# Generalizing Temporal Dependencies for Non-Temporal Dimensions

Jef Wijsen and Raymond T. Ng

December 21, 1999

TR-48

A TIMECENTER Technical Report

| | |
|---|---|
| Title | Generalizing Temporal Dependencies for Non-Temporal Dimensions |
| |  |
| Author(s) | Jef Wijsen and Raymond T. Ng |
| Publication History | This report contains the same results as [23] but introduces different examples. December 1999. A TIMECENTER Technical Report. |

## TIMECENTER Participants

**Aalborg University, Denmark**
Christian S. Jensen (codirector), Michael H. Böhlen, Heidi Gregersen, Dieter Pfoser, Simonas Šaltenis, Janne Skyt, Giedrius Slivinskas, Kristian Torp

**University of Arizona, USA**
Richard T. Snodgrass (codirector), Bongki Moon

**Individual participants**
Curtis E. Dyreson, Bond University, Australia
Fabio Grandi, University of Bologna, Italy
Nick Kline, Microsoft, USA
Gerhard Knolmayer, Universty of Bern, Switzerland
Thomas Myrach, Universty of Bern, Switzerland
Kwang W. Nam, Chungbuk National University, Korea
Mario A. Nascimento, University of Alberta, Canada
John F. Roddick, University of South Australia, Australia
Keun H. Ryu, Chungbuk National University, Korea
Michael D. Soo, amazon.com, USA
Andreas Steiner, TimeConsult, Switzerland
Vassilis Tsotras, University of California, Riverside, USA
Jef Wijsen, University of Mons-Hainaut, Belgium
Carlo Zaniolo, University of California, Los Angeles, USA

For additional information, see The TIMECENTER Homepage:
URL: `<http://www.cs.auc.dk/TimeCenter>`



The TIMECENTER icon on the cover combines two "arrows." These "arrows" are letters in the so-called *Rune* alphabet used one millennium ago by the Vikings, as well as by their precedessors and successors. The Rune alphabet (second phase) has 16 letters, all of which have angular shapes and lack horizontal lines because the primary storage medium was wood. Runes may also be found on jewelry, tools, and weapons and were perceived by many as having magic, hidden powers.

The two Rune arrows in the icon denote "T" and "C," respectively.

**Abstract**

Recently, there has been a lot of interest in *temporal granularity*, and its applications in temporal dependency theory and data mining. *Generalization hierarchies* used in multi-dimensional databases and OLAP serve a role similar to that of time granularity in temporal databases, but they also apply to non-temporal dimensions, like space.

In this paper, we first generalize temporal functional dependencies for non-temporal dimensions, which leads to the notion of roll-up dependency (RUD). We show the applicability of RUDs in conceptual modeling and data mining. We then indicate that the notion of time granularity used in temporal databases is generally more expressive than the generalization hierarchies in multi-dimensional databases, and show how this surplus expressiveness can be introduced in non-temporal dimensions, which leads to the formalism of RUD with negation (RUD$^\neg$). A complete axiomatization for reasoning about RUD$^\neg$ is given.

# 1 Introduction

Recently, there has been a lot of interest in OLAP and data mining. In these domains, *generalization hierarchies* play an important role [5, 8, 10, 14]. A typical example is the time hierarchy, where years are partitioned into months, months into days, and so on. Nevertheless, many other hierarchies have been exemplified in the literature; for example, the earth surface can be partitioned into continents, countries, states, and so on. Also recently, there has been a lot of research focusing almost exclusively on *time granularity* [3]. Demonstrably, time granularity has useful applications in temporal dependency theory [19, 21, 20] and temporal data mining [4, 24].

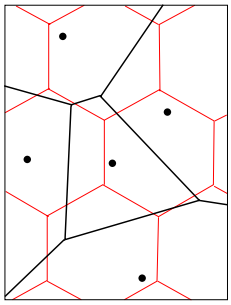
Clearly, generalization hierarchies in OLAP serve a role similar to that of time granularity in temporal databases. A natural and important question then is: what precisely are the differences/commonalities (if any) between both concepts? This basic question has many facets, including the following:

- What properties (if any) are so typical of the temporal dimension that justify its special treatment? To which extent does temporal dependency theory carry over to multiple dimensions? What is the spatial analog of time granularity?
- How much dimensional semantics can be captured in the "star" schemas used in ROLAP (see, e.g., [10]), where all dimensions are basically treated uniformly, and where dimensions and facts share the same representation formalism, namely the relational model? In particular, how accurate can time be modeled in this formalism?

These and similar questions underlie the research presented in this paper. Careful reading of the literature gives a number of useful hints.

- Jensen et al. [12] study temporal dependencies, and mention [12, page 579] that their work can be generalized to spatial dimensions. This work, however, does not deal with temporal or spatial granularity.
- Wang et al. [19, page 119] mention an approach where time is treated as a conventional attribute, and the time hierarchy is captured by FDs like $\texttt{DATE} \rightarrow \texttt{MONTH}$ and $\texttt{MONTH} \rightarrow \texttt{YEAR}$. They give two concrete examples where this naive approach falls short. Although the examples are interesting, they are rather intricate and do not explain under which conditions the naive approach fails.

The outline of the paper is as follows. Section 2 extends temporal functional dependencies (TFDs) to non-temporal generalization hierarchies, including spatial ones. After a motivating example, the construct of roll-up is formalized. The notion of roll-up dependency (RUD) is defined, and a sound and complete axiomatization for reasoning about RUDs is given. Section 3 shows two interesting applications: one in conceptual modeling, and one in data mining. Section 4 starts by illustrating that the concept of time granularity used in temporal databases is generally more expressive than the information hierarchies used in OLAP. In simple words, whereas generalization hierarchies are confined to finer-than relationships (for example, month is finer than year), temporal granularity also considers more complex relationships, including disjunctive ones (for example, every week is entirely contained in a year *or* a fiscal year, where a fiscal year runs from July 1 to June 30). We show how this surplus expressiveness can be generalized for non-temporal dimensions in an elegant way, by allowing negation in RUDs, which leads to the formalism of RUD$^\neg$. A sound and complete axiomatization for reasoning about RUD$^\neg$ is given.



VORONOI (thick lines) and HEXAGON (background)

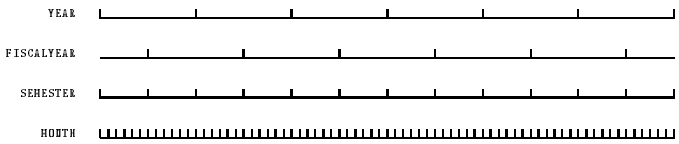



Figure 1: Spatial and temporal granularity.

