Developing Model-Driven Quality-Aware Data Warehouses with a UML Profile

Omran Al-Badarneh, Devoteam, Riyadh, Saudi Arabia
Asim El Sheikh, Al-Ahliyya Amman University, Jordan
Amer Al-Badarneh, Jordan University of Science and Technology

Abstract

Nowadays, Model-Driven Architecture (MDA) is playing a major role in today's system's development methodologies. Data Warehouse (DW) researchers try to apply MDA standard on DW development project. After surveying the related literature, we found that the main focus of MDA and DW is not as much on performance aspects of DW as on database issues. With the aim to facilitate building a quality-aware DW, we present an extension of the Unified Modeling Language (UML) to create performance UML2 profile and its corresponding metamodel. The profile is defined by a set of new stereotypes to enable DW team to elegantly represent the DW performance requirements with MultiDimensional (MD) properties at the conceptual level. The proposed approach is MDA compliant and uses Query-View-Transformation (QVT) and Model-to-Text (MTL) languages for automatic generation of Platform Specific Model (PSM) and the implementation code in target platform. One key advantage of the proposed approach is that the conceptual modeling of quality-aware DWs is we accomplished independently of the DW target platform, allowing the implementation of the regarded DWs on most of commercial database management systems. Finally, this work has been exemplified and validated by developing a case study in which the proposed profile is used. The outcomes from the validation process give clear evidences that the proposed development framework for DWs outperforms other development methods that do not handle performance requirements at early stages of system development.

Keywords: Data warehousing, Model driven Architecture (MDA), Platform Independent Model (PIM). Platform Specific Model (PSM), Common Warehouse Metamodel (CWM), XML Metadata Interchange (XMI)

1. INTRODUCTION

In recent years Data Warehouse (DW) has become known as a powerful technology for integrating heterogeneous distributed operational data sources into a comprehensive analytical system to predict and make decisions for near future business reengineering [PAIM et al. 2002]. Designing such systems is different from the design of the operational systems that supply data to the warehouse. DW team should elicit functional requirements of decision makers and also the structure requirements of the source-provider systems that will be used in a complex process of extracting, transforming, and aggregating data. Moreover, DW team should also focused on non-functional requirements manage to deploy a system that precisely, timely integrates with a number of heterogeneous distributed data sources; presents analytical results in a reliable, accurate format; offers flexibility to end users to execute time-efficient ad-hoc queries[PAIM et al. 2002].

Many data warehousing projects fail to provide the needed information in the right time because the performance aspects of the DW project are considered in the final stages of implementation after end users start blaming that the system is not working as expected. So this paper suggests that performance aspects (Non-Functional Requirements) of the DW should be addressed in the early stages of design by modeling them at conceptual level while modeling the user requirements (Functional Requirement) to have a quality-aware system that meet both operational and strategic visions of organization. Hence, both types of requirements have to be wrapped up in a multidimensional model to meet corporative analytical requirements and provide decision-support functionality as well as strong performance constraints [PAIM et al 2002].

To handle DW performance issues we have to follow a methodological approach. One of the current trends in software development that gains more importance is the model-driven approach. The ideas behind the Model-Driven Architecture (MDA) [OMG 2003b] proposal can facilitate and improve the DW performance solutions. Consequently, to solve DW performance problems, a development framework based on the MDA principles could be used. MDA considers models as first class elements during system design and implementation and establishes a separation of the development process in three abstraction levels, namely CIM, PIM and PSM [MARCOS et al. 2008].

One key advantage of using the MDA is the definition of mappings between the models defined in each level, what makes possible the automation of the development process. Another key advantage is that we can accomplish the conceptual modeling of quality-aware DWs independently of the target platform where the DW has to be implemented, allowing the implementation of the regarded DWs on any commercial database management system [EDUARDO et al. 2006a] [MARCOS et al. 2008].

As mentioned by the UML superstructure specification document [OMG 2005b], there are two extension mechanisms for UML 2.0: (1) the profile mechanism, which is not a first-class extension mechanism by which the modification of existing metamodels is not allowed and (2) the first-class extensibility which is handled through Meta Object Facility (MOF) [OMG 2008], in which there are no restrictions on modifying a metamodel. It is possible to add and remove metaclasses and relationships as is necessary [EDUARDO et al. 2006b] [OMG 2005b]. Hence, to achieve the goals of this work that address the performance requirements at early stages of DW system development using MDA standards [OMG 2003b][ANNEKE et al. 2003], an extension of the unified modeling language (UML) has been created using UML profile. This profile uses stereotypes to represent main DW performance concepts such as Materialized View, Bitmap Index, Normal Index, Function-Based Index and Partitioning.

UML [OMG 2009] has been used because UML is widely accepted as standard for Object-Oriented (OO) modeling language known by most of designers and developers which minimizes the learning curve for understanding new notations or methodologies [SERGIO et al. 2006]. On the other hand, UML is an extensible language that help designer to model new concepts for new applications such as web application, business modeling, database application, service oriented architecture (SOA) application, etc. [MARCOS et al. 2008], [CHRISTOF et al. 2007].

The rest of the paper is organized as follow. Section 2 presents the related works. Section 3 presents the new concepts involved in the proposed performance profile. In Section 4, the Platform Independent Model (PIM) UML2 metamodel with its new stereotypes is explained. Section 5 presents the proposed performance RDBMS Platform

Specific Metamodel. In section 6 we implement the case study and present the validation results. Finally, Section 7 presents the main conclusions and future works.

2. Related Work

This work is a based on a previous comprehensive survey conducted by the authors that targeted the research direction in MDA and DW. Most of MDA-related works that have been conducted on DWs can be classified into the following groups

1. Creating CWM-based DW Modeling Tools [KUMPON et al. 2003].
2. Specifying Metamodel transformations for DW design [LEOPOLDO et al. 2005].
3. Extending UML for MultiDimensional Modeling [SERGIO et al. 2006].
4. MDA-related DW frameworks [JOSE-NORBERTO et al. 2005][JOSE-NORBERTO et al. 2008].
5. Securing DW using MDA [JUAN et al. 2009a][RODOLFO et al. 2006][EDUARDO et al. 2006a][EMILIO et al. 2007a][EMILIO et al. 2006][EDUARDO et al. 2006b][EMILIO et al. 2008a] [EMILIO et al. 2008b] [EMILIO et al. 2007b].
6. Designing Spatial Data Warehouse using MDA Techniques [OCTAVIO et al. 2008].
7. Conceptual OLAP Platform-independent Queries [JESUS et al. 2008].
8. MDA Framework for Designing Spatial DWs [OCTAVIO et al. 2009].
9. MDA Secure Engineering Process for DWs [JUAN et al. 2009b].

Based on this in-deep study and to the best of our knowledge, none of the previous works addresses modeling the DW performance aspects at conceptual level using MDA. Hence, we consider this performance issue a challenging open problem that can be handled using MDA. [PAIM et al. 2002] addressed the enhancement of Data Warehouse design by extending the Non-Functional Requirement (NFR) Framework proposed by [CHUNG et al. 2000]. Catalogues of major DW NFR types and related operational methods has been defined. [PAIM et al. 2002] highlighted the importance of set of quality factors such as integrity, accessibility, performance, and other domain-specific Non-Functional Requirements (NFRs) that governs the success of the DW project. Based on [PAIM et al. 2002], we identified the needed performance objects that can be conceptually modeled at early stages of system development to enhance the DW performance.

