

# 一种通用聚合算法在统计工作中的应用研究

吕攀<sup>1</sup>, 余芳<sup>2</sup>

(1. 上海师范大学, 上海 200234;

2. 新乡医学院 管理学院, 河南 新乡 453003)

**摘要:** 目前政府统计部门正在积极建设统计数据库, 统计数据库具有典型的多维特征。同时, 统计业务人员经常要对各种级别的统计数据尤其是汇总级数据进行所谓的即席查询(Ad-hoc query)。为了实现这一点, 大多数统计查询和分析系统依赖于关系数据库平台本身提供的 OLAP 功能和接口, 造成应用系统与数据库系统紧耦合反而降低了应用系统的可移植性。文中根据统计数据库多维特征和 OLAP 聚合操作的原理提出一种跨数据库系统平台的通用聚合算法, 并成功应用于统计工作中, 有效地解决了上述问题。

**关键词:** 统计数据库; 统计分组; 联机分析处理; 通用聚合算法

**中图分类号:** TP301.6

**文献标识码:** A

**文章编号:** 1673-629X(2007)01-0219-03

## Research and Application of a Generic Aggregation Algorithm to Government Statistics

LÜ Pan<sup>1</sup>, YU Fang<sup>2</sup>

(1. Shanghai Normal University, Shanghai 200234, China;

2. Management Institute, Xinxiang Medical College, Xinxiang 453003, China)

**Abstract:** Nowadays, governmental statistical departments are building statistical databases actively. Statistical database has typically multidimensional character. On the other hand, statisticians often implement ad-hoc queries to all kinds of criteria for statistical data, especially to aggregation. To achieve this, the majority of statistical information and analysis systems rely on the OLAP functions and interfaces provided by relational database platform itself, resulting in applications and database systems tight coupling, which reduces the portability of applications. So design and implement a generic aggregation algorithm in the thesis that doesn't depend on a specific RDBMS according to multidimensional character of statistical database and principle of aggregation. The algorithm has been applied to statistics successfully, and provides a solution of the problem.

**Key words:** statistic database; statistical grouping; OLAP; generic aggregation algorithm

### 0 引言

统计数据库(Statistic Database)是一种用来对统计数据进行存储、统计、分析的数据库系统。统计数据具有层次型特点,但并不完全是层次型结构;统计数据也有关系型的特点,但关系型也不完全满足需要。概括起来,统计数据库具有以下的基本特征:

- \* 分类属性和统计属性<sup>[1,2]</sup>
- \* 多维性
- \* 分类属性的层次结构
- \* 微数据(Micro Data)和宏数据(Macro Data)

将具有分类属性的统计指标及其分类具体内容用维度表来存储,而对于具有计量属性的统计指标及其数值保存在事实表中,这样它们就以星型模型的结构组织起来。

与此同时,联机分析处理(OLAP)是多维数据如数据仓库系统的主要应用,支持复杂的分析操作,侧重决策支持,并且提供直观易懂的查询结果。具体地说,OLAP 是使分析人员、管理人员或执行人员能够从多角度对信息进行快速、一致、交互地存取,从而获得对数据的更深入的了解的一类软件技术。OLAP 的目标是满足决策支持或者满足在多维环境下特定的查询和报表需求,它的技术核心是“维”这个概念<sup>[3-5]</sup>。

显然,OLAP 操作适应统计数据的本质特征。也正因为如此,可以设计一套 OLAP 聚合算法,使其不依赖于保存统计数据的具体数据库系统,满足用户的查询和分析要求。

### 1 根据维度和度量实现聚合操作

在进行聚合操作之前,必须首先指定具有分类属性统计指标间的嵌套组合关系,这也就意味着所有维度中所含成员对应特定的组合。每一种组合情况映射到待汇总的

收稿日期:2006-04-10

作者简介:吕攀(1979-),男,河南开封人,硕士,工程师,研究方向为数据仓库与数据挖掘。

源数据视图上都可能生成一条汇总结果,除非源数据中根本不存在这种组合<sup>[6]</sup>。这里源数据视图其实就是汇总中涉及的所有统计指标所在表关联后获得的数据集合,如图 1 所示。

