Cabanac, G., Chevalier, M., Ravat, F. et Teste, O. (2007d). An Annotation Management System for Multidimensional Databases. *In* Song, I.-Y., Eder, J. and Nguyen, T.M., editors : *DaWaK'07: Proceedings of the 9th International Conference on Data Warehousing and Knowledge Discovery*, volume 4654 of *LNCS*, pages 89–98. Springer.

Published paper: http://dx.doi.org/10.1007/978-3-540-74553-2_9

An Annotation Management System for Multidimensional Databases

Guillaume Cabanac[1], Max Chevalier[1,2], Franck Ravat[1], Olivier Teste[1]

[1] IRIT (UMR 5505), SIG, 118 route de Narbonne
F-31062 Toulouse cedex 9, France
[2] LGC (ÉA 2043), 129A avenue de Rangueil, BP 67701
F-31077 Toulouse cedex 4, France

{cabanac, chevalier, ravat, teste}@irit.fr

Abstract. This paper deals with an annotation-based decisional system. The decisional system we present is based on multidimensional databases, which are composed of facts, dimensions, measures, hierarchies and parameters. The expertise of decision-makers is modelled, shared and stored through annotations. These annotations allow decision-makers to make an active reading and to collaborate with other decision-makers about a common analysis project. Each annotation may be related to multidimensional structures as well as to values available in multidimensional tables.

Keywords: Multidimensional Database, annotation, expertise memory.

1 Introduction & Motivations

Multidimensional data analysis consists in manipulations through aggregations of data drawn from various transactional databases. This approach is often based on *multidimensional databases* (MDB). MDB schemas are composed of facts and dimensions; facts model subjects of analysis whereas each dimension represents an analysis axis [11]. In this context, decision-making consists in analysing these multidimensional data. Nevertheless, due to its numeric nature it is often difficult to interpret business data, simply by looking at it. To make a decision from multidimensional data, decision-makers must analyse and interpret data. This expertise that can be considered as an *immaterial capital* is very valuable but it is not exploited in traditional multidimensional systems.

As paper annotations convey information between readers [10], we argue that *annotations* may support this immaterial capital coming from decision-makers on MDB. In the context of decisional-support systems, we consider an annotation as a high value-added component of MDB from the users' point of view. Such components can be used for a *personal use* to remind any information concerning studied data as well as for a *collective use* to share information that makes complex analyses easier. This collective use of annotations would be the base to build an expertise memory that stores decisions, commentaries in analysis context, etc.

This paper addresses the problem of integrating the annotation concept into MDB management systems. Annotations are designed with the objective of assisting decision-makers and making their expertise persistent and reusable.

Related works and discussion. To the best of our knowledge, the integration of annotations in multidimensional databases has not been studied yet. The closest works are related to annotation integration in relational database management systems. First, in the DBNotes system [12, 1, 2, 5], zero or several annotations are associated to a relation component. These annotations are transparently propagated along as data is being transformed (through SQL queries). This annotation system traces the origin and the flow of data. Second, the authors of [6] and [7] specify an annotation oriented data model for the manipulation and the querying of both data and annotations. This model is based on the concept of block to annotate both a single value and a set of values. The authors also introduce MONDRIAN, a prototype implementation of the annotation practice. Third, similar to the previous systems, the works described in [3] and [4] consist in annotating relational data. DBNotes and MONDRIAN use relational data to express annotations whereas this last work models annotations using XML. The model also allows users to cross-reference related annotations.

As conceptual structures of a multidimensional database are semantically richer, the outlined works cannot be directly applied to our context. That is why we specify a specific model to annotate MDB; it consists the following properties:

- An annotation is characterised by a type, an author, and a creation date.
- Annotations can be traced according to an ancestor link. These cross-references are stored to facilitate collaborative work through discussion threads.
- Each annotation is associated to an anchor. The anchor is a path expression for associating annotations to the MDB components (structure or value). Thanks to this anchor, annotations can be associated to different levels of granularity.
- To facilitate user interactions, annotations are defined and displayed through a conceptual view of the MDB where the annotations are transparently propagated and stored into R-OLAP structures.

Paper outline. The remainder of the paper is organised as follows. Section 2 extends a conceptual multidimensional model for integrating annotations. Section 3 describes an R-OLAP system to manage annotations into MDB.

