Reverse Pivoting in Conceptual Information Systems

Jo Hereth[1] and Gerd Stumme[2]

[1] Technische Universität Darmstadt, Fachbereich Mathematik
Schlossgartenstraße 7, D-64289 Darmstadt, hereth@mathematik.tu-darmstadt.de
[2] Universität Karlsruhe (TH), Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB), D-76128 Karlsruhe, stumme@aifb.uni-karlsruhe.de

Abstract. In database marketing, the behavior of customers is analyzed by studying the transactions they have performed. In order to get a global picture of the behavior of a customer, his single transactions have to be composed together. In On-Line Analytical Processing, this operation is known as *reverse pivoting*. With the ongoing data analysis process, reverse pivoting has to be repeated several times, usually requiring an implementation in SQL. In this paper, we present a construction for conceptual scales for reverse pivoting in Conceptual Information Systems, and also discuss the visualization. The construction allows the reuse of previously created queries without reprogramming and offers a visualization of the results by line diagrams.

1 Introduction

With increasing ease of gathering data about customer behavior, Database Marketing has become an essential issue for many companies. Its aim is to facilitate decision support, customer-tailored production, and target-group specific advertising. These goals often require a clustering on the set of customers in order to retrieve groups of 'similar' customers which can then be treated in a uniform manner. The clustering is based on attributes like age, gender, address, etc. on the one hand, and their behavior (in terms of credit card transactions, web access patterns, etc.) on the other hand.

The data are usually stored in data warehouses [Ki96]. The classical conceptual database model is the star schema. It consists of a *fact table* and several *dimension tables*. The fact table of a star schema may for instance contain credit card transactions; and the dimension tables provide information about the structure of attributes like the time of purchase, the point of sales, the amount of the transaction, etc. Another star schema may consist of a fact table containing the customers, surrounded by dimension tables for the age, the gender, the address, etc. If the second star schema replaces a vertex of the first star, we obtain a *snowflake schema*. A variety of On-Line Analytical Processing (OLAP) [Th97] tools are on the market, which allow an intuitive access to data stored in such star and snowflake schemes.

The situation becomes more difficult when data has to be analyzed which is not in the center of a star/snowflake. This is for instance the case when clusters of customers are to be determined who are not solely 'similar' according to their demographic attributes, but also according to their buying behavior. As each transaction contains a customer ID, basically all information is available. The difficulty lies in the point that there are many transactions related to one customer, and it is *a priori* not clear in which way the information has to be extracted and aggregated. For instance, there are obvious queries like the total amount of money spent, but also more sophisticated queries like the distribution of the purchase dates over time. Usually, such queries require an explicit implementation in SQL. As the requests of the data analysts change over time, queries have to be implemented several times. Therefore, constructions are needed which allow to derive, from existing queries, queries against a table which is not in the center of a snowflake.

Conceptual Information Systems [SSVWW93] have been extensively applied to data where the conceptual database model is similar to a star schema. The implementation of the management system TOSCANA for Conceptual Information Systems [VW95] is based on a relational database management system, and offers different views on the data by means of *conceptual scales*. Conceptual scales can be considered as conceptual hierarchies analogue to the dimensions of the star schema [St00]. At the moment, Conceptual Information Systems do not support reverse pivoting. In this paper, we discuss how reverse pivoting can be done within Conceptual Information Systems. The idea is to construct new scales for the analysis of the customers based on the already existing scales for the transactions. We present a new scale construction and discuss its use. The advantage of this mathematically founded construction is two-fold: (*i*) the construction goes along with the visualization technique offered by line diagrams of concept lattices, and (*ii*) it allows to bypass SQL programming by offering a predefined operation.

This work has been done in cooperation with the database marketing department of a Swiss department store. In the project, we observed that reverse pivoting is not the only feature required for Conceptual Information System in a database marketing application. For instance, an important functionality in database marketing is the comparison of the same attributes for different groups, e. g. the comparison of sales in consecutive quarters or in different regions. In order to enable this functionality in Conceptual Information Systems, we have introduced *scale prisms* which allow to display a conceptual scale (or different ones) several time, each time applied to different data.

The remainder of this paper is organized as follows: The database marketing scenario is described in the next section (see also [HSWW00]). In Section 3 we recall the basics of the star and the snowflake schema as conceptual database model and explain the idea of reverse pivoting. In Section 4 we discuss reverse pivoting in Conceptual Information Systems. In this paper, we will recall only very briefly the basic notions of Conceptual Information Systems. We assume that the reader is familiar with the notions of formal contexts and concept lattices

as given, for instance, in [GW99].[1] In Section 5, we present the scale construction and discuss its use and its visualization. Here we also give the definition of scale prisms and provide an example. Section 6 concludes the paper.

