进行数据分析。例如对 location 进行概念分层,可以把销售的城市聚集成国家。关于概念分层,将在 3.4.3 节详细讨论。由于数据立方体提供对预先计算的汇总数据进行快速访问,因此适合联机数据分析和数据挖掘。

3.4.2 概念模型

在 3.4.1 节中讨论了维表、事实表的概念和多维分析的过程,其中维表和事实表在数据立方体中起到了关键的作用。但是维表、事实表与普通的数据库库表有什么区别? 二者之间的关系是什么呢?

首先,通常数据库的设计与实现中常用到的是"实体-关系(ER)"数据模型,数据库中表之间的联系由 ER 图中的信息进行表示,这种模型适合 OLTP 操作;而数据仓库中的数据模型是多维数据模型,多维数据模型更适合 OLAP 操作。根据数据仓库中不同数据维度之间的关系,数据仓库中常见的数据模型有如下几种类型:星型模式、雪花模式和事实星座模式。这三种数据模型的区别主要是维表和事实表之间的关系。

1. 星型模式

星型模式是数据仓库中最常见的数据模型。这种数据模型中,事实表处于中心位置,事实表和其他维表相关联。为了更加形象地说明星型模型的概念,图 3.4 描述了电子商品销售数据仓库中的星型模式数据模型。

图 3.4 所示的数据模型从时间(Time)、商品(Item)、部门(Branch)和地域(Location)4 个维度描绘了数据仓库中的数据。这 4 个维度的信息分别由 4 张维表来描述,维表中描述

了各个维度的属性信息。而销售数据(Sales)是事实表,事实表中包含销售金额、单位销售金额和平均销售金额等事实数据信息。另外,此事实表中通过 4 个编码字段(time_key、item_key、branch_key 和 location_key)与维表进行关联。从图 3.4 中可以看出,星型模型的一个显著特点是所有维表都直接连接到事实表。

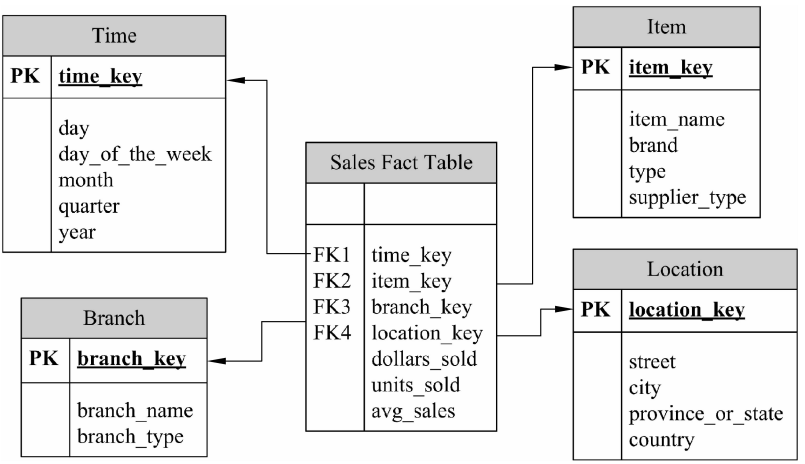

图 3.4　星型模式

2. 雪花模式

雪花模式是星型模式的进一步改进,其中把一些维表细分得到一系列更低层次的维表,最终形成的多个层次化的维表就像雪花一样,故称为雪花模式。雪花模式的一个显著特点是一个或多个维表没有直接连接到事实表上,而是通过其他维表连接到事实表上。

电子商品销售数据仓库的雪花模式表示如图 3.5 所示,通过对比图 3.4 和图 3.5,可以看出星型模式和雪花模式的主要区别在于维表,星型模式中 Item 维表和 Location 维表进一步细分,Item 维表细分成 Item 维表和 Supplier 维表,Location 维表细分成 Location 维表和 City 维表。

通过对比星型模式和雪花模式,可以看出:雪花模式更加规范,解决了部分冗余数据信息,能够有效地减少数据量,但是在雪花模式条件下,查询功能需要更多个表之间的连接操作来实现,相比星型模式,雪花模式的执行效率会比较低。

相比之下,星型模式由于最大限度地减少数据存储量以及集成了较小的维表,因此其查询性能较好,在数据冗余可以接受的范围条件下常常采用星型模式,以提高查询和维度分析的速度。所以在实际项目中,需要根据数据情况和项目要求确定采用哪一种数据模型。

3. 事实星座模式

数据模型中如果出现多个事实表共享一个或多个维表,此类数据模型称为事实星座模式。事实星座可以看作多个星型模式的集合。该模式的显著条件是模式中含有多个事实表并且事实表共享维表。

