J. Albrecht, W. Lehner, M. Teschke, T. Kirsche

Building a real data warehouse for market research

Erstveröffentlichung in / First published in:

Database and Expert Systems Applications. 8th International Conference, DEXA '97. Toulouse, 01.-02.09.1997. IEEE, 2002. S. 651-656. ISBN 0-8186-8147-0

DOI: https://doi.org/10.1109/DEXA.1997.617407

BUILDING A REAL DATA WAREHOUSE FOR MARKET RESEARCH

Jens Albrecht, Wolfgang Lehner, Michael Teschke
IMMD 6, University of Erlangen-Nuremberg
e-mail: {jalbrecht, lehner, teschke}
@informatik.uni-erlangen.de

Thomas Kirsche
GfK Handelsforschung GmbH
Germany
e-mail: thomas.kirsche@gfk.de

Abstract

This paper reflects the results of the evaluation phase of building a data production system for the retail research division of the GfK, Europe's largest market research company. The application specific requirements like end-user needs or data volume are very much different from data warehouses discussed in the literature, making it a real data warehouse. In a case study, these requirements are compared with state-of-the-art solutions offered by leading software vendors. Each of the common architectures (MOLAP, ROLAP, HOLAP) was represented by a product. The result of this comparison is that all systems have to be massively tailored to GfK's needs, especially to cope with meta data management or the maintenance of aggregations.

1. Introduction

The process of retail research includes three major steps. First of all, the fact data, e.g. sales, stock, or price figures of products in the monitored shops are gathered periodically. In the data production phase, the monitored data are cleaned and extrapolated to a representative production data basis. Finally, market analysts use this data basis to generate reports in which the data is selected, aggregated and compared according to application criteria.

The data of the production data basis is divided into qualifying and quantifying information. The qualifying data consists roughly of the three dimensions *product*, *time* and *shop*, along which the quantifying data, i.e. the measured facts, are organized. The production and analysis of the data is always organized according to reporting periods, although some data for the next period might be collected while the current period is still being produced.

To improve the current production system and build a real data warehouse (the so-called *U*niform *D*ata *S*upply project), the GfK invited several leading software vendors to offer solutions meeting their requirements in a case study. All vendors had the possibility to implement a prototypical solution for a specified scenario. The implementations were performed on state-of-the-art systems available in December 1996. Promises like "in version X, everything will be perfect" were neglected in the product evaluation. The case studies did not cover the cleansing and transformation phase but were based on a sample clean production data basis.

One of the main problems of the GfK-scenario is to switch from a retail oriented data delivery to customer oriented reports based on product groups. Thus, the collected data have to be completely reorganized and calculated. The task is not finished when the data warehouse has been built, but the work starts with its exploitation.

The remainder of the paper is organized as follows. First, the requirements outlined in this introduction are detailed in section 2 considering conceptual as well as implementational aspects. These requirements serve as a skeleton for the description of the case studies described in section 3. In sections 4 and 5, two of the requirements, the management of master data and of aggregations are highlighted both, from a related scientific work and existing products point of view. The paper concludes with a summary of the results of the case studies.

2. External Requirements

Two classes of users must be sufficiently supported: On the one hand, at the end of every analysis period, a collection of predefined reports (about 1000 pages each) is printed in a batch-oriented manner and sold to the customers. On the other hand, ad-hoc users must be supported during the complete analysis phase of the available data. These end-users are mostly consultants or statisticians which would like to speak in business terms, e.g. channel types or product features and navigate with a point-and-click interface in a multidimensional data cube instead of formulating SQL-queries. Their kind of work is typical "*On-line* Analytical Processing" (OLAP, [6]), meaning that the results of complex queries have to be delivered within seconds. The second external requirement is to handle extremely large data volumes. Currently, the fact data volume for a single country (Germany = 3000 monitored shops) within a single reporting period is 5 GByte.

