Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 1327–1333

DOI: 10.15439/2015F267
ACSIS, Vol. 5

Assessment of query execution performance using selected Business Intelligence tools and experimental agile oriented data modeling approach

Radek Němec
VŠB – Technical University of
Ostrava, Faculty of Economics,
701 21 Ostrava 1, Czech Republic
Email: radek.nemec@vsb.cz

Abstract—**The paper deals with the assessment of an experimental data modeling approach which is intended to support the agile oriented data modeling. The approach is based on the Anchor Data Modeling technique and is applied on a multidimensional data model. The assessed approach is expected to facilitate more effective execution of queries in the data mart environment. The emphasis is placed on the comparison of the query execution performance using database schemas, each built using traditional and the experimental approach. The tests are done in the environment of selected modern Business Intelligence tools, and using two test queries with varying output dataset sizes. The results show that the use of the database schema, created according to the experimental data modeling approach, had positive impact on the querying performance in several cases. The magnitude of impact on the querying performance, however, varied depending on each query's respective resulting dataset size.**

I. Introduction

Multidimensional data modeling principles are one of the cornerstones of the Business Intelligence (BI) system's design and development process. These principles were introduced more than 3 decades ago by Ralph Kimball and then developed to today's well-known bus architecture and dimensional modeling methodology [1]. While the BI system is usually a critical decision-making and management support system, it is crucial to devote the best of care to its development and management. Although current trends strongly promote big data as the new Holy Grail for today's CIOs, the fact is that standard relational data marts will still be a thing in the coming future. Vast amounts of company's historical business data are stored predominantly in traditional relational database environments. Moreover, the business process management will always rely on the analysis of historical facts in relation to current real-time data. This is also supported in [2] where the authors state that the data and information integration are the most fundamental issues in the integration of decision support systems into processes, to enhance decision support performance.

The dynamics of event incidence and subsequent changes in the business world produce new business process management related requirements. Therefore, high impact on almost all aspects of the data mart architecture design, usage and its continuous adaptation and evolution is experienced almost on a regular basis. According to [3], one of the information system's success dimensions is the information timeliness and currency with respect to related business processes. The adaptation to the time aspect is therefore very important and it has an impact on the quality of data used in the decision-making process. This is supported also by [4] where the perception of data warehouse system's success dimensions by its users is studied. The authors determined that the quality of the underlying data is emphasized as an important antecedent of information quality and indirectly also of the system quality as important aspects of information system's success. Other aspects, like system performance (including query execution performance), service quality and usefulness of the BI system's toolset were also studied (the query execution performance is one of the main themes in this paper).

Changes usually encompass a set of data updates and schema changes, constraint modifications (keys, value containment) or metadata adjustments [5]. Kimball has already introduced and developed methods for the time-aware evidence of critical data in business dimensions represented in a multidimensional database schema. In terms of the data mart development, the standard Kimball's approach offers several non-destructive methods of capturing changes in values of dimensional attributes (i.e. horizontal changes) [6]. Although these methods are actively used in practice, some space is still left for further research in this part of the well-established data mart development paradigm [7]. Drawbacks of these methods usually reveal themselves sooner or later during most BI projects. These include mainly notable data volume increases and obvious limits in the length of change

This paper was made with financial support of the European Social Fund within the project CZ.1.07/2.3.00/20.0296.

capture time period. The assessment of an innovative approach addressing these issues in the data modeling practice are main research interests of the author of this paper.

Issues of business change adaptation dynamics are closely related to the agile development paradigm. Agile principles have already built a strong position in the software development area. In the data modeling practice, during the information system development life cycle, there is also an effort put into the adoption of such practices and principles. Generally, the agile approach emphasizes intense cooperation with customers as a vital asset. A core principle which is promoted by the "agile", and is especially relevant to the data modeling, is the modularity principle. The modularity facilitates easier adaptation of a software solution to changes in business requirements.

Also the multidimensional data model must reflect the most current business requirements and easy adaptation of the data model to these changes is an issue that needs to be solved most effectively. [8], [9] write about such agile oriented data modeling techniques applicable in the data warehouse development field. These techniques, although referenced as very effective in the practice, pose, however, rather agile oriented data modeling process management solutions than particular agile oriented data definition solutions. This issue is particularly relevant also to the development of the data mart architecture and is addressed by the experimental approach.

In recent works [10], [11], [12] an experimental (hybrid) data modeling approach was introduced. The proposed approach is focused on representing database schema of the multidimensional data model (designed using standard Kimball's dimensional modeling process) in an agile oriented fashion – i.e. as an implicitly modular data model. For this purpose, a data modeling technique, called the anchor data modeling (ADM), is leveraged and its principles and guidelines are adapted to the needs of the multidimensional data modeling field. The ADM technique was created by Olle Regardt, Lars Rönnbäck and their colleges, and fully described in [13].

