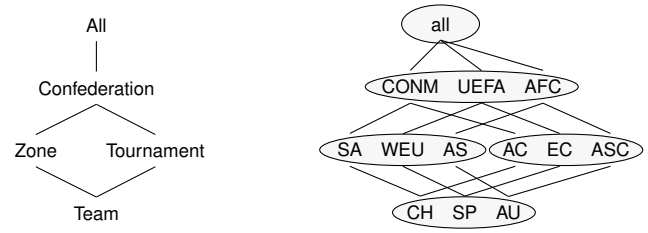
Handling Inconsistencies in Data Warehouses with Extended Dimensions

Juan Ramírez
Universidad del Bío-Bío
Concepción, Chile
juaramir@ubiobio.cl

Mónica Caniupán
Universidad del Bío-Bío
Concepción, Chile
mcaniupa@ubiobio.cl

Loreto Bravo
Universidad de Concepción
Concepción, Chile
lbravo@udec.cl

***Abstract*—Dimensions in Data Warehouses (DWs) are modeled using a hierarchical schema of categories. A dimension should satisfy a set of constraints to ensure that queries can be answered efficiently using pre-computed answers. For many reasons a dimension might become inconsistent and there might be several ways to fix it. In order to represent this uncertainty, we introduce the concept of *extended* dimensions, that is, dimensions where categories contain *sets* of elements which allow to represent ambiguity. In this article we formalize extended dimensions and a suitable way to answer queries from them.**



(a) Dimension hierarchy (b) Dimension $\mathcal{D}_{\text{FT}}$

Fig. 1. Football Teams Dimension

I. Problem Statement

Data Warehouses (DWs) are data repositories that integrate data from different sources, and keep historical data for analysis and decision support. DWs organize data according to the multidimensional model, in which, dimensions reflect the perspectives upon which facts are viewed, and the facts correspond to events which are usually associated to numeric values known as *measures*, and are referenced using the dimension elements. Dimensions are modeled as hierarchies of elements, where each element belongs to a category. The categories are also organized into a hierarchy called hierarchy schema. Figure 1(a) shows a Football Teams dimension with a *bottom* category Team, that rolls-up to Zone, and Tournament. Categories Zone and Tournament go to Confederation which reaches category All. Figure 1(b) shows the elements for categories of the dimension $\mathcal{D}_{\text{FT}}$, and the *rollup relations* between them.

The multidimensional structure allows users to compute queries at different levels of granularity. For example, in the dimension in Figure 1 we can compute summaries grouped by zone or confederation and so on. Efficient query answering in DWs relies in the use of pre-computed results at lower categories to compute aggregates at higher levels in dimensions hierarchies. In order to be able to do this, the dimensions should satisfy *strictness* and *covering* constraints [1]–[3].

For many reasons a dimension might become inconsistent and the need of repairing it arises so that correct answers can be computed when using pre-computed answers. Alternatives to fix inconsistencies have been proposed: (i) finding a new dimension, called a *minimal repair*, which satisfies the constraints and that is close to the inconsistent one [3] or (ii) by constructing a *canonical repair* which adds extra elements to categories to take into consideration the ambiguity introduced by the possible ways to fix the violations of the constraints [4]. In order to represent this uncertainty in repairs, we introduce the concept of *extended* dimension, that is, a dimension where categories contain *sets* of elements. In this way, a rollup relation between sets s_1 and s_0 implies that all rollup combinations between elements in s_0 and s_1 can possibly be valid. This ambiguity allows the representation of the unknown ways in which the inconsistent dimension can be fixed and allows to provide ranges of answers between the aggregate values that are known to be part of the answer and values that *might* be part of it. Thus, the answers from an extended dimension will not be exact, but a range within which the answer is known to belong. In [4], no formalization is given for this type of dimension nor for its query answering semantics. In the next section we formalize *extended dimensions* and how to query them.

