Extended Cascaded Star Schema and ECOLAP Operations for Spatial Data Warehouse

Marcin Gorawski

Institute of Computer Science, Silesian University of Technology,
Akademicka 16, 44-100 Gliwice, Poland
Marcin.Gorawski@polsl.pl

Abstract. In this paper several new aspects of spatial data warehouse modeling are presented. The extended cascaded star schema in spatial telemetric data warehouse SDW(t) was defined. Research proven that there is a strong need for building many SDW's extended cascaded star schemas as an outcome of separate spatio-temporal conceptual models. For one of these new data schemas, the definitions of cascaded ECOLAP operations were presented. These operations base on a relation algebra, and make possible ad-hoc queries executing.

1 Introduction

The designing of every data warehouse begins with creation of logical model that in an effective way presents described phenomenon. The rapid development in the field of the data warehouse is the cause for more sophisticated concepts to be modeled. Particularly, the Spatial Data Warehouses (SDW) store multidimensional data (data hyper-dimensionality) and describe processes that consists of multitheme [1,5,13]. Hence exists a necessity to defining new logic schemas of SDW and models of on-line analytical processing OLAP, both for spatial and flat data [12]. Significant research in the area of SDW is due to [2,6,7,8,9,10,11,12]. Previous works in the field of data modeling have contributed to creation of few, commonly used schemas. The star schema is a logical structure that consists of one, containing a huge amount of data, fact table and few smaller dimension tables. Between all dimension tables and the fact table the referential integrity is defined. Dimension tables are denormalized, however, the fact table is normalized. The advantages of this schema are: simple structure and high efficiency in query execution, that results from a low number of defined connections. One of the disadvantages of this schema is long data loading time, that results from tables denormalization. Another problem is that the star schema can be used only for data related to a single theme.

In the snowflake schema the dimension tables are not denormalized. As a result of separation of dimension tables, the hierarchy is explicitly highlighted. Thanks to table normalization, in this model there is no data redundancy, and the schema is easy to modify. An Additional virtue of normalization is faster data loading, but it has an influence on the answer time, which is longer than in a star schema. In some cases there is a need that few fact tables share the same dimension. That is where the fact constellation is used. This model is used very rarely, since there are not many concepts which can be modeled with that schema.

E. Corchado and H. Yin (Eds.): IDEAL 2009, LNCS 5788, pp. 251–259, 2009.

Logical structures of OLAP models described above are not able to optimally model spatial data with regard for theirs hyper-dimensional and multitheme. The mentioned schemas are suitable for representing data which are single-process oriented and the associated OLAP operations are processed along the dimension hierarchy. According to this, using a star or a snowflake schema, presenting a single process, causes that a complex query in SDW relating to many parallel processes described with different dimension levels, must be defined as several separate queries, which results are combined by the user.

Section 2 contains the formal definition of the new data model: expanded cascaded star, and in Section 3 the definition of OLAP operations for the new schema is presented. Section 4 summarizes the paper.

2 Formalization of the Extended Cascaded Star Schema

In the paper [13] the new formalized logical ROLAP model called Cascaded Star has been presented, which allows modeling hyper-dimensional data and multithematic decision support processes. The Cascaded Star S^C is characterized by this, that some of its dimensions form separate nested single star schemas or cascaded stars. Even though the cascaded star schema has extended the capabilities of modeling logic schemas for SDW, there are still situations in which this model is not sufficient for describing spatial, multithematic phenomenon. An example of such system is a spatial telemetric data warehouse SDW(t), which the APAS Group is currently working on [5]. Only one example of the SDW(t) schema, for which the cascaded star schema is insufficient, has been shown [4].

In the paper [13] the concept of a single star cube S^1 was defined incorrectly because of: (a) ambiguity of the terminology (cube) and used definitions (dimension tables, fact tables), (b) lack of conceptual completeness of the schema – the influence of data granulation is not taken under consideration. In the new Definition 1 of the single star schema these formal deficiencies were eliminated.

Definition 1. Single star
The single star schema is defined as $S^S = (D, F)$, where:

1. D is a set of flat dimensions tables D_i. Every dimension table $D_i \in D$ contains $[A_i, IK_i, G_i]$ where:
   - A_i is a set of attributes of D_i,
   - IK_i is a set of surrogate, numerical key identifying dimension D_i,
   - G_i is a set of data granulations level of dimension D_i.
2. $F = [V_S, PK_S, G_S]$ is the fact table, where:
   - PK_S is the primary key of S^S, $PK_S = \{IK_i\}$, where IK_i is a key that identifies $D_i \mid D_i \in D$,
   - V_S is a set of facts, $V_S = \mathrm{f}(G_S)$, where f is a function, that for a given granulation level $G_S \mid G_S \in \{G_i\}$, generates the values of numerical facts V_S.