3. Profile Concepts

This section presents the new concepts that will be involved in the proposed performance UML profile. Figure 1 gives a big picture about these concepts which will be described in detail in this section.

3.1 Fact

In data warehousing, information is structured into facts and dimensions using MultiDimensional Model [ALBERTO et al. 2001][KIMBALL 2002]. A fact can be seen as a point in a MultiDimensional space to which some measurable business facts such as “quantity sold”, “amount sold” are assigned [JENS et al. 2003]. Most of business

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

processes have significant measures that contained in a fact such as “unit cost”, “unit price”. So we can define a fact as an item of interest for an organization that has a set of attributes called measures or fact attributes. This set of measures is linked to a set of dimensions that surrounding the fact [SERGIO et al. 2006]. As presented in Figure 1, we have represented this concept using “OmFact” stereotype. In the case study as presented in Figure 2, we have conceptually defined a fact for a sales process at a computer store. We named the fact “SALES” and it contains “amount_sold” as fact attribute.

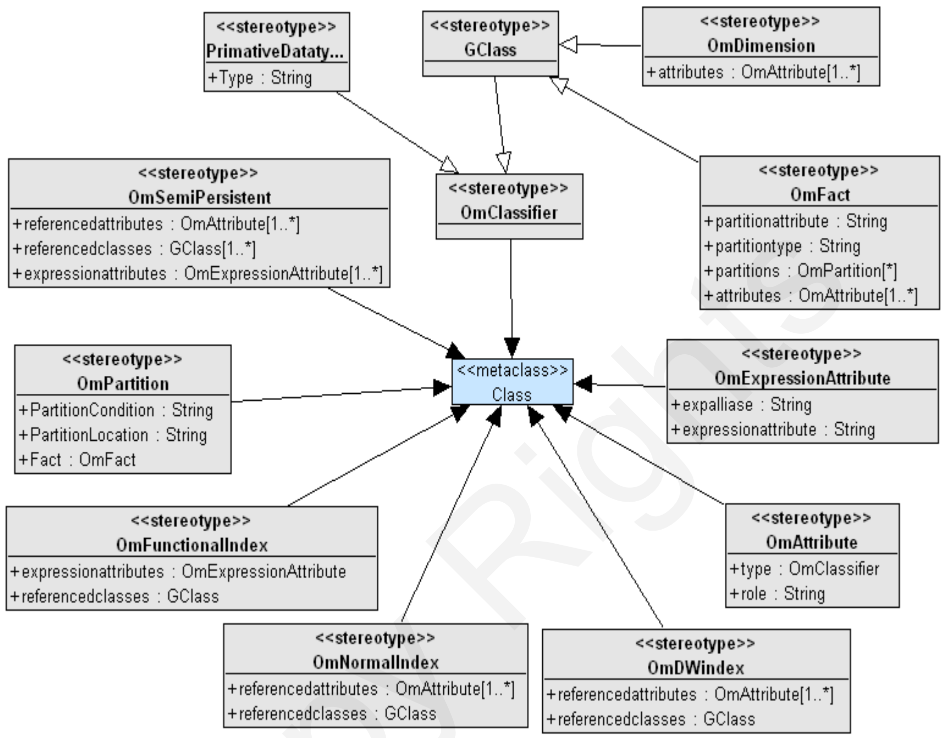


Figure 1: PIM-level Performance Metamodel for DW.

3.2 Dimension

Dimensions provide the context in which facts are to be analyzed. Every fact has set of surrounding dimensions (product, customer, time, etc.) represents the area of interest that is going to be analyzed [MARK 2003]. Unlike Fact concept, dimensions are characterized by descriptive attributes (customer name, product description), which are usually called dimension attributes [INMON 2005]. As presented in Figure 1, we have represented this concept using “OmDimension” stereotype. The case study as presented in Figure 2 contains five dimensions surrounding the SALES fact. These dimensions are TIME, CUSTOMERS, PRODUCTS, CHANNELS, and PROMOTIONS.

3.3 DW Index

DW index is a concept that represents an indexing approach suitable for Data warehousing applications that link large facts with its surrounding small dimensions [MICHAL et al. 2009]. In this case, low cardinality attributes (small number of distinct values) such as Gender (male, female) or State (MD, NY, etc.) are included in the query of interest. For such applications, DW indexing provides a reduced response time for such types of queries [PATRICK et al. 1997]. DW index is a concept that will be mapped into Bitmap Index which has a significant space and performance advantage over other structures for such data. Bitmap Index uses bitmaps to answer queries by executing bitwise logical operations (XOR, NOT, OR, AND) on these bitmap [GUADALUPE et al. 2009][NAVNEET et al. 2009]. As depicted in Figure 1, we have represented this concept using “OmDWindex” stereotype. In the case study as presented in Figure 3, while creating the PIM input model, we have used this stereotype to model the following indices: SALES_CHANNEL_BIX, SALES_CUST_BIX, SALES_PROD_BIX, SALES_PROMO_BIX, SALES_TIME_BIX, CUSTOMERS_GENDER_BIX and CUSTOMERS_MARITAL_BIX.

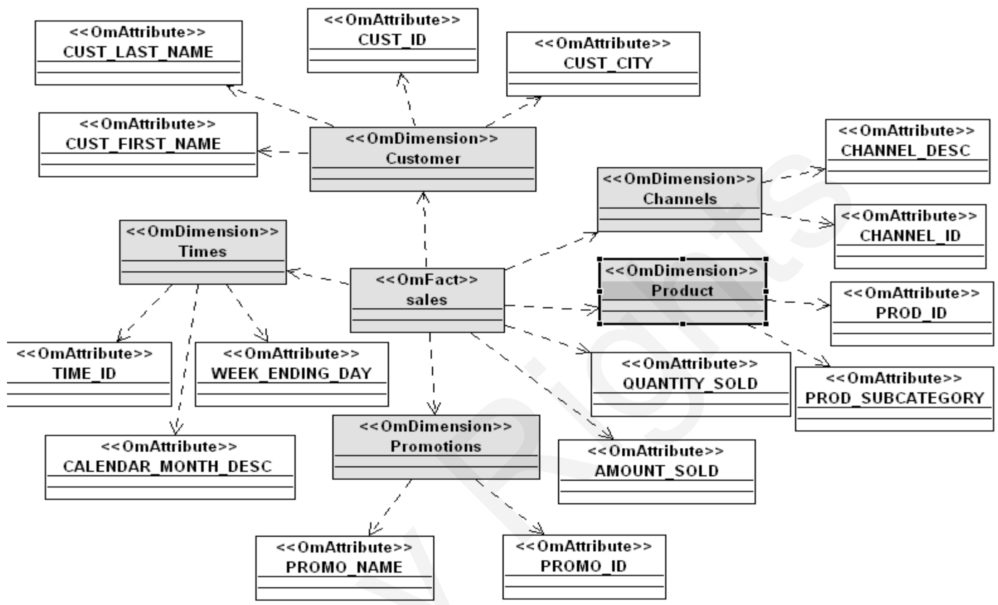


Figure 2: Modeled Star Schema.

