Data Warehouse Schema Evolution: State of the Art

Danijela Subotić, Patrizia Poščić
Department of Informatics
University of Rijeka
Radmile Matejčić 2, 51000 Rijeka, Croatia
{dsubotic, patrizia}@inf.uniri.hr

Vladan Jovanović
Department of Computer Science
Georgia Southern University
1332 Southern Dr, Statesboro, GA 30458, USA
vladan@georgiasouthern.edu

Abstract. *The paper presents an overview of research related to the problem of the data warehouse (DW) evolution. Related research can usually be grouped into three approaches - schema evolution, schema versioning and view maintenance. The main contributions of the paper are: a) DW evolution state of the art, and b) critical analysis of existing methods and approaches to the DW schema evolution. Also, general idea and direction for our future work is briefly described.*

Keywords. data warehouse evolution, schema evolution, schema versioning, view maintenance

1 Introduction

A data warehouse (DW) represents a database used for reporting and data analysis. It integrates current and historical data from a number of heterogeneous data sources, creates a well organized central repository of data and provides for a quick and efficient business analysis. Nowadays the DW is greatly affected by the changes that occur constantly in data sources (in their content and structure). The DW must always contain the latest information to be able to reflect the evolving state of the real world which makes it necessary to properly manage all types of changes and appropriately update the DW. This is the core of the DW evolution problem. However we observe the DW evolution problem separately from the database (DB) evolution problem. Compared to the DB requirements, the DW requirements are increased in respect to the scope and preservation of the history of changes. The problem being explored in the DW evolution is preserving the history of scope changes and data (and metadata) structure changes - in a longer time period. The DW scope today is ever-expanding, and includes multiple sources and multiple data types. Also, in order to provide adequate support to the decision-making processes, the DW must include historical data and trend projections, and not just the last updated state (as is the case with the DB). Because of accelerated business and technology changes, it becomes increasingly important to find an appropriate solution to this problem. With respect to the literature [1], the DW evolution can generally be traced through three approaches - schema evolution ([9]-[21]), schema versioning ([22]-[31]) and view maintenance ([32]-[44]), although some authors [2] propose another classification, based on two research topics – handling data/schema changes in the DW and handling data/schema changes in the data mart (DM). However, almost in every case these two categories can be placed in one of the three above mentioned approaches. We will use a three-approach classification because we believe it is more suitable for a general overview and because it focuses on different ways of dealing with schema changes (both in the DW and the DM) which is more relevant for our future work.

The aim of this paper is to present a state of the art related to the problem of the DW evolution.

The paper is organized as follows – in section II a state of the art is presented, in section III related research is analyzed, and section IV presents the conclusion and directions for our future work.

2 State of the Art

The schema evolution problem was first present in the DB research [3]. Some early research like [4], in which authors presented a version model for the DB schema evolution, or [5] in which authors define historical relational data model and the corresponding set of temporal operators for instant-based events and interval-based states, are interesting. However, the solution presented in [5] does not support full historicizing of the model, because only tuples (rather than individual attributes) are historicized. This causes the increase in the required memory space and reduces performance. In [6] the authors presented PRISM, a system that supports schema evolution for web information systems such as Wikipedia and public scientific databases. In such systems the frequency of schema changes is increased due to the large number of contributors and the tolerance for the system downtime is almost nonexistent. This is similar to the DW and a large number of heterogeneous data sources. However, although the evolution process in this approach is fairly automated a number of necessary changes for the transformation from the old to the new version is still high. Maybe the biggest contributions of this approach are related to the automation of query rewriting and gathering metadata that describes the history of the schema evolution. In [7] and [8] the authors study the problem of temporal DB and the use of temporal data in relational DB, respectively.

As we already mentioned, the DW requirements are increased in respect to the scope and preservation of the history of changes, compared to the DB requirements. This means that the DW schema evolution research also has a broader scope. A state of the art of DW schema evolution research, grouped into three main approaches, is presented next.