# 2 Roll-Up Dependency

## 2.1 Motivating Example

Assume we have daily petrol prices from $N$ petrol stations over a period of multiple years. This information is stored as a number of tuples over the schema $(D : \texttt{DATE})(L : \texttt{LOCATION})(Price : \texttt{EUROCENT})$, where a tuple $(D : x)(L : y)(Price : z)$ means that the price of 100 gallons of petrol on day $x$ at the station located at $y$ amounted to $z$ Eurocent. Hence, we are faced with a large number of prices, giving such a profusion of detailed information that direct comparison is impossible. The information has first to be summarized into a few numbers that measure in some way the various aspects of the large masses of data in which we are interested.

Suppose there are 5 major petrol terminals, and every petrol station is supplied by the nearest petrol terminal. In Figure 1 *top*, if one thinks of the dots as terminal locations, then the thick lines partition the plane into cells such that two points within the same cell have the same nearest terminal. Such a partitioning of the plane based on proximity relations is called a Voronoi diagram [7]. We may find the following rule: if all prices are rounded to the nearest integral Euro, thereby ignoring price differences below one Euro, then the price of 100 gallons of petrol does not differ within a week among stations that are supplied by the same terminal. This can be expressed by the *roll-up dependency*:

$$D^{\texttt{WEEK}}\ L^{\texttt{VORONOI}} \rightarrow Price^{\texttt{EURO}}\ . \tag{1}$$

That is, if $(D : x_1)(L : y_1)(Price : z_1)$ and $(D : x_2)(L : y_2)(Price : z_2)$ are two price records, and $x_1$ and $x_2$ belong to the same week, and $y_1$ and $y_2$ are supplied by the same terminal, then $z_1$ and $z_2$ are the same if rounded to the nearest integral Euro. This rule is very useful, because it allows us to reduce the number of price records with a factor $\frac{7N}{5}$ (the $N$ stations are supplied by 5 terminals, and there are 7 days in a week). Such generalizations are significant in data mining [14].

**Comparison With TFD** RUDs extend temporal functional dependencies (TFDs) [19, 21] to non-temporal dimensions. TFDs only support generalization for the temporal dimension. For example, the construct of TFD can express *"The petrol price at a given location does not change within a week,"* as follows:

$$L \rightarrow_{\texttt{WEEK}} Price\ .$$

Note the special position of the time indicator WEEK. In our formalism this constraint is expressed by the RUD

$$L^{\texttt{LOCATION}}\ D^{\texttt{WEEK}} \rightarrow Price^{\texttt{EUROCENT}}\ ,$$

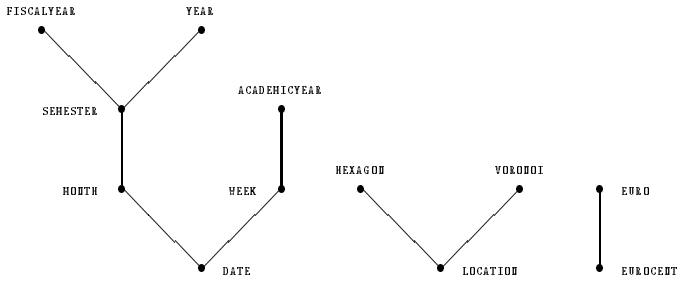



Figure 2: Partially ordered set of levels.

which can further be abbreviated (because LOCATION and EUROCENT are the domains of $L$ and $Price$ respectively) giving

$$L^{=}\ D^{\texttt{WEEK}} \rightarrow Price^{=}\ .$$

The latter formulation makes explicit that only the temporal attribute ($D$) is subject to roll-up. RUDs, unlike TFDs, allow to roll up any attribute, as illustrated in the motivating example. In particular, the RUD (1) cannot be expressed as a TFD. For a more extensive overview of temporal dependencies in databases, see [12, 19, 20].

## 2.2 Roll-Up

Figure 1 illustrates spatial and temporal granularities. Time can be partitioned into (civil) years, fiscal years, semesters, and months. We assume that a fiscal year runs from July 1 to June 30 of the next (civil) year. Semesters run from January 1 to June 30, and from July 1 to December 31. Space can be partitioned by tiling hexagons or a Voronoi diagram [7], which reflects a proximity relation to a set of given points. Months constitute a finer granularity than years, as every month is properly contained in a year; we say that every month rolls up to its year. This is denoted $\texttt{MONTH} \preceq \texttt{YEAR}$, and there is a function mapping every month to the year it belongs to. On the other hand, in Figure 1, hexagons do not divide evenly into Voronoi cells, nor vice versa. The following definition is adapted from [5].

**Definition 1** We assume the existence of a partially ordered set $(\mathcal{L}, \preceq)$ of *levels*. Every level $L$ of $\mathcal{L}$ has associated with it a set of values, denoted $ext(L)$.

A *roll-up instantiation* $U$ is a set of functions as follows: for every $L_1, L_2 \in \mathcal{L}$ with $L_1 \preceq L_2$, there is a total function, denoted $U_{L_1}^{L_2}$, from $ext(L_1)$ into $ext(L_2)$, satisfying the following conditions:

*Transitivity:* For every $L_1, L_2, L_3 \in \mathcal{L}$ with $L_1 \preceq L_2 \preceq L_3$, $U_{L_1}^{L_3} = U_{L_2}^{L_3} \circ U_{L_1}^{L_2}$, and

*Reflexivity:* For every $L \in \mathcal{L}$, $U_L^L$ is the identity on $ext(L)$.

We will write $U^L(v)$ instead of $U_{L'}^{L}(v)$ if $L'$ is clear from the context. If $U^L(v) = w$, we say that $v$ *rolls up to* $w$ in $L$, where $U$ is implicitly understood and $L$ can be omitted if it is clear from the context.

The set $(\mathcal{L}, \preceq)$ is illustrated in Figure 2. The Transitivity requirement in Definition 1 states that if month $m$ rolls up to $s$ in SEMESTER, and $s$ rolls up to $y$ in YEAR, then $m$ rolls up to $y$ in YEAR.

We are now going to introduce the notion of schema and generalization schema. The schema introduced in Section 2.1 is $(D : \texttt{DATE})(L : \texttt{LOCATION})(Price : \texttt{EUROCENT})$, which for convenience will be denoted $D^{\texttt{DATE}} L^{\texttt{LOCATION}} Price^{\texttt{EUROCENT}}$. A generalization schema is obtained from a schema by duplicating attributes, by omitting attributes, or by substituting superlevels for levels (we say that $L$ is a superlevel of $L'$ if $L' \preceq L$). For example, $D^{\texttt{MONTH}} L^{\texttt{HEXAGON}} L^{\texttt{VORONOI}}$ is a generalization schema of the schema $D^{\texttt{DATE}} L^{\texttt{LOCATION}} Price^{\texttt{EUROCENT}}$ used earlier: the attribute $Price$ has been omitted, the attribute $L$ has been duplicated, and superlevels have been substituted for the levels in the original schema.

