Model Based Materialized View Evolution: A Review

Anjana Gosain\textsuperscript{a}, Sangeeta Sabharwal\textsuperscript{b}, Rolly Gupta\textsuperscript{c*}

\textsuperscript{a}Professor, USICT, Guru Gobind Singh Indraprastha University, Delhi, India
\textsuperscript{b}Professor, NSIT, Delhi University, Delhi, India
\textsuperscript{c}Research Scholar, NSIT, Delhi University, Delhi, India

Abstract

Materialized views evolve in order to meet the user’s requirement in the dynamically changing data warehouse environment. Therefore, materialized view evolution approach focuses on choosing materialized views in the design process of data warehouses or maintaining a materialized view in response to data changes or to data sources changes and sometimes to monitor the DW quality under schema evolution. Although few researchers have addressed materialized view evolution problem for evolving an appropriate set of views. But, none of the surveys provides a classification of materialized view evolution approaches in order to identify their advantages and disadvantages. This survey tries to fill this gap. The present paper provides a review of model based materialized view evolution methods by identifying the three main dimensions namely; (i) Framework, (ii) Architecture and (iii) Model/Design Model, that are the basis in the classification of materialized view evolution methods. The goal of this paper is to provide a comparative study on model based materialized view evolution methods, by identifying respective potentials and limits.

Keywords: Model; View Maintenance; Materialized view evolution

1. Introduction

A data warehouse (DW) acts as a centralized repository of data received from external data sources. The purpose of DW is integration of data that are further analyzed by On-Line Analytical Processing (OLAP) applications. The results of the analysis are the basis of strategic business decisions. Typically, external data sources...
are autonomous and heterogeneous transactional systems, or files, etc. They have dynamic nature, manifested by changes of their content as well as changes to their schema or structures of dimensions. So, to keep the content of a DW up-to-date, the process of Extraction, Transformation and Loading (ETL) is run periodically. So, DW refreshing with respect to its content is implemented by materialized views.

Evolution in data warehouse may be generated by due to changes in schema, changes in software and the changes in data warehouse requirement. Further, Data warehouse evolution may be classified into three different approaches namely schema evolution, schema versioning and materialized view evolution, in which materialized view evolution approach focuses on choosing materialized views in the design process of data warehouses or maintaining a materialized view in response to data changes or to data sources changes and sometimes to monitor the DW quality under schema evolution. The materialized view evolution issue has been investigated in context of query optimization, data placement in a distributed setting, warehouse design, web databases, etc. [Labrinidis et. al 2009, Dhote et. al. 2009, Halevy 2001,]. None of the surveys provides a classification of materialized view evolution approaches in order to identify their advantages and disadvantages. Our survey tries to fill this gap.

This paper aims at studying the materialized view evolution in relational databases and data warehouses as well as in a distributed setting. First, we define a dimension for highlighting the materialized view evolution problem. Thus, we present a classification of materialized view evolution methods based on the main materialized view evolution dimensions, as identified. This study also reviews existing model or design based materialized view evolution method by identifying respective potentials and limits. The rest of the paper is organized as follows: Section 2 identifies the main materialized view evolution dimensions along which materialized view evolution methods can be classified. Section 3 and 4, presents a comparative study and review of the various research works explored. Section 5 contains the conclusion and discusses open issues.

2. State of Art

Materialized view evolution simplifies the design, implementation, maintenance and management of data warehousing approaches. We classified the materialized view evolution into following dimensions as -- Framework, Architecture and Model/Design Model. On the ground of methods involved in evolution of materialized views in a data warehousing dimensions, they can be categorized further as – View Evolution, Basic View Maintenance, Incremental VM, Self Maintainable Maintenance, Not self Maintainable Maintenance, View selection, View Synchronization, View Adaptation (Fig.1). We present a comparative study of the various researches works explored in context of model based dimensions and methods.

