A model-driven approach for enforcing summarizability in multidimensional modeling

Jose-Norberto Mazón[1], Jens Lechtenbörger[2], and Juan Trujillo[1]

[1] Lucentia, Dept. of Software and Computing Systems, University of Alicante, Spain
{jnmazon,jtrujillo}@dlsi.ua.es
[2] Dept. of Information Systems, University of Münster, Germany
lechten@wi.uni-muenster.de

Abstract. The development of a data warehouse system is based on a conceptual multidimensional model, which provides a high level of abstraction in the accurate and expressive description of real-world situations. Once this model has been designed, the corresponding logical representation must be obtained as the basis of the implementation of the data warehouse according to one specific technology. However, there is a semantic gap between the dimension hierarchies modeled in a conceptual multidimensional model and its implementation. This gap particularly complicates a suitable treatment of summarizability issues, which may in turn lead to erroneous results from business intelligence tools. Therefore, it is crucial not only to capture adequate dimension hierarchies in the conceptual multidimensional model of the data warehouse, but also to correctly transform these multidimensional structures in a summarizability-compliant representation. A model-driven normalization process is therefore defined in this paper to address this summarizability-aware transformation of the dimension hierarchies in rich conceptual models.

1 Introduction

Business intelligence tools, such as OLAP (On-Line Analytical Processing) tools depend on the multidimensional (MD) structures of a data warehouse that allow analysts to explore, navigate, and aggregate information at different levels of detail to support the decision making process. Current approaches for data warehouse design advocate to start the development by defining a conceptual model in order to describe real-world situations by using MD structures [13]. These structures contain two main elements. On one hand, dimensions which specify different ways the data can be viewed, aggregated, and sorted (e.g., according to time, store, customer, product, etc.). On the other hand, events of interest for an analyst (e.g., sales of products, treatments of patients, duration of processes, etc.) are represented as facts which are described in terms of a set of measures. Every fact is based on a set of dimensions that determine the granularity adopted for representing the fact's measures. Dimensions, in turn, are organized as hierarchies of levels that allow analysts to aggregate data at

different levels of detail. Importantly, dimension hierarchies must ensure summarizability, which refers to the possibility of accurately computing aggregate values with a coarser level of detail from values with a finer level of detail. If summarizability is violated, then incorrect results can be derived in business intelligence tools, and therefore erroneous analysis decisions [3, 4]. In addition, summarizability is a necessary precondition for performance optimizations based on pre-aggregation [10].

Usually, summarizability are not solved at the conceptual level, but at late stages of the development by using information from the data contained in the implemented data warehouse [11, 9]. This way of proceeding poses problems to data warehouse designers, since great efforts are required to ensure summarizability due to the huge amount of data stored. However, if one tries to ensure summarizability at the conceptual level, one typically obtains MD models that are more difficult to understand as they contain an excessive amount of detail that is not required in the initial design steps. As understandability must be one inherent property of conceptual MD models, designers should be able to specify rich conceptual models without being concerned with summarizability problems. Then, a normalization process should be applied to transform the designed MD model into a constrained conceptual model, which is restricted to those MD structures that do not violate summarizability. This normalized model should also provide a high level of expressiveness in describing real-world situations.

Bearing these considerations in mind, in this paper, we take advantage of model-driven development to propose a comprehensive normalization process by using widely adopted formalisms. In essence, we describe how to (i) design different kinds of dimension hierarchies in a conceptual model in order to easily and understandably represent real-world situations regardless of summarizability problems, and (ii) automatically derive equivalent normalized conceptual models, which are constrained to those kind of dimension hierarchies that do not violate summarizability. From this normalized MD model, an implementation that satisfies summarizability can be easily deployed in any database platform and can be accurately queried by any business intelligence tool.

The rest of this paper is structured as follows. Related work is described in Section 2. Our normalization process for ensuring summarizability in dimension hierarchies is presented in Section 3. Finally, conclusions and potential for future work are provided in Section 4.