**Definition 2** We assume the existence of a set $\mathcal{A}$ of *attributes*. A *schema* is a set $S = \{{A_1}^{L_1}, \ldots, {A_n}^{L_n}\}$ where $n \geq 0$, and $A_1, \ldots, A_n$ are pairwise distinct attributes, and $L_1, \ldots, L_n$ are (not necessarily distinct) levels. We will write $S(A) = L$ if $A^L \in S$. We also write ${A_1}^{L_1}{A_2}^{L_2} \ldots {A_n}^{L_n}$ as a shorthand for $\{{A_1}^{L_1}, {A_2}^{L_2}, \ldots, {A_n}^{L_n}\}$.

A *generalization schema* of the schema $S$ is a set $P = \{{A_{i_1}}^{L_{i_1}}, \ldots, {A_{i_m}}^{L_{i_m}}\}$ where $A_{i_1}, \ldots, A_{i_m}$ are (not necessarily distinct) attributes of $\{A_1, \ldots, A_n\}$, and $L_{i_1}, \ldots, L_{i_m}$ are levels satisfying the following condition: if $A_{i_j} = A_k$ then $L_k \preceq L_{i_j}$ ($j \in [1, m]$, $k \in [1, n]$).

Let $P$ be a generalization schema of the schema $S$. If $A^L \in P$ and $A^L \in S$, then we can substitute $A^=$ for $A^L$ in $P$.

We briefly comment on the implementation of levels and roll-up functions. Certain roll-ups will typically be stored as binary relations in a relational database. The roll-up of cities to states is an example. Other roll-ups, such as the roll-up from Eurocent to Euro, will typically be defined by a function in some programming language. Such a function can be seen as a finite representation of an infinite binary relation.

## 2.3 RUD

The generalization schema $D^{\texttt{WEEK}}L^{\texttt{VORONOI}}$ induces a partitioning of the set of tuples over the schema $D^{\texttt{DATE}}L^{\texttt{LOCATION}}Price^{\texttt{EUROCENT}}$ in the following way: two tuples belong to the same partition if their $D$-values roll up to the same week, and their $L$-values roll up to the same Voronoi cell. A RUD $P \rightarrow Q$, where $P$ and $Q$ are generalization schemas, states that whenever two tuples belong to the same $P$-partition, then they must belong to the same $Q$-partition.

**Definition 3** Let $S = \{{A_1}^{L_1}, \ldots, {A_n}^{L_n}\}$ be a schema. A *tuple* over $S$ is a set $\{(A_1 : v_1), \ldots, (A_n : v_n)\}$ where $v_i \in ext(L_i)$ for each $i \in [1, n]$. A *relation* $I$ over $S$ is a finite set of tuples over $S$.

Let $U$ be a roll-up instantiation. Let $P$ be a generalization schema of $S$. Let $t_1, t_2$ be tuples over $S$. We write $t_1 \sim_{P,U} t_2$ iff for every $A^L$ in $P$,

$$U^L(t_1(A)) = U^L(t_2(A)) \ .$$

Obviously, if $I$ is a relation over $S$, then the relation $\sim_{P,U}$ on the tuples of $I$ is an equivalence relation.

A *Roll-Up Dependency* (RUD) over $S$ is a statement $P \rightarrow Q$ where $P$ and $Q$ are generalization schemas of $S$. Given a roll-up instantiation $U$, a relation $I$ over $S$ is said to *satisfy* $P \rightarrow Q$ iff for all tuples $t_1, t_2 \in I$, if $t_1 \sim_{P,U} t_2$ then $t_1 \sim_{Q,U} t_2$.

Logical implication is defined in the classical way. Let $\Sigma$ be a set of RUDs and let $\sigma$ be a single RUD (all over the same schema $S$). Let a roll-up instantiation $U$ be given. $\Sigma$ is said to *logically imply* $\sigma$ *under* $U$, denoted $\Sigma \models^U_{RUD} \sigma$, iff for every relation $I$ over $S$, if $I$ satisfies every RUD of $\Sigma$, then $I$ satisfies $\sigma$.

$\Sigma$ is said to *logically imply* $\sigma$, denoted $\Sigma \models_{RUD} \sigma$, iff $\Sigma \models^U_{RUD} \sigma$ for every roll-up instantiation $U$.

## 2.4 Reasoning About RUDs

**Roll-Up Lattice** The set of generalization schemas of a given schema can be ordered by a binary relation, denoted $\trianglelefteq$, capturing the notion of less-general-than between generalization schemas. For example, $D^{\texttt{MONTH}}L^{\texttt{HEXAGON}}L^{\texttt{VORONOI}} \trianglelefteq D^{\texttt{YEAR}}L^{\texttt{HEXAGON}}$, because for every attribute-level pair $A^L$ in the second schema, there is a pair $A^{L'}$ in the first schema with $L' \preceq L$.

**Definition 4** Let $P$ and $Q$ be generalization schemas of the schema $S$. $P$ is said to be *less general than* $Q$, denoted $P \trianglelefteq Q$, iff for every $A^L$ in $Q$, there is some $A^{L'}$ in $P$ such that $L' \preceq L$.

The generalization schema $P$ is called *irreducible* iff whenever $P$ contains $A^L$ and $A^{L'}$ with $L \neq L'$ then $L \parallel L'$. [1]

[1] We write $L \parallel L'$ iff neither $L \preceq L'$ nor $L' \preceq L$.

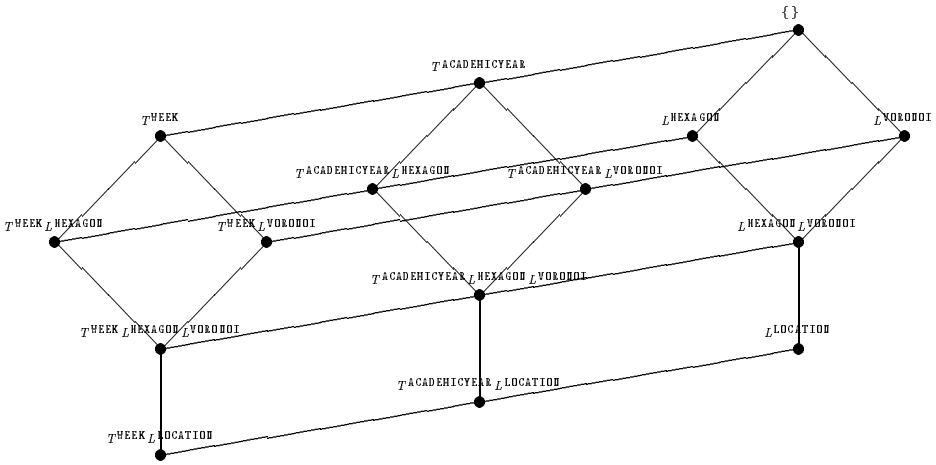



Figure 3: The family of generalization schemas of $T^{\texttt{WEEK}}L^{\texttt{LOCATION}}$ ordered by $\trianglelefteq$.