The disadvantages of the star schema mentioned above can be eliminated by using the snowflake schema. The main change, which was introduced in the snowflake schema is: a) the departure from the full dimension tables denormalization towards their normalization – explicit presentation of the real, hierarchical structure of the dimensions, b) the facts tables store only one level of the aggregation, c) the maintenance of the relations between attributes of the same dimension with multiplicity of 1:1, 1:N, N:1 and N:M. Depending on dimensions table's degree of normalization, four types of the snowflake schema can be distinguished (Def.2).

Definition 2. Snowflake
The snowflake schema is defined as $S^F = (H, R, F)$, where:

1. H is a set of hierarchical dimensions, where for the relation R of type:
   - {1:1, 1:N, N:1} every hierarchical dimension $H_i \in H$ has a determined degree L_i, defined as a number of levels of the hierarchy. H_i is a set of tables creating a normalized dimension with a minimal cardinality L_i. Table $h_j \in H_i$ contains $[A_{h_j}, PK_{h_j}, IK_{h_j}]$, where A_{h_j} is a set of attributes h_j, PK_{h_j} is the primary key of h_j, and IK_{h_j} is a set of foreign keys migrating from tables h_{j+1}, ..., h_{L_i}. Set IK_{h_j} is empty on the highest level of the hierarchy and for the one-level hierarchy (flat dimension),
   - {N:M} set R is a set of tables of relations $\{r_{ik\_}\, H_{i/ik+r}\}$ between attributes of one dimension $H_i \mid H_i \in$ H, such that $r_{ik\_}H_{i/ik+r}$, contains combinations of identifiers of the attributes of tables h_{jk}, being in relation in the dimension H_i with multiplicity N:M
2. $F = [V_{SF}, PK_H]$ is a fact table, where V_{SF} is a set of measurements, and PK_H is a set of primary keys $\{PK_i\}$ of the dimensions H_i.

As mentioned before, the current analysis of the research and the results of experiments on the cascaded star schema for the Spatio-Temporal Telemetric Data Warehouse System [4,5] justify the creation of a new SDW schema called Extended Cascaded Star $ES^C_{\{P\text{-}t\}}$, where P is schema's degree described by the type and kind of the substars and by their multi-version, and t is a number of themes simultaneously described. The most recent experiential logic model of SDW(t) system represents the extended star of II-nd degree schema $ES^C_{\{II\text{-}1\}}$ $SDW(t)$ oriented on measurement (Fig. 1). The presented schema $ES^C_{\{II\text{-}1\}}SDW(t)$ is the S^C (SDW(t)) schema extended of two components. The first expansion of S^C (SDW(t)) schema concerns the using of the hierarchical dimensions. According to definition in [13], the S^C schema allows only the existence of flat dimensions and cascaded dimensions. Hierarchical dimensions, described in def. 3, are used in the snowflake schema. In the case of the $ES^C_{\{II\text{-}1\}}SDW(t)$ schema the hierarchical dimensions are: time dimension (e.g. measure time, meter installation time, map expiry date, etc) and the dimension defining the clients address. The second expansion of S^C (SDW(t)) schema concerns the way of linking fact tables with cascaded dimensions. In the cascaded star the key of the star and the function *g,* connected with this key, is used according to definition presented by Yu et al. [13]. Yet, in order to avoid complications connected with the definition of the *g* function, surrogate keys in fact tables are generated. For every record in the fact table a consecutive number is generated.