2 The Database Marketing Scenario

The results presented in this paper have been inspired by a joint research project of Darmstadt University of Technology and the database marketing department of a Swiss department store (cf. [HSWW00, He00]). The database marketing department bases its market analysis on data collected from bonus card holders. The bonus card is a credit card which can be used in the shops of about a dozen companies. The operational systems of the companies collect information from more than 300,000 customers. The database marketing department has to incorporate the data from different sources and create a coherent database. The department analyzes these data in order to improve the marketing by so-called *direct mailings*, i. e. personalized advertisement by mail. For this purpose the database marketing department has to select the corresponding target group. The information is also used to strengthen the cooperation between the companies.

Here are some typical questions in this context:

1. How high were the scales in the shops of partner X?
2. Which customers are between 45 and 65 years old?
3. How many male card holders live near Zurich?
4. Which customers of Partner X are not yet customers of Partner Y?
5. How many customers have been inactive for more than twelve months?
6. Which card holders were active in a shop from Partner X last year?

Query 1 can be answered within a standard Conceptual Information System which is focused on the transactions. Queries 2 and 3 can be answered within a standard Conceptual Information System which is focused on the customers. Queries 4, 5, and 6 — which are the most interesting for the database marketing department —, however, need reverse pivoting.

If the set of questions to be asked is predefined, the reverse pivoting can be performed once, and the necessary data can be stored in the data table underlying the Conceptual Information System. However, in the process of analysis, the users sometimes wants to see yet another aspect of the data. This implies a rework of the underlying data table and a rewrite of the SQL-statements. In the project with the database marketing department, the reverse pivoting was performed manually, in order to gain experiences about the types of queries needed. The resulting system is still in use today in the database marketing department. Based on the experiences made, we developed the constructions presented in this paper.

[1] They can also be found in [HSWW00]: http://wwwbib.mathematik.tu-darmstadt.de/Math-Net/Preprints/Listen/shadow/pp2092.html

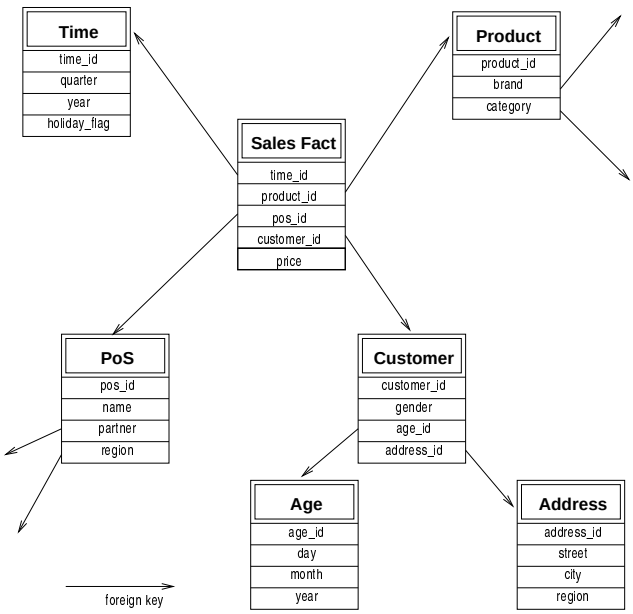

Fig. 1. Example of a snowflake schema

3 Snowflake Schemes and Reverse Pivoting

In a date warehouse like the one of our database marketing department, the standard approach to model the data is by *star schemes* and *snowflake schemes*. (cf. [Ki96]). Figure 1 shows the snowflake schema for the transaction data. The transaction data themselves are stored in a large, central *fact table* (the table `Sales Fact`); and information about the structure of attributes like 'time of purchase', 'point of sale' etc. is stored in *dimension tables*. These tables are linked to the central fact table by foreign keys.

The transactions are in the center of the schema. A typical query against these data is for instance the first query from above. This means that aggregations along (some of) the dimensions have to be performed to answer the queries. As transaction data are most often numerical — such as price, cost, or number of units sold —, computations based on these values are well supported by current database systems and allow fast answers. The responsiveness of analysis systems is fundamental for the required interactive use in database marketing. Most current implementations of Data Warehouse systems support therefore only analysis based on those numerical values.

