

如何有效使用雪花模型

曹叶虹, 奚建清

(华南理工大学 计算机系, 广东 广州 510640)

摘要: 雪花模型是大家所接受的适合数据仓库的模型. 为了说明如何有效地使用雪花模型, 本文针对现实生活中存在的一些问题, 提出了三种雪花模型的变种, 提高了雪花模型的解决问题的能力.

关键词: 数据仓库; 星型模型; 雪花模型; 维

中图分类号: TP311. 13

文献标识码: A

文章编号: 1007-7162(2002)02-0024-04

数据仓库是支持管理决策过程、面向主题、集成、稳定、不同时间的数据集合, 是存储数据的一种组织形式^[1]. 它从传统数据库中获得原始数据, 先按辅助决策的主题要求形成基本数据层, 再按综合决策的要求形成综合数据层. 随着时间的推移, 由时间控制机制将当前基本数据层转为历史数据层. 数据仓库中数据的物理存储形式有多维结构数据组织形式和基于关系数据组织形式两种. 其中基于关系数据库的组织形式又分为星型模型和雪花模型两种.

星型模型和雪花模型的基本结构都是事实表和维表, 通过维表中的主关键字列和事实表中的外来关键字列的匹配来连接维表和事实表. 每当维表存储在单独的表格中时, 数据库的结构被称为星型设计, 当数据库的维存储在一系列的表中时, 数据库的结构被称为雪花型设计. 因为雪花看来就像射出光芒的星星一样^[2]. 相比星型模型, 雪花模型减少了数据仓库结构的直观性, 降低了系统的检索性能. 但是采用雪花方式, 可以处理诸如多对多关系的结构、非分析性数据和层次结构. 在一个单独的数据仓库中, 有些维表可以使用星型设计而其他使用雪花型设计. 因此这里存在着一个问题: 究竟什么时候采用雪花模型才有效呢? 怎样在性能和可理解性程度之间取得一个比较好的折中呢? 客观世界是复杂的, 许多问题单靠星型模型或者传统的雪花模型是无法解决的. 这里提出了三种雪花模型的变种来解决现实生活中存在的这些复杂问题, 希望可以回答上面提出的问题.

1 雪花模型用于大型客户维的建立

在数据仓库中, 客户维是一个普遍存在的维, 也是一个十分复杂的维. 在一个大型的组织中, 客户维一般都是相当大的, 可能拥有上百万条记录和各种各样的属性. 以一个大型网站为例, 我们往往需要建立一个跟踪互联网的立方体, 以便统计网站的被访问情况和访问网站的客户情况. 一般我们将访问网站的客户分为两类: 临时用户, 即一般的访问者(visitor) 和固定用户(customer). 临时用户是匿名登录的, 虽然他们可能会多次出现, 但是我们不清楚他们的姓名、地址和相关的背景资料, 仅仅拥有一份表明他们访问过该网站的记录; 而固定用户相反, 他们都经

收稿日期: 2001-09-06

作者简介: 曹叶虹(1976), 女, 硕士研究生, 主要研究方向为数据库.

过详细可靠的注册. 我们可以从网站或是第三方清楚地知道他们的姓名、住址、历史记录和其他任何感兴趣的资料.

在多数情况下, 临时用户的数量是远远超过固定用户的数量的. 假设存放能够展现所需的量度的最底层细节数据的事实表中, 70% 的记录是关于临时用户的, 剩下的 30% 的记录是固定用户的. 我们仅仅考虑临时用户的两个行为属性: 最后一次访问的时间 (Lastvisittime) 和访问的频率 (Frequency). 另一方面我们假设每个固定用户拥有一共 50 个属性和量度, 包括该用户的所有资料, 比如姓名、住址、付费情况、信用程度、直接得来的资料和从第三方购买得来的资料.

下面我们将临时用户和固定用户逻辑上统一为一个维, 称为客户维 (Client). 每一个客户, 不论是固定的还是临时的, 都有一个唯一固定的标识符 (CleintID). 因此, 在逻辑上客户维拥有以下属性.

