Drill Across & Visualization of Cubes with Non-Conformed Dimensions

Dariush Riazati James A. Thom Xiuzhen Zhang
School of Computer Science and Information Technology
RMIT University, Melbourne, Australia, 3001

dariush.riazati@student.rmit.edu.au, james.thom@rmit.edu.au, xiuzhen.zhang@rmit.edu.au

Abstract

Data analysts would benefit greatly from the ability to navigate and view combined multidimensional data from multiple sources, a key requirement of which is the conformity between their dimensions. The strict requirements of conformity restrict navigating to related multidimensional data from unseen or unfamiliar sources.

In this paper we make a distinction between conformed dimension tables and conformed dimension attributes and discuss the merits of relaxing the conformity requirement. We propose extending the navigation operation *drill across* to include the non-conformed dimensions and introduce *Nested Pivot Tables* as an extension to Pivot Tables to show how sources that have conformed as well as non-conformed dimensions can be viewed and analyzed together.

Keywords: Multidimensional Data, Dimension Conformity, Drill Across, Loss Ratio.

1. Introduction

Data analysts would benefit greatly from the combined view of similar and related data from heterogeneous multidimensional databases.

A key requirement in navigating multidimensional data from several sources, according to Kimball and Ross (2000) is that they must have conformed dimension (tables). Their view of conformed dimensions which is widely adopted requires the conformed dimensions to be either from the same instance or identical in terms of schema and data. Navigating between sources with dimensions that have identical schema but fall short of having identical values can result in loss of data and consequently inaccurate aggregation results.

The non-conformity problem manifests itself more often when attempting to combine or integrate multidimensional data from unfamiliar heterogeneous sources. Inclusion of non-conformed dimensions in the navigation and visualization of multidimensional data from multiple sources provides the analyst with the ability to view and analyze data that would be otherwise lost due to the exclusion of non-conformed dimensions and helps discover possible relationships between seemingly non-conformed dimensions.

We make three contributions in this paper: 1) with the presence of non-conformed dimension tables, we identify conformity within the dimension attributes, and describe methods to measure the loss resulting from the join between conformed dimension attributes with dissimilar values; 2) we extend the definition of the navigation operation *drill across* to include (selective) non-conformed dimension attributes*;* 3) we introduce *Nested Pivot Tables* as an extension to Pivot Tables to show the result of our extended *drill across*.

The structure of this paper is as follows. Section 2 describes our motivation for navigating and visualizing of data cubes with non-conformed dimensions. Section 3 introduces key concepts used in this paper. Section 4 discusses ways to measure *loss ratio*; Section 5 describes our extended operation *drill across*; Section 6 describes *Nested Pivot Tables*; Section 7 discusses related work on navigation and data visualization methods, and finally Section 8 presents the conclusion and our future work.

2. Motivation

As Internet technology and protocols around web services improve, easier and more secure access to heterogeneous data sources increase the potential for data analysts to access multidimensional data from unfamiliar or unseen sources.

An insurance venture has taken over several competing companies offering Comprehensive (COMP), Third Party (TP), Caravan (CAR) and Motorcycle (MOT) insurance policies. The companies identify the geographic location of where the insurance claims have originated from, based on their operational units or postcodes.

The following scenarios use the schema in Figure 1 where the two dimension tables `Period` and `Product` are conformed, but the two dimension tables `Operational_Unit` and `Location` are not.

Navigation between sources with non-conformed dimension attributes: In this scenario we consider the two dimension attributes `Location.postcode` and `Operational_Unit.unit` as non-conformed. Although there is no given concrete relationship between the dimension attributes `Location.postcode` and `Operational_Unit.unit`, the analyst wants to be able to view and compare the values for these two attributes as well as their related number of claims because s/he suspects that some of the postcodes and units concern the same or similar geographic location.

This requirement can be met only if the navigation between the two sources returns both the aggregated data for the conformed attributes `Period.year` and `Product.product_id` as well as for the non-conformed dimension attributes `Operational_Unit.unit` and `Location.postcode`. Furthermore, the analyst must be able to view and analyze all results together without having to leave the scene.

Next we describe why it is not possible to achieve these objectives using the existing navigation operation and pivot tables. We use Table 1 as a sample set of data for the schema in Figure 1 to illustrate our examples throughout this paper.

Claim (*C1*)

year	product_id	unit	no_of_claims
2003	COMP	3054	3,175
2003	COMP	3200	2,180
2003	TP	2010	1,145
2003	TP	2100	1,165
2003	CAR	4000	115
2004	COMP	3054	2,990
2004	COMP	3200	3,195
2004	TP	2100	1,178

Claim_Header (*C2*)

year	product_id	post code	total_claims
2003	COMP	3054	4,210
2003	COMP	3200	3,312
2003	TP	2010	1,743
2003	TP	2011	1,512
2004	TP	2012	1,127
2004	TP	2014	1,168
2004	TP	2016	1,871
2005	COMP	3054	4,817
2005	COMP	3200	3,125
2005	TP	2012	2,154

Table 1: An instance of the schema in Figure 1

Using the existing navigation operation (*drill across* which excludes the non-conformed dimension attributes from the navigation) we would require three separate pivot tables:

1. to show the result of the navigation between `Claim` and `Claim_Header` using the conformed dimension attributes `Period.year` and `Product.product_id`,
2. to show all data in `Claim`, and
3. to show all data in `Claim_Header`.