3.4 Normal Index

Normal Index is a concept that will be used to help us to conceptually model indices that will be mapped to a normal B-tree index in relational database. Even though this type of concept is not heavily used in DW environments [GOETZ 2006], it is needed to create a normal B-tree index for some attributes used mainly by dimensions. This concept is suitable to be used to index attributes that have high cardinality (large number of distinct values) such as “product number” and “customer number” [IBRAHIM et al. 2006]. In the proposed metamodel, as depicted in Figure 1, we have represented this concept using “OmNormalindex” stereotype. In the case study as presented by Figure 3, while creating the PIM input model, we have used this stereotype to model one index “CUST_LAST_NAME_IDX” to index “LAST_NAME” attribute of “CUSTOMERS” class.

3.5 Function-Based Index

A Function-Based Index is a concept that is used to create an index that solves the problem of retrieving case-sensitive information using case-insensitive predicates. In Data warehousing environments, most of end users perform ad hock queries to retrieve data using case-insensitive predicates. For example we may need to retrieve all sales transactions that occurred in specific city such as “Amman” using where initcap (city) =

‘Amman’. Because “City” information may be entered in different format such as “AMMAN” or “amman” and we need records related to Amman regardless of its case, we have to have a Function-Based Index based on this formula Initcap (city) to effectively retrieve the needed records. Indexing “City” attribute using traditional Normal Index will not solve the problem and will cause a full table scan and more I/O operations. So for such situation, using Function-Based Indexes will be our super silver bullet for optimizing our Data Warehouse to do a minimum amount of I/O to get the needed information [RICHARD 1999]. In the proposed metamodel, as depicted in Figure 1, we have represented this concept using “OmFunctionalIndex” stereotype. In the case study as presented by Figure 3, we have created UPPER_CUST_CITY_IDX as an instance of “OmFunctionalIndex” Class in the PIM input model; this instance has been mapped to relational Function-Based Index in the proposed PSM RDBMS model.

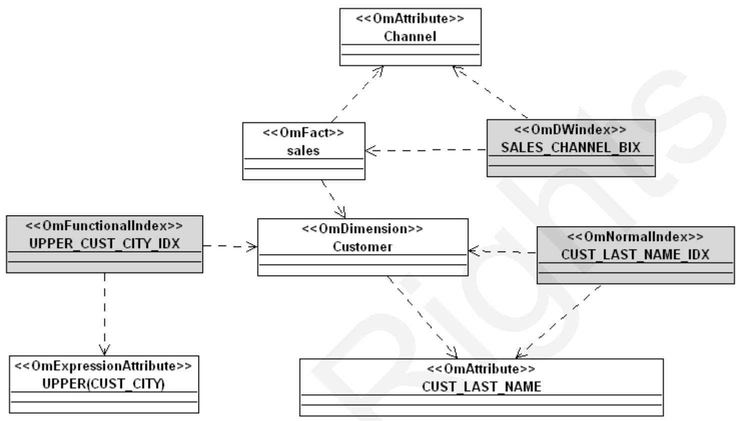


Figure 3. Modeled Indices.

3.6 Semi Persistent

Semi Persistent is a concept that will be mapped to a relational Materialized View [LEONARDO et al. 2009]. This concept is heavily used in Data warehousing environment as well as in Online Transaction Processing Systems (OLTP) to cache expensive queries in persistent state. This concept is one of the most important performance tuning tools that gives the facility to pre-join complex data source and pre-compute summaries for super-fast response time by reducing repetitive I/O [GORET et al. 1999]. Unnecessary long-full-table scan can be avoided while doing aggregations and summarization by accessing the needed information from already existing Materialized View that contains the expected result [MING et al. 2007]. In the proposed metamodel, as depicted in Figure 1, we represented this concept using “OmSemiPersistent” stereotype. As presented in Figure 4, while validating the proposed profile, we have created CAL_MONTH_SALES_MV as instances of “OmSemiPersistent” Class in the PIM input model; this concept has been mapped to relational Materialized Views in the proposed PSM RDBMS model.

3.7 Partition

Partitioning is the process of dividing a logical data source into distinct parts for the sake of increasing the performance and availability by locating each partition on different file system (Hard drive), different databases or servers [GEORGE et al. 2008]. This concept is deeply used in Data Warehouse environments to spread very large data source

(Fact table) into set of partitions to reduce the amount of data read so that overall response time is reduced by accessing only the needed partitions and skipping the others [VINCENT et al. 2003]. Partitioning is usually done for persistent objects such as database tables. There are two major forms of partitioning [SANJAY et al. 2004]:

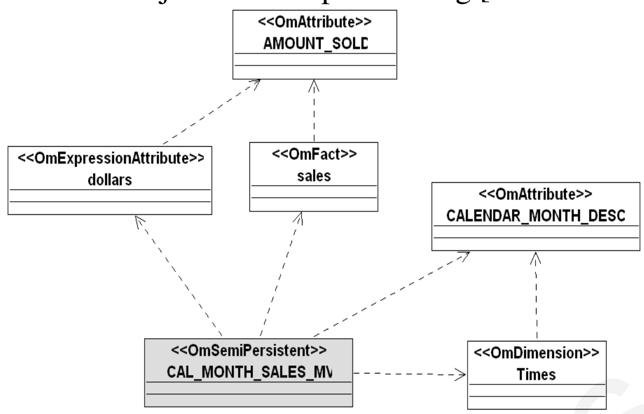


Figure 4: Modeled Materialized View

- Horizontal Partitioning - this form of partitioning divides available data records into distinct group of physical datasets that can be accessed individually or collectively. All attributes defined to a data source are found in each set of partitions so no actual attributes are missing. Most of partitioning operations in data warehousing use this form of partitions.
- Vertical Partitioning - this partitioning approach is used to reduce the width of a target data source by dividing the data source vertically so that only certain attributes are included in a one partition and the remainder attributes are included in another partition, with each partition including all rows. An example of vertical partitioning might be a data source that contains a number of very wide text attribute that aren't used often being broken into two partitions that has the most referenced attributes in one partition and the seldom-referenced text data in another.

Horizontal partitioning can be done in three main modes [SANJAY et al. 2004]:

- Range - this partitioning mode allows us to specify various ranges for which data is assigned. For example, we may create a partitioned data source that is divided by four partitions that contain data for the 1970's, 1980’s, 1990's, and everything beyond and including the year 2000.
- Hash - this partitioning mode allows us to separate data based on a computed hash key that is defined on one or more data source attributes, with the end goal being an equal distribution of values among partitions.
- List - this partitioning mode allows us to partition data based on a pre-defined list of values. For example, we may create a partitioned data source that contains three partitions based on the main cities in the regions. Irbid, Jerash, and Ajloun for North_Region. Amman, Zarka and Salt for Mid_region. Karak, Ma’an and Aqaba for South_region.

To conceptually model partitions in the PIM model, as depicted in Figure 1, we have created a new class with a stereotype name “OmPartition” to create partitions for OmFact classes. Each class may have two or more partitions. As presented in **Figure 5**, while validating the proposed profile, we have created 20 partitions for the "SALES" fact class using range partition mode on "TIME_ID" attribute.