2.1 Schema evolution

Schema evolution approach assumes that the schema can only have one version at a given time - the current version. Data is stored in the current version and all changes are made over the current version which is then transformed into a new version. In this approach the DW is defined as a multidimensional schema with the fact and dimension tables and data cubes. Schema evolution has several different levels of updates, like updating dimension, instance, level, fact, attribute, data cube, hierarchy, measure, constraint, quality or structure.

In [9] the authors study the schema changes in multidimensional DB, based solely on the dimension updates. The authors suggested five operators for updating the dimensions and two operators to update instances. In [10] the authors additionally propose 4 composite operators for instance changes.

In [11] the authors define a formal framework for describing the evolution of multidimensional schema. The framework contains a description of the schema and its instances. Having defined the model, the authors propose a set of formal operations for the DW evolution. However, the authors focused only on the DW schema changes that occur due to changes in business requirements.

In [12] the authors propose an extension of the work from [9] and [10] - they propose a visualization tool for dimensions and data cubes.

In [13] the authors define 6 operators for the schema evolution - add/delete/rename attribute and add/delete/rename table.

In [14] the authors explore the process of generating mappings between data sources and logical structure of the DW (i.e. dimensional model). The mechanism is proposed, for obtaining the DW logical schema through pre-defined transformations applied to the logical schema of the data sources. This allows for tracing and documenting the design and mappings between the DW and data sources logical structures. The underlying model for the proposed transformations is the relational model, which can complicate the schema evolution process in terms of keeping and maintaining referential integrity.

In [15] the authors focus on the metadata, which is used for loading and incremental maintenance of the DW and for tracing the origin of the data contained in DW or data marts (DM). AutoMed is presented, a heterogeneous system for transformation and integration of data which has the ability to integrate data across multiple data models. However, the impact of existing and revised constraints on the schema evolution and system performance is not clearly defined.

In [16] the authors define WHES (Warehouse Evolution System) prototype that describes the DW evolution and updates the dimensions and data cubes. Also, authors expanded the SQL language in MDL language (Multidimensional Data Definition Language).

In [17] the authors propose eight operators for schema evolution, which are mainly focused on updating the dimensions, levels, dimension attributes and measures.

In [18] the authors have proposed a formalism for representing the DW schema and determining the correctness of the evolution operators applied to the schema. Also, operators for the extended hierarchy updates have been proposed.

In [19] the authors presented an overview of current research of the DW schema evolution problem and proposed five operators to update the aggregate fact table.

In [20] the authors study the complex expanded hierarchies and suggest evolution operators and constraints for ensuring data integrity and schema validity.

In [21] the authors study the impact of the DW schema evolution to a set of its dependent data marts. The proposed architecture includes horizontal and vertical evolution. Vertical evolution, among other things, contains a Generic Propagation Model which describes how a change in the DW schema transforms into a change in the data mart.

A comparison of research related to schema evolution approach is presented in Table 1. We can see that authors focus on schema changes in the DM (mainly dimension, attribute and fact changes). It should be noted that none of these approaches did take into account the preservation of the history of changes in the schema or the data. The main problem of schema evolution approach is the loss of history - the previous versions are not preserved, and because of that the previously available structures and data can become unavailable in the new version (reconstruction of structures and data is not possible). For this reason, schema evolution approach can be viewed as weaker case of schema versioning approach.

Table 1. Comparison of research related to schema evolution and versioning approach