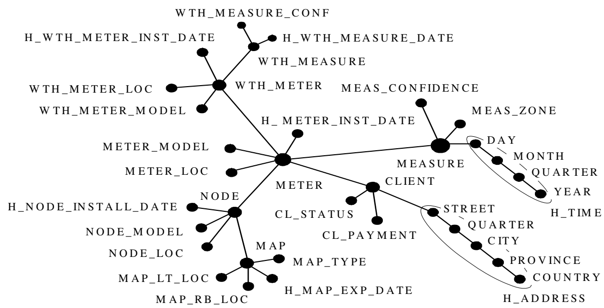

Fig. 1. $ES^C_{\{II\text{-}1\}}SDW(t)$ schema oriented on measurement

The disadvantage of such approach is the necessity of defining an additional column for storing the key in the fact table (this disadvantage concerns also the approach that uses the star's key). However, using the surrogate primary keys allows to avoid problems connected with defining the *g* function. Most of the recent relational database systems allow to define sequences. The use of the sequences enables the automatic numbering of the table rows. In Definition 3, the schema of the extended cascaded star of II-nd degree, single-theme (t=1), is described.

Definition 3. Extended Cascaded Star of II-nd degree for *t*=1
The schema of the extended cascaded star of II-nd degree for t=1 $ES^C_{\{II\text{-}1\}}$ is defined as $ES^C_{\{II\text{-}1\}} = (SS, H^C, T^C, L^C)$, where:

1. SS is a set of cascaded dimensions – data schemas, such that $S_i \in SS$ is: a) a single star schema S^S, or b) a snowflake schema S^F, or c) $ES^C_{\{II\text{-}1\}}$ schema.
2. H^C is a set of hierachical dimensions of ES^C. Every hierarchical dimension $H_i \in H^C$ has a given degree L_i defined as a number of hierachy levels in the S^F schema.
3. $T^C = [V^C, PK_{SR}, PK_H]$ is the cascaded fact table, where V^C is a set of central fact *measures*, $V^C = \{V_{SF} \cup V_S\}$. $PK_{SR} = \{SR_i\}$ is a set of surrogate keys for cascaded dimensions $S_i \mid S_i \in SS$, and $PK_H = \{PK_i\}$ is a set of primary keys of hierarchical dimensions $H_i \mid H_i \in H^C$. The primary key PK_i of dimension H_i is the primary key of table h_0.

The $ES^C_{\{II\text{-}1\}}$ schema can contain nested star schemas as well as nested snowflake schemas. This expansion allows to carry out the normalization of dimension tables.

3 ECOLAP Operations for Extended Cascaded Star

Using the new structure $ES^C_{\{P\text{-}t\}}$ in OLAP analysis requires defining appropriate operations to aid this analysis [3]. Below the Extended Cascaded OLAP (ECOLAP) operations are defined for the extended cascaded star with $ES^C_{\{II\text{-}1\}}$ schema. Using the new structure $ES^C_{\{P\text{-}t\}}$, in OLAP analysis requires defining appropriate operations to aid this analysis.

Hierarchical join is an operation that performs joins for the given hierarchy. The join is performed from the lowest level in the hierarchy to the level predetermined by the second parameter of the operation. This operation is defined in the following way:

$$Hierachical\text{-}Join(H_i, h_i) = \pi_{R_H}(\sigma(h{=}C)(h1 \bowtie \ldots \bowtie hi))$$

where: the first parameter H_i defines the hierarchy from the set of hierarchies of the given snowflake schema, the next parameter h_i is a table, till which the join should be performed. If the table h_i is not specified, then the result of the operations is an empty set. R_H is a set of the projections attributes, h is a set of attributes within the confines of this hierarchy and C is a set of selections conditions, which have to be fulfilled.

Schema Hierarchical Join (SHJ) is an operation which receives, as parameters, a set of hierarchical dimensions and a set of tables, till which the join should be performed. The operation generates a set of hierarchical dimensions after the joins are performed.

$$SHJ\text{: } H{\times}I{\rightarrow}N;\ SHJ(H_i, h_i) = Hierarchical\text{-}Join(H_i, h_i),$$

where: $H = \{H_i\}$ – set of hierarchical dimensions, $I = \{h_i\}$ – set of tables, which for a given hierarchy H_i define the border level of the join h_i within the confines of this hierarchy, N – set of hierarchical dimensions after the joins are performed.

eTraverse (**extended Traverse**) allows to perform the basic OLAP operations within the confines of a single dimension S^S or S^F for a certain level of star $M = k$, $k > 0$. It consists of Relation Operators (*SPJ)* on simple attributes and *SHJ* operations for hierarchical dimensions. Using the expression of relational algebra this operation can be written down as:

$$eTraverse(S, I) = \pi_{R_S}(\sigma(d{=}C)(S \bowtie D \bowtie SHJ(H, I))),$$

where: S is a S^S or S^F schema, D is a set of flat dimensions, H is a set of hierarchical dimensions, I is a set of tables, which for a given hierarchy $H_i \in H$ define the border level of the join h_i within the confines of this hierarchy, d is a set of attributes for D and H, C is a set of selections conditions, R_S is a set of the projections attributes S.