2 An Annotation-featured Multidimensional Model

We extend the conceptual multidimensional model defined in [11].

2.1 Multidimensional Concepts

The conceptual model we define represents data as a constellation [9] regrouping several subjects of analysis (facts), which are studied according to several axes of analysis (dimensions) possibly shared between facts.

Definition. A *constellation* is defined as (N^C, F^C, D^C, $Star^C$, $Annotate^C$) where N^C is the constellation name, F^C is a set of facts, D^C is a set of dimensions, $Star^C$: $F^C \rightarrow 2^{DC}$

associates each fact to its linked dimensions, Annotate^{C} is a set of *global annotations* of the constellation elements (see section 3).

A dimension reflects information according to which subjects will be analysed. A dimension is composed of parameters organised through one or several hierarchies.

Definition. A *dimension* D^{i} is defined by $(N^{Di}, A^{Di}, H^{Di}, Ext^{Di})$ where N^{Di} is the dimension name, $A^{Di} = \{a_1,\ldots, a_q\}\cup\{Id, All\}$ is a set of dimension attributes, $H^{Di} = \{H^{Di}{}_1,\ldots, H^{Di}{}_W\}$ is a set of hierarchies, $Ext^{Di} = \{i^{Di}{}_1,\ldots, i^{Di}{}_Y\}$ is a set of dimension instances, called dimension extension. Each dimension instance is defined by $\forall k\in[1..Y]$, $i^{Di}{}_k = [Id{:}id, a_1{:}v_1,\ldots, a_q{:}v_q, All{:}all]$ where "id" is an unique identifier, "$v_1,\ldots, v_q$, all" are dimension attribute values.

Dimension attributes are modelled according to one or several hierarchies, which organise data granularities of analysis data.

Definition. A *hierarchy* $H^{Di}{}_j$ is defined by $(N^{HDi}{}_j, P^{HDi}{}_j, WA^{HDi}{}_j)$ where $N^{HDi}{}_j$ is the hierarchy name, $P^{HDi}{}_j = \langle Id, p_1,\ldots, p_s, All\rangle$ is an ordered set of dimension attributes, called parameters, $\forall k\in[1..s]$, $p_k\in A^{Di}$. Each parameter specifies a granularity level of the analysis. The $WA^{HDi}{}_j : P^{HDi}{}_j \rightarrow 2^{ADi}$ function associates each parameter to a set of *weak attributes* for adding semantic information to the parameter. Note that all hierarchies start from the *root parameter* noted Id and all hierarchies end to the *extremity parameter* noted All.

A fact regroups indicators called measures that have to be analysed according to several linked dimensions.

Definition. A *fact* F^{i} is defined by $(N^{Fi}, M^{Fi}, Ext^{Fi}, IStar^{Fi})$ where N^{Fi} is the fact name, $M^{Fi} = \{f_1(m_1),\ldots, f_p(m_p)\}$ is a set of measures $m_1,\ldots, m_p$ associated to aggregation functions $f_1,\ldots, f_p$, $Ext^{Fi} = \{i^{Fi}{}_1,\ldots, i^{Fi}{}_x\}$ is a set of fact instances, called fact extension. Each fact instance is defined by $\forall k\in[1..x]$, $i^{Fi}{}_k = [m_1{:}v_1,\ldots, m_p{:}v_p]$ where $v_1,\ldots, v_p$ are the measure values. $IStar^{Fi}: Ext^{Fi}\rightarrow Ext^{Star(Fi)}$ associates each fact instance to its linked dimension instances.

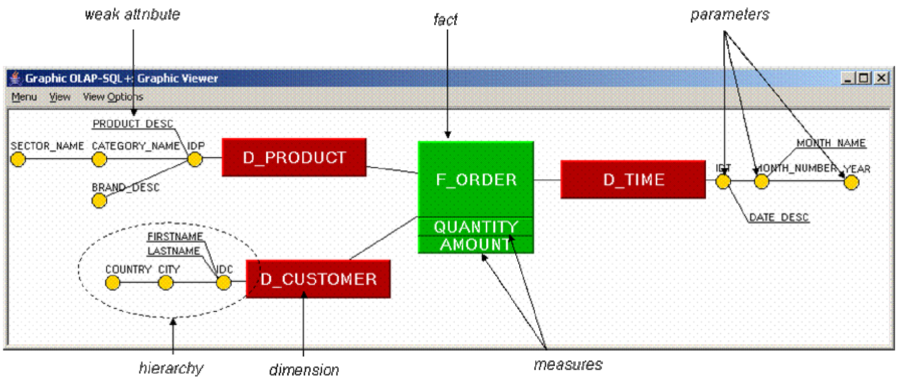

Fig. 1. Example of constellation schema from the commercial domain.