	Attribute changes	Constraint changes	Cube changes	Dimension changes	Fact changes	Hierarchy changes	Instance changes	Level changes	Measure changes	Aggregate fact changes	New schema version	Real and alternative version	ETL changes
Hurtado [9]				•			•	•					
Hurtado [10]							•						
Blaschka [11]	•			•	•			•					
Vaisman [12]			•	•			•	•					
Chen [13]	•			•	•								
Marotta [14]	•			•	•	•			•				
Fan [15]	•			•	•	•	•						
Benitez [16]	•		•	•	•			•	•				
Kaas [17]	•			•				•	•				
Banerjee [18]			•	•	•	•		•					
Meenakshi [19]										•			
Talwar [20]		•				•							
Feki [21]	•	•		•	•	•		•	•				
Eder [22]	•			•		•	•	•			•		
Body [23]	•			•		•	•	•			•		
Morzy [24]	•	•		•	•	•		•			•		
Golfarelli [25]	•			•		•		•			•		
Bebel [26]											•	•	
Bebel [27]	•			•	•		•	•			•		
Papastefanatos [28]	•	•											•
Golfarelli [29]											•		
Solodovnikova [30]											•		•
Zouari Turki [31]		•		•	•		•				•		

2.2 Schema versioning

Schema versioning approach consists of transferring data from existing schema to the new schema. All changes are made to the new schema. All the old schema versions are kept through the use of time extension to the schema version or the physical storage of the version.

In [22] the authors propose a temporal multidimensional model, which allows the storage of temporal versions of dimensional data. The problem with this approach is that it does not support the storage of historical versions of the scheme, but only of the data.

In [23] the authors propose operators for dimension and instance update. For any change a new, time-limited version is defined. The proposed mechanism supports temporal versions of the data, but all versions of the data are stored in a single fact table. This means that only changes in the dimensions and their instances are supported.

In [24] the authors propose a 15 operators to modify the DW schema, which will be applied on the new DW schema version.

In [25] the authors have studied the evolution of the hierarchies within dimensions. Authors propose saving augmented schema with the new schema version. However, this approach supports only four basic operators for the schema modification (adding and deleting attributes, adding and deleting functional dependencies). In [29] the work from [25] is extended and X-Time, a research prototype for managing schema versioning in relational DW, is presented.

In [26] the authors present a prototype of a multi-version DW that supports changes in the schema structure and "what-if" analysis. Two types of versions are defined: the real version and the alternative version. However, alternative versions are associated with additional instances, which limits the flexibility and scope of the user's what-if analysis.

In [27] the authors have extended the work from [26] by proposing two groups of operators: operators for schema changes (fifteen operators, mainly focused on updating dimensions, instances, attributes, levels and facts) and operators for structural changes in the dimension instances (five operators, mainly focused on updating the instance levels).

In [28] the authors have studied the problem of performing "what-if" analysis for the changes that occur over the data source schema. A graph model which uniformly models relations, queries, views, ETL activities and their states, is presented. This graph model enables predicting the effect of changes to the overall system.

In [30] a framework for the DW evolution is presented. The proposed framework automates the DW schema evolution and creation of the new versions. Also, it allows adjustment of ETL processes and existing reports on the DW schema. The emphasis is on the metadata repository, which manages versions and supports schema changes. However, a metadata repository is not historicized and because of that it is not able to reflect the history of metadata changes.

In [31] the authors propose a MV-DW model for managing a multi-version DW. Model is based on the DW schema and instance versioning.

A comparison of research related to schema versioning approach is presented in Table 1. Here authors also focus on schema changes in the DM (mainly dimension, attribute and hierarchy changes), but all changes are implemented in a new schema version (history is preserved).

Table 2. Comparison of research related to view maintenance approach

	Basic maintenance	Incremental maintenance	View selection	Select-Project-Join view	Metadata	Implementation and language	Quality
Gupta [32]	●	●		●		●	
Baralis [33]			●				
Bouzeghoub [34]	●						
Bellahsene [35]		●					
Rajaraman [36]						●	
Lakshmanan [37]					●	●	
Cui [38]		●		●			
Rundensteiner [39]						●	
Zhang [40]		●					
Rundensteiner [41]		●					
Quix [42]	●						●
Akaichi [43]						●	
Almazyad [44]		●			●		

2.3 View maintenance

A data warehouse can be defined as a set of materialized views over data sources ([35], [39], [41]). The main problems studied here are how to effectively maintain views (how to adjust or synchronize the view after changes in the definition or scope of the view) [32], [41] and how to efficiently select views for greater effect (which aggregation on which level increases the performance with respect to storage constraints) [33]. View maintenance can be categorized into two approaches: view adaptation and view synchronization. In view adaptation approach in order to adapt to changes metadata that contains the latest structural information is added to materialized view. View synchronization consists of rewriting the view after the changes in data sources.