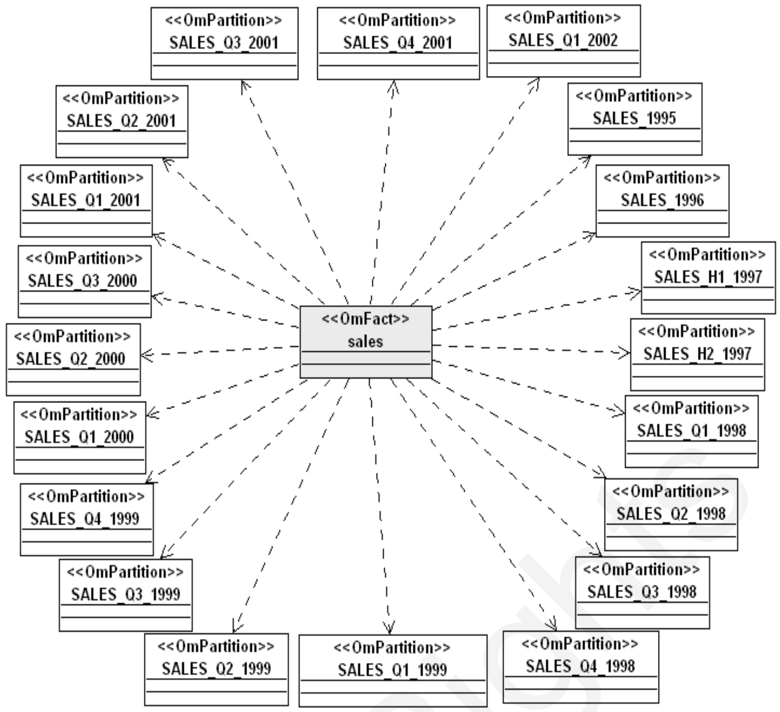


Figure 5: Modeled Data partitions

4. Expression Attribute

Expression attribute is a concept used to model complex attribute such as the sum of two attribute or create function-based attribute such as uppercase (customer_name). This concept is not a major Data Warehouse performance technique; but it is a mandatory model element used to handle complex expression data types at conceptual level. This concept will be used by semi-persistent concept (Materialized View) to formulate a complex attribute, if needed. Moreover, it is used also by Function-Based Index concept to determine the associated function for the index key.

To conceptually model "Expression Attribute" in the PIM model, the metamodel has a new class with a stereotype name "OmExpressionAttribute" to create an expression attribute for OmFunctionIndex classes or OmSemiPersistent classes. While validating the proposed UML profile, as presented in Figure 4, we have created an expression attribute named "dollars" to represent the sum of the amount sold "sum (amount_sold)"; this attribute is one of CAL_MONTH_SALES_MV semi-persistent class attributes.

5. UML Performance Profile

In this section we show the main stereotypes needed to represent the concepts described previously. Table 1 lists the stereotypes that we have defined, the related concept for each stereotype, as well as the base UML metaclass from which stereotypes are derived and brief description of the stereotypes.

Table 1. Stereotypes for the UML Performance Profile.

NOTATION	RELATED CONCEPT	BASE UML META CLASS	DESCRIPTION
<<OmFact>>	Fact	Class	Represents the main Fact table
<<OmDimensin>>	Dimension	Class	Represents the surrounding Dimension tables
<<OmSemiPersistet>>	Semi Persistent	Class	Represents Materialized View
<<OmNormalIndex>>	Normal Index	Class	Represents a Normal Btree index
<<OmFunctionalIindex>>	Function-Based Index	Class	Represents a Function-Based Index
<<OmDWindex>>	DW index	Class	Represents a Bitmap Index
<<OmPartition>>	Partition	Class	Represents database partition for table
<<OmExpressionAttribute>>	Expression Attribute	Class	Represent expression column for Materialized View or Function for Function-Based Index

6. Proposed RDBMS Platform Specific Metamodel (PSM)

This section presents the proposed Platform Specific Metamodel which is an updated version of the RDBMS PSM metamodel created by Eclipse Tutorials. Because most of data warehousing systems are running under relational databases, we choose our Platform Specific Metamodel to be an RDBMS one.

The first purpose of this metamodel is to enable us to store all needed information about the database objects (schema, tables, Materialized Views, indices, partitions) that we will generated from PIM model using the transformation process. Moreover, this metamodel will be used as an input to the code generation process. This code generation process will create a database script file (xxxxx.sql) that will be used to create all Data Warehouse objects including the performance-related objects such as Table Partitions, Bitmap Indices and Materialized Views. Figure 6 shows a UML class diagram for all needed classes that compose the proposed RDBMS metamodel.

7. Validation and Analysis

This section demonstrates how we build the case study to validate the UML performance profile. Next a detail analysis for the validation results is presented. This work validates and analyzes the performance profile by:

(i) Creating a DW system that does not contain any object of the performance profile.
(ii) Creating a DW system using the proposed performance profile.
(iii) Comparing the number of I/O operations between the two systems.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

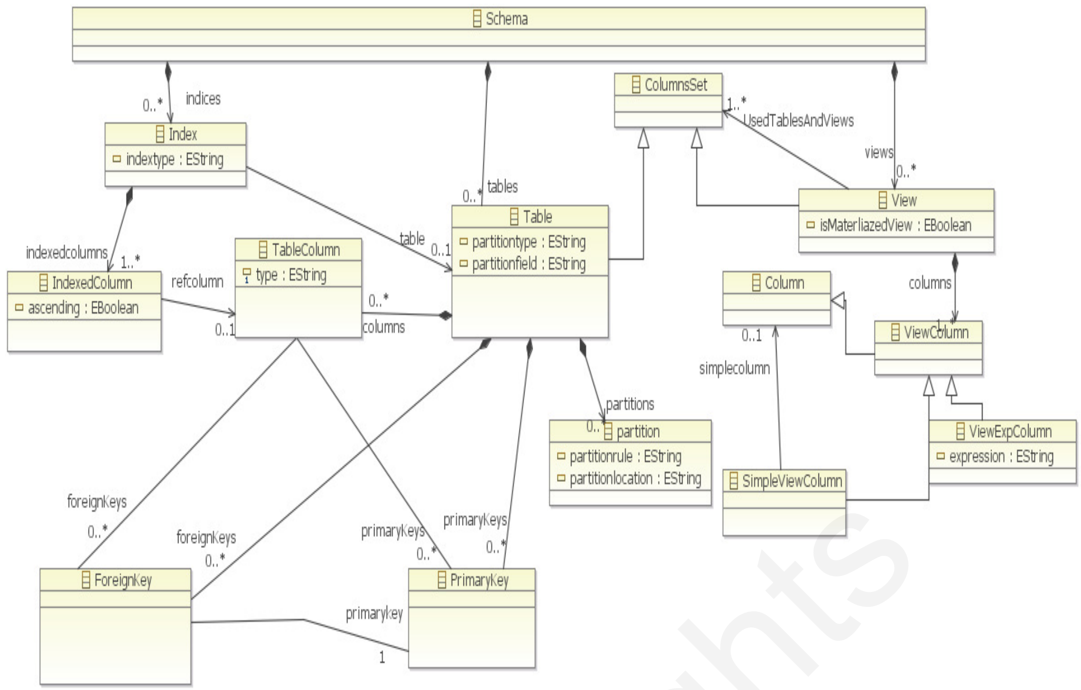


Figure 6. Proposed RDBMS PSM Metamodel.

The case study starts with developing of a PIM UML model (see Figure 2, Figure **3**, Figure **4** and Figure **5**) that describes at conceptual level a quality-aware Data Warehouse system. This conceptual PIM model is a subtype of the proposed PIM metamodel (see Figure 1). From the conceptual PIM model we built a physical PSM model using QVT model transformation. From the physical PSM model, we generate the SQL code to create the DW system using Acceleo Model to Text (M2T) language.