In order to perform any comparison, the analyst would then have to *zoom* into one or more groups of data (e.g. where `year=2004` and `product_id='TP'`) in each pivot table resulting in pivot tables shown in Tables 2a, 2b and 2c below:

year	product_id	no_of_claims	total_no_of_claims
2004	TP	1,178	4,166
	Total	1,178	4,166
Total		1,178	4,166

Table 2a: Pivot table 1 restricted to `year=2004` and `product_id='TP'`

year	product_id	unit	no_of_claims
2004	TP	2100	1178
		Total	1,178
	Total		1,178
Total			1,178

Table 2b: Pivot table 2 restricted to `year=2004` and `product_id='TP'`

year	product_id	postcode	no_of_claims
2004	TP	2012	1,127
		2014	1,168
		2016	1,871
		Total	4,166
	Total		4,166
Total			4,166

Table 2c: Pivot table 3 restricted to `year=2004` and `product_id='TP'`

This approach is prone to human errors, cumbersome and time-consuming because:

1. There is no combined visualization of the three pivot tables; the analyst would have to constantly toggle between three independent pivot tables.
2. The analyst would have to repeat the *zoom* (or *restrict*) operation in pivot tables 1, 2 and 3 for different groups of data.
3. There is no overall view of all groups of data in pivot table 1 after it is restricted (as shown in Table 2a).
4. Any pivoting function (such as *roll-in*) applied to pivot 1 must be manually applied to pivot tables 2 and 3.
5. There is no potential for (a semi-automated) mapping of data between `unit` and `postcode` as pivot tables 2 and 3 are independent of one another.

To overcome these problems we propose:

- in Section 5 extending the navigation operation *drill across* to return the aggregated data for the conformed as well as non-conformed dimension attributes, and
- in Section 6 extending the pivot tables to show the aggregated data for the conformed as well as the non-conformed dimension attributes.

Navigating between sources with conformed but intersecting dimension attributes: The analyst considers that although `Location` and `Operational_Unit` are not conformed, the two dimension attributes `Location.postcode` and `Operational_Unit.unit` refer to specific geographic locations within Australia but for which there may not be a full match between them. The analyst considers the two attributes as conformed and would like to compare the number of claims for matching units and postcodes or only for those units that the insurer operates in. At the same time s/he would like to know how (dis)similar they are and whether it is possible to eliminate their dissimilarity by changing the analysis space. The analyst's

requirements can be met if we identify conformity at the attribute level and treat these two attributes as conformed even though they have different names and have dissimilar values.

Allowing attributes with intersecting values to be considered as conformed provides greater flexibility in navigating between two sources but leads to loss of data and incorrect aggregations resulting from the join between two such attributes. It is therefore necessary to be able to quantify the loss to help the analyst to make one of the following decisions:

- The loss can be tolerated because the analyst is interested in one or more of the intersecting values.
- The loss can be eliminated (or reduced) by changing the analysis space through constraining the values for other dimension attributes.
- The loss cannot be tolerated and hence the pair of attributes is treated as non-conformed.

We propose *loss ratio* as a measure to quantify the (dis)similarity between two dimension attributes. In Section 4 we describe two methods to measure the *loss ratio* and show it when visualizing navigation between sources that have intersecting dimension attributes. Given Table 1, calculation of dissimilarity between `unit` and `postcode` enables the analyst to explore the options listed above.

3. Concepts and Terminology

This section describes some of the key concepts that this paper refers to.

Star Schema Model according to Kimball and Ross (2000) is a generic representation of a dimensional model in a relational database. This model according to Giovinazzo (2003) as well as Kimball and Ross (2000) consists of a *fact table* in the center with foreign key constraints to one or more *dimension tables* around it (Giovinazzo 2003, Kimball and Ross 2000).

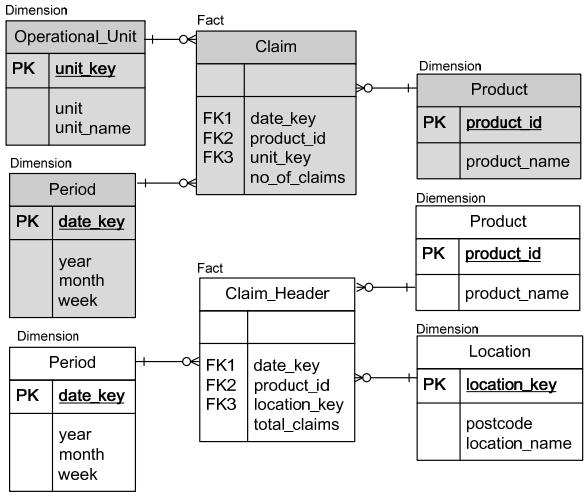

Figure 1: Star Schema

Dimension Tables according to Giovinazzo (2003) can be described as access points to the measures in the fact tables. For example the `total_claims` is determined by `date_key`, `product_id` and `location_key`. Tuples within dimension tables are uniquely identified using a primary key or a natural key. For simplicity but without loss of generality we only consider primary keys with a single attribute. Elements within dimension tables are known as *dimension attributes* which may belong to different levels of the hierarchy within the dimension table. For example the dimension attribute `year` is at a higher level of the hierarchy than `month` is.