固定用户临时用户都有的属性:

Clientkey: 维表主键

Client ID

Lastvisittime: 最后一次访问时间

Frequency: 访问的频率

只有固定用户具有的属性:

5 个关于名字的属性: Name

10 个关于地点的属性: Location

10 个关于行为的属性: Behavior

25 个关于统计的属性: Demographic

注意, 在这里我们将 Lastvisittime 和 Frequency 作为维属性放在维表中而不是放在事实表中, 随着时间进行更新. 这样可以使客户维更加有效, 我们可以直接从维表中对客户进行划分而不需要连接到事实表.

假设固定用户拥有的 50 个属性中, 大部分都是文本类型的, 一条记录的长度最少为 500 个字节. 假设有一千万用户, 其中七百万为临时用户, 三百万为固定用户. 我们很容易的发现, 在一个大小为 10GB 的维表中, 占 70% 记录空间的临时用户, 因为不具备求固定用户具有的专有属性, 而使得每个临时用户至少 50 个域没有数据, 造成了空间的极大浪费. 这种情况不能不引起我们的注意. 因此这个时候我们需要引入雪花模型, 将这个客户维分解为一个基本的客户维表和下一级的子维表. 在子维表中, 每一条记录对应着一个固定用户, 记录中可以包含空 (NULL) 属性值. 如图 1 所示.

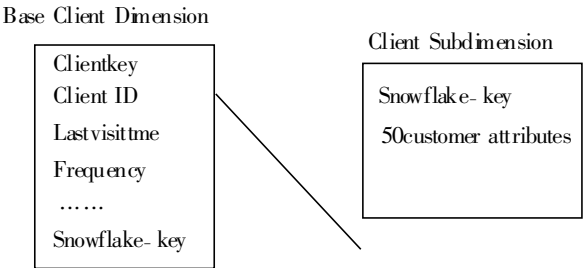


图 1 用雪花模型建立客户维

这种方法的思想类似于关系代数中的特殊化/概括思想^[3]. Client 好比于一个实体集, 而 visitor 和 customer 就是其中的两个实体. Visitor 和 customer 拥有一些共同的属性, 同时 customer 包含了一些 visitor 所没有的属性, 这样我们可以根据差异特征对 Client 进行特殊化, 唯一不同的是, 这里分离出来的不是实体, 而是部分实体具有的特殊属性.

在一个固定宽度的数据库中, 采用以上的假设, Base Client Dimension 的大小是 10 000 000*

25 B= 250MB, Client Subdimension 的大小是 3 000000* 475 B= 1.4GB. 可见采用雪花模型, 我们节省了至少 8GB 的空间.

2 雪花模型用于财政、金融机构中产品维的建立

银行、中介机构及保险公司在为他们的产品建维的时候总会遇到这样的问题: 每一个独立的产品都有许多不同于其他产品的特殊属性. 除了一些共有的核心属性, 一个经常帐户与抵押单据、存款单等产品是截然不同的, 甚至属性的个数也是不同的. 那么如何为这些产品建立一个维呢?

如果你建立一个拥有所有产品的单一的产品维, 包括所有产品所需的属性, 那么你可能发现居然有上百个不同的属性. 对于每一个产品对应的那条记录, 可能大部分的属性都是空的. 可以想象, 这样建维的后果是很严重的.

我们可以采用“基于内容”的雪花模型解决这个问题. 同上面的例子类似, 我们将共有的核心属性放入基本的产品维表中, 然后为每一个产品建立一个延伸的子维, 中间用键 (snowflake key) 相连, 如图 2 所示.

这里并不是传统的关系型的连接, Snowflake-key 必须能够根据产品的定义连接到对应的产品子维中. 我们可以建立一张关系表, 指明每一种产品所对应的子维表的连接路径.

