Extension of OLAP Operators to the Multiversion Data Warehouse context

Mourad Hsan, Ines Zouari, Faiza Ghozzi, Rafik Bouaziz

Extension of OLAP Operators to the Multiversion Data Warehouse context

Mourad HSAN — Ines ZOUARI — Faiza GHOZZI— Rafik BOUAZIZ

University of Sfax, Faculty of Economics and Management, Computer Department,
P.O.Box. 1088, 3018 Sfax, Tunisia
Multimedia InfoRmAtion Computing Laboratory (MIRACL)
E-Mail: mouradhsan@yahoo.fr; ines.zouari@isimsf.rnu.tn; jedidi_faiza@yahoo.fr; raf.bouaziz@fsegs.rnu.tn

ABSTRACT

A Data Warehouse (DW) stores data coming from heterogeneous data sources at decisional ends. To query the stored data, there are specific operators, called OLAP operators which facilitate the analysis and the visualization of data. However, these operators, as they are defined in classic DW context, do not allow querying a MultiVersion Data Warehouse (MVDW) which takes account of the evolutions which can affect the members of dimension and/or the DW schema components. In order to appropriately analyze multiversion data, an extension to the traditional OLAP operator is required. In this paper, we propose a MVDW query language called MVOLAP. To this end, we extend the classic OLAP operators to adapt them to the MVDW context.

Keywords: *multiversion, Data Warehouse, OLAP Operators, Multidimensional Table.*

1. INTRODUCTION

A data warehouse (DW) is a large database that integrates data from autonomous and heterogeneous external data sources. It allows organizing the available data in a multidimensional space for the purpose of detailed or aggregated analysis. DW data are analyzed by the **O**n-**L**ine **A**nalysis **P**rocessing (**OLAP**) operators, e.g. Display, Slice, Drilldown, which facilitate the interactive analysis and the visualization of a great volume of data [5]. Besides, these operators allow exploiting relevant information in order to make the good decisions in the brief times.

The multidimensional model is the most used in the DW modelling. But, the static structure of this model can be a real limitation to keep track of the evolutions that can affect the schema and the content of the DW. In fact, these evolutions can emanate from data source changes or from the new decision-maker needs. To tackle the problem of content and schema changes of data sources, many temporal multidimensional models have been proposed ([3, 6, 8, 10]). Our approach consists in a multiversion data warehouse (MVDW) model that supports the temporal versioning of dimension member (DMem) and schema components [15].

Due to temporal and versioning extensions that enhance MVDW model, the analysis of the data stored in a MVDW requires new analytical tools and an extended query language. The classic (non temporal) OLAP operators are not adequate for such a language. Hence, they need to be extended and redefined. For this end, we propose to extend the textual definition of the classic OLAP operators with temporal clauses, in order to be able to address more than one DMem version. In this paper, we present a MVDW query language, called MVOLAP. It is based on a specific grammar which is defined according to the BNF form (Backus Normal Form). The visualization structure of MVOLAP operators is ensured through multidimensional tables (MT) displaying data that correspond to one fact and two or more of its linked dimensions. Besides, we present our prototype that shows the implementation of the extended OLAP operators.

The rest of this paper is organized as follows. Section 2 describes works that deal with temporal DW and query languages. In section 3, we present the basic principles and the formal description of MVOLP query language. Section 4 focuses on the experimental validation of the proposed solutions. Section 5 concludes the paper.

2. RELATED WORK

This section overviews related work in multiple research areas that are relevant to the work presented in this paper.

2.1. OLAP OPERATORS IN NO TEMPORAL DW

[4] proposes two DW query languages. The first one is textual. It is based on the relational algebra. It defines specific operators such as selection, aggregation and rollup. The second one is graphic. It allows defining queries as sequence of graphs going from the input schema graph to the result schema graph. In these query languages, the definition of OLAP operators is ambiguous.

In [11], the authors propose a textual query language, called OLAP-SQL that allows manipulating constellation databases. This language constitutes an extension of the SQL language. The Select clause is extended in order to express operators of the multidimensional algebra (Rollup, Drilldown, …).

[12] proposes an algebraic and graphic language for OLAP manipulations. This algebra defines OLAP operators (e.g. Rollup, Drilldown, Slice, Dice,…) that allow complex analyzes and produce multidimensional tables for displaying analyzed data. Besides, a graphic language is defined based on this algebra.

2.2. TEMPORAL DW AND OLAP OPERATORS

Handling evolution in DW can be classified in three approaches: schema evolution [1], temporal extensions [2] and versioning extensions [3, 6, 10]. Schema evolution approach maintains only the current DW structure and a set of data that evolves in time. Temporal extensions approaches timestamp modified data in order to create temporal versions. Versioning extensions techniques handle evolution by means of DW schema and data separated versions corresponding to various temporal periods.

There are few works dealing with the manipulation and the querying of temporal DW. Among these works, we distinguish the following ones.