For example, $D^{\texttt{MONTH}}D^{\texttt{YEAR}}L^{\texttt{VORONOI}}$ is *not* irreducible, because $\texttt{MONTH} \preceq \texttt{YEAR}$; the same partition is defined by the irreducible generalization schema $D^{\texttt{MONTH}}L^{\texttt{VORONOI}}$. The proof of the following theorem can be found in [22]:

**Theorem 1** *Let $S$ be a schema. The set of all irreducible generalization schemas of $S$, ordered by $\trianglelefteq$, is a complete lattice.*

The set of all generalization schemas of $S$, ordered by $\trianglelefteq$, will be called the *roll-up lattice* of $S$. A roll-up lattice is shown in Figure 3. It should be stressed that $\preceq$, unlike $\trianglelefteq$, does not need to be a lattice. Our notion of roll-up lattice extends and generalizes several earlier proposals found in the literature. Our notion is more general than the one in [11], because the same attribute can appear more than once in a lattice element, as in $L^{\texttt{HEXAGON}}L^{\texttt{VORONOI}}$. This extension is both natural and useful. In an OLAP application, for example, one may want to group data by both grids simultaneously. Dimensionality reduction [8] is embedded implicitly in our roll-up lattice.

**Axiomatization** A sound and complete axiomatization for reasoning about RUDs is given next.

**Definition 5** The axioms for reasoning about RUDs are as follows ($P, Q, R$ are generalization schemas over a given schema):

$$\vdash_{RUD} \quad P \to Q \text{ if } P \trianglelefteq Q \tag{2}$$

$$P \to Q \quad \vdash_{RUD} \quad PR \to QR \tag{3}$$

$$P \to Q \text{ and } Q \to R \quad \vdash_{RUD} \quad P \to R \tag{4}$$

In [22], we proved the following result.

**Theorem 2** *Let $\Sigma$ be a set of RUDs and let $\sigma$ be a single RUD (all over the same schema).*
*$\Sigma \vdash_{RUD} \sigma$ iff $\Sigma \models_{RUD} \sigma$.*

The axioms are almost Armstrong's axioms [1]; the only difference is that axiom (2) refers to $\trianglelefteq$, whereas the corresponding Armstrong's axiom uses simple set inclusion. Following an approach stipulated in [19, page 119], we are now going to "push" the relation $\preceq$ within the RUD formalism. If $(L : \texttt{LOCATION})$ is part of the database schema, we add the RUDs $L^{\texttt{LOCATION}} \to L^{\texttt{HEXAGON}}$ and $L^{\texttt{LOCATION}} \to L^{\texttt{VORONOI}}$. By Armstrong's axioms, we can derive $L^{\texttt{LOCATION}} \to L^{\texttt{HEXAGON}}L^{\texttt{VORONOI}}$ (this is known as the Union rule for FDs). The same RUD is derived in a different way by using the axioms of Definition 5. In particular, $L^{\texttt{LOCATION}} \trianglelefteq L^{\texttt{HEXAGON}}L^{\texttt{VORONOI}}$ and hence $L^{\texttt{LOCATION}} \to L^{\texttt{HEXAGON}}L^{\texttt{VORONOI}}$ follows immediately by axiom (2). Theorem 3 generalizes the above observation.

*Has*

| *OilCompany* | *Outlet* | $L$ |
|---|---|---|
| Shell | O1 | $x = 37, y = 34$ |
| Shell | O2 | $x = 27, y = 56$ |
| Texaco | O3 | $x = 12, y = 98$ |
| ... | ... | ... |

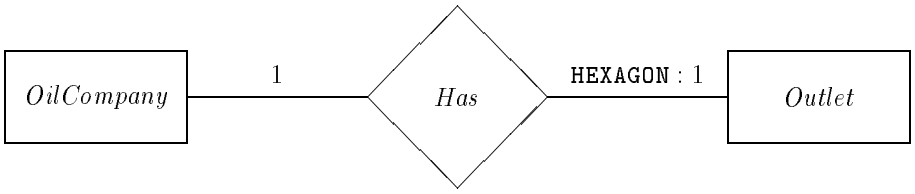



Figure 4: Extending cardinality constraints in ER-diagrams.

**Definition 6** Let $S$ be a schema. We write $S^{\mathcal{L}}$ for the smallest set of RUDs containing $A^{L_1} \rightarrow A^{L_2}$ whenever $A^L \in S$ and $L \preceq L_1 \preceq L_2$.

**Theorem 3** *Let $\Sigma$ be a set of RUDs and let $\sigma$ be a single RUD (all over the same schema $S$). $\Sigma \vdash_{RUD} \sigma$ iff $\Sigma \cup S^{\mathcal{L}} \vdash_A \sigma$, where $\vdash_A$ denotes derivability using Armstrong's axioms.*

**Proof.** Both directions can be proved by induction on the derivation of $\sigma$. □

This means that, after expressing $\preceq$ by RUDs, reasoning about RUDs can be captured by Armstrong's axioms. It should be stressed, however, that there is a clear conceptual distinction between database relations and RUDs on the one hand, and roll-up instantiations and $\preceq$ on the other hand. The RUDs of $S^{\mathcal{L}}$ express certain inherent properties of the roll-up lattice of $S$. Interestingly, along the same lines, RUDs can be used to impose additional properties on the roll-up lattice, as is shown next.

### 2.5 Adding Axioms to Capture More Meaning

Note that whenever two days fall within the same year, as well as within the same fiscal year, then these days must necessarily belong to the same semester. This is expressed by the RUD:

$$D^{\texttt{YEAR}} D^{\texttt{FISCALYEAR}} \rightarrow D^{\texttt{SEMESTER}} \quad .$$

It is important to note that the foregoing RUD is *not* implied by $S^{\mathcal{L}}$, and really imposes new constraints on the roll-up lattice. In Section 4, we are going to extend RUDs to capture more complex constraints on the roll-up lattice. But first we are going to discuss two important applications of RUDs.

## 3 Applications

### 3.1 Conceptual Modeling

There have been several proposals to extend the Entity-Relationship (ER) model to capture more temporal and spatial semantics [15]. A recent survey of temporal extensions of ER models is [9]. In [21], we use temporal functional dependencies (TFDs) to refine the cardinality construct. Tauzovich [18] distinguishes between *snapshot cardinality* and *lifetime cardinality*. In [21], we show that TFDs allow to specify cardinality constraints at any granularity level, snapshot and lifetime being two extremes. In this section, we show that RUDs permit to lift this extension to any dimension, including the spatial one. We extend the petrol scenario to illustrate our approach.