The 250.000 articles of the product dimension are classified into 400 product groups. Queries refer to these classification hierarchies as well as to attributes describing the articles of the corresponding product group in detail. Each product group has roughly 3-50 specific attributes. The geography dimension contains the shops reporting the measured facts. Reports are required on a regional (e.g. North vs. South Bavaria) as well as on an international basis. The time dimension is expected to be refined in two ways: the gathering frequency shall be shortened from currently four weeks to one week and the data is kept on-line for a longer time (five years instead of one year) to enable long-term, i.e. time sequence, analysis.

3. Case Studies

The focus of the case studies, performed in the product evaluation phase of the UDS-project at GfK, was to search for a solution, which is suitable for the requirements listed above. Thus, the central question was: "How much do I have to implement myself and how much is covered by the system?" Within this section, the different architectures and products involved in the studies are outlined. For each architectural design, a commercial product is characterized briefly (Figure 1). In the second part of this section, the experiences with the systems are reflected and evaluated. It will be seen that two requirements are essential: "management of master data" and "management of aggregations".

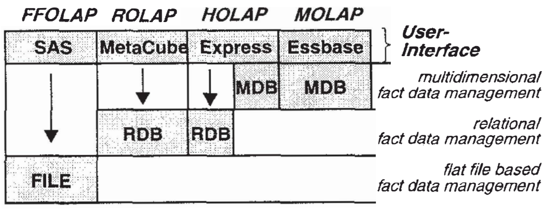

FFOLAP
ROLAP
HOLAP
MOLAP
SAS
MetaCube
Express
Essbase
User-Interface
MDB
MDB
multidimensional fact data management
RDB
RDB
relational fact data management
FILE
flat file based fact data management

Figure 1: Architectural designs used in the case study

3.1. Architectural Designs

A prerequisite for the products being evaluated was to meet the external requirements. First, the user level must provide a multidimensional user interface enabling the data analysis by use of "business terms". Second, the system should be capable of handling the large data volumes.

MOLAP ("Multidimensional OLAP")

In "Multidimensional OLAP", the multidimensional view is directly reflected in multidimensional storage structures to efficiently support "slice"/"dice" and "drill-down"/ "roll-up" operations. The representative of a pure MOLAP-approach was Essbase from Arborsoft ([1]). The system consists of two components. The client is a typical OLAP-interface which communicates with the Essbase Analysis Server via an open communication protocol. The Essbase philosophy is to periodically load the multidimensional server from SQL database systems or other legacy systems and allow read-only access to the multidimensional data. Updates are considered with the next reload of the multidimensional analysis server. After the raw data is loaded, Essbase pre-aggregates all possible aggregation combinations wrt. the hierarchical dimensional structures.

ROLAP ("Relational OLAP")

In opposite to the MOLAP-approach, "Relational OLAP" visualizes a relation as a multidimensional cube. Therefore, all OLAP-operations are directly translated into corresponding relational operations, i.e. into complex SQL statements. No special multidimensional storage structures are needed. The representative of the ROLAP-approach in the case study was MetaCube from Informix ([14]). The center of the MetaCube architecture consists of the MetaCube OLAP Server. This server translates queries, formulated in the MetaCube Explorer or other ODBC compliant client applications, into SQL92 conform statements, which are processed by an Informix OnLine Dynamic Server. Additional agents allow to automate the work of users and administrators. For example, the "Query Back" agent runs long queries in the background, and the "Aggregator" agent creates and maintains simple pre-aggregates in the data warehouse environment.

HOLAP ("Hybrid OLAP")

The "Hybrid OLAP" approach tries to combine the advantages from both MOLAP and ROLAP. Here, the multidimensional component acts as a cache for the data stored in the relational database. In order to answer an OLAP-query, first the multidimensional cache is checked. If the requested data cannot be found there, appropriate SQL-queries are generated and sent to the relational database. Thus, without sacrifying the multidimensional paradigm, large data sets can be handled. If cached data are requested, the query performance is higher than in pure ROLAP systems. The representative of the HOLAP-approach was Oracle's Express ([19]) as multidimensional database system in combination with the Relational Access Manager (RAM, [20]) which coordinates the flow of data between the Oracle's relational engine and Express. As with MetaCube, all mappings of multidimensional to the relational data must be performed by the DBA.

