Chapter IV

Decisional Annotations: Integrating and Preserving Decision-Makers' Expertise in Multidimensional Systems

Guillaume Cabanac, Université de Toulouse, IRIT (UMR 5505), SIG,
118 route de Narbonne, F-31062 Toulouse cedex 9, France
Max Chevalier, Université de Toulouse, IRIT (UMR 5505), SIG,
118 route de Narbonne, F-31062 Toulouse cedex 9, France
Franck Ravat, Université de Toulouse, IRIT (UMR 5505), SIG,
118 route de Narbonne, F-31062 Toulouse cedex 9, France
Olivier Teste, Université de Toulouse, IRIT (UMR 5505), SIG,
118 route de Narbonne, F-31062 Toulouse cedex 9, France

Abstract

This chapter deals with an annotation-based decisional system. The decisional system we present is based on multidimensional databases, which are composed of facts and dimensions. The expertise of decision-makers is modelled, shared and stored through annotations. These annotations allow decision-makers to carry on active analysis and to collaborate with other decision-makers on a common analysis.

Introduction

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 (subjects of analysis) and dimensions (axes of analysis) (Ravat et al., 2008). Decision-making consists in analysing these multidimensional data. Nevertheless, due to its numeric nature it is difficult to interpret business data. This work requires decision-makers to achieve a tedious cognitive effort, which is an *immaterial capital*. To take relevant decisions this required expertise is very valuable but it cannot be expressed, stored, and exploited in traditional multidimensional systems. Such an expertise can be qualified as ephemeral from the organization standpoint.

As paper annotations convey information between readers (Marshall, 1998), we argue that *annotations* can also support this immaterial capital for MDB. We consider an annotation as a high value-added component of MDB from the users' standpoint. Such components can be used for a *personal use* to remind any information concerning the data under study, as well as for a *collective use* to share information that makes complex analyses easier. This collective use of annotations would serve as a basis for building an expertise memory that stores previous decisions and commentaries. Moreover as Foshay et al. (2007) state "Metadata helps data warehouse end users to understand the various types of information resources available from a data warehouse/business intelligence environment." As a consequence, in our proposition, annotations and their contents enable end users to analyse, discuss and share knowledge in context during the decision making process.

This chapter addresses the problem of integrating the annotation concept into MDB management systems. Annotations are designed to assist decision-makers and to turn their expertise persistent and reusable.

Related works and discussion. To the best of our knowledge, integrating annotations in the MDB context has not been studied yet. The closest works are related to annotation integration in Relational DataBase Management Systems (RDBMS). First, in the DBNotes system (Bhagwat et al. 2004, 2005; Chiticariu et al., 2005; Tan, 2003) zero or several annotations are associated with a relation element. 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 in (Cong et al., 2006) and (Geerts et al., 2006) specify an annotation-oriented data model for the manipulation and the search 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. A prototype, called MONDRIAN, supports this annotation model. Third, similar to the previous systems, the works presented in (Bhatnagar et al., 2007a) and (Bhatnagar et al., 2007b) consist in annotating relational data. DBNotes and MONDRIAN use relational data to express annotations whereas this last work models annotations using eXtensible Markup Language (XML). The model allows users to cross-reference related annotations.

As conceptual structures of a MDB are semantically richer, the outlined works cannot be directly applied to our context.

- Contrary to RDBMS where a unique data structure is used to both store and display data, in our MDB context, the storage structures are more complex and a specific display is required.
- In the RDBMS framework, annotations are straightforwardly attached to tuples or cell values (Bhagwat et al., 2004). Due to the MDB structures, annotations must be attached to more complex data; *e.g.* dimension attributes are organised according to hierarchies and displayed decisional data are often computed from aggregations.

To annotate a MDB we define a specific model having the following properties:

- An annotation is characterised by a type, an author, and a creation date.
- Each annotation is associated with an anchor, which is based on a path expression tying the annotations to the MDB components (structure or value). Thanks to this anchor, annotations can be associated with different data granularities.
- Annotations can spark off debates called "discussion threads," which enables asynchronous communication during collaborative work.
- To facilitate user interactions, annotations are defined and displayed through a conceptual view of the MDB where they are transparently propagated and stored into R-OLAP structures.

Chapter outline. Section 2 extends the conceptual multidimensional model defined in (Ravat et al., 2008) for integrating annotations. Section 3 describes the R-OLAP implementation of an annotated MDB. Section 4 presents a system for managing annotations on a MDB.

An Annotation-Featured Multidimensional Model

In this section, we describe the multidimensional model concepts. First, we define basic concepts like fact, dimension, hierarchy and constellation. The conceptual model we define is close to the user's standpoint and independent of implementation choices. This model intends to facilitate correlations between several subjects of analysis through a constellation of facts and dimensions, and it supports several data granularities according to which subjects may be analyzed. Second, we extend the model by integrating annotations. Annotations are used both to comment multidimensional data and to share various user standpoints during the analysis processes.

Multidimensional Concepts

Concept of Constellation.

The conceptual model we define represents data as a constellation (Kimball, 1996) gathering several subjects of analysis (facts), which are studied according to several axes of analysis (dimensions).

Definition. A constellation C 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).

Example. The case study is a business example. The multidimensional database supports the analysis of sales through quantities and amounts of products sold to a customer at a specific date. The constellation is composed of three dimensions named *time*, *customer*, and *product*, and an unique fact named *order*; i.e. the constellation is formally defined by ("SALES", {F^{ORDER}}, {D^{TIME}, $D^{PRODUCT}$, $D^{CUSTOMER}$}, $Star^{SALES}$, $Annotate^{SALES}$) where $Star^{SALES}(F^{ORDER})$ = {D^{TIME}, $D^{PRODUCT}$, $D^{CUSTOMER}$ }.

