Wolfgang Lehner, Thomas Ruf Michael Teschke

CROSS-DB: a feature-extended multidimensional data model for statistical and scientific databases

Erstveröffentlichung in / First published in:

CIKM96: Conference on Information and Knowledge Management, Rockville 12.-16.11.1996. ACM Digital Library, S. 253–260. ISBN 978-0-89791-873-2.

DOI: https://doi.org/10.1145/238355.238547

Final edited form was published in "CIKM96: Conference on Information and Knowledge Management. Rockville 1996", S. 253–260, ISBN 978-0-89791-873-2
https://doi.org/10.1145/238355.238547

CROSS-DB
A FEATURE-EXTENDED MULTIDIMENSIONAL DATA MODEL FOR STATISTICAL AND SCIENTIFIC DATABASES

Wolfgang Lehner
University of Erlangen-Nuremberg
Germany
lehner@informatik.uni-erlangen.de

Thomas Ruf
GfK Marketing Services Europe
Germany
thomas.ruf@gfk.de

Michael Teschke
University of Erlangen-Nuremberg
Germany
teschke@informatik.uni-erlangen.de

Abstract

Statistical and scientific computing applications exhibit characteristics that are fundamentally different from classical database system application domains. The CROSS-DB data model presented in this paper is optimized for use in such applications by providing advanced data modelling methods and application-oriented query facilities, thus providing a framework for optimized data management procedures. CROSS-DB (which stands for Classification-oriented, Redundancy-based Optimization of Statistical and Scientific DataBases) is based on a multidimensional data view. The model differs from other approaches by offering two complementary mechanisms for structuring qualifying information, classification and feature description. Using these mechanisms results in a normalized, low-dimensional database schema which ensures both, modelling uniqueness and understandability while providing enhanced modelling flexibility.

1 Introduction

Statistical and scientific computing application domains like earth sciences, biomedicine and socio-economics require database support that fundamentally differs from classical database applications like banking and administration on the logical data modelling level as well as on the physical data management side ([Shos82]). Various proposals have been made to overcome the shortcomings of traditional database systems to improve their support for the statistical and scientific computing domain (Section 6).

The applications we are aiming at with our approach, reflect a typical "Online Analytical Processing" scenario with mainly read-only access performing data aggregating operators on ultra large databases ([LeRT95a]). Figure 1 depictes a sample market research table, describing sales figures of electronic equipment in different countries. The table displays the data at different levels of detail. Video-equipment for example is split up into their subsumed classes *Camcorder* and *HomeVCR*, represented as shaded heading cells, as well as partitioned according to the applicable characteristics like *VideoSystem*. The nested table structure may be regarded as a flattened multi-dimensional representation issuing two major tasks from a database research point of view. First, the structure of nested table headings with their nested attributes has to be modelled in an appropriate way. Second, the data fields of the table have to be efficiently calculated.

Obviously, the table depicted in Figure 1 reflects only one of many possible views on the data material. The CROSS-DB data model allows to describe generic schemata of applications on a conceptual level, thus providing the basis for a flexible and highly efficient generation of arbitrary table instances on an external level, as will be shown throughout the remainder of the paper.

In the next section, the general architecture of the CROSS-DB model is sketched from a design and usage point of view according to the ANSI/SPARC three schema database architecture. Section 3 covers the modelling techniques of hierarchical and feature extended qualifing information whereas section 4 explains the user access and external view on the data. Section 5 sketches the characteristics of the internal level. Section 6 discusses related work and shows that none of the statistical data models described so far in the literature allows for the fine-grained, feature-extended modelling capabilities of CROSS-DB along with a natural and classification driven data aggregation concept. Section 7 summarizes the approach and gives an outlook on future work.

2 Three Level CROSS-DB Model

The CROSS-DB model strictly follows the ANSI/SPARC three schema database architecture ([TsKl78]) in external level, conceptual level and internal level, thus preserving logical data independence between external and conceptual level as well as physical data independence between conceptual and internal level. In the following sub-sections, the different schema levels will be described briefly.

DOI: https://doi.org/10.1145/238355.238547

Final edited form was published in "CIKM96: Conference on Information and Knowledge Management. Rockville 1996", S. 253–260, ISBN 978-0-89791-873-2
https://doi.org/10.1145/238355.238547

