Reasoning about Summarizability in Heterogeneous Multidimensional Schemas

Carlos A. Hurtado[1] and Alberto O. Mendelzon[2]

[1] University of Toronto (chl@db.toronto.edu)
[2] University of Toronto (mendel@db.toronto.edu)

Abstract. In OLAP applications, data are modeled as points in a multidimensional space. Dimensions themselves have structure, described by a schema and an instance; the schema is basically a directed acyclic graph of granularity levels, and the instance consists of a set of elements for each level and mappings between these elements, usually called *rollup functions*. Current dimension models restrict dimensions in various ways; for example, rollup functions are restricted to be total. We relax these restrictions, yielding what we call *heterogeneous schemas*, which describe more naturally and cleanly many practical situations. In the context of heterogeneous schemas, the notion of *summarizability* becomes more complex. An aggregate view defined at some granularity level is *summarizable* from a set of precomputed views defined at other levels if the rollup functions can be used to compute the first view from the set of views. In order to study summarizability in heterogeneous schemas, we introduce a class of constraints on dimension instances that enrich the semantics of dimension hierarchies, and we show how to use the constraints to characterize and test for summarizability.

1 Introduction

The multidimensional model is becoming increasingly important as a logical layer for visualizing and querying data in OLAP scenarios. A key aspect of multidimensional data is the separation of *factual* and *dimensional* data. While dimensions represent descriptive and relatively static data, facts depict event-based data, represented as points in spaces defined by dimensions.

A number of multidimensional models for OLAP [CT97][HMV99a][LAW98] [JLS99] have recently incorporated dimensions as first-class entities in query and update languages. In the logical layer, a dimension is composed of a schema and an instance. The dimension schema includes a directed acyclic graph (DAG) of levels, called *hierarchy schema*, where levels may have attributes associated with them. Levels can be viewed as foreign keys of tables containing values for the attributes.

On the other hand, a dimension instance consists of a set of members for each level, called *member sets*; and a hierarchy relation that models the ancestor/descendant relation between the members. For instance, we may have *Toronto* as a member of a level *City* in a dimension representing locations.

In most of the models [CT97] [HMV99a] [HMV99b], the hierarchy relation is represented by a set of functions between member sets, called *rollup functions*. Usually, we say that a level l_1 *rolls up* to another level l_2 when there exists an edge from l_1 to l_2 in the hierarchy, meaning that there is a rollup function from l_1 to l_2.

1.1 Heterogeneous Dimensions

Suppose we have a dimension representing stores in Canada and the USA. While the stores in Canada roll up to cities and to provinces, the stores in the USA roll up to cities and to states. Figures 1 (B), and (C) show two possible dimension hierarchies for the location dimension (we abstract away the attributes); a possible instance is depicted in Figure 1(D). Notice that in the schemas (B) and (C) the rollup functions are total. An alternative schema for representing this dimension is depicted in Figure (A). However, here we have that the rollup functions between $City$ and $State$, and between $City$ and $Province$ are partial, because some cities do not have states while others do not have provinces.

At this point, we need to introduce some terminology that will be central in this paper, and which we formalize in the next section. A dimension schema is *homogeneous* if for every pair of levels l_1 and l_2 such that l_1 rolls up to l_2 we have that in every dimension instance conveyed in the schema, the rollup function is a total function from the member set of l_1 to the member set of l_2. A dimension schema is *strictly homogeneous* if it is homogeneous and it has only one level that is at the bottom of the hierarchy schema; a dimension schema is *heterogeneous* if it is not homogeneous.

Coming back to our example, the schema (A) is heterogeneous becuse in the instance of Figure 1 (D), the rollup function between $City$ and $State$ is a partial function from $\{NewYork, Toronto\}$ to $\{NYstate\}$. On the other hand, the schema (B) is homogeneous, and the schema (C) is is strictly homogeneous. The choice between schemas (A), (B), or (C) depends on factors like the attributes that the levels share, and how we would like to group members into levels in the dimension. The schemas that allow heterogeneity could be better in many situations. As an example, if we restrict the schemas (B) and (C) to be homogeneous, the insertion of the stores in Washington D.C. would require a new sub-hierarchy *StoreWashington*-*CityWashington* below the level *Country*, where *CityWashington* contains, in any instance, only one member (because *Washington* is the only city in USA that does not have a *State*.) Modeling heterogeneity allows the fusion of levels that represent the same granularity of aggregation, reducing the complexity of the schema and, therefore, of query formulation. The benefit of having a more flexible model at the logical layer can also be propagated to the storage layer. For instance, heterogeneity allows the fusion of levels that share attributes, while keeping separate levels with different attributes.

We end this section by noting that current dimension models [CT97] [HMV99a] [HMV99b] [JLS99] do not allow heterogeneity.

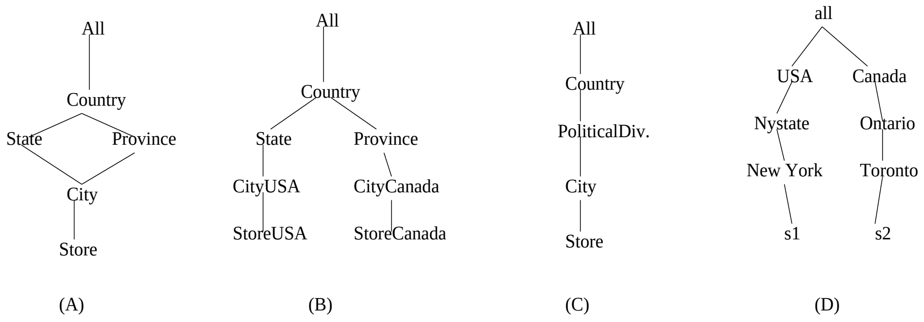

Fig. 1. (A) (B) (C) Three alternative dimension hierarchies for the location dimension. (D) An instance of the location dimension.