In this paper, the emphasis is placed on the assessment of the usage of the experimental data modeling approach in the environment of selected client-side Business Intelligence tools. The goal of this paper is then to compare query execution performance in selected tools using the ADM based multidimensional database schema on one side, and a more traditional multidimensional database schema on the other side (both schemas are derived from a sample multidimensional data model described further in the text). The results will help to build more evidence of the possible applicability of the mentioned experimental data modeling approach. The mentioned approach, along with other methodological background, is described in the section II and the query performance assessment is presented in section III.

II. METHODOLOGY

A. Short overview of the Anchor Data Modeling technique

By definition, the ADM technique's application should lead to the creation of a highly decomposed database schema. The relations in such schema can change in time separately, both in terms of attribute values and structure in a more effective fashion. The implicit modularity feature brings the possibility of passing changes in the semantic background of the data model to changes in the final database schema without breaking the structure of source entities. This can make the data modeling more flexible and possibly promote non-destructive ways of managing changes even in multidimensional data models, according to most recent changes in the business environment.

The ADM technique is based on the usage of several distinctive conceptual constructors that are designed for easy and understandable representation of semantic terms and the evolution of the database schema. The resulting database schema is called the anchor database schema by default. The technique is intended to be a part of the relational data management paradigm, but the usage can easily span also in the object-oriented data modeling field [13]. Each entity is represented by one *Anchor* constructor and a set of *Attribute* constructors – both terms conceptually represent basic semantic features of an entity (name and properties). Each *Attribute* is dedicated to serve as a representation of each entity's property (i.e. attribute), containing usually only identifier (composite key) and an actual value of the entity's property. *Anchors* are connected using *Ties* which represent, by default, *M to N* relationships between entities (i.e. *Anchors*).

B. Description of the experimental data modeling approach and differences from the traditional approach

Let a simple multidimensional data model be defined as a set of i dimensions DIM_i and a fact records subset. The fact records subset is matrix of records comprised of n quantitative process performance measurements M so that each fact record is represented as $facts = \{M_1, M_2, \dots, M_n\}$. Each dimension has j dimensional properties P, i.e., each $DIM_i = \{P_{i,1}, P_{i,2}, \dots, P_{i,j}\}$. The description of the difference between traditional and the anchor data modeling (ADM) based database schemas follows.

Let the traditional multidimensional relational database schema, derived from the simple multidimensional data model ("trad."), be a set of i relational tables, each for one dimension DIM_i, and one relational table for the fact records subset (the fact table). Each relational table DIM_i consists of j+1 columns C, one column C_{K} for the primary key and others for each property P_j. The whole tuple of the i-th dimension is then $DIM_i = \{C_{i,\mathrm{K}}, C_{i,1}, C_{i,2}, \dots, C_{i,j}\}$. All dimensional properties are then included in one relational table. The fact table consists of n respective metrics M and a subset of composite key columns K^*_{DIM}, each referencing the C_{K} column in one of the DIM_i. The whole tuple of the fact table of the

"trad." schema is then $facts = \{K^*_{DIM_1}, K^*_{DIM_2}, \dots, K^*_{DIM_i}, M_1, M_2, \dots, M_n\}$.

Let the ADM based multidimensional relational database schema, derived from the simple multidimensional data model ("ADM"), be defined as a set of i *Anchor* relational tables A_i^{DIM} (one for each dimension DIM_i) and a set of $i \times j$ *Attribute* relational tables $Attr_{i,j}^{DIM}$ (one for each property P_j of i-th dimension DIM_i). Each A_i^{DIM} contains only identifying columns C_K of i-th dimension, i.e. $A_i^{DIM} = \{C_{i,\mathrm{K}}\}$. Inclusion of more attributes is, however, possible, e.g. to store ETL (ELT) process metadata (e.g. data quality related information). Each *Attribute* relation should contain only identifying composite key column and another column C^* for j-th property of i-th dimension, i.e. $Attr_{i,j}^{DIM} = \{C^*_{i,\mathrm{K}}, C_{i,j}\}$. The primary key of the *Attribute* relation can then be extended with additional datetime columns if the historization is to be applied to the respective dimension's property. So one less *Attribute* relation must be created to represent the same dimension as it is in the traditional schema, but each *Attribute* should have identifying column, which is the obvious drawback of the approach, in this regard. The composite key column in the *Attribute* relation relates to the primary key column in the *Anchor* relation. Most modern database systems can perform elimination of tables in joins in query optimization procedures. This feature mitigates join demands of a hierarchically more extensive query by excluding tables from join from which no columns are selected to be used in the output of a query [13]. The fact table again consists of n respective metrics M and then a subset of composite key columns $K^*_{A^{DIM}}$, each referencing the C_K column in one of the A_i^{DIM} relations. The whole tuple of the fact table is then, in fact, the same as for the "trad." schema, therefore $facts = \{K^*_{A_1^{DIM}}, K^*_{A_2^{DIM}}, \dots, K^*_{A_i^{DIM}}, M_1, M_2, \dots, M_n\}$.