To ease the process of creating and populating such DW with relevant information, we use the Data Warehouse example provided by Oracle to create a similar Data Warehouse system [ORACLE 2009]. Oracle DW schema is a very practical DW example because it include all performance elements that are needed to have a highly efficient Data Warehouse system. Table 2 presents these database objects that are modeled conceptually based on the proposed PIM performance metamodel and the corresponding relational database objects that will be mapped to using QVT transformation program.

7.1 Selecting Modeling and Database Tools

To implement the case study, we need a modeling tool for building new metamodels or updating existing metamodels and support Model-to-Model (M2M) transformation using Query-View-Transformation (QVT) language as well as support Model-to-Text (M2T) transformation based on MDA standard.

Eclipse is a well-known tool that supports most of MDA standards that are needed in this work. Moreover Eclipse allows us to depict models in different formats, using tree or diagram format. In addition, Eclipse has an M2T language which is based on MDA standard named “Acceleo”. This language used to transform the proposed PSM model to SQL code. Furthermore, Eclipse supports the OMG QVT standard for M2M

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

transformations. So based on these advantages of Eclipse, we used Eclipse as our modeling tool. We used Eclipse to:

1. Build the proposed PIM performance metamodel.
2. Build the proposed PSM performance RDBMS metamodel.
3. Build the PIM input model example for the case study.
4. Develop the M2M QVT transformation program.
5. Develop the M2T transformation program to generate the SQL code.

To build the DW systems, we used TOAD utility from QUEST SOFTWARE [QUEST 2009] which can be connected to Oracle database and provide all performance tuning metrics for any executed query.

Table 2. DW PIM model elements.

Model element name	Concept	Stereotype	Mapped Relational DB object
SALES	Fact	<<OmFact>>	Table
CUSTOMER	Dimension	<<OmDimension>>	Table
PROMOTION	Dimension	<<OmDimension>>	Table
TIME	Dimension	<<OmDimension>>	Table
PRODUCT	Dimension	<<OmDimension>>	Table
CHANNELS	Dimension	<<OmDimension>>	Table
CAL_MONTH_SALES_MV	semipersistent	<<Omsemipersistent>>	Materialized View
SALES_CHANNEL_BIX	DW Index	<<OmDWIndex>>	Bitmap Index
UPPER_CUST_CITY_IDX	Function-Based Index	<<OmFunction Index>>	Function-Based Index
CUST_LAST_NAME_IDX	Normal Index	<<OmNormalIndex>>	B-tree index
SALES_1995	Partition	<<OmPartition>>	Partition
SALES_1996	Partition	<<OmPartition>>	Partition
SALES_H1_1997	Partition	<<OmPartition>>	Partition
SALES_H2_1997	Partition	<<OmPartition>>	Partition
SALES_Q1_1998	Partition	<<OmPartition>>	Partition
SALES_Q2_1998	Partition	<<OmPartition>>	Partition
SALES_Q3_1998	Partition	<<OmPartition>>	Partition
SALES_Q4_1998	Partition	<<OmPartition>>	Partition
SALES_Q1_1999	Partition	<<OmPartition>>	Partition
SALES_Q2_1999	Partition	<<OmPartition>>	Partition
SALES_Q3_1999	Partition	<<OmPartition>>	Partition
SALES_Q4_1999	Partition	<<OmPartition>>	Partition
. . .	. . .	. . .	. . .
SALES_Q4_2004	Partition	<<OmPartition>>	Partition

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

7.2 Measuring I/O Operations

Because DW system represents a very large database and normally deals with multi-millions of records, doing more I/O operation will degrade the system performance. I/O operation can be read logically from memory or physically from hard disk. So we compare the number of I/O operations for a specific query between the two systems. The validation process covers the following performance elements: Materialized View, Table Partitioning, Bitmap Index, Function-Based Index and Normal Index.

7.3 Validating the Usage of Table Partitioning

To validate the usage of Table Partitioning, we use the following query that retrieves the sum of amount sold for every month.

```
SELECT t.calendar_month_desc,sum(s.amount_sold) AS
dollars
FROM  sales s, times t
WHERE  s.time_id = t.time_id
GROUP BY t.calendar_month_desc
```

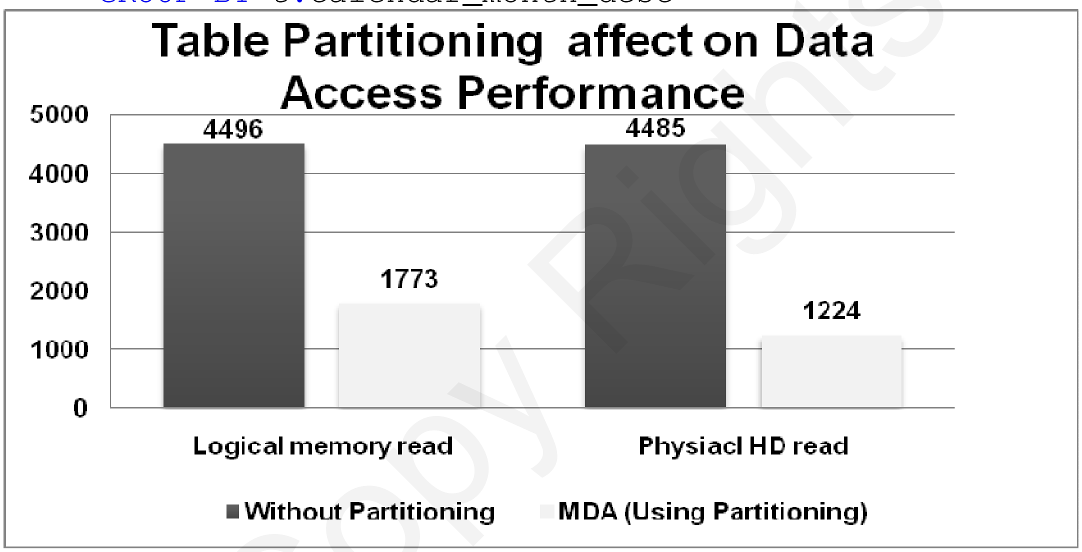
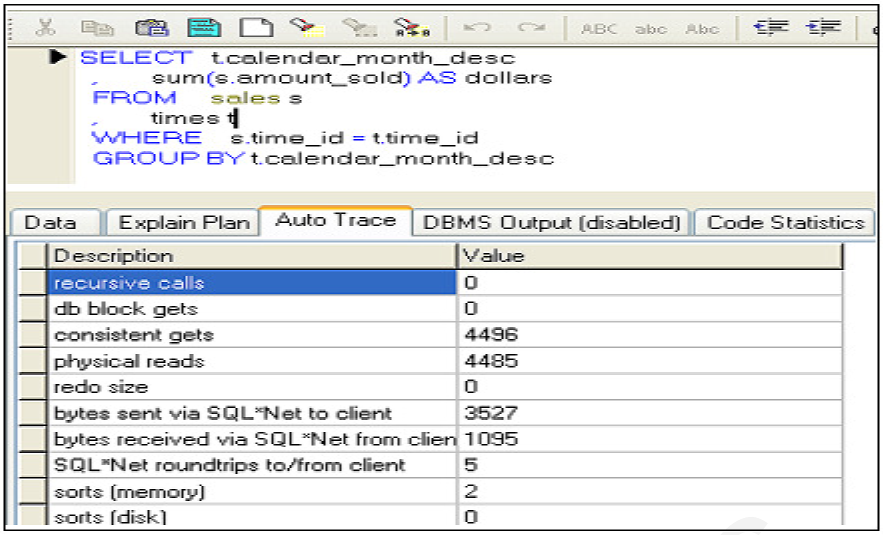


Figure 7. Performance affect for Table Partitioning.

