Performing OLAP Operations in Data Cube with Dimensions Having Irregular Hierarchical Relationships

Anna G. Rozeva[1]

Abstract - **OLAP provides for analysis of large collections of historical factual data aggregated by the levels of hierarchically structured dimensions. In case that a hierarchy is irregular aggregation results will be incorrect. Algorithms for rendering irregular hierarchies summarizable thus ensuring correctness of aggregations are proposed. Mechanism for performing basic OLAP operations in relational environment is designed.**

Keywords – **OLAP, Data cube, Summarizability, Hierarchy transformation, OLAP operations**

I. INTRODUCTION

On-Line Analytical Processing (OLAP) systems aim to ease the process of extracting useful information from large amounts of detailed transactional data. The modeling approach that fits most naturally to data analysis problems is the multidimensional one. It provides for fast, consistent, interactive access to a wide variety of possible views of information, automatic application of pre-specified aggregation functions, visual querying and good query performance due to the use of pre-aggregation [6]. Multidimensional view is captured in several data models referred to as cube models or data cubes [3], [8]. Data is examined as n-dimensional cube. It's divided into measures (facts) on which calculations are performed and dimensions that characterize them. Each dimension has a number of attributes that can be used for selection and grouping. Typical OLAP queries involve calculating aggregations from large amounts of measure data. Aggregations are performed on dimensions that are organized along hierarchy levels. Cube models differ depending on whether relationships between hierarchy levels are captured explicitly or not by the scheme. A hierarchy that is explicitly captured in the scheme provides better guidance for navigating the cube.

Quick access to aggregated measures is provided by pre-computing and storing them. Several types of pre-aggregation can be implemented that differ in the query response time. Fastest response time is provided by pre-computing and storing aggregations for all possible combinations of dimensions' values. The storage requirements in this case grow rapidly resulting in data explosion.

[1]Anna G. Rozeva is with the University of Forestry, Faculty of Business Management, Kliment Ohridski 10, 1756 Sofia, Bulgaria, E-mail: arozeva@ltu.bg

It occurs because the number of possible aggregation combinations increases fast with the increase of the number of dimensions. The sparseness of the multidimensional space decreases in higher dimension levels. Thus the space taken by aggregates at higher levels almost equals the one of lower levels. Another approach deals with selecting a subset of aggregation levels to pre-compute and obtaining aggregates for the upper levels on the fly from the pre-computed ones. The technique is referred to as summarizability [4]. The hierarchy structure should provide for correctness of aggregations obtained along dimension hierarchy levels. One aspect of incorrect aggregation is double-counting of data. It occurs in case of many-to-many relationship between fact and dimensions or dimension members being part of more than one higher level. Another aspect of incorrect aggregation consists in not counting data. This occurs if measure value is not present at the lowest hierarchy level or when the fact-to-dimension mapping has varying granularity. These problems occur with non-summarizable dimension hierarchies. Hierarchy properties and transformation techniques for achieving summarizability are discussed in [1], [2], [7].

Our work concerns design and implementation of mechanism for obtaining correct aggregations in a relational OLAP environment since relational technology turns out to be the most commonly used platform for OLAP applications and the major relational data base management systems (RDBMS) use pre-aggregated data for enhancing query response times. The aspects discussed are:

- Summarizability violations;
- Implementation of summarizability in the data cube scheme;
- Performance of OLAP operations on summarizable hierarchies.

Further on in the paper irregularities in hierarchies and procedures for obtaining summarizability are examined. Technique for pre-computing aggregations along hierarchies with ensured summarizability by means of standard relational technology is designed. Finally procedures for implementing summarizable pre-aggregations in OLAP operations are presented.

II. CASE STUDY WITH SUMMARIZABILITY VIOLATIONS

Summarizability is an important cube property. It states when lower-level aggregates being pre-computed can be used to calculate higher-level aggregates as well as when they are to be computed from source base data. An aggregate function is summarizable if aggregated results from lower level aggregates when combined give the same result as when the aggregate is derived directly from base data. As calculation of aggregates from base data increases significantly the computation cost it's important to ensure summarizability

along cube dimensions. Summarizability is achieved by proper ordering of dimension values in hierarchy levels. The ordering is referred to as strict and covering [4], [8]. A dimension hierarchy is strict if no member has more than one parent from the same level. It's covering if no path skips a level. This implies that dimension hierarchies should be balanced trees. If this isn't fulfilled some lower level values will be either double counted or not counted at all.