eDecompose (**extended Decompose**) operation can be expressed, using the relational algebra, as:

$$eDecompose(Q, P, I) = \pi_{R_Q}(\sigma(d{=}C)(Q \bowtie traverse(P, I))),$$

where: P is a S^S or S^F schema, containing a set of hierarchical dimension H and it is a single dimension for Q, which is a extended cascaded star schema $ES^C_{\{II\text{-}1\}}$, I is a set of tables, which for a given hierarchy $H_i \in H$ define the border level of join h_i within the confines of this hierarchy, d is a set of attributes for P, C is a set of selections conditions, $R_q$is a set of the projections attributes within Q.

eJump (**extended Jump**). The operation of a two stars join, between which there is no direct relation, results in obtaining a set , which is a cartesian product of both stars. Such set will contain a huge number of records. For this reason the definition of this operation should be understood as a join of two stars with using additional fact tables that contain appropriate relations. *eJump* operation can be expressed:

$$eJump(S_i, S_j, I_i, I_j) = \pi_{R_{i,j}}(eTraverse(S_i, I_i) \bowtie eTraverse(S_j, I_j))),$$

where: S_i and S_j indicate the source and destination of a S^S or S^F schema, containing the sets of hierarchical dimensions H_{S_i} and H_{S_j}, $R_{i,j}$ is a set of projection operations attributes from S_i and S_j, I_i and I_j are sets of tables, which for a given hierarchy $H_i \in H_{S_i}$ and $H_j \in H_{S_j}$ define the border level of the join h_i and h_j.

eCascaded-roll-up. The Roll-up operation decreases the number of the presented data details, as a result some dimensions will be omitted. The starting point is a single star S^S or a snowflake S^F. Performing *eTraverse* operation, a set of attributes from a set of single schema is received. Then there is a move to a lower level to the parent star, which is a $ES^C_{\{II-1\}}$ schema. In order to join both levels the operation *eDecompose* should be performed. In this manner the lowest level is reached M=0. Using expression of algebra relation this process can be written down as:

$$eCascaded\_roll\_up(Q, P, I_q, I_p) = eDecompose(Q, \text{eTraverse}(P, I_p), I_q),$$

where: Q is the $ES^C_{\{II-1\}}$ schema, P is a single star schema S^S or a snowflake schema S^F containing a set of hierarchical dimension H, I_p and I_q are sets of tables, which for a given hierarchy $H_p \in \text{H}$ and $H_q \in \text{H}$ define the border level of join h_p and h_q within the confines of these hierarchy.

eCascaded-drill-down. This operation is the opposite to *eCascaded-roll-up*. It increases the number of details of the presented data, and in some particular cases the *eCascaded-drill-down* operation causes the appearance of new dimensions. The move is performed from the centre of the star to its highest level, additional substars or dimensions are added. Starting from the central star and performing consecutive *eDecompose* operations, a single star level is reached, where the operation *eTraverse* is performed. Notation in relational algebra is as following:

$$eCascaded\_drill\_down(Q, P, I_{pq}, I_q) = eTraverse(eDecompose(Q, P, I_{pq}), I_q),$$

where: Q is the $ES^C_{\{II-1\}}$ schema, P is a single star schema S^S or a snowflake schema S^F containing a set of hierarchical dimension H, I_{pq} and I_q are sets of tables, which for given hierarchy $H_{pq} \in \text{H}$ and $H_q \in \text{H}$ define the border level of join h_{pq} and h_q within the confines of these hierarchy.

eCascaded-slice and **eCascaded-dice.** These operations reduce the number of presented elements for a multidimensional cube by setting additional selection conditions for given attributes of the substars (for eCascaded-slice operation they are attributes of a single dimension – star or snowflake). Notation of both of these operations using relational algebra is:

$$eCascaded\_slice(ES^C_{\{II-1\}}, S, I_S) = Decompose(ES^C_{\{II-1\}}, S, I_S),$$

where: S is a S^S or S^F schema located on level $M=k$, containing a set of hierarchical dimension H and it is a substar for $ES^C_{\{II-1\}}$ ($M=k-1$), I_S is a set of tables, which for a given hierarchy $H_S \in H$ define the border level of join h_S.;