Figure 7 shows that without using Table Partitioning the DW system used 8981(sum of logical reads and physical reads) read operations to retrieve the requested information while our MDA approach gets the same result using 2997 read operations; this means our MDA approach outperforms traditional approach by 66%. Figure 8 below shows the performance metrics after executing the mentioned query without using Table Partitioning; as we can see the consistent gets (logical reads) is 4496 and the physical reads is 4485. The reason for this large number of I/O operations is the use of "FULL TABLE SCAN " as mentioned by the SQL execution plan depicted by Figure 9 below.

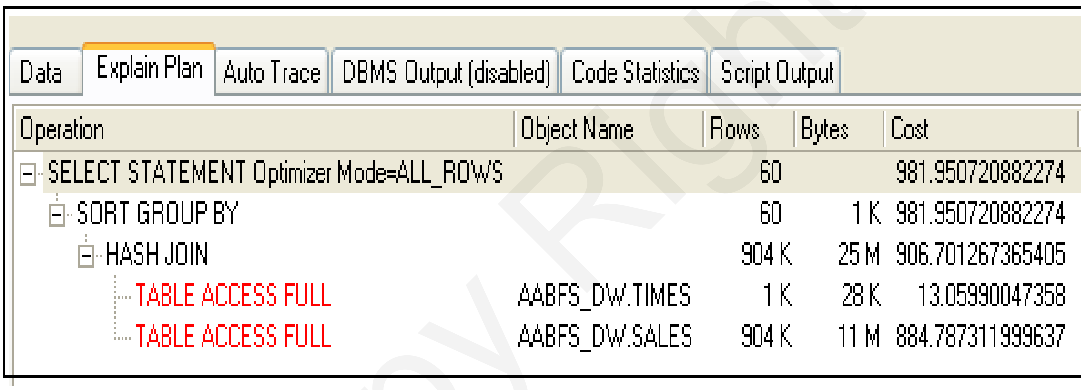
https://sites.google.com/site/ijcsis/
ISSN 1947-5500



```
SELECT  t.calendar_month_desc
      , sum(s.amount_sold) AS dollars
FROM    sales s
      , times t
WHERE   s.time_id = t.time_id
GROUP BY t.calendar_month_desc
```


Description	Value
recursive calls	0
db block gets	0
consistent gets	4496
physical reads	4485
redo size	0
bytes sent via SQL*Net to client	3527
bytes received via SQL*Net from clien	1095
SQL*Net roundtrips to/from client	5
sorts (memory)	2
sorts (disk)	0

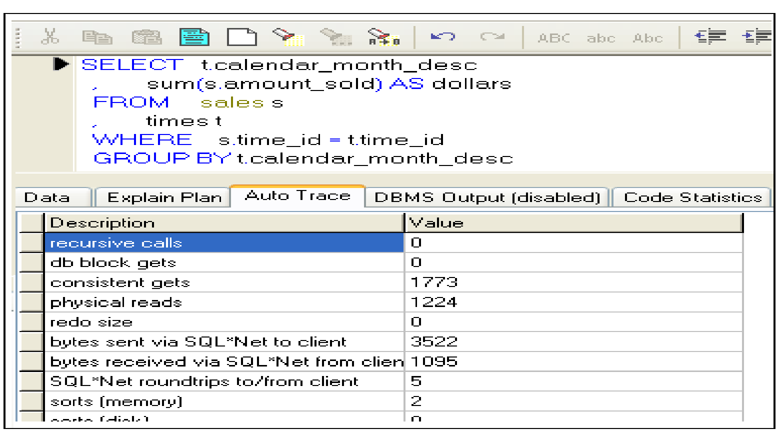
Figure 8. I/O operations without partitioning.



Operation	Object Name	Rows	Bytes	Cost
SELECT STATEMENT Optimizer Mode=ALL_ROWS		60		981.950720882274
SORT GROUP BY		60	1 K	981.950720882274
HASH JOIN		904 K	25 M	906.701267365405
TABLE ACCESS FULL	AABFS_DW.TIMES	1 K	28 K	13.05990047358
TABLE ACCESS FULL	AABFS_DW.SALES	904 K	11 M	884.787311999637

Figure 9: Execution Plan without using partitioning.

Figure 10 below shows the performance metrics after executing the mentioned query using Table Partitioning; as we can see the consistent gets (logical reads) is 1773 and the physical reads is 1224. The reason for this reasonable number of I/O operations is the use of partitions as mentioned by the SQL execution plan as depicted in Figure 11 below.



```
SELECT  t.calendar_month_desc
      , sum(s.amount_sold) AS dollars
FROM    sales s
      , times t
WHERE   s.time_id = t.time_id
GROUP BY t.calendar_month_desc
```


Description	Value
recursive calls	0
db block gets	0
consistent gets	1773
physical reads	1224
redo size	0
bytes sent via SQL*Net to client	3522
bytes received via SQL*Net from clien	1095
SQL*Net roundtrips to/from client	5
sorts (memory)	2

Figure 10. I/O operations with Table Partitioning.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

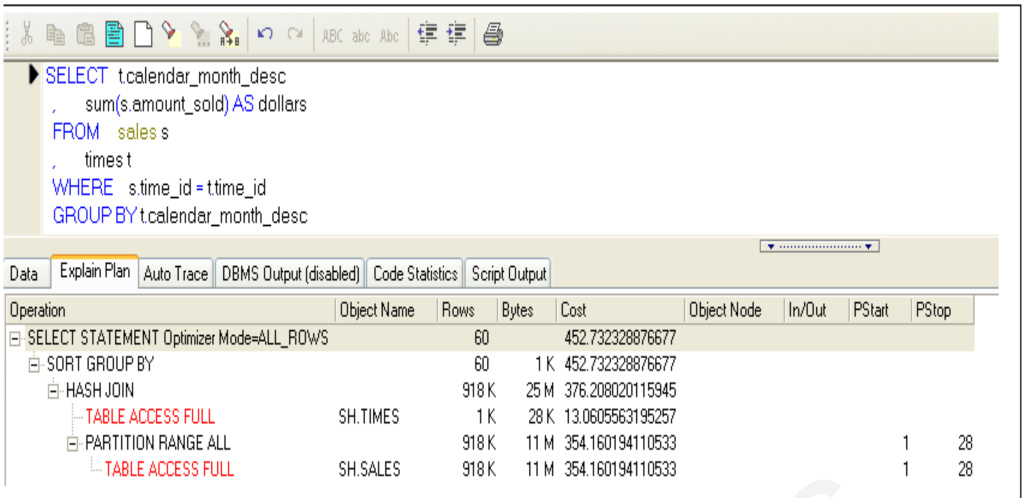


Figure 11. Execution Plan with Table Partitioning.

Due to the scope constraints of this paper, Table 3 and Figure 12 present a performance comparison between our MDA approach and non-MDA approaches for all proposed DW performance objects. The outcomes from the validation process gave clear evidences that the proposed development framework for DW will outperform any development method that does not handle performance requirement at early stages of system development.