A marketing firm keeps track of the location of petrol (sales) outlets by oil company. These data may show, for example, that there is a Shell outlet at location $x = 37, y = 34$. The following cardinality constraint applies:

*"Outlets of the same oil company are spread over the country such that no two outlets of the same oil company lie within the same hexagon cell."* This cardinality constraint is difficult to model by classical ER models. A possible instance of a relationship between oil companies and outlets is shown in Figure 4 *top*. Note that an oil company can (and will) have more than one outlet, and that an outlet belongs to one oil company. Hence, a typical relationship would be one-to-many, and unable to capture the constraint mentioned above. To fix this problem, one could consider incorporating "hexagon cell" as a new entity, and introduce a ternary relationship. However, such solution tends to be problematic and far from conceptual. Our more elegant solution based on RUDs is shown in Figure 4 *bottom*. The cardinality constraint "HEXAGON : 1" states that no oil company can have two outlets *within the same hexagon cell*:

$$OilCompany^{=}\ L^{\texttt{HEXAGON}} \rightarrow Outlet^{=}\ .$$

Remark that the ER diagram shows the strongest cardinality constraint that applies. For example, from the diagram in Figure 4 it is correct to conclude that the same oil company can have two outlets within the same Voronoi cell; otherwise the diagram would (also) have shown "VORONOI : 1."

### 3.2 Data Mining

In Section 2.1, we explained that RUDs can express certain spatio-temporal regularities, which indicate good abstraction levels for summarizing data. In particular, for the schema $D^{\texttt{DATE}}L^{\texttt{LOCATION}}Price^{\texttt{EUROCENT}}$ we argue that if petrol prices do not differ much within a week and a Voronoi cell, then it makes sense to roll-up $D$-values to the level WEEK, $L$-values to VORONOI, and to summarize $Price$-values by rolling them up to EURO and/or applying a central tendency measure, like average.

The emerging data mining problem is the following. We fix the right-hand of a RUD, and then we try to find a left-hand such that the resulting RUD is satisfied by the data. In the example, we fix $Price^{\texttt{EURO}}$ because we are interested in finding price regularities; we choose $Price^{\texttt{EURO}}$ instead of $Price^{\texttt{EUROCENT}}$ because we want to abstract from minor price changes below one Euro. Every generalization schema over $D^{\texttt{DATE}}L^{\texttt{LOCATION}}$ is a candidate left-hand. In general, the number of candidates is exponential in the number of attributes; two candidates in this example are $D^{\texttt{MONTH}}L^{\texttt{HEXAGON}}$ and $D^{\texttt{WEEK}}L^{\texttt{VORONOI}}$. The data mining problem is to find good left-hand generalization schemas that determine the petrol price in Euro. (In this discussion, we assume a roll-up instantiation $U$ is fixed.)

Note that a RUD is falsified by a relation $I$ as soon as $I$ contains two tuples that contradict the RUD, even if a majority of tuples in $I$ supports the RUD. In data mining, one is generally not only interested in "exact" regularities, but also in "strong" regularities. To capture the strength of a RUD, we adapted the notion of *confidence* that is common in association rule mining [6]. The confidence of a RUD $P \rightarrow Q$ is the conditional probability that two tuples $t_1, t_2$, which are randomly selected from $I$ without replacement, satisfy $t_1 \sim_{Q,U} t_2$, given they already satisfy $t_1 \sim_{P,U} t_2$. The task then is to mine RUDs that satisfy a certain threshold confidence.

We have finished a $C^{++}$ implementation for mining RUDs. The first experiments are promising, and are reported in [24]. More details about the complexity of mining RUDs can be found in [22].

## 4 Adding Inequality

### 4.1 Introductory Examples

Recall that a fiscal year runs from July 1 to June 30. Clearly, (civil) years and fiscal years are not comparable by $\preceq$, i.e., YEAR $\|$ FISCALYEAR. Some weeks span two civil years, and some other weeks span two fiscal years. Hence, WEEK $\|$ YEAR and WEEK $\|$ FISCALYEAR. See Figure 2. Assume the schema $S = D^{\texttt{DATE}}Price^{\texttt{EURO}}$, which can be used to store a time series of prices of a particular product. Consider the following set $\Sigma$ of RUDs

$$\Sigma = \{D^{\texttt{YEAR}} \rightarrow Price^{=},\ D^{\texttt{FISCALYEAR}} \rightarrow Price^{=}\}\ ,$$

and the RUD

$$\sigma = D^{\texttt{WEEK}} \rightarrow Price^{=}\ .$$

The RUDs of $\Sigma$ state that two tuples over the schema $S$ whose $D$-values roll up to the same civil year or to the same fiscal year, must agree on $Price$. For a "real-life" roll-up instantiation $U$, we would have $\Sigma \models^{U}_{RUD} \sigma$ because two

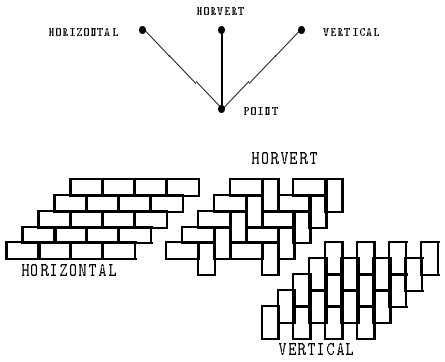



Figure 5: Tiling grids.

days of the same week cannot belong to two distinct civil years and to two distinct fiscal years at the same time. However, Definition 1 permits a roll-up instantiation $U'$ with two values $d_1, d_2 \in ext(\texttt{DATE})$ satisfying:

$$\begin{aligned} U'^{\texttt{WEEK}}(d_1) &= U'^{\texttt{WEEK}}(d_2) \ , \\ U'^{\texttt{YEAR}}(d_1) &\neq U'^{\texttt{YEAR}}(d_2) \ , \\ U'^{\texttt{FISCALYEAR}}(d_1) &\neq U'^{\texttt{FISCALYEAR}}(d_2) \ . \end{aligned}$$

That is, $d_1$ and $d_2$ roll up to the same week, but to distinct years and to distinct fiscal years. Then the relation with two tuples $\{(D : d_1)(Price : 20), (D : d_2)(Price : 40)\}$ shows that $\Sigma \not\models^{U'}_{RUD} \sigma$, and hence $\Sigma \not\models_{RUD} \sigma$. Although $U'$ does not correspond to a "real-life" calendar, it satisfies the definition of roll-up instantiation. Clearly, it is desirable to impose additional constraints so as to exclude $U'$; therefor we are going to extend RUDs with negation.

The foregoing example illustrates the concept *collectively-finer-than* of Wang et al. [19]. We say that `WEEK` is collectively-finer-than $\{\texttt{YEAR}, \texttt{FISCALYEAR}\}$ meaning that every week falls entirely within a civil year, or a fiscal year, or both. This concept turns out very important in temporal reasoning, and is not expressible in our formalism so far. We could introduce collectively-finer-than at the level of roll-up, as we did with finer-than ($\preceq$). However, it is clean to keep our definition of roll-up and extend RUDs so that collectively-finer-than can be expressed. The following rule expresses that, whenever two dates roll-up to the same week, then they must either roll up to the same year, or to the same fiscal year, or both:

$$D^{\texttt{WEEK}} \rightarrow D^{\texttt{YEAR}} \vee D^{\texttt{FISCALYEAR}} \ .$$

Using propositional calculus, we can eliminate the disjunction at the cost of introducing negation:

$$\neg D^{\texttt{FISCALYEAR}} \neg D^{\texttt{YEAR}} \rightarrow \neg D^{\texttt{WEEK}} \ .$$

The latter statement is called a RUD$^\neg$.