2 Related Work

In [9] we present a survey on summarizability issues in MD modeling. Briefly, MD modeling aims to represent measurable facts for real-world events under different perspectives of detail, which are specified via dimensions. For example, using the UML profile for MD modeling proposed in [5], Fig. 1 represents *Sales* via a *Fact* (▦) class and *Date*, *Product*, and *Customer* via *Dimension* (⍁) classes. Facts are composed of measures or fact attributes, which are represented as attributes with the *FactAttribute* stereotype (**FA**) such as *Price* and *Quantity* in Fig. 1. Moreover,

dimension levels, which allow to analyze measures at a specific level of detail, are specified by *Base* classes (B) such as *Day* and *Week*. Associations (represented by the stereotype *Rolls-UpTo*, ◎) between pairs of *Base* classes form dimension hierarchies. Importantly, *Rolls-UpTo* associations between *Base* classes as well as associations between *Fact* and *Dimension* classes are annotated with UML multiplicities. E.g., the multiplicities for the association between *Country* and *Region* in Fig. 1 indicate that every region belongs to exactly one country ("1" in role *r* at the *Country* end) whereas there are countries (such as "Andorra", "Monaco", etc.) without associated regions ("0..*" in role *d* at the *Region* end). Finally, UML generalization relationships between *Base* classes can be used to represent optional dimension levels within a hierarchy.

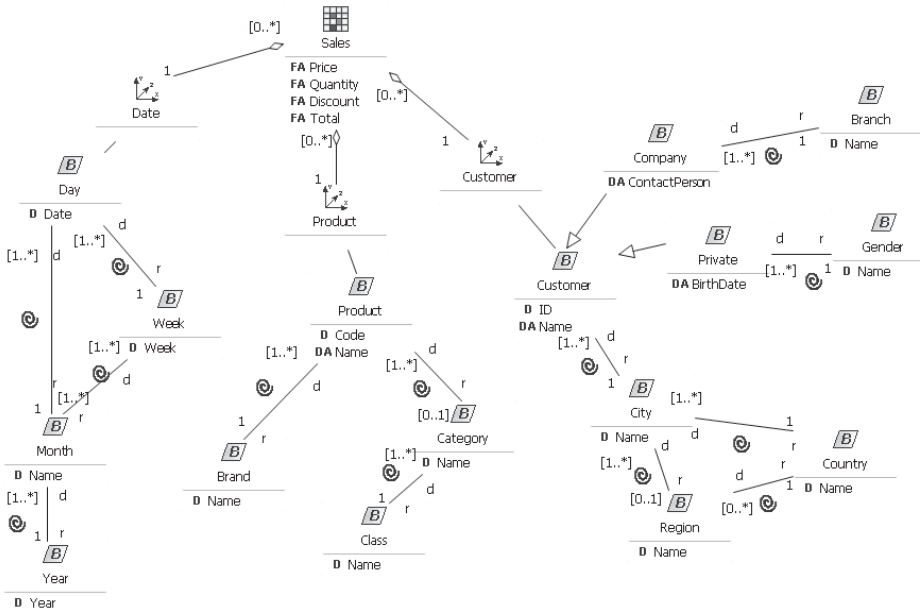


Fig. 1. Sample scenario.

In the context of MD data, summarizability presents a guarantee concerning the correctness of aggregated measures. The notion of summarizability was originally introduced in [12] and later characterized in [4]. As observed in [9], minimum and maximum multiplicities of associations characterize sub-structures where summarizability is violated. In earlier work [8], we have shown how to normalize MD schemata with non-summarizable fact-dimension associations into summarizable ones. With respect to non-summarizable *Rolls-UpTo* associations, however, a similar approach still needs to be designed.

Indeed, in a fundamental work Pedersen et al. [10, 11] propose instance level algorithms to automatically transform dimension hierarchies to achieve summarizability. As this proposal works at the instance level, it is necessary to transform the data that will populate the DW, which may involve considerable efforts of preprocessing. In particular, ETL processes become more complex, as summarizability checks must be incorporated and executed for every update. In addition, as the data transformations produce artificial data values, data analysis becomes more complex.