Table 3. Performance Analysis

Performance Object	**Other approaches Read Operations**	**Our approach Read Operations**	**Our approach effectiveness**
Partition	8981	2997	66%
Materialized View	9138	678	92%
Bitmap Index	3387	344	89%
Normal Index	11837	3387	71%
Function-Based	12005	3696	69%

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

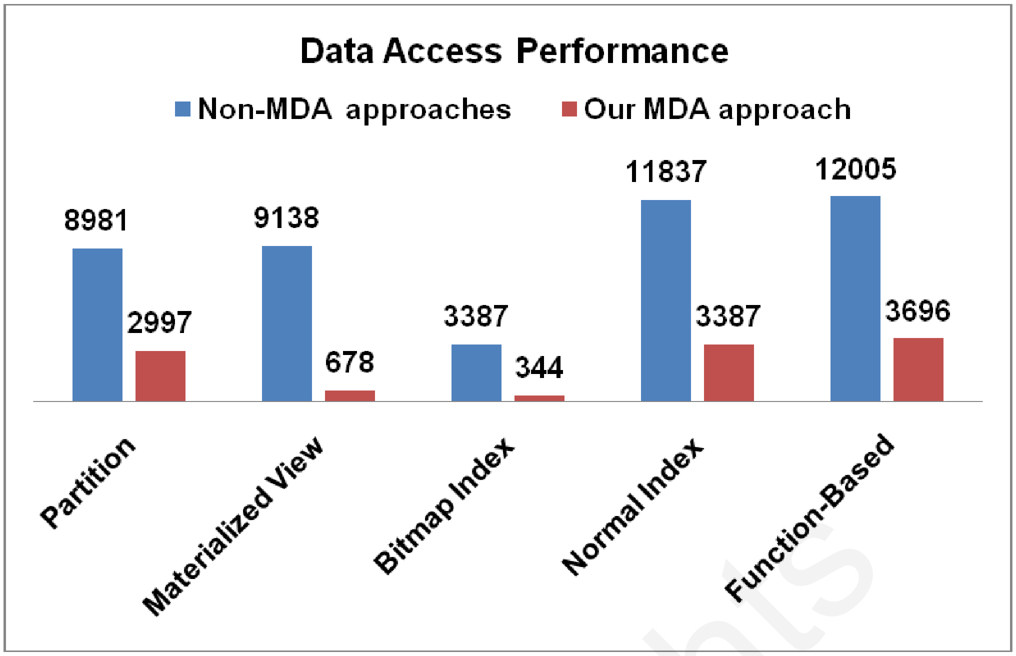


Figure 12. Performance Comparisons using Number of Read Operations.

8. Conclusions

This paper has targeted the development of quality-aware DW framework that is based on MDA standard by addressing the DW performance requirements at early stages of system development. After surveying the related literature, the main focus of MDA and DW framework is not as much on the performance needs as on database issues; and so this work presented an extension of the Unified Modeling Language using UML2 profile. This profile is defined by set of new stereotypes to enable DW team to elegantly represent the DW performance requirements with MD properties at the conceptual level. The proposed approach is MDA compliant and uses Query-View-Transformation and Model-to-Text languages for automatic generation of Platform Specific Model and code in target platform. Finally, this work has been validated and showed the benefit of the proposed profile by developing a case study for sales DW system using Eclipse modeling tool. The outcomes from the validation process gave clear evidences that the proposed development framework for DW will outperform any development method that does not handle performance requirement at early stages of system development.

Regarding future work, promising challenges wait in many areas touched by this paper. Our work focus only on performance part of Non-functional requirements while other types such as user friendliness and multidimensionality have promising challenges. Even though our work handle commonly used performance elements such as partitioning and indexing, it neither handle the parallelism while executing a specific query nor it deals with optimizing disk usage. Moreover, our work focused on developing Relational PSM and its related transformation rules; hence there is a space for handling other PSM such as object and object-relational databases.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

REFERENCES

ALBERTO ABELLÓ , J. SAMOS, F. SALTOR. 2001. A framework for the classification and description of multidimensional data models. In *the proceeding of Conference on Database and Expert Systems Applications (DEXA'01)*B(Munich, Germany). September 3– 7.

ANNEKE KLEPPE, JOS WARMER, WIM BAST. 2003. *MDA Explained the Model-Driven Architecture: Practice and Promise.* Boston: Addison-Wesley.

CHRISTOF SIMONS, AND G. WIRTZ. 2007. Modeling context in mobile distributed systems with the UML. *Journal of Visual Languages & Computing*. 18, 4, 420-439

CHUNG, L., NIXON, B., YU, E., MYLOPOULOS, J. 2000. Non-Functional Requirements in Software Engineering. *Kluwer Academic Publishers.* Boston Hardbound.

EDUARDO FERNANDEZ-MEDINA, J. TRUJILLO, R. VILLARROEL, AND M.PIATTINI. 2006a. Access control and audit model for the multidimensional modeling of Data Warehouses. DSS. 42, 1270-1289.

EDUARDO FERNANDEZ-MEDINA ET AL. 2006b. Developing secure Data Warehouses with a UML extension. *Information Systems*. 32, 6, 826-856

EMILIO SOLER,VILLARROEL R., TRUJILLO J., PIATTINI, FERNÁNDEZ-MEDINA. 2006. Representing security and audit rules for DW at the logical level by using the CWM. In the Proceedings *of The First International Conference on Availability, Reliability and Security(ARES2006)* (Vienna, Austria). April 20-22.

EMILIO SOLER, JUAN TRUJILLO, EDUARDO FERNÁNDEZ-MEDINA AND MARIO PIATTINI. 2007a. A Framework for the Development of Secure Data Warehouses based on MDA and QVT. In the Proceedings of the second International Conference on Availability, Reliability and Security (ARES'07) (Vienna , Austria). April 10-13.

EMILIO SOLER, J. TRUJILLO, E. FERNÁNDEZ-MEDINA, M. PIATTINI. 2007b. Application of QVT for the development of secure Data Warehouses: a case study.. *In the Proceedings of the second International Conference on Availability, Reliability and Security (ARES'07)* (Vienna , Austria). April 10-13.

EMILIO SOLER, VERONIKA STEFANOV, JOSE-NORBERTO MAZ´ONZ, JUAN TRUJILLO, EDUARDO FERN ANDEZ-MEDINA AND MARIO PIATTINI. 2008a. Towards Comprehensive Requirement Analysis for Data Warehouses: Considering Security Requirements. In the Proceedings of the Third International Conference on Availability, Reliability and Security, (ARES 2008) (BARCELONA, SPAIN). March 4-7.

EMILIO SOLER, J. TRUJILLO, E. FERNÁNDEZ-MEDINA, M. PIATTINI. 2008b. Building a secure star schema in Data Warehouses by an extension of the relational package from CWM. *Computer Standards & Interfaces.* 30, 6, 341–350.

GOETZ GRAEFE. 2006. B-tree indexes for high update rates. *ACM SIGMOD Record*. 35, 1, 39 – 44.

GEORGE EADON, EUGENE INSEOK CHONG, SHRIKANTH SHANKAR, ANANTH RAGHAVAN, JAGANNATHAN SRINIVASAN, SOURIPRIYA DAS. 2008. Supporting Table Partitioning By Reference in Oracle. *In the Proceedings of the 2008 ACM SIGMOD international conference on Management of data* (Vancouver, BC, Canada). June 9–12.

GORET TI K.Y. CHAN, QING LI, LING FENG. 1999. Design and Selection of Materialized Views in a Data Warehousing Environment: A Case Study. In *the*

Proceedings of the 2nd ACM international workshop on Data warehousing and OLAP. (Kansas City, MO, USA). November 02 – 06.