FFOLAP ("Flat File OLAP")

The fourth system being investigated during the UDS project was the SAS system ([25]). SAS is based on the file system and provides a sophisticated query language for programming statistical queries. The SAS data warehouse is specifically designed for data retrieval and not for writing. Therefore, SAS does not carry the transaction performance overhead of typical database systems. Relationships beween data are determined dynamically upon request by the metadata stored in a special repository, thus enabling rapid processing of the request. Metadata are updated dynamically with usage data generated during the exploitation of the data warehouse. SAS makes use of special data set indexes to increase query performance. Data compression techniques are used for reducing the data volume and thus reducing I/O traffic. For the access, manipulation and reporting SAS provides a comprehensive set of tools.

3.2. Evaluation of the Case Studies

The case studies were performed at the GfK by consultants of the product vendors. All participants got about one week of time to implement a specified subset of the typical query scenario (13 GfK market research analysis queries).

User Interfaces

All products have been proven to provide an adequate user interface. Therefore, all systems were able to perform OLAP-functionality at the user level. Moreover, MetaCube and Essbase provide an open interface to attach a 3rd-party user interface product.

Data Modeling

The SAS system provides a non-standard data model (data sets) and a 4GL query language (data steps) and is therefore in a somewhat special position. Without an expert knowledge of data steps, the mapping of the application scenario done by the consultants to this data model could not be figured out clearly. Furthermore, the implementation explicitly used internal knowledge of the system. Especially meta information was retrieved from system tables thus giving up logical as well as physical data independance.

The multidimensional representatives had difficulties in modeling the classification dependent attributes within the product and shop dimension. Basically, there are three possible alternatives. The first alternative, proposed by Essbase, is to model each attribute in a single dimension. Therefore, beside the product, time and region dimension, there would be further dimensions like brand, water usage or video system (Figure 2a). This modeling approach is the straight-forward solution with regard to the end user's view. The big drawback in this approach is to maintain the possible large data cube at the conceptual and internal layer.

The second alternative, proposed by the Express consultants, is to model the attributes in a special relation. In Express terminology, it means that a special "feature" dimension has to be created, where the values of the attributes are the dimension elements and the nodes at the first classification level in this dimension would correspond to the attribute names (Figure 2b). An Express relation is used to create a mapping between the products resp. shops and their features. The selection of splits into different features is reduced to a slice operation wrt. the feature dimension. The disadvantage is the misusage of classification structures to reflect the schema and instances of single attributes. Since dimensional attributes must be unique within a dimension, specific attribute values (like "Yes" or "No") are shared by different attributes. Therefore, the selection process over more than one attribute has to instantiate the feature dimension for each split.

MetaCube's people first tried to introduce a "-1"-level in each classification hierarchy holding the attribute/value pairs of each former leaf node (Figure 2c). The advantage of this approach is to reflect the idea of a feature split directly within the drill-down semantics. To report totals without any split, an explicit ALL-value for each attribute was introduced as an artificial attribute value. Furthermore, this approach implies that the fact table does not hold the article identifier but an identifier for each attribute/value pair.

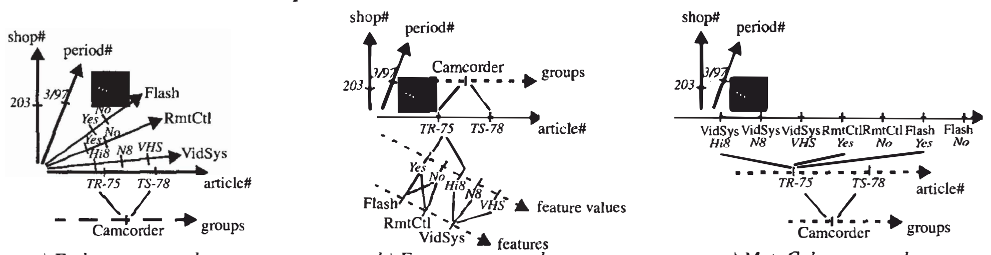
In a second try, the MetaCube people modeled the features as attributes in the lookup-tables of the underlying relational star-schema. This approach leads to a clean design (distinction of classification and description, article identifier in the fact table), but results in the general big drawback of modeling features globally: Since the feature schema is local to product groups and depends on the classification structure, the lookup-table becomes extremely sparse.