[9] proposes a temporal multidimensional model and a temporal query language named TOLAP (Temporal OLAP). This language uses predicative rules. It allows supporting several types of schema and data evolution, such as the insertion of a level in a dimension and the DMem subdivision. The definition of a TOLAP query requires the definition of atoms: rollup atom, fact atom, descriptive atom and constraint atom. TOLAP is enriched in [14] which proposes the translation of TOLAP queries into the SQL language.

[6] proposes a temporal multidimensional model, called COMET, based on the time-stamping of dimension members and all member relations by the validity time interval [Ts, Te] in order to create structure versions (SV). This model uses transformation functions based on weight factors in order to ensure the data conversion between two SV. It does not deal with querying DW data.

In [3], authors propose a temporal multidimensional model using multiversion fact table. This latter collects data from different DW versions by using a specific dimension that describes several temporal presentation modes, such as the temporally consistent mode. This model allows data visualization relatively to all DW versions when applying the correspondent transformation functions. For this reason, it does not support a query language.

[10] defines a multiversion DW as a set of real and alternative DW versions. Each DW version includes schema and data versions. In order to query such a MVDW, the authors propose an extension of the traditional SQL language. A MVDW query that addresses more than one version is treated in two steps. In the first step, this query is decomposed to the set of independent partial queries, each one for a separate DW version. Every partial query is then executed in its appropriate DW version. In the second step, partial results of the partial queries are integrated, if possible, into a common set of data.

In [7], the DW schema is defined as a graph of simple functional dependencies. Schema modification operations are proposed to create new DW schema versions. Here, authors adopt the augmented schema technique to define cross-version queries, i.e., queries whose temporal horizon spans multiple versions. Based on this approach, [13] presents "X-Time", a prototype for managing schema versioning in relational DW and supporting cross-version queries. The approach discussed in [7, 13] deals with the DW schema versioning, but not with the DMem versioning.

3. MULTIVERSION QUERY LANGUAGE: MVOLAP

In this section, we describe the basic principles of MVOLAP, our multiversion query language, and the proposed OLAP operators.

3.1 ILLUSTRATIVE EXAMPLE

To illustrate the OLAP operators proposed in our language, we refer to a real case stemming from the commercial activity of the National Society of Exploitation and Distribution of Waters in Tunisia (SONEDE) [16].

Figure 1 presents the cube "**Water_Meter_Management**". This cube analyzes the measure "Meter_Action_Nbre" (number of actions done on water meters) according to the following dimensions:

- ***Meter_Action***: it describes the actions done on water meters, e.g., *Creation_Meter*, *Change_Meter*, *Termination_Meter*, *Restoring_Meter*. The hierarchy of this dimension consists of a single level: *Meter_Action*.
- ***Time***: it is structured according to the hierarchy "*Month → Year*".
- ***Meter_Model***: it describes the models of water meters. It consists of the hierarchy "*Brand → Kind*".
- ***Organisational_Unit***: it contains the following levels: *Exploitation_Center*, *District* and *Regional_Direction*. These levels are organized as follows: "*Exploitation_Center → District → Regional_Direction*".

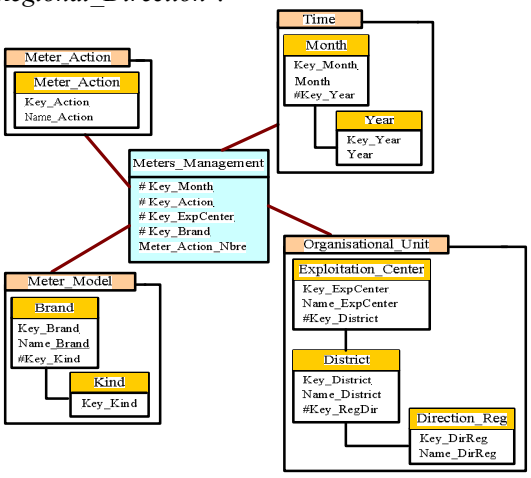

Figure 1: Schema of the cube "*Water Meter Management*"

3.2 BASIC PRINCIPLES

MVOLAP language is based on the definition of

OLAP operators in a multiversion schema context. It is based on a specific grammar which is defined according to the BNF form (Backus Normal Form). This grammar contains the necessary rules to define the MVOLAP operators. Indeed, using such a grammar is simpler than using a predicative language, such as the language TOLAP proposed in [9].

Each of the MVOLAP operators (TDISPLAY, TSLICE, …) constitutes an extension of a classic OLAP operator (DISPLAY, SLICE …). This extension consists in enriching operator definitions by an analysis interval [TDAn, TFAn], where TDAn represents the time of the beginning of analysis and TFAn represents the time of the end of analysis. Consequently, an MVOLAP operator can implicate several versions of Dimension Member (DMem), when the specified analysis interval contains several versions of DMem. In that case, it is necessary to decompose this interval into sub-intervals according to the versions of the implied DMem and to treat the operator with regard to each of these intervals. In fact, extending the definition of an OLAP operator by a temporal interval differs from the temporal extension of the SQL query that corresponds to this operator, as it is proposed in [10].