1.2 Summarizability

Data cubes [GBLHP96] comprise the computation of a set of aggregate views, called cube views, which represent facts aggregated at different granularities (set of levels taken from a set of dimensions). Different fundamental functionalities for OLAP, like *aggregate navigation*, *cube computation*, and *cube maintenance* [CD97], among others, require the derivation, using pre-defined aggregate queries, of cube views from other pre-computed cube views, transparently to the user. In order to do this, the system must determine which derivations are correct; in other words, whether a cube view can be derived from other cube views through predefined aggregate queries.

The notion of summarizability [RS90] refers to the conditions under which we can correctly derive any cube view defined at a level l_2 from a cube view defined at a level l_1 by aggregating using the rollup functions between l_1 and l_2 of a particular dimension. In order to allow summarizability the data cube must be *distributive*, i.e., its aggregate functions must be distributive [GBLHP96]. A *distributive aggregate function af* can be computed on a set by partitioning the set into disjoint subsets, aggregating each separately, and then computing the aggregation of these partial results with another aggregate function[1] that we denote by af^c.

The central problem we address in this paper is: how we can infer summarizability in distributive data cubes from the dimension schema, without having to analyze the dimension instance? In particular, it has been shown [HRU96] that, in strictly homogeneous dimensions, correct summarizations correspond to the edges of the dimension hierarchy, as depicted in the following example. We give the formalization using the relational algebra with bag semantics, extended with the *generalized projection operator* to express aggregation [AGS⁺96]. The generalized projection operator, Π_A, is an extension of the duplicate-eliminating

[1] Among the SQL aggregate functions, $COUNT$, SUM, MIN, and MAX are distributive. We have that $COUNT^c = SUM$; for SUM, MIN, and MAX we have $af^c = af$.

projection, where A can include both regular and aggregate attributes. For simplicity, all the aggregates of cube views are assumed to be on a single attribute.

Example 1. Consider the dimension with schema depicted in Figure 1 (B). From the schema we can infer the correctness of the summary operation that computes the total sales per province from the total sales per cities in Canada: $\Pi_{Province,Sum(Sales)}(Sales\_CityCan \bowtie \Gamma^{Province}_{CityCan})$, where $\Gamma^{Province}_{CityCan}$ represents the rollup function from the cities of Canada to the provinces.

In this paper we extend the notion of summarizability, previously introduced for dimensions over strictly homogeneous schemas, in order to consider derivations from *sets of levels* to levels in dimensions over dimension schemas that are not strictly hierarchical. The following example shows the extra complexity of inferring summarizability in heterogeneous dimension schemas.

Example 2. Consider the hierarchy schema of the location dimension of Figure 1 (A). Consider the cube views $Sale\_Province$, and $Sale\_State$ representing the total sales per province and state, respectively, and defined as follows:

$$\begin{array}{lcl} Sale\_Province & = & \Pi_{Province,Sum(Sales)}(Sales\_Store \bowtie \Gamma^{Province}_{Store}) \\ Sale\_State & = & \Pi_{State,Sum(Sales)}(Sales\_Store \bowtie \Gamma^{State}_{Store}) \end{array}$$

Consider the following aggregation that derives the total sales per Country from $Sale\_Province$, and $Sale\_State$:

$$\Pi_{Country,Sum(Sales)}\ (\Pi_{Country,Sum(Sales)}(Sales\_Province \bowtie \Gamma^{Country}_{Province})) \uplus (\Pi_{Country,Sum(Sales)}(Sales\_State \bowtie \Gamma^{Country}_{State}))$$

where $\Gamma^{l_2}_{l_1}$ represents the rollup function from l_1 to l_2; and $\uplus$ represents the *additive union* which adds the multiplicity of the tuples. Intuitively, we could say that this derivation is correct, because there are no stores that roll up to a state and to province at the same time. However, we cannot infer the correctness of this derivation from the schema (A), because it does not capture precisely the possible set of instances we are modeling; in particular, it does not explicitly disallow a store that rolls up to both a state and a province.

The question that arises at this point is: what additional constraints we have to add in order to keep the ability to reason about summarizability in heterogeneous schemas.

1.3 Contributions and Outline

In this paper we introduce a dimensional model that accounts for heterogeneity. We define the notion of summarizability for heterogeneous dimensions, which smoothly extends the notion of summarizability for homogeneous dimensions. We identify four classes of dimension schemas, that go from strictly homogeneous to a class of schemas we call *hierarchical schemas*, that allow heterogeneity but keep a notion of ordering between the granularities defined by levels. We introduce a class of constraints, *split constraints*, that enrich the semantics of dimension hierarchies. Finally, we solve the problem of deciding summarizability in hierarchical dimensions constrained with split constraints. The solution is obtained from a sound and complete axiomatization of a subclass of split constraints.

The remainder of the paper is organized as follows. Section 2 introduces the model for dimensions, along with the formalization of the notion of summarizability. Section 3 introduces the new constraints we propose. The inference problem for summarizability is studied in Section 4. In Section 5 and Section 6 we show related work, conclude and outline some of the prospects for future work.

2 Heterogeneous Dimensions

In this section, we describe our framework for modeling dimensions. A dimension schema will consist of a hierarchy schema and a set of constraints.

2.1 Hierarchy Schema

We define a hierarchy schema in the same fashion as in [JLS99] and [HMV99a], but allowing heterogeneity, multiple hierarchical paths, and multiple bottom levels. Multiple hierarchical paths are frequently required as argued in [HMV99a], [CT97] . Allowing multiple bottom levels makes it possible to have more natural schemas in several situations as shown in [JLS99].