The modular nature of the resulting anchor database schema implies normalization of relations into the 5th normal form. The schema can then be qualified as a highly decomposed[1]. Moreover, if the historization of *Attribute* values is applied, the *Attribute* relation is in the 6th normal form [14] and the primary key of the *Attribute* relation is complemented with a time validity aspect (i.e. column with a date and time data type).

The normalization of all relations in the anchor database schema into the 6th normal form is implied by original ADM technique's guidelines [13]. However, the results in previous related publication [15], concerning one of the first applications of the ADM in the multidimensional data modeling field, indicated that the normalization of the fact table leads to severe querying performance problems. Therefore, in the presented experimental approach, only Attribute relations in the schema should be normalized into the 6th normal form, if it is desired.

[1] From a conceptual point of view, the topology of the data model (and the database schema in the end) gets close to the snowflake topology, while the traditional data model stays with the classical star topology.

One of the expected benefits from using the ADM technique, when constructing a multidimensional database schema, relates to typical user behavior in the analysis and reporting – users typically select only few dimensional properties in their queries. Also, relevant values of these properties are usually filtered so that only a limited set of values is used to calculate the final results of the query. The high degree of normalization that is applied in the ADM based multidimensional database schema, then leads to the elimination of the need for scanning whole rows of a dimensional table (potentially large one). This aspect is then mirrored in expected query execution performance benefits. Also the straightforwardness and ease of applying implicit *Attribute* values' historization (each attribute can be historized separately) is a notable asset of the experimental approach. The high degree of dimensional properties' decomposition offers also a possibility of effective use of specific query processing optimization techniques, including compression and table data pre-ordering. This may bring some enhancements into already used relational data management environments, similar to those known to be present in columnar storage engines.

C. Description of the sample multidimensional data model

The multidimensional data model that was used as a source for the creation of the sample database schemas is a typical banking model with 6 dimensions, as presented by Kimball and Ross [1]. Fact records represent monthly account data with cardinality of 50 million rows. Dimensions also come from the same publication and provide descriptive data to Accounts (220 000 rows), Households (i.e. Customers, 200 000 rows), Banking products (20 rows), Branches (1000 rows) and Account states (3 rows). The time dimension defining the monthly granularity of facts spans over 12 years (2000–2012; 156 rows). Fig. 1 provides a conceptual outlook on the structure of the sample multidimensional data model.

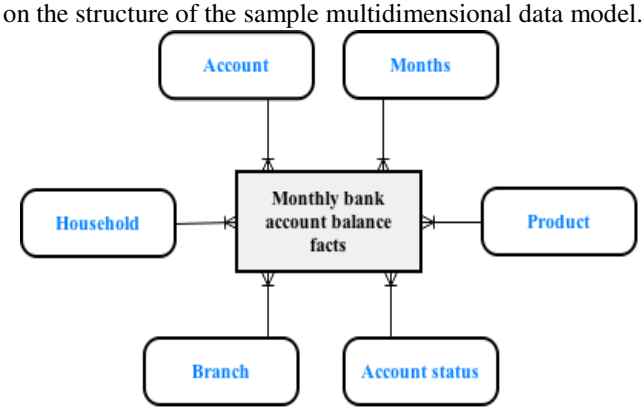

Fig. 1 Conceptual view on the sample multidimensional data model

D. Description of the query testing process

None of the client-side tools used, however, support usage of an aggregated SQL query definition of data source for a

report. Both test queries therefore had to be stripped of aggregate functions and group by statements. This paper will, therefore, deal with query execution time (QET) results related to full dataset extraction because this mode is supported by all of the BI tools used in the test. The time dimension filtering condition was set to filter 5 specific calendar years (2008 to 2012) along with additional filtering options. In the appendix, there are both SQL queries listed, in both versions for both schema variants (with mentioned additional filtering options). Each query was then executed in each BI tool and query performance results were gathered using Microsoft SQL Profiler. This tool was used to get data on QET results, as well as actual CPU demands and the count of logical data reads performed during the execution of each query in each tool. Also the Apache jMeter 2.10 tool was used for the execution of specified test queries directly on the database server.