$$eCascaded\_dice(ES^C_{\{II-1\}}, S_1, S_2, I_1, I_2, I_{1,2}) =$$
$$eDecompose(ES^C_{\{II-1\}}, eJump(S_1, S_2, I_1, I_2)), I_{1,2}),$$

where: S_1 and S_2 *are* a S^S or S^F schemas located on level $M=k$, containing a set of hierarchical dimensions H_1 and H_2 respectively and they are (S_1 and S_2) substars for $ES^C_{\{II-1\}}$ ($M=k-1$), I_1 and I_2 are sets of tables, which for a given hierarchy $H_{S_1} \in H_1$ and $H_{S_2} \in H_2$ define the border level of join h_S within the confines of this hierarchy; and $I_{1,2}$ is a set of tables which is defining the border level of join for hierarchies that belong to the set $H_1 \cup H_2$.

Multiple cubes operation. (MCUBE) performs recursive calculations of type CUBE on relation such as base tables or materialized views. In relational algebra it can be expressed with the equation:

$$MCUBE(Q) = op(V, eCascaded\_roll\_up(Q, P, I_q, I_p)),$$

where: *op* is a relational aggregation operator like sum, *V* is a set of central measurements *Q, P* is a set of substars for *Q*, and it can be $ES^{C}_{\{II-1\}}$, S^S or S^F schema which contains a sets of hierarchical dimensions *H*, I_p and I_q are sets of tables, which for a given hierarchy $H_p \in H$ and $H_q \in H$ define the border level of join h_p and h_q within the confines of these hierarchy. Basing on the above expression the *MCUBE* is an aggregation operation for multiple *eCascaded-roll-up*.

Example

Now the execution of one of the earlier defined operations for the $ES^{C}_{\{II-1\}}SDW(t)$ schema, showed on Fig. 1, is presented.

Query: *Find information about nodes (identifier and node models identifier) that own map of areas, that expiry date expires in year 2009.*

The query is related to the cascaded dimension NODE as well as to the snowflake schema MAP. Therefore operation *eDecompose* has to be used. The query using expression of algebra relation is notated as follows:

$$eDecompose\ (NODE, MAP, \{day\}) = \pi_{node.id, node.\mathrm{mod}el\_id}(\sigma_{(year=2008)}(NODE \bowtie eTraverse\ (MAP, \{day\})))$$

eTraverse operation can be written as:

$$eTraverse\ (MAP, \{day\}) = \pi_{map.id, map\_type.description, map.scale}(\sigma_{(year=2008)}(MAP \bowtie SHJ\ (\{H\_TIME\}, \{day\})))$$

Because time hierarchy contains on every level fields from the higher levels (snowflake schema of type 3), there is no need to perform additional joins within the confines of the hierarchy. It is a special case, which is presented in order to show, that *SHJ* operation is also correct in such situations. The *SHJ* operation returns as a result the table DAY, which is joined with the table MAP. The result of the *eTraverse* operation is a table of map identifiers, which fulfill the selection condition. Basing on this, nodes that are located on these maps are selected. Substituting the expansion of

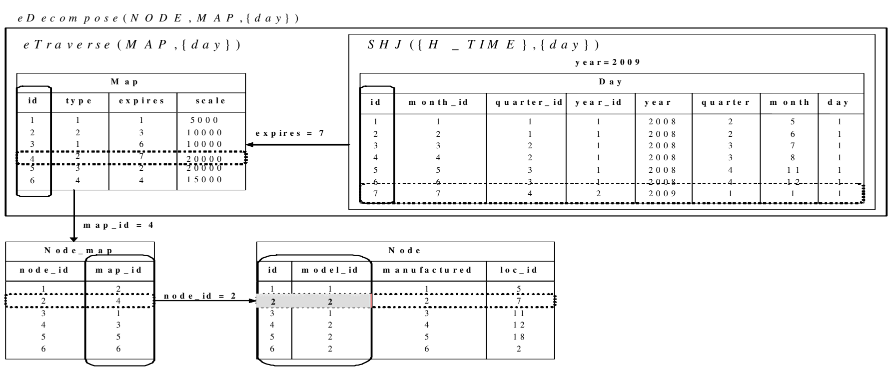

Fig. 2. The execution of the *eDecompose* operation in $ES^{C}_{\{II-1\}}SDW(t)$

the *eTraverse* operation and respecting the fact, that the join between table NODE and MAP can be made only using the table MAP_NODE, it should be notated as follows:

$$eDecompose\ (NODE, MAP, \{day\}) = \pi_{node.id, node.\mathrm{mod}\,el\_id}(NODE \rhd\lhd\ MAP\ \_\ NODE$$
$$\rhd\lhd\ (\pi_{map.id}(\sigma_{(year=2009)}(MAP \rhd\lhd\ SHJ(\{H\_TIME\}, \{day\}))))$$

On Fig. 2 the execution of the above given query on hypothetical data is presented. The result of this query is a tuple in the form: Result [id=2/model id=2].