电子商品销售数据仓库的事实星座模式表示如图 3.6 所示。在该模型中,一共有两个事实表 Sales 和 Shipping 事实表,Sales 事实表与前面星型模式中的事实表一致,新增加的

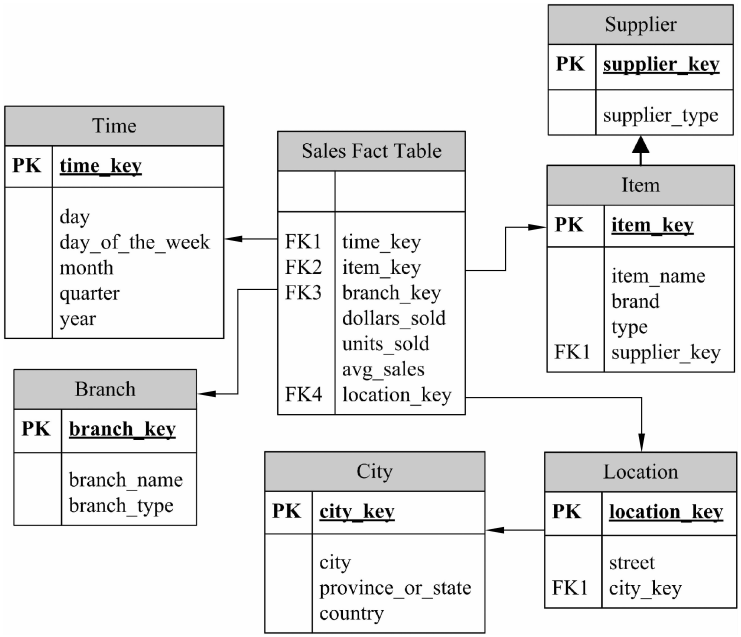

图 3.5　雪花模式

Shipping 事实表含有(item_key、time_key、shipper_key、from_location、to_location)分别与 Location 维表、Shipper 维表、Item 维表和 Time 维表相连,其中 Shipping 事实表和 Sales 事实表分别共享 Location 维表、Item 维表和 Time 维表。

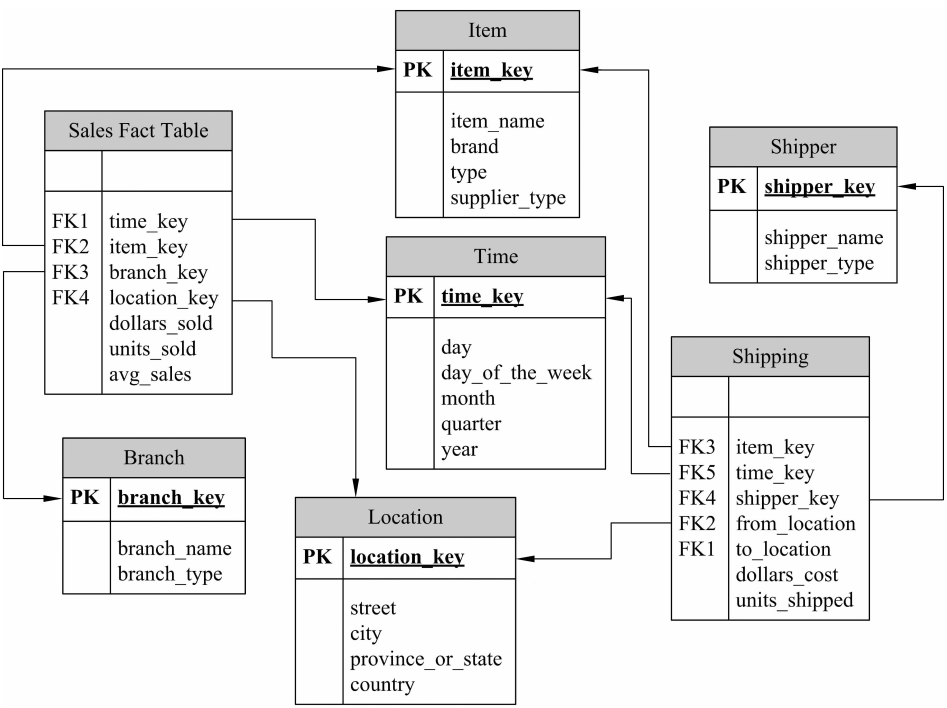

图 3.6　事实星座模式

如图 3.6 所示，事实星座模式可以对应多个分析和挖掘的主题。现实中，企业数据仓库的应用内容比较复杂，常常需要挖掘和分析多个主题相关的内容，所以在这种情况下通常使用事实星座模式。

