# 一种改进的时变维度版本控制方法①

## An Improved Version Control Method OF Time Variance Dimension

刘黎志　（武汉工程大学计算机学院　武汉 430073）

**摘要**：本文针对一般时变维度版本控制方法的不足，提出了一种改进的时变维度版本控制方法。改进后的时变维度版本控制方法可以利用星型模型方便的形成当前多维数据集和历史多维数据集，从而提高对当前数据的查询效率及正确反映维度记录的历史数据。

**关键词**：时变维度　版本控制　数据仓库　星型模型

数据仓库是一个面向主题的、集成的、非易失的且随时间变化的数据集合，用来支持管理人员的决策[1]。数据仓库中的数据随时间变化一般表现为内容变化和维度变化。内容变化指数据仓库中的数据的自然累积过程，数据仓库每进行一次 ETL 过程，数据仓库的内容就变化一次。维度变化指型数据仓库多维数据

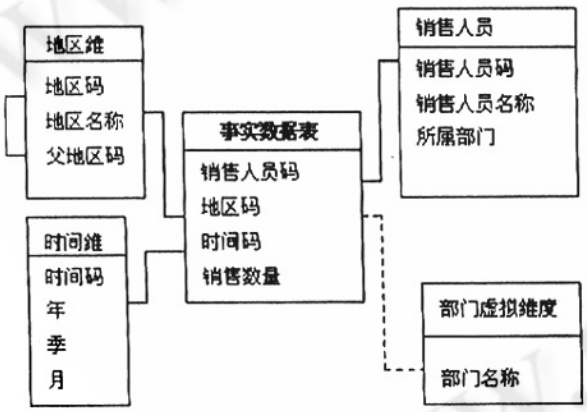


图 1　星型多维数据模型

模型中的维度表中的记录属性随时间的自然变化，这种变化不如内容变化频繁，也称缓慢时变维度[2]。时变维度往往会引起数据仓库中数据在时间上的不一致，这种时间上的不一致导致数据聚集计算错误，错误的聚集数据导致错误的决策，因此如何有效控制时变维度是个值得研究的课题。

## 1　问题提出

### 1.1　星型多维数据模型

由于关系理论和关系数据库的成熟，星型模型成为构建数据仓库多维数据模型中较流行的方法。星型模型的核心思想是：在关系模型的基础上，通过维度表和事实数据表之间的链接模拟多维数据模型。事实数据表居于整个星型模型的中心，它描述了决策分析所需的主题数据，所以包含的数据量大，而且随时间的推移不断增加。维度表分布在事实数据表周围，是观察事实数据的特定角度，一个表示企业销售情况的星型多维数据模型如图 1 所示。

地区维表中的记录为自引用结构，构成具有父子层次结构的维度模型。时间维是典型的星型层次架构，本文为简化起见，将时间的最小粒度设定为月。部门维度是根据销售人员所属部门构造的虚拟维度。

### 1.2　时变维度引起的问题

为描述时变维度引起的数据不一致问题，设定销售人员维度表数据如表 1 所示，2004 年事实数据如表 2 所示，2004 年按部门汇总销售情况的数据如表 3 所示。

表 1　销售人员表

| 人员码 | 名称 | 部门 |
|---|---|---|
| E1 | 张三 | A |
| E2 | 李四 | B |
| E3 | 王五 | C |

① 科学技术部科技型中小企业创新基金项目 项目代码 03C26214201044

表 2 2004 年事实数据表

| 地区码 | 时间码 | 人员码 | 销售数量 |
|---|---|---|---|
| 420114 | 2 | E1 | 40 |
| 420115 | 4 | E1 | 80 |
| 420114 | 2 | E2 | 85 |
| 420115 | 4 | E2 | 25 |
| 420115 | 7 | E3 | 10 |
| 420281 | 6 | E3 | 15 |