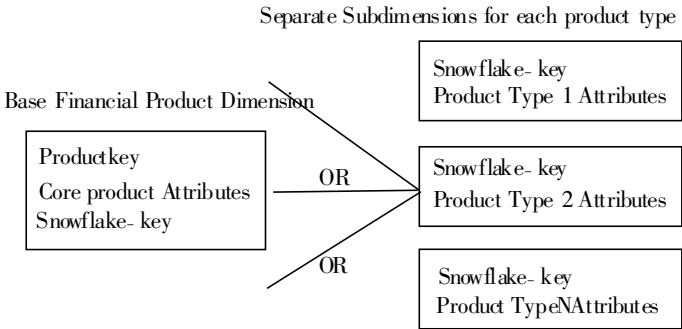


图 2 用雪花模型建立的产品维

3 雪花模型用于日历维的建立

日历维, 可以认为是时间维的一种. 虽然市场上很多的联机分析工具可以帮助我们很容易建立起一个包含有各种层次的时间维 (例如 MS SQL Server Analysis Services 中的 Time 维)^[2]. 但是, 在一个分布式数据仓库的环境中, 想要为各种各样的组织或者公司建立一个统一的日历维也不是一件容易的事情. 因为每个组织都有他们自己特殊的财政年, 其中对于季度的划分, 甚至是休假的时段也不一样. 虽然可以通过归类, 将相似财政年的公司放在一起帮助我们的商业分析, 但是用户总是希望从一个公司的角度去衡量其他公司的情况.

与上述的产品维不同, 每一种财政日历都有相同数目的属性, 如财政年、季度、假期等等. 但是这里有可能存在上百个不同的财政日历, 甚至更多. 一个大型的跨国公司必须能够处理位于不同地域的子公司的财政日历. 这也是一件麻烦的事情.

同样我们可以用雪花模型来处理这个问题. 如图 3 所示.

在图 3 中, 同样有一个基本的维表和一个子维表. 基本维表中含有不同的财政日历所共同的属性和连接子维表的

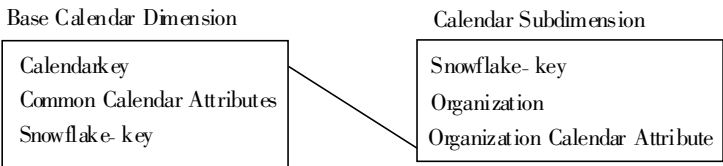


图 3 基于雪花模型的日历维

Snowflake_key. 这个 Snowflake_key 不仅是连接子维表的外键, 也是连接不同公司的外键. 与传统的雪花模型不同, 这里子维表的基数要比维表的基数高.

在每次连接维表和子维表之前, 必须在子维表中预先指明一个公司, 然后才能进行连接. 这时, 子维表到维表体现的是一个一对一的关系, 两者看起来就好像是一个单一的实体.

4 结束语

上述提出的三种雪花模型的变种都在不同程度上提高了雪花模型的解决问题的能力、拓宽了适用的范围, 希望可以给大家带来启发. 有一点要注意的是, 在选择维中所应该包含的属性时, 必须小心. 在实际情况中, 往往存在一些并不那么重要的数据, 但又必须在数据仓库中保存这些数据. 通过在维表中包含这些字段, 我们可以生成大型的、难以处理的表, 这将影响查询的工作, 并最终影响数据仓库的效率^[5].

参考文献:

- [1] W H Inmon. 数据仓库[M]. 北京: 机械工业出版社, 2000. 29-86.
- [2] Reed Jacobson. SQL Server 2000 Analysis Services 学习指南[M]. 北京: 机械工业出版社, 2001. 1-172.
- [3] Abraham Silberschatz. 数据库系统概念[M]. 北京: 机械工业出版社, 2000. 15-39.
- [4] Lou Agosta. 数据仓库技术指南[M]. 北京: 人民邮电出版社, 2000. 101-117.
- [5] William A Giovinazzo. 面向对象数据仓库设计[M]. 北京: 人民邮电出版社, 2000. 89-116.
- [6] Erik Thomsen. Microsoft OLAP 解决方案[M]. 北京: 人民邮电出版社, 2000. 1-21.
- [7] Alex Berson. Data Warehousing, Data Mining, & OLAP[第二版][M]. 北京: 世界图书出版公司北京公司, 1999. 169-187.

How to Use a Snowflake Effectively

CAO Ye-hong, XI Jian-qing

(Dept. of Computer Science South China University of Technology, Guangzhou 510640)

Abstract: One of the schema methodologies that is gaining widespread acceptance for data warehousing is a snowflake model. In order to explain how to use the snowflake model effectively, this paper puts forward three kinds of changes from the snowflake model. As a result, it can improve the application of existing model.

Key words: Data Warehousing; Star schema; Snowflake; Dimension