RESEARCH PAPERS OF WROCŁAW UNIVERSITY OF ECONOMICS
No. 85 2009
Advanced Information Technologies for Management – AITM 2009

Jerzy Korczak, Błażej Oleszkiewicz

Wrocław University of Economics, Wrocław, Poland
jerzy.korczak@ue.wroc.pl, blajan@poczta.fm

MODELLING OF DATA WAREHOUSE DIMENSIONS FOR AML SYSTEMS

Abstract: In this article the three main models of data warehouse dimensions (two classical models and hierarchical, based on hierarchical tables) will be evaluated within the context of the use in Anti-Money Laundering systems. The main focus will be given to a new extension of the Analytical SQL Server: hierarchical tables and vector types used to identify elements in hierarchical tables. A number of examples of time dimension modelling and of Cartesian products computing OLAP will also be presented.

1. Introduction

Dimensional modelling is one of the most important tasks within the data warehouse design process. Its objectives concern the optimization of decision support query performance; the reduction of data redundancy of the data model; the efficient retrieval of information containing specific properties; and the simplification of data warehouse maintenance. In general, a logical data model is developed according to the application process to be served. This also determines the granularity of the fact table that is the focus of dimensional modelling. The fact table is surrounded by dimension tables creating visually a star or snowflake schemas.

Dimensional modelling has also a great influence on the implementation of data warehousing [Adamson 2006; Szirtes 2007]. There are two classical approaches to designing a data warehouse model: relational modelling and dimensional modelling [Inmon 2005; Kimball, Ross 2002]. However both these approaches present weaknesses as seen within the context of Anti-Money Laundering (AML) systems.

In this paper, a hierarchical data model is presented and evaluated in relation to a real-life application.

In the Analytical SQL Server two categories of dimensions have been implemented: the uniform, that corresponds to dimensions in classical data warehouses; and the non-uniform [Korczak et al. 2008]. These two categories may be implemented using relational or dimensional models. However experience shows that each of these models demonstrates serious limitations and drawbacks.

To illustrate the advantages of the hierarchical data model, *Time* dimension will be taken as an example. First, we will consider different models of time dimension (in terms of Analytical SQL as uniform dimension) and then show the benefits of applying a hierarchical model for Charts of Accounts of General Ledger in the bank. The example of *Time* dimension was chosen because we can only compare the classical models used in data warehouses with a hierarchical model Analytical SQL (uniform model). Analytical SQL heterogeneous models, as exemplified by the dimension of Chart of Accounts of General Ledger, have no representation in the classical models. Analytical SQL non uniform models, as exemplified by the dimension of Charts of Accounts of General Ledger, do not have equivalent representation in the classical data warehouse models.

Dimension modelling in the data warehouse is one of the most important processes related to data warehouse exploitation. During the data warehouse design we need to consider [Inmon 2005; Kimball 2002]:

– the possibility to represent the modelled reality,
– the simplicity of implementation,
– the efficiency of queries based on the size of models.

The remainder of this article will show that the use of classical dimension models of a data warehouse may cause many problems associated with implementation; mainly this is because of the impossibility of modelling all the actual data structures, and the lack of correlation between simple implementation and the efficiency of queries based on dimensions.

In this article examples of *Time* dimension models and their associated constraints will be considered. These examples will refer to the more general problem of dimension modelling in data warehouses.

In the Analytical SQL Server two categories of data warehouse dimensions are introduced:

– uniform dimensions (corresponding to the dimensions used in the classic data warehouses);
– non-uniform dimensions.

Referring to traditional data warehouse solutions, two models of uniform dimensions are used: the non-normalized dimension, and normalized dimension. However, because both these data structures have some drawbacks and limitations, an additional data structure called a hierarchical dimension will be introduced in this article.

2. Classical models of time dimension

2.1. Non-normalized time dimension

The simplest model of time dimension used in conventional data warehouse is a flat data structure describing the daily points in time, while retaining the additional information on the time points in the hierarchy of time dimensions (month,

quarter, year). In Table 1 an example of the table-time corresponding to SQL table is shown.

Table 1. Example of the relation describing *Time*

Id	Date	Year	Quarter	Month	Day
1	2009-01-01	2009	1	1	1
2	2009-01-02	2009	1	1	2
…	…	…	…	…	…
31	2009-01-31	2009	1	1	31
32	2009-02-01	2009	1	2	1
…	…	…	…	…	…

Source: own elaboration.

It should be noted that the use of this type of structure has several advantages, notably:

– the simple structure of the SQL table is easy to implement,
– a single table in the database,
– easy to define SQL queries relating to the elements of the dimension,
– simplicity of interpretation,
– easy to identify elements within the structure.