The traditional approach of adding up or counting the transaction values does not help for an analysis of customers and customer groups. The principal object that needs to be modeled is the customer itself, not the transaction. The analyst wants to learn about the behavior of a customer, not about mere numbers. The straightforward solution seems to be to simply consider the table `Customer` as center of the snowflake, instead of the table `Sales Fact`. The problem is that now

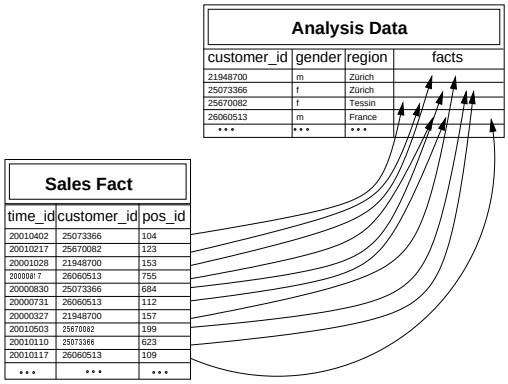

Fig. 2. The Reverse-Pivot from a transaction-oriented to a customer-oriented data model

the foreign key between both tables points in the wrong direction. Therefore the dimensions around the `Sales Fact` table cannot be used directly for aggregation of customer-centered data. For instance, while it makes perfectly sense to assign a customer feature (like the gender) to a transaction, it is not clear *a priori* how to assign a transaction feature (like point in time) to a customer.

The standard approach to solve this problem is to add fields derived from the `Sales Fact` table to the `Customer` table. This process is known as *reverse pivoting*. It is sketched in Figure 2. What part of the transaction data is of interest for the analysis depends on the context. In [Py99] it is therefore suggested to ask a domain expert which fields should be added to the customer data table and be present for future examination. For instance, we might be interested in the time of the first purchase, the time of the last purchase, or in the duration of the longest interval between two purchases.

4 Conceptual Information Systems and Reverse Pivoting

A *Conceptual Information System* is a navigation tool that allows dynamic browsing through and zooming into the data based on the visualization of conceptual hierarchies (called *conceptual scales*) by line diagrams. In this paper, we recall only very briefly the basic notions of Conceptual Information Systems, for further details we refer to [SSVWW93], [VW95], or [HSWW00]. We assume that the reader is familiar with the definition of formal contexts and formal concepts as defined, for instance, in [GW99].

Definition 1 (Conceptual Information Systems). *A* many-valued context *is a tuple* $\mathbb{K} = (G, M, W, I)$ *where* G*,* M*, and* W *are sets, and* $I \subseteq G \times M \times W$ *is the graph of a partial function from* $G \times M$ *to* W *(i. e.,* $(g, m, w_1) \in I$ *and* $(g, m, w_2) \in I$ *always implies* $w_1 = w_2$*). The elements of* G*,* M*, and* W *are called* objects*,* attributes*, and* attribute values*, respectively, and* $(g, m, w) \in I$ *is read:* "object g has attribute value w for attribute m"*. An attribute* m *may*

be considered as a partial mapping from G to W; therefore, $m(g) = w$ is often written instead of $(g, m, w) \in I$.

A conceptual scale *for an attribute $m \in M$ is a one-valued context $\mathbb{S}_m := (G_m, M_m, I_m)$ with $m(G) \subseteq G_m$. The context $\mathbb{R}_m := (G, M_m, J_m)$ with $(g, m) \in J_m :\iff (m(g), n) \in I_m$ is called the* realized scale *for the attribute $m \in M$. The* derived context *of $\mathbb{K}$ with respect to the conceptual scales $\mathbb{S}_m := (G_m, M_m, I_m)$, $m \in M$, is the formal context $(G, \bigcup_{m \in M} \{m\} \times M_m, J)$ with $(g, (m, n)) \in J :\iff (m(g), n) \in I_m$. A many-valued context together with a collection of conceptual scales with line diagrams of their concept lattices is called a* Conceptual Information System.

There is a strong analogy between the conceptual model of Conceptual Information Systems and star schemes. The fact table corresponds to the many-valued context (with the minor difference that the existence of set G is equivalent to a primary key in the fact table, which is not required in star schemes[2]). The conceptual scales correspond to the dimension tables of the star schema. Although the above definition is not directly equivalent to snowflake schemes, it can easily be extended.

Conceptual Information Systems are thus able (as OLAP systems) to handle data which are structured along a star or snowflake schema. The difference is that Conceptual Information Systems focus more on conceptual aspects of the data, while OLAP systems are centered on numerical evaluations. An extension of Conceptual Information Systems by numerical features is presented in [SW98], and the use of Conceptual Information Systems as OLAP tools is discussed in [St00].

Figure 3 shows the snowflake schema from Figure 1 as schema of a Conceptual Information System. Each conceptual scale can be used directly for a fact table (i. e., a many-valued context), as long as there is a path of foreign keys in the appropriate direction. For instance, all conceptual scales for customers can be directly reused for the transactions: we can reuse the scale 'age' of the customer dimension to cluster the transactions according to the age of the customers.

However, the main purpose of the database marketing group is analyzing the customers — and not the transactions — in order to find homogeneous groups of customers for direct mailing actions. Hence the `Customer` table has now to be considered as fact table. The scales for age, gender, and address can of course still be used, but as the foreign key between the `Customer` table and the `Sales Fact` table points in the wrong direction, the scales of the `Sales Fact` table cannot be reused directly. A reverse pivoting is necessary.