Example. The case study is taken from the commercial domain. Figure 1 shows a schema that supports analyses about amount orders and ordered product quantities. It is composed of one fact, named *Order*, and three dimensions, named *Time*, *Customer*, and *Product*.

In a constellation, every component can be annotated with zero or more annotations (see section 2.3).

2.2 Multidimensional Table

The visualisation of constellations consists in displaying one fact according to several dimensions into a multidimensional table (MT). A MT is more complex than relations because it is organised according to a non-clear separation between structural aspects and data contents [8].

Definition. A multidimensional table T is defined as (S^T, L^T, C^T, R^T, $Annotate^T$)

- S^T = (F, {$f_1(m_1)$, …, $f_p(m_p)$}) is the subject of analysis, which is represented by the fact and its displayed measures $f_1(m_1)$,…, $f_p(m_p)$,
- L^T = (DL, HL, PL) represents the horizontal analysis axis where PL=<All, $p^{HL}{}_{max}$, …, $p^{HL}{}_{min}$> are displayed parameters of DL$\in Star^C(F)$ and HL$\in H^{DL}$ is the *current hierarchy* of DL,
- C^T = (DC, HC, PC>) represents the vertical analysis axis where PC=<All, $p^{HC}{}_{max}$, …, $p^{HC}{}_{min}$>, HC$\in H^{DC}$ and DC$\in Star^C(F)$, HC is the *current hierarchy*,
- $R^T = pred_1 \wedge … \wedge pred_s$ is a normalised conjunction of predicates (restrictions of dimension data and fact data).
- $Annotate^T$ is a set of *local annotations* of the MT elements (see section 2.3).

Example. The following figure depicts an MT example, which displays amount orders according to the temporal axis and the customer axis. Note that a MT represents an excerpt of data recorded in a constellation.

$T_1 = (S_1, L_1, C_1, R_1, \varnothing)$ with:

- S_1=(F_ORDER, {SUM(Amount)});
- L_1=(D_TIME, HTPS, <All, YEAR, MONTH_NUMBER>);
- C_1=(D_CUSTOMER, HGEO, <All, COUNTRY, CITY>);
- R_1=D_PRODUCT.ALL = ‘all’.

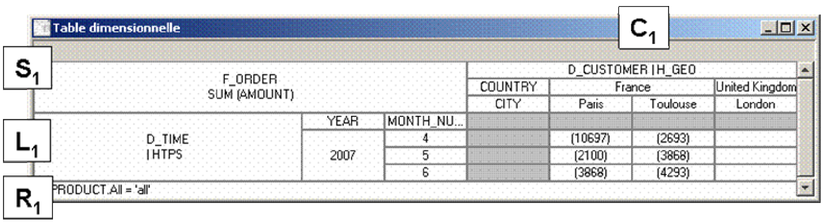

Fig. 2. Example of multidimensional table displaying an analysis.

As it is the case for a constellation, every row or every cell of a MT may be annotated with zero or more local annotations (see next section).

2.3 An Integrated Annotation Model

In order to annotate a multidimensional database, we provide a specific annotation model that is directly integrated into the multidimensional model. The proposed annotation model is collaboration-oriented. It provides functionalities that allow users to share information and to discuss directly in context, through the MT. Therefore, an annotation can be defined at two specific levels of the MDB.

- A *global* annotation concerns a conceptual component of a constellation (fact, measure, dimension, hierarchy, parameter, weak attribute). Such annotations are independent of any context (MT); they are displayed in any MT including the globally annotated element.
- A *local* annotation concerns a specific component of a given MT. These annotations are only displayed in the specific context corresponding to the MT.