The testing process was done using a database server with Microsoft SQL Server 2012 64-bit bundle installed, with following server hardware configuration: Intel Xeon Quad-core 2.66 GHz CPU and 4 GB RAM. Although the server hardware is not the best-of-breed in the data warehousing field, I was able to get meaningful query execution performance results with it. Client-side tools were installed and run on a standard PC with 8 GB RAM and Intel Quad-Core processor.

E. Business Intelligence client-side tools for the assessment

Gartner Research [16] published the 2015 version of their annual 'Magic Quadrant for Business Intelligence and Analytics Platforms' research report. The list of tools in this analytical report was a main source of tools that were assessed for the use in the process of query execution performance analysis. BI tool selection process respected following criteria:

1) the software is a client-side and stand-alone tool, without the need for installation of any application and/or OLAP server,
2) the software has a trial or free desktop version available for download,
3) the software is a BI reporting/dashboard management tool,
4) the software allows seamless connection to a relational database (SQL Server 2012 specifically).

The final list (Table I) includes 6 tools that passed the criteria and were successfully installed and run without stability and connectivity issues (both queries were executed without crashes and/or timeout problems). All tools are modern solutions, allowing for fast and intuitive data discovery and visualization. User interfaces offer self-service functionalities and also several data analysis features. Database connectivity features include connection to structured as well as unstructured data management solutions. The Tableau Desktop software also offers very fast on-the-fly in-memory computing capabilities. According to the Gartner Research report [16], all vendors of these tools, except for Tibco, are recognized as 'Leaders' in the magic quadrant chart (Tibco is viewed as a 'Visionary' and is located very close to the 'Leaders' quadrant). So the way how the data is handled and processed internally in each tool's software environment, and using respective connectivity interfaces, is the main differentiating factor for all 6 tools.

TABLE I.
SELECTED BI TOOLS AND DATABASE CONNECTIVITY PROPERTIES

Short code	BI tool name	Database connectivity interfaces
Tibco	Tibco Spotfire Desktop 7.0	.NET data provider
Tableau	Tableau Desktop 8.3	ODBC
Lumira	SAP Lumira 1.23	JDBC
MSTR	Microstrategy Analytics Desktop 9.4	ODBC
Qlik	QlikSense Desktop 1.0	ODBC, OLE DB
MSPP	Microsoft PowerPivot for Excel 2013	OLE DB, .NET data provider, ODBC

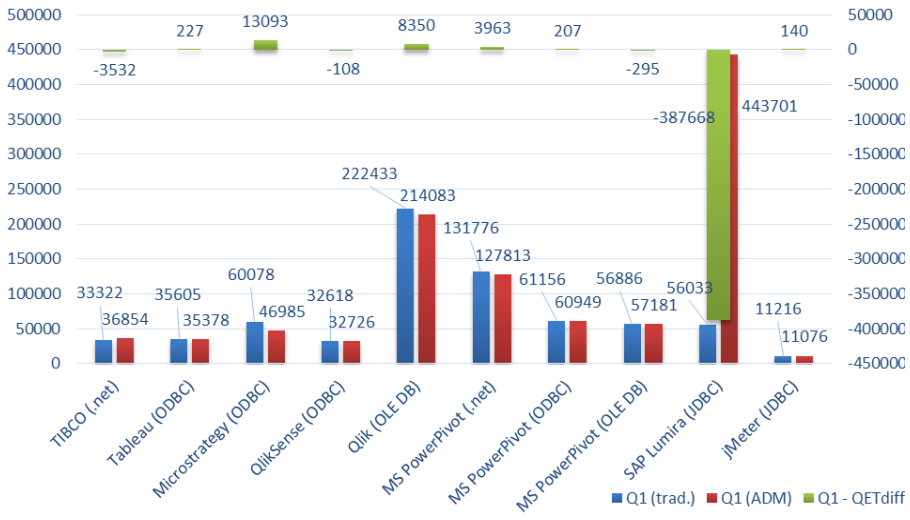

Fig. 2 QET results for the query Q1 in selected tools with QET differences between schema variants