Fact Tables according to Kimball and Ross (2000) are in the center of the Star Schema model surrounded by dimension tables and contain a set of mainly numeric elements called *measures*. `Claim` and `Claim_Header` tables in Figure 1 are examples of fact tables.

OLAP and OLAP Operations: Kimball and Ross (2000) define Online Analytical Processing (OLAP) as a set of principles that provide a dimensional framework for decision support. According to Chaudhuri and Dayal (1997) OLAP operations enable analysts to dissect multidimensional data through operations such as: *Drill-Up* (or *Roll-Up*) to group data, *Drill-Down* (or *Roll-In*) to ungroup data, *Pivoting* to re-orient a data cube around its dimensions, and *Dice & Slice* to change the analysis space by projecting the data over selected values of dimensions. This paper is concerned with relational implementation of OLAP (ROLAP).

Data Cube is defined by Gray et al. (1997) as representing aggregations of data for all possible levels of granularity. Each node in the *data cube* is called a *cuboid* and contains the same data aggregated using a distinct combination of dimension attributes. A data cube with n dimension attributes has 2^n cuboids.

Data Mart is defined by Kimball and Ross (2000) as a logical and physical subset of the data warehouse. A data mart based on a single Star schema can be visualized as a data cube.

Drill Across is an OLAP operation used to navigate from an origin data cube to a destination data cube using the same coordinates. In the case of ROLAP, *drill across* is implemented using natural join between the conformed dimension tables.

4. Conformed Dimension Attributes and Loss Ratio

Conformed Dimension Attributes: Where dimension tables are not conformed, we identify *conformed dimension attributes*. We consider two dimension attributes conformed if their domains are considered to refer the same thing but their values could be different, i.e. $d : dom(d)$ is conformed with $d' : dom(d')$ if $dom(d_i) = dom(d'_j)$.

For example `make` and `model` both could determine the type of a car and could be considered conformed, but a pair of dimension attributes both named `color` with identical values could not be conformed if one referred to the exterior color and the other referred to the interior color. We consider that name difference between conformed dimension attributes can be resolved during the integration process. Next, we describe the *loss ratio*

as a measure of loss resulting from dissimilarity of values between conformed dimension attributes.

Loss Ratio: Let us consider two fact tables F and F'. F is joined to $s+p+q$ dimension tables, s of the dimension tables are shared with F'. Of the $p+q$ dimension tables in F, p of the dimension tables are not conformed but have at least one conformed attribute with a corresponding dimension table joined to F' (for simplicity but without loss of generality we assume each of these p dimensions only has a single conformed attribute) and q dimensions are exclusive to F, and a further r *dimensions* are *exclusive to* F'. The exclusive tables have no conformed dimension attributes. Attributes on the s shared dimension tables are conformed and will not result in any loss.

We can represent the conformed dimensions tables as:

$$D_1(c_{1,1}, c_{1,2}, ...)$$
$$...$$
$$D_s(c_{s,1}, c_{s,2}, ...)$$

and the p dimensions that have at least one conformed attribute as:

$$D_{s+1}(c_{s+1}, a_{s+1,1}, .a_{s+1,2}, ...)$$
$$...$$
$$D_{s+p}(c_{s+p}, ..., a_{s+p,1}, .a_{s+p,2}, ...)$$

And the q dimensions joining to F that have no conformed attribute as:

$$D_{s+p+1}(a_{s+p+1,1}, a_{s+p+1,2}, ...)$$
$$...$$
$$D_q(a_{s+p+q,1}, a_{s+p+q,2}, ...)$$

Similarly for F' we have:

$$D'_{s+1}(c_{s+1}, a_{s+1,1}, a_{s+1,2}, ...)$$
$$...$$
$$D'_{s+p}(c_{s+p}, a_{s+p,1}, a_{s+p,2}, ...)$$
$$D_{s+p+1}(a_{s+p+1,1}, a_{s+p+1,2}, ...)$$
$$...$$
$$D_r(a_{s+p+r,1}, a_{s+p+r,2}, ...)$$

Based on our schema in Figure 1, the two fact tables `Claim` and `Claim_Header` are joined to two conformed dimension tables `Period` and `Product`. They also have a pair of dimension tables `Operational_Unit` and `Location` with one conformed dimension attribute `Operational_Unit.unit` and `Location.postcode`.

D_1, D'_1 = `Period(date_key,year,month)`
D_2, D'_2 = `Product(product_id,product_name)`
D_3 = `Operational_Unit(unit_key,unit)`
D'_3 = `Location(location_key,postcode)`

For simplification we have summarized `Claim` and `Claim_Header` over a subset of their dimension attributes:

F = `Claim(year,product_id,unit, no_of_claims)`

F' = `Claim_Header(year, product_id, postcode, total_claims)`

In our examples below we measure the loss resulting from the join between two conformed attributes `Operational_Unit.unit` and `Location.postcode` using the following dimension values and the sample data in Table 1 from Section 2.

```
Period.year={'2003','2004','2005'}
Product.product_id={'TP','COMP','CAR','MOT'}
Operational_Unit.unit={'2010','2020','2050',
'2100','3054','3200','4000'}
Location.postcode={'2010','2011','2012','2014',
'2016','3054','3100','3192','3200'}
```

Next, we describe three methods to calculate the *loss ratio*.