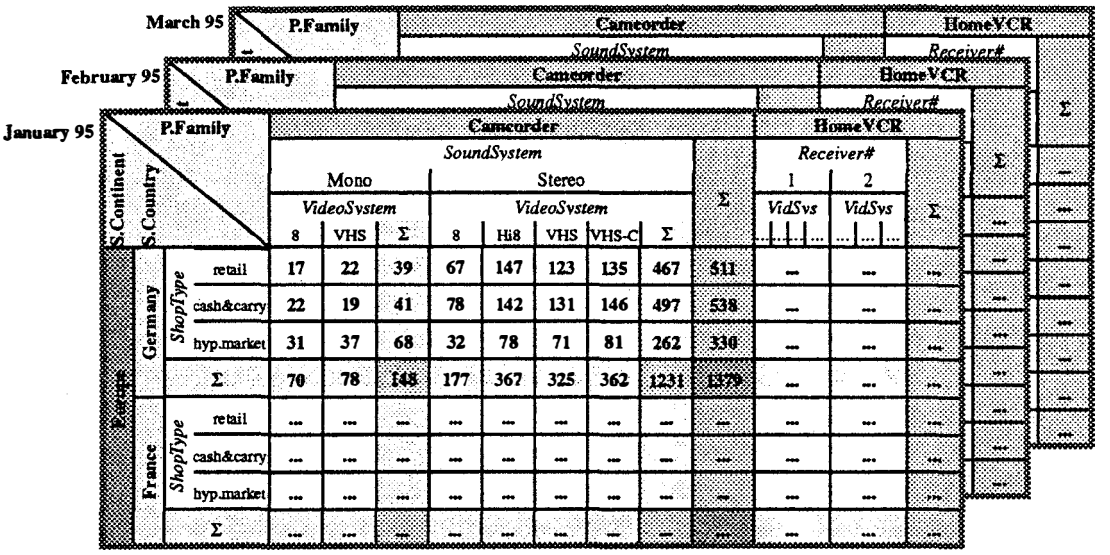

Figure 1: A Sample Market Research Table

2.1 Conceptual Level

In a multi-dimensional data model, the distinction of qualifying and quantifying data is fundamental. Qualifying data, also known as "category attributes" ([Shos82]), structurally describe the application domain and are used for accessing quantifying data items during an analysis process. For example, sales figures may be characterized by a product, shop and time dimension. Typically, qualifying data for single dimensions can be arranged in a classification hierarchy. For example, products may be classified into product groups and these, in turn, into product areas. The modelling of qualifying data, i.e. the construction of classification hierarchies and the feature mechanism are detailed in the next section.

Quantifying data, also known as "summary attributes", are typically numerical values which are the object of the analyzing process in a multi-dimensional context (e.g. daily sales figures for single product sales in individual shops). To reflect the multi-dimensionality of the CROSS-DB data model, a single quantifying value is called a cell, referencable in an n-dimensional space with a fixed granularity. Potentially, there may exist as many data values in the database as there are cells in the multi-dimensional array, but usually, some cells are structurally empty (e.g. due to the fact that not every product is sold in every shop).

2.2 External Level

On the external level, the different dimensions which are modelled and instantiated independently from one another on the conceptual level are combined into an n-dimensional context that reflects the user's multidimensional view. Operators on data can be visualized as transformations of cells within the granularity context stretched by different dimensions. Every cell within a data cube, containing one or more atomic values, may at last be split up according to feature combinations defined upon the cell (e.g. sales by *Videosystem* and *ShopType*, Figure 2). The user's view and data access is detailed in Section 5.

2.3 Internal Level

The main objective on the internal level in the CROSS-DB approach is to handle quantifying data by providing an intelligent storage and query optimization system for multidimensional data. The storage management system clusters raw data as well as materialized aggregation data according to an application-defined schema and provides a "data retirement" strategy to automatically off-load expensive storage devices by transferring data to cheaper devices. Section 5 sketches these tasks of the internal level.

3 Qualifying Data at the Conceptual Level

Even if scientific data usually have to be displayed in a two-dimensional way, a multi-dimensional data model is the natural choice for the statistical and scientific application domain from a conceptual point of view ([Shos82]). Qualifying data may be considered as "master data" which describe the part of the real world known by the database system in terms of a conceptual schema. The CROSS-DB model distinguishes between object and meta level as well as schema and instance level ([Ortn95]). In this section, hierarchical classification and the notion of features on schema level as well as the complex process of classification schema instantiation and the notion of categorizations on meta level are described.

3.1 Classification Schemata

Compared to the other approaches towards statistical and scientific databases (SSDBs) mentioned in Section 6, one of the main achievements of the CROSS-DB model is the pos-

Final edited form was published in "CIKM96: Conference on Information and Knowledge Management. Rockville 1996", S. 253–260, ISBN 978-0-89791-873-2
https://doi.org/10.1145/238355.238547

sibility to specify detailed classifications of qualifying data in a very flexible way. These classifications enable a user-friendly addressing scheme and, at the same time, provide a basis for advanced query optimization. The desired hierarchy of terms on schema level is described in an object-oriented modelling approach which reflects the repeated generalization steps performed during the constitution of the hierarchy of concepts.

DEFINITION: A *classification schema* S is an inheritance hierarchy where the root of the structure is the generic superclass "*" and every node, except the root node, is a subclass of its predecessor node.