表 3 2004 看销售汇总表

| | A | B | C | 总和 |
|---|---|---|---|---|
| ID | 2004 | 2004 | 2004 | |
| E1 | 120 | | | 120 |
| E2 | | 110 | | 110 |
| E3 | | | 25 | 25 |

现假设 2005 年张三由 A 部门调到 B 部门，李四由 B 部门调到 A 部门，即维表记录的自然属性发生变化，若直接对销售人员维度表作更新操作，则销售人员维度表数据为表 4 所示，2004、2005 年事实数据为表 5 所示，重构多维数据集得到的 2004、2005 年销售情况汇总数据如表 6 所示：

表 4 销售人员表

| 人员码 | 名称 | 部门 |
|---|---|---|
| E1 | 张三 | A |
| E2 | 李四 | B |
| E3 | 王五 | C |

对比表三和表六发现，张三 2004 年在 A 部门的销售数量 120 在更新维度表后，被计入了张三 2004 年在 B 部门的销售数量，事实上张三 2004 年根本就不在 B 部门而在 A 部门，同样的情况也发生在李四身上。维度记录属性随时间自然变化的情况很多，如年龄、收入、婚姻情况等，若对这些自然变化不加控制，则数据仓库中的聚集结果就是错误或不可理解的，并由此可导致决策的错误。

## 2 时变维度版本控制方法

### 2.1 时变维度版本控制方法

为解决时变维度带来的问题，文[3－5]提出了几种解决方法，其中较为广泛应用的是维度版本控制方

表 5 2004、2005 年事实数据表

| 地区 | 时间码 | 人员码 | 销售数量 |
|---|---|---|---|
| 420114 | 2 | E1 | 40 |
| 420115 | 4 | E1 | 80 |
| 420114 | 2 | E2 | 85 |
| 420115 | 4 | E2 | 25 |
| 420115 | 7 | E3 | 10 |
| 420281 | 6 | E3 | 15 |
| 420214 | 15 | E1 | 100 |
| 420216 | 16 | E1 | 120 |
| 420201 | 20 | E2 | 80 |
| 420208 | 21 | E2 | 100 |
| 420281 | 19 | E3 | 150 |

表 6 2004、2005 年销售汇总表

| | A | | B | | C | | 总和 |
|---|---|---|---|---|---|---|---|
| ID | 2004 | 2005 | 2004 | 2005 | 2004 | 2005 | |
| E1 | | | 120 | 220 | | | 340 |
| E2 | 110 | 180 | | | | | 290 |
| E3 | | | | | 25 | 150 | 175 |

法。维度版本控制方法首先对需要进行版本控制的维表进行改造，一般是增加三个字段[NewId，StartDate，EndDate]，其中 NewId 字段取代原维表中的 ID 字段成为维表的主码。NewId 一般设置为标识种子，其值由系统自动生成。StartDate 表示维表记录的开始时间，EndtDate 表示维表记录的结束时间，EndDate 为空表示该维表记录处于当前(有效)状态。若维表记录自然属性发生变化，则执行以下算法

```
If(Exist(@ID)) //ID 为维表的原始 ID，以参数的形式传入{
    Select MaxNewId = Max(NewId) from t where t.ID = @ID; //t 表示维度表
    Set EndDate = Now where t.NewId = MaxNewId; //更新维表记录的最新版本，使之失效
    Insert t values(@ID, Properties, Now, Null) //插入新记录，NewId 由系统自动生成，表示最新版本
}
```

使用该版本控制算法后，第一节中的假设 2005 年张三由 A 部门调到 B 部门，李四由 B 部门调到 A 部门，产生的销售人员维度表如表 7 所示：

表 7　销售人员表

| 人员码 | 人员原始码 | 名称 | 部门 | 开始时间 | 结束时间 |
|---|---|---|---|---|---|
| 1 | E1 | 张三 | A | 20040101 | 20050101 |
| 2 | E2 | 李四 | B | 20040101 | 20050101 |
| 3 | E3 | 王五 | C | 20040101 | Null |
| 4 | E1 | 张三 | B | 20050101 | Null |
| 5 | E2 | 李四 | A | 20050101 | Null |