In the project with the database marketing department, the reverse pivoting was performed manually (i. e., implementing SQL queries) as described in the previous section. The aim was to gain experience about the types of queries needed. The result was the Conceptual Information System described in [HSWW00]. We could observe that in fact the system enabled an interactive process of conceptual data analysis leading to useful knowledge.

[2] In our database marketing application however, the transaction ID plays the role of the primary key.

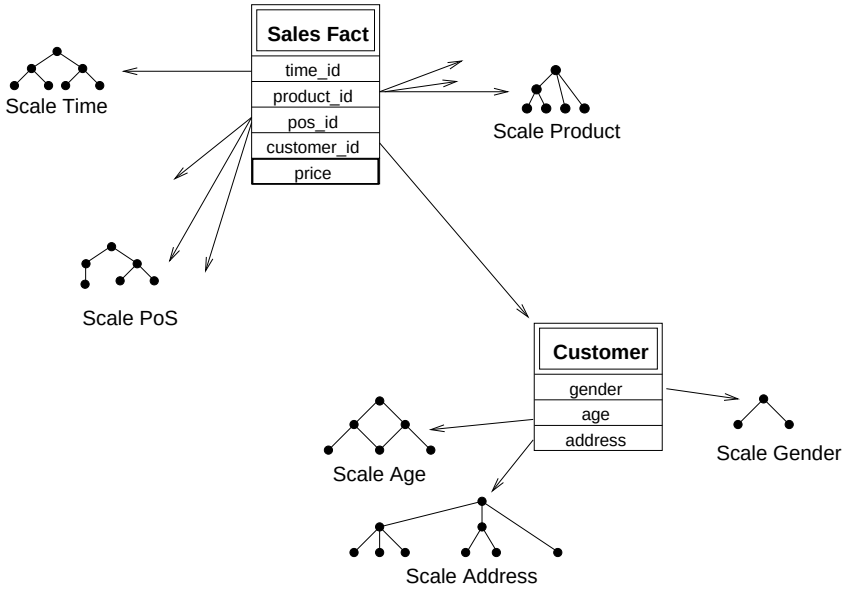

Fig. 3. The schema from Figure 1 as schema of a Conceptual Information System.

The system was constructed in tight cooperation with the database marketing department. Therefore the domain experts were involved from the beginning. The fields considered useful were included in the data table. After some time of data analysis however, the analysts got more and more familiar with the data and the information system. This led to a request for more in-depth information and to the need for more sophisticated queries. Therefore the data table had to be adapted and expanded several times. It became clear that this process had to be supported by a methodology, such that it (or at least parts of it) could be automated.

Very often the new information the analysts were looking for was describable in terms of the knowledge stated in already given scales. Therefore we developed different constructions for deriving new scales from the already given ones. The constructions substitute the explicit coding of the reverse pivoting in the database table. A modification of the fact table as in the previous section need not be performed any longer.[3] In the next section, we present the construction and discuss its use.

5 Conceptual Scaling Constructions

How can a scale that was designed for the transactions be reused for the analysis of the customers? We show how to construct new scales which can then be applied

[3] However, in trade-off for a better computational performance, it may sometimes still be sensible to do so.

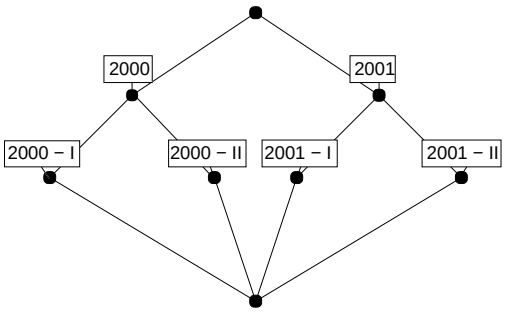

Fig. 4. The scale $\mathbb{S}_{\text{Time}}$.

for clustering the set of customers, i. e., which can be used for the reverse-pivoted data. We also discuss how their visualization can be supported by *iceberg concept lattices* [S+01].

There are two more constructions which support other aspects of reverse pivoting: adaptable and parameterized scales. Because of space restriction, we will not present them here. They are described in detail in [He00]. Instead, we present another construction which is not directly related to reverse pivoting, but which has nevertheless proven to be very useful in the database marketing scenario: so-called *scale prisms*, which allow the comparison of the same attributes for different groups, e. g. the comparison of sales in consecutive quarters or in different regions.