![Figure 1. Classification of materialized view evolution](image-url)
3. Comparative Study

We have analyzed model based materialized view evolution methods on several parameters and presented their comparative results in the table below:

**TABLE1: COMPARISON OF MODEL BASED MVE METHODS**

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Authors</th>
<th>Technologies/Categor y Adapted</th>
<th>Issues Addressed/Change s Handle d</th>
<th>Model support / perspective</th>
<th>Method's Activities/Goals</th>
<th>Addressed attributes</th>
<th>Applicable framework stage</th>
<th>Advantages</th>
<th>Disadvantage s</th>
<th>Type s of Queries/Operation</th>
<th>Tool Support/Impl ementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>A. Mostefaoui, et.al.</td>
<td>VM</td>
<td>Distributed computation problem</td>
<td>Formal Model</td>
<td>Distributed computing and Database cross</td>
<td>Concurrency control</td>
<td>VM</td>
<td>Generalizabl e to address multi-term view.</td>
<td>To be explored</td>
<td>RM</td>
<td>----</td>
</tr>
<tr>
<td>2.</td>
<td>Jalel Akaichi</td>
<td>VM</td>
<td>Schema changes</td>
<td>EDW framework</td>
<td>Automati c replaceme nt process</td>
<td>Availability , Efficiency, accessibility , consistency</td>
<td>VM</td>
<td>Information al resource availability</td>
<td>Network saturation, cost and execution time.</td>
<td>RM</td>
<td>E-SQL</td>
</tr>
<tr>
<td>3.</td>
<td>Cécile Favre , et.al.</td>
<td>VE</td>
<td>Knowledge driven evolution</td>
<td>R-DW Model</td>
<td>Fact table &amp; evolving part (Rules)</td>
<td>Concurrency, Flexibility</td>
<td>D/VM</td>
<td>Dynamicall y create dimension hierarchies, allows analysis context evolution</td>
<td>---</td>
<td>RM</td>
<td>---</td>
</tr>
<tr>
<td>5.</td>
<td>Darja Solodovnikov a , et.al.</td>
<td>D/VM/VE</td>
<td>Manage DW evolution automaticall y or semi-automaticall y</td>
<td>Evolution oriented user centric design</td>
<td>Versionin g of DW, schemata &amp; data semantics</td>
<td>Data Quality</td>
<td>Design/VM</td>
<td>User-centric approach</td>
<td>Personalizati on of report</td>
<td>RM</td>
<td>SQL</td>
</tr>
<tr>
<td>6.</td>
<td>Christoph Quix</td>
<td>VE</td>
<td>Capture dynamics of DW, Evolution of DW views</td>
<td>DW Process model</td>
<td>Evolution operators, Quality factors, schema evolution</td>
<td>Data Quality</td>
<td>Design/VM</td>
<td>Mentor quality of DW, Query and analyze metadata</td>
<td>More detailed study required</td>
<td>RM</td>
<td>Metada ta Reposit ory Concep tthase</td>
</tr>
<tr>
<td>7.</td>
<td>Ericka-Janet Rechy-Rami`rez , et.al.</td>
<td>D/VM/VE</td>
<td>Evolve the implementati on</td>
<td>Conceptual evolution Model</td>
<td>Bitemporal versionin g of multidime nsional schemas</td>
<td>Better performanc e</td>
<td>Design/VM</td>
<td>Computatio nal tool to support DW schema evolution</td>
<td>Tool Required</td>
<td>RM</td>
<td>SQL-like Evoluti on languag e</td>
</tr>
<tr>
<td>8.</td>
<td>Dragan Sahapski, et.al.</td>
<td>D/VM</td>
<td>DYNAMICALLY evolve the design of implementat ion schema</td>
<td>Model supporting Multiversio nal schema</td>
<td>Derivatio n procedure , optimizati on of</td>
<td>Efficient, Better performanc e</td>
<td>Design/VM</td>
<td>Dynamicall y self-improve design, Multiversio nal</td>
<td>Explore abstraction technique,</td>
<td>RM</td>
<td>Experi mental Analysi s</td>
</tr>
<tr>
<td>No.</td>
<td>Authors</td>
<td>Domain</td>
<td>Approach</td>
<td>Design</td>
<td>Implementation</td>
<td>Tools/Techniques</td>
<td>Research Focus</td>
<td>Future Work</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>----------</td>
<td>--------</td>
<td>----------------</td>
<td>----------------</td>
<td>---------------</td>
<td>-------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>Robert M. Bruckner, et.al.</td>
<td>VE</td>
<td>Combine web OLAP &amp; DW resources</td>
<td>OLAP Model</td>
<td>Interoperability</td>
<td>DSS</td>
<td>Decision support and knowledge management on web</td>
<td>More work required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.</td>
<td>P Ding, et.al.</td>
<td>D/VM</td>
<td>Support presentation, control and comparison of version evolution</td>
<td>MetaData Model</td>
<td>Integrity</td>
<td>VM</td>
<td>Preserves consistency of metadata repository</td>
<td>More work required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.</td>
<td>Claudine Bréant, et.al.</td>
<td>VE, VM/D</td>
<td>Analyze granularity of Database lifecycle &amp; its architecture</td>
<td>Archimed DW Multidimensional Model</td>
<td>Clarity, Better performance, scalability</td>
<td>Design/ VM</td>
<td>Access to integrated and coherent view</td>
<td>More work required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.</td>
<td>Bartosz Bebel, et.al.</td>
<td>VE</td>
<td>Handle structural changes &amp; evolution of DW schema</td>
<td>Transaction model</td>
<td>Concurrent transactions</td>
<td>D/M</td>
<td>Model for managing structural changes</td>
<td>Implement- ation not done</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13.</td>
<td>Shahrzad M.K, et.al.</td>
<td>VE</td>
<td>Formalize schema &amp; VE, creation &amp; management of multiple versions</td>
<td>V-algebra</td>
<td>Multiple versioning and easy way of investigating various versions</td>
<td>Design/ M</td>
<td>Need transparent retrieval of analytical results</td>
<td>Exemplar based study</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14.</td>
<td>Edgar Benitez Guerrero, et.al.</td>
<td>VE</td>
<td>Creation &amp; evolution of DW</td>
<td>WHER</td>
<td>Better Performance</td>
<td>Design/ M</td>
<td>MDL language</td>
<td>Explore other schema evolution approaches</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15.</td>
<td>George Papastefanatos, et.al.</td>
<td>VE</td>
<td>Propose metrics to access DW design quality</td>
<td>Graph-based model</td>
<td>Data Quality</td>
<td>Design/ M</td>
<td>Proposed metrics, Assessment of DW design</td>
<td>Further experimentation required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16.</td>
<td>B. Bež Bel, et.al.</td>
<td>VE</td>
<td>Handle DW evolution based on schema &amp; data versioning</td>
<td>Formal MV model</td>
<td>Better performance</td>
<td>Design/ M</td>
<td>Presented a multi-version DW prototype system</td>
<td>No implementation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17.</td>
<td>Elena Ferrari, et.al.</td>
<td>VM</td>
<td>Increase efficiency of access control</td>
<td>Temporal authorization model</td>
<td>Efficient, Better Performance</td>
<td>VM</td>
<td>Model specifies temporal authorization &amp; derivation rules</td>
<td>Performance analysis required</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18.</td>
<td>Serge Abiteboul, et.al.</td>
<td>VM</td>
<td>Handle dynamic interactions between web components</td>
<td>Axiom formal model</td>
<td>Relevant, satisfiability provenance</td>
<td>VM</td>
<td>Axilg widget processor, Distributed monitoring system</td>
<td>Require more work to be done</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20.</td>
<td>Matthias Jarke, et.al.</td>
<td>Design/ VM</td>
<td>Improve Quality of DW design &amp; operation</td>
<td>Concept based design</td>
<td>Accuracy, consistency, complete</td>
<td>D/VM</td>
<td>Optimal design</td>
<td>Require more work</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Analyse granularity of Database lifecycle & its architecture**