II. Extended Dimension

An *extended dimension* $\mathcal{X}$ is a tuple $(\mathcal{H}_\mathcal{X}, \mathcal{E}_\mathcal{X}, \mathsf{CElem}_\mathcal{X}, \ll_\mathcal{X})$, where $\mathcal{H}_\mathcal{X} = (\mathcal{C}_{\mathcal{H}_\mathcal{X}}, \nearrow_{\mathcal{H}_\mathcal{X}})$ is a hierarchy schema; $\mathcal{E}_\mathcal{X}$ is a set of constants, called elements; $\mathsf{CElem}_\mathcal{X} : \mathcal{C}_{\mathcal{H}_\mathcal{X}} \rightarrow \mathcal{P}(\mathcal{P}(\mathcal{E}_\mathcal{X}))$is a function that, given a category returns a set of subsets of elements in $\mathcal{E}_\mathcal{X}$; and the relation $\ll_\mathcal{X} \subseteq \mathcal{P}(\mathcal{E}_\mathcal{X}) \times \mathcal{P}(\mathcal{E}_\mathcal{X})$ that corresponds to the child/parent relation between elements of different categories. We denote with $\ll^*_\mathcal{X}$ the reflexive and transitive closure of $\ll_\mathcal{X}$. The following conditions hold: (i) $\mathsf{all}_\mathcal{X}$ is the only element in category $\mathsf{All}_{\mathcal{H}_\mathcal{X}}$. (ii) For all $\mathsf{c}_i, \mathsf{c}_j \in \mathcal{C}_{\mathcal{H}_\mathcal{X}}$, if $\mathsf{c}_i \neq \mathsf{c}_j$ then $\mathsf{CElem}_\mathcal{X}(\mathsf{c}_i) \cap \mathsf{CElem}_\mathcal{X}(\mathsf{c}_j) = \emptyset$. (iii) For all pair of elements $a \in \mathsf{CElem}_\mathcal{X}(\mathsf{c}_i)$ and $b \in \mathsf{CElem}_\mathcal{X}(\mathsf{c}_j)$ if $a \ll_\mathcal{X} b$ then $\mathsf{c}_i \nearrow_{\mathcal{H}_\mathcal{X}} \mathsf{c}_j$. (iv) For each $\mathsf{c}_i \in \mathcal{C}_{\mathcal{H}_\mathcal{X}}$ it holds that: (a) $\emptyset \notin \mathsf{CElem}_\mathcal{X}(\mathsf{c}_i)$. (b) If $e \in \mathsf{CElem}_\mathcal{X}(\mathsf{c}_i)$ then for each element