In [32] the authors describe materialized views, the types of application, problems and techniques for their maintenance. Also they describe several algorithms for incremental view maintenance.

In [34] the authors present a formal model for the DW schema evolution, which has only two operators - Add and Delete view. This approach (as well as other approaches that are based on the view synchronization) consumes a lot of resources and affects system performance because every time a change occurs the view has to be rewritten.

In [35] the author proposes a user-defined view versions, in order to simulate DW schema changes. This simplifies the process - a new view is created from the old view, and not from data sources. However, this approach does not support multidimensional schema evolution (facts and dimensions).

In [36] the authors define a C-SQL (Cooperative SQL), an extension of the SQL language for view writing.

In [37] the authors define the S-SQL (Schema SQL) for the integration of relational databases and metadata.

In [38] the authors have studied the problem of view lineage tracing. They formalized the problem and developed an algorithm for data lineage tracing for relational ASPJ (aggregate-select-project-join) views.

In [39] the authors present an EVE prototype, a system to automate the view definition rewriting.

In [40] the authors have presented a formal framework SDCC (Schema change and Data update Concurrency Control), as an approach to solving the problem of simultaneous schema and data update.

In [41] authors have studied how to avoid duplication of data during the process of refreshing the DW after the change. However, authors did not take into account the changes in the dimensions and the clear solution for applying the changes to the fact and dimension table is not presented.

In [42] the quality of data during the schema evolution is studied. The emphasis is on the metadata repository that contains all relevant DW metadata. This allows a user to analyze errors and inconsistencies in the architecture, quality, processes and evolution of the DW.

In [43], based on the EVE system, the authors define MAVIE - a system for view synchronization in dynamic and distributed environment. MAVIE system is based on the mobile agents technology so it reduces the synchronization time and avoids network saturation.

In [44] the authors propose an algorithm for incremental view maintenance, in order to solve the problem of anomalies and inconsistent view changes. The concept of storing versions for older versions of the tables updated on the data source was used.

A comparison of research related to view maintenance approach is presented in Table 2. Here authors used view maintenance approach to solve the problem of propagation of changes form data sources to the DW. We can see that the authors recognized two categories of view maintenance – basic and incremental, and they mostly focused their research on incremental view maintenance, which tries to find ways (mostly through creation and adaptation of proposed algorithms) to avoid rewriting the view from a beginning. However, all those approaches are quite complex and require a lot of time and resources.

3 State of the Art Analysis

Research papers presented above are studying the schema evolution in the context of relational and dimensional model and materialized views. As we can see from the Table 1, the problem of propagation of changes in data marts (DM) is quite well researched and some interesting approaches are proposed. We can say that a general consensus exists in respect to this problem. However, the problem of propagation of changes form data sources to the DW still leaves room for further research. Authors mainly tried to solve this problem with a view maintenance approach (see Table 2.), but the view maintenance process can cause the network saturation depending on the amount of updated views and the amount of information they contain. It is still a common practice manual writing of the view after the changes in the sources. To the best of our knowledge, the problems of anomalies and inconsistent changes in the views are still unresolved and the proposed approaches for view maintenance are still limited in terms of efficiency and performance. Also, view maintenance approach completely ignores the ETL processes which are an integral part of most of today's DW.