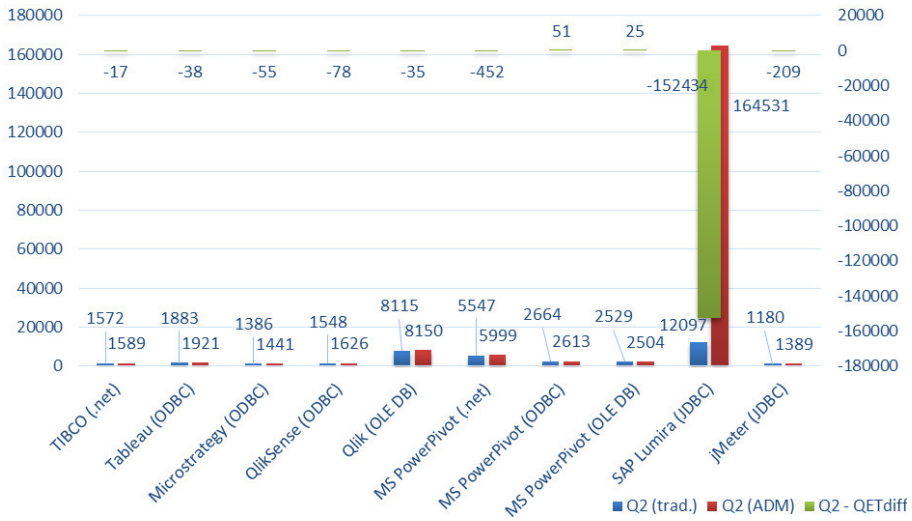

Fig. 3 QET results for the query Q2 in selected tools with QET differences between schema variants

III. Results of Query Execution Performance Tests and Discussion

The results of the query performance testing process, which took into consideration both the highly decomposed ADM based database schema of the sample multidimensional data model and the traditionally structured counterpart provided interesting insights.

Fig. 2 (Q1) and Fig. 3 (Q2) show full dataset extraction QET results acquired for both queries after its execution in each tool as well as the on-server result obtained using the jMeter. Figures show also difference values QET_{diff} that were calculated according to the formula: $QET_{diff} = QET_{trad.} - QET_{ADM}$.

The results for the query Q1 show that the usage of the ADM based database schema was beneficial in 4 cases as seen in the Fig. 2. The Tibco tool had bigger problems with the ADM based schema. The Qlik had slight problems too, but the QET_{diff} was only -108 ms in this case. The usage of the OLE DB interface in the MSPP resulted in the QET_{diff} = -295 ms which was also still relatively close to zero.

Since the query Q1 results in a large dataset to be extracted from the database (9 615 712 rows), the greatest difference may lie in the way of how each tool processes large datasets are being extracted. The Lumira is doing a bad job in this matter because resulting QET_{diff} values were noticeably greater in comparison with other tools. The difference between QET value obtained for the ADM based schema and the traditional one came out largely in favor of the traditional schema. The client-side usage of the JDBC interface may have some part in the magnitude of QET value differences along with the connection to the database server through LAN. All in all, the Lumira, as a stand-alone client-side BI tools, is clearly not beneficial for usage with the ADM based schema. Table II shows a comparison of actual CPU demands and the amount of data reads performed during the usage of each tool.

TABLE II.
PHYSICAL CHARACTERISTICS OF QUERY Q1'S EXECUTION

BI tool	Interface	Q1 (trad.)		Q1 (ADM)	
		CPU	Reads	CPU	Reads
Tableau	ODBC	13530	199556	14547	199586
Tibco	.NET d. p.	8827	199556	11391	199586
MSTR	ODBC	10109	199575	10938	199562
Qlik	ODBC	9141	199542	12812	200395
	OLE DB	12482	200081	12687	199626
MSPP	.NET d. p.	11857	199954	14033	200137
	ODBC	11936	199542	14124	199562
	OLE DB	12860	200107	14238	200375
mean		**11343**	**199739.1**	**13096**	**199853.6**
std. dev.		**1761**	**259.1**	**1370.9**	**379.9**
jMeter	JDBC	8360	199825	10000	199848
Lumira	JDBC	51641	38662071	429640	230975841

In the table II, it is visible that each tool's internal algorithm of a large dataset processing plays an important role when using a more normalized data source (besides the perceived problem with Lumira).

Results for the query Q2 (Fig. 3) show that the usage of the ADM based database schema was beneficial in 2 cases (if we take into account only positive QET_{diff} values). The lowest QET_{diff} value is indicated only in the case of MSPP using .NET data provider interface (QET_{diff} = -452 ms). In almost all cases, however, the QET_{diff} was relatively low (in absolute terms and again excluding the Lumira case). The QET result obtained using the jMeter was very close to 4 tools which may, however, be mainly due to the nature of the query Q2 and resulting row count. Nevertheless, the QET_{diff} = -209 ms which is relatively lower than most client-side tools (excluding cases of Lumira and MSPP using .NET interface). The size of the Q2's resulting dataset (507 978 rows) points out on one possible effect that stems

from the high normalization of the ADM based database scheme – if the size of the dataset gets lower the QET_{diff} differences get closer to zero (except some cases).