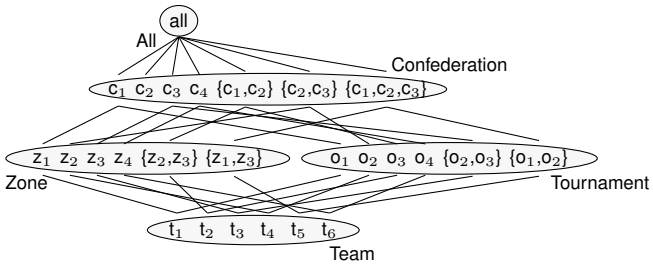


Fig. 2. Extended Dimension $\mathcal{X}_{\text{FT}}$

$e' \in e$ it holds that $\{e'\} \in \mathsf{CElem}_{\mathcal{X}}(\mathsf{c}_i)$. (v) for the bottom category $\mathsf{c}_b \in \mathcal{C}_{\mathcal{H}_\mathcal{X}}$ it holds that for every $e \in \mathsf{CElem}_{\mathcal{X}}(\mathsf{c}_b)$, e is a singleton. Condition (iii) ensures that the child/parent relation ($\ll_{\mathcal{X}}$) only connects elements of categories that are connected in the schema. Condition (iv) ensures that every element that is a set in a category contains only elements of this category. Condition (v) enforces that elements in the bottom category are only singleton elements, and in this way, we will be able to join with the data in the fact tables. In order to simplify presentation we will sometimes replace a singleton element $\{e\}$ by e. Given an extended dimension $(\mathcal{H}_\mathcal{X}, \mathcal{E}_\mathcal{X}, \mathsf{CElem}_\mathcal{X}, \ll_\mathcal{X})$ with hierarchy schema $\mathcal{H}_\mathcal{X} = (\mathcal{C}_{\mathcal{H}_\mathcal{X}}, \nearrow_{\mathcal{H}_\mathcal{X}})$. For each pair of categories $\mathsf{c}_i, \mathsf{c}_j \in \mathcal{C}_{\mathcal{H}_\mathcal{X}}$ such that $\mathsf{c}_i \nearrow^*_{\mathcal{H}_\mathcal{X}} \mathsf{c}_j$, there is a rollup relation, denoted by $\mathcal{R}_\mathcal{X}(\mathsf{c}_i, \mathsf{c}_j)$, that has the set of pairs $\{(a,b) | a \in \mathsf{CElem}_\mathcal{X}(\mathsf{c}_i), b \in \mathsf{CElem}_\mathcal{X}(\mathsf{c}_j)$ and $a \ll^*_\mathcal{X} b\}$.

A possible extended dimension $\mathcal{X}_{\text{FT}}$ for the hierarchy schema in Figure 1(a) is shown in Figure 2. Note that some of the elements in Zone, Tournament and Confederation are sets. The extended dimension can be treated as a traditional dimension $\mathcal{T}(\mathcal{X})$ where all sets belonging to a category are considered as simple elements. For example, dimension $\mathcal{T}(\mathcal{X})$ for the extended dimension $\mathcal{X}_{\text{FT}}$ in Figure 2, is a dimension where categories Team, Zone and Tournament contain six distinct elements and category Confederation contains seven distinct elements. However, note that this results in loosing the information about the connection between elements. For example, it would interpret that there is no connection between elements o_1 and $\{\mathsf{o}_1, \mathsf{o}_2\}$. Thus, we could think that when posing a query over $\mathcal{T}(\mathcal{X})$ we could use techniques currently implemented in DWs systems. However, as we will show, query answering needs to be redefined for extended dimensions to take full advantage of the flexibility given by them.

Consider a Sales fact table (which is not shown here because of space restrictions) associated to dimensions football team ($\mathcal{X}_{\text{FT}}$) and time ($\mathcal{X}_{\text{TIME}}$). The table below shows the results obtained from posing aggregate queries $\mathcal{Q}_{\text{SUM}}$ and $\mathcal{Q}_{\text{COUNT}}$ that compute, respectively, the SUM and COUNT of incomes grouped by category Confederation of dimension $\mathcal{T}(\mathcal{X}_{\text{FT}})$, and category Year of dimension $\mathcal{T}(\mathcal{X}_{\text{TIME}})$.

	Conf.	Year	$\mathcal{Q}_{\text{SUM}}$	$\mathcal{Q}_{\text{COUNT}}$
t_1	c_1	2010	700	1
t_2	$\{c_1, c_2\}$	2010	200	1
t_3	$\{c_2, c_3\}$	2010	200	1
t_4	c_3	{2010,2011}	600	1
t_5	c_4	2010	500	1

As it can be seen, the results obtained by treating the extended dimension as a normal dimension do not capture the fact that different elements might contribute to the same aggregation. For example, tuple t_2 in the table might contribute to the answer in tuple t_1 since c_1 is contained in element $\{\mathsf{c}_1, \mathsf{c}_2\}$.