Figure 2 shows a possible classification schema for product information. For example, *video* is a superclass of *camcorders* and *homeVCR*'s; in the opposite direction, the class *video* is a subclass of the class *electronic equipment.*

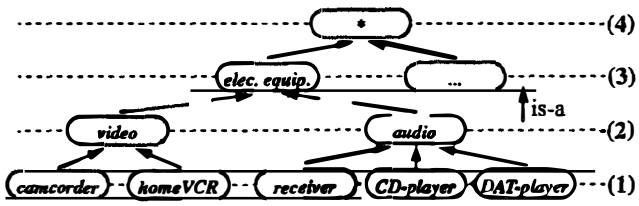

Figure 2: Sample Product Classification Schema

DEFINITION: A classification schema is called *well-formed* if it is a partitioning (i.e. non-overlapping) and balanced (i.e. the path length from the root to any leaf is the same for all leaves) classification tree.

As classification schemata are established in a normative way, the constraints for well-formedness have no severe impact in real-world applications, because they can be fulfilled in a constructive manner by decomposing overlapping classes into separate subclasses and by introducing "placeholder" classes in a path, if necessary. Besides providing high-level terms for user data access specification, well-formed classification schemata have the great advantage that they may be used for storage organization as well (Section 5). The fact that the classes describe a full, intersection-free partitioning of the corresponding data elements at each level in the classification hierarchy also makes it possible to systematically use precomputed and materialzed aggregation values for the computation of queries. Section 5 will discuss this issue.

DEFINITION: The inheritance level in a classification schema is called its *granularity*. The generic superclass "*" holds the coarsest level of granularity N, all leaf nodes of the classification schema have the finest level of granularity 1.

Referring again to Figure 2, the schema nodes of *camcorders* and *homeVCR*'s possess a level of granularity of (1), and the layer of *video* and *audio* equipment is assigned the level of granularity (2). In general, every generalization process increments the granularity value.

3.2 Feature Descriptions

The normative construction of a schema hierarchy arranges the conceptual world by the intensional concept of generalization ([SmSm77]), which does not necessarily manifest in the extensional world. In other words, the only way of determining the assigned class of a given entity is to refer to the specification of the classification itself. There is, however, a second, extensional way of grouping things into a classification hierarchy: feature assignment. For example, a feature *VideoSystem* may be used to assign one of the values '8', 'Hi8', 'VHS' or 'VHS-C' to an entity belonging to the class *camcorder*. Then, entities can be grouped by features values, as can be seen in Figure 3. Formally, the enrichment of the specification of a class in a classification schema by a detailing feature description is defined as follows:

DEFINITION: A *feature* at the schema level consists of an identifier and a discrete domain $D = \{v_1, ..., v_n\}$ of possible values v_i.

In our example, the *VideoSystem* feature of the classification node *camcorder* would be modelled as (*VidSys*, {'8', 'Hi8', 'VHS', 'VHS-C'}). The requirement of discrete domains does not impose a restriction for the modelling process: in practice, domains which are used for feature descriptions are made discrete by grouping values into classes for reporting purposes anyway (e.g. weight figures are grouped into classes 'light', 'medium', and 'heavy').

When generating a classification schema, an arbitrary number of features may be assigned to the classes of the hierarchy. The feature schema is defined on class level, whereas the specific values are assigned to single instances of the classes. One advantage of the object-oriented modelling approach adopted on schema level is the mechanism of feature inheritance. Every subclass inherits all the features of the superclass; however, the feature domain of the subclass may be only a subset of the corresponding feature domain of the superclass. This "top-down" approach of inheriting possible feature values along the classification hierarchy is depicted in Figure 3. The class *camcorder* inherits all the features from the superclass *video* with the restriction that there are no camcorders in the real world with a video system "Beta". Even the set of possible brands is restricted, because in the modelled world, not every video equipment producer sells camcorders. On the other hand, new features may be on the lower levels of the classification hierarchy which are only valid for subsets of the classes (e.g. only *camcorders* have a feature *BatteryLifeTime*, but not *homeVCRs*).

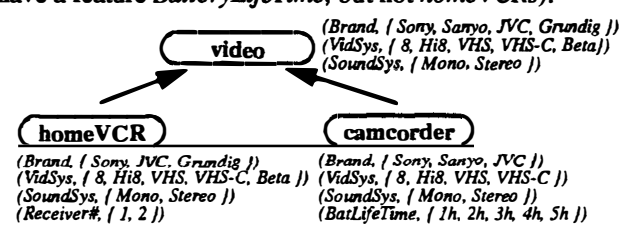

Figure 3: Feature Inheritance

Final edited form was published in "CIKM96: Conference on Information and Knowledge Management. Rockville 1996", S. 253–260, ISBN 978-0-89791-873-2
https://doi.org/10.1145/238355.238547