GUADALUPE CANAHUATE, TAN APAYDIN, AHMET SACAN, HAKAN FERHATOSMANOGLU. 2009. Secondary Bitmap Indexes with vertical and horizontal partitioning. In *the Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology* (Saint Petersburg, Russia). March 24 – 26.

IBRAHIM JALUTA, SEPPO SIPPU AND ELJAS SOISALON-SOININEN. 2006. B-Tree Concurrency Control and Recovery in Page-Server Database Systems, *ACM Transactions on Database Systems*, 31, 1, 82–132.

INMON,B. 2005. Building the Data Warehouse. Wiley Publishing Inc. Fourth Edition.

JENS LECHTENBÖRGER, GOTTFRIED VOSSEN. 2003. Multidimensional normal forms for Data Warehouse design. *Information Systems.* 28, 5, 415-434.

JESUS PARDILLO, JOSE-NORBERTO MAZÓN, JUAN TRUJILLO. 2008. Bridging the semantic gap in OLAP models: platform-independent queries. In the Proceedings *of the DOLAP'08* (Napa Valley, California, USA). October 30.

JOSE-NORBERTO MAZÓN, JUAN TRUJILLO, MANUEL SERRANO, MARIO PIATTINI. 2005. Applying MDA to the Development of Data Warehouses. In *the Proceedings of the DOLAP'05* (Bremen, Germany). November 04–05.

JOSE-NORBERTO MAZON, J. TRUJILLO. 2008. An MDA approach for the development of Data Warehouses. *Decision Support Systems*. 45, 1, 41–58.

JUAN TRUJILLO, EMILIO SOLER, EDUARDO FERNÁNDEZ-MEDINA, MARIO PIATTINI. 2009a. A UML 2.0 profile to define security requirements for Data Warehouses. *Computer Standards & Interfaces.* 31, 5, 969-983.

JUAN TRUJILLO, EMILIO SOLER, EDUARDO FERNANDEZ-MEDINA, MARIO PIATTINI. 2009b. An engineering process for developing Secure Data Warehouses. *Information and Software Technology.* 51, 1033–1051.

KIMBALL, R. 2002. The Data Warehouse Toolkit. Second Edition. Wiley Publishing.

KUMPON FARPINYO AND TWITTIE SENIVONGSE. 2003. Designing and creating relational schemas with a CWM-based tool. In the Proceedings *of the 1st international symposium on Information and communication technologies (ISICT '03)* (Dublin, Ireland). September 24 – 26.

LEONARDO WEISS F. CHAVES, ERIK BUCHMANN, FABIAN HUESKE, KLEMENS BOHM. 2009. Towards Materialized View Selection for Distributed Databases. In *the Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology* (Saint Petersburg, Russia). March 24–26.

LEOPOLDO ZEPEDA, MATILDE CELMA. 2005. Specifying Metamodel transformations for Data Warehouse design. *In the Proceedings of the ACM Southeast Conference*(Kennesaw, GA, USA). March.

MARCOS LOPEZ-SANZ, CESAR J. ACUNA CARLOS E. CUESTA AND ESPERANZA MARCOS. 2008. Modelling of Service-Oriented Architectures with UML. *Electronic Notes in Theoretical Computer Science.* 194, 4, 23-37.

MARK LEVENE AND GEORGE LOIZOU. 2003. Why is the snowflake schema a good Data Warehouse design? *Information Systems*. 28, 3, 225-240.

MICHAL STABNO, ROBERT WREMBEL. 2009. RLH: Bitmap compression technique based on run-length and Huffman
Encoding. *Information Systems.* 34, 4-5, 400-414.

MING CHUAN HUNG, MAN-LIN HUANG, DON-LIN YANG, NIEN-LIN HSUEH. 2007. Efficient approaches for Materialized Views selection in a Data Warehouse. *Information Sciences.* 177, 1333–1348.

NAVNEET GOYAL, YASHVARDHAN SHARMA. 2009. New binning strategy for bitmap indices on high cardinality attributes. *In the Proceedings of the 2nd Bangalore Annual Compute Conference* (Bangalore, India). January 09-10.

OCTAVIO GLORIO, JUAN TRUJILLO. 2008. An MDA Approach for the Development of Spatial Data Warehouses. In *the Proceedings of the ACM 10th international conference on Data Warehousing and Knowledge Dis* (Turin, Italy). *LNCS*. 5182, 23-32.

OCTAVIO GLORIO, JUAN TRUJILLO. 2009. Designing Data Warehouses for Geographic OLAP Querying by using MDA. *LNCS Computational Science and Its Applications.* 5592, 505-519.

OMG. 2003b. OBJECT MANAGEMENT GROUP. MDA Guide Version 1.0.1.

OMG. 2005b. OBJECT MANAGEMENT GROUP. UML Superstructure Specification, v2.0. http://www.omg.org/cgi-bin/doc?formal/05-07-04.

OMG. 2008. OBJECT MANAGEMENT GROUP. Meta Object Facility (MOF) 2.0 Query/ View/ Transformation Specification Version 1.0. OMG Document Number: formal/2008-04-03.

OMG. 2009. OBJECT MANAGEMENT GROUP. Unified Modeling Language(UML), Infrastructure Version 2.2. http://www.omg.org/spec/UML/2.2/Infrastructure/PDF.

ORACLE. 2009. Oracle Database Data Warehousing Guide 10g. *www.oracle.com*. Accessed on 10 Sep 2009.

PAIM FÁBIO RILSTON SILVA AND JAELSON F. B. CASTRO. 2002. Enhancing Data Warehouse Design with the NFR Framework. In *the Proceedings of the fifth Workshop on Requirements Engineering (WER 2002)*(Valencia, Spain). November 11-12.

PATRICK O'NEIL, DALLAN QUASS. 1997. Improved query performance with variant indexes, In *the Proceedings of the 1997 ACM SIGMOD international conference on Management of data* (Tucson, Arizona, USA). May 11-15.

QUEST. 2009. QUEST SOFTWARE, Toad for Oracle. *www.quest.com.* Accessed on Sep 10 2009.

RICHARD J. NEIMIEC. 1999. Oracle Performance Tuning: TIPS & TECHNIQUES. Osborn/McGraw-Hill Inc.

RODOLFO VILLARROEL, EDUARDO FERNÁNDEZ-MEDINA, MARIO PIATTINI, JUAN TRUJILLO. 2006. A UML 2.0/OCL Extension for Designing Secure Data Warehouses. *Journal of Research and Practice in Information Technology*. 38, 1.

SANJAY AGRAWAL, VIVEK NARASAYYA, BEVERLY YANG. 2004. Integrating Vertical and Horizontal Partitioning into Automated Physical Database Design. In *the Proceedings of the 2004 ACM SIGMOD international conference on Management of data* (Paris, France). June 13–18.

SERGIO MORA, JUAN TRUJILLO, YEOL SONG. 2006. A UML profile for multidimensional modeling in Data Warehouses. *Data & Knowledge Engineering*. 59, 725–769.

VINCENT NG , DIK MAN LAW , NARASIMHAIAH GORLA , CHI KONG CHAN. 2003. Applying genetic algorithms in database partitioning, In *the Proceedings of the 2003 ACM symposium on Applied computing* (Melbourne, Florida). March 09-12.