Thus, we need a query answering semantics specially designed for extended dimensions that takes into consideration the relationship between elements within a category. We first define set $\mathcal{A}(t, \mathcal{Q}, \{\mathcal{X}_1, \ldots, \mathcal{X}_i, \ldots, \mathcal{X}_n\}, F)$ that contains all the tuples that might contribute to an answer t. For example, Given a query $\mathcal{Q}$, an extended dimension $\mathcal{X} = (\mathcal{H}_\mathcal{X}, \mathcal{E}_\mathcal{X}, \mathsf{CElem}_\mathcal{X}, \ll_\mathcal{X})$, a fact table F, and a tuple $t = \langle e_1, \ldots, e_n \rangle$ with $e_i \in \mathcal{E}_\mathcal{X}$ for every $i \in [1, n]$, the set of *tuples associated to* t *in* $\mathcal{Q}(\{\mathcal{T}(\mathcal{X}_1), \ldots, \mathcal{T}(\mathcal{X}_i), \ \ldots, \mathcal{T}(\mathcal{X}_n)\}, F)$, denoted by $\mathcal{A}(t, \mathcal{Q}, \{\mathcal{X}_1, \ \ldots, \mathcal{X}_i, \ \ldots, \mathcal{X}_n\}, F)$, is $\{a | a = \langle s_1, \ldots, s_n, v \rangle, a \in \mathcal{Q}(\mathcal{T}(\mathcal{X}_1), \ldots, \ \mathcal{T}(\mathcal{X}_i), \ \ldots, \mathcal{T}(\mathcal{X}_n), F), e_i \in s_i$ for $i \in [1, n]\}$. We use the set of associated tuples to answers a query $\mathcal{Q}$ with aggregate functions SUM or $COUNT$ over a set of extended dimension S and a fact table F.

If we go back to our ongoing example with queries $\mathcal{Q}_{\text{SUM}}$ and $\mathcal{Q}_{\text{COUNT}}$ we see that we need to consider the interaction between the elements in the given table to provide their answers. For tuple $t = \langle c_1, 2010 \rangle$ set $\mathcal{A}(t, \ \mathcal{Q}_{\text{SUM}}, \ \{\mathcal{X}_{\text{FT}}, \ \mathcal{X}_{\text{TIME}}\}, \text{Sales})$ contains tuples t_1 and t_2, thus the final answer for SUM is $\langle c_1, 2010, [700, 900] \rangle$, where 900 is the sum of the aggregate values for tuples in set $\mathcal{A}(t, \mathcal{Q}_{\text{SUM}}, \{\mathcal{X}_{\text{FT}}, \ \mathcal{X}_{\text{TIME}}\}, \text{Sales})$. For tuple $t = \langle c_2, 2010 \rangle$ set $\mathcal{A}$ contains tuples t_2 and t_3, since there is no answer to t in the table, the final answer for SUM is $\langle c_2, 2010, [0, 400] \rangle$ where value 400 is the sum of the aggregate values for tuples in $\mathcal{A}$. For tuple $t = \langle c_3, 2010 \rangle$ set $\mathcal{A}$ contains tuples t_3 and t_4, again there is no answer to t, then the final answer is $\langle c_3, 2010, [0, 800] \rangle$ where value 800 is the sum of the aggregate values for tuples in $\mathcal{A}$. For tuple $t = \langle c_3, 2011 \rangle$ set $\mathcal{A}$ contains tuple t_4, since again there is no answer to t, the final answer is $\langle c_3, 2011, [0, 600] \rangle$. Finally, for the tuple $t = \langle c_4, 2010 \rangle$ set $\mathcal{A}$ has the unique tuple t_5, then, the final answer is $\langle c_4, 2010, [500, 500] \rangle$. An analogous analysis can be done for the aggregate function COUNT.

Even though the notion of extended dimension we propose is motivated for cleaning inconsistent dimensions, extended dimensions can be used to express dimensions that are in nature imprecise, or have a certain level of uncertainty and ambiguity on the rollup relations among its elements.

REFERENCES

[1] H.-J. Lenz and A. Shoshani, "Summarizability in OLAP and Statistical Data Bases," in *SSDBM'97*, 1997, pp. 132–143.

[2] C. Hurtado, C. Gutierrez, and A. Mendelzon, "Capturing Summarizability with Integrity Constraints in OLAP," *ACM Transactions on Database Systems*, vol. 30, no. 3, pp. 854–886, 2005.

[3] M. Caniupán, L. Bravo, and C. A. Hurtado, "Repairing inconsistent dimensions in data warehouses," *Data Knowl. Eng.*, vol. 79-80, pp. 17–39, 2012.

[4] L. Bertossi, L. Bravo, and M. Caniupán, "Consistent Query Answering in Data Warehouses," in *AMW'09*, vol. 450, 2009.