It is important to understand that the proposed extension is generic, and in no ways confined to time. Figure 5 shows three different ways of tiling the Euclidean plane. (While the picture only shows twenty tiles for each tiling, one has to think of each tiling as extending in all directions.) Every point of the Euclidean plane rolls up to a tile of `HORIZONTAL`, and to a tile of `HORVERT`, and to a tile of `VERTICAL`. The poset of levels (Figure 5 *top*) does not show any relationship between the three ways of tiling. By RUD$^\neg$ we can express that every tile of `HORVERT` is either a tile of `HORIZONTAL` or a tile of `VERTICAL` ($P$ is an attribute with domain `POINT`):

$$\neg P^{\texttt{HORIZONTAL}} \neg P^{\texttt{VERTICAL}} \rightarrow \neg P^{\texttt{HORVERT}} \ .$$

Clearly, not only can RUD$^\neg$ impose constraints on roll-up instantiations, but also on data in relations. An example is

$$D^{\texttt{WEEK}} \neg D^{\texttt{FISCALYEAR}} L^{\texttt{VORONOI}} \rightarrow Price^{\texttt{EUROCENT}} \ ,$$

expressing *"The petrol price remains constant within a Voronoi cell during weeks in which a new fiscal years starts."* Hence, RUD$^\neg$ constitutes an expressive formalism, combining functional dependency, roll-up, and negation.

## 4.2 RUD$^\neg$

The following definition extends RUDs by allowing a negation sign in front of any attribute.

**Definition 7** Let $S$ be a schema. If $S$ contains $A^L$ and $L \preceq L'$, then $A^{L'}$ and $\neg A^{L'}$ are literals over $S$. A *term* over $S$ is a set of literals over $S$. Let $P$ be a term over $S$. Given a roll-up instantiation $U$, two tuples $t_1$ and $t_2$ over $S$ are said to *satisfy* $P$ iff

- for every positive literal $A^L$ in $P$, $U^L(t_1(A)) = U^L(t_2(A))$, and
- for every negative literal $\neg A^L$ in $P$, $U^L(t_1(A)) \neq U^L(t_2(A))$.

A *Roll-Up Dependency With Negation* (RUD$^\neg$) over $S$ is a statement $P \rightarrow Q$ where $P$ and $Q$ are terms over $S$ such that either (i) $P$ contains a negative literal, or (ii) $Q$ does not contain a negative literal. Given a roll-up instantiation $U$, a relation $I$ over $S$ is said to *satisfy* $P \rightarrow Q$ iff for all tuples $t_1, t_2 \in I$, if $t_1$ and $t_2$ satisfy $P$, then they satisfy $Q$.

$\models_{RUD}$ is extended to $\models_{RUD^\neg}$ in the obvious way.

Note that we disallow expressions like $D^{\texttt{YEAR}} \rightarrow \neg Price^{\texttt{EURO}}$, because such an expression could not possibly be satisfied. This is because the tuples $t_1$ and $t_2$ in Definition 7 are not required to be distinct, and every individual tuple becomes a counterexample for $D^{\texttt{YEAR}} \rightarrow \neg Price^{\texttt{EURO}}$. Similar observations appear in [2, 17]. Note that every generalization schema of $S$ is a term over $S$, but terms, unlike generalization schemas, do not induce a partitioning of the tuples over $S$.

## 4.3 Reasoning About RUD$^\neg$

Armstrong's axioms are no longer complete for reasoning about RUD$^\neg$s. We now give a sound and complete axiomatization for reasoning about RUD$^\neg$s.

**Definition 8** The axioms for reasoning about RUD$^\neg$s are as follows ($P, Q, R$ are generalization schemas over a given schema; $p$ is a literal; $\neg p$ is denoted $\overline{p}$):

$$\vdash_{PC} \quad P \rightarrow Q \text{ iff } Q \subseteq P \tag{5}$$

$$P \rightarrow Q \quad \vdash_{PC} \quad PR \rightarrow QR \tag{6}$$

$$P \rightarrow Q \text{ and } Q \rightarrow R \quad \vdash_{PC} \quad P \rightarrow R \tag{7}$$

$$pP \rightarrow Q \text{ and } \overline{p}P \rightarrow Q \quad \vdash_{PC} \quad P \rightarrow Q \tag{8}$$

$$\vdash_{PC} \quad \overline{p}p \rightarrow P \tag{9}$$

Figure 6 shows a derivation for the example introduced in Section 4.1. The following theorem expresses that the axiomatization is sound and complete. It is the analog of Theorem 3, but is much harder to prove.

**Theorem 4** *Let $\Sigma$ be a set of RUD$^\neg$s, and let $\sigma$ be a single RUD$^\neg$ (all over the same schema).*
*$\Sigma \models_{RUD^\neg} \sigma$ iff $\Sigma \cup S^{\mathcal{L}} \vdash_{PC} \sigma$.*

**Proof.** From Theorem 5 and Theorem 6. See Appendix A. □

The subscript in $\vdash_{PC}$ is chosen because Appendix A also shows an equivalence between RUD$^\neg$ and positive Propositional Calculus.

| | | |
|---|---|---|
| $D^{\texttt{YEAR}} \rightarrow Price^{\texttt{EURO}}$ | given | (a) |
| $D^{\texttt{FISCALYEAR}} \rightarrow Price^{\texttt{EURO}}$ | given | (b) |
| $\neg D^{\texttt{FISCALYEAR}} \neg D^{\texttt{YEAR}} \rightarrow \neg D^{\texttt{WEEK}}$ | given | (c) |
| $D^{\texttt{FISCALYEAR}} D^{\texttt{WEEK}} \rightarrow Price^{\texttt{EURO}} D^{\texttt{WEEK}}$ | from (b) by (6) | (d) |
| $Price^{\texttt{EURO}} D^{\texttt{WEEK}} \rightarrow Price^{\texttt{EURO}}$ | by (5) | (e) |
| $D^{\texttt{FISCALYEAR}} D^{\texttt{WEEK}} \rightarrow Price^{\texttt{EURO}}$ | from (d) and (e) by (7) | (f) |
| $\neg D^{\texttt{FISCALYEAR}} \neg D^{\texttt{YEAR}} D^{\texttt{WEEK}} \rightarrow D^{\texttt{WEEK}} \neg D^{\texttt{WEEK}}$ | from (c) by (6) | (g) |
| $D^{\texttt{WEEK}} \neg D^{\texttt{WEEK}} \rightarrow D^{\texttt{YEAR}}$ | by (9) | (h) |
| $\neg D^{\texttt{FISCALYEAR}} \neg D^{\texttt{YEAR}} D^{\texttt{WEEK}} \rightarrow D^{\texttt{YEAR}}$ | from (g) and (h) by (7) | (i) |
| $\neg D^{\texttt{FISCALYEAR}} D^{\texttt{YEAR}} D^{\texttt{WEEK}} \rightarrow D^{\texttt{YEAR}}$ | by (5) | (j) |
| $\neg D^{\texttt{FISCALYEAR}} D^{\texttt{WEEK}} \rightarrow D^{\texttt{YEAR}}$ | from (i) and (j) by (8) | (k) |
| $\neg D^{\texttt{FISCALYEAR}} D^{\texttt{WEEK}} \rightarrow Price^{\texttt{EURO}}$ | from (k) and (a) by (7) | (l) |
| $D^{\texttt{WEEK}} \rightarrow Price^{\texttt{EURO}}$ | from (f) and (l) by (8) | (m) |

Figure 6: Example derivation.