在介绍完数据仓库的数据模型后，需要明确上述三种模型的应用范围。首先需要明确有关数据集市的基本概念。一个典型的数据集市受限于选定的挖掘和分析主题，而且其数据往往是企业数据的子集。例如电子商品销售数据，把主题限定于顾客、商品和销售。数据集市中的数据仅仅是企业数据的一部分或者是企业数据的汇总性数据。

如表 3.4 所示，对数据仓库和数据集市的相关方面进行了一个简单对比。

表 3.4　数据仓库和数据集市对比

	数 据 仓 库	数 据 集 市
主题	整个组织	选定的主题
范围	企业范围	部门范围
概念模型	事实星座模式	星型或雪花模式

3.4.3　概念分层

所谓概念分层是指定义了一个映射序列，这个映射序列把底层概念映射成较高层的概念，更一般化的抽象概念。对于给定的某一维，往往不仅有一层的概念层，进行概念分层的主要目的是为了在多个层次上对数据进行挖掘和分析。

在上例中的 location 维，location 的值中城市一列包括多伦多、温哥华等，可以把城市信息映射到国家，如多伦多和温哥华城市映射到加拿大；国家信息可以映射到洲，如加拿大映射到北美洲，依此类推，这便是概念分层，如图 3.7 所示。

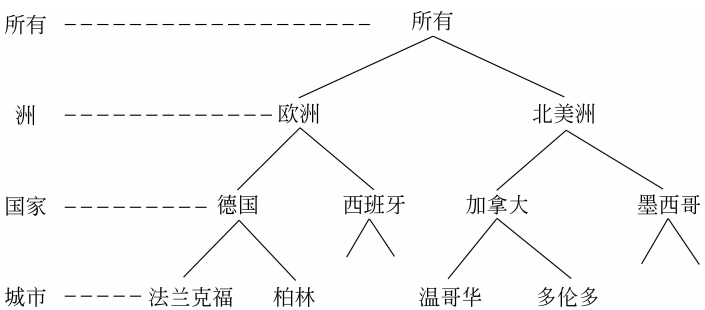

图 3.7　location 维的概念分层

按照概念分层的思想，可将维度 Product、Location 和 Time 等进行如图 3.8 所示的概念分层，在分层表示的概念中 Product＜Category＜Industry 和 Office＜City＜Country＜Region 满足全序的关系，Day＜{Month＜Quarter，Week}＜Year 满足偏序关系，这种属性的全序或者偏序的概念分层称作模式分层。

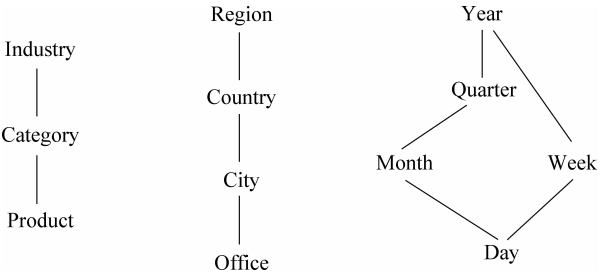

图 3.8　location 维的概念分层

3.4.4　典型 OLAP 操作

如何利用概念分层进一步挖掘数据中存在的知识呢？为了解决这个问题，首先需要引入一些典型的 OLAP 操作，包括上卷、下钻、切片、切块和旋转，通过这些操作可以很清晰地体会到概念分层是如何得到应用的。

OLAP 是由关系数据库之父 E. F. Codd 于 1993 年提出的一种动态数据分析模型，OLAP 操作对来自多源异构的经过集成和预处理后的数据采用多维结构的数据模型进行访问和操作。为了更加形象地理解 OLAP 操作，通过存储对电子商品销售的数据仓库进行演示，其中数据立方体包含地域(Location)、时间(Time)和商品(Item)等维度信息，地域维度按照城市进行聚集，时间维度按照季度进行聚集，商品维度按照商品的类型进行聚集，所显示的数值是销售额(单位：美元)。下述几种 OLAP 操作用图 3.9 进行了示例。

1. 上卷(roll-up)

通过在一个维度上的概念分层向上攀升或者通过维归约(即维度信息由细粒度向粗粒度归约)的方式在数据立方体中进行聚集，其本质是数据聚集到概念的上一层，进而得到汇总结果。如可以对数据立方体进行上卷操作，统计一年中不同城市、不同商品的销售情况。上卷操作可以通过消除一个或者多个维度，进而从更宏观的角度分析数据。