As for paper, an MDB annotation is twofold, it consists in:

- *subjective* information that corresponds to its content (*e.g.* a text typed by decision-makers) and at least one type. Indeed, an annotation may be associated with one or many types in order to allow experts to describe it. This makes its understanding easier without having to read its content. In a decisional-support system context, we define some specific types: a commentary, a question, an answer to an existing annotation, a conclusion, many links to another MT, many links to legacy documents…
- *objective* data (also called meta-data) that corresponds to its unique identifier, its creation date, its creator identifier (the user identifier who has created this annotation), a link to the parent annotation (when answering to another annotation) and an anchor to annotated data.

When an annotation is created, the set of objective data is automatically generated by the system, e.g. the anchor that corresponds to the location of the annotated data. The set of subjective information is formulated by the user that creates the annotation.

To illustrate the concept of annotation, we detail in next sections the definition of an annotation anchor and we give examples of anchors from the case study.

2.3.1 Anchor Definition

During an analysis, decision makers visualize synthesized data through MT. These MT can be manipulated by the use of commands defined by the related agebra [11]. Thus, an MT evolves and its annotations should evolve the same way. That is why anchors cannot be specified with a coordinate-based system. That is why we define a unique anchoring structure, which relies on a path-like notation that can be generalized to any multidimensional database. It allows to express both local and global annotations, and is independent of any chosen implantation.

In next definitions, we note `[elt]?` to indicate that `elt` is optional, `[elt]*` to indicate that `elt` can be repeated *0* or *n* times, `{elt1|elt2}` to indicate an alternative between `elt1` and `elt2`, and `^` corresponds to an empty path.

Let us consider *CONS* as a constellation, *MT* as a multidimensional table, *Fact* as a fact, *measure* as a measure, *f(m)* as an aggregation function applied to a measure. Moreover, *val* corresponds to a specific value of the measure, *Dim* is a dimension, *Hier* is a hierarchy, *param* is a dimension attribute (parameter or a weak attribute) and *valueP* is a value of a dimension attribute.

Definition. An anchor is defined as `(S, D1, D2)` where

– S = ^ | {CONS | MT}[.Fact[[/measure | /f(m)][=val]?]*]? designates a path to any fact or measure used in a constellation or in a MT,

– D_1 = ^ | Dim[.Hier[/param[=valueP]?]*]? designates a path concerning the a first dimension (row or column of the MT),

– D_2 = ^ | Dim[.Hier[/param[=valueP]?]*]? designates a path concerning the second dimension (row or column of the MT).

If the two dimensions D_1 and D_2 are given, the system is able to identify a specific cell in the MT. Thanks to this anchoring structure and to the different combinations of values that it allows, annotations can be easily stored in the multidimensional database and be retrieved and displayed in a specific MT for instance.

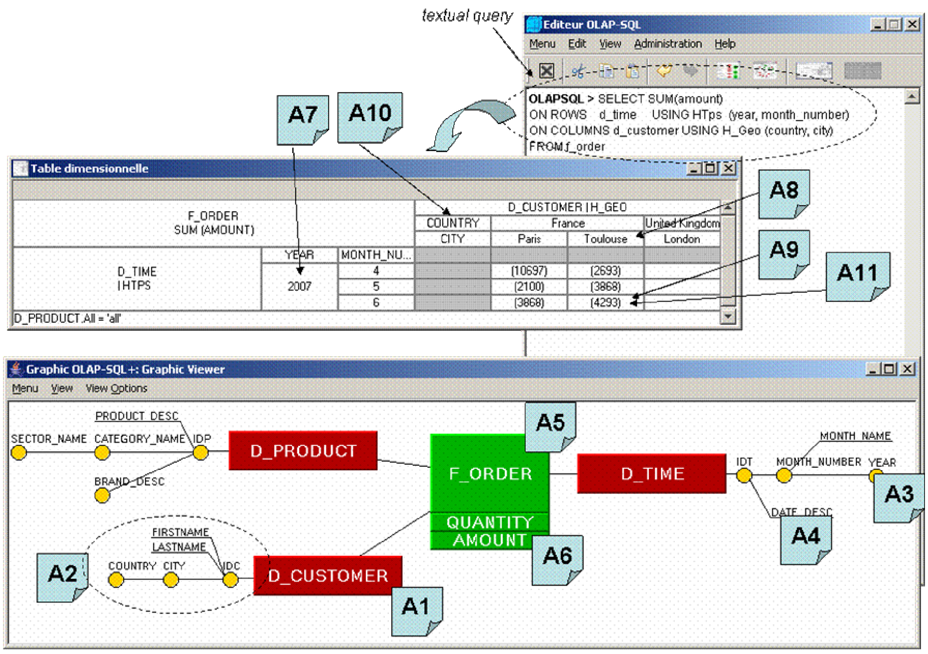

Fig. 3. Example of annotations on a multidimensional table as well as on the MDB schema.