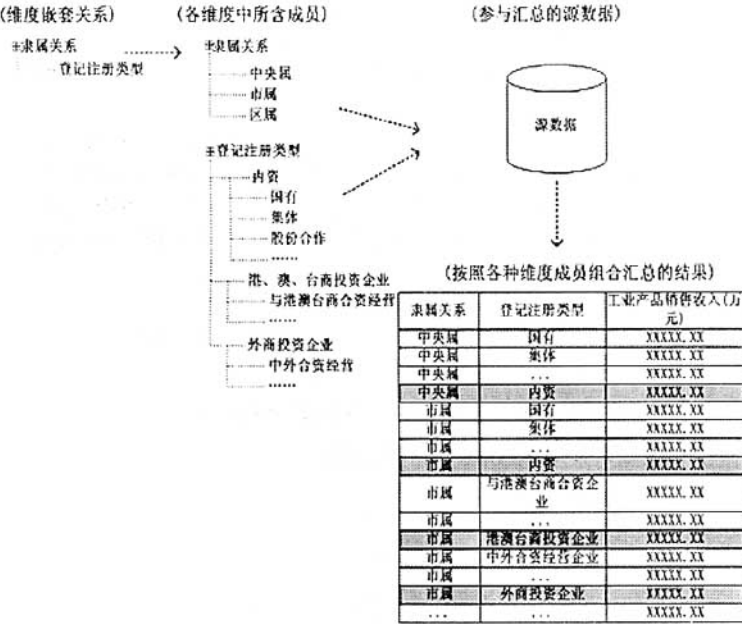


图 1 维度成员组合情况在源数据上投影并指导聚合

基于以上的基本原理,可以解析出用户定义的维度和度量信息并通过一套完整的通用聚合算法实现汇总过程。在执行汇总的时候,大致可以分为五个步骤,这五个步骤环环相扣,完成聚合源数据的提取,各种维度成员组合情况的生成,进行初步汇总获得所有最低级别成员组合对应的聚合数据,再次汇总获得所有较高级别直到最高级别成员组合对应的聚合数据,汇总结果展现等操作。下面就根据汇总操作执行步骤的前后顺序分别加以说明。

1.1 获取聚合所需的源数据视图

由于聚合操作主要是在内存中进行的,这样可以最大限度地减少访问数据库时的 IO 时间,从而大大提高了汇总过程的效率<sup>[7,8]</sup>。因此,首先根据用户指定的源数据表(维度表和事实表)和汇总条件一次性地从数据库中将所有满足条件的数据抽取出来,并装载到内存的数据集合中。

1.2 构建指导聚合的分组矩阵

操作人员在定义汇总操作信息的时候应根据一定的统计意义指定维度间的嵌套关系,即哪些维度是上下级的父子关系,哪些维度间是同级的兄弟关系。对于不同的嵌套关系,生成的汇总结果数据也不同。操作人员定义的维度嵌套关系被系统解析,构建一个记录着这种嵌套关系对应的所有维度成员组合的划分矩阵,也就是对上一步操作生成的源数据视图进行过滤并汇总的条件集合,用来指导聚合。本步骤就负责生成这样的一个矩阵,如表 1 所示。

表 1 维度成员组合矩阵

DimenIndic <sub>1</sub> ... hierarchy	DimenIndic <sub>N</sub> ... hierarchy	DimenIndic <sub>1</sub> ... hierarchy	DimenIndic <sub>N</sub> ... hierarchy	DimenIndic <sub>1</sub> ... hierarchy	DimenIndic <sub>N</sub> ... hierarchy
...	...	...	...	...	...