In [6, 7] the authors present a classification of different kinds of complex dimension *hierarchies*, and they define the MultiDimER model for the conceptual design of complex MD models based on an extension of the well-known Entity-Relationship (ER) model. The idea is that this classification guides developers to properly capture at a conceptual level the precise semantics of different kinds of hierarchies without being limited by current data analysis tools. Furthermore, the authors discuss how to map these conceptual hierarchies to the relational model (enabling implementation in commercial tools). Unfortunately, the mapping between the conceptual and the logical level is described informally. In addition, the commented mapping is tool-dependent and it may vary depending on the scenario.

3 Model-driven development of dimension hierarchies

While a conceptual MD model should represent the real world in a way that is easy to understand and that supports discussions with end users, a logical MD model must support the implementation in a chosen target technology in such a way that data can be accurately analyzed. To bridge this semantic gap, we propose a model-driven normalization process to derive an intermediate model, which should still be expressed at the conceptual level (to avoid limitations imposed by particular target models) but where MD structures that violate summarizability conditions are replaced with summarizable alternatives (which are typically more precise as we will see below). In particular, we address the three types of non-summarizable *Rolls-UpTo* associations enumerated in [9]: Non-strict, roll-up incomplete, drill-down incomplete

Non-strict associations are those where the maximum multiplicity at role r is $*$ (instead of 1). For example, the association between *Week* and *Month* in Fig. 1 is non-strict, and requires special care to avoid the well-known double counting problem. Next, an association is *drill-down incomplete* if the minimum multiplicity at role d is 0; otherwise, it is drill-down complete. For example, the association between *Country* and *Region* in Fig. 1 is drill-down incomplete as there are countries (such as "Andorra", "Monaco", etc.) without associated regions. As explained in [9] in this case one has to be careful when drilling down from aggregate values at the level *Country* towards the level *Region* as values for countries without regions may not be accounted for, leading to inconsistent grand totals. Finally, an association is *roll-up incomplete* if the minimum multiplicity at role r is 0. For example, the association between *Product* and *Category* in

Fig. 1 is roll-up incomplete. As explained in [9] in this case one has to be careful when rolling up from aggregate values at the level *Product* towards the level *Category* as values for products that are not assigned to any category will not be accounted for, leading again to inconsistent grand totals.

In the following subsections, we show how to carry out a normalization process to obtain a conceptual MD model that ensures summarizability while accurately capturing the expressiveness of the demanded real-world situation. This process is composed of a set of transformations to obtain a normalized MD model constrained to hierarchies that contain only those elements and relationships that do not violate summarizability. For the sake of understanding, these transformations are first defined informally and only two of them are formally described in QVT (Query/View/Transformation) due to space constraints.