Each MVOLAP operator has, as an input, a multidimensional table (MT). Its output consists in one or several MT. The results of the MVOLAP operators can be presented in two modes: Consistent time mode and transformation mode. The first one displays a set of partial results; each one is relative to a sub-interval of the analysis interval, i.e. to a DMem version. The second one displays the resulting data according to a particular DMem version by applying transformation functions.

To ensure the visualization of the results of MVOLAP operators, in an ergonomic way, we chose to use the multidimensional table (MT) structure. This it constitutes an adaptation of the MT defined in [11]. Such a MT allows showing the operator-specified dimensions, on lines and columns as well as their hierarchies. The values of the analyzed measures are placed in the intersection of lines and columns.

3.3 MVOLAP OPERATORS

In this section, we describe four MVOLAP operators: TDisplay, TSlice, TDrilldown and TRollup. For each of these operators, we present the formal definition, the required constraints and the execution scenario.

3.3.1 TDisplay (Temporal Display)

Definition 1: This operator allows displaying cube data that correspond to a fact and several dimensions according to an analysis time interval. It is defined as follows:

TDISPLAY *(C, TDAn, TFAn, {(m_i, f_i)}, {(D_j, H_{jp}, L_{jk}, emp_j)})* = $\{MT_i^S\}$, such as:

- **C**, indicates the name of the cube to be analyzed;
- **TDAn** (respectively **TFAn**), indicates the time of the beginning of analysis (respectively the time of the end of analysis);
- **{(m_i, f_i)}**, is a set of measures accompanied by their aggregation functions ;
- **{(D_i, H_{jp}, L_{jk}, emp_j)}**, describes the set of the displayed dimensions. Every quadruplet is constituted by a dimension (D_j), a hierarchy (H_{jp}), a level (L_{jk}) belonging to this hierarchy, and a place (emp_j) ("line" or "column") in the MT result.

Constraints:

- The cube C must be already defined in the set of cubes;
- There is at least one version of C, such that its application time interval and the analysis time interval [TDAn, TFAn] are overlapping;
- The dimensions D_j must be already defined in the dimension set;
- The dimensions D_j must be assigned to the cube C during the analysis time interval [TDAn, TFAn].
- Every level L_{jk} must be already defined in the level set $\mathcal{L}$ and has to belong to the dimension D_j during the interval [TDAn, TFAn].
- The measures have to belong to the versions of the cube C, which are in application during the analysis time interval [TDAn, TFAn].

Output:

- $\{MT_i^S\}$: is a set of MT displaying the fact of the cube C and the dimensions D_j according to their places emp_j, such that $i \in \{1,\ldots, n\}$ and n is the number of sub-intervals belonging to the interval [TDAn, TFAn].

Execution Scenarios:

- After the check of the constraints described above, the system determines the versions of the components concerned by the analysis (the cube C, the dimensions D_j, the levels L_{jk} and the DMem of L_{jk}). Besides, the system has to look for the overlapping between the application time interval of these component versions with the analysis time interval [TDAn, TFAn] specified in the operator;
- Scenario 1: *mono-version execution*
  The analysis interval [TDAn, TFAn] belongs to the intersection of all the application time the analysis intervals of the component versions (dimension, level, DMem, …), concerned by the analysis. In that case, the execution of the operator TDISPLAY is similar to that of the classic operator DISPLAY. A single MT is supplied as a result;
- Scenario 2: *multiversion execution in the consistent time mode*
  The analysis time interval [TDAn, TFAn] contains several versions of the components implicated by the operator. For example, the cube C has two versions in [TDAn, TFAn], and the DMem Mem_k has three versions inside [TDAn, TFAn]. In that case, the interval [TDAn, TFAn] is decomposed into sub-intervals describing the noticed component versions. Consequently, the query corresponding to

the operator is decomposed into a set of independent partial queries; each will be executed in its corresponding sub-interval. A resulting MT is shown for each of these queries. It is what we call displaying data in the consistent time mode;

- Scenario 3: *multiversion execution with transformation*
  Further to the displaying of the result in consistent time mode, the user can ask for data displaying according to a particular version of DMem, chosen among the DMem versions deducted during the decomposition of the analysis time interval. In this case, it is necessary to apply transformation functions based on weight factors to ensure the transformation of the resulted measure values according to the chosen DMem version.

3.3.2 TSlice (Temporal Slice)

Definition 2: This operator allows specifying restriction predicates on dimension data. Since, it is executed after the TDISPLAY operator, TSlice uses the same analysis time interval as TDISPLAY. TSlice is defined as follows :

TSLICE $(\{MT_i^I\}, \{(D_j, L_{jk})\}, \{restriction_j\}) = \{MT_j^S\}$ such as:

- $\{MT_i^I\} = \{(C^I, TDAn, TFAn, \{(m_i^I, f_i^I)\}, \{(D_j^I, H_{jp}^I, L_{jk}^I, Emp_j^I)\})\}$, is a set of the initial MT, obtained as a result of the previous TDisplay operator. Every table MT_i^I is defined in one sub-intervals being a member of the analysis interval (interval defined in the first executed operator, TDISPLAY).
- $\{(D_j, L_{jk})\}$, is a set of the dimensions concerned by the defined restrictions. Every couple is constituted by a dimension D_j and a level L_{jk} belonging to this dimension.
- $\{restriction_j\}$, is a set of the selection conditions associated with the level L_{jk} of the dimension D_j.