5.1 Power Scales

As example we use the scale $\mathbb{S}_{\text{Time}} := (G_{\mathbb{S}}, M_{\mathbb{S}}, I_{\mathbb{S}})$ in Figure 4. Applied on the transactions, it clusters them by half year intervals. Now we show how it can be used for the construction of a new scale which can be applied to the customers.

An analyst from the database marketing department often thinks of customers as "customers in the year 2000" or "customers in the first half of year 2001", but the scale in Figure 4 does not allow for all possibilities: Suppose a customer bought something in the second half of year 2000 and something in the first half of year 2001, but nothing in the other intervals. There is no appropriate concept in the scale for containing this customer. The only concept below the attributes "2000 - II" and "2001 - I" is the bottom element – which would imply all other intervals as well.

The problem is obviously that customers may have *combinations* of attributes where transactions only have one of them. For our first construction, the attributes we want to associate with a customer are all attributes of all his transactions. In other words, the customer should have an attribute assigned if there exists at least one transaction, which is in relation to this attribute. Formally we define:

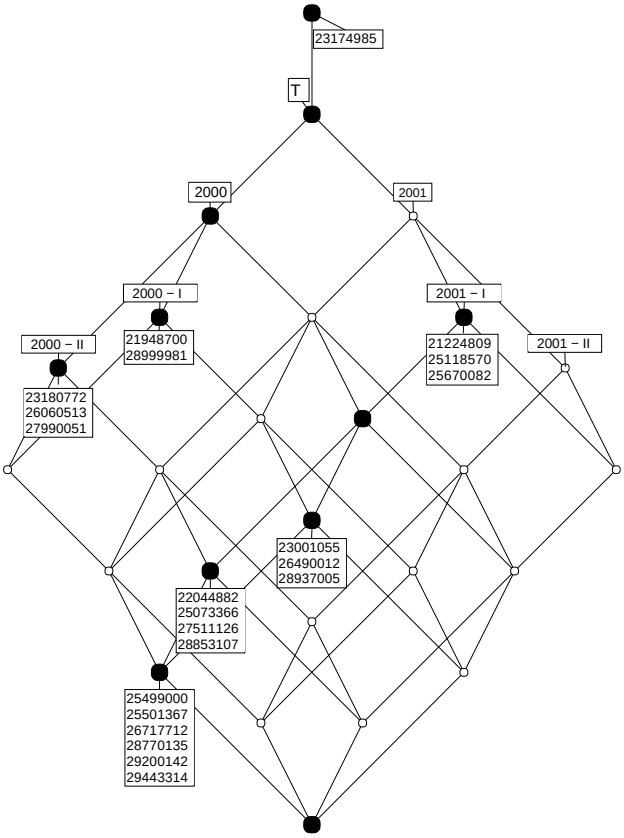

Fig. 5. The power scale $\mathbb{S}^{\mathfrak{P}}_{\text{Time}}$

Definition 2 (Power Scale). *Let* $\mathbb{S} := (G_{\mathbb{S}}, M_{\mathbb{S}}, I_{\mathbb{S}})$ *be a conceptual scale. Then the* power scale of $\mathbb{S}$ *is the scale* $\mathbb{S}^{\mathfrak{P}} := \left(\mathfrak{P}(G_{\mathbb{S}}), M_{\mathbb{S}}, I_{\mathbb{S}}^{\mathfrak{P}}\right)$ *with* $(A, m) \in I_{\mathbb{S}}^{\mathfrak{P}} :\Leftrightarrow \exists g \in A$ *with* $(g, m) \in I_{\mathbb{S}}$*, for* $A \subseteq G_{\mathbb{S}}$ *and* $m \in M$*.*

The line diagram of the power scale in our example is shown in Figure 5. We have applied it to a small example database containing 22 customers. It shows for instance that there are three customers (listed in the lower middle of the diagram) who have bought something in the first half of year 2000 and in the first half of year 2001, but nothing in the two other half year intervals.

The resulting concept lattices can be large relative to the concept lattice of the original scale:

Lemma 1. *If a scale* $\mathbb{S}$ *has* n *concepts, the following equation holds for the number* m *of concepts of* $\mathbb{S}^{\mathfrak{P}}$*:*

$$n \leq m \leq 2^n \quad .$$

The concept lattice of the power scale therefore can be much more complex than the concept lattice of the original scale. A first step to derive a line diagram for the power scale based on the line diagram of the original scale is the following theorem, which asserts that the structure of the original concept lattice is preserved. The proof of this theorem can be found in [He00].

Theorem 1. *In the concept lattice* $\underline{\mathfrak{B}}(\mathbb{S}^{\mathfrak{P}})$ *exists a* $\vee$*-subsemilattice isomorphic to the concept lattice* $\underline{\mathfrak{B}}(\mathbb{S})$ *of the original scale. The concepts of this subsemilattice are all concepts from* $\mathfrak{B}(\mathbb{S}^{\mathfrak{P}})$ *whose intents are intents of a concept in* $\mathfrak{B}(\mathbb{S})$.