On the other hand, the process of schema evolution and versioning is still demanding in terms of invested time and resources. We believe it is necessary to balance the resource requirements and the quality of the DW evolution performance. In the context of schema versioning, the following problems are still present: a) slow and expensive transformation and migration of data between different schema versions, b) loss of information during the transformation and migration processes (the problem of preservation of schema consistency and data integrity still exists), c) inefficient adaptation of existing queries and user applications to the new schema version, d) inefficient and error prone work with multiple schema versions at the same time, and e) lack of effective integration, organization and management of metadata. In the context of schema evolution and versioning the academic community has definitely made steps towards solving these problems. However we believe there is still room for improvement (especially with regard to software support, temporal query language [45] and query performance), as well as for defining general solution and fully effective commercial solution (there is still a lack of an integrated system-of-records).

4 Conclusion and future work

The paper presents an overview and analysis of the DW schema evolution research. Related research was presented according to three approaches - schema evolution, schema versioning and view maintenance. However, we noticed that the related research can also be presented as the research on managing the data/schema changes in the data warehouse and the research on managing the data/schema changes in the data mart (changes in facts and dimensions, due to growing business requirements). State of the art shows that related research is generally focused on one aspect of the DW evolution, mainly on the managing the data changes in the data mart. We can say here that there is a general consensus on possible solutions for managing the changes in dimensional data (data marts). As for the other aspects of the DW evolution, some interesting approaches were proposed, but there is still no widely accepted solution and a general framework for managing the DW schema changes. Also, we noticed that the current research does not emphasize the fact that the requirements of the DW, in this day and age, are increased in terms of the scope and structure of data and metadata, and that the researchers are still studying the DW based on the relational model. We can conclude that, although the relational model is largely responsible for physical data independence, it is not convenient to simply and effectively support the (historical) evolution of the logical structure of the DW schema.

We believe it is necessary to find a new, simple and complete solution for preserving the schema consistency and data integrity and for solving all of the still existing problems mentioned in the paper. We will try to find a new perspective and solution to the DW schema evolution problem, by addressing the problem at the general level dealing with all these problems together. The general idea for our approach is that the problem of schema evolution should be observed foremost as a dual problem (or a double issue), and as such our solution should be a dual solution. We will observe the problem from the DW perspective [49],[50],[51] and from the master data management (MDM) perspective [46]. MDM represents the master and reference data and related metadata, set of policies, governance, standards, processes and tools that defines and manages the master and reference data (of an organization) to provide a single point of reference. From the DW perspective every fact that is associated with the dimensions is observed. From the MDM perspective, every master entity (dimension) that is associated with events (facts) is observed. In the case of MDM, as in a DW case, there is the problem of the schema evolution after changes in the data sources or user requirements.

Our general research idea includes a Data Vault meta-model based modeling approach [47],[48] that will integrate the DW with the MDM metadata in a common model [53],[54],[55],[56]. Data Vault (DV) is a relatively new data modeling method that supports design of data warehouses for long-term storage of historical data collected from various data sources. We expect that our dual solution will resolve the issues of tracking the origin of data and the history of changes, avoiding loss of information, enabling fast parallel loading, reconstruction of data and trend projections. The preservation of DW integrity would be facilitated due to the long-term storage of historical data and tracking the origin of data. More importantly, the DW would then contain both a "single version of the fact" [47] and a "single version of the truth" [49]. Also, it could be used as a complete and integrated system of records [52].

We just started our research so among the directions for the future research are development and formalization of a Data Vault based metadata model, development of change cases and an incremental development of an Meta-Data Vault prototype. More details can be found in [57], [58].

5 Acknowledgements

This paper is based upon work supported by the University of Rijeka under project titled "Metode i modeli za dizajn i evoluciju skladišta podataka".

References

[1] Oueslati, W., Akaichi, J. A survey on Data warehouse evolution. *International Journal of Database Management Systems (IJDMS)*, vol.2, no.4, 2010.

[2] Golfarelli, M., Rizzi, S. A survey on temporal data warehousing. *International Journal of Data Warehousing and Mining (IJDWM)*, 5(1), 1-17, 2009.

[3] Roddick, J. F. A Survey of Schema Versioning Issues for Database Systems. *Information and Software Technology*, 37(7):383–393, 1996.