如图 3.9 所示，上卷操作是按照 location 维的概念分层，由 city 层上卷到 country 层，导致的结果是数据立方体按照 country 进行分组，而不是 city。上卷操作往往会合并一个或者多个维度。

2. 下钻(drill-down)

下钻是上卷的反向操作，从概念分层的上层到下层或者是将粗粒度的维度信息扩展成多个细粒度的维度信息。如可以把季度这一维度进行下钻，进而得到不同月份在不同城市不同商品的销售情况。此外，下钻操作还包含在原有的数据立方体的条件下添加维度。

如图 3.9 所示，按照 time 维的概念分层，由季度概念层下钻到月份概念层，导致的结果是数据立方体按照月份进行分组，而不是季度。

3. 切片和切块(slice-dice)

切片操作是指在给定的数据立方体中选择一个维度进行分析，如图 3.9 所示的数据立

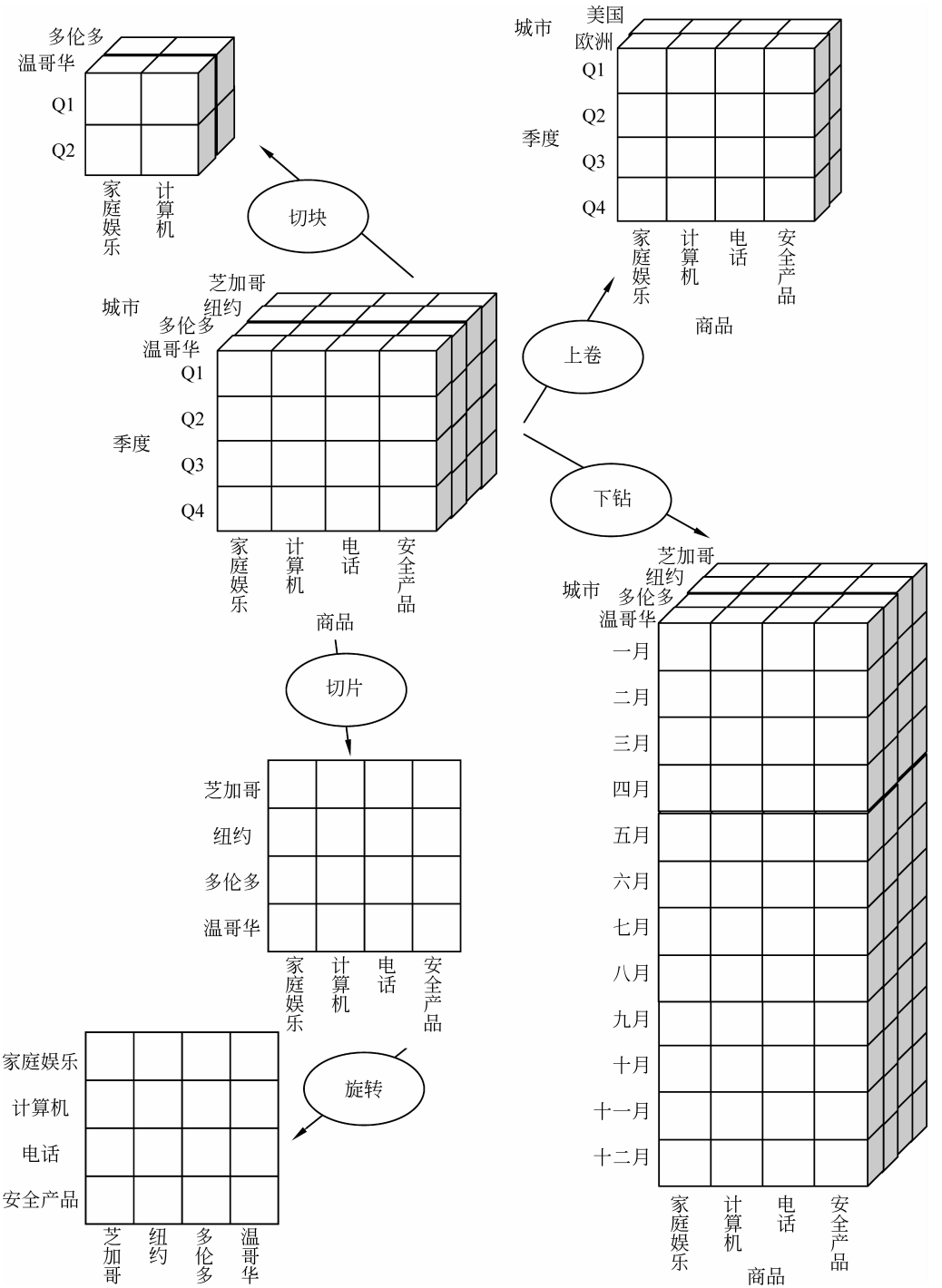

图 3.9　多维数据模型中典型的 OLAP 操作

方体可以通过切片操作得到第一季度中不同城市、不同商品的销售情况。

切块操作是指通过在两个或者多个维度上进行划分，从而得到子数据立方体。在如图 3.9 所示的数据立方体中，通过切块操作选择第二季度和第三季度在温哥华和多伦多，

计算机和家庭娱乐商品的销售情况。

4. 旋转(pivot)

通过对数据立方体进行不同角度的旋转,可以获取不同视角所呈现的数据立方体。

如图 3.9 所示,将 item 和 location 在一个 2D 层面上进行旋转操作。同理,可以在 3D 层面上进行旋转操作等。

5. 其他 OLAP 操作

钻过(Drill Cross)是指操作中涉及多个事实表中的数据。钻透(Drill-Through)是指通过执行 SQL 的语句形式,透过数据立方体的最底层,最终操作后台的关系表。通常情况下此类操作只有部分 OLAP 系统支持。

经过上面关于数据仓库中 OLAP 的介绍,读者可能会联想到统计数据仓库的概念,因为 OLAP 中上卷、下钻等操作也存在于数据库统计工作中。但是它们的侧重点不同,数据仓库侧重于商务上的数据分析和决策支持的应用,而数据库统计工作侧重于社会经济方面的应用。OLAP 操作的具体实现可参考本章有关参考文献。关于 OLAP 操作的应用程序接口(API),详见本章最后的参考文献。