Constraints:

- Every dimension D_j must be already defined in the dimensions set $\mathcal{D}$ of the cube C and must belong to the initial tables MT_i^I;
- Every level L_{jk} must be already defined in the level set $\mathcal{L}$ and has to belong to the dimension D_j and to a table among the initial tables $\{MT_i^I\}$;

Output:

- The TSlice operator result consists in the set of multidimensional tables MT_j^S which display DMem that satisfy the selection conditions associated with the dimensions D_j.

Execution Scenarios:

- After the check of the constraints described above, the system determines the concerned component versions according to the operator.
- Scenario 1: *mono-version execution*: similar to that of the TDISPLAY operator.
- Scenario 2: *multiversions execution in consistent mode*: similar to that of the TDISPLAY operator. Besides, certain sub-intervals used in MT^I have to be abandoned in case the restriction conditions concern DMem that haven't changed during the analysis interval, i.e. each of these DMem has a unique version during the analysis interval.
- Scenario 3: *multiversion execution with transformation*: similar to that of the TDISPLAY operator.

3.3.3 TDrilldown (Temporal Drilldown)

Definition 3: TDrilldown operator consists in displaying data according to a more detailed hierarchy level. This operator uses the same analysis time interval that the one specified in the TDISPLAY operator. TDrilldown is defined as follows:

TDRILLDOWN $(\{MT_i^I\}, \{(D_j, L_{jm})\}) = \{MT_j^S\}$ such as:

- $\{MT_i^I\} = \{(C^I, TDAn, TFAn, \{(m_i^I, f_i^I)\}, \{(D_j^I, H_{jp}^I, L_{jk}^I, emp_j^I)\})\}$, is the initial MT set;
- $\{(D_j, L_{jm})\}$ This set describes the more detailed levels according to which cube data have to be displayed. Each of these levels is associated with its correspondent dimension D_j.

Constraints:

- The dimension D_j must be assigned to the initial table;
- The level L_{jm} must be assigned to the dimension D_j. This level must be more detailed than the level $L_{jk}^I \in D_J$, initially chosen in the TDisplay operator;
- The level L_{jm} has to belong to the dimension D_j during the analysis time interval [TDAn, TFAn].

Output:

- $\{MT_j^S\} = \{(C^I, TDAn, TFAn, \{(m_i^I, f_i^I)\}, \{(D_j^I, H_{jp}^S, L_{jk}^I \cup L_{jm}, Emp_j^I)\})\}$: is the set of MT results.

Execution Scenarios:

- The system starts by checking the constraints defined above, that correspond to this operator
- If these constraints are validated, the system determines the new analysis sub-intervals (if they exist) when looking for overlaps between the analysis time interval and the application time intervals of the DMem versions which are instances of the new selected level L_{jm}.
- After calculating the analysis sub-intervals, the correspondent MT are displayed as it is explained in scenario (2) of the TDisplay operator.
- The user can choose to visualize TDrillDown results according to one DMem version which should be chosen from DMem versions visualized in scenario (2). For this, we have to run scenario (3) of the TDisplay operator.

3.3.4 TRollup (Temporal Rollup)

Definition 4: The TRollup operator consists in displaying the data according to coarser less detailed hierarchy level. This operator uses the same analysis time interval that the one specified in the TDISPLAY operator. TRollup is defined as follows:

TROLLUP $(\{MT_i^I\}, \{(D_j, L_{jm})\}) = \{MT_j^S\}$ such as:

- $\{MT_i^I\} = \{(C^I, TDAn, TFAn, \{(m_i^I, f_i^I)\}, \{(D_j^I, H_{jp}^I, L_{jk}^I, Emp_j^I)\})\}$, is a set of the initial MT;
- **$\{(D_j, L_{jm})\}$:** This set describes the less detailed levels according to which cube data have to be

displayed. Each of these levels is associated with its correspondent dimension D_j.

Constraints:

- Every dimension D_j must be already defined in the dimensions set $\mathcal{D}$ of the cube C and belongs to initial tables MT_i^I;
- Every level L_{jm} must be already defined in the level set $\mathcal{L}$ and has to belong to the dimension D_j during the interval [TDAn, TFAn]. This level must be less detailed than L_{jk}^I.

Output:

- $\{MT_j^S\} = \{(C^I, TDAn, TFAn, \{(m_i^I, f_i^I)\}, \{(D_j^I, H_{jp}^S, L_{jk}^I/L_{jm}, Emp_j^I)\})\}$, is a set of the MT results.