Practical Tests

Since the Essbase system philosophy is to calculate all possible aggregation combinations, and the number of possible combinations is very high in the context of feature description in combination with classification hierarchies, the raw data volumes would be multiplied in order of magnitudes. Thus, Essbase did not perform any practical tests.

The Express people struggled hard to find a solution integrating both their relational database and their multidimensional database system. Clearly, neither one of them alone could meet the requirements. On the one hand, the multidimensional Express server is not able to handle the required data volumes. On the other hand, the relational engine does not support multidimensional analyses, a fact that results in sometimes mediocre performance for multidimensional operations. The Express people implemented only a small subset of the required query scenario based on only a small subset of the specified data volume in Express. Unfortunately, by the time of the evaluation, Oracle's Relational Access Manager had not yet been released. Mainly due to this fact, Oracle could not come up with an integrated solution.

The MetaCube people implemented the two alternatives mentioned above based on the relational database engine (1.5 million fact tuples). Simple queries with none or only a single attribute split performed very well. In the case of



a) Essbase approach *b) Express approach* *c) MetaCube approach*

Figure 2: Different modelling techniques used in the case study

more than one split, the solution with the "-1" level seemed very complicated and performed even worse. Thus, the second approach was implemented as a pure star-schema with the result that the queries were sped up by the factor 2-3. However, due to the limitation on a single product group, the scalability issue for queries spanning product groups with different feature schemas could not be addressed.

Within the SAS test, all required sample queries were implemented in the SAS query language. Fact data are partitioned by the DBA according to the application requirement and are kept in flat files. Master data are stored separately. The SAS test was performed on a data basis of 175 millions records (15.5 GByte) on a mid-range UNIX machine. The use of the very low-level programming interface of SAS enabled the usage of physical structure of the raw data. In summary, the SAS prototypical implementation resulted in a very high performance compared to database based systems (upto 9 Mbyte/s). Furthermore it could be seen that the query processing speed was limited only by the I/O-capacity of the hardware. The storage overhead in this solution tends to zero.

During the UDS project, two requirements namely the management of master data and the management of aggregations arose as the central issues. Since we believe that these points are important in general, we concentrate on each of these requirements and analyze them from the commercial products as well as from the scientific literature point of view in the remainder of this paper.

4. Management of Master Data

Qualifying information, also known as "Master Data" or "Meta Data" in this context, describes the world of interest, i.e. the skeleton for data analysis and is used to identify quantifying data. In our application scenario, two types of master data may be distinguished: First, *hierarchical classifications*, e.g. the grouping of single articles into product families and these in turn into product groups. Second, *feature descriptions*, which are used for describing single articles by their features. However, the description of the articles depends heavily on the classification structure. For example, washing machines are described by their water and energy usage whereas features like video or sound system are assigned the product group video recorders. Each class may hold 3 to 50 specific features with 2 to 100 values for each feature.

Correct master data is important from a modeling as well as an implementational point of view. For example, the decision of assigning articles of concurrent brands to the same product group or not would result in different answers of market leadership analysis and would have great implications on subsequent marketing strategies. From the implementational point of view, the classification hierarchy could be seen as a high-level access path and could be used for partitioning strategies and distribution within the storage hierarchy.

Global Access to Master Data

All statistic end-user tools participating in the analysis process must share the same consistent view to the master data base. Moreover, master data are also used directly by consultants in the sense of an expert knowledge base to support trend analysis which is not only based on numeric figures but is influenced by the consultant's expert knowledge. The problem of global master data access becomes even more important, when international analysis methods based on data bases spread all over the world are considered.

This implicates that the basis for a master data management system must be a highly available, distributed, and highly performant (and thus replicated) database system with transaction support to keep a consistent view. Furthermore, a standardized mechanism must exist to gain access to different end-user tools.