2.3.2 Application

In figure 3, two users annotate the constellation C1 and a multidimensional table MT1. The first user, noted U1, annotates MT1 with A7, A8, A9, and A10. The annotation A9 is a question. This is the root of a discussion thread. The second user, noted U2, has annotated C1 with the annotations from A1 to A6. He also answers to A9 with the annotation A11.

The anchor for each annotation from figure 3 is:

– ***A1***: (^, D_CUSTOMER, ^) or (^, ^, D_CUSTOMER). This anchor supposes that the annotation concerns the D_CUSTOMER dimension. The annotation is displayed every time the D_CUSTOMER dimension is used. To associate this annotation to this dimension when it is linked to the fact F_ORDER the anchor should be transformed in (C1.F_ORDER, D_CUSTOMER, ^).
– ***A2***: (^, D_CUSTOMER.HGEO, ^) or (^, ^, D_CUSTOMER.HGEO).
– ***A3***: (^, D_CUSTOMER.HGEO/YEAR, ^) or (^, ^, D_CUSTOMER.HGEO/YEAR).
– ***A4***: (^, D_CUSTOMER.HGEO/YEAR/DATE_DESC, ^).
– ***A5***: (C1.F_ORDER, ^, ^).
– ***A6***: (C1.F_ORDER/AMOUNT, ^, ^).
– ***A7***: (MT1, D_TIME.HTPS/YEAR='2007', ^).
– ***A8***: (MT1, D_CUSTOMER.HGEO/COUNTRY='France'/CITY='Toulouse', ^).
– ***A9***: (MT1.F_ORDER/SUM(AMOUNT), D_TIME.HTPS/YEAR = '2007'/MONTH_NUMBER = '6', D_CUSTOMER.HGEO/COUNTRY = 'France'/CITY = 'Toulouse'). The annotation concerns the different values stored in the related cell. To annotate only one value corresponding to a specific measure of this cell, the annotation should be transformed in: (MT1.F_ORDER/AMOUNT.SUM = '4293', D_TIME.HTPS/YEAR = '2007'/MONTH-NUMBER = '6', D_CUSTOMER.HGEO/COUNTRY = 'France'/CITY = 'Toulouse')
– ***A10***: (MT1, D_TIME.HTPS/YEAR='2007', ^).
– ***A11***: A11 is identical to A9, only its content differs from the one of A9.

3 R-OLAP Implementation

The architecture of our annotation management system is illustrated in the following figure. We distinguish three main modules: the display interfaces, the engine and the R-OLAP storage.

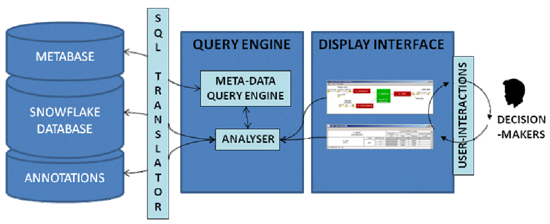

Fig. 4. Threefold annotation management system for multidimensional databases architecture.

The display interfaces (GUI) allow decision-makers to make interactions with the constellation. They enable to display analyses through MT, which are defined from the constellation schema. Thanks to the annotation management system integrated to the MDB, users can annotate the constellation schema and the MT via global and local annotations.

The query engine translates user interactions into SQL queries. Correctness of query expressions is validated through the meta-base. These SQL queries are sent to the databases; results are sent back to the GUI.

The R-OLAP data warehouse is an RDBMS storing multidimensional data, meta-data and annotations.

3.1 Metabase

The constellations are implemented in an R-OLAP context. To store the multidimensional structures, we have defined meta-tables that describe the constellation.