Absolute loss ratio: This is calculated for every pair of conformed dimension attributes and measures the degree of (dis)similarity between their values. We calculate the *absolute loss ratio* by dividing the cardinality of the values of the paired conformed dimension attributes over the cardinality of the values of the conformed attribute for which we calculate the loss. The *absolute loss* ratio for conformed dimension attributes joining to F is:

$$1-\left(\frac{\left|\pi_{c_{s+1}}(D_{s+1}) \cap \pi_{c_{s+1}}(D'_{s+1})\right|}{\left|\pi_{c_{s+1}}(D_{s+1})\right|}\right)$$
$$...$$
$$1-\left(\frac{\left|\pi_{c_{s+p}}(D_{s+p}) \cap \pi_{c_{s+p}}(D'_{s+p})\right|}{\left|\pi_{c_{s+p}}(D_{s+p})\right|}\right)$$

The *absolute loss ratio* for conformed dimension attributes joining to F' is:

$$1-\left(\frac{\left|\pi_{c_{s+1}}(D_{s+1}) \cap \pi_{c_{s+1}}(D'_{s+1})\right|}{\left|\pi_{c_{s+1}}(D'_{s+1})\right|}\right)$$
$$...$$
$$1-\left(\frac{\left|\pi_{c_{s+p}}(D_{s+p}) \cap \pi_{c_{s+p}}(D'_{s+p})\right|}{\left|\pi_{c_{s+p}}(D'_{s+p})\right|}\right)$$

The *absolute loss ratio* resulting from the join between `Operational_Unit.unit` and `Location.postcode` for F is 57% (1–3/7) and for F' is 67% (1–3/9).

Relative Loss Ratio: Not all of the attribute values point to a tuple in the fact table it relates to. A more accurate method of calculating the *loss ratio* is to calculate the *relative loss ratio* in respect to the participation of the values of the dimension attributes in the fact tables. The *relative loss* ratio for F:

$$1-\left(\frac{\left|\pi_{c_{s+1}}(D_{s+1} \triangleright\!\triangleleft F) \cap \pi_{c_{s+1}}(D'_{s+1} \triangleright\!\triangleleft F')\right|}{\left|\pi_{c_{s+1}}(D_{s+1} \triangleright\!\triangleleft F)\right|}\right)$$

...

$$1-\left(\frac{\left|\pi_{c_{s+p}}(D_{s+p} \triangleright\triangleleft F) \cap \pi_{c_{s+p}}(D'_{s+p} \triangleright\triangleleft F')\right|}{\left|\pi_{c_{s+p}}(D_{s+p} \triangleright\triangleleft F)\right|}\right)$$

The *relative loss ratio* for F':

$$1-\left(\frac{\left|\pi_{c_{s+1}}(D_{s+1} \triangleright\triangleleft F) \cap \pi_{c_{s+1}}(D'_{s+1} \triangleright\triangleleft F')\right|}{\left|\pi_{c_{s+1}}(D'_{s+1} \triangleright\triangleleft F')\right|}\right)$$

...

$$1-\left(\frac{\left|\pi_{c_{s+p}}(D_{s+p} \triangleright\triangleleft F) \cap \pi_{c_{s+p}}(D'_{s+p} \triangleright\triangleleft F')\right|}{\left|\pi_{c_{s+p}}(D'_{s+p} \triangleright\triangleleft F')\right|}\right)$$

where $\triangleright\triangleleft$ denotes a natural join.

Given the same attributes as the previous example, the *relative loss ratio* relative to F is 40% (1–3/5). The difference with the *absolute loss ratio* (of 57%) is due to the fact that `units` '2020' and '2050' do not appear in `Claim`. The *relative loss ratio* relative to F' is 57% (1–3/7) since postcodes '3100' and '3192' do not appear in `Claim_Header`. In this example we have assumed the data is aggregated to `Operational_Unit.unit` and `Location.postcode`. Cabbibo and Torlone (2004) refer to such a case as a data loss that can be tolerated since they are not present in one of the original data marts.

Special Case (Constrained Loss Ratio): The relative loss ratio can be further reduced if the data analyst decides that s/he is interested in a subset of the data. In this case the *relative loss ratio* is applied against F and F' after one or more of their conformed dimension attributes are reduced by some conditions.

$$F = \sigma_{condition}(D_1 \triangleright\triangleleft ... \triangleright\triangleleft D_{s+p+q} \triangleright\triangleleft F)$$

$$F' = \sigma_{condition}(D_1 \triangleright\triangleleft ... \triangleright\triangleleft D_{s+p+r} \triangleright\triangleleft F')$$

To illustrate this case we apply the following conditions to our previous example:

```
condition=Period.year=2003,
Product.product_id ='COMP'
```

As a result, the *constrained loss ratio* resulting from the join between `Operational_Unit.unit` and `Location.postcode` for both fact tables becomes zero. In this case the aggregations for the `number_of_claims` and `total_claims` will be correct for the given conditions.

Applications of loss ratio: Inclusion of *relative loss* ratio during the visualization of multidimensional data is helpful in determining the correctness of the aggregations and if the inclusion of a certain dimension attribute in *drill across* will result in any loss of data. Similarly by changing the analysis space using the *dice & slice* operations the *constrained loss ratio* can be further reduced. Abello, Samos and Saltor (2002, 2003)have also considered reducing the analysis space to meaningful values (in other words where there is no loss) but by remodeling of the data at the design stage.