Concept of Dimension and Hierarchy.

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,…, a_q, All} is a set of dimension attributes,
- H^{Di} = {H^{Di}_1,…, H^{Di}_W} is a set of hierarchies,
- Ext^{Di} = {i^{Di}_1,…, i^{Di}_Y} is a set of dimension instances.

Example. The dimension $D^{PRODUCT}$ is defined by ("D_PRODUCT", {IdP, Product_Desc, Brand_Desc, Category_Name, Sector_Name, All}, {H^{Brand}, H^{Sector}}, $Ext^{DProduct}$).

The dimension $D^{CUSTOMER}$ is defined by ("D_CUSTOMER", {IdC, Firstname, Lastname, City, Country, All}, {$H^{Country}$}, $Ext^{DCustomer}$). The following table (table 1) gives some dimension instances.

Table 1. Some dimension instances of ExtDCustomer.

IdC	Firstname	Lastname	City	Country	All
$i^{DCUSTOMER}_1$	Pierre	Dupond	Paris	France	All
$i^{DCUSTOMER}_2$	Paul	Durand	Paris	France	All
$i^{DCUSTOMER}_3$	Jean	Martin	Toulouse	France	All
$i^{DCUSTOMER}_4$	Marie	Martin	Toulouse	France	All
$i^{DCUSTOMER}_5$	John	Smith	London	United Kingdom	All

Analysis axes are dimensions seen through a particular perspective, namely a hierarchy. This hierarchy of dimension attributes organises the different graduations of the analysis axis.

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 = <Id, p_1,… ,p_s ,All> 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.

Entire hierarchies in one dimension start with a same parameter, noted Id called root parameter. Entire hierarchies end with a same parameter, noted All called extremity parameter.

Example. The dimension $D^{PRODUCT}$ is composed of two hierarchies, which are defined as (“H_Brand”, <IdP, Brand_Desc, All>, WA^{HBrand}(IdP) = {Product_Desc}) and (“H_Sector”, <IdP, Category_Name, Sector_Name, All>, $WA^{HSector}$(IdP) = {Product_Desc}).

Concept of Fact

A fact regroups indicators called measures that have to be analysed.

Definition. A fact F_i is defined by (N^{Fi}, M^{Fi}, Ext^{Fi}, $IStar^{Fi}$) where

- N^{Fi} is the fact name,
- $M^{Fi} = \{f_1(m_1),\dots, f_p(m_p)\}$ is a set of measures $m_1,\dots, m_p$ associated with aggregation functions $f_1,\dots,f_p$,
- $Ext^{Fi} = \{i^{Fi}_1,\dots, i^{Fi}_x\}$ is a set of fact instances.
- $IStar^{Fi}$: $Ext^{Fi} \rightarrow Ext^{Star(Fi)}$ associates each fact instance to its linked dimension instances.

Example. The fact F^{ORDER} is defined by (“F_ORDER”, {SUM(Quantity), SUM(Amount)}, Ext^{FOrder}, $IStar^{FOrder}$). The following table gives some fact instances. Note that these instances refer to six products, three dates and the four customer instances, which are defined in the above section. More precisely:

- i^{DTIME}_1 refers to a date in April, 2007,
- i^{DTIME}_2 refers to a date in May, 2007,
- i^{DTIME}_3 refers to a date in June, 2007.

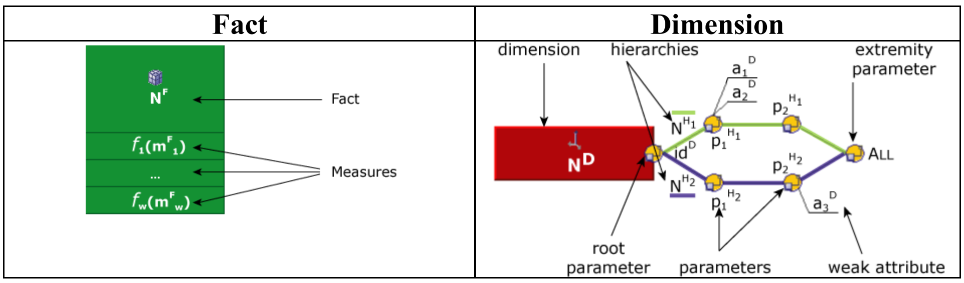
Table 2. Some fact instances of Ext^{FOrder}.

IdC	IdT	IdP	Quantity	Amount
$i^{DCUSTOMER}_1$	i^{DTIME}_1	$i^{DPRODUCT}_1$	5	5000
$i^{DCUSTOMER}_1$	i^{DTIME}_1	$i^{DPRODUCT}_2$	3	697
$i^{DCUSTOMER}_2$	i^{DTIME}_1	$i^{DPRODUCT}_1$	5	5000
$i^{DCUSTOMER}_1$	i^{DTIME}_2	$i^{DPRODUCT}_1$	2	2000
$i^{DCUSTOMER}_2$	i^{DTIME}_2	$i^{DPRODUCT}_3$	1	100
$i^{DCUSTOMER}_2$	i^{DTIME}_3	$i^{DPRODUCT}_4$	1	3868
$i^{DCUSTOMER}_3$	i^{DTIME}_1	$i^{DPRODUCT}_1$	2	2000
$i^{DCUSTOMER}_4$	i^{DTIME}_1	$i^{DPRODUCT}_5$	1	693
$i^{DCUSTOMER}_4$	i^{DTIME}_2	$i^{DPRODUCT}_4$	1	3868
$i^{DCUSTOMER}_3$	i^{DTIME}_3	$i^{DPRODUCT}_1$	4	4000
$i^{DCUSTOMER}_4$	i^{DTIME}_3	$i^{DPRODUCT}_6$	1	293

Graphical notations

We introduce graphical notations to design multidimensional databases. These notations extend the notations introduced in (Golfarelli et al., 1998). Table 3 describes graphical notations of facts and dimensions with their hierarchies.