# 5 Concluding Remarks

The concept of RUD combines functional dependency and roll-up. It has interesting applications in conceptual modeling and data mining. It allows to express the functional determinacies present in generalization hierarchies, but cannot express certain complex relationships between levels that have been studied for temporal databases. To this extent, RUDs have been extended with negation. The concept of RUD$^\neg$ expresses and generalizes these complex relationships for arbitrary levels, including spatial ones. A sound and complete axiomatization of RUD$^\neg$ is an interesting and important result.

# A Completeness Proof

To simplify the notations, the completeness proof exploits an equivalence between RUD$^\neg$ and positive propositional calculus. Similar equivalences have appeared in the literature [2, 13, 16, 17].

**Definition 9** Let $B$ be a set of *Boolean variables*. If $p$ is a Boolean variable, then $p$ and $\neg p$ are *literals*. For convenience, $\neg p$ can be denoted $\overline{p}$. Greek letters $\alpha$ and $\beta$ are used to denote literals. $\overline{\overline{\alpha}}$ equals $\alpha$. A set $T$ of literals is called a *valuation* iff every Boolean variable occurs exactly once in $T$. Every valuation $T$ extends uniquely to a map $\widehat{T}$ from the set of all Boolean formulas to $\{0, 1\}$ with

- $\widehat{T}(p) = 1$ if $p \in T$,
- $\widehat{T}(p) = 0$ if $\overline{p} \in T$, and
- $\widehat{T}(\varphi(p_1, \ldots, p_n)) = \varphi(\widehat{T}(p_1), \ldots, \widehat{T}(p_n))$, where $\varphi(p_1, \ldots, p_n)$ is a Boolean formula, and $\varphi(\widehat{T}(p_1), \ldots, \widehat{T}(p_n))$ is evaluated over $\{0, 1\}$ using the standard definitions of the operations $\wedge$, $\vee$, $\rightarrow$, and $\neg$ .

We say that $T$ *satisfies* $\varphi$ iff $\widehat{T}(\varphi) = 1$. A Boolean formula is *satisfiable* iff there exists a valuation $T$ satisfying $\varphi$; otherwise it is *unsatisfiable*.

A *term* is a conjunction of literals. A *Boolean rule* is a Boolean formula of the form $P \rightarrow Q$ where $P$ and $Q$ are terms. A Boolean rule $P \rightarrow Q$ is *positive* iff either (i) $P$ contains a negative literal, or (ii) $Q$ does not contain a negative literal. For convenience, sets of literals will be used for terms. That is, the set $\{\alpha_1, \ldots, \alpha_n\}$ is used for $\alpha_1 \wedge \ldots \wedge \alpha_n$. Then $P$ is satisfied by a valuation $T$ iff $P \subseteq T$. Logical implication is defined in the classical way and is denoted $\models_{PC}$.

Let $S$ be the schema under consideration. We let the set $B$ of Boolean variables coincide with $\{A^{L'} \mid A^L \in S \text{ and } L \preceq L'\}$.

**Theorem 5** $\Sigma \models_{RUD^\neg} \sigma$ *iff* $\Sigma \cup S^{\mathcal{L}} \models_{PC} \sigma$ .

**Proof.** A similar proof appears in [2]. □

**Definition 10** Let $\Sigma$ be a set of Boolean rules, and let $P$ be a term. The *closure* of $P$ w.r.t. $\Sigma$, denoted $P^+$, is the smallest term containing the literal $\alpha$ whenever $\Sigma \vdash_{PC} P \to \alpha$.

**Lemma 1** *$P \to Q$ and $P \to R \vdash_{PC} P \to QR$.*

**Proof.** $P \to PQ$ can be derived from $P \to Q$ by (6). Likewise, $PQ \to QR$ can be derived from $P \to R$ by (6). By (7), $P \to QR$. □

**Lemma 2** *Let $\Sigma$ be a set of Boolean rules. $Q \subseteq P^+$ iff $\Sigma \vdash_{PC} P \to Q$.*

**Proof.** $\Rightarrow$. Let $\alpha \in Q$. By the premise, $\alpha \in P^+$, hence $\Sigma \vdash_{PC} P \to \alpha$. By repeated application of Lemma 1, $\Sigma \vdash_{PC} P \to Q \Leftarrow$. Let $\alpha \in Q$. By (5), $\Sigma \vdash_{PC} Q \to \alpha$. By the premise and (7), $\Sigma \vdash_{PC} P \to \alpha$. Hence, $\alpha \in P^+$. □

**Lemma 3** *Let $\Sigma$ be a set of Boolean rules. Let $P$ be a set of literals. $P \subseteq P^+$.*

**Proof.** By (5), $\vdash_{PC} P \to P$. By Lemma 2, $P \subseteq P^+$. □

**Lemma 4** *Let $\Sigma$ be a set of Boolean rules. $(P^+)^+ \subseteq P^+$.*

**Proof.** Let $\alpha \in (P^+)^+$. Hence $\Sigma \vdash_{PC} P^+ \to \alpha$. We have $\Sigma \vdash_{PC} P \to P^+$ as a corollary of Lemma 2. By (7), $\Sigma \vdash_{PC} P \to \alpha$. Hence, $\alpha \in P^+$. □

**Lemma 5** *Let $P$ and $Q$ be terms. If $P$ is unsatisfiable, then $\vdash_{PC} P \to Q$.*

**Proof.** Assume $P$ unsatisfiable. Without loss of generality, $P$ contains $\overline{p}p$. By (5), $\vdash_{PC} P \to \overline{p}p$. By (9) and (7), $\vdash_{PC} P \to Q$. □

**Lemma 6** *Let $\Sigma$ be a set of Boolean rules, and let $P \to \alpha$ be a Boolean rule. If $\Sigma \nvdash_{PC} P \to \alpha$ then there exists a valuation $T$ containing $P^+$ such that $\overline{\alpha} \in T$ and $T^+ = T$.*