5. Drill Across in Hybrid Cubes

Hybrid Cubes: Given the definition of the conformed dimension attributes we have adopted in Section 4 we define *hybrid cubes* as a group of data cubes that have at least one conformed and one non-conformed dimension attributes between them.

Drill across between two cubes *C1* and *C2* where they have the same or equal dimension attributes is natural join between corresponding dimension attributes.

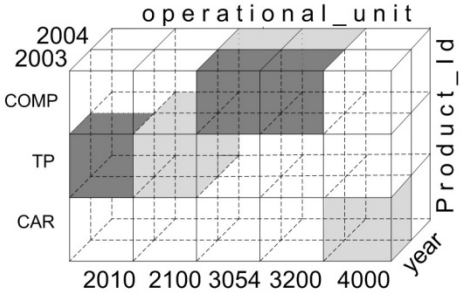

Figure 2a: Cube representation of Claim data

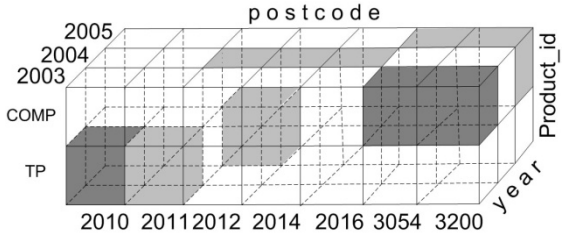

Figure 2b: Cube representation of Claim_Header

A slightly different scenario is where the dimension attributes are conformed but they have intersecting values. This results in loss of data during the join. In Figures 2a and 2b (above) the white cells are empty, light gray cells have data, and dark gray cells have data and have common coordinates with the other source. The mismatch of values between `Operational_Unit.unit` and `Location.postcode` could be responsible for some of the cells (with data in light gray) having no common coordinates in the other source.

A simple *drill across* between `Claim` and `Claim_Header` will return the dark gray cells, but we would lose all the light gray cells (which have data).

Let us assume *C1* has *p* dimension tables summarized over *s* dimension attributes, *k* of which are conformed. *C2* has *q* dimension tables summarized over *t* attributes, *k* of which are conformed. The number of measures in the fact tables F and F' is *i* and *j* respectively. *C1a* and *C2a* are *C1* and *C2* respectively summarized over their conformed attributes. The common data space *CX* is their combined data summarized over the common values of their conformed dimension attributes using natural join (Figure 3):

$$C1 = \Upsilon_{(a_1,...,a_k,a_{k+1},...,a_s),}Sum(m_1,...,m_i)$$
$$(D_1 \rhd\lhd ... \rhd\lhd D_p \rhd\lhd F)$$

$$C1a = \Upsilon_{(a_1,...,a_k),}Sum(m_1,...,m_i)$$
$$(D_1 \rhd\lhd ... \rhd\lhd D_p \rhd\lhd F)$$

$$C2 = \Upsilon_{(a_1,...,a_k,a'_{k+1},...,a'_t),}Sum(n_1,...,n_j)$$
$$(D'_1 \rhd\lhd ... \rhd\lhd D'_q \rhd\lhd F')$$

$$C2a = \Upsilon_{(a_1,...,a_k),}Sum(n_1,...,n_j)$$
$$(D'_1 \rhd\lhd ... \rhd\lhd D'_q \rhd\lhd F')$$

$$CX = \pi_{a_1,...,a_k,m_1,...,m_i,n_1,...,n_j}(C1a \rhd\lhd C2a)$$

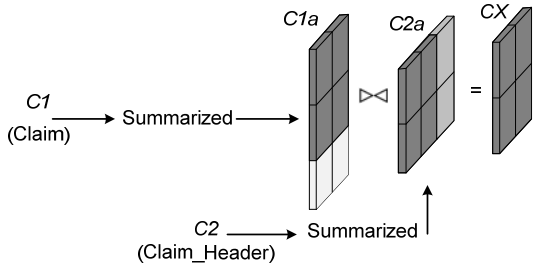

Figure 3: Obtaining of the common data space

The uncommon data spaces $C1'_1,...,C'_n$ and $C2'_1,...,C2'_n$ (in Figures 4 and 5) are cuboids within *C1* and *C2* summarized over their non-conformed dimension attributes and for a specific cell in *CX*. We extend *drill across* to also return the uncommon data spaces $C1'_1,...,C'_n$ and $C2'_1,...,C2'_n$. *n* is the number of grouping sets in *CX*.

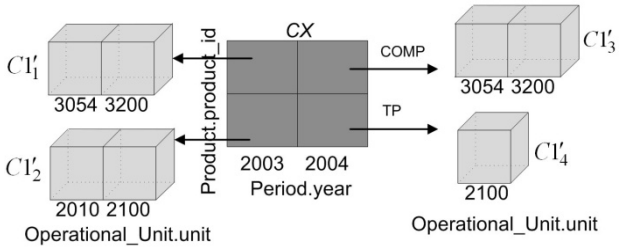

Figure 4: Drill Across from a Common Data Space to *C1*

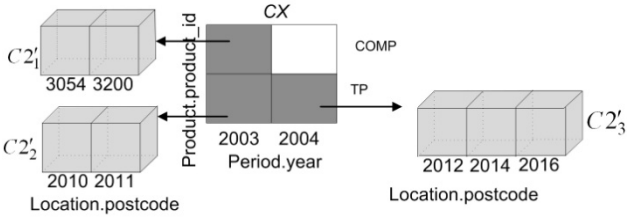

Figure 5: Drill Across from a common data space to *C2*

The obtaining of $C1'_1,...,C'_n$ and $C2'_1,...,C2'_n$ can be compared with the natural join (between conformed attributes) where the non-conformed attribute is also returned but does not participate in the join. Next, we describe the formation of the cuboids resulting from *drill across* in Relational Algebra.