As the order is preserved, the layout of the original scale may be used as a basis for the layout of the power scale. Additionally we know that the concept lattices of power scales are distributive (cf. [He00]), which allows to apply the algorithms presented in [Sk89], yielding readable line diagrams.

5.2 Power Scales of the &–Product

A particular advantage of conceptual scales is that they can easily be combined to give a more detailed view on the data. This possibility allows for more flexibility in the analysis process. Combining scales is usually done by building the semi-product of the scales [GW99]. The new conceptual scale can be represented by a nested line diagram. E. g., if there are two scales $\mathbb{S}_{\text{Year}}$ and $\mathbb{S}_{\text{PoS}}$ showing the year in which, and the point of sale where a transaction was performed, the scale $\mathbb{S}_{\text{Year}} \between \mathbb{S}_{\text{PoS}}$ shows the information of both scales simultaneously: The user can for instance see which transactions were performed at the department store in the year 2000.

The straightforward approach to study the customer behavior according to both the times and the locations of purchase would be to apply the power scale construction on the semi-product $\mathbb{S}_{\text{Year}} \between \mathbb{S}_{\text{PoS}}$: The power scale $(\mathbb{S}_{\text{Year}} \between \mathbb{S}_{\text{PoS}})^{\mathfrak{P}}$ allows to see, for instance, the group of all customers who bought something in the year 2000 and who bought something in the department store. But unfortunately, this is not equal to the group of customers who have effectuated at least one transaction in the department store in the year 2000. If a customer bought something in the department store in 2001 and something else somewhere else in year 2000, he still fulfills the two criteria. Hence we obtain a larger group of customers than expected.

But obviously the information we need exists in the data. The solution is to replace the semi-product by the so-called &-product. The &-product construction is already described in [GW99], but has not yet been investigated or used much before. The &-product of two scales is defined as follows:

Definition 3 (&-product). *Let* $\mathbb{S}_1 := (G_1, M_1, I_1)$ *and* $\mathbb{S}_2 := (G_2, M_2, I_2)$ *be conceptual scales. The* &-product *of these scales is defined as* $\mathbb{S}_1 \,\&\, \mathbb{S}_2 := (G_1 \times G_2, M_1 \times M_2, J)$ *with* $((g,h),(m,n)) \in J :\Leftrightarrow (g,m) \in I_1 \wedge (h,n) \in I_2$.

The concept lattice of the &-product has less concepts (and hence less information) than the one of the semi-product. It is isomorphic to a $\wedge$-subsemilattice

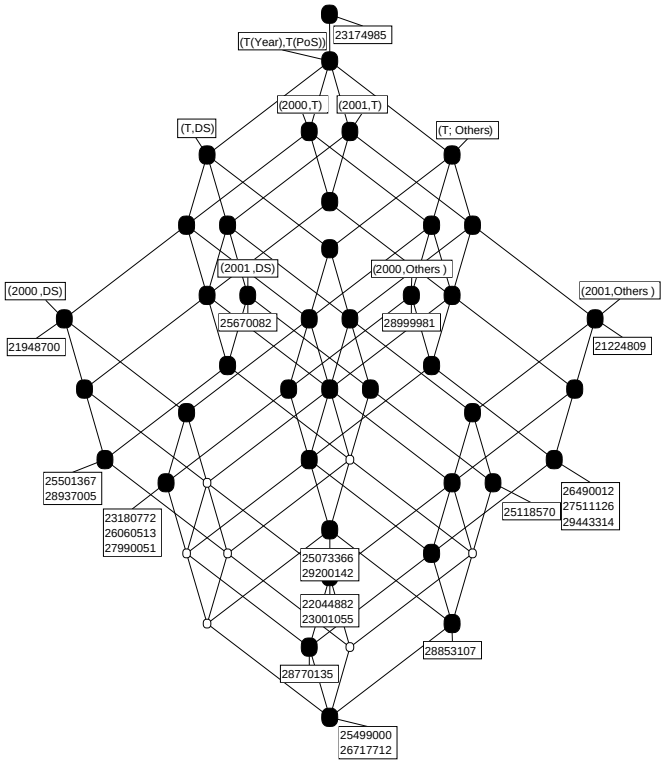

Fig. 6. The power scale $(\mathbb{S}_{\text{Year}} \,\&\, \mathbb{S}_{\text{PoS}})^{\mathfrak{P}}$

of the latter one. On the other hand, its context is considerably larger. Therefore the semi-product has been advantageous in the Conceptual Information Systems built in the past (which had no need of reverse-pivoting).