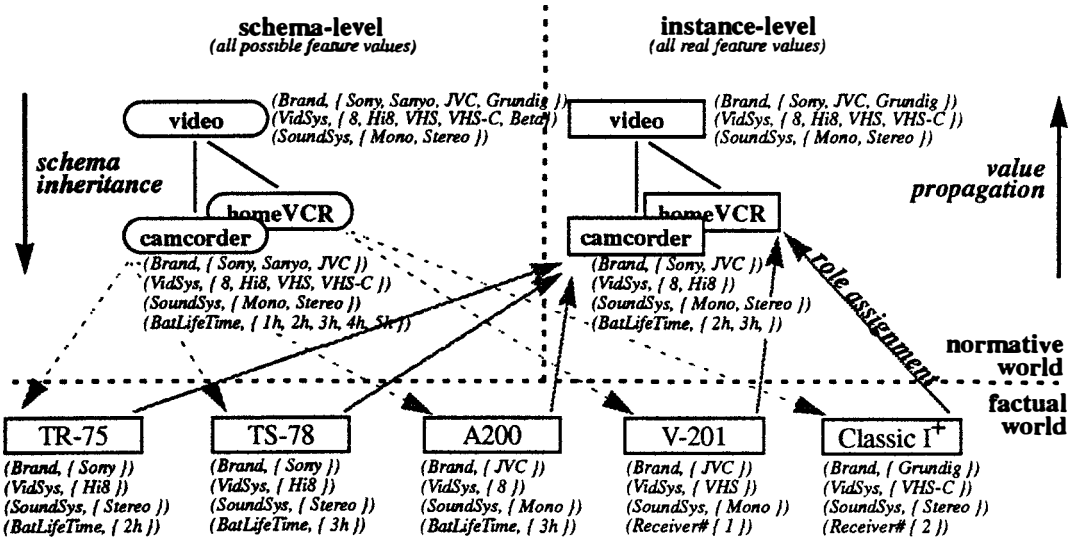

Figure 4: Feature Values Propagation

The feature inheritance mechanism is of great importance for data production as well as for data evaluation. In the production phase, feature schemata help to guide the classification of incoming values for items who's identification is unclear. During the data analysis process, it may be decided from the master data alone whether a query may generate results by simply checking the feature domains; in the example shown in Figure 3, the answer that the total of sales of camcorders with video system "Beta" is zero can be given without accessing any raw data.

3.3 Instantiation of Classification Schemata

So far, classifications and feature assignments based upon them were introduced only at schema level. Next, we describe the two-step process to instantiate a classification schema. In the first step, all factual instances of a dimension must be assigned to an appropriate and feature-compatible class that can serve as their schema. The second step consists of creating exactly one instance for every class in the classification hierarchy, including the "bottom-up" instantiation of the corresponding features.

DEFINITION: A *factual instance* consists of a name and a set of features. At any point in time, a feature may hold at most one (empirically determinable) value.

For example, five factual instances representing some single products are illustrated in Figure 4 by rectangles in the lower half of the picture. Note that the values at instance level have to be selected from the list of possible values determined by the parent node in the classification hierarchy. If the value is not found in this list, the value may either be incorrect or belong to an item which has not been introduced in the dimension yet. In the latter case, the list of possible feature values at the parent node has to be extended before the new item can be introduced in the database. As mentioned before, this mechanism also helps to guide the classification of new items into the hierarchy as well as to check the validity of a query at schema level.

Figure 4 shows how the connection between the world of factual instances and the world of normative classifications is established in the CROSS-DB model. By assigning a factual instance to a class in the classification hierarchy, it assumes a *role* in which it may be used on the external level of the database system. When assuming a role, the features of the factual instances are mapped to the feature list of the corresponding class. Next, the feature values of the factual instances are propagated to the class instances. Thus, the feature descriptions of the class instances reflect the current values of their child nodes. This may significantly reduce the search process when evaluating a feature-oriented query; in many cases, it may be determined on a high classification level that a whole subtree doesn't have to be included in the search, because the value of interest is not present in any of the successor nodes. For example, a query for the sales of "Grundig" video equipment may skip the camcorder instances completely, because from the camcorder class at instance level, it is clear that there are no "Grundig" camcorders modelled in the database.

Now we are able to specify precisely what a classification and a corresponding categorization is in the CROSS-DB model:

DEFINITION: A *classification* is the result of the instantiation of a classification scheme, comprising the steps of:

- assigning factual instances to leaf classes of the classification schema describing the finest level of granularity 0;
- instantiating the classes in the classification schema exactly once;

Final edited form was published in "CIKM96: Conference on Information and Knowledge Management. Rockville 1996", S. 253–260, ISBN 978-0-89791-873-2
https://doi.org/10.1145/238355.238547

- instantiating the features of the normative instances by all values of the features of the subsumed instances;

Classifications with the corresponding categorizations at the meta data level provide the fundamental bridge from the conceptual world of factual instances to the user data access level on the external level of our architecture.

DEFINITION: A list of application defined identifiers for granularity level specifications ordered from the factual instances up to the coarsest level of granularity of the generic normative instance "*" forms the *categorization* of the corresponding classification.

In opposite to the classification schema and an instantiated classification, the notion categorization reflects meta data information, namely user defineable identifiers for granularity levels. Referring to the ongoing example, Figure 5 shows a part of the classification at object level with the corresponding categorization.