Consider a set of members $\mathbf{E}$, a set of levels $\mathbf{L}$, and set of attributes $\mathbf{A}$.

Definition 1 (Hierarchy Schema). *A* hierarchy schema *is a tuple* $G = (L, \nearrow, \mathcal{A}, \sigma)$*, where* $L \subseteq \mathbf{L}$ *is a set of levels with a distinguished level All;* $\nearrow$ *is a binary relation on* L *such that* $(L, \nearrow)$ *conforms a rooted DAG with All as root (we denote by* $\nearrow^*$ *the transitive closure of* $\nearrow$*);* $\mathcal{A} \subseteq \mathbf{A}$ *is a set of attributes; and* $\sigma : L \rightarrow 2^{\mathcal{A}}$ *assigns to each level a set of attributes.*

We refer to the set of bottom levels of a hierarchy schema, $\{l \in L \mid \neg\exists l' \in L : l' \nearrow l\}$, as L_{Bottom}. Given two levels $l_a, l_b \in L$, we denote by Υ_{l_a,l_b} the set of paths between l_a and l_b in G.

2.2 Dimension Instance and Schema

An instance of a dimension is obtained by specifying a set of members for each level, along with the descendant/ancestor relation $<$ between them.

Definition 2 (Dimension Instance). *A* dimension instance *is a tuple* $(G, \mathcal{E}, <, \mathcal{T})$*, where* G *is a hierarchy schema;* $\mathcal{E}$ *is a set of sets of members in* $\mathbf{E}$*, one set* $\mathcal{E}_l$ *for each level* $l \in L$*, where we denote by* $\mathcal{E}_d$ *the union of the member sets of a dimension* d*;* $<$ *is a relation between members conforming a rooted DAG with root* `all`*, where we denote by* $<<$ *the transitive closure of* $<$*; and finally,* $\mathcal{T}$ *is a* *set of relations that contains one relation $\mathcal{T}_l$, with attributes $\sigma(l) \cup \{l\}$, for every level $l \in L$. The following conditions hold: (1) the member set of a level l, $\mathcal{E}_l$, is the active domain of l in $\mathcal{T}_l$, and l is a key of $\mathcal{T}_l$; (2) for every pair of members e_a, e_b such that $e_a \in \mathcal{E}_{l_a}$, $e_b \in \mathcal{E}_{l_b}$ and $e_a < e_b$, we have $l_a \nearrow l_b$, and there are no members $e_1, \ldots, e_n \in \mathcal{E}_d$ such that $e_a < e_1 < \ldots e_n < e_b$; (3) for every member $e \in \mathcal{E}_d$ such that $e \neq$ `all` we have $e <<$ `all`.*

The first condition says that the relation for level l contains exactly one tuple for each element in $\mathcal{E}_l$. The second condition essentially says that the edges in the dimension hierarchy represent links between levels, that must exist whenever we have a *direct* descendant/ancestor relation between some pair of members in the levels. And finally, the last condition states that all the members reach the top member `all`. Note that the member sets are not necessarily disjoint.

Given a dimension d, a *leaf member* is a member $e \in \mathcal{E}_d$ with no descendant members. An important feature of the model is that it may have leaf members in non-bottom levels. This allows updating the dimension, as shown in [HMV99b] (for instance, we might want to add a city but do not yet have stores that belong to it.) However, in order to simplify the presentation, we make two assumptions: (a) all the leaf members belong to the bottom levels; and (b) the member sets of the bottom levels are pairwise disjoint. We define a *base level*, denoted by l_{base}, containing the union of the member sets of the bottom levels. The results in this paper can be extended to dimensions where (a) and (b) does not hold by a more detailed treatement of base members, which basically consists in defining them as id's of the leaves.

Definition 3 (Rollup Operators). *Given a dimension instance h, we define the direct rollup operator, that takes a dimension and two of its levels l_1, l_2 and gives a relation with attributes l_1 and l_2 defined as follows: $d\Gamma_{l_1}^{l_2} = \{(x_1, x_2) \mid x_1 \in \mathcal{E}_{l_1} \wedge x_2 \in \mathcal{E}_{l_2} \wedge x_1 < x_2\}$. We have the rollup operator with the same signature of $d\Gamma$ which gives a relation with attributes l_a and l_2 defined as follows: $\Gamma_{l_1}^{l_2} = \{(x_1, x_2) \mid x_1 \in \mathcal{E}_{l_1} \wedge x_2 \in \mathcal{E}_{l_2} \wedge x_1 << x_2\}$. The* base rollup operator *takes a level l a defines the relation, with attributes l_{base} and l, that groups the base elements to it, and is defined as follows: $\Gamma_{l_{base}}^{l} = \{(x, y) \mid x \in \mathcal{E}_{l_{base}} \wedge y \in \mathcal{E}_l \wedge x << y\}$.*

The following are some basic properties of the rollup operators: given a dimension instance d we have: (a) if $\neg(l_a \nearrow l_b)$ then $d\Gamma_{l_a}^{l_b} = \emptyset$; (b) if $\neg(l_a \nearrow^* l_b)$ then $\Gamma_{l_a}^{l_b} = \emptyset$; (c) if $\Upsilon_{l_a,l_b} = \{l_a l_b\}$ then $\Gamma_{l_a}^{l_b} = d\Gamma_{l_a}^{l_b}$.

Definition 4 (Partitioned Instances). *A dimension instance is* partitioned *when all its rollup relations are single valued (partial functions). The partitioning property appears as an inherent constraint in the the dimension models of [CT97], [HMV99a], [LAW98], and [JLS99]. It requires that each member in the base level reach, through $\Gamma_{l_{base}}^{l}$, not more that one member in each level l. In this sense, each levels represent partitioned classifications of the base members. In the sequel we assume that all dimension instances are partitioned.*

We are ready to define dimension schema as a hierarchy schema plus a set of constraints in some constraint language $\mathcal{CL}$. In Section 3 we introduce a specific $\mathcal{CL}$, the language of split constraints.