Schema Evolution Requirement

The great problem dealing with master data is the permanent change of that data. For example, with 250.000 products monitored, products appear to and disappear from the market on a daily basis. Nevertheless, the master data must be designed to enable a long term access to fact information for time-related, i.e. time sequence analysis. In general, the following types of changes can be observed in the master data:

- *the set of elements:*
  A new element, e.g. article is added to or removed from the dimension.
- *the set of attributes:*
  A new attribute is added or a attribute is deleted. For example, the product group computer gets a feature "Green Line" with the domain "Yes" and "No".
- *the set of classes within the classification hierarchy:*
  A new product group is introduced or a product group is deleted. In the first case, articles of another class are moved into the new one. In the second case, the remaining elements are possibly distributed to other classes.
- *the structure of the classification:*
  In this (very rare) case, a new classification level is introduced.

Changes on master data have also great impact on the fact information. For example, the correction of an incorrect assignment of an article to a product group makes it necessary to adopt the summary information (pre-aggregations), which is built upon the concerned groups (see section 5).

These points lead to the requirement of a convenient and sophisticated maintenance of master data, where the versioning of master data is allowed only at specific points of time. This means that updates become valid only step-wise, e.g. on a daily basis or on a analysis period basis.

The topic of versioning has a long tradition especially in the area of technical databases ([8], [28], [24]). The notion of validity however stems from the temporal database research area. A lot of different specific or extended relational data models were proposed in the literature. [27] provides a good overview of the different approaches. Furthermore, a lot of work has already been done in the temporal database area at the implementational level, e.g. in the area of specialized data access paths ([15]).

Despite of the large amount of related work, none of the observed commercial products support schema evolution or even a time-oriented data model. Time is seen as a normal dimension for evaluation. The special role of the time in the sense of schema versioning is not acknowledged so far.

Master Data Interchange Format

The second requirement wrt. the master data is the need for a standardized interchange format. This is especially important as the data warehouse may be seen as a central point efficiently delivering information, i.e. raw or summary data. The presentation of the data is done in separate (possibly home-made) applications. All these user-level applications need the same view on the data which implies that they all must have access to the master data management system. Furthermore, in our application, parts of the data warehouse may be sold to customers to explore the data with their own utilities. This implies that the corresponding master data must be delivered together with the raw data and understood by the customers' utilities.

This requires that master data must be exchangeable and the master data management system must be based on a standard keeping all the structural semantics within an export and import procedure. This problem is an open issue from the scientific literature as well as from the commercial products points of view. The Meta Data Coalition initiative was founded to address these interchange format issues ([17]). *"The Metadata Coalition regroups vendors and users allied with a common purpose of driving forward the definition, implementation and ongoing evolution of a metadata interchange format and its support mechanisms."* Unfortunately, the current MDIS 1.0-proposal seems not adequate to be used as a strategic platform.

5. Management of Aggregations

As outlined in the external requirements, the system must provide efficient online data analysis. Beside the modeling requirements, the system is required for an efficient navigation within the prescribed master data structure. Thus, classification-oriented operations like "drill-down"/ "roll-up" based on dimension elements must be supported. Since the analysts at the user level are primarily interested in features of articles or shops, dimensional attribute oriented operations must be supported by use of pre-aggregated data. To confirm this thesis, a former case study investigated and proved the performance gains of typical market retail research queries running against raw data and aggregations on different levels of detail ([16]).

Materialization of Aggregates

Essential for the achievable performance gain is the decision of the selection of aggregates to be materialized. The extreme answer to this question is to store all possible aggregation combinations. Once computed, this approach results in the best performance on the one hand but in a not acceptable storage overhead on the other hand. Therefore, several approaches are known to tackle the problem of selecting the "best" subset of all possible combinations. Moreover, as seen in the mentioned aggregation case study ([16]), it is often very desirable to materialize not the aggregation data for specific queries but to materialize the "finest common parent" for a set of queries. This pre-aggregation is therefore used by many queries and reduces the storage overhead wrt. a single query.