A case study concerning wood retail business is examined. The data cube is shown by diagram with UML notation in Fig.1.

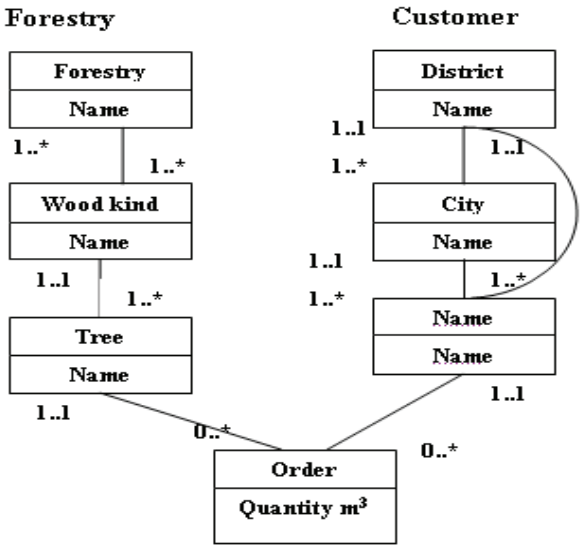

Fig.1. Wood retail business – UML diagram

For the purposes of investigation concerning hierarchy structure and summarizability the examined fact Orders is characterized by only two dimensions, i.e. Forestry and Customer and single measure Quantity. Forestry states the tree name, the wood kind (conifer / broad-leaved) and the place it was cut. The following summarizability violations are present:

1. Not-counting Orders

♦ Customer dimension - Orders of Customers whose Addresses aren't in a City when aggregating at City level;

♦ Forestry dimension – Orders made for Wood kind when aggregating at Tree level;

2. Double-counting Orders – Forestry dimension when aggregating at Wood kind level as Wood may be cut from several Forestry.

III. Managing Summarizability Violations

Our previous work on hierarchy irregularities, transformations and hierarchy normalization is presented in [1] and [2]. What is treated here concerns summarizability violations in the data cube scheme. The approach implemented deals with designing metadata for dimension hierarchy structure that provides for summarizability. This is achieved by modifying the initial hierarchy for resolving non-counting and double-counting violations. Metadata is designed in relational environment. The level ordering of a dimension with parent-child relationships is defined by structure table – Table I. Parent and Child fields mean identifiers of members of the dimension table.

TABLE I
Dimensional hierarchy structure table

Parent	Child	Type	CLevelId

Top-level members have CLevelId = 0. Type column holds 1 for real Parent and 0 for artificially defined one. Metadata table may be generated by exploring the dimension tables as shown in [2] or may be explicitly defined by user. Handling non-counting violations implies eliminating paths that skip levels like the one in the Customer dimension and ensuring that all non-bottom level members have children. This is the case with the Forestry dimension when examining sawdust that is of no specific Tree. Algorithms for handling hierarchical irregularities are presented in [7]. Our approach corresponds to the structure of the metadata table with hierarchy levels explicitly stated. This facilitates hierarchy exploration for multiple paths and childless non-bottom level members. In the case study diagram members from the address level may skip the city level and connect to district directly, i.e. customer's address is not in a city. The algorithm for handling skipping level paths inserts artificial members at the skipped level and the metadata table is modified accordingly. It's shown in Fig.2.

```
Procedure SkippingPaths (C)
for each Parent = Child and Type = 1 do
begin
for each C
find Parent as P
find Child = P, CLevelId as PLevel
where PLevel < Clevel – 1 do
begin
        Insert Child, Child, 0, CLevelId
        Insert Parent, Child, 1, ClevelId-1
        Delete Parent, Child, 1, ClevelId
end
SkippingPaths (P)
end
end
```

Fig.2. Handling non-counting violation due to skipping paths

The algorithm explores dimension metadata table by levels. For each level member starting from bottom level it identifies its parents and gets their levels (the find statements). In case that a parent found is at level with number smaller than the one of the upper adjacent level modification of the metadata table is performed. Two rows are inserted – one for the child member being examined with parent having the same value and 0 in the Type field. This row holds the link of the child member to its upper adjacent level. The values for parent and child fields are the same for keeping the original values in the newly created links. Another row is inserted which holds the

link of the artificial parent to the child's real parent. Finally the row holding the skipping level path is deleted. The procedure is recursively called until hierarchy top level is reached. The metadata table structure proposed facilitates hierarchy transformation by eliminating join operations in case of hierarchy level relationships being kept in separate tables.