[4] Andany, J., Leonard, M., Palisser, C. Management Of Schema Evolution In Databases. *In Proceedings of the 17th International Conference on Very Large Databases*, Barcelona, 1991.

[5] Das, A.K.., Tu, S.W., Musen, M.A. A Historical Relational Data Model for Managing Temporal Data. *Report, Stanford University, Section on Medical Informatics*, San Diego, 1992.

[6] Curino, C.A., Moon, H.J., Zaniolo, C. Graceful Database Schema Evolution: the PRISM Workbench. *In Proceedings of the Very Large Database Endowment*, Auckland, New Zeland, 2008.

[7] Date, C.J., Darwen, H., Lorentzos, N. *Temporal Data & the Relational Model.* Morgan Kaufmann Publishers, Burlington, USA, 2002.

[8] Johnston, T., Weis, R. *Managing Time in Relational Databases: How to Design, Update and Query Temporal Data.* Morgan Kaufmann Publishers, Burlington, USA, 2010.

[9] Hurtado, C. A., Mendelzon, A. O., Vaisman, A. A. Maintaining Data Cubes under Dimension Updates. *In Proceedings of the 15th International Conference on Data Engineering (ICDE)*, Sydney, Australia, pages 346–355. IEEE Computer Society, 1999.

[10] Hurtado, C. A., Mendelzon, A. O., Vaisman, A. A. Updating OLAP Dimensions. *In Proceedings of the 2nd International Workshop on Data Warehousing and OLAP*, Kansas City, Missouri, USA, 1999.

[11] Blaschka, M., Sapia, C., Hofling, G. On Schema Evolution in Multi-dimensional Databases. *In 1st International Conference on Data Warehousing and Knowledge Discovery (DaWaK 99)*, Florence, Italy, vol. 1676 of LNCS, pages 153–164. Springer, 1999.

[12] Vaisman, A.A., Mendelzon, A.O., Ruaro, W., Cymerman, S.G. Supporting Dimension Updates in an OLAP Server. *In Proceedings of the CAISE02 Conference*,Canada, 2002.

[13] Chen, J., Chen, S., Rundensteiner, E. A transactional model for data warehouse maintenance. *In: Spaccapietra, S., March, S.T., Kambayashi, Y. (eds.) ER 2002*. LNCS, vol. 2503, pp. 247–262. Springer, Heidelberg, 2002.

[14] Marotta, A., Ruggia, R. Data Warehouse Design: A schema-transformation approach. *22nd International Conference of the Chilean Computer Science Society (SCCC)*, Copiapo, Chile, 2002.

[15] Fan, H., Poulovassilis, A. Schema Evolution in Data Warehousing Environments – a schema transformation-based approach. *In Proceedings of Conceptual Modeling - ER, 23rd International Conference on Conceptual Modeling*, Shanghai, China, 2004.

[16] Benitez-Guerrero, E., Collet, C., Adiba, M. The WHES Approach to Data Warehouse Evolution . *e-Gn osis(online)*, vol.2 , 2004.

[17] Kaas, C.E., Pedersen, T.B., Rasmussen, B.D. Schema Evolution for Stars and Snowflakes. *In Proceedings of the International Conference on Enterprise Information Systems (ICEIS 2004)*, Portugal, 2004.

[18] Banerjee, S., Davis, K.C. Modeling Data Warehouse Schema Evolution over Extended Hierarchy Semantics, S.Spaccapietra et.al (EDs): *Journal on Data Semantics XIII*, LNCS 5530, pp.72-96,Springer- Berlin, Heidelberg, 2009.

[19] Meenakshi, A., Gosain, A. Schema Evolution for Data Warehouse: A Survey. *International Journal of Computer Applications*, vol.22, no.6, 2011.

[20] Talwar, K., Gosain, A. Implementing Schema Evolution in Data Warehouse through Complex Hierarchy Semantics. *International Journal of Scientific & Engineering Research*, vol. 3, iss. 7, 2012.