Based on the relational model, [13] proposes a greedy algorithm resulting in those aggregation combinations having the greatest benefit wrt. query execution times by a given storage boundary. This approach is extended in [9] considering existing secondary access paths to speed certain aggregation combinations. The approach of Gupta ([12]) proposes a polynomial-time heuristics based on the notion of AND-OR view graphs for the general NP-hard problem of selecting views for materialization. Compared to [13], this approach includes weighting factors, i.e. frequency counters, reflecting the current user behavior. From the commercial products point of view, only MetaCube provides a so called 'Aggregator Agent'. This tool simply monitors the user access and suggests aggregation combinations which are recommended to be materialized by the DBA. No commercial system has taken the ideas from the literature into account and provides an intelligent aggregation manager for implicitly creating and dropping aggregation combinations.

Usage of Materialized Aggregates

The idea of answering aggregation queries by reduction to summary data with equal or even finer granularity is as old as statistical data processing. First approaches stem from the area of census data processing in the early 1980's. [26] for example introduces the theory of derivability of summary data. The well known summary data model (as an extension of the relational model) can be seen as a milestone in reusing summary data. [2] handles the NP-hard problem of intersection of predicates by reducing the derivability to convex data partitions ("orthogonal categories") in the sense of multidimensional sub-cubes. An overview of newer work in the relational community can be found in [3]. Beside the technique of restructuring query plans containing aggregation operators to get cheaper execution plans ([29], [4]), the approach of Chaudhuri and Shim is extended in [5] to use materialized views implicitly. This idea is the main focus of the approach of generalized projections ([10]). At a semantic level, the work of [7] considers union-queries to combine the results of partitioned pre-aggregations.

Maintenance of Materialized Aggregates

Because new data is loaded into the data warehouse periodically, the materialized aggregations have to be maintained in order to keep them up to date. The aggregations must be maintained incrementally, because the huge data volume of the application forbids a complete re-materialization. The research has shown massive interest in this field lately. General incremental update strategies for materialized views are sketched in [11] or [22]. The important aggregation functions are considered in [21]. [18] discusses the maintenance of multiple views, whereas in [23], the specific aspects of bulk updates to the data warehouse are addressed.

The scientific results are not yet reflected in commercial products. To the best of our knowledge, only the MetaCube Aggregator supports incremental aggregation update to some degree. But even this product can only handle summation as aggregation function.

6. Summary and Conclusion

The results of the presented case studies can be summarized as follows: At the user requirement level, all systems allow the use of business terms and are capable of hiding relation names, access paths and so on. The MetaCube system is the only one which is capable of transparently using classification oriented pre-aggregations to speed up queries. In SAS and Express systems, materialized aggregations must be computed and used in the queries explicitly.

The two key components for developing a successful data warehouse in the application area of market research are master data management and aggregation management. The first one is neither solved in theory nor in practice. For application programmers it is hard to solve the problem, because many different commercial products are involved in keeping the data warehouse running, which all handle the master data differently. The need for an appropriate standard is obvious in this area. The aggregation management is a hot-topic in the research community with many promising approaches, but commercial implementations of these approaches are missing up to now. Thus, the complete management of aggregations, especially their maintenance is left to the application programmer.

With the background of market research requirements, we have shown that a data warehouse can not be bought off the shelf. Massive adaptions of the warehouse solutions vendors provided today have to be made. Thus, much functionality has to be programmed by the application designer. For this reason it is very important to choose the right partner with sufficient know-how and experience.

References

[1] N.N.: *Essbase Analysis Server - Bringing Dynamic Data Access and Analysis to Workgroups Across the Enterprise*, Product Information, Arbor Software Corporation, 1995

[2] Chen, M.C.; McNamee, L.P.; Melkanoff, M.: A Model of Summary Data and its Applications in Statistical Databases, in: *4SSDBM*, (Rome, Italy, June 21-23, 1988)

[3] Chaudhuri, S.; Shim, K.: An Overview of Cost-based Optimization of Queries with Aggregates, *Data Engineering Bulletin 18 (1995) 3*

[4] Chaudhuri, S.; Shim, K.: Including Group-By in Query Optimization, in: *VLDB'94*, (Santiago de Chile, Chile, Sept. 12-15, 1994)

[5] Chaudhuri, S.; Shim, K.: Optimizing Queries with Aggregate Views, in: *EDBT'96*, (Avignon, France, March 25-29), 1996

[6] Codd, E.F.: Codd, S.B.; Salley, C.T.: *Providing OLAP (On-line Analytical Processing) to User Analysts: An IT Mandate*, White Paper, Arbor Software Corporation, 1993