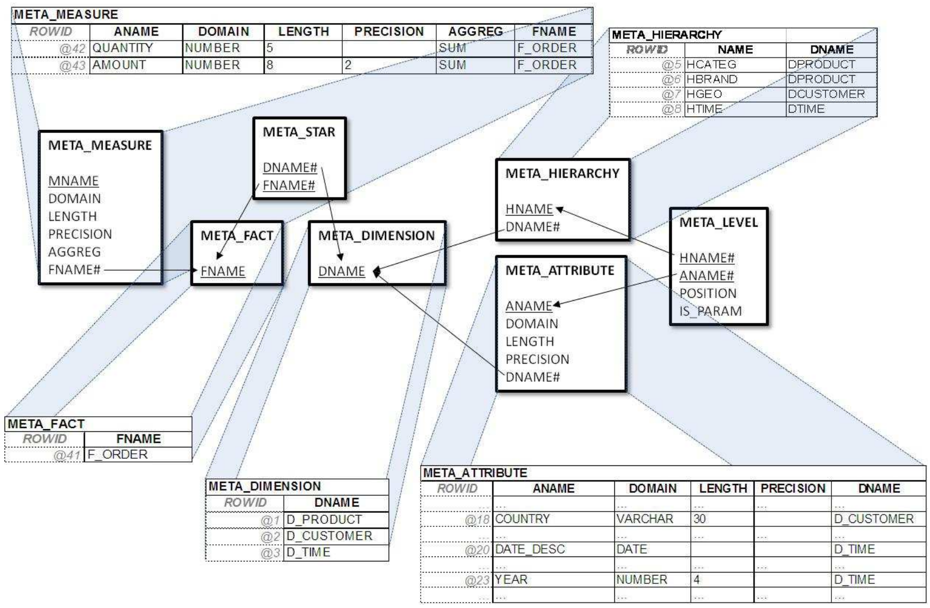

Fig. 5. Metabase for storing a constellation.

Example. Figure 5 describes the constellation structure illustrated in Fig. 1. The multidimensional concepts are described by META_FACT, META_DIMENSION and META_HIERARCHY. Each star schema (one fact and its linked dimensions) is described by META_STAR. META_MEASURE and META_ATTRIBUTE describe respectively the fact measures and the dimension attributes. The hierarchical organisation of dimension attributes is described by META_LEVEL.

3.2 Snowflake Database

An important challenge for storing annotations is the implementation of anchors. To associate each annotation to a unique row in the R-OLAP database, we adopt a snowflake data schema [9]. A snowflake schema consists in normalising dimensions according to hierarchies to eliminate redundancy; the annotations require a unique "anchor" in the database. Note that this implementation choice keeps the maintenance of large dimension tables easy.

Example. The following figure shows the R-OLAP implementation of the constellation illustrated in Fig. 1 according to a snowflake modelling.

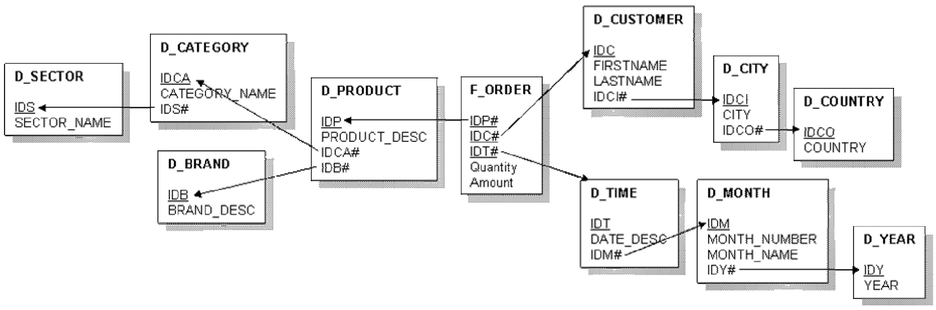

Fig. 6. R-OLAP snowflake schema.

We use a snowflake schema to store detailed data. It is important to note that these tables must be completed with pre-aggregated tables for improving query performances.

3.3 Annotation Storage

We have to provide a mechanism for storing global and local annotations into the same structure. The main problem consists in implementing the formal anchoring technique (described in section 2.3.1) meanwhile providing a homogeneous way to manage the annotations such that:

– some annotations may be anchored to meta-data,
– some annotations may be directly associated to analysed data,
– some annotations may be anchored to aggregated data, which may not be stored.