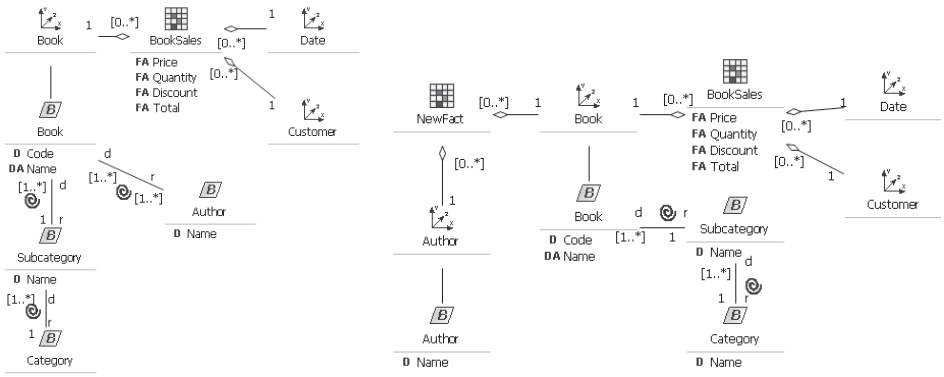
3.1 Eliminating Non-Strictness

Non-strictness is eliminated by one transformation which is based on one QVT relation (named as *nonStrictBases*) that replaces all occurrences of non-strict *Rolls-UpTo* associations in the source model with constructs that do not violate summarizability. The rationale behind this QVT relation is as follows. A non-strict association, i.e., a many-to-many relationship among dimension instances, represents the least restrictive possible case. In fact, even if no association is declared among *Base* classes then adding a non-strict (and incomplete) association does not impose any additional restriction on the instances. E.g., in Fig. 1 every *Product* has a *Brand* and may have a *Category*. As the relationship among *Brand* and *Category* is not declared explicitly, we may reasonably assume to find many *Brands* associated with the *Products* for each *Category* and vice versa. Hence, we may safely add a non-strict association from *Category* to *Brand* without changing the meaning of the model.

More generally, consider a non-strict association between *Base* classes b_1 and b_2. On the one hand, this association is redundant and can be removed safely if there are alternative strict *Rolls-UpTo* associations to b_2 (which is the *Base* class that causes the non-strictness). On the other, the association cannot be removed if the *Base* class b_2 does not play role r in some strict association. In this latter case, removing the association would isolate b_1 from b_2. Moreover, in this case the instances of b_2 are in a many-to-many relationship not only with b_1 but also with all *Base* classes that roll-up from (possibly transitively) b_1. This many-to-many relation is naturally expressed by moving b_2 into a newly created dimension, which again leads to the removal of the non-strict association.

As an example for a many-to-many relation that should be represented via a newly created dimension, assume that *Products* in the scenario of Fig. 1 are books for which the authors are of interest. Then, we may add the *Base* class *Author* along with a non-strict association from *Book* to *Author* (see Fig. 2(a)).

Although such an association allows to represent the relationship between books and their authors, this relationship is unrelated to the sales of individual books. Besides, computing sales volumes per author based on that relationship may easily lead to summarizability problems due to double counting. Hence, our



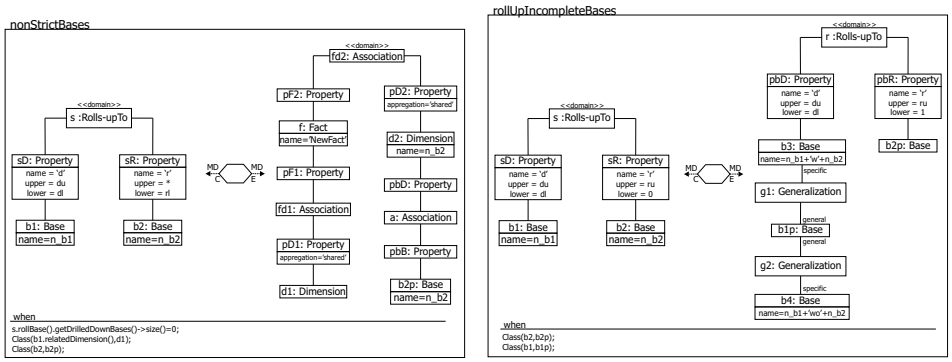
(a) Excerpt of the source model. (b) Excerpt of the target model.

Fig. 2. Newly created dimension for eliminating non-strictness: the Book sales example.

normalization process removes *Author* from the *Fact* class *Sales* and transforms the non-strict association into the natural many-to-many relationship of a newly created fact, in this example into a two-dimensional *Fact* class *NewFact* with *Dimension* classes *Book* and *Author*. As *Dimension* class *Book* is common to both facts, queries involving *Sales* for *Authors* are still possible via drill-across operations. Moreover, to support summarizable queries, the designer may want to manually add additional measures, e.g., each author's share per book, to the new fact schema. The target model is shown in Fig. 2(b).