3.4.5 星型网络的查询模型

在介绍完有关 OLAP 操作的基本概念后,可将此类操作应用于多维数据库查询。多维数据查询可以基于星型网络模型。所谓星型网络模型是指由一个中心点和多个射线组成,其中每一个射线代表一个概念分层,在射线上根据该维度给概念分层的信息确定 OLAP 操作或数据挖掘的粒度,将射线上对应的多个概念层次连接的折线称之为数据立方体的指纹(Footprint)。

电子产品销售数据仓库的一个星型网络模型如图 3.10 所示。在这个星型网络模型中,可以执行上述 OLAP 操作,如 time 层;可以沿着 time 维进行下钻,由 quarter 到 month;或者对 location 维进行上卷,由 city 到 country。用较高层抽象值替换较低层抽象值可以实现数据泛化,用较低层抽象替换高层抽象实现数据特殊化。

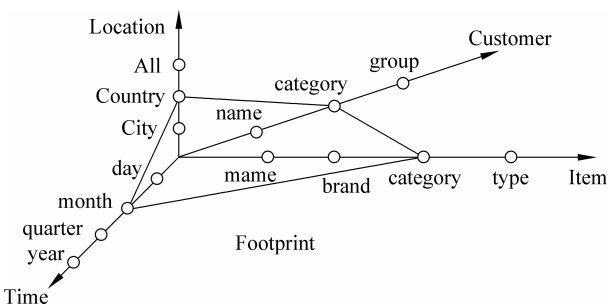

图 3.10 应用于查询的星网模型

3.5 数据仓库结构

数据仓库是一种多维度的数据模型，通过对数据立方体进行 OLAP 操作，可以从多个层次和多个维度实现分析和挖掘工作，进而获取隐藏在数据内部的有用知识。

本节将重点介绍数据仓库的体系结构。3.5.1 节首先介绍数据仓库的视图、设计方法和实现步骤；3.5.2 节介绍数据仓库的多维体系结构以及每一层的原理，为设计和实现一个完整的数据仓库做好准备。

3.5.1 数据仓库设计

1. 数据仓库设计视图

一个软件项目的研发过程是以需求分析为起点开展的。在进行项目需求分析时，需按照不同的分析对象和视图进行分析，如在构建商务网站时，往往需要从顾客、管理者和销售人员等相关的视图开始进行分析。

同样，在设计数据仓库时，首先要确定设计视图，它们分别是自顶向下视图、数据源视图、数据仓库视图、商务查询视图，这些视图综合在一起便形成一个完整的系统框架。不同的视图对应着不同的实现对象，如自顶向下的视图对应着自顶向下驱动的对象，数据仓库视图则对应数据仓库驱动的对象。下面简单介绍上述 4 种视图的基本概念。