Our solution consists in storing annotations into a single table whose schema is composed of the following columns:

– PK is the annotation identifier,
– NTABLE is the table or a meta-table where the annotated data is stored,
– ROW is an internal row identifier used in the database system related to the annotated data of the NTABLE,
– COL stores the attribute name of annotated data. Note that if the annotation is anchored to the multidimensional structure, the annotation is anchored to a row

in a meta-table (COL is null) whereas if the annotation is associated to a value, COL is valued.
- DESC stores the annotation content.
- LOCAL represents the annotation scope. When the annotation is local to a multidimensional table, then this attribute is valued.
- TYPE describes the annotation type (comment, question, answer, conclusion…).
- DATE stores the creation date of the annotation.
- PARENT represents a relationship between annotations. The attribute is used to store a discussion thread (for example, an answer following a question).
- AUTHOR is the author of the annotation.

Example. The following table (Fig. 7) stores annotations. The 1st row stores the global annotation A1, which is anchored to the dimension D_CUSTOMER. This anchor, conceptually noted (^, D_CUSTOMER, ^), is stored into the columns NTABLE and ROW; the annotation A1 is anchored to the row identified by @2 in the meta-table META_DIMENSION.

ANNOTATION									
PK	**NTABLE**	**ROW**	**COL**	**DESC**	**LOCAL**	**TYPE**	**DATE**	**PARENT**	**AUTHOR**
1	META_DIMENSION	@2		A1		Comment	02/04/2007		U2
2	META_HIERARCHY	@7		A2		Comment	02/04/2007		U2
3	META_ATTRIBUTE	@23		A3		Comment	02/04/2007		U2
4	META_ATTRIBUTE	@20		A4		Comment	02/04/2007		U2
5	META_FACT	@41		A5		Comment	02/04/2007		U2
6	META_MEASURE	@43		A6		Comment	02/04/2007		U2
7	D_YEAR	@100	YEAR	A7		Comment	02/04/2007		U1
8	D_CITY	@101	CITY	A8	V1	Question	02/04/2007		U1
9	MV1	@200	AMOUNT	A9	V1	Comment	02/04/2007		U1
10	META_ATTRIBUTE	@18		A10	V1	Comment	02/04/2007		U1
11	MV1	@200	AMOUNT	A11	V1	Answer	03/04/2007	9	U2

Fig. 7. Storage of the annotations.

Note that annotations A1 to A6 are global annotations associated to the constellation schema. They are associated to the meta-data tables. In the same way, the local annotation A10 is associated to the meta-data tables.

The global annotation A7 and the local annotation A8 are related to detailed values (of dimension parameters). Let us considers that D_YEAR and D_CITY contain respectively the following rows [@100, y1, 2007] and [@101, ci1, Toulouse, co1]; these rows are linked in the storage table by annotations. The attributed ROW and COL are used to locate these annotate data of the snowflake tables.

The annotations A9 and A11 are related to an aggregated value of the measure AMOUNT. In order to define this annotation anchor, this aggregated data must be materialised. The MT is calculated from the following SQL query, noted V1:

```
SELECT year,month_number,country,city,SUM(amount) AS amount
FROM F_ORDER,
     D_CUSTOMER, D_CITY, D_COUNTRY,
     D_TIME, D_MONTH, D_YEAR
WHERE F_ORDER.idc    =D_CUSTOMER.idc
  AND D_CUSTOMER.idci=D_CITY.idci
  AND D_CITY.idco    =D_COUNTRY.idco
```

```
  AND F_ORDER.idt    =D_TIME.idt
  AND D_TIME.idm     =D_MONTH.idm
  AND D_MONTH.idy    =D_YEAR.idy
GROUP BY year,month_number,country,city;
```

When an annotation is a local annotation, the column LOCAL is valued by its SQL query name; *e.g.* local annotations A8, A9, A10 and A11 are associated to V1.

An important problem is the storage of annotations associated with aggregated values in the MT because these aggregated values are not stored. To store these annotations, we trigger an aggregation of calculated aggregated values. In our example, we note MV1 the materialized view of V1; MV1 stores only aggregated values which are annotated from the multidimensional table.