可以看到,该矩阵结构中的每一数据行都对应各维度成员的一种组合情况,该组合关系不但包含了当前组合中维度成员的信息(DimenIndic<sub>1</sub>至 DimenIndic<sub>N</sub>表示这些域内记录着维度成员代码),还记录了每个成员对应本维度中的直接上级成员代码(由域 DimenIndic<sub>1</sub>-hierarchy至 DimenIndic<sub>N</sub>-hierarchy 存储)以及每个成员对应本维度中的所有最下级成员代码(由域 DimenIndic<sub>1</sub>-members至 DimenIndic<sub>N</sub>-members 存储)。这样一来,如果获得了该矩阵中某种维度成员组合的情况,通过查找此矩阵还可以获知每个维度成员的上级成员是谁,以及所含的下级成员有哪些。系统中也正是利用这些信息,在生成最低级别成员组合对应的汇总数据后接着生成较高级别乃至最高级别成员组合情况下的汇总数据。

特别地,本步骤中只填入组合中包含的维度成员和相应上级成员信息,对应的下级成员在第五步再汇总的过程中才能填入,因为此时还不方便获取所有下级成员的信息。

1.3 生成多维数据集结构

最终的汇总结果数据存放在一个 DataTable 数据对象中,该 DataTable 记录着所有存在的维度成员组合信息,还有这些组合信息对应的汇总公式的计算数据。因此它的结构实际上是由两部分指标构成的,一部分是分类信息(维度 Dimension 信息),另一部分是统计信息(度量 Measures 信息)。这样也就是在内存中临时生成了一个逻辑上的多维数据集,操作人员可以对其进行各种定制的操作。该数据对象的数据结构如表 2 所示。

表 2 聚合结果集的数据结构

目录指标 1	...	目录指标 n	汇总公式 1	...	汇总公式 n

1.4 根据分组矩阵执行初步汇总

该步骤将前面生成的维度成员组合矩阵中的每条数据项作为过滤条件,从第一步生成的源数据视图内提取所有满足条件的记录并按照计算公式进行汇总。特别地,源数据视图内一般只包含维度中最底层(叶级)成员的信息,因此这样计算出来的汇总数据就一定对应各维度内叶级成员的相互组合,所以称为最低级别汇总数据。对于每个维度中上级成员间以及上级成员与下级成员间相互组合对应的汇总数据(这里相应地称为高级别汇总数据),则需

要在下一个步骤中依据最低级别汇总数据逐级向上汇总获得。

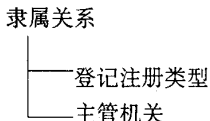
### 1.5 逐级向上汇总获得粗粒度聚合数据

在上一步骤中获得的汇总数据都是最低级别的,实际上较高层次的统计汇总数据应该出现在聚合结果集合中,因为统计业务人员和决策人员最关心的可能是这部分数据。

但是,为了获得高级别汇总数据,系统首先要知道某个底层成员的上级成员是谁,并且反之对于某个上级成员其包含的所有下级成员又是哪些,这样才能够据此信息对源数据视图再作汇总处理。另外,由于每个维度间的低级别成员与高级别成员或高级别成员与高级别成员还会相互组合,那么就要按照一定的顺序对所有维度内成员从低到高的交叉再汇总,最终获得各维度间下级成员与上级成员,以及上级成员与上级成员等各种组合情况下的汇总数据。这里需要再次扫描维度信息集合,构建维度层次搜索路径 explorationPaths,该路径恰恰决定了对每个维度间成员再汇总的顺序。

explorationPaths 实际上是一组链接表,记录了维度嵌套关系中从上到下每一条路径所经过的所有维度的代码信息,代码间用“;”号分隔,以便于后面操作中对维度节点的解析。

比如对于下面的维度嵌套关系



将会生成两条维度层次探测路径,用字符串形式表达分别为:

隶属关系;登记注册类型

隶属关系;主管机关

其中一条是从维度节点“隶属关系”到节点“登记注册类型”,另一条是从节点“隶属关系”到节点“主管机关”。不同的路径对应维度间各级别成员的组合情况也不同,这样就应当依据不同情况分别进行再汇总,保证彼此之间不会产生干扰。