[21] Feki, J., Taktak, S. Impacts of Data Warehouse Evolution on Data Marts. *In Proceedings of the 13th International Arab Conference on Information Technology (ACIT)*, University of Balamand, Lebanon, 2012.

[22] Eder, J., Koncilla, C. Evolution of Dimension Data in Temporal Data Warehouses. *Technical Report*, 2000.

[23] Body, M. , Miquel, M., Bedard, Y., Tchounikine, A. A Multidimensional and Multiversion Structure for OLAP Applications. *In 5th ACM International Workshop on Data Warehousing and OLAP (DOLAP 02)*, McLean, Virginia, USA, pages 1–6. ACM Press, 2002.

[24] Morzy, T., Wrembel, R. On Querying Versions of Multiversion Data Warehouse. *In Proceedings of the International Workshop on Data Warehousing and OLAP, DOLAP"04*, Washington, USA, 2004.

[25] Golfarelli, M., Lechtenbörger, J., Rizzi, S., Vossen, G.: Schema Versioning in Data Warehouses. *In: Wang, S., Tanaka, K., Zhou, S., Ling, T.-W., Guan, J., Yang, D.Grandi, F., Mangina, E.E., Song, I.-Y., Mayr, H.C. (eds.) ER Workshops 2004*. LNCS, vol. 3289, pages 415–428. Springer, Heidelberg 2004.

[26] Bebel, B., Eder, J., Koncilia, C. , Morzy, T., Wrembel, R. Creation and Management of Versions in Multiversion Data Warehouse. *In 19th ACM Symposium on Applied Computing (SAC 04)*, Nicosia, Cyprus, pages 717–723. ACM Press, 2004.

[27] Bebel, B., Krolinkowski, Z., Wrembel, R. Formal approach to modeling a multiversion data warehouse. *Bulletin of the Polish academy of sciences*, Technical Sciences, vol. 54, no. 1, 2006.

[28] Papastefanatos, G., Vassiliadis, P., Simitsis, A., Vassiliou, Y. What-if Analysis for Data Warehouse Evolution. *In Proceedings of the 9th International Conference on Data Warehousing and Knowledge Discovery*, Regensburg, Germany, 2007.

[29] Golfarelli, M., Rizzi, S. X-Time: Schema Versioning and Cross-Version Querying in Data Warehouses. *In Proceedings of the 23rd International Conference on Data Engineering (ICDE 2007)*, Istanbul, Turkey, pages 1471-1472, 2007.

[30] Solodovnikova, D. Data Warehouse Evolution Framework. *In Proceedings of the Spring Young Researcher's Colloquium On Database and Information Systems SYRCoDIS*, Moscow, Russia, 2007.

[31] Zouari Turki, I., Ghozzi Jedidi, F., Bouaziz, R. Constraints to manage consistency in multiversion data warehouse. *In Proceedings of the 35th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia, 2012.

[32] Gupta, A., Mumick, I. Maintenance of Materialized Views: Problems, Techniques, and Applications. *Data Engineering Bulletin*, 1995.

[33] Baralis, E., Paraboschi, S., Teniente, E. Materialized view selection in a multidimensional database. *In Proceedings of the Conference on Very Large Databases*, Athens, Greece, 1997.

[34] Bouzeghoub, M., Kedad, Z. A Logical Model for Data Warehouse Design and Evolution. *In Proceedings of the 2nd International Conference on Data Warehousing and Knowledge Discovery (DaWaK)*, London, UK, pages 178-188, 2000.

[35] Bellahsene, Z. Schema Evolution in Data Warehouses. *Knowledge and Information Systems*, 4(3):283–304, 2002.

[36] Rajaraman, Y. Sagiv, J. D. Ullman. Answering Queries Using Templates With Binding Patterns. *In Proceedings of ACM Symposium on Principles in Database Systems*, pages 105-112, 1995.

[37] Lakshmanan, L. V. S., Sadri, F. , Subramanian, I. N. Schema SQL: a Language for

Interoperability in Relational Multi-Databases Systems. *In Proceedings of the 22nd International Conference on Very Large Databases*, pages 239-250, 1996.