Definition 5 (Dimension Schema). *A $\mathcal{CL}$-*dimension schema *is a tuple $ds = (G, \Sigma)$ where G is a hierarchy schema; and $\Sigma \subseteq \mathcal{CL}$.*

The constraint language must have a notion of satisfaction, denoted by $\models_{\mathcal{CL}}$. A dimension instance d is over a dimension schema ds if $G_d = G_{ds}$, $d \models_{\mathcal{CL}} \Sigma$. Given a dimension schema ds we denote by $\mathcal{I}(ds)$ the set of dimension instances that are over ds.

2.3 Classes of Dimension Schemas

The following definition formalizes the classes of dimension schemas we mentioned in Section 1.

Definition 6 (Homogeneous and Heterogeneous Dimension Schemas). *A dimension schema ds is* homogeneous *if every dimension d over ds satisfies: for every pair of levels l_1, l_2 such that $l_1 \nearrow l_2$ we have that $\Gamma_{l_1}^{l_2} : \mathcal{E}_{l_1} \rightarrow \mathcal{E}_{l_2}$ is a total function. A dimension schema is* strictly homogeneous *if it is homogeneous, and has a single bottom level. A dimension schema is* heterogeneous *if it is not homogeneous.*

In a homogeneous dimension instance, the ordering of the levels provided by the graph is exactly the same as the ordering of the grains represented by the levels. In other words, as we move up the hierarchy schema we reach levels that represent coarser partitions of the base members. Furthermore, the rollup functions capture precisely the containment relation between the partitions of levels connected in the graph. These two properties can no longer be true in heterogeneous dimension instances.

In order to formalize this intuition, let us define the notion of overlap, as introduced for classification schemas in statistical databases [Mal93]. The overlap between two levels l_1 and l_2 is a relation $\Theta_{l_1}^{l_2}$ with signature $\mathcal{E}_{l_1} \times \mathcal{E}_{l_2}$ that has an edge between the members e_1 and e_2 iff the intersection of the sets of base members that reach them is non-empty. The overlap between two levels can be defined with the following relational-algebra expression: $\Theta_{l_1}^{l_2} = \pi_{l_1,l_2}(\Gamma_{l_{base}}^{l_1} \bowtie \Gamma_{l_{base}}^{l_2})$. We say that $l_1 \preceq l_2$ in a dimension instance d if $\Theta_{l_1}^{l_2} = \Gamma_{l_1}^{l_2}$, That is, $l_1 \preceq l_2$ means: (a) the grain of l_1 is finer than the grain of l_2 because the overlap is a function (possibly partial); and (b) the rollup function $\Gamma_{l_1}^{l_2}$ captures precisely the containment relation between the grains of l_1 and l_2.

Example 3. The dimension instance of Figure 2 satisfies $City \preceq State$, i.e., there is no City c and State s such that they overlap and c does not roll up to s. Another way to think of the fact that $City \preceq State$ is as follows: if there is a store that rolls up to a city c and to a state s, then c rolls up to s.

It is important to note, that in general $\preceq$ is not transitive, i.e., if a dimension instance h satisfies $l_1 \preceq l_2$, and $l_2 \preceq l_3$ it does not necessarily satisfy $l_1 \preceq l_3$.

Example 4. Consider the dimension instance depicted in Figure 2. Although we have $City \preceq SaleDistrict$, and $SaleDistrict \preceq SaleRegion$, we do not have $City \preceq SaleRegion$, because two stores which roll up to $NewYork$, such as s_1 and s_2, roll up to two different sale regions, r_1 and r_2.

Definition 7 (Hierarchical Dimension Schemas). *A dimension schema ds is* hierarchical *if for every pair of levels $l_1, l_2 \in L_{ds}$ such that $l_1 \nearrow l_2$ we have that every dimension instance $d \in \mathcal{I}(ds)$ satisfies $l_1 \preceq l_2$. A dimension schema ds is* strictly hierarchical *if for every pair of levels $l_1, l_2 \in L_{ds}$ such that $l_1 \nearrow^* l_2$ we have that every dimension instance $d \in \mathcal{I}(ds)$ satisfies $l_1 \preceq l_2$.*

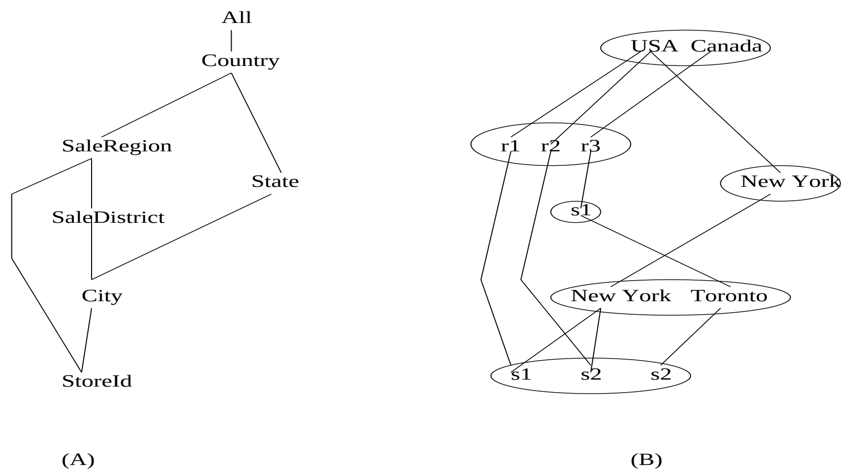

Fig. 2. (A) A hierarchy schema of a Location dimension; (B) A dimension instance for (A).