**Archimed DW Multidimensional Model**

**Clarity, Better performance, scalability**

**Design/ VM**

**Access to integrated and coherent view**

**Model for managing structural changes**

**Need transparent retrieval of analytical results**

**Exemplar based study**

**MDL language**

**Proposed metrics, Assessment of DW design**

**Further experimentation required**

**Presented a multi-version DW prototype system**

**No implementation**

**Performance analysis required**

**Tool development**

**Exemplar based study**

22. Ajit Singh, et.al. | Design | Seek ERP needs | Analytical DW | Integratio n approach | Freshness, better access | D | Investigated ways to build successful data driven IS | Require Implement ation | RM | Examp le-based

23. Adriana Marotta, et.al. | D | Designing issues | DW Design model | Schema transform ation | Improve design | D | Refined logical design of DW & mapping process | Address DW evolution by schema | OO Model | Exampl e-based study

24. Howard Ong | D | Develop real application | Corporate based DW design | Evolution ary trends | --- | D | Examines analytical information | To be explored | --- | ---

25. Thilini Ariyachandra, Hugh Watson | D | DW selection issues | DWA | Survey & multinomi al logistic regression | Efficiency | D | Designed model for prediction | To be explored | --- | Hypoth esis based

26. Zekri, M | D/M | Considering request subject to operational production bases | Multidimensio nal DW design | CDM/gra ph based | Efficient | D/M | Design steps used graphs to represent data model decision making | More work to be done | --- | ---

27. Fred R. McFadden | D | Determine the impact of factors | Research based model | Survey based | Data quality & efficient interface | D | Approach applicable for EIS & DSS | More exploratio n | --- | ---