(1) 自顶向下视图。这种视图从全局宏观角度设计数据仓库。

(2) 数据源视图。这种视图揭示了被操作系统获取、存储和管理的数据信息。

(3) 数据仓库视图。这种视图包含了多个事实表和多个维表。

(4) 商务查询视图。这种视图是从终端用户的角度观察在数据仓库中的数据。

介绍了数据仓库的设计视图后，下面介绍有关设计和实现数据仓库的主要方法。

2. 数据仓库设计方法

从不同的角度分析，数据仓库设计有不同的方法。

(1) 常见的方法有自顶向下、自底向上和上述两种方法的混合方法。

自顶向下是指对数据仓库的设计从总体的设计和规划开始，一直延续到低层的设计和实现工作。这种方法比较适用于对于被挖掘对象的应用需求具有明确把握和掌控的情况。由于具有对挖掘对象和目标的总体了解，这种方法的优点是可以从总体上规划数据仓库。

自底向上是指数据仓库的设计从实验系统和原型系统开始，这种方法在开发的早期比较实用，因为这种方法的主要优点在于设计速度快。

混合方法是指结合了自顶向下和自底向上各自的优势，既可以自顶向下从全局的角度规划设计数据仓库，也可以自底向上进行快速的数据仓库设计。

为了对比自顶向下和自底向上两种方法的区别，引入数据集市的概念，所谓数据集市是指企业范围内数据的一个子集，数据集市针对特定的用户群。图 3.11 显示的是自顶向下的设计方法，图 3.12 显示的是自底向上的设计方法。

通过图 3.11 和图 3.12，可以总结出上述两种设计方法的优点和缺点，如表 3.5 所示。在设计数据仓库的时候，根据需求选择适宜的数据仓库设计和实现方法。

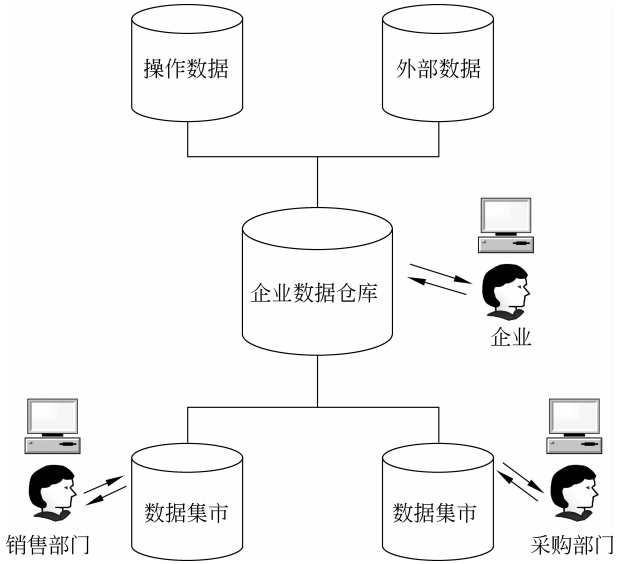

图 3.11　自顶向下的设计方法

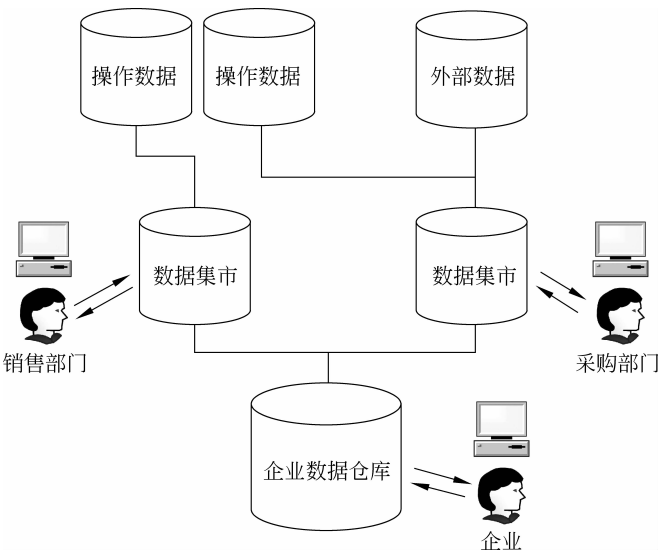

图 3.12　自底向上的设计方法

表 3.5　自底向上和自顶向下设计方法优缺点

方　法	优　点	缺　点
自顶向下	1. 一次性地完成数据重构工作 2. 最小化数据冗余度和不一致性 3. 存储详细的历史数据	1. 数据集市直接依赖于数据仓库的可用性 2. 投资成本不易实现短期回报，因为一次性建立企业数据仓库成本较高
自底向上	1. 快速投资回报收益 2. 设计方案可伸缩性强 3. 对不同部门的应用容易复制	1. 对每个数据集市需要数据重构 2. 存在一定的冗余及不一致性 3. 限制在一个主题区域