[7] Dar, S.; Jagadish, H.V.; Levy, A.Y.; Srivastava, D.: Answering Queries with Aggregates Using Views, in: *VLDB'96*, (Bombay, India, Sept. 3-6, 1996)

[8] Dittrich, K.R.; Lorie, R.A.: Version Support for Engineering Database Systems, *IEEE Transactions on Software Engineering*, 14(1988)4

[9] Gupta, H.; Harinarayan, V.; Rajaraman, A.; Ullman, J.D.: Index Selection for OLAP, in: *ICDE'97*, (Birmingham, UK, April, 7-11, 1997)

[10] Gupta, A.; Harinarayan, V.; Quass, D.: Generalized Projections: A Powerful Approach to Aggregation, in: *VLDB'95*, (Zurich, Switzerland, Sept. 11-15, 1995)

[11] Gupta, A.; Mumick, I.: Maintenance of Materialized Views: Problems, Techniques, and Applications, in: *IEEE Data Engineering Bulletin, Special Issue on Materialized Views & Data Warehousing* 18(1995)2

[12] Gupta, H.: Selection of Views to Materialize in a Data Warehouse, in: *ICDT'97*, Delphi, Greece, Jan 8-10, 1997)

[13] Harinarayan, V.; Rajaraman, A.; Ullman, J.D.: Implementing Data Cubes Efficiently, in: *SIGMOD'96*, (Montreal, Quebec, Canada, June 4-6, 1996)

[14] N.N.: *The INFORMIX-MetaCube Approach*, Product Information, Informix Software, Inc., 1996

[15] Kolovson, C.P.: Indexing Techniques for Historical Databases, in: *[27]*

[16] Lehner, W.; Ruf, T.; Teschke, M.: Improving Query Response Time in Scientific Databases Using Data Aggregation, in: *DEXA'96*, (Zurich, Switzerland, Sept. 9-13, 1996)

[17] Meta Data Coalition: *http://www.metadata.org/index.html*

[18] Mumick, I.; Quass, D.; Mumick, B.: Maintenance of Data Cubes and Summary Tables in a Warehouse, in: *SIGMOD'97*, (Tuscon, AZ, May 12-15, 1997)

[19] N.N.: *Personal Express User's Guide 5.0*, Oracle Cooperation, 1996

[20] N.N.: *Relational Access Manager, Reference Manual*, Oracle Cooperation, 1997

[21] Quass, D.: Maintenance Expressions for Views with Aggregation, in: *ACM Workshop on Materialized Views: Techniques and Applications* (Montreal, Canada, June 7, 1996)

[22] Quass, D.; Gupta A.; Mumick, I.; Widom, J.: Making Views Self-Maintainable for Data Warehousing, in: *PDIS'96*, (Miami Beach, FL, Dec. 18-20, 1996)

[23] Quass, D; Widom J.: On-Line Warehouse View Maintenance for Batch Updates, in: *SIGMOD'97*, (Tuscon, AZ, May 15-20, 1997)

[24] Revesz, P.Z.: On the Semantics of Theory Change: Arbitration between Old and New Information, in: PODS'93, (Washington, D.C, May 25-28, 1993)

[25] N.N.: *Building a SAS Data Warehouse*, Product Information, SAS Institute, 1995

[26] Sato, H.: Handling Summary Information in a Database: Derivability, in: *SIGMOD'81*, (Ann Arbor, Michigan, April 29 - May 1), 1981

[27] Tansel, A.U.; Clifford, J.; Gadia, S.; Jajodia, S.; Segev, A.; Snodgrass, R.: *Temporal Databases*, Redwood City e.a.: Benjamin/Cummings, 1993

[28] Wedekind, H.: Are the Terms "Version" and "Variant" Orthogonal to One Another? A Critical Assessment of the STEP Standardization, *SIGMOD Record* 23(1994)4

[29] Yan, W.P.; Larson, P-A.: Eager Aggregation and Lazy Aggregation, in: *VLDB'95*, (Zurich, Switzerland, Sept. 11-15), 1995