Table III shows a comparison of actual CPU demands and the amount of data reads performed during the usage of each tool and query Q2.

TABLE III.
PHYSICAL CHARACTERISTICS OF QUERY Q2'S EXECUTION

BI tool	Interface	Q2 (trad.)		Q2 (ADM)	
		CPU	Reads	CPU	Reads
Tableau	ODBC	1781	16033	1516	13959
Tibco	.NET d. p.	1281	15471	1595	13407
MSTR	ODBC	1514	15579	1418	13484
Qlik	ODBC	1655	16057	1671	13435
	OLE DB	1688	16145	2141	14456
MSPP	.NET d. p.	1968	16033	1874	13937
	ODBC	1640	15451	1891	13379
	OLE DB	1936	16033	1936	16042
mean		**1683**	**15850.3**	**1755.3**	**14012.4**
std. dev.		**222.9**	**294.4**	**244.3**	**902.8**
jMeter	JDBC	1595	15451	1717	13379
Lumira	JDBC	11625	12529763	94125	3824425

For this relatively smaller dataset (in comparison to the Q1's resulting dataset), it is evident that each tool's internal algorithm of the extracted dataset processing played relatively less important role, i.e. when using a more normalized data source (again besides the already perceived problem with Lumira's usage).

As for the related works, there are, regrettably, still no other works yet that deal specifically with the assessment of ADM technique's use. Besides the original paper [13] there are books [8], [9] that address agile oriented data modeling process guidelines in the traditional development of a data warehouse. However, there are works that deal with certain data modeling issues that are also addressed by the experimental approach (although mostly the conceptual part of the data modeling process is handled). Evolutionary aspects of the data in the data warehouse are dealt with in [17] and [18]. These works deal with the addition of temporal aspects into the UML based logical multidimensional data model (i.e. class model in which the conceptual model decomposition can be handled quite easily). In the paper [18], a rather general solution using only specific classes with temporal properties (a prototype based example of the application is presented) was proposed. In the paper [17], a more complex conceptual modeling approach was proposed. Along with modelling time-varying dimensional property objects, the authors propose also a solution for modelling time-varying hierarchies and hierarchy levels. Moreover, the way of mapping the resulting UML class schema to the entity relationship model of a data mart is proposed, although the paper lacks a practical example. In the paper [19], a graph theory based hybrid modeling method is introduced. The approach decomposes entity properties (attributes) and mutual relationships into graph nodes and edges. The schematic representation of data sources and requirements is then combined with the graph based representation and a conceptual multidimensional data model is derived. The changes in entities of the data model are induced by checking requirement-derived constraints, but time-validity aspects are not considered in the approach. The authors also present a short application example.

IV. CONCLUSION

The query execution performance results were analyzed with the conclusion that the ADM based schema performed better in specific cases (BI tools). The performance differences were more in favor of the ADM based schema if the source dataset was larger rather than smaller. The larger dataset related results had, however, wider spread in maximum/minimum QET results. These facts will be studied in more detail in further research, also in contrast with the aggregated query execution performance results. Also, differences in the use and demands of particular physical query processing operators will provide important insights on how are the test queries internally processed.

The results indicate that the ADM based hybrid data modeling approach has certain future potential, although more evidence will be vital to justify its practical usefulness. The potential may be further increased if the application of more advanced query optimization techniques will have a positive effect on the query execution performance – both synthetic on-server and in-tool test results will be compared in this matter.

Also, further research effort will focus on using other BI tools, especially those that can or need to use separate OLAP/application server which may provide additional benefits. The expectation here is that the direct communication of the database server and the OLAP/application server may have additional benefits regarding query execution performance when using the ADM based multidimensional database schema.

REFERENCES

[1] R. Kimball a M. Ross, *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*, 3rd ed., New York: Wiley, 2013.

[2] S. Liu, A. H. B. Duffy, R. I. Whitfield a I. M. Boyle, „Integration of decision support systems to improve decision support performance," *Knowledge Information Systems,* vol. 22, no. 3, pp. 261-286, 2010. DOI: 10.1007/s10115-009-0192-4.

[3] W. H. DeLone a E. R. McLean, „Measuring Success: Applying the DeLone & McLean Information Systems Success Model," *International Journal of Electronic Commerce*, vol. 9, no. 1, pp. 31-47, 2004.

[4] R. R. Nelson, P. A. Todd a B. H. Wixom, „Antecedents of Information and System Quality: An Empirical Examination Within the Context of Data Warehousing," *Journal of Management Information Systems*, vol. 21, no. 4, pp. 199-235, 2005. DOI: 10.1080/07421222.2005.11045823.