[38] Cui, Y., Widom, J. Practical Lineage Tracing in Data Warehouses. *In Proceedings of the 16th International Conference on Data Engineering (ICDE'00)*, San Diego, California, 2000.

[39] Rundensteiner, E. A. , Koeller, A., Zhang, X., Lee, A.J., Nica, A. Evolvable View Environment EVE: A Data Warehouse System Handling Schema and Data Changes of Distributed Sources. *In Proceedings of the International Database Engineering and Application Symposium (IDEAS'99)*, Montreal, Canada, 1999.

[40] Zhang, X., Rundensteiner, E.A. Data Warehouse Maintenance Under Concurrent Schema and Data Updates. *In Proceedings of the International Conference on Data Engineering (ICDE'99)*, Sydney, 1999.

[41] Rundensteiner, E. A., Koeller, A., Zhang, X. Maintaining Data Warehouses Over Changing Information Sources. *In Communications of the ACM*, vol.43, pages 57-62, New York, USA, 2000.

[42] Quix. Repository Support for Data Warehouse Evolution. *In Proceedings of the International Workshop DMDW*, Heidelberg, Germany 2004.

[43] Akaichi, J., Oueslati, W. MAVIE: A Mobile Agents View synchronization system. *In: First international conference on the applications of digital information and web technology.* Ostravem, pages 145-150, 2008.

[44] Almazyad, A.S., Siddiqui, M.K. Incremental View Maintenance: An Algorithmic Approach. *International Journal of Electrical and Computer Sciences (IJECS-IJENS)*, vol.10, no. 3, 2010.

[45] Snodgrass, R. T., *The TSQL2 Temporal Query Language*. Kluwer Academic Publishers, 1995.

[46] Berson, A., Dubbov, L., *Master data management and data governance, 2nd edition*, McGraw Hill, 2011.

[47] Linstedt, D. *SuperCharge Your Data Watehouse: Invaluable Data Modeling Rules to Implement Your Data Vault.* CreateSpace Independent Publishing Platform, USA, 2011.

[48] Jovanović, V., Bojičić, I. Conceptual Data Vault Model. *In Proceedings of the Southern Association for Information Systems Conference*, Atlanta, USA, 2012.

[49] Inmon, W.H., Strauss, D., Neushloss, G. *DW 2.0: The Architecture for the Next Generation of Data Warehousing.* Morgan Kaufmann Publishers, Burlington, USA, 2008.

[50] Kimball, R., Ross, M., *The data warehouse toolkit, 3rd edition*, John Wiley, 2013.

[51] Inmon, W., Zachman, J., Geiger, J., *Data stores, data warehousing, and the Zachman's framweork*, McGraw Hill, 1997.

[52] Jovanović, V., Bojičić, I., Knowles, C., Pavlic, M. Persistent Staging Area Models For Data Warehouses. *Issues in Information Systems,* vol.13, iss. 1, pages 121-132, 2012.

[53] Marco, D., Jennings, M., *Universal meta data models*, John Willey, 2004.

[54] Tannenbaum, A., *Metadata solution*, Addison Wesley, 2002.

[55] Jiang, B., *Constructing data warehouse with metadata-driven generic operators, and more,* DBJ Publishing, 2011.

[56] Object Management Group (OMG). *Common Warehouse Metamodel Specification*. Version 1.1., Volume 1, 2003.

[57] Subotić, D., Jovanović, V., Poščić, P.: Data Warehouse and Master Data Management Evolution - A Meta-Data_vault Approach. *In: Proceedings of 54th International Conference IACIS 2014* (in print). Las Vegas, Nevada, USA, 2014.

[58] Subotić, D.: Data Warehouse Schema Evolution Perspectives. *In: Proceedings of 18th East-European Conference on Advances in Databases and Information Systems ADBIS 2014* (in print). Ohrid, Macedonia, 2014.