The *nonStrictBases* relation is shown in Fig. 3(a). It checks that there is a non-strict *Rolls-UpTo* association between two *Base* classes (b_1 and b_2) in the source model (multiplicity $*$ in the role r) in order to create the appropriate elements in the target model to obtain a representation without non-strict associations. The two cases concerning the treatment of non-strict associations explained above are captured by a condition that must hold to execute this relation (see *when* clause of Fig. 3(a)). This condition checks whether *Base* class b_2 plays role r in some strict association in the source model. On the one hand, if this condition does not hold then the QVT relation is not launched, and *no* new *Rolls-UpTo* association is created in the target model, since the two *Base* classes are related via a many-to-many relationship by default. On the other, if the condition is satisfied then b_2 does not play role r in some strict association. In this case, a new *Fact* class f is created which is associated with two new *Dimension* classes: d_1 and d_2. Specifically, d_1 corresponds to the *Dimension* class related to the *Base* classes b_1 and b_2 of the source model, while d_2 is a new *Dimension* class whose defining *Base* class is b_2p (which is a copy of the *Base* class that causes the non-strictness in the source model b_2).

After the execution of the *nonStrictBases* relation, an excerpt of the target model related to the *Date* dimension is shown in Fig. 4(a). Here, the non-strict *Rolls-UpTo* association between *Month* and *Week* has disappeared, since in this



(a) Eliminating non-strictness. (b) Eliminating roll up incompleteness.

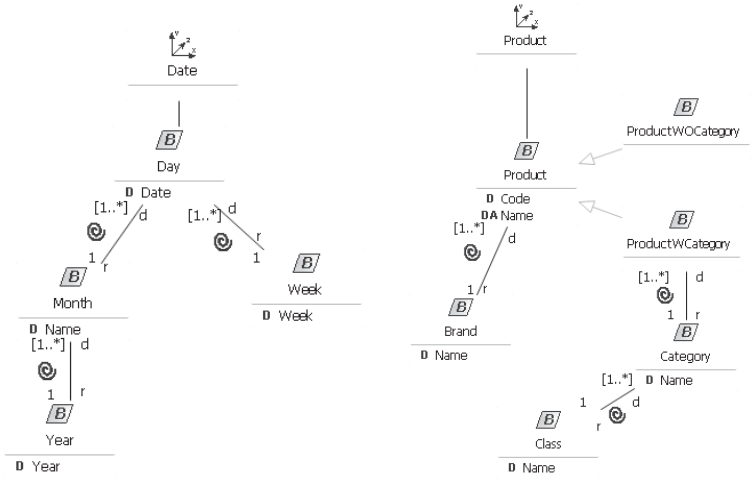
Fig. 3. QVT relations for normalizing hierarchies.

way both *Bases* classes are already many-to-many related, and there is no need for the creation of a new *Fact* class. The remaining parts of the source model that do not present non-strictness problems are copied directly to the target model. It is worth noting that all the possible strict *Rolls-UpTo* association between *Base* classes are assumed to be explicitly modeled in the source model and they will also appear in the target model. In this way, analysis capabilities of the source model are not negatively affected. Anyway, if users would still like to navigate between aggregate values by means of a non-strict relationship, then a new type of association could be added to the conceptual model; such "navigational" arcs, however, would need special support from implementing OLAP tools and are beyond the scope of this paper.

With respect to the soundness of the *nonStrictBases* relation shown in Fig. 3(a) we focus on the non-strict association of the source model, say association s between b_1 and b_2, which is represented differently in the target model. We consider two cases: First, the QVT relation just removes the non-strict association. In this case, the *when* clause of the relation assures that b_2 is alternatively reachable via strict *Rolls-UpTo* associations. Hence, in the target model both *Base* classes still belong to a common dimension hierarchy, where all pairs of *Base* classes are related via many-to-many relationships by default as we have explained above, and there is no loss of information when removing the non-strict association. Second, the QVT relation creates a new *Fact* class to represent the many-to-many relationship. In this case, the associations between instances of b_1 and b_2 of the source model are simply stored as instances of the new *Fact* class in the target model. Hence, the information that can be represented under the source model can also be represented under the target model.