(2) 从软件工程角度分析数据仓库设计方法。

在一个典型的软件工程项目中,软件项目的开发方法有瀑布式和螺旋式两种主流方法,同理可将这两种方法应用到数据仓库的设计中。在这里首先回顾一下瀑布式方法和螺旋式方法的主要思想。

瀑布式方法:瀑布式是软件行业最早普遍采用的开发方法,此类方法通过将项目划分为多个有限阶段并按顺序逐步完成各阶段的开发任务。简而言之,瀑布式是指在每次进行下一步设计时,每一步都进行系统结构的详细分析与设计。

螺旋式方法:螺旋式是一种演化软件开发过程模型,以演化的开发方式为中心,每一个阶段使用的方法是瀑布式方法,此方法会快速产生功能渐变的系统,新功能产生的周期很短。

3. 数据仓库设计步骤

通常情况下设计数据仓库的步骤如下。

(1) 针对相应的商业业务流程进行建模,如下订单的过程、开发票的过程等。

(2) 确定商业业务流程中被处理的信息粒度。信息粒度的确定是由数据仓库设计人员决定的,如信息粒度可以是一天的交易记录等。

(3) 选择用于每一个事实表记录的维度,维度的选择往往是时间、地域等维度信息。

(4) 选择用于度量每一个事实表记录的度量信息。如商品销售额信息。

3.5.2 多层体系结构

1. 数据仓库三层体系结构

数据仓库通常采用三层体系结构,如图 3.13 所示,从最底层到最高层依次为数据仓库、OLAP 服务器和前端工具等。

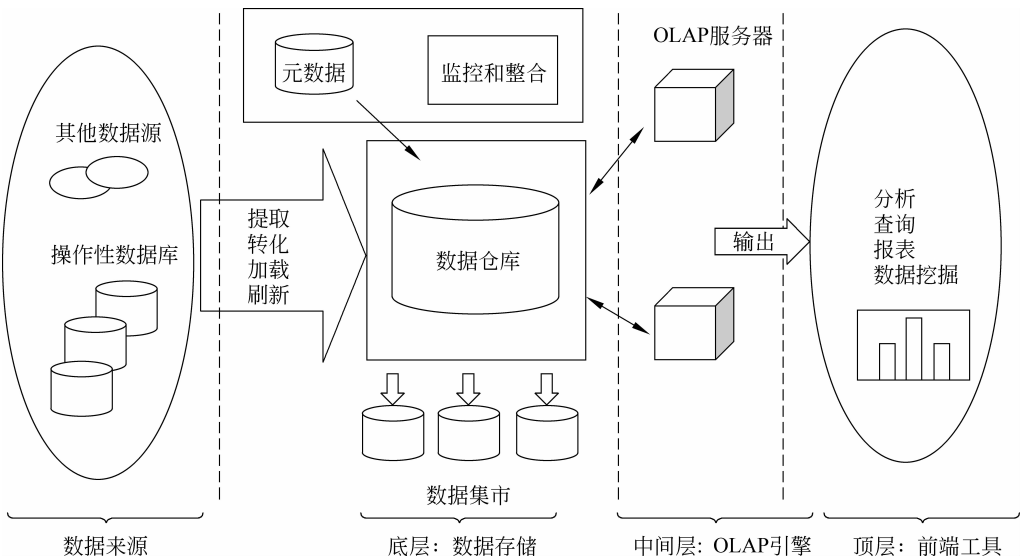

图 3.13 数据仓库三层体系结构

(1) 最底层是数据仓库。数据仓库的主要作用是用于存储数据，数据来源是通过集成操作性数据库和其他异构数据源中的数据，然后经过数据清洗、抽取、转换和集成，最后把数据放入数据仓库中。

(2) 中间层是 OLAP 服务器。OLAP 服务器是专门用于实现多维数据挖掘和分析的服务器。OLAP 处理的数据仓库操作类型或提供服务，包括关系 OLAP(ROLAP)服务、多维 OLAP(MOLAP)服务、混合 OLAP(HOLAP)服务和特殊的 SQL 服务。