### 2.2 存在问题

可以发现利用新的销售人员维度表进行 2004、2005 年销售情况汇总查询所得的结果是正确的。但该维度版本控制方法存在以下问题：

(1) 若要查询销售人员当前销售情况，则必须首先对维度表进行查询后，才能确定记录的最新版本集合，然后再将记录的最新版本集合与事实数据表进行关联查询，才能得到结果。随着数据仓库技术的广泛应用，对数据仓库的实时性能要求越来越高，特别是对当前数据的查询频率要大大高于对历史数据的查询频率。显然随着时间的推移，由维度版本控制方法所产生的维度表会越来越大，确定最新记录集合所需的时间会影响到查询销售人员当前销售情况的效率。

(2) 事实数据表中无明显的记录历史关系，在事实数据表中，很难确定人员码 1 和 4 表示的是同一个人在不同部门、不同历史时期的销售数量。要查询某个销售人员的所有销售历史记录，同样要关联到销售人员维度表中进行查询。

(3) StartDate 与 EndDate 字段设置值得商榷。StartDate 与 EndDate 字段的存在一是为了确定记录的最新版本，二是查询销售人员在某个时间段的销售情况。若更好的办法确定记录的最新版本及通过时间维对销售人员在某个时间段的销售情况进行查询，则可省去对 StartDate 和 EndDate 字段设置。

## 3　一种改进的时变维度版本控制方法

本文针对一般时变维度版本控制方法的不足，提出了一种改进的时变维度版本控制方法。首先对一般时变维度版本控制方法中的维表改造过程作如下改进，只增加一个字段[HisId]，表示维表记录更新的历史情况，原维表中的 ID 字段不动，并将 HisId 设为标识种子。若维表记录自然属性发生变化，则执行以下算法：

```
If( Exist( @ ID) ) //ID 以参数的形式传入
{
    Set t. ID = 0 where t. ID = ID; //更新维表记录的最新版本，使之失效，t 表示维度表
    Insert t values( @ ID, Properties) //HisId 由系统自动生成，新记录表示最新版本
}
```

使用改进的时变维度版本控制算法后，第一节中的假设 2005 年张三由 A 部门调到 B 部门，李四由 B 部门调到 A 部门，产生的销售人员维度表如表 8 所示。

表 8　销售人员表

| 人员码 | 历史码 | 名称 | 部门 |
|---|---|---|---|
| 0 | 1 | 张三 | A |
| 0 | 2 | 李四 | B |
| E3 | 3 | 王五 | C |
| E1 | 4 | 张三 | B |
| E2 | 5 | 李四 | A |

观察表 8 可以发现，销售人员码不为 0 的记录就是最新版本，从而省去了对 StartDate 及 EndDate 的设置。销售人员码与销售人员历史码之间存在一对多的关系。利用这个关系对事实数据表及星型多维数据模型进行改进，就可以方便的对销售人员的当前销售情况进行高效率的查询及建立明确的记录历史关系以方便对销售人员的历史销售情况进行查询。为对销售人员的当前销售情况进行高效率的查询，对星型多维数据模型进行改进，如图 2 所示。

根据图 2 所产生的事实数据表如表 9 所示。结合图 2、表 8、表 9 可以发现，确定维度表中的最新版本的记录集合不需要首先对维度表进行查询，因为所有销售人员码不为 0 记录就是最新版本的记录集合，又因为事实数据表中不存在销售人员码为 0 的记录，所以直接关联销售人员维度表与事实数据表的销售人员

码、销售人员历史码就可以得到销售人员的当前销售情况。考虑对当前销售情况查询的效率，可以进一步控制多维数据模型中的时间维，可以根据实际情况缩小时间维中的时间的表示范围。例如可以控制图 2 中的时间维只表示 2005 年的时间范围，从而使得频繁查询当前销售情况的效率大大提高，并且若由于某种原因不能对数据仓库进行增量更新，重构整个多维数据集的时间代价也不大。