Applying the power scale construction to these products inverts the size relation. If there is an attribute in each scale which is related to all objects, the concept lattice $\underline{\mathfrak{B}}((\mathbb{S}_1 \,\overline{\&}\, \mathbb{S}_2)^{\mathfrak{P}})$ contains more concepts (and more information) than the concept lattice $\underline{\mathfrak{B}}((\mathbb{S}_1 \,\&\, \mathbb{S}_2)^{\mathfrak{P}})$, which is isomorphic to a sublattice of the former one. In Figure 6 we see the power scale $(\mathbb{S}_{\text{Year}} \,\&\, \mathbb{S}_{\text{PoS}})^{\mathfrak{P}}$. Its left-most concept shows, for instance, that customer 21948700 and all $2+3+2+1+2 = 10$ customers listed further below bought something in the department store in year 2000. The immediate super-concept of this concept contains all customers who bought something in the department store and something (eventually different) in year 2000.

5.3 Iceberg Concept Lattices

As seen in the last two subsections, concept lattices of power-scales can become very large. But in most cases the user is not interested to see all concepts of the lattice at once. For analyzing overall trends, the concepts containing no or only

few customers are not of interest. For those cases it is useful to display only a part of the concept lattice. This can be done by *iceberg concept lattices* as described in [S+01], i. e. by hiding all concepts whose extent is below a certain percentage of the total set of customers. This way, the number of concepts to be displayed can be reduced considerably, making the diagram easier to read. In Figure 7 the iceberg lattice version of the diagram in Figure 6 is given. Only those concepts whose extents contain at least 50 % of the 22 customers are shown.

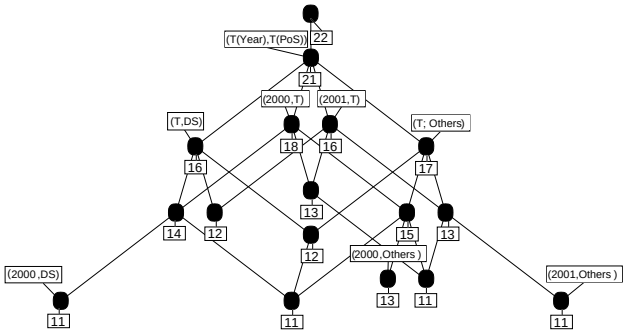

Fig. 7. An iceberg lattice of the power-scale $(\mathbb{S}_{\text{Year}} \& \mathbb{S}_{\text{PoS}})^{\mathfrak{P}}$

There is no problem in labeling iceberg concept lattices as long as the labeling is restricted to the cardinalities (which most often is sufficient for database marketing). Problems may arise when the names of the objects have to be displayed. This is discussed in [He00] (for the more general case of a 'Skalenbund').

5.4 Scale Prisms

An important functionality in database marketing is the comparison of the same attributes for different groups, e. g. the comparison of sales in consecutive quarters or in different regions. A conceptual scale allows a detailed view on the data for one part only. For the comparison of several parts or facets, the construction of a *scale prism* can be useful.

Definition 4 (Scale Prisms). *A scale prism consists of an ordered set* $(P, \leq)$ *and a set of conceptual scales* $(\mathbb{S}_p)_{p \in P}$ *which are consistent with the context.*

One could argue that the definition of scale prisms should include some common features of the indexed scales. However, our experience showed that the situations in which one wants to apply the construction are so many-fold, that such a constraint would be too restrictive. It is a modeling decision of the user to choose scales which are reasonably related.

The scale prism is displayed like a nested line diagram: A line diagram of the ordered set $(P, \leq)$ is drawn as outer diagram, and each realized scale is drawn in the corresponding node. To emphasize that it is not meant to be read as

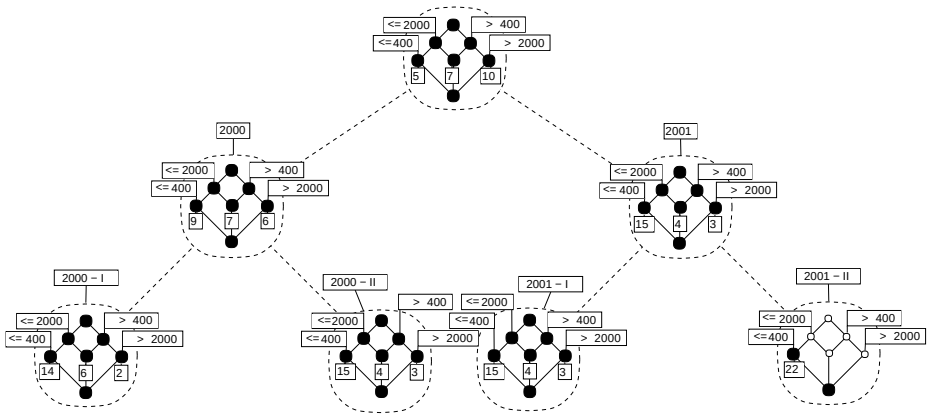

Fig. 8. A scale prism showing the distribution of the amount of money spent per customer in several time intervals

an ordinary nested line diagram (since different orders are meant in the outer diagram and the inner diagrams), the lines of the outer diagram are drawn as dotted lines.