The symbol Υ denotes '*Group By*' operation. Let us assume x_1 to x_n are non-empty cells with distinct coordinates in *CX:*

$$x_1(a_1 = val_{1,1},...,a_k = val_{1,k})$$
...
$$x_n(a_1 = val_{n,1},...,a_n = val_{n,k})$$

$C1'_1,...,C'_n$ and $C2'_1,...,C2'_n$ are obtained as follows:

$$C1'_1 = \Upsilon_{(a_{k+1},...,a_s),}Sum(m_1,...,m_i)\sigma_{a_1=val_{1,1},...,a_k=val_{1,k}}$$
$$(D_1 \rhd\lhd ... \rhd\lhd D_p \rhd\lhd F)$$
...
$$C1'_n = \Upsilon_{(a_{k+1},...,a_s),}Sum(m_1,...,m_i)\sigma_{a_1=val_{n,1},...,a_k=val_{n,k}}$$
$$(D_1 \rhd\lhd ... \rhd\lhd D_p \rhd\lhd F)$$
$$C2'_1 = \Upsilon_{(a'_{k+1},...,a_t),}Sum(n_1,...,n_j)\sigma_{a_1=val_{n,1},...,a_k=val_{n,k}}$$
$$(D'_1 \rhd\lhd ... \rhd\lhd D'_q \rhd\lhd F')$$
...
$$C2'_n = \Upsilon_{(a'_{k+1},...,a_t),}Sum(n_1,...,n_j)\sigma_{a_1=val_{n,1},...,a_k=val_{n,k}}$$
$$(D'_1 \rhd\lhd ... \rhd\lhd D'_q \rhd\lhd F')$$

Example:
Given the sample data in Table 1, the common data space is:

CX			
year	**product_id**	**no_of_ claims**	**total_ claims**
2003	COMP	5,355	7,522
2003	TP	2,310	3,255
2004	TP	1,178	4,166

Table 3a: The common data space between Claim and Claim_Header

The uncommon data space for a given cell in *CX* where `year=2004` and `product_id= 'TP'` is:

$C1'_4$	
unit	**no_of_claims**
2100	1,178

$C2'_3$	
postcode	**total_claims**
2012	1,127
2014	1,168
2016	1,871

Tables 3b, 3c: The uncommon data space between Claim and Claim_Header

Applications of the extended drill across: The extended *drill across* returns the data related to the non conformed dimension attributes in both sources. As we will see in the next section, visualization of the common as well as the related uncommon data spaces enables the analyst to view both sources together, compare and examine relationships between seemingly non-conformed dimension attributes.

6. Nested Pivot Tables

A pivot table is a two-dimensional and tabular representation of a single cube (or multiple cubes but using the same dimensions) with a specific orientation. As we saw in the previous sections, the result of the *drill across* between hybrid cubes *C1* and *C2* is the common as well uncommon spaces of data (*CX*, $C1'_1,...,C'_n$ and $C2'_1,...,C2'_n$). We suggest extending pivot tables to show the common data space as the *Parent Pivot Table* (PPT) and the uncommon data spaces ($C1'_1,...,C'_n$ and $C2'_1,...,C2'_n$) as *Nested Pivot Tables* (NPT) . The PPT and the NPTs are fully functional pivot tables. Any OLAP operation against a NPT does not affect the PPT. Any OLAP operation against a NPT (for example $C1'_1$) is also applied against all of its siblings ($C1'_2,...,C'_n$). Operations such as *roll-up*, *roll-in*, *dice*, *slice* and *pivoting* of dimension attributes against the PPT require recalculation of the NPTs.

Figure 6 shows how we envisage a conceptual layout of *Nested Pivot Tables* showing the result of the *drill across* between `Claim` and `Claim_Header` using the data in Table 1 and a specific orientation for PPT and NPTs.

CX | *C1′* | *C2′*

year	product_id	Claim		Claim_Header	
		unit	no_of_claims	postcode	total_claims
2003	COMP	3054	3,175	3054	4,210
		3200	2,180	3200	3,312
		Total	5355	Total	7,522
	TP	Claim		Claim_Header	
		unit	no_of_claims	postcode	total_claims
		2010	1,145	2010	1,743
		2100	1,165	2011	1,512
		Total	2,310	Total	3,255
2004	TP	Claim		Claim_Header	
		unit	no_of_claims	postcode	total_claims
		2100	1,178	2012	1,127
		Total	1,178	2014	1,168
				2016	1,871
				Total	4,166
	Total		8,843		14,943

Figure 6: Visualization of NPTs

The relationship between PPT and its NPTs can be represented using nested relations (assuming 2 cubes for simplicity).

$$CX(a_1,...,a_k,m_1,...,m_i,n_1,...,n_j)$$
$$(C1'(a_{k+1},...,a_s,m_1,...,m_i)*,$$
$$C2'(a'_{k+1},...,a_t,n_1,...,n_j)*)$$

Every tuple in CX is related to one or more tuples in $C1'$ (a union of $C1'_1,...,C'_n$) and/or $C2'$ (a union of $C2'_1,...,C2'_n$). Figure 7 shows implementation of PPT and NPTs using relational tables.