Algorithm is designed for managing summarizability violations concerning non-counting measures due to the presence of childless non bottom-level members. In the Forestry dimension of the case study wood kind may not have associated tree. The algorithm is shown in Fig.3.

```
Procedure Childless (P)
for CLevelId = 0 to Max(CLevelId) - 1 do
begin
for each P
find Child as C, LevelId
if not found do
        begin
        for CLevId = CLevelId to 1 do
                begin
                Insert Parent, Parent, 0, CLevelId – 1
                end
Childless (C)
end
end
```

Fig.3. Handling non-counting violation due to childless members

Starting from top level the algorithm looks for children of level members. In case that they are missing rows are inserted into metadata table with artificial children and parents respectively and 0 as type value until bottom level is reached. The rows inserted have the same values for parent and child thus making possible the mapping of fact to the inserted artificial level members. The algorithm is called recursively until reaching level that is adjacent to the bottom level.

Another summarizability violation concerns double-counting members. This occurs in case of members of the dimension hierarchy having multiple parents from the same level. In the case study presented this occurs in the Forestry dimension between Forestry and Wood kind levels as wood kind may originate from more than one Forestry. Algorithm for managing it is shown in Fig.4. It takes as input a metadata table MT and looks for parents of each level member starting from the bottom level. Parents of Child C are counted and those with count greater than 1 are selected. A table "Multiple parents" is created for C and the selected parents are inserted therein. Rows with artificial parents for C are inserted in MT until top level is reached. Finally the rows of C's multiple parents are deleted. The procedure is called recursively until members of all levels are checked. The algorithm is performed upon hierarchies with paths of the same length and links from children to their immediate parents. This can be achieved initially by applying the algorithms resolving non-counting violation.

```
Procedure DoubleCount (C, MT)
for each Parent = C  and Type = 1 do
begin
select  Parent as P, Count (P)
from MT Where Child = C
If ParentCount >1
begin
create Multiple Parent
(Field1 as Child, Field2 as Parent, Field3 as CLevelId)
append C, Parent, CLevelId
end
for Level = CLevelId to 0
insert C, C, 0, CLevelId – 1in MT
for Level = CLevelId to 0
delete *
from MT where Child = C
DoubleCount (P, MT)
end
end
```

Fig.4. Handling double-counting violation

As a result of the algorithms presented hierarchies are implemented in the data cube scheme through metadata tables holding level relationships with explicitly denoted real and artificial parents and multiple parent tables for children with multiple parents.

IV. Performing OLAP Operations Implementing Hierarchy Metadata

OLAP operations consist of 80% navigational queries that explore dimension hierarchies and 20% aggregation queries that summarize data at various levels of detail [5]. Implementation technique for OLAP queries on transformed dimension hierarchies is presented in [9]. Sample navigational query concerning our case study is one for showing Customers per District. Dimension Customer is represented in Table II:

TABLE II
DIMENSION TABLE

Id	MemberName	GroupName

Navigational query is internal to a dimension. Input tables for the query are dimension and hierarchy tables. Initially top and bottom levels for the navigation are determined. The Child members for Parent members at the top level are selected. Only real parent members are examined by the filter condition "<>0" on the Type field in the hierarchy table. The dynaset of Child members obtained is joined with the hierarchy table on the parent field. The query drills down to the next level by selecting child members for the parents resulting from the equijoin. The query is performed on the hierarchy table joined with the dimension table for outputting members' names. The Join/Select queries are performed until reaching the bottom level. The query flow is shown in Fig. 5.

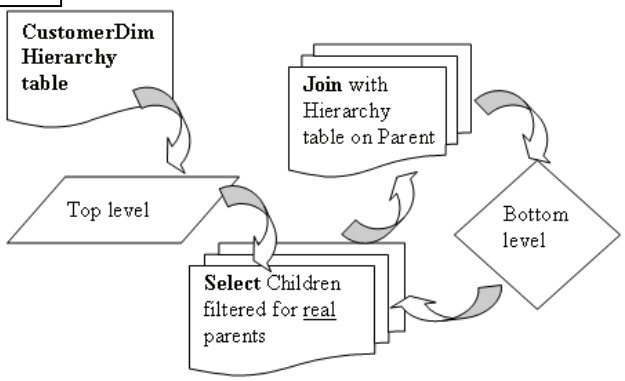

Fig.5. Navigational query – subquery flow

Aggregation queries involve facts and pre-aggregated measures. Sample aggregation query is one for counting Orders grouped by Wood kind or by District. Aggregation function used may be different.