[5] E. A. Rundensteiner, A. Koeller a X. Zhang, „Maintaining Data Warehouses over Changing Information Sources," *Communications of the ACM*, vol. 43, no. 6, pp. 57-62, 2000. DOI: 10.1145/336460.336475.

[6] T. Torey, S. Lightstone, T. Nadeau a H. Jagadish, *Database Modeling and Design: Logical Design*, 5th ed., Burlington: Morgan Kaufmann, 2011.
[7] S. Rizzi, „Conceptual Modeling Solutions for the Data Warehouse," In *Data Warehouses and OLAP : Concepts, Architectures and Solutions*, Hershey, IGI Global, 2007, pp. 1-26.
[8] S. Ambler, *Agile Database Techniques: Effective Strategies for the Agile Software Developer*, New Jersey: Wiley, 2003.
[9] L. Corr, *Agile Data Warehouse Design*, Leeds: DecisionOne Press, 2014.
[10] R. Němec a F. Zapletal, „The Design of Multidimensional Data Model Using Principles of the Anchor Data Modeling: An Assessment of Experimental Approach Based on Query Execution Performance," *WSEAS Transactions on Computers,* vol. 13, pp. 177-194, 2014.
[11] R. Němec a F. Zapletal, „Analysis of Query Execution Performance Factors in the Anchor Multidimensional Database Schema Environment," in *Selected Paper of MEKON 2014 Conference*, Ostrava, 2014, pp. 105-117.
[12] R. Němec, „The Analysis of Historization Technique in Context of Handling Changes in Dimensions in Multidimensional Model and Anchor Data Modeling," in *Proceedings of the 10th International Conference on Strategic Management and Its Support By Information Systems SMSIS 2013*, Ostrava, 2013, pp. 135-146.
[13] O. Regardt, L. Rönnbäck, M. Bergholtz, P. Johannesson a P. Wohed, „Anchor Modeling: An Agile Modeling Technique Using the Sixth Normal Form for Structurally and Temporally Evolving Data," in *Conceptual Modeling - ER 2009 (Lecture Notes in Computer Science 5829)*, Rio Grande do Sul, 2009, pp. 234-250. DOI: 10.1007/978-3-642-04840-1_19.
[14] C. J. Date, H. Darwen a N. A. Lorentzos, *Temporal Data and the Relational Model: A Detailed Investigation into the Application of Interval and Relation Theory to the Problem of Temporal Database Management*, Oxford: Elsevier LTD., 2003.
[15] R. Němec, „The Comparison of Anchor and Star Schema from a Query Performance Perspective," in *World Academy of Science, Engineering and Technology, issue 71*, Paris, 2012, pp. 1718-1722.
[16] R. L. Sallam, B. Hostmann, K. Schlegel, J. Tapadinhas, J. Parenteau and T. W. Oestreich, "Magic Quadrant for Business Intelligence and Analytics Platforms 2015," Gartner Research, 2015, 2015-02-23, URL: http://www.gartner.com/technology/reprints.do?id=1-2ACLP1P&ct=150220&st=sb.
[17] E. Malinowski a E. Zimányi, „A conceptual solution for representing time in data warehouse dimensions," in *Proceedings of the 3rd Asia-Pacific conference on Conceptual modelling*, Darlinghurst, 2006, pp. 45-54. DOI: 10.1145/1151855.1151861.
[18] F. Ravat, O. Teste a G. Zurfluh, „Towards Data Warehouse Design," in *Proceedings of the eighth international conference on Information and knowledge management*, Kansas City, 1999, pp. 359-366. DOI: 10.1145/319950.320028.
[19] F. Di Tria, E. Lefons a F. Tangorra, „GrHyMM: A Graph-Oriented Hybrid Multidimensional Model," in *Advances in Conceptual Modeling (ER 2011 Workshops, LNCS 6999)*, Brussels, 2011, pp. 86-97. DOI: 10.1007/978-3-642-24574-9_12.