Execution Scenarios:

- The system starts by checking the constraints defined above, that correspond to this operator.
- If these constraints are validated, the system determines the analysis sub-intervals (if they exist) by applying the scenario (2) of the TDISPLAY operator. In fact, DMem of the new chosen levels (less detailed) may have several versions during the analysis time interval. Besides, scenario (3) of the TDisplay operator can be applied to visualize results according to one DMem version.

4. EXPERIMENTAL VALIDATION OF THE PROPOSED SOLUTIONS

We opted for an experimental validation of the proposed MVOLAP operators. For this end, we implemented a prototype, called MVOLAP (MultiVersions OLAP), under the environment of development JBuilderX and Oracle10g. In this section, we show two operators TDisplay and TSlice.

4.1 TDISPALY

The decision-maker wants to analyze the sum of action meter numbers and to visualize data according to the dimensions Brand and year, during the period of time [01/01/2006 - 31/12/2007]. Figure 2 shows the formulation of TDisplay with the operator editor of our prototype.

The system starts by the decomposing the analysis interval specified in the operator. Indeed, the system notices that the analysis interval contains two sub-intervals: [01/01/2006 - 31/12/2006] that contains DMem versions *Brand01* and *Brand05* and [01/01/2007 - 31/12/2007] that contains DMem versions *Brand011*, *Brand012*, *Brand013* and *Brand05*. The system also notices that the brand *Brand01* is subdivided into three brands *Brand011*, *Brand012* and *Brand013* with effect from "01/01/2007".

Figure 3 shows the result of this operator in the consistent time mode. Figure 4 shows the result according to sub-intervals [01/01/2006 - 31/12/2006] and [01/01/2007 - 31/12/2007] with the application of transformation functions.