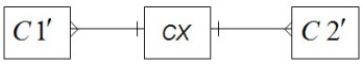

Figure 7: PPT and NPTs using relational tables

Although we have limited nesting to one level, it is possible to consider further nesting to other cubes. This however introduces unwarranted complexity since in the majority of cases we are concerned with two sources at a time.

The inclusion of the *relative loss ratio* in pivot tables similar to that shown in Figure 8 can help the analyst decide if there is a loss because of any of the attributes and if it can be tolerated.

Relative Loss Ratio:				
Claim: 0% Claim_Header:0%	Claim: 0% Claim_Header:0%	Claim: 40% Claim_Header:57%		
year	product_id	unit / postcode	no_of_claims	total_claims
2003	COMP	3054	3,175	4,210
		3200	2,180	3,312
		Total	5,355	7,522
	TP	2010	1,145	1,743
		Total	1,145	1,743
	Total		6,500	9,265

Figure 8: Pivot table showing loss ratios

It will be also possible to try to eliminate the (*constrained*) loss by limiting the values of one or more of the conformed attributes (interactively). Figure 8 (above) shows how the pivot in Figure 6 would look like if `Operational_Unit.unit` and `Location.postcode` were considered and treated as conformed attributes.

NPTs may increase the amount of visual clutter. This can be reduced by turning individual NPTs into hyperlinks. The activation of each link would result in the display of the NPT in a separate window. Figure 9 is a conceptual example of using hyperlinks to show *Nested Pivots*.

year	product_id	Claim	Claim_Header
2003	COMP	unit for year=2003,product_id=COMP	postcode for 2003 COMP
	TP	unit for year=2003,product_id=TP	postcode for 2003 TP
2004	TP	unit for year=2004,product_id=TP	postcode for 2004 TP
	Total	8,843	14,943

Figure 9: Based on Figure 6 using hyperlinks for NPT

6.1 Applications of NPT

The ability to visualize *hybrid cubes* makes it possible to view and analyze multiple cubes with non-conformed dimension attributes. NPT enable analysis of multiple related cubes by applying single OLAP operations to multiple cubes in the same space. Referring to Figure 6 it is possible to compare the values for `postcode` and `unit` and compare the number of claims not only for the same year and product but also for certain `postcode` and `unit` which may or may not point to the same geographic location. An implementation of NPT extended by visual data mapping can be an effective tool in improving the intersection between non-conformed dimension attributes.

7. Related Work

7.1 Conformity

Kimball and Ross (2000) define two dimension (tables) conformed if they have identical keys and attributes; they must also have identical values or one must be a subset of another. They also add that conformed dimension (tables) must mean the same thing. Giovinazzo (2003) requires the same instance of a conformed dimension table to be joined to multiple fact tables. Mundy, Thornwaite and Kimball (2006) define two dimension (tables) conformed if they have the same name and contents. As acknowledged by Cabibbo and Torlone (2004), in absence of conformity it will not be possible to combine or perform meaningful aggregation of measures across

two data marts with non-conformed dimension tables due to the loss of data resulting from the join between these dimensions tables.

The requirements of dimension conformity at the table level are restrictive to the integration of autonomous data marts because it is unlikely that dimension tables from two sources are identical in every sense. Moreover, it will be even less likely to achieve conformity if the dimension tables use generated keys. The requirement of conformity at the attribute level is less restrictive because the conformity is applied to dimension attributes as opposed to dimension tables. Also, it does not require the names or contents to be identical, but it still requires that the two dimension attributes refer to the same thing.

Cabbibo and Torlone (2004) acknowledge that Kimball's definition of conformed dimensions is not suitable to autonomous data marts and define *dimension compatibility* as an alternative. According to this definition two dimensions are compatible if they intersect and there exist lossless expressions *E1* and *E2* such that when applied to their respective dimensions, makes them to be equivalent.

The definition we have adopted is closer to the definition of compatible dimension in the sense that it is applied to dimension attributes and recognizes that they may not have identical values; it differs however in the sense that it does not require a lossless expression to make them equal to begin with. Instead, we calculate the loss ratio as a quality factor of the conformity to be considered during the visualization or in a manual or automated integration process.

Abello , Samos and Saltor (2002, 2003) also find the strict requirement of dimension conformity to be restrictive for the operation of *drill across* which according to them requires selected instances of the dimensions to determine instances in another and that the domains are related in some way. This definition also appears to require the dimension attributes to be or to become equal by means of some relationship between their domains.

7.2 Drill Across

Abello, Samos and Saltor (2002, 2003) define the operation *drill across* as changing subject (facts) in the same analysis space. The authors have identified semantic relationships: *Derivation*, *Generalization*, *Association* and *Flow* to extend possibilities to *drill across*. These relationships improve the conformity between attributes but the approach does not address the loss of data related to non-conformed attributes.