The relationship among the classes of schemas is given by the following chain of inclusions: Strictly Homogeneous $\subset$ Homogeneous $\subset$ Strictly Hierarchical $\subset$ Hierarchical .

2.4 Summarizability

In this section we extend the notion of summarizability to heterogeneous dimensions. A level represents a granularity of aggregation. In this sense, a level l can be associated with a one-dimensional cube view with no select conditions, that has the following form: $\Pi_{l,m=af(m)}(\Gamma^{l}_{l_{base}} d \bowtie F)$, where d is a dimension; F is a relation, called a *fact table*, with a special attribute m called the *measure*, and with l_{base} included in its attributes; af is an aggregate function; and l is a level. This cube view will be abbreviated as $cv(d, F, l, af(m))$.

Sumarizability of levels in a dimension is related to the correct derivation of one dimensional cube views from other one dimensional cube views using rollup relations for grouping. Such derivations involve queries of the form: $\Pi_{l,af^c(m)}(\biguplus_{i\in 1\dots n}(\pi_{l,m}\Gamma^l_{l_i}d \bowtie F_i))$. Intuitively, we are aggregating a set of fact tables $F_1, \dots, F_n$, using the rollup mappings $\Gamma^l_{l_1}, \dots, \Gamma^l_{l_n}$.

Definition 8 (Summarizability). *Given a dimension instance d, a set of levels $L = \{l_1, \dots, l_n\}$, and a level l, l is* summarizable *from L in d if for every fact table F, and distributive aggregate function af, we have: $cv(d, F, l, af(m)) = \Pi_{l,af^c(m)}(\biguplus_{i\in 1\dots n}(\pi_{l,m}(\Gamma^l_{l_i} \bowtie cv(d, F, l_i, af(m)))))$ Given a dimension schema ds, a set of levels $L = \{l_1, \dots, l_n\}$, and a level l, l is* summarizable *from L in ds if l is summarizable from L in every instance d in $\mathcal{I}(ds)$.*

Example 5. In the dimension instace of Figure 2, we have that $Country$ is summarizable from $\{SaleDistrict, State\}$, but it is not summarizable from $\{SaleRegion, State\}$.

The following lemma gives an alternative definition of summarizability.

Lemma 1 (Summarizability). *A level l is summarizable from a set of levels $L = \{l_1, \dots, l_n\}$ in a dimension instance d iff $\Gamma^l_{l_{base}} = \biguplus_{i\in 1\dots n}\pi_{l_{base},l}(\Gamma^{l_i}_{l_{base}} \bowtie \Gamma^l_{l_i})$*

Recall that we are assuming that the dimension is partitioned. From Lemma 1 we have that a level l is summarizable from a single level l_1 in a dimension d iff $\Gamma^l_{l_{base}} = \Gamma^{l_1}_{l_{base}} \bowtie \Gamma^l_{l_1}$.

3 Split Constraints

In this section we propose a class of constraint we call *split constraints*. The intuition behind them is that usually there are dependencies between the rollup functions that start from a common level. For example, we could say that the cities that roll up to states do not roll up to any province in the location dimension of Example 1.

To start, a *split expression $\alpha(x)$ for level l* is a propositional formula (with the usual connectives $\neg, \wedge, \vee, \supset, \Leftrightarrow$, and $\oplus$ for exclusive disjunction) over atoms of the form $\exists x_i : \Gamma^{l_i}_l(x, x_i)$, or $\exists x_i : d\Gamma^{l_i}_l(x, x_i)$. Note that all the atoms represent rollup from a common level l, and the only free variable is x. We use $\perp$ and $\top$ to denote the false and the true proposition respectively.

Definition 9 (Split Constraints). *A* split constraint for level l *is an expression $\forall x \in \mathcal{E}_l : \alpha(x)$, where $\alpha(x)$ is a split expression for level l. A dimension instance d satisfies a split constraint s, denoted $d \models s$, if s is true when each fact $\Gamma^{l_i}_l(x, x_i)$ and $d\Gamma^{l_i}_l(x, x_i)$ is interpreted respectively by the direct and indirect rollup relations of d. A* base split constraint *is a split constraint whose atoms all rollup from l_{base}.*

Example 6. The split constraint: $\exists x_1 : \Gamma_l^{l_1}(x, x_1) \supset \exists x_2 : \Gamma_l^{l_2}(x, x_2)$ says that if a member e in l rolls up to a member e_1 in l_1, then e rolls up to a member in l_2.

Note that, if we associate the variables of the atoms with the level that they bind to, abstracting away the arguments and the existential quantifiers in the atoms, we can write split constraints in an abbreviated form. For example, the split constraint in the last example can be written as $\Gamma_l^{l_1} \vee \Gamma_l^{l_2}$.

Example 7. We can impose the following set of split constraints over the skeleton of Figure 2 (A). $\Sigma_S = \{(a)\ \Gamma_{StoreId}^{City}, (b)\ \Gamma_{City}^{SaleDistrict} \oplus \Gamma_{City}^{State}, (c)\ \Gamma_{SaleDistrict}^{SaleRegion}, (d)\ \Gamma_{SaleRegion}^{Country}, (e)\ \Gamma_{State}^{Country}\}$.

The following example depicts the use of direct rollup atoms in a split constraint.

Example 8. The split constraint: $d\Gamma_{City}^{State} \supset \neg\Gamma_{City}^{County}$ says that if a city rolls up *directly* to a state it does not roll up to any county.