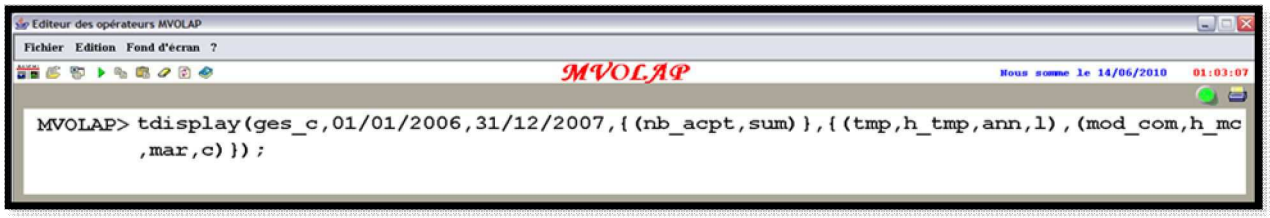

Figure 2: Formulation of the operator *TDisplay*

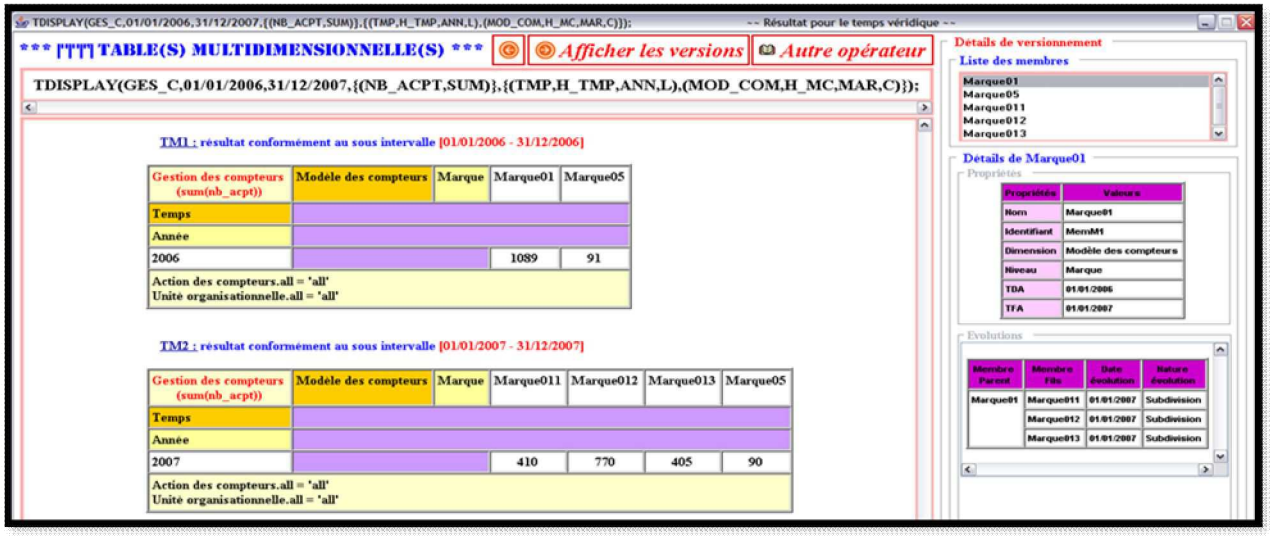

Figure 3: Result of the operator *TDISPLAY* in the consistent time mode

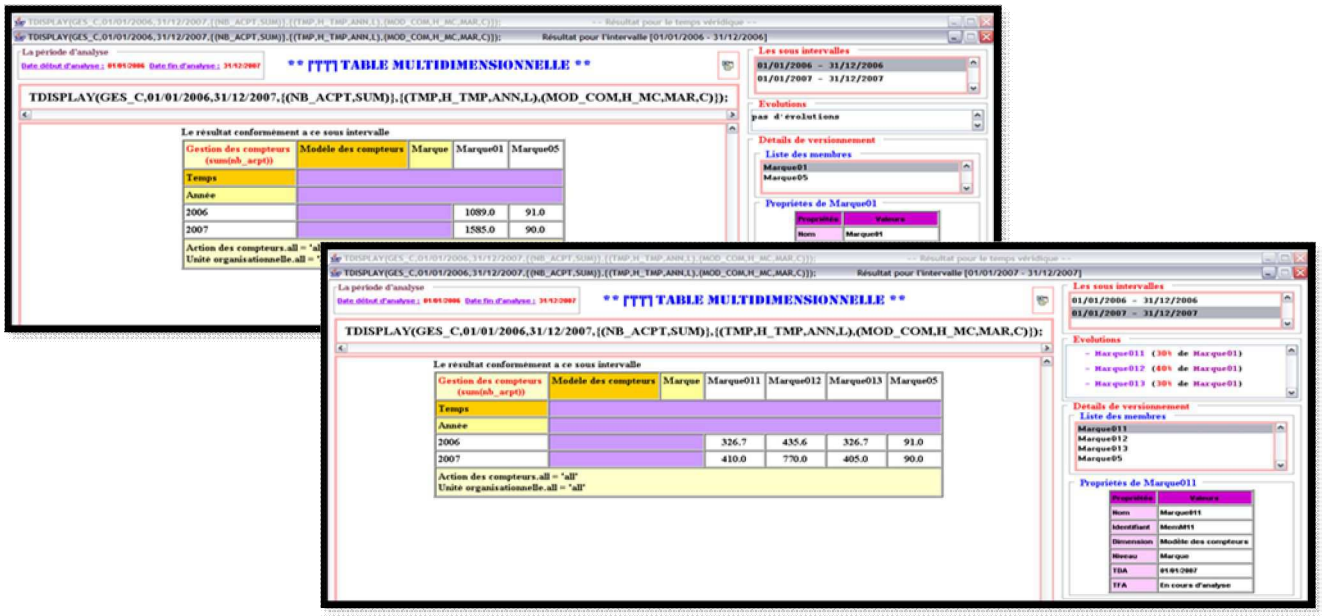

Figure 4: Result of the operator *TDisplay* according to the intervals [01/01/2006 - 31/12/2006] and [01/01/2007 - 31/12/2007]

4.2 TSLICE

The decision-maker wants to select the sum of the meter action numbers for the brands *Brand05* and *Brand013*. This request allows selecting a subset of the data visualized by the operator TDISPLAY when specifying a restriction condition. Figure 5 shows the formulation of this operator with the operator editor of our prototype.

Figure 6 presents the result in the consistent time mode for every sub-intervals and Figure 7 shows the result according to the time interval [01/01/2006 - 31/12/2006] with the application of transformation functions.