Cabbibo and Torlone (2004) define *drill across* as an extension to the natural join where the intersection of the two dimensions is aggregated at the finest grain of the dimensions. Our proposed extension to *drill across* goes further and returns the related uncommon spaces of data (Section 5).

7.3 Visualization of Multiple Sources

The ability to explore multiple cubes and the inability of pivot tables to provide this function has been extensively studied. Alternative methods have been proposed to visualize multiple cubes and provide flexible representation of dimension structure and different graphical representations of data.

Vinnik and Mansmann (2006) show a tree like structure of dimensions as an effective method in visualizing dimension structures.

According to Stolte, Tang and Hanrahan (2002), Polaris achieves visualization of multiple cubes by partitioning data into groups and allocating them into panes.

ADVIZOR introduced by Eick (2000) uses a set of linked views displayed on the same screen with each view used to explain a number of measures. There is however no evidence that ADVIZOR can visualize multiple related cubes.

Visual Pivot introduced by Conkin, Prabhakar and North (2002) aims at visualization of data structures composed of multiple intersecting hierarchies called Polyarchies sharing at least one node. The aim of Visual Pivot is not to combine two similar hierarchies but to track similar information in multiple hierarchies.

CoDecide introduced by Gebhardt, Jarke and Jeusfeld (1998) is an OLAP data visualization tool that enables multiple users to have different views of one or multiple data cubes and participate in a cooperative analysis of the subject.

These alternative methods address the shortcoming of the pivot tables to visualize multiple cubes, but they cannot fully substitute pivot tables. The tabular representation of data in pivot tables makes it easy to read and rotate data by novice users. Pivot tables are used in many of the Business Intelligence applications and used by majority of the analysts. *Nested Pivot Tables* maintain the strengths of pivot tables and allow multiple sources that share at least one conformed attribute to be viewed and operated together.

8. Conclusion and Future Work

In this paper we have described the benefits of identifying conformed dimension attributes and considered the degree of (dis)similarity between attribute values a quality factor of the conformity between them. We described methods to measure the loss that results from joining conformed but intersecting dimension attributes. We described how the operation *drill across* can be extended to also return the uncommon data spaces relative to the common data space. We introduced *Nested Pivot Tables* as an extension to the Pivot Tables to show the results of our extended *drill across*.

In our future work we intend to implement our extension to *drill across* and visualization of *Nested Pivot Tables*, and use it in a framework that aims at accessing and integrating multidimensional data sources.

9. References

[1] Abello, A., Samos, J. and Saltor, F. (2003): Implementing Operations to Navigate Semantic Star Schemas, *Proceedings of the 6th ACM International Workshop on Data Warehousing and OLAP*, New Orleans, Louisiana, USA, ACM Press, pp 56-62.

[2] Abello, A., Samos, J. and Saltor, F., (2002): On relationships offering new drill-across possibilities. *Proceedings of the 5th ACM international Workshop on Data Warehousing and OLAP*, McLean, Virginia, USA, ACM Press, pp 7-13.

[3] Cabibbo, L. and Torlone, R. (2004): Dimension compatibility for data mart integration. *Proceedings of the 12th Italian Symposium on Advanced Database System*s, Cagliari, Italy, pp. 6-17.

[4] Chaudhuri, S. and Dayal, U. (1997): An overview of data warehousing and OLAP technology. *ACM SIGMOD Record **26**(1)*, 65-74.

[5] Conkin, N., Prabhakar, S. and North, C. (2002): Multiple foci-drill-down through tuple and attribute aggregation polyarchies in tabular data. *IEEE Symposium on Information Visualization*, Minnesota, USA, pp 131-134.

[6] Eick, S. (2000): Visualizing multi-dimensional data. *ACM SIGGRAPH Computer Graphics **34**(1)*, 61-67.

[7] Gebhardt, M., Jarke, M., Jeusfeld, M. A. Quix, C. and Sklorz, S. (1998): Tools for Data Warehouse Quality. In *Proceedings of the 10th international Conference on Scientific and Statistical Database Management*, Washington, DC, IEEE Computer Society, pp 229-232.

[8] Giovinazzo, W. (2003): *Object Oriented Data Warehouse Design: Building a Star Schema*, Prentice Hall PTR, NJ.

[9] Gray, J., Chaudhuri, S., Bosworth, A., Layman, A., Reichart, D., Venkatro, M., Pellow, F. and Pirahesh H. (1997): Data Cube: A Relational Aggregation Operator Generalizing Group By, Cross-Tab and Sub-Totals. *Data Mining and Knowledge Discovery* **1**(1), 29-53.

[10] Kimball, R. and Ross, M. (2000): *The Data Warehouse Toolkit*, Wiley.

[11] Mundy, J., Thornthwaite, W. and Kimball, R. (2006): *The Microsoft Data Warehouse Toolkit*, Wiley.

[12] Stolte, C., Tang, D. and Hanrahan, P. (2002): Polaris: A System for Query, Analysis and Visualization of Multidimensional Relational Databases. *IEEE Trans. Visualization and Computer Graphics*, **8**(1), 52-65.

[13] Vinnik, S. and Mansmann, F. (2006): From Analysis to Interactive Exploration: Building Visual Hierarchies from OLAP Cubes. *Proceedings of 10th International Conference on Extending Database Technology*, Munich, Germany, Lecture Notes in Computer Science **3896**, Springer, pp. 496-514.