**Proof.** Assume $\Sigma \nvdash_{PC} P \to \alpha$. $P$ is satisfiable, or else $\vdash_{PC} P \to \alpha$ by Lemma 5, a contradiction. Assume the desired valuation $T$ does not exist; i.e., for every valuation $T$ containing $P^+$, $\alpha \in T$ or $T^+ \neq T$. $\alpha \in T$ implies $\alpha \in T^+$ by Lemma 3. Assume $T^+ \neq T$. Since $T \subseteq T^+$ by Lemma 3, $T^+$ must contain a literal (say $\beta$) not in $T$. Since every Boolean variable occurs in $T$, $T$ contains $\overline{\beta}$, and so does $T^+$ by Lemma 3. Hence, $T^+$ contains $\overline{\beta}\beta$. Hence, $\vdash_{PC} T^+ \to \alpha$ by Lemma 5, and $\alpha \in (T^+)^+$. By Lemma 4, $\alpha \in T^+$. Hence, for every valuation $T$ containing $P^+$, $\Sigma \vdash_{PC} T \to \alpha$. By repeated application of (8), $\Sigma \vdash_{PC} P^+ \to \alpha$. Hence, $\alpha \in (P^+)^+$. Hence, $\alpha \in P^+$ by Lemma 4. Consequently, $\Sigma \vdash_{PC} P \to \alpha$, a contradiction. We conclude by contradiction that $T$ exists. □

**Lemma 7** *Let $\Sigma$ be a set of Boolean rules, and let $P \to \alpha$ be a Boolean rule. If $\Sigma \models_{PC} P \to \alpha$ then $\Sigma \vdash_{PC} P \to \alpha$.*

**Proof.** Assume $\Sigma \nvdash_{PC} P \to \alpha$. We need to show $\Sigma \not\models_{PC} P \to \alpha$. By Lemma 6, there exists a valuation $T$ containing $P^+$ such that $\overline{\alpha} \in T$ and $T^+ = T$. Since $P \subseteq P^+ \subseteq T$ and $\alpha \notin T$, $T$ falsifies $P \to \alpha$. Let $R \to S$ be a Boolean rule in $\Sigma$ that is falsified by $T$. That is, $R \subseteq T$ and $S \not\subseteq T$. By (5), $\vdash_{PC} T \to R$, hence $\Sigma \vdash_{PC} T \to S$ by (7). Hence, $S \subseteq T^+$ by Lemma 2. But then $S \subseteq T$, a contradiction. We conclude by contradiction that $T$ satisfies $\Sigma$. □

**Theorem 6** *Let $\Sigma$ be a set of Boolean rules, and let $\sigma$ be a Boolean rule. $\Sigma \models_{PC} \sigma$ iff $\Sigma \vdash_{PC} \sigma$.*

**Proof.** $\Rightarrow$ *(Completeness).* Follows from Lemma 7 and Lemma 1. $\Leftarrow$ *(Soundness).* Straightforward. □

# References


[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] J. Berman and W. J. Blok. Positive boolean dependencies. *Information Processing Letters*, 27:147–150, 1988.

[3] C. Bettini, C. Dyreson, W. Evans, R. Snodgrass, and X. Wang. A glossary of time granularity concepts. In O. Etzion, S. Jajodia, and S. Sripada, editors, *Temporal Databases: Research and Practice*, number 1399 in LNCS State-of-the-art Survey, pages 406–413. Springer-Verlag, 1998.

[4] C. Bettini, X. Wang, and S. Jajodia. Testing complex temporal relationships involving multiple granularities and its application to data mining. In *Proc. ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 68–78, Montreal, Canada, June 1996. ACM Press.

[5] L. Cabibbo and R. Torlone. Querying multidimensional databases. In *Sixth Int. Workshop on Database Programming Languages*, pages 253–269, 1997.

[6] M.-S. Chen, J. Han, and P. Yu. Data mining: An overview from a database perspective. *IEEE Trans. on Knowledge and Data Engineering*, 8(6):866–883, 1996.

[7] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1987.

[8] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data Mining and Knowledge Discovery*, 1:29–53, 1997.

[9] H. Gregersen and C. S. Jensen. Temporal Entity-Relationship models—A survey. *IEEE Trans. on Knowledge and Data Engineering*, 11(3):464–497, 1999.

[10] J. Han. OLAP mining: An integration of OLAP with data mining. In *Proceedings of the 7th IFIP 2.6 Working Conference on Database Semantics (DS-7)*, pages 1–9, 1997.

[11] V. Harinarayan, A. Rajaraman, and J. Ullman. Implementing data cubes efficiently. In *Proc. ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 205–216, Montreal, Canada, 1996.

[12] C. Jensen, R. Snodgrass, and M. Soo. Extending existing dependency theory to temporal databases. *IEEE Trans. on Knowledge and Data Engineering*, 8(4):563–582, 1996.

[13] R. Khardon, H. Mannila, and D. Roth. Reasoning with examples: Propositional formulae and database dependencies. To appear, 1999.

[14] K. Koperski, J. Han, and J. Adhikary. Mining knowledge in geographical data. To appear in *Communications of the ACM*.

[15] C. Parent, S. Spaccapietra, and E. Zimanyi. Spatio-temporal information systems: a conceptual perspective. Tutorial at ER'98, 1998.

[16] Y. Sagiv, C. Delobel, D. S. Parker, Jr., and R. Fagin. An equivalence between relational database dependencies and a fragment of propositional logic. *Journal of the ACM*, 28(3):435–453, 1981.

[17] Y. Sagiv, C. Delobel, D. S. Parker, Jr., and R. Fagin. Correction to "An equivalence between relational database dependencies and a fragment of propositional logic". *Journal of the ACM*, 34(4):1016–1018, 1987.

[18] B. Tauzovich. Towards temporal extensions to the Entity-Relationship model. In *Proc. 10th. Int. Conf. on Entity-Relationship Approach*, pages 163–179. ER Institute, 1991.

[19] X. Wang, C. Bettini, A. Brodsky, and S. Jajodia. Logical design for temporal databases with multiple granularities. *ACM Trans. on Database Systems*, 22(2):115–170, 1997.

[20] J. Wijsen. Reasoning about qualitative trends in databases. *Information Systems*, 23(7):469–493, 1998.

[21] J. Wijsen. Temporal FDs on complex objects. *ACM Trans. on Database Systems*, 24(1):127–176, 1999.

[22] J. Wijsen and R. Ng. Discovering roll-up dependencies. Technical report, The University of British Columbia, Dept. of Computer Science, 1998. Also available at `http://www.uia.ua.ac.be/u/jwijsen/`.

[23] J. Wijsen and R. Ng. Temporal dependencies generalized for spatial and other dimensions. In *Proc. Int. Workshop on Spatio-Temporal Database Management (STDBM'99)*, LNCS 1678, pages 189–203. Springer, 1999.

[24] J. Wijsen, R. Ng, and T. Calders. Discovering roll-up dependencies. In *Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pages 213–222, San Diego, CA, 1999.