(3) 最顶层是前端工具。可以通过中间层的分析，输出数据的报表信息、分析结果，并在中间层的基础上对数据进行数据挖掘与分析操作。

2. 元数据

在图 3.13 中，中间层中的元数据是关于数据的知识或信息，元数据是用于定义数据仓库对象的数据或信息。

元数据功能：元数据能提供基于用户的信息，如记录数据项的业务描述信息的元数据能帮助用户使用数据。元数据能支持系统对数据的管理和维护，如关于数据项存储方法的元数据能支持系统以最有效的方式访问数据。

元数据通常记录如下几类关于数据仓库的信息：

(1) 描述存储在数据仓库中的数据。

(2) 定义要进入数据仓库中的数据和从数据仓库中产生的数据。

(3) 记录根据业务事件发生而随之进行的数据抽取、清洗、转换和集成的调度计划。

(4) 记录数据一致性的要求和检测结果。

(5) 记录评估数据质量的方法和相关结果。

3.6 数据仓库的功能

在 3.5 节讨论了数据仓库这种多维的数据模型及其三层体系结构和设计方法。那么应该如何实现数据仓库的功能呢？所谓实现是指如何操作数据立方体，进而完成数据的查询等操作。

数据仓库存储的数据规模具有海量性。如何高效地进行 OLAP 操作变成了一个非常关键的问题。本节将讨论数据仓库中高效的数据立方体计算的实现方法。在 3.6.1 节，学习数据立方体的有效计算，其中包括数据立方体的计算、数据立方体的物化、优化数据立方体计算策略、多路数组聚集方法计算数据立方体等主要内容，在 3.6.2 节，将介绍索引 OLAP 数据，学习如何进一步地提高数据立方体操作的速度。最后在 3.6.3 节，将学习对数据立方体进行 OLAP 查询处理的步骤。

3.6.1 数据立方体的有效计算

1. 数据立方体个数计算

为了更加清楚地理解数据立方体，在这里引进一个例子：对电子商品销售构建一个数据立方体，分别有 city、year 和 item 三个维度，事实信息为 sales_in_dollars，如图 3.14 所

示。可以在这个数据立方体中按照 city、year 查询销售数据，也可以单独按照 city 或者 item 查询销售数据。

通过图 3.14 明显看出共 8 个立方体，分别是(city，item，year)、(city，item)、(city，year)、(item，year)、(city)、(item)、(year)和()，其中()表示分组为空，不进行任何分组。

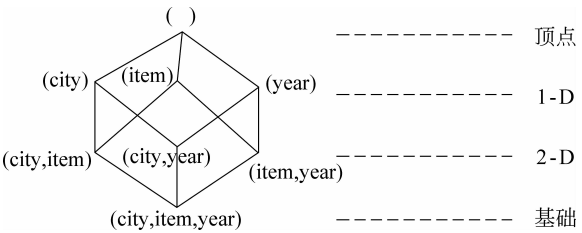

图 3.14　电子商品销售数据立方体

在 3.4.1 节曾经提到过，一个包含所有维的方体被称为基础方体，它是组成整个立方体的单元。不包含维的基础方体是存放在最高层，被称作顶端立方体，是最高泛化的方体，对应图 3.14 的立方体，顶点立方体是指分组为空的方体。顶点方体是所有数据的汇总，基础方体是指同时包含 city、item 和 year 维的方体，也就是(city，item，year)，该方体返回 city、item 和 year 维度组合的销售数据，其中 1-D 方体、2-D 方体也在图 3.14 中进行了标示。

通过上面分析，在没有概念分层的条件下，对于一个维度为 N 的数据立方体，数据立方体的个数为 2^n，但是在实际情况中往往存在概念分层的情况，如 year 概念分层为 day＜month＜quarter＜year。在这种情况下，可能的数据立方体的个数计算公式如下：

$$T = \prod_{i=1}^{n} (L_i + 1) \tag{3-1}$$

其中 L_i 是与维度 i 相关联的概念层数，T 是方体的总数。

在 DBMS 中可以用 SQL 语言对数据库中的表进行定义和操作，同理也可以用类似于 SQL 对上述数据立方体进行定义和计算。

数据立方体定义语句：

```
define cube sales[item, city, year]: sum (sales_in_dollars)
```

数据立方体计算语句：

```
compute cube sales
```

对于数据仓库中的操作而言，可以将数据仓库的操作符 cube by 引入到 SQL 语句(cube by 由 Gray 在 1996 年提出)：

```
SELECT item, city, year, SUM (amount)
FROM SALES
CUBE BY item, city, year
```

2. 物化数据立方体

在式(3-1)中，如果数据立方体 year 维度的概念分层为 day＜month＜quarter＜year，则 year 维有 4 个概念层，加上虚拟层，共 5 个概念层。同理，假设数据立方体共 10 个维度，且