We can characterize, using base split constraints, structural anomalies that can be present in a schema. A level could be *empty*, i.e., l could be constrained to not have any members in any instances of the schema; formally, this is captured by $\neg\Gamma_{l_{base}}^{l}$. Even if we have a pair of levels l_1, l_2 such that $l_1 \nearrow l_2$, the rollup function between them could be empty in all the instances of the schema; in this case we have $\neg\Gamma_{l_{base}}^{l_1} \vee \neg\Gamma_{l_{base}}^{l_2}$. Finally, we can have a dimension schema that is unsatisfiable; this can be characterized as $\neg\Gamma_{l_{base}}^{l}$ for every $l \in l_{base}$. As an example, if we imposed Γ_{City}^{State} to the schema of Example 7 *SaleDistrict* would be an empty level.

4 Inferring Summarizability

In this section we give an algorithm for testing summarizability in the dimension instances defined by a given hierarchical dimension schema, where the constraint language consists of split constraints. Therefore, unless otherwise stated, we assume that the dimension schemas mentioned are hierarchical and $\mathcal{CL}$ is the class of split constraints.

4.1 Conditions for Summarizability

In this section, we show that the problem reduces to testing inference of base split constraints. Given a dimension schema ds, and two of its levels, l_1, l_n, we define the following split expression:

$$\Delta_{l_1,l_n} \equiv_{def} \begin{cases} \bigvee_{l_1 \ldots l_n \in \Upsilon_{l_1,l_n}} (\Gamma_{l_{base}}^{l_1} \wedge \ldots \wedge \Gamma_{l_{base}}^{l_n}) & \text{if } l_1 \nearrow^* l_n \\ \bot & \text{otherwise.} \end{cases}$$

Note that if $l_1 \nearrow l_n$ then $\Delta_{l_1}^{l_n}$ is equivalent to $\Gamma_{l_{base}}^{l_1} \wedge \Gamma_{l_{base}}^{l_n}$, and if there is no path from l_1 to l_n it is equivalent to $\bot$. The intuition behind the above expression is that if we have Δ_{l_1,l_n}, then every base member e_b reaches an element e_n in

l_n passing through some element e_1 in l_1. For instance, if we state $\Delta_{l_1,l} \oplus \Delta_{l_2,l}$, then every base element reaches an element in l passing trough l_1, or through l_2, but not through l_1 and l_2. Note that the above statements do not always hold for non hierarchical dimension schemas.

Now, we give a lemma that characterizes summarizability in terms of a base split constraint.

Lemma 2. *A level l is summarizable from a set of levels $L = \{l_1, \ldots, l_n\}$ in a hierarchical dimension d iff $d \models \Gamma^{l}_{l_{base}} \supset (\Delta_{l_1,l} \oplus \ldots \oplus \Delta_{l_n,l})$.*

The intuition behind Lemma 2 is that in order for l to be summarized from L, we need that every base member e_b that reaches a member e in l, reaches e passing trough one and only one of the levels in L.

Example 9. Consider the Location dimension depicted in Figure 2. Assume that it is hierarchical, and the base level is denoted by $Store$. We have that $Country$ is summarizable from $\{SaleDistrict, State\}$ iff $Location \models \Gamma^{Country}_{StoreId} \supset ((\Gamma^{SaleDistrict}_{StoreId} \wedge \Gamma^{SaleRegion}_{StoreId} \wedge \Gamma^{Country}_{StoreId}) \oplus (\Gamma^{State}_{StoreId} \wedge \Gamma^{Country}_{StoreId}))$. Note that the above split expression can be simplified to: $\Gamma^{Country}_{StoreId} \supset ((\Gamma^{SaleDistrict}_{StoreId} \wedge \Gamma^{SaleRegion}_{StoreId}) \oplus \Gamma^{State}_{StoreId})$.

4.2 Derivation of Base Split Constraints

In the previous section, we found necessary and sufficient conditions for summarizability in terms of base split constraints. In this section, we give a sound and complete set of inference rules for implication of base split constraints. The notion of implication is stated as follows: given a dimension schema ds and split constraint α, we say that $ds \models \alpha$ if for every dimension instance $d \in \mathcal{I}(ds)$ we have $d \models \alpha$. We denote by Σ^{+}_{B} the set of base split constraints $\{\beta \mid ds \models \beta\}$.

The first two derivation rules we introduce reflects the assumptions (a) and (b) in Section 2.2, respectively.

Rule 1 *For every level $l \in L \setminus L_{Bottom}$ we have $\Gamma^{l}_{l_{base}} \supset \bigvee_{\{l_i \mid l_i \nearrow l\}} \Gamma^{l_i}_{l_{base}}$.*

Rule 2 *We have $\bigoplus_{l_i \in L_{Bottom}} \Gamma^{l_i}_{l_{base}}$.*

Rule 1 states that there are no leaves in the internal levels, and Rule 2 says that every base member reaches one and only one member in a bottom level. The next rule reflects the condition (3) of Definition 2.

Rule 3 *for every level $l \in L$ such that $l \neq All$, we have Γ^{All}_{l}.*

Next, we show how to transform split constraints into equivalent base split constraints. We will do this by expressing atoms in terms of base rollup atoms. We need the following definition: given a dimension schema ds and two of its levels l_a and l_b we have:

$$d\Delta_{l_a,l_b} \equiv_{def}$$

$$\begin{cases} \Gamma^{l_a}_{l_{base}} \wedge \Gamma^{l_b}_{l_{base}} \wedge \neg(\bigvee_{l_a l_1 \ldots l_n l_b \in \Upsilon_{l_a,l_b}} \Gamma^{l_1}_{l_{base}} \wedge \ldots \wedge \Gamma^{l_n}_{l_{base}}) & \text{If } \{l_a l_b\} \subset \Upsilon_{l_a,l_b} \\ \Gamma^{l_a}_{l_{base}} \wedge \Gamma^{l_b}_{l_{base}} & \text{if } \Upsilon_{l_a,l_b} = \{l_a l_b\} \\ \bot & \text{otherwise} \end{cases}$$

For instance, the base split constraint $d\Delta_{l_a,l_b}$ states that every base element reaches an element e_b in l_b, passing trough an element e_a in l_a that reaches directly e_b.