4. Review and Results

Following sub-sections provide a review of materialized view evolution in a data warehousing model based dimension using various aspects:

4.1. Technique

With respect to technique, summarizing materialized view evolution i.e. which methods involved in evolution of materialized views in a data warehousing model dimension, is most focused viz. View Evolution, Basic View Maintenance, Incremental VM, Self Maintainable Maintenance, Not self Maintainable Maintenance, View selection, View Synchronization, View Adaptation. The study found 12 authors discussing view evolution [3,4], 14 authors discussing basic VM [2,5], 15 authors discussing design [26], but none on incremental VM, view synchronization, self Maintainable Maintenance, Not self Maintainable Maintenance, View selection, and View Adaptation, giving a total of 27 papers under the stated methods in a data warehousing model dimension (Fig.2).

4.2 Tool supported

In order to be effective and useful, methods involved in evolution of materialized views in a data warehousing model dimension must be implemented or analyzed effectively. Theoretical analysis or experimental analysis of methods involved in evolution of materialized views can also be done. Authors in [7,9] provided the
implementation of materialized view evolution methods using front-end or back-end languages. Theoretical analysis has been carried out for the materialized view evolution methods in [13] by the respective authors. While, experimental analysis has been carried out in [21] by the respective authors. Although the methods in [3] have not been analyzed nor implemented at all by the authors. As can be inferred, theoretical analysis have been used more than any other analysis technique for methods in a data warehousing model dimension (Fig.3).

4.3. Attributes Addressed

The materialized view evolution methods considered in this study focus on various external quality attributes like accessibility [2], scalability[11],consistency[2], effectiveness[2,8], etc . But most of papers lacked implementation results for validating their claims. There are also some model based materialized view evolution methods which have not been associated with any external quality attribute..

4.4. Types of queries

Based on the types of queries, the relational model [1] have been most frequently used by the authors for addressing materialized view evolution methods, while others used Object oriented [3] model. There are some materialized view evolution methods which have not been associated with any particular model by the authors. (Fig.4).

4.5. Applicable framework

The materialized view evolution methods considered in this study also focus on applicable framework stage. The study found 14 authors discussing view maintenance as applicable framework stage, while 15 authors discussing design as applicable framework stage.
4.6. Issues addressed

In general, there are two type of algorithms for imposing constraints i.e. - view maintenance algorithms and view update algorithms. Some authors have addressed issues using immediate and deferred data synchronization algorithms [5,10,21], but many authors have mentioned other techniques [3] for imposing constraints in order to handle the required changes in the distributed database environment.

4.7. Method Supported

Immediate and deferred data synchronization algorithms can be further classified on the basis of method’s activities as optimization [10], Data integration [3] or others [1].

4.8. Model supported

Different authors have proposed different types of models for materialized view evolution. Based on the various perspectives on the role of model, they are differently named as EDW[2], Formal MV model [16], Alog formal model [18] and so on. These models are all useful in different situations, however they quantify for different perspective or model support.

4.9. Advantages & Disadvantages

Mostly all the authors are handling maintenance anomalies, provided a model for managing structural changes and have addressed the designing of algorithm for the materialized view evolution problem in order to reduce recomputation. They have provided a practical implication based model for materialized view evolution [14]. But some of the authors have not designed the algorithm nor have provided the validation / implementation studies [15] in regard to model based materialized view evolution.

5. Conclusion & Future Scope

This study provides a critical survey of different approaches in which the model based materialized view evolution has been studied in relational databases and data warehouses as well as in a distributed setting. We have defined formally the materialized view evolution problem and identified the main materialized view evolution dimensions along with materialized view evolution methods have been classified. Based on the classification, we have discussed model based materialized view evolution methods.

Analysis of materialized view evolution has shown that there is very few work on materialized view evolution in databases [Stanchev et.al. 2001]. An in-depth analysis of the literature revealed that the research not always devoted a comprehensive attention to all aspects of materialized view evolution. Some interesting solutions have been proposed, no broad and shared model has been devised yet. As future work, we propose to investigate the materialized view evolution problem more efficiently in a distributed setting and semantic web databases.

References

Intelligence and Applications, Vol. 143, 271-278.
11. Claudine Bréant, Gérald Thurler, François Borst, Antoine Geissbuhler, “Design of a Multi Dimensional Database for the Archimed DataWarehouse” Connecting Medical Informatics and Bio-Informatics R. Engelbrecht et al. (Eds.) ENMI, 2005