3.2 Eliminating Roll-up Incompleteness

In line with the general analysis concerning optional properties for conceptual design in [1] we argue that multiplicities of 0 should be avoided in *understandable*



(a) Normalized *Date* dimension. (b) Normalized *Product* dimension.

Fig. 4. Examples of normalized hierarchies.

conceptual models whenever possible. Instead, we advocate to apply generalization for conceptual MD modeling as well. To eliminate roll-up incompleteness, the transformation is based on one QVT relation that replaces all occurrences of roll-up incomplete *Rolls-UpTo* associations in the source model with generalization constructs. In this case, the optional *Base* class (which has multiplicity 0 at the role r) should be associated with a suitable sub-class in a generalization between *Base* classes: one to reflect instances with the optional property and the other one to reflect instances without that property.

The corresponding QVT relation (*rollUpIncompleteBases*) is shown in Fig. 3(b). It checks roll-up incompleteness in the *Rolls-UpTo* association between *Base* classes in the source model. Specifically, if a 0 multiplicity is detected in the role r of a *Rolls-UpTo* association s between two *Base* classes b_1 (e.g., *Product* in Fig. 1) and b_2 (e.g., *Category* in Fig. 1), then the relation enforces the creation of new elements in the target model as follows: Two new *Base* classes b_1p and b_2p that correspond to the source *Base* classes b_1 and b_2, respectively. In addition, two new *Base* sub-classes of b_1p, namely b_3 and b_4, are created via new generalization relationships g_1 and g_2. Here, b_3 reflects the instances of its super-class b_1p that are associated with some instance of the optional *Base* class b_2p, and b_4 reflects the remaining instances of b_1p. Furthermore, the roll-up incomplete association s between b_1 and b_2 is replaced with a roll-up complete association r between b_3 and b_2p.

After the execution of the *rollUpIncompleteBases* relation, an excerpt of the target model related to the *Product* dimension is shown in Fig. 4(b). Here, two new *Base* classes, *ProductWCategory* and *ProductWOCategory*, are created to reflect those products that belong to a category or not, respectively. Again, those

parts of the source model that do not present roll-up incompleteness problems are copied directly to the target model.

With respect to the soundness of the *rollUpIncompleteBases* relation, we focus on the roll-up incomplete association of the source model, say association s between b_1 in the role d and b_2 in the role r, which is represented differently in the target model. First, b_1 and b_2 are still present in the target model (as b_1p and b_2p). Moreover, if an instance of b_1 is not related to any instance of b_2 (which exemplifies the incompleteness of the association) then this instance is simply stored as an instance of the same class under the target model (i.e. b_1p) and as an instance of its subclass b_4. If, however, an instance i_1 of b_1 is related to some instance i_2 of b_2 in the source model, then in the target model i_1 will be an instance of b_1p *and* simultaneously an instance of b_1p's sub-class b_3. Now, this sub-class b_3 has a roll-up complete association with b_2p, which allows to retrieve the associated instance i_2 in the target model. Hence, the information that can be represented under the source model can also be represented under the target model.

3.3 Eliminating Drill-down Incompleteness

The transformation that eliminates drill-down incompleteness is based on one QVT relation that replaces all occurrences of drill-down incomplete *Rolls-UpTo* associations with normalized elements Due to space constraints this transformation is not shown, but the rationale of this QVT relation is removing summarizability problems by using generalization constructs. In this way, the optional *Base* class (which has multiplicity 0 at the role d, e.g., *Region* in Fig. 1) should be associated with a sub-class in a generalization between *Base* classes.