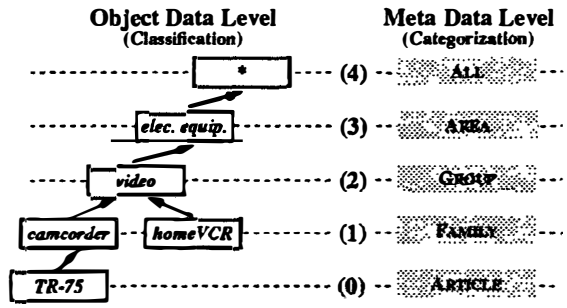

Figure 5: Object vs. Meta Data Level

3.4 Dimensions of the CROSS-DB data model

With the notions of factual instance and classification schema in mind, the notion of a dimension within the CROSS-DB data model is defined as follows:

DEFINITION: A *dimension* consists of a globally unique identifier, a set of classifications each with a corresponding categorization and a set of factual instances.

The set of all factual instances of the universe is partitioned into sets each belonging to another dimension named by the unique identifier. Unlike other approaches, each set builds the basis for multiple classification trees with each obtaining another feature inheritance schema. Furthermore, each dimension is handled independendly at the conceptual level to preserve logical data independence ([LeRT95b]).

4 User Data Access at the External Level

When accessing quantifying data at the external level, dimensions have to be brought into a multi-dimensional context. This section explains the process of providing multidimensional access to the database and presenting the results of a query according to some user-defined split criteria. The examples will be given in Cube-Query-Language CQL, a multidimensional query language derived from SQL. Due to space limitations, the syntax and semantics of Cube-Query-Language will not be addressed in this paper. For a detailed discussion including a comparison of CQL and the "Data Cube" approach ([GBLP96]), the reader is encouraged to refer to [BaLe96]. The "Data Cube"approach maps a relational table to a multidimensional cube by viewing primary key attributes as dimensions and numeric non-key attributes as quantifying data cells. It further extends SQL by a cube clause expressing that aggregation operators have to work on all possible combinations of independantly grouped "dimensions". To flaten a multidimensional data cube resulting from an aggregation operation, the approach introduces an artifical qualifying "ALL"-value, representing an aggregation wrt. the corresponding attribute.

As depicted in Figure 6, the external level of the CROSS-DB model is subdivided into three layers. The lower layer spans the granularity context for a query by building the cross-product of the categorizations of all participating dimensions. The middle layer provides operators for transforming data cubes. On the upper layer, the query's results are presented by using the actual feature values of the participating classifications nodes. The different layers are explained in more detail in the following subsections.

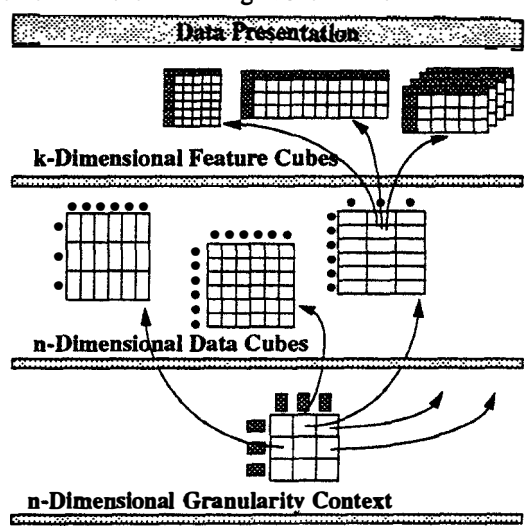

Figure 6: Data Access and Presentation Layers

4.1 Constructing the Query Context

The first step to provide a multidimensional view to the user is to combine dimensions, i.e. sets of factual instances with corresponding categorizations, into a view which is appropriate to the user's demands. The categorizations, depicted by a list of shaded rectangles in the lower level of Figure 6, constitute the so-called granularity space of a query.

DEFINITION: The n-dimensional *granularity space G* of a query is the cross-product of all used categorizations with the granularity 0 up to N^i for the i-th categorization:

$$G = (0,...,N^1) \otimes (0,...,N^2) \otimes ... \otimes (0,...,N^n)$$

Final edited form was published in "CIKM96: Conference on Information and Knowledge Management. Rockville 1996", S. 253–260, ISBN 978-0-89791-873-2
https://doi.org/10.1145/238355.238547

The spanning of the granularity space is described in CQL in the FROM-clause of the SELECT-statement. For example, line 2 of the statement below fixes the granularity space of the query to the product, shop and the timeline categorization which groups days by months.

```
(1) select SALES
(2) from Product P, TimeByMonth T, Region R
(3) where P.Group = 'video',
    T.Year = '1996'
```

To select a part of a data cube from within the multidimensional granularity space, two different mechanisms are provided. Usually, selections are based on the dimensional classification hierarchies, specified in the WHERE-clause of the SELECT-statement (line 3). The more general and complex addressing scheme used extensively in query optimization is introduced in [LeRT 95b]. Additionally, predicates over feature values (e.g. `with P->VidSys != 'Beta'`) and quantifying data (e.g. `restrict SALES between (5,100)`) are supported.