APPENDIX

Both queries Q1 and Q2, in both versions for both multidimensional database schema variants, follow:

```
/*Q1 – trad.*/
SELECT
  bproduct_nazev, /*SUM*/ pocet_transakci, kal_rok, mesic_nazev,
  bproduct_typ
FROM
  FACTmesicni_stav_uctu_snimek
  INNER JOIN DIMmesic ON FACTmesicni_stav_uctu_snimek.monthID
  = DIMmesic.monthID
  INNER JOIN DIMbankovni_produkt ON
  FACTmesicni_stav_uctu_snimek.bproductID =
  DIMbankovni_produkt.bproductID
WHERE
   kal_rok IN (2012, 2011, 2010, 2009, 2008)
   AND bproduct_typ IN ('běžný účet', 'hypotéka', 'leasing', 'termínovaný
   vklad', 'spotřebitelský úvěr')

/*Q1 – ADM*/
SELECT
  BP_BPN_bproduct_nazev, /*SUM*/ pocet_transakci,
  ME_KRO_kal_rok, ME_MNA_mesic_nazev, BP_BPT_bproduct_typ
FROM
  FACTmesicni_stav_uctu_snimek
  INNER JOIN BP_DIMbankovni_produkt ON
  FACTmesicni_stav_uctu_snimek.BP_ID_byBProdukt =
  BP_DIMbankovni_produkt.BP_ID
  INNER JOIN BP_BPN_bproduct_nazev ON
  BP_BPN_bproduct_nazev.bp_id = BP_DIMbankovni_produkt.bp_id
  INNER JOIN BP_BPT_bproduct_typ ON BP_BPT_bproduct_typ.bp_id
  = BP_DIMbankovni_produkt.bp_id
  INNER JOIN ME_DIMmesic ON
  FACTmesicni_stav_uctu_snimek.ME_ID_byMesic =
  ME_DIMmesic.ME_ID
  INNER JOIN ME_KRO_kal_rok ON ME_KRO_kal_rok.me_id =
  ME_DIMmesic.ME_ID
  INNER JOIN ME_MNA_mesic_nazev ON
  ME_MNA_mesic_nazev.me_id = ME_DIMmesic.ME_ID
WHERE
   ME_KRO_kal_rok IN (2012, 2011, 2010, 2009, 2008)
   AND BP_BPT_bproduct_typ IN ('běžný účet', 'hypotéka', 'leasing',
   'termínovaný vklad', 'spotřebitelský úvěr')

/*Q2 – trad.*/
SELECT
  pobockaID, /*COUNT*/ ucetID, DIMmesic.kal_rok,
  DIMpobocka.pobocka_adresa_mesto, DIMmesic.mesic_nazev
FROM
  FACTmesicni_stav_uctu_snimek
  INNER JOIN DIMucet ON FACTmesicni_stav_uctu_snimek.ucetID =
  DIMucet.ucetID
  INNER JOIN DIMmesic ON FACTmesicni_stav_uctu_snimek.monthID
  = DIMmesic.monthID
  INNER JOIN DIMpobocka ON
  FACTmesicni_stav_uctu_snimek.pobockaID = DIMpobocka.pobockaID
WHERE
   Year (DIMucet.ucet_otevren) IN (2008, 2009, 2010, 2011, 2012)
   AND DIMmesic.kal_rok IN (2012, 2011, 2010, 2009, 2008)
  AND DIMpobocka.pobocka_adresa_mesto IN ('Praha', 'Bohumín',
    'Ostrava', 'Jindřichův Hradec', 'Olomouc')

/*Q2 – ADM*/
SELECT
  PB_ID_byPobocka, /*COUNT*/ UC_ID_byUcet, ME_KRO_kal_rok,
  PB_PME_pobocka_adresa_mesto, ME_MNA_mesic_nazev
FROM
  FACTmesicni_stav_uctu_snimek
  INNER JOIN UC_DIMucet ON
  FACTmesicni_stav_uctu_snimek.UC_ID_byUcet = UC_DIMucet.uc_id
  INNER JOIN UC_UOT_ucet_otevren ON UC_UOT_ucet_otevren.uc_id
  = UC_DIMucet.uc_id
  INNER JOIN ME_DIMmesic ON
  FACTmesicni_stav_uctu_snimek.ME_ID_byMesic =
  ME_DIMmesic.me_id
  INNER JOIN ME_KRO_kal_rok ON ME_KRO_kal_rok.me_id =
  ME_DIMmesic.me_id
  INNER JOIN ME_MNA_mesic_nazev ON
  ME_MNA_mesic_nazev.me_id = ME_DIMmesic.me_id
  INNER JOIN PB_DIMpobocka ON
  FACTmesicni_stav_uctu_snimek.PB_ID_byPobocka =
  PB_DIMpobocka.pb_id
  INNER JOIN PB_PME_pobocka_adresa_mesto ON
  PB_PME_pobocka_adresa_mesto.pb_id = PB_DIMpobocka.pb_id
WHERE
  Year (UC_UOT_ucet_otevren) IN (2008, 2009, 2010, 2011, 2012)
  AND ME_KRO_kal_rok IN (2012, 2011, 2010, 2009, 2008)
  AND PB_PME_pobocka_adresa_mesto IN ('Praha', 'Bohumín', 'Ostrava',
  'Jindřichův Hradec', 'Olomouc')
```