4 Conclusions and Future Work

Data warehouse designers have to make a great effort in defining dimension hierarchies that accurately reflect real-world situations in a MD model, whilst summarizability problems are avoided. In this paper, we have described a normalization approach for ensuring that the implemented MD model will be queried without the summarizability problems derived from the dimension hierarchies. Following our approach, designers can define dimension hierarchies that violate summarizability conditions in a conceptual model by using our UML profile. This conceptual model reflects real-world situations in an understandable way. Later, several transformations can be applied to automatically obtain a normalized MD model whose dimension hierarchies do not allow situations that attempt against summarizability, thus avoiding erroneous analysis of data.

Finally, we remark that the multidimensional normal forms defined in [2], which formalize quality aspects of MD models, deal with schemata with an expressiveness that is similar to the one of our normalized models. However, so far there is no work that considers the definition of such normal forms for semantically rich conceptual models. As an avenue for future work it appears

attractive to define normal forms at the level of our normalized model: The generalization constructs included in this level enable a simplified treatment of optional levels (in [2], generalization is "simulated" by a careful application of context dependencies).

Our planned future work consists of evaluating and giving mechanisms to deal with the notion of "type compatibility" for summarizability of [4], which checks that the statistical function associated with the measure is summarizable according to the type of the measure and the type of the related dimensions.

Acknowledgments. This work has been partially supported by the following projects: SERENIDAD (PEII-11-0327-7035) from Junta de Comunidades de Castilla-La Mancha (Spain) and by the MESOLAP (TIN2010-14860) project from the Spanish Ministry of Education and Science.

References

1. Bodart, F., Patel, A., Sim, M., Weber, R.: Should optional properties be used in conceptual modelling? a theory and three empirical tests. Info. Sys. Research 12(4), 384–405 (2001)
2. Lechtenbörger, J., Vossen, G.: Multidimensional normal forms for data warehouse design. Inf. Syst. 28(5), 415–434 (2003)
3. Lehner, W., Albrecht, J., Wedekind, H.: Normal forms for multidimensional databases. In: Rafanelli, M., Jarke, M. (eds.) SSDBM. pp. 63–72. IEEE Computer Society (1998)
4. Lenz, H.J., Shoshani, A.: Summarizability in OLAP and statistical data bases. In: Ioannidis, Y.E., Hansen, D.M. (eds.) SSDBM. pp. 132–143. IEEE Computer Society (1997)
5. Luján-Mora, S., Trujillo, J., Song, I.Y.: A UML profile for multidimensional modeling in data warehouses. Data Knowl. Eng. 59(3), 725–769 (2006)
6. Malinowski, E., Zimányi, E.: Hierarchies in a multidimensional model: From conceptual modeling to logical representation. Data Knowl. Eng. 59(2), 348–377 (2006)
7. Malinowski, E., Zimányi, E.: Advanced data warehouse design: From conventional to spatial and temporal applications. Springer-Verlag (2008)
8. Mazón, J.N., Lechtenbörger, J., Trujillo, J.: Solving summarizability problems in fact-dimension relationships for multidimensional models. In: Song, I.Y., Abelló, A. (eds.) DOLAP. pp. 57–64. ACM (2008)
9. Mazón, J.N., Lechtenbörger, J., Trujillo, J.: A survey on summarizability issues in multidimensional modeling. Data Knowl. Eng. 68(12), 1452–1469 (2009)
10. Pedersen, T.B., Jensen, C.S., Dyreson, C.E.: Extending practical pre-aggregation in on-line analytical processing. In: VLDB. pp. 663–674 (1999)
11. Pedersen, T.B., Jensen, C.S., Dyreson, C.E.: A foundation for capturing and querying complex multidimensional data. Inf. Syst. 26(5), 383–423 (2001)
12. Rafanelli, M., Shoshani, A.: STORM: A statistical object representation model. In: Michalewicz, Z. (ed.) SSDBM. Lecture Notes in Computer Science, vol. 420, pp. 14–29. Springer (1990)
13. Rizzi, S., Abelló, A., Lechtenbörger, J., Trujillo, J.: Research in data warehouse modeling and design: dead or alive? In: Song, I.Y., Vassiliadis, P. (eds.) DOLAP. pp. 3–10. ACM (2006)