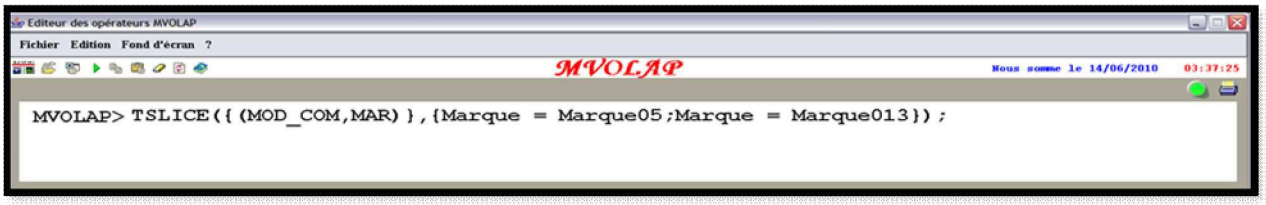

Figure 5: Formulation of the operator *TSlice*

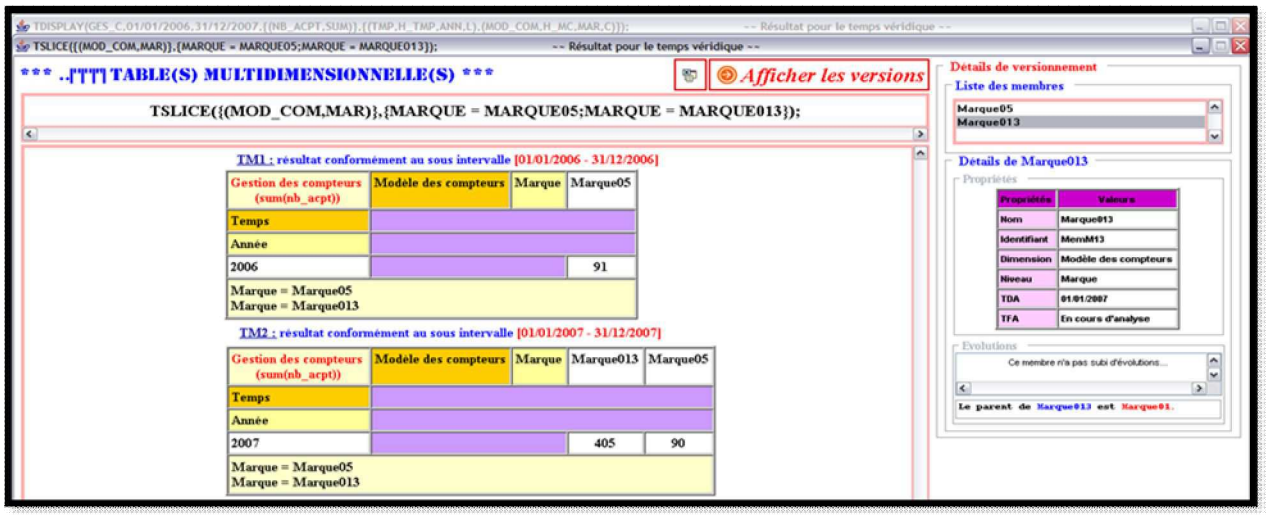

Figure 6: Result of the operator *TSlice* in the consistent time mode

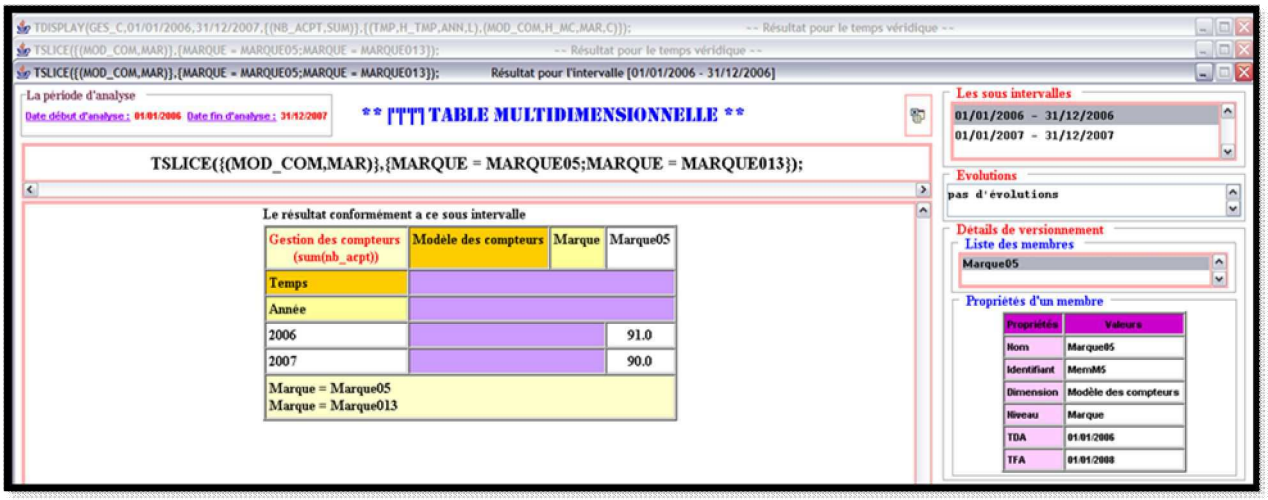

Figure 7: Result of the operator *TSlice* according to the interval [01/01/2006 - 31/12/2006]

In Table 1, we present a synthesis of the proposed MVOLAP operators.

Currently, we are studying the temporal extension and the implementation of others OLAP operators, such as Rotate, Push, Pull, Nest, etc.

Operator	Description
TDISPLAY	**Formal description:** *TDISPLAY (C, TDAn, TFAn, {(m_i, f_i)}, {(D_j, H_{jp}, N_{jk}, emp_j)})*
	Textual description: The temporal operator TDISPLAY allows visualizing cube data that correspond to a fact and several dimensions according to an analysis time interval.
TSLICE	**Formal description:** *TSLICE* ($\{MT_i^I\}$, {(D_j, N_{jk})}, {$restriction_j$}) = $\{MT_j^S\}$
	Textual description: the temporal operator TSLICE allows selecting data that satisfy a restriction condition towards a particular dimension while respecting the analysis time interval which is specified in the TDISPLAY operator.
TDRILLDOWN	**Formal description:** *TDRILLDOWN* ($\{MT_i^I\}$, {(D_j, N_{jm})}) = $\{MT_j^S\}$
	Textual description: the temporal operator TDRILLDOWN allows increasing the granularity of the data analyzed while respecting the analysis time interval which is specified in the TDISPLAY operator.
TROLLUP	**Formal description:** *TROLLUP* ($\{MT_i^I\}$, {(D_j, N_{jm})}) = $\{MT_j^S\}$
	Textual description: the temporal operator *TROLLUP* allows reducing the granularity of the data analyzed while respecting the analysis time interval which is specified in the TDISPLAY operator.