Aggregation query for retrieving Orders count per District is examined. The query format is as follows:

SELECT Customer.District, **SUM** (Order.Count)
FROM Customer, Order
WHERE Customer.CustomerID = Order.CustomerId
GROUP BY Customer.District

By means of the algorithms designed in the previous section dimensional hierarchies are rendered summarizable. We assume that Order counts for the bottom level of the Customer dimension are pre-computed and stored in a separate table with structure shown in Table III.

TABLE III
PRE-AGGREGATION TABLE

Customer Id	OrderCount

The aggregation query will use the pre-aggregated measures for calculating Order counts for the higher hierarchy levels. Order counts per Customer are to be summarized for obtaining these per Cities and Districts respectively. The query flow in SQL statements is shown in Fig.6.

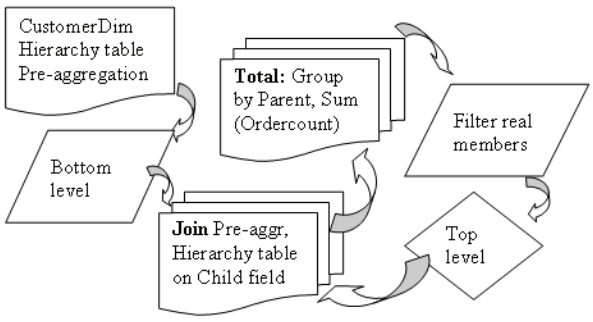

Fig.6. Aggregation query - subquery flow

Input tables for the query flow are Customer dimension and hierarchy tables and pre-aggregated measures for customers. Top and bottom levels for aggregations are determined. The pre-aggregation table for the bottom level is joined with the hierarchy table on Child field and Parent level members are obtained. Order counts are then summarized for the resultant parents. The dynaset obtained is further joined with the hierarchy table until reaching the top level. When outputting results real members are filtered. Pre-aggregation assumes that facts are related to the bottom level of dimensions and the relationship between facts and dimension members is many-to-one. In case of violation algorithms for making non-bottom members have children or for handling double-counting are initially performed.

V. CONCLUSION

The process of providing summarizability to dimension hierarchies in order to enhance the performance of OLAP operations by using pre-computed aggregates for calculating higher level ones with ensured correctness of results is investigated. Schema support for dimensional hierarchies in ROLAP environment is provided. Metadata for hierarchy structure is designed. Dimensional hierarchy structures that violate summarizability are explored. Algorithms for managing the most general violations as non-counting or double-counting measures in aggregations are proposed. Mechanism involving standard SQL query flow for implementing hierarchy metadata in OLAP navigation and aggregation operations is presented.

Future work is intended in investigating scheme support of dimension updates and mechanism for maintaining hierarchy metadata and pre-aggregations under dimension updates.

REFERENCES

[1] A. Rozeva, Dimensional Hierarchies – Implementation in Data Warehouse Logical Scheme Design, *Proceedings of the International Conference on Computer Systems and Technologies CompSysTech'07*, Rousse, Bulgaria, 2007

[2] A. Rozeva, Designing Navigational Framework in Data Warehouse Logical Scheme, *Proceedings of the International Conference Automatics and Informatics'07*, Sofia, Bulgaria, 2007

[3] D. Pedersen, K. Riis, T.B. Pedersen, A Powerful and SQL-Compatible Data Model for OLAP, *13th Australasian Database Conference ADC2002 Proceedings*, Melbourne, Australia, 2002

[4] H. Lenz, A. Shoshani, Summarizability in OLAP and Statistical Databases, *SSDBM Conference Proceedings*, pp. 39-48, 1997

[5] R. Kimball, *The Data Warehouse Toolkit*, Wiley Computer Publishing, 1996

[6] S. Youness, *Data Warehousing with SQL Server 7.0 and OLAP Services*,Wrox Press Ltd, 2000

[7] T. Pedersen, C. Jensen, C. Dyreson, Extending Practical Pre-Aggregation in On-Line Analytical Processing, *25th VLDB Conference Proceedings*, Edinburg, Scotland, 1999

[8] T. Pedersen, C. Jensen, Multidimensional Data Modeling for Complex Data, *ICDE'99 Conference Proceedings*, 1999

[9] T. Pedersen, C. Jensen, C. Dyreson, The TreeScape System: Reuse of Pre-Computed Aggregates over Irregular OLAP Hierarchies, *VLDB'00 Conference Proceedings*, pp. 595-598, 2000