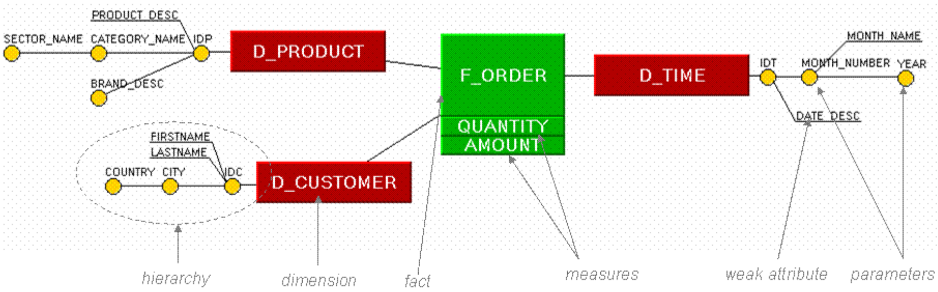
Table 3. Graphical notations of multidimensional concepts.



Example. Figure 1 shows the multidimensional schema of the previous examples. The illustrated constellation schema is composed of one fact named F_ORDER and three dimensions respectively named D_PRODUCT, D_CUSTOMER and D_TIME. This constellation supports analyses of sales through quantities and amounts of products sold to customers at several dates.

The dimension attributes are organised according to one or several hierarchies; i.e. each path starting from Id and ending by All represents a hierarchy. Note that the extremity parameter (All) is not displayed in the graphical representation as this parameter tends to confuse users (Malinowsky & Zimányi, 2006).

Figure 1. Example of constellation schema.



Multidimensional Table

Constellation schemas depict MDB structures whereas user analyses are based on tabular representations (Gyssens & Lakshmanan, 1997) where structures and data are displayed. 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 (Gyssens & Lakshmanan, 1997).

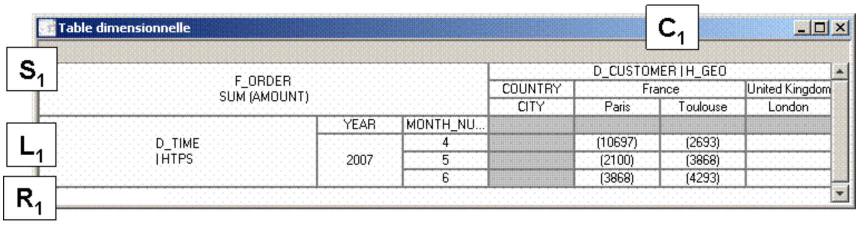
Definition. A multidimensional table T is defined as (S^T, L^T, C^T, R^T, $Annotate^T$)

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

Example. Figure 2 depicts an example of MT that displays amount orders according to the temporal axis and the customer axis. $T_1 = (S_1, L_1, C_1, R_1, \emptyset)$ 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 = true.

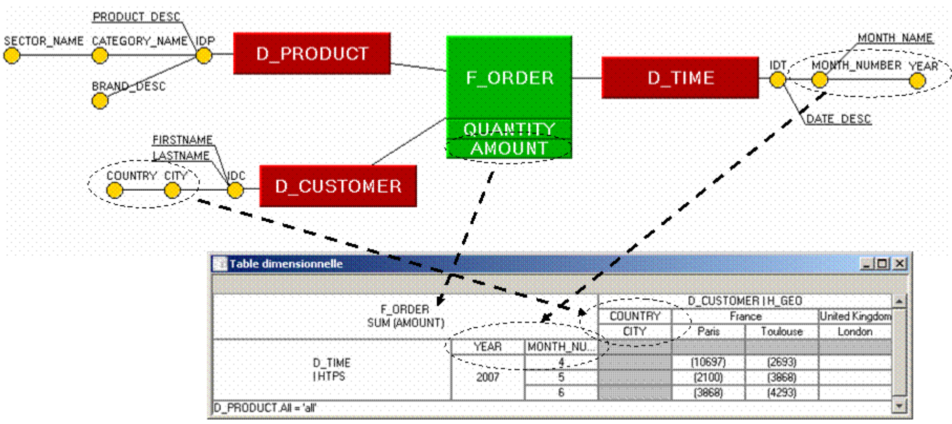
Figure 2. Example of MT.



Note that a MT represents an excerpt of data recorded in a constellation. Measures are displayed according to a bidimensional space, which is defined through two dimensions.

Example. Figure 3 shows the previous MT and the corresponding constellation elements from which the MT is extracted. S_1 is extracted from the fact named F_ORDER ($F^{ORDER} \in F^C$) whereas L_1 and C_1 are derived from two linked dimensions of F^T ($D^{TIME} \in Star^C(F^{ORDER})$ and $D^{CUSTOMER} \in Star^C(F^{ORDER})$). Along each current dimension, a current hierarchy is fixed ($H^{TPS} \in H^{TIME}$ and $H^{GEO} \in H^{CUSTOMER}$) according to which some parameters (and/or weak attributes) are projected in the MT. These projected attributes fixed the displayed data granularities; *i.e.* they represents the graduation of the analysis axes.