4 Conclusions

In this paper a complex data modeling schema was presented – the extended cascaded star for spatial data warehouses, with reference to the single star schema and the snowflake schema. Also the basic cascaded ECOLAP operations were defined for the extended cascaded star schema. The presented ECOLAP operations: drill-down, roll-up, slice & dice and MCUBE, allow to analyze data, with usage of the new data model $ES^{C}_{\{II\text{-}1\}}$. Thanks to the use of expressions of the relational algebra it is showed, that the extended cascaded star schema can be built on the basis of the traditional star and snowflake schemas. Future works will be focused on defining ECOLAP operations for other schemas of the extended cascaded star and on the expansion of the set of operations on next elements taking under consideration data exploration functions i.e. classification and prediction.

References

1. Atluri, V., Adam, N., Yu, S., Yesha, Y.: Efficient Storage and Management of Environmental Information. In: 19th IEEE Symposium on Mass Storage Systems, Maryland, USA (2002)
2. Bédard, Y., Merret, T., Han, J.: Fundamentals of Spatial Data Warehousing for Geographic Knowledge Discovery. In: Geographic Data Mining and Knowledge Discovery, ch. 3. Research Monographs in GIS, pp. 53–73. Taylor & Francis, Abington (2001)
3. Gorawski, M., Bugdol, M.: Cascaded ECOLAP Operations. Studia Informatica 28(3A), 43–63 (2007)
4. Gorawski, M., Gębczyk, W.: Distributed Approach of Continuous Queries with kNN Join Processing in Spatial Telemetric Data Warehouse. In: Taniar, D. (ed.) Progressive Methods in Data Warehousing and Business Intelligence, IGI Global, pp. 271–279 (2009)
5. Gorawski, M., Malczok, R.: Materialized aR-tree in Distributed Spatial Data Warehouse. International Journal Intelligent Data Analysis 10(4), 361–377 (2006)
6. Gupta, H., Mumick, I.: Selection of Views to Materialize in a Data Warehouse. Transactions of Knowledge and Data Engineering (TKDE) 17, 24–43 (2005)
7. Han, J., Stefanovic, N., Kopersky, K.: Selective Materialization: An Efficient Method for Spatial Data Cube Construction. In: Wu, X., Kotagiri, R., Korb, K.B. (eds.) PAKDD 1998. LNCS, vol. 1394. Springer, Heidelberg (1998)
8. Jensen, C., Kligys, A., Pedersen, T., Timko, I.: Multidimensional Data Modeling for Location-based Services. VLDB Journal 13, 1–21 (2004)
9. Malinowski, E., Zimanyi, E.: Representing Spatiality in a Conceptual Multidimensional Model. In: ACM Int. Workshop on Geographic Information Systems, GIS 2004 (2004)

10. Shekhar, S., Lu, C., Tan, X., Chawla, S.: Map Cube: A Visualization Tool for Spatial Data Warehouses. In: Geographic Data Mining and Knowledge Discovery, pp. 74–100. Taylor and Francis, Abington (2001)
11. Stefanovic, N., Han, J., Koperski, K.: Object-based Selective Materialization for Efficient Implementation of Spatial Data Cubes. IEEE Transactions on Knowledge and Data Engineering (TKDE), 938–958 (2000)
12. Timoko, I., Pedersen, T.: Capturing Complex Multidimensional Data in Location-based Warehouses. In: ACM Int. Workshop on Geographic Information Systems, GIS 2004 (2004)
13. Yu, S., Atluri, V., Adam, N.: Cascaded Star: A Hyper-dimensional Model for a Data Warehouse. In: Bressan, S., Küng, J., Wagner, R. (eds.) DEXA 2006. LNCS, vol. 4080, pp. 439–448. Springer, Heidelberg (2006)