接下来开始执行再汇总操作,与此同时将维度成员组合矩阵中各维度的下级成员信息填入。实际上,只有某个维度成员的下级成员域内容填写完整后,才能获得该成员所在组合的再汇总数据。因此,矩阵中的下级成员域也是在扫描过程中动态扩充的。

整个操作过程是递归的,对于每一条探测路径都从最底层维度开始处理,并对该维度中各级成员进行逐层汇总直至获取所有路径中所有维度的所有级别成员组合对应的汇总数据,则整个聚合操作完成。

采用.NET C#语言实现了这套算法,最终产生的聚合结果以多维分层的形式展现,如图2所示。

目录: 年份、月份、登记注册类型,统计计量: 工业销售产值

年份	月份	港、澳、台商	港澳台商独资	港澳台商投资	股份合作	股份有限公司
2003	01-06月份	23338746	8002393	325207	3096226	316836
	07-12月份	54408393	20318742	483686	7403979	815296
	10月	12229398	4620332	106047	1662089	190592
	11月	13722673	5221898	119693	1839619	214083
	1季度	6299635	2065454	126959	820745	89912
	1月	1033434	316539	25715	160802	16343
	2季度	17039121	5936939	196236	2275481	226923
	2月	2042769	680414	47200	262322	31388
	3季度	28456312	10475912	257926	3902271	410621
	3月	3223432	1088501	54054	397621	42181
	4季度	25952071	9842830	225740	3501708	404675
	4月	4424080	1515481	60241	551692	59545
	5月	5653419	1968033	66394	763876	75457
	6月	6961642	2455425	71603	959913	92621
	7月	8240089	2936562	78167	1140186	115890
	8月	9392221	3471806	84671	1300516	136715
	9月	10324002	4065544	94888	1461567	159016
	Total	233241417	84963405	2426619	31900615	3396393
	01-06月份	27455743	10828290	436094	3357657	342837
	07-12月份	66594724	27377107	915101	8046277	936381

图2 聚合结果的多维分层展现

## 2 结 论

显然,由于各个维度各种层次上成员的组合关系对应聚合结果都已经计算出来,配合以数据立方体的转换和报表展现(最简单的就是生成交叉表 Crosstab),可以很方便地由高级别成员下钻到低级别成员上查看聚合数据。另外,这里采用的聚合方法是通用的,不依赖于任何关系数据库系统本身提供的功能接口,只要是多维数据环境都能采用这种方法进行处理。

采用这种聚合方式也有一定的局限性,最关键的是汇总操作各个步骤中要大量使用递归算法反复扫描源数据视图、维度成员组合矩阵甚至是聚合结果集合本身,所以对系统资源的消耗是非常可观的。在以后的研究中,将进一步改进该聚合算法,比如提高其执行效率,减少资源消耗,加强出错处理,以及更好地支持多线程等。

### 参考文献:

- [1] 王文博,赵昌昌.统计学-经济社会统计[M].西安:西安交通大学出版社,2005.
- [2] 袁卫,吴喜之,贾俊平.描述统计学[M].北京:中国统计出版社,1996.
- [3] 康晓东.基于数据仓库的数据挖掘技术[M].北京:机械工业出版社,2004.
- [4] Ponniah P.数据仓库基础[M].段云峰等译.北京:电子工业出版社,2004.
- [5] Inmon W H.数据仓库[M].王志海,林友芳,等译.北京:机械工业出版社,2003.
- [6] 姚家奕.多维数据分析原理与应用[M].北京:清华大学出版社,2004.
- [7] 蒋旭东,冯建华,周立柱.联机分析查询处理中的一种聚集算法[J].软件学报,2002,13(1):65-70.
- [8] Graefe G. Query evaluation techniques for large database[J]. ACM Computing Surveys,1993,25(2):70-130.