4.2 Data Cube Operators

Every point in the granularity space of the query context represents an n-dimensional data cube which holds quantifying data of the corresponding granularity. As illustrated in Figure 6 by arrows from the lower to the middle level, a granularity space builds the frame that contains all data cubes for quantifying data that can be referenced or generated in a query. In a specific query, one of these n-dimensional data cubes has to be selected, which may then be transformed according to the queries' demands. For data cube transformations, the CROSS-DB model supports two classes of operators: cell-oriented and aggregating operators. Cell-oriented operators are used to combine quantifying data from one or more cells into a new value, whereas aggregating operators are used to introduce some higher-level data summaries under user control.

With cell-oriented operators, input data for an operator are mapped to a cube of equal or even finer granularity. From a reporting perspective, however, the main objective in analyzing data is to "compress" the detailed quantifying raw data into some few summary values. Aggregating operators allow for a user-oriented specification of such summary values. The following CQL-statement shows the use of the SUM-operator:

```
(1) select SUM(SALES)
(2) from Product P, TimeByMonth T, Region R
(3) where P.Group = 'video',
    T.Year = '1996', R.Continent = 'europe'
(4) upto P.Familiy, T.Month, R.Country
```

The sample query generates monthly sales figures for the different product families and countries across *europe* for the product group *video* in the year *1996*. As can be seen from the example, aggregating operators need an explicit specification of the target granularity in the UPTO-clause (line 4).

4.3 Presentation of Query Results

Up to now, features were introduced only at conceptual level. Now we will detail what happens when features are used in a multidimensional context. As was seen in the last sub-section, new data may be generated in the data cubes by using aggregation operators. The results of these operators can be split according to the features of all instances referencing the cell when the data is to be presented to the user (upper layer of Figure 6). The set of features of an n-dimensional cell consists of the union of all the features of the n instances addressing the cell. Thus, simultaneous feature splits in arbitrary dimensions are possible.

In CQL, the feature split is specified in the BY-clause (line 5 in the example below). Note that the data value is split only at the users level in the feature cube; on the conceptual level, only cells from the data cubes are known, as the corresponding classification hierarchies are defined on the conceptual level, whereas feature-oriented splits of values are only specified in the user's query. The powerful feature presentation mechanism is summarized in the following CQL query which generates the table presented in Figure 1.

```
(1) select SUM(SALES)
(2) from Product P, TimeByMonth T, Region R
(3) where P.Group = 'video',
    T.Year ='1996', R.Continent = 'europe'
(4) upto P.Familiy, T.Month, R.Country
(5) by P->VidSys, P->SoundSys, R->ShopType
```

5 The Package Storage System at the Internal Level

SSDBs in general handle huge amounts of quantifying data. The CROSS-DB storage system provides a configurable and cost-based logical storage interface for the query optimizer. Packages reflect single storage units and contain application-oriented clusters of data items and a label describing the content of the package. In analogy to conventional access paths, the physical clustering of the multidimensional data implies the explicit definition of primary and secondary classifications. For example, raw data values for sales of products clustered by different product groups may be put into a package and described by a label (left side of Figure 7) specifying the geographical area and the time frame which is covered by the individual values in the package.

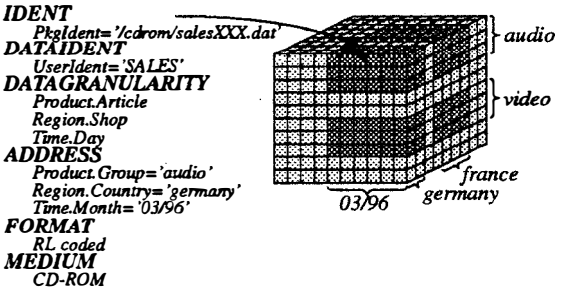

Figure 7: Data Packages and their Meta Data

Final edited form was published in "CIKM96: Conference on Information and Knowledge Management. Rockville 1996", S. 253–260, ISBN 978-0-89791-873-2
https://doi.org/10.1145/238355.238547

The contents of the packages are handled as BLOB's without any internal structure. This enables the storage subsystem to distribute different package contents over disk drives like a global striping algorithm, for example. In the same manner, contents may be spread over different physical devices within the storage hierarchy. This enables configurable "data migration" strategies, to automatically off-load expensive storage devices by transferring data to cheaper devices, e.g. after a predefined period of time has elapsed. The package labels are maintained in a main-memory label description table which includes location- and media-specific access cost information as an input of the query optimizer. The query optimizer uses the granularity context to find packages within each addressed classification where the granularity is as high as possible (wrt. the target granularity) and the access cost is as low as possible. A formal description of the query optimization algorithm can be found in [LeRu96].