Table 1: MVOLAP operators

5. CONCLUSION

We have proposed an approach to extend classic OLAP operators to be useful in a multiversion DW context. We have redefined four classic operators (DISPLAY, SLICE, DRILLDOWN and ROLLUP) in a multiversion DW (MVDW) context. We have defined a textual query language using a set of operators TDISPLAY, TSLICE, TDRILLDOWN and TROLLUP.

We have defined and implanting under JavaCC (*Java **C**ompile to **C**ompile*) a grammar built according to the shape BNF, to allow the syntactic and lexical analysis of these operators.

This language allows answering queries implying one or several dimension member versions.

We have chosen to use a structure of multidimensional table which allows decision-maker to display in an ergonomic way the results of the MVOLAP operators. This table allows showing data according to the dimensions specified by the user and placed in lines and columns. We have developed a prototype, entitled MVOLAP (MultiVersions OLAP), allowing the formulation and the execution of the defined Temporal OLAP operators.

For future work we consider enrich our prototype by further temporal OLAP operators such as TRotate, TNest, etc. For this end, we are studying the temporal extension of the corresponding classic OLAP operators (Rotate, Nest, etc).

REFERENCES

[1] BLASCHKA M., SAPIA C., HOFLING G. (1999), « *On Schema Evolution in Multidimensional Databases* », *Proceedings of the DaWaK'99 Conference*, Italie, p. 153-164.

[2] BLIUJUTE R., SALTENIS S., SLIVINSKAS G., JENSEN C. S. (1998), « *Systematic Change*

Management in Dimensional Data Warehousing », *Proceedings of the International Baltic Workshop on Data Bases and Information Systems*, Riga, Latvia, p. 27-41.

[3] BODY M., MIQUEL M., BÉDARD Y., TCHOUNIKINE A. (2003), « *Handling Evolutions in Multidimensional Structures* ». IEEE 19th International Conference on Data Engineering (ICDE 2003), Bangalore, India.

[4] CABIBBO L., TORLONE R., (1998). « *From a procedural to a visual query language for OLAP*». In Proceedings of the 10th Int. Conf. on Scientific and Statistical Database Management (SSDBM 1998), IEEE Computer Society, p. 74-83.

[5] CODD E. F. A., CODD S. B., SALLEY C. T. (1993), « *Providing OLAP (On Line Analytical Processing) to Users-Analysts: An IT Mondate* ». Technical Report, E.F. Codd and Associates, 1993.

[6] EDER J., KONCILIA C., MORZY T., « *The COMET Metamodel for a temporal Data Warehouse* », In Proc. CAISE, (Toronto, Canada 2002), p. 83-99.

[7] GOLFARELLI M., LECHTENBÖRGER J., RIZZI S., VOSSEN G., (2006), « *Schema versioning in data warehouses: Enabling cross-version querying via schema augmentation* ». Data & Knowledge Engineering Volume 59, Issue 2, p. 435-459 Including: Sixth ACM International Workshop on Web Information and Data Management.

[8] GOLFARELLI M., RIZZI S., (2009) « *A Survey on Temporal Data Warehousing* », International Journal of Data Warehousing & Mining, 5 (1) p. 1-17.

[9] MENDELZON A.O., VAISMAN A.A., (2000), « *Temporal Queries in OLAP* ». Dans 26th international Conference on Very Large Data Bases (VLDB 2000), Cairo, Egypte, p. 242-253.

[10] MORZY T., WREMBEL R., (2004), « *On Querying Versions of Multiversions Data Warehouse* ». Proceedings of the 7th ACM Int. Workshop on Data Warehousing and OLAP (DOLAP 2004), p. 92-101.

[11] RAVAT F., TESTE O., ZURFLUH G. (2002), « *Langage pour bases multidimensionnelles : OLAP-SQL* ». Revue des Sciences et Technologies de l'Information.

[12] RAVAT F., TESTE O., RONAN T., ZURFLUH G. (2008), « *Algebraic and Graphic Languages for OLAP Manipulations».* International Journal of Data Warehousing & Mining, 4 (1) p. 17-46.

[13] RIZZI F., MATTEO G., (2007), « *X-Time: Schema Versioning and Cross-Version Querying in Data Warehouses* ».

[14] VAISMAN A. and O., MENDELZON A., (2002), « *A Temporal Query language for OLAP* ». G. Grahne and G. Ghelli (Eds.): DBPL 2001, LNCS 2397, p. 78–96.

[15] ZOUARI I., GHOZZI F., BOUAZIZ R. (2008), « *Impact of Nomenclature Evolution on Data Warehouse Versioning* » (in French). Journal *RSTI-ISI*, Vol. 13/6-2008, p. 85-114.

[16] ZOUARI I., GHOZZI F., BOUAZIZ R. (2009), « *New Evolution Operators for Multiversion Data Warehouse* ». In International Conference MIPRO Business Intelligence System, p. 175-180.