Now, we give a characterization of the relationship between split and base split constraints.

Lemma 3. *Given a dimension schema ds, and a split constraint α, then for every hierarchical dimension d over ds we have $d \models \alpha$ iff $d \models \Gamma^{l}_{l_{inf}} \supset \alpha'$, where α' is obtained from α by replacing every indirect rollup atom $\Gamma^{l_i}_{l}$ with Δ_{l,l_i}, and replacing every direct rollup atom $d\Gamma^{l_i}_{l}$ with $d\Delta_{l,l_i}$.*

The following rule that transforms split constraints to base split constraints comes from Lemma 3.

Rule 4 *Given a split constraint α, then we have $\Gamma^{l}_{l_{inf}} \supset \alpha'$, where α' is obtained from α by replacing every indirect rollup atom $\Gamma^{l_i}_{l}$ with Δ_{l,l_i}, and replacing every direct rollup atom $d\Gamma^{l_i}_{l}$ with $d\Delta_{l,l_i}$.*

The last rule we give shows that base split constraints can be derived from other base split constraints using propositional logic derivation. Let $\models_{prop}$ be the propositional implication of two base split constraint when considering rollup atoms as propositional variables.

Rule 5 *If we have the split constraints $\alpha_1, \ldots, \alpha_n$, and $\alpha_1, \ldots, \alpha_n \models_{prop} \beta$, then we have β.*

Finally, we show that the given axiomatization is sound and complete for deriving Σ^{+}_{B}.

Theorem 1 (Soundness and Completeness of Base Split Inference). *Rules 1, 2, 3, 4, and 5 are sound and complete for Σ^{+}_{B}.*

Example 10. Consider the base split constraint that we have to test in Example 9 in order to decide whether $Country$ is summarizable from $\{SaleDistrict, State\}$: $(*)\ \Gamma^{Country}_{StoreId} \supset ((\Gamma^{saleDistrict}_{StoreId} \wedge \Gamma^{SaleRegion}_{StoreId}) \oplus \Gamma^{State}_{StoreId})$. Now, we give a derivation for $(*)$.

(1) $\Gamma^{City}_{StoreId} \supset \Gamma^{SaleDistrict}_{StoreId} \oplus \Gamma^{State}_{StoreId}$	Rule 4 from split constraint (b)
(2) $\Gamma^{SaleDistrict}_{StoreId} \oplus \Gamma^{State}_{StoreId}$	Rule 5 from (1) and split constraint (a)
(3) $\Gamma^{SaleDistrict}_{StoreId} \Leftrightarrow (\Gamma^{SaleDistrict}_{StoreId} \wedge \Gamma^{SaleRegion}_{StoreId})$	Rule 5 from split constraint (c)
(4) $(\Gamma^{SaleDistrict}_{StoreId} \wedge \Gamma^{SaleRegion}_{StoreId}) \oplus \Gamma^{State}_{StoreId}$	Rule 5 from (2) and (3)
(5) $\Gamma^{Country}_{StoreId}$	sequence of applications of Rules 1 and 5
(*)	Rule (5) from (4) and (5)

4.3 Complexity

The following theorem gives a lower bound for the problem of inferring summarizability from a given dimension schema.

Theorem 2. *Given a dimension schema ds, a level l and a set of levels L, whether l is summarizable from L in every dimension $d \in \mathcal{I}(ds)$ is coNP-hard.*

Proof. (Sketch) Transformation from VALIDITY.

An upper bound of $O(n^2 2^n)$, where n is the number of levels of the hierarchy schema, can be obtained by the algorithm in Figure 3. The upper bound is basically caused by the steps (2) and (3) of the algorithm, which are in $O(n^2 2^n)$.

Input: A heterogeneous dimension schema ds, a level $l \in L_{ds}$, and a set $L \subseteq L_{ds}$
Output: Whether l is summarizable from L in all the instances of ds
(1) Apply Rules 1 and 2, and 3, having Σ_I
(2) Apply Rule 4 to every split constraint in Σ_I and Σ, giving Σ_{BS}.
(3) For every level $l_i \in L$ compute $\Delta_{l_i,l}$
(4) If $\Sigma_{BS} \models_{prop} \Gamma^{l}_{l_{inf}} \supset (\Delta_{l_1,l} \oplus \ldots \oplus \Delta_{l_n,l})$ then return true; else return false

Input: A homogeneous dimension schema ds, a level $l \in L_{ds}$, and a set $L \subseteq L_{ds}$
Output: Whether l is summarizable from L all the instances of ds
(1) Let $L' = L \cap \{l' \mid l' \nearrow^* l\}$
(2) If every bottom level that reaches l reaches exactly one level in L' then return true; else return false.

Fig. 3. Algorithms for testing summarizability: (Above) in a hierarchical dimension schema with split constraints, (Below) in a Homogeneous Dimension Schema without empty levels.

It is easy to show that testing summarizability in strictly homogeneous schemas is in polytime; we only have to check whether there is exactly one level in L that reaches l in the dimension hierarchy. The following lemma shows that testing summarizability in homogeneous schemas without empty levels is also in polytime.

Lemma 4. *Testing summarizability in an homogeneous schema without empty levels is in polytime.*

Proof. (Sketch) A polytime algorithm is given in Figure 3.