6 Related Work

The roots of research in statistical and scientific databases (SSDB) may be dated back to the beginning of the 80's. From an overall perspective and without claiming to be complete, the proposals that have been made for the data modelling side may be grouped into three major streams (Figure 7). On the left side of the figure, work geared at representing given statistical tables on a conceptual level is represented. Both SUBJECT ([ChSh81]) and its direct successor, GRASS ([RaRi87]), use a graphical notation to describe the structure of statistical tables in terms of cross products of dimensions, each of which can be further classified in a hierarchy of terms. GRASS extends SUBJECT by introducing new node types besides the cluster and cross product nodes provided in SUBJECT, allowing for a better conceptualization of derived summary data calculated from the underlying table structure. Both approaches model specific table instances only. STORM ([RaSh90]) extends GRASS by explicitly distinguishing an extensional and one or more intensional data modelling levels. This results in a much greater independence of the model from the initial table structure. In particular, the ambiguity of category schema and instance (e.g. *camcorder* may be regarded as schema for different products, but also as an instance of the category *video*) can be resolved. All the models in this line are suited to serve as a basis for implementing a statistical and scientific database management system (SSDBMS) in principal, although little attention has been given to the performance of the resulting systems.

A conceptualization of the underlying data structure has been the main focus of the approaches depicted in the middle of Figure 7 from the very beginning on. SAM* ([Su83]), SDM4S ([Sato88]) and CSM ([BaBa88]) all provide a rich set of constructs for modelling the complex semantic relations between entities in a statistical or scientific database. Although at least SDM4S was exploited as a modelling basis for an actual SSDB used by the National Land Agency in

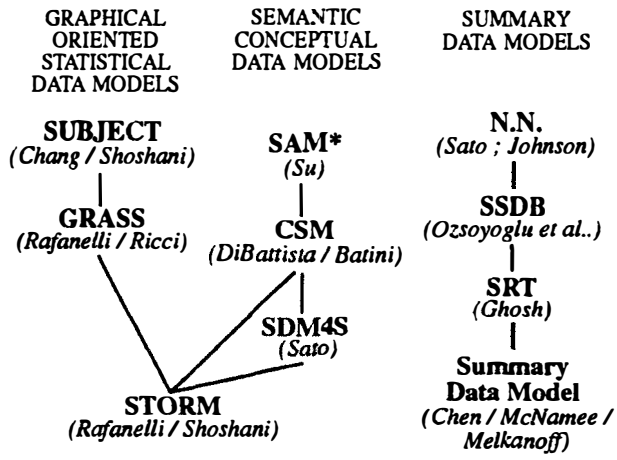

Figure 8: Streams of SSDB Research

Japan, the focus is clearly on the data modelling side and on the user interface; little attention is given to the issues on the physical database level. CSM, for example, explicitly models raw data and derived data in two separate data models, which makes it hard to support optimization of "drill down" queries typically found in SSDB applications, in which an analysis session starts from highly aggregated values, which are step-wise refined down to the raw data level.

The third stream of SSDB research, depicted at the right hand side of Figure 7, emphasizes the use of materialized summary data to speed up query execution times in an SSDBMS. Early beginnings were made independently by Sato ([Sato81]) and Johnson ([John81]), followed by the SSDB model of Z.M. Ozsoyoglu and G. Ozsoyoglu ([OzOz84]) and Ghosh´s extensions of the relational database model to capture the notion of statistical summaries ([Ghos86]). Both approaches are based on an extension of relational algebra for manipulating aggregated summaries in the database. The Summary Data Model introduced by Chen, McNamee and Melkanoff ([ChMM88]) not only describes how to derive new aggregations from existing ones, but also provides a specific indexing method, the Logical Summary Data tree, down to the implementation level. Here, the internal level of the database system is much more emphasized as in the approaches described above, but the modelling concepts like classification hierarchies are much more restricted than in those.

Altogether, although some facets of the modelling and optimization techniques found in the CROSS-DB approach may be found in various other approaches as well, none of these integrates all of them in a comprehensive way.

7 Summary and Future Work

In this paper, we presented the CROSS-DB data and access model which equally concentrates on the logical modelling side and the physical implementation issues for an SSDBMS within a common framework. On the logical side, concepts like independent dimensions with classification hierarchies

defined upon them seem to be a necessity, particularly in order to ease the task of query formulation at the user's level. Also, the conceptualization of the application domain is independent from a particular data or table structure. Additionally, it is possible in CROSS-DB to provide different views of the data at the user interface, like in traditional database systems. On the physical database side, the details of how raw and summarized data are held and managed in the storage systems are hidden from the user by the abstract "Package Storage System".

Currently, we are working on implementing our conceptual model for qualifying information in an object-oriented database system which will then serve as a common reference for all other system components. On the user interface side, we have implemented an CQL-parser and are currently designing a graphical table layout module. On the internal level, we are currently working on an intelligent storage manager and the memory mapping algorithms for storing and retrieving sparse multidimensional data.