表 9 2004、2005 年事实数据表

| 地区 | 时间码 | 人员码 | 历史码 | 销售数量 |
|---|---|---|---|---|
| 420114 | 2 | E1 | 1 | 40 |
| 420115 | 4 | E1 | 1 | 80 |
| 420114 | 2 | E2 | 2 | 85 |
| 420115 | 4 | E2 | 2 | 25 |
| 420115 | 7 | E3 | 3 | 10 |
| 420281 | 6 | E3 | 3 | 15 |
| 420214 | 15 | E1 | 4 | 100 |
| 420216 | 16 | E1 | 4 | 120 |
| 420201 | 20 | E2 | 5 | 80 |
| 420208 | 21 | E2 | 5 | 100 |
| 420281 | 19 | E3 | 3 | 150 |

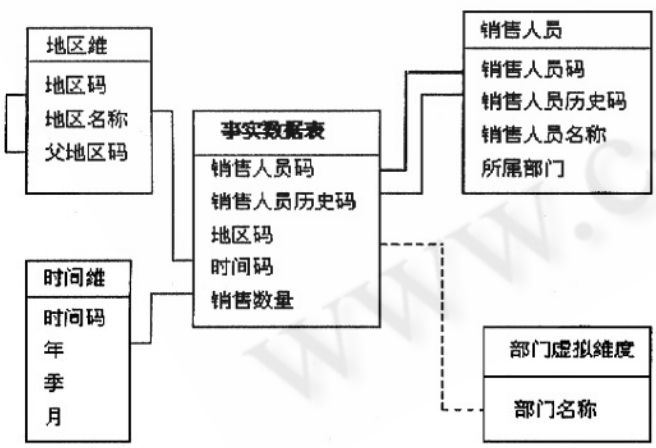


图 2 星型多维数据模型

观察表 9 可以看到销售人员 E1 对应两个历史版本 1、4，即在事实数据表中建立了明确的历史关系，直接对事实数据表进行查询就可以得到 E1 的历史销售情况。或者修改图二所示星型模型，去掉关联销售人员维度表与事实数据表的销售人员码（图中的粗线），即只关联销售人员维度表与事实数据表的销售人员历史码所得到的新的多维数据集就是整个销售情况的历史多维数据集。

## 4 总结

维表中记录的属性随时间变化在数据仓库的应用过程中是一个必须考虑的重要问题，时变维度版本控制方法利用增加冗余的维表记录形成不同历史版本，从而保证了数据仓库中的数据在时间上的一致性，解决了时变维度带来的问题。本文对一般时变维度版本控制方法进行了改进，利用关联事实数据表与维表的主码和历史码形成当前多维数据集，提高了频繁查询当前数据的效率；利用关联事实数据表与维表的历史码形成历史多维数据集，正确的反映了时变维度记录的历史数据。改进的时变维度版本控制方法已经在实践中得到应用，并取得了良好的效果。

## 参考文献

1 Inmon, W. H.: Building the Data Warehouse[M]; Jon Wiley &Sons, Inc.; New York, Chichester, Brisbane, Toronto, Singapore; Second Edition, 1996.

2 Donald J. Berndt and John W. Fisher, Understanding Dimension Volatility in Data Warehouses [C]. Sixth INFORMS Conference on Information Systems and Technology, 2001.

3 徐骥、陶树平，基于星型模型的数据仓库中维变技术的研究[J]，计算机工程，2002，28(4)：91-93.

4 Shahzad, M. K., J. A. Nasir, CEV-DW: Creation and Evolution of Versions in Data Warehouse [J]. Asian Journal of Information Technology 4 (10): 910-917, 2005.

5 Pasha, M. A. J. A. Nasir and M. K. Shahzad, SVF: Schema Versioning Framework for data warehouse[C]. Proceedings of ITAC, Pakistan, 2005.