Figure 3. Example of MT.



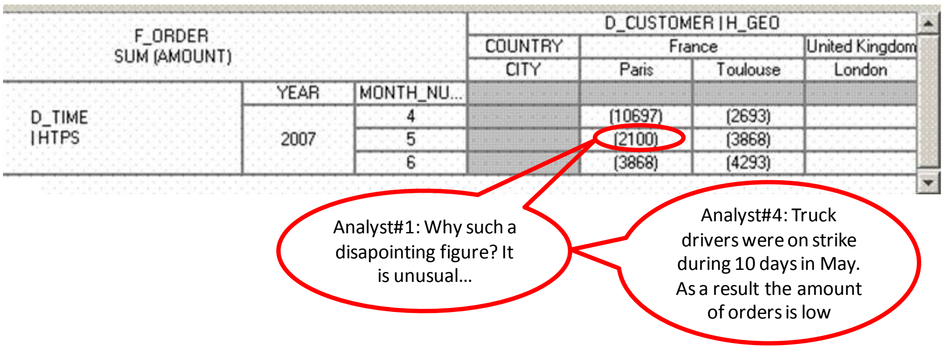
An Integrated Annotation Model

In order to annotate a MDB, we provide a specific annotation model that is incorporated into the multidimensional model. As for paper-based and digital document annotations (Cabanac et al., 2007), a MDB annotation is twofold; it consists in:

– *subjective* information that corresponds to its content (e.g. a text typed in by decision-makers) and at least one "annotation type" to understand its content easier, i.e. without having to read its content. We define some basic types (a comment, a question, an answer to an existing annotation, a conclusion…) which can be extended with domain-specific types.
– *objective* data (also called meta-data) that correspond to the annotation unique identifier, its creation date, its creator identifier, a link to the parent annotation (when answering to another annotation) and an anchor to annotated data.

The system automatically generates the set of objective data whereas the annotation creator formulates the set of subjective data.

Figure 4. Example of a real-life discussion (thread) in the context of a MT.



The proposed annotation model is collaboration-oriented. It provides functionalities that allow users/designers to share information that is relevant to analyses/designs and to discuss and debate directly in the context of any MT through discussion threads (thanks to the link to the parent annotation). Figure 4 shows an example of such discussion between two analysts (users). Annotations and discussions/debates may concern a single analysis or may be more general as they concern every analysis containing the annotated elements. In addition annotating schema elements enables designers and users to share comments, in order to improve their understanding of the annotated elements. Thus, in our approach we define an annotation at two levels:

– A *local* annotation is ***only*** displayed in a specific context corresponding to a specific MT.
– A *global* annotation is shown in ***any*** MT displaying the globally annotated element(s).

As a result various global and local annotations can be associated with a unique element according to the annotator's need.

During an analysis, decision makers visualize synthesized data through MT. The MT content can be modified by the use of commands associated with a related algebra (Ravat et al., 2008). Annotations should follow these changes. As a consequence annotation anchors cannot be specified with a coordinate-based system. That is why we define a unique anchoring notation. This later relies on a path-like notation that allows the anchoring of any annotation to any element displayed in a MT or existing in a constellation. Moreover, the proposed anchoring notation takes into account local and global levels associated with annotations.

In the following definitions, λ denotes the 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 *m*, *val* as a specific measure value, *Dim* as a dimension, *Hier* as a hierarchy, *param* as a dimension attribute (parameter or weak attribute) and *valueP* as a value of a dimension attribute.

Definition. An anchor is defined as (S, D_1, D_2) where[1]:

– S = λ | [CONS | MT][.Fact "(" (/measure | /f(m) ")" [=val])*] denotes a path to any fact or measure used in a constellation or in a MT.
– D_1 = λ | Dim[.Hier(/param[=valueP])*] denotes a path concerning the first dimension of the MT.
– D_2 = λ | Dim[.Hier(/param[=valueP])*] denotes a path concerning the second dimension 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 notation and to the different combinations of values that it allows, annotations can be easily stored in the MDB, retrieved, and displayed in a specific MT for instance.

Example. In Figure 5, two users annotate elements related to the constellation C1 and elements displayed in the multidimensional table MT1. User U1 creates the annotations A7, A8, A9, and A10. The annotation A9 is a question that corresponds to the root of a discussion thread. User U2 creates the annotations from A1 to A6. He also answers A9 through the A11 annotation. Figure 5 only shows elements concerned by every annotation: it does not show the way annotations are displayed in the MT.

[1] This notation complies with the Extended Backus-Naur Form (EBNF) notation (ISO-14977).

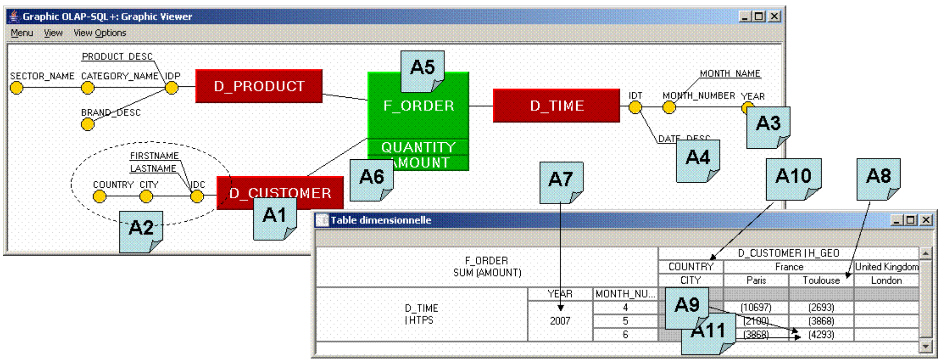
The anchor for each annotation is:

- ***A1***: (λ,D_CUSTOMER,λ) or (λ,λ,D_CUSTOMER) which are equivalent paths[2]. This anchor implies that the annotation concerns the D_CUSTOMER dimension in any constellation associated with this dimension. The annotation will be displayed every time D_CUSTOMER is used. To limit the scope of this annotation and to display it only when the constellation C1 and the F_ORDER fact are used together for instance, one has to transform the anchor into (C1.F_ORDER,D_CUSTOMER,λ) or (C1.F_ORDER,λ,D_CUSTOMER).
- ***A2***: (λ,D_CUSTOMER.HGEO,λ).
- ***A3***: (λ,D_TIME.HTPS/YEAR,λ).
- ***A4***: (λ,D_TIME.HTPS/IDT/DATE_DESC,λ).
- ***A5***: (C1.F_ORDER,λ,λ). This annotation will only be displayed when the fact F_ORDER is associated with the constellation C1.
- ***A6***: (C1.F_ORDER/AMOUNT,λ,λ).

Previous anchor paths refer to global annotations since they do not contain any element specific to any MT. These annotations will be displayed every time annotated elements are used. If needed, we can limit the scope to a specific MT (local annotation) of the annotation A5 for instance by transforming its anchoring path into (MT1.F_ORDER,λ,λ). This means that this annotation will only be displayed in MT1. Most of the following annotations are local ones.

- ***A7***: (MT1, D_TIME.HTPS/YEAR='2007',λ).
- ***A8***: (MT1, D_CUSTOMER.HGEO/COUNTRY='France'/CITY='Toulouse',λ).
- ***A9***: (MT1.F_ORDER/SUM(AMOUNT),D_TIME.HTPS/YEAR='2007'/MONTH_NUMBER='6',D_CUSTOMER.HGEO/COUNTRY='France'/CITY='Toulouse'). This latter anchoring path refers to the potentially evolving measure value contained in the specified MT1 cell. To annotate the specific value of this measure one have to include it into the anchoring path: (MT1.F_ORDER/SUM(AMOUNT***)='4293'***, D_TIME.HTPS/YEAR='2007'/MONTH_NUMBER='6',D_CUSTOMER.HGEO/COUNTRY='France'/CITY='Toulouse'). This means that the corresponding annotation will only be displayed if this value is unchanged.
- ***A10***: (MT1, D_CUSTOMER.HGEO/COUNTRY, λ) .
- ***A11***: The anchoring path of A11 is identical to A9, only its content is different. The link between A9 and A11 is stored as an objective meta-data in A11.

Figure 5. Example of annotations on a MT as well as on the MDB schema.



The anchoring path we propose is suitable to express any kind of annotation that users may link to any multidimensional element (schema, multidimensional table). Even if the notation used complies with the EBNF notation, concretely users may not express themselves such anchoring paths. On the contrary, this anchoring path is automatically generated by the system according to the elements selected by the user.

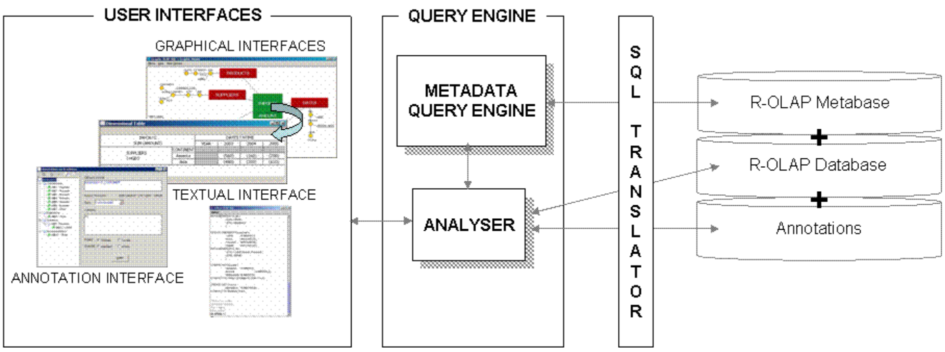
[2] To improve readability we do not specify every equivalent anchoring paths in the examples.

R-OLAP Implementation

In order to validate the solution that we presented in this chapter, we developed an annotated multidimensional management system. As mentioned in Figure 6, the architecture of our annotation management system is composed of three main modules:

- The display interfaces (GUI) enable decision-makers (1) to annotate the constellation schema and the MT via global and local annotations, and (2) to display analyses through annotated MT.
- The query engine translates user interactions into SQL queries. Correctness of query expressions is validated through meta-data. 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.