References

BaBa88 Di Battista, G.; Batini, C.: Design of Statistical Databases: A Methodology for the Conceptual Step, *Information Systems 13(1988)4*, pp. 423-428

BaLe96 Bauer, A; Lehner, W.: CQL - A Powerful Query Language for Flexible Analysis in Scientific and Statistical Database Systems, *Technical Report, Department of Computer Science VI, Univ. of Erlangen-Nuremberg, 1996, submitted for publication*

ChSh81 Chan, P.; Shoshani, A.: SUBJECT: A Directory Driven System for Organizing and Accessing Large Statistical Data Bases, in: *Proceedings of the 7th International Conference on Very Large Data Bases (VLDB'81, Cannes, France, September 9-11)*, 1981, pp. 553-563

ChMM88 Chen, M. C.; McNamee, L.; Melkanoff, M.: A Model of Summary Data and its Applications in Statistical Databases, in: Rafanelli, M.; Klensin, J. C.; Svensson, P. (eds.): *Proceedings of the 4th International Working Conference on Statistical and Scientific Database Management (4SSDBM, Rome, Italy, June 21-23)*, Lecture Notes in Computer Science 339, Berlin e. a.: Springer-Verlag, 1988, pp. 356-372

GBLP96 Gray, J.; Bosworth A.; Layman A.; Pirahesh, H.: Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Total, in: *Proceedings of the 12th IEEE International Conference on Data Engineering* (DE´96, New Orleans, Louisiana, Feb. 26 -Mar. 1), 1996, pp. 152-159

Ghos86 Ghosh, S.P.: Statistical Relational Tables for Statistical Database Management, *IEEE Transactions on Software Engineering 12(1986)12*, pp. 1106-1116

John81 Johnson, R.: Modelling Summary Data, in: *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'81, Ann Arbor, Michigan, April 29-May 1)*, 1981, pp. 93-97

LeRT95a Lehner, W.; Ruf, T.; Teschke, M.: Data Management in Scientific Computing: A study in Market Research, in: *Proceedings of the International Conference on Applications of Databases (ADB'95, Santa Clara, California, December 13-15)*, 1995, pp. 31-35

LeRT95b Lehner, W.; Ruf, T.; Teschke, M.: Optimizing Database Access Performance in Scientific Applications without Compromizing Logical Data Independence, in: *Proceedings of the International Conference on Applications of Databases (ADB'95, Santa Clara, California, December 13-15)*, 1995, pp. 120-135

LeRu96 Lehner, W.; Ruf, T.: An Redundancy-Based Optimization Approach for Aggregation Queries in Scientific and Statistical Databases, *Technical Report, Department of Computer Science VI, Univ. of Erlangen-Nuremberg, 1996, submitted for publication*

Ortn95 Ortner, E.: Elemente einer methodenneutralen Konstruktionssprache für Informationssysteme, in: *Informatik Forschung und Entwicklung (1995)10*, pp. 148-160 (in German)

OzOz84 Ozsoyoglu, Z.M.; Ozsoyoglu, G.: SSDB: An Architecture for Statistical Databases, in: *Proceedings of the 4th International Jerusalem Conference on Information Technology* (IJCIT´84, Jerusalem, Israel), 1984, pp. 327 341

RaRi83 Rafanelli, M.; Ricci, F.: Proposal of a Logical Model for Statistical Databases, in: *Proceedings of the Second International Workshop on Statistical Database Management*, Los Altos, CA, 1983

RaRi87 Rafanelli, M.; Ricci, F.: A Graphical Approach for Statistical Summaries: The GRASS Model, in: *Proceedings of the ISMM International Symposium on Microcomputers and their Applications*, 1987

RaSh90 Rafanelli, M.; Shoshani, A.: STORM: A Statistical Object Representation Model, in: Michalewicz, Z. (ed.): *Proceedings of the 5th International Conference on Statistical and Scientific Database Management* (5SSDBM, Charlotte, NC, April 3-5), Lecture Notes in Computer Science 420, Berlin e. a.: Springer-Verlag, 1990, pp. 14-29

Sato81 Sato, H.: Handling Summary Information in a Database: Derivability, in: *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD'81, Ann Arbor, Michigan, April 29-May 1)*, 1981, pp. 98-107

Sato88 Sato, H.: A Data Model, Knowledge Base, and Natural Language Processing for Sharing a Large Statistical Database, in: Rafanelli, M.; Klensin, J.C.; Svensson, P. (Eds.): *Proceedings of the 4th International Working Conference on Statistical and Scientific Database Management (4SSDBM, Rome, Italy, June 21-23)*, 1988, pp. 207-225

Shos82 Shoshani, A.: Statistical Databases: Characteristics, Problems, and Some Solutions, in: *Proceedings of the 8th International Conference on Very Large Data Bases* (VLDB´82, Mexico City, Mexico, Sept. 8-10), 1982, pp. 208-222

SmSm77 Smith, J.M.; Smith, D.C.P.: Database Abstractions: Aggregation and Generalization, *ACM Transactions on Database Systems 2(1977)2*, pp. 105-133

Su83 Su, S.Y.W.: SAM*: A Semantic Association Model for Corporate and Scientific-Statistical Databases, *Journal of Information Sciences 29*, 1983, pp. 151-199

TsKl78 Tsichritzis, D.C.; Klug, A.: The ANSI/X3/SPARC DBMS framework report of the study group on database management systems, *Information Systems 3(1978)3*, pp. 173-191