MV1					
ROWID	year	month_number	country	city	amount
@200	2007	6	France	Toulouse	4293

Fig. 8. Storage of annotated aggregated values.

Note that only annotated aggregated values may be materialised. A more complex approach may exploit pre-aggregated views, which are used to improve query calculus in current data warehousing approaches. We do not investigate this solution in this paper.

5 Concluding Remarks & Future Works

This paper described an implementation of an annotation management system for multidimensional databases. Every piece of multidimensional data can be associated with zero or more superimposed information called annotations. We conceive annotations with the objective of storing decision-makers remarks about multidimensional data that would otherwise have not been kept in a traditional database. We provide annotations for a personal use to remind any information concerning analysed data as well as for a collective use to materialise and to share decision-makers' expertise to facilitate collaborative analyses and decisions.

Our solution allows decision-makers to annotate multidimensional data at various levels of granularities – fact, dimension, hierarchy, attributes, detailed or aggregated values. Annotations assist users in understanding MDB structures and decisional analysis expressed through MT. Global annotations are expressed on the MDB schema components and they are transparently propagated into the MT where local annotations are added according to the analysis context.

We investigated how the global and local annotations can be stored into a homogeneous structure. We develop a relational meta-database describing constellation components; these metadata are associated to global annotations. We also described an R-OLAP environment where multidimensional data are stored into snowflake relations. The normalised dimensions allow the system to annotate detailed multidimensional data. In this normalised framework, we are interested in determining which aggregated information to materialise annotated aggregated values.

In our current system, global annotations are propagated into local analysis contexts. In addition, it would also be interesting to prospect how to detect similarities between analyses in order to propagate annotations from the local analysis context to similar analysis contexts. We also investigate opportunities for integrating annotations into the lattice of materialised views, which is only used to improve query computation in our current approach. Future works will revisit materialised view selection algorithms for determining relevant materialised views according to annotations.

References

1. Bhagwat D., Chiticariu L., Tan W-C., Vijayvargiya G.: An Annotation Management System for Relational Databases. 30th International Conference on Very Large Databases, VLDB’04, Toronto (Canada), 2004.
2. Bhagwat D., Chiticariu L., Tan W-C., Vijayvargiya G.: An Annotation Management System for Relational Databases. VLDB Journal 14(4), pp. 373-396, 2005.
3. Bhatnagar N., Juliano B.A., Renner R.S.: Data Annotation Models. 3rd International Conference on Intelligent Computing and Information Systems, ICICIS’07, Cairo (Egypt), 2007.
4. Bhatnagar N., Juliano B.A., Renner R.S.: Data Annotation Models and Annotation Query Language. 4th International Conference on Knowledge Mining, ICKM’07, Bankok (Thaïland), 2007.
5. Chiticariu L., Tan W-C., Vijayvargiya G.: DBNotes: A Post-It System for Relational Databases based on Provenance. ACM SIGMOD International Conference on Management of Data, SIGMOD’05, pp.942-944, 2005.
6. Cong G., Fan W., Geerts F.: Annotation Propagation Revisited for Key Preserving Views. 15th Conference on Information and Knowledge Management, CIKM’06, pp. 632-641, Arlington (USA), 2006.
7. Geerts F., Kementsietsidis A., Milano D.: MONDRIAN: Annotating and querying databases through colors and blocks. 22th International Conference on Data Engineering, ICDE’06, p.82, 2006.
8. Gyssens M., Lakshmanan L. V. S.: A Foundation for Multi-Dimensional Databases. 23rd International Conference on Very Large Data Bases, VLDB’97, pp.106-115, Athens (Greece), 1997.
9. Kimball R.: The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses. John Wiley & Sons, USA, 1996.
10. Marshall C. C.: Toward an ecology of hypertext annotation. HYPERTEXT'98: Proceedings of the 9th ACM conference on Hypertext and hypermedia: links, objects, time and space---structure in hypermedia systems, pp. 40-49, 1998.
11. Ravat F., Teste O., Tournier R., Zurfluh G.: Algebraic and graphic languages for OLAP manipulations. International Journal of Data Warehousing and Mining – IGI Publishing, 2007 (to appear).
12. Tan W-C.: Containment of Relational Queries with Annotation Propagation. International Workshop on Database and programming Language, DBLP’03 (LNCS 2921, pp.37-53), 2003.