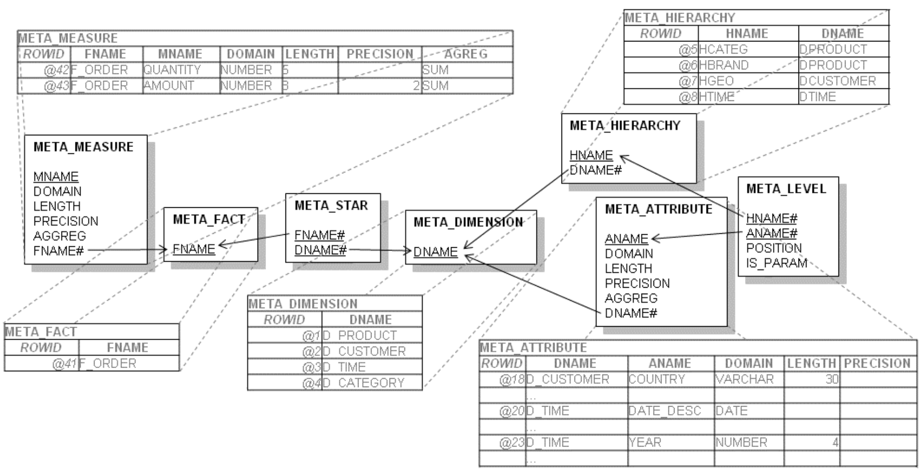
Figure 6. Threefold annotation management system for MDB s architecture.



Metabase structure

Constellations are implemented in an R-OLAP context. To store the multidimensional structures, we have defined meta-tables that describe the constellation (META_FACT, META_DIMENSION, META_HIERARCHY…). For example, Figure 7 describes the constellation structure illustrated in Figure 1; this example presents metatables of one constellation.

Figure 7. Metabase for storing a constellation.

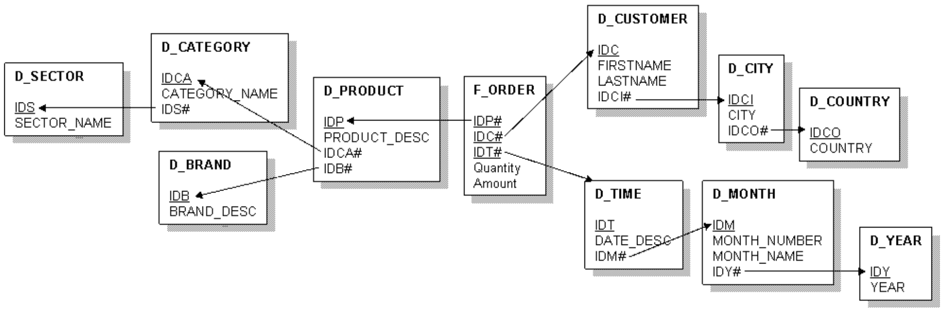


Snowflake Database

An important challenge for storing annotations is the implementation of anchors. To associate each annotation with a unique row in the R-OLAP database, we opted for a snowflake data schema (Kimball, 1996). It consists in normalising dimensions according to hierarchies so as to eliminate redundancy; the annotation anchors point towards a unique data.

Example. Figure 8 shows the R-OLAP implementation of the constellation illustrated in Figure 1 according to a snowflake modelling.

Figure 8. R-OLAP snowflake schema.



Note that these tables of the snowflake schema must be completed with pre-aggregated tables for improving query performances. Moreover, as argued in (Bhagwat et al., 2004), adding and propagating annotations in RDBMS must drop performances down. Their experimental results show that for large databases (500MB and 1GB), the queries integrating annotations took only about 18% more time to execute than their corresponding SQL queries.

Annotation storage

We provide a mechanism for storing global and local annotations into the same structure. The main problem consists in implementing the formal anchoring notation while providing a homogeneous way of managing the annotations that may be anchored to detailed data, aggregated data or meta-data.

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

Example. The following table (Figure 9) stores annotations defined in section 2.2.

Figure 9. Storage of the annotations.

ANNOTATION									
PK	NTABLE	ROWID	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

These annotations are anchored to three levels.