An example is given in Figure 8. If we compare, for instance, the left-most concepts in the four lowest nodes of the outer diagram, we observe that the number of customers who spent relatively few money (i. e., below 400 Swiss francs) has increased from 14 in the first half of year 2000 to 15 in the second half of year 2000. It remained constant for the next half year, and increased again to 22 in the second half of year 2001.

6 Conclusion

In this paper, we have presented a mathematically founded construction for conceptual scales which supports reverse pivoting in Conceptual Information Systems. We have shown that the power-scale and the power-scale of the &-product offer three advantages:

- The constructions can be considered as predefined operations for data preparation which allow to bypass SQL programming every time the data analyst requires a new conceptual scale.
- The constructions go along with the visualization technique of line diagrams of concept lattices. This allows to closely integrate the reverse pivoting operations in the graphical user interface.
- The constructions can be combined with the technique of iceberg concept lattices, i. e., all but the most general concepts are pruned in order to reduce the complexity of the visualization.

We have further presented scale prisms as means for data analysis with Conceptual Information Systems in an OLAP-style application.

In order to test the usefulness of the constructions, the operations have been first performed manually in the described database marketing scenario. In order to make the techniques available to a larger group of users, it is planned to implement the techniques in the management system TOSCANA for Conceptual Information Systems.

An interesting open research question is the combination of the power-scale constructions with other data mining techniques. For instance, they might be used to improve the performance of algorithms for computing association rules or for discovering subgroups by restricting the search space.

References

[GW99] B. Ganter, R. Wille: *Formal Concept Analysis : Mathematical Foundations*. Springer Verlag, Berlin – Heidelberg – New York, 1999.

[He00] J. Hereth: *Formale Begriffsanalyse im Data Warehousing*. Diploma thesis, TU Darmstadt, 2000.

[HSWW00] J. Hereth, G. Stumme, U. Wille, R. Wille: Conceptual Knowledge Discovery and Data Analysis. In: B. Ganter, G. W. Mineau (eds.): *Conceptual Structures: Logical, Linguistic, and Computational Issues*. LNAI 1867. Springer, Berlin – Heidelberg – New York 2000, 421-437.

[Ki96] R. Kimball: *The Data Warehouse Toolkit*. John Wiley & Sons, New York 1996.

[Py99] D. Pyle: *Data preparation for data mining*. Morgan Kaufman Publishers, San Francisco 1999.

[SSVWW93] P. Scheich, M. Skorsky, F. Vogt, C. Wachter, R. Wille: Conceptual Data Systems. In: O. Opitz, B. Lausen, R. Klar (eds.): *Information and Classification*. Springer, Berlin–Heidelberg 1993, 72–84

[Sk89] M. Skorsky: How to draw a concept lattice with parallelograms. In: R. Wille (ed.): *Klassifikation und Ordnung*. Indeks-Verlag, Frankfurt 1989, 191–196.

[St00] G. Stumme: Conceptual On-Line Analytical Processing. In: K. Tanaka, S. Ghandeharizadeh, Y. Kambayashi (eds.): *Information Organization and Databases*. Chpt. 14. Kluwer, Boston–Dordrecht–London 2000

[S+01] G. Stumme, R. Taouil, Y. Bastide, N. Pasqier, L. Lakhal: Computing Iceberg Concept Lattices with Titanic. *J. on Knowledge and Data Engineering* (submitted)

[SW98] G. Stumme, K. E. Wolff: Numerical Aspects in the Data Model of Conceptual Information Systems. In: Y. Kambayashi, Dik Kun Lee, Ee-Peng Lim, M. K. Mohania, Y. Masunaga (eds.): *Advances in Database Technologies. Proc. Intl. Workshop on Data Warehousing and Data Mining, 17th Intl. Conf. on Conceptual Modeling (ER '98)*. LNCS **1552**, Springer, Heidelberg 1999, 117–128

[Th97] E. Thomsen: *OLAP Solutions: Building Multidimensional Information Systems*. John Wiley & Sons, New York 1997.

[VW95] F. Vogt, R. Wille: TOSCANA — a graphical tool for analyzing and exploring data. In: R. Tamassia, I. G. Tollis (eds.): *Graph Drawing '94*. LNCS **894**. Springer, Berlin–Heidelberg 1995, 226–233