5 Related Work

Cabibbo and Torlone [CT97] introduced one of the first formal models for OLAP dimensions; it allows representing only strictly homogeneous dimensions. Further models [LAW98], [HMV99a] have the same restriction. Jagadish et al. [JLS99] introduce in their model the ability to represent non-strict homogeneous schemas, referring to them as *unbalanced*. Although they motivate the need for modeling heterogeneity, their model does not go beyond homogeneity. The model of Lehner et al. [LAW98] considered a case of heterogeneity arising between levels and attributes, that has a trivial treatement using null values.

The notion of summarizability was introduced by Rafanelli and Shoshani [RS90] as a property of statistical objects. Lenz and Shoshani [LS97] give conditions for summarizability of a level from a single level. Using a different, rather informal framework, they basically state that if (a) $l_1 \nearrow^* l_2$, (b) the rollup relations are functions, and (c) they are total between member sets, then l_1 is summarizable from l_2 in a dimension d. They refer to condition (b) as the rollup relations being *disjoint*, and condition (c) as being *complete*. Assuming that the dimension is partitioned (or disjoint in their terminology), the above conditions are basically equivalent to the special case of Lemma 1 when summarizability is from a single level.

Split constraints have the flavor of the *disjunctive existential constraints* (dec's) introduced by Goldstein [Gol81], which basically specify where null values may occur in a relation. Let us denote by $\mathcal{T}_d$ the relational table that represents the dimension instance d realized as a single table in the star schema. Intuitively, $\mathcal{T}_d$ is the relation that has as attributes the levels $L \cup \{l_{base}\}$, and is defined by the outer-join of the base rollup relations. It easy to show that the partitioned property of a dimension d is equivalent to having l_{base} as a key for $\mathcal{T}_d$. Basically, a dec says that whenever a tuple is non-null for an attribute, it must be non-null for all the attributes in at least one set of attributes in a given list. Only a subset of bottom split constraints can be represented using dec's over $\mathcal{T}_d$; split constraints are more general because they allow any propositional condition on non-null attributes.

6 Conclusion

The restriction of homogeneity in current dimension models leads to unnatural dimension schemas in many real situations. The relaxation of this condition, however, weakens the ability to reason about summarizability in the schema. We identify a class of constraints that overcome this problem and enrich the semantics of dimension schemas. We solve the the problem of inferring summarizability in a class of heterogeneous schemas. The problem is still open for non partitioned, and heterogeneous schemas in general. The exploration of further classes of constraints and their relation with summarizability is also a challenging problem to pursue.

It is interesting to note that homogeneous dimension schemas convey data in exactly the same way as hierarchical data models. Furthermore, the notion of heterogeneity and the results of this paper extend to hierarchical data modeling in general.

Summarizability is a particular case of the problem of using materialized views to compute aggregate views [SSJL96]. Little has been said about the interplay between constraints and aggregate view rewriting in the context of OLAP dimensions and hierarchies. Our work establishes the foundations for further research in this area.

Acknowledgements

This research was supported by the National Science and Engineering Research Council and the Institute for Robotics and Intelligent Systems of Canada.

References

[AGS+96] R. Agrawal, A. Gupta, S. Sarawagi, P. Deshpande, S. Agarwal, J. Naughton, and R. Ramakrishnan. On the computation of multidimensional aggregates. In *Proceedings of the 22nd VLDB Conference*, Bombay, India, 1996.

[CD97] S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. In *ACM SIGMOD Record 26(1)*, March 1997.

[CT97] L. Cabibbo and R. Torlone. Querying multidimensional databases. In *Proceedings of the 6th DBPL Workshop*, East Park, Colorado, USA, 1997.

[GBLHP96] J. Gray, A. Bosworth, A. Layman, and H. H. Pirahesh. Data cube : A relational operator generalizing group-by, cross-tab and sub-totals. In *Proceedings of the 12th IEEE-ICDE Conference*, New Orleans, Los Angeles, USA, 1996.

[Gol81] B. A. Goldstein. Constraints on null values in relational databases. In *Proceedings of the 7th VLDB Conference*, Cannes, France, 1981.

[HMV99a] C. Hurtado, A. Mendelzon, and A. Vaisman. Maintaining data cubes under dimension updates. In *Proceedings of the 15th IEEE-ICDE Conference.*, Sydney, Australia, 1999.

[HMV99b] C. Hurtado, A. Mendelzon, and A. Vaisman. Updating OLAP dimensions. In *Proceedings of the 2nd IEEE-DOLAP Workshop*, Kansas City, Missouri, USA, 1999.

[HRU96] V. Harinarayan, A. Rajaraman, and J. Ullman. Implementing data cubes efficiently. In *Proceedings of the 1996 ACM-SIGMOD Conference*, Montreal, Canada, 1996.

[JLS99] H. V. Jagadish, L. V. S. Lakshmanan, and D. Srivastava. What can hierarchies do for data warehouses? In *Proc. of the 25th VLDB Conference*, Edinburgh, Scotland, UK, 1999.

[LAW98] W. Lehner, H. Albrecht, and H. Wedekind. Multidimensional normal forms. In *Proceedings of the 10th SSDBM Conference*, Capri, Italy., 1998.

[LS97] H. J. Lenz and A. Shoshani. Summarizability in OLAP and statistical databases. In *Proceedings of the 9th SSDBM Conference*, Olympia, Washington, USA, 1997.

[Mal93] Francesco Malvestuto. A universal-scheme approach to statistical databases containing homogeneous summary tables. In *ACM Transactions on Database Systems, Vol. 18, No. 4.*, December 1993.

[RS90] M. Rafanelli and A. Shoshani. Storm: A statistical object representation model. In *Proceedings of the 5th SSDBM Conference*, Charlotte, N.C., USA, 1990.

[SSJL96] D. Srivastava, D. Shaul, H. V. Jagadish, and A. Levy. Answering queries with aggregation using views. In *Proceedings of the 22nd VLDB Conference*, Bombay, India, 1996.