- Annotations A1 to A6 as well as A10 are associated with the **meta-data tables**; *e.g.* A1, stored in the 1st row and conceptually noted (λ, D_CUSTOMER,λ), is anchored to the row identified by @2 into the META_DIMENSION table.
- The global annotation A7 and the local annotation A8 are anchored to **detailed values** (of parameters). The attributes named ROW and COL are used to locate these annotated data. In Figure 8 we assume that D_YEAR and D_CITY contain respectively the rows [@100, y1, 2007] and [@101, ci1, Toulouse, co1].
- The annotations A9 and A11 are anchored to **aggregated values** of the measure AMOUNT. In order to define the anchor, 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 or,D_CUSTOMER cu,D_CITY ci,D_COUNTRY co,D_TIME ti,D_MONTH mo,D_YEAR ye
WHERE or.idc=cu.idc AND cu.idci=ci.idci AND ci.idco=co.idco
  AND or.idt=ti.idt AND ti.idm=mo.idm   AND mo.idy=ye.idy
GROUP BY year,month_number,country,city;
```

To store these annotations we define the materialized view of V1, noted MV1 that stores only annotated aggregated values as illustrated in Figure 10.

Figure 10. Storage of annotated aggregated values.

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

An Annotated Multidimensional Database Management System

Our annotated multidimensional database management system is based on several GUI (Graphical User Interface) implemented in Java 6 on top of the Oracle 10g RDBMS (see figure 6). It allows the definition, manipulation, and querying of constellation and their annotations.

The constellation schema is defined through SQL-like commands. The textual interface allows users to express these orders. The system generates R-OLAP structures to store decisional data and it populates its metabase where multidimensional structures are depicted. Figure 11 shows the command for creating the dimension named D_PRODUCT and the command for creating the fact named F_ORDER.

Figure 11. Example of SQL-like commands.

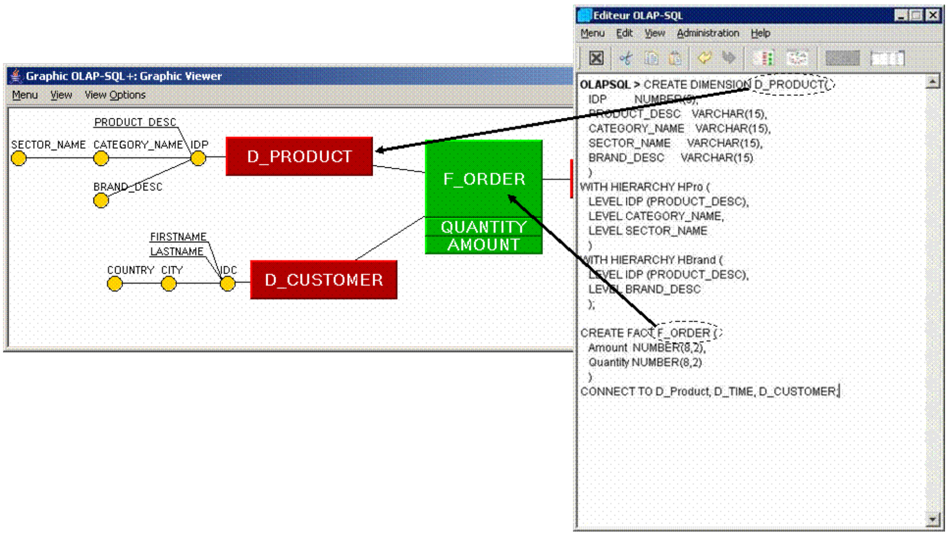
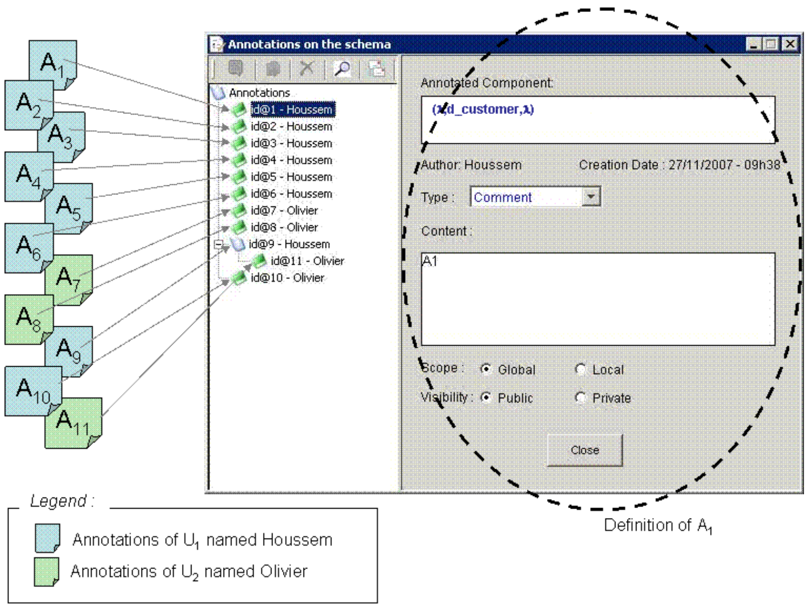


Figure 12. Example of annotation definition.



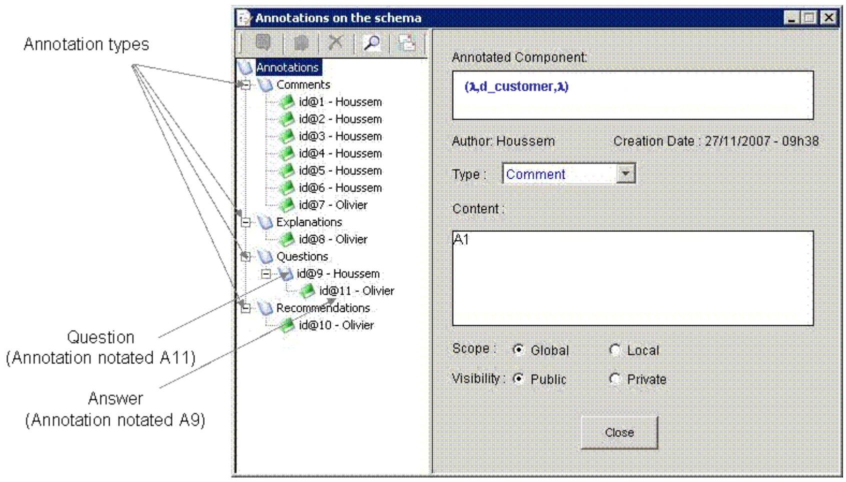
The constellation schema is displayed through a specific GUI:

– The displayed constellation schema is composed of facts, dimensions and hierarchies. The graphical notations are based on the conceptual notations that we presented in section 2.1.4.
– Users analyze decisional data through multidimensional tables. These multidimensional tables are computed from extracted data of the R-OLAP database.

In order to improve their decision-making process, the system provides annotation features to users. The annotations are defined from the constellation schema and/or from the multidimensional tables. Figure 13 shows an example of annotation creation.

Our current annotated multidimensional database management system provides interfaces to display annotations by authors, by types, or by MDB concepts. The following figure gives an example of annotations by types; note that users can display questions as well as their answers.

Figure 13. Annotations by types.



Concluding Remarks & Future Works

This chapter described the implementation of an MDB integrating annotations. Every piece of multidimensional data can be associated with zero or more annotations. We conceive annotations as a means of storing decision-makers remarks about multidimensional data that would otherwise not be kept in a traditional database. Indeed every annotation contains high value information since it the annotation content is contextualized within a specific analysis context. Thus, from the organization standpoint, it is worth to store and reuse them. In our proposition annotations are provided for a personal use to remind any information concerning the analyzed data, as well as for a collective use to materialize and to share decision-makers' expertise, thus facilitating collaborative analyses and decisions. The model we propose allows end-users to point troubles related to any schema element (hierarchy reorganisation, need for attribute details, wrong/missing values...) through annotations, which can be exploited by designers so that they can modify the database accordingly.

Our solution enables 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 displayed into all MT integrating the annotated data, whereas local annotations are displayed according to a analysis context, *i.e.* a specific MT.

We investigated how global and local annotations can be stored into a homogeneous data structure. We developed a relational meta-database describing constellation components; these metadata are associated with global annotations. We also described an R-OLAP environment where multidimensional data are stored into snowflake relations. The normalized dimensions enable the system to annotate detailed multidimensional data. In this normalized framework, we are interested in determining which aggregated information to materialize annotated values. The implementation solution we describe provides straightforward, uniform and efficient storage structures of decisional annotations over multidimensional data.

In the proposed model, annotations can only be "public" or "private". Unfortunately, these simple security levels are not suitable to the real-life context. Thus, we have to develop security management policies in order to better fit the enterprise needs. It would be interesting to prospect how to detect similarities between analyses in order to propagate annotations from the local analysis context to any similar analysis contexts. We also investigate opportunities for integrating annotations into the lattice of materialized views to improve query computation in our current approach. Future works will revisit materialized view selection algorithms for determining relevant materialized views according to annotations. A new challenge raised compared to the RDBMS context is the annotation propagation along aggregated data. To do this, an interesting trail can be seen through the aggregation of annotations content (text) with specific aggregation functions (Ravat et al., 2007).

REFERENCES

Bhagwat, D., Chiticariu, L., Tan, W.-C., & Vijayvargiya, G. (2004). An Annotation Management System for Relational Databases. In Proceedings of 30th VLDB International Conference on Very Large Databases, Toronto, Canada (pp. 900-911).

Bhagwat, D., Chiticariu, L., Tan, W.-C., & Vijayvargiya, G. (2005). An Annotation Management System for Relational Databases. VLDB Journal, 14(4), (pp. 373-396).

Bhatnagar, N., Juliano, B. A., & Renner, R. S. (2007). Data Annotation Models. In Proceedings of 3rd ICICIS International Conference on Intelligent Computing and Information Systems, Cairo, Egypt.

Bhatnagar, N., Juliano, B. A., & Renner R. S. (2007). Data Annotation Models and Annotation Query Language. In Proceedings of the ICKM International Conference on Knowledge Mining, Bangkok, Thailand (pp. 440-445).

Cabanac, G., Chevalier, M., Chrisment, C., & Julien, C. (2007). Collective Annotation: Perspectives for Information Retrieval Improvement. In *Proceedings of the 8th RIAO International Conference on Information Retrieval and its Applications*, Pittsburgh, PA, USA (electronic edition http://riao.online.fr/papers/99.pdf).

Chiticariu, L., Tan, W.-C., & Vijayvargiya, G. (2005). DBNotes: A Post-It System for Relational Databases based on Provenance. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, MD, USA (pp. 942-944).

Cong, G., Fan, W., & Geerts, F. (2006). Annotation Propagation Revisited for Key Preserving Views. In Proceedings of the 15th ACM CIKM International Conference on Information and Knowledge Management, Arlington, VA, USA (pp. 632-641).

Foshay, N., Mukherjee, A., & Taylor, A. (2007). Does Data Warehouse End-User Metadata Add Value?, Communications of the ACM, 50(11), (pp. 70-77).

Geerts, F., Kementsietsidis, A., & Milano, D. (2006). MONDRIAN: Annotating and querying databases through colors and blocks. In Proceedings of the 22th IEEE ICDE International Conference on Data Engineering, Atlanta, GA, USA (pp. 82).

Golfarelli, M., Maio, D., & Rizzi, S. (1998). Conceptual design of data warehouses from E/R schemes. In Proceedings of the 31st HICSS Annual Hawaii International Conference on System Sciences, Hawaii (pp. 334).

Gyssens, M., & Lakshmanan, L. V. S. (1997). A Foundation for Multi-Dimensional Databases. In Proceedings of the 23rd VLDB International Conference on Very Large Databases, Athens, Greece (pp. 106-115).

Kimball, R. (1996). The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses. John Wiley & Sons, USA.

Malinowski, E., & Zimányi, E. (2006). Hierarchies in a multidimensional model: From conceptual modeling to logical representation. Journal of Data & Knowledge Engineering, 59(2) (pp. 348-377).

Marshall, C. C. (1998). Toward an Ecology of Hypertext Annotation. In Proceedings of the 9th ACM HYPERTEXT International Conference on Hypertext and Hypermedia, Pittsburgh, PA, USA (pp. 40-49).

Ravat, F., Teste, O., Tournier, R., & Zurfluh, G. (2008). Algebraic and graphic languages for OLAP manipulations. International Journal of Data Warehousing and Mining, 4(1), (pp. 17-46).

Ravat, F., Teste, O., & Tournier, R. (2007). OLAP Aggregation Function for Textual Data Warehouse. In Proceedings of the 9th ICEIS International Conference on Enterprise Information Systems, Funchal, Portugal (pp. 151-156).

Tan, W.-C. (2003). Containment of Relational Queries with Annotation Propagation. In Proceedings of the 9th DBPL International Workshop on Database and Programming Languages, LNCS 2921, Potsdam, Germany (pp. 37-53).