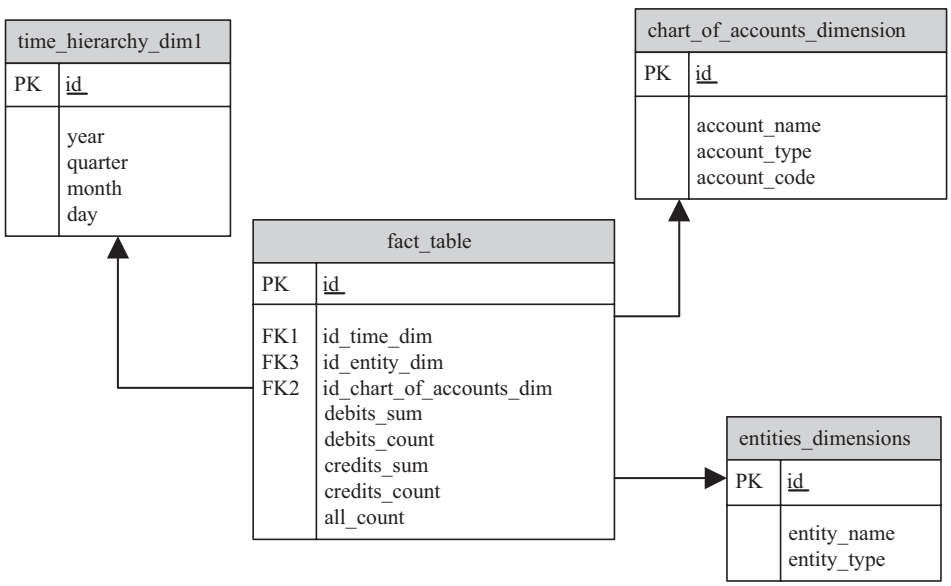


Figure 1. Example of a star model

Source: own elaboration.

However, designed in such a way this dimension has several major drawbacks:
– granularity only at the lowest level (in most cases it is the day),
– the lack of unambiguous identification of all elements of time dimension (in this example, you cannot clearly identify: month, quarter, year).

The use of "non normalized" dimension of time is typical of the star model as shown in Figure 1.

The figure shows a star model based on the fact table and three dimension tables (including a table with the time dimension).

Thus, data warehouse operations that relate to the time dimension do not require costly join operations. On the other hand, the aggregation of data warehousing at higher levels than the day require the conversion of aggregates in each case.

2.2. Normalized time dimension

A much more complex SQL structure is called a "normalized" form of time dimension. In this we are dealing with a constellation of tables, logically linked together respecting the principles of normalization and reflecting the complete hierarchical structure of time.

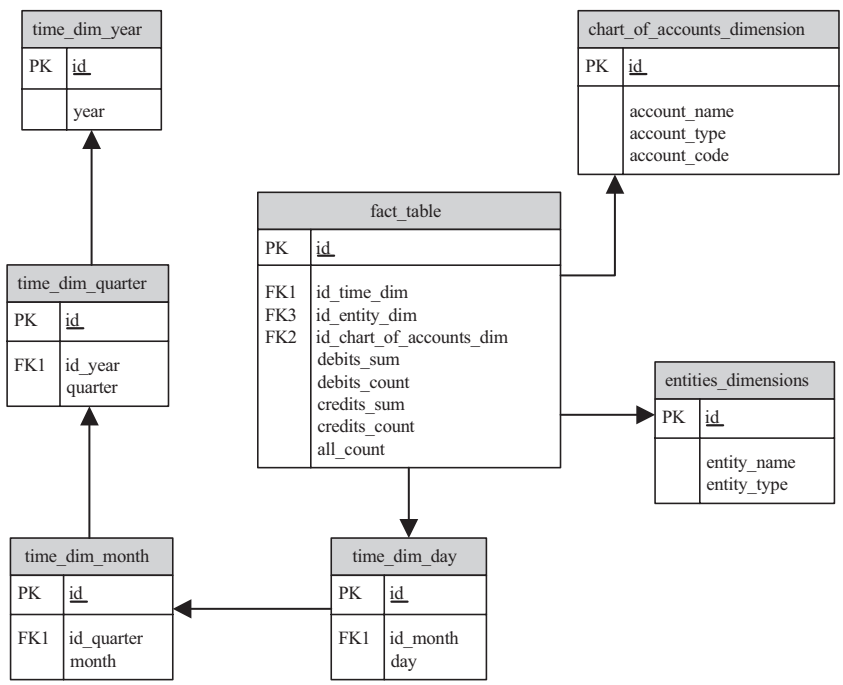


Figure 2. A snowflake schema

Source: own elaboration.

In relation to "non normalized" time dimension, the "normalized" model has a number of advantages:
– granularity at all levels of the hierarchy of time dimension,
– possibility to identify all elements in the time hierarchy.

On the other hand, this model also has several limitations:
– constellation tables, which causes problems with the defining of SQL queries,
– complex identification of elements in a hierarchy, resulting mainly from a constellation of tables and the necessity of analysis of the links between them,
– the need for complex identifiers,
– table identifiers and element identifiers,
– difficulties in interpretation.

The model of data warehouse is in this case called a snowflake schema; an example is show in Figure 2.

The figure illustrates a schema based on one fact table and three dimension tables where the time dimension is the normalized structure consisting of four tables.

The primary benefit in the use of normalized time dimension is the possibility of reusing the already computed aggregates. However, it should also be noted that the references to the time dimension result in extremely expensive join operations.

3. Hierarchical representation of time dimension

Due to the limitations of the classical structure of time dimension, non-normalized as well as normalized, in Analytical SQL Server a hierarchical model of time dimension based on hierarchical tables (the specific extension ASQL) has been developed. An example of hierarchical representation of time dimension is show in Table 2.

Table 2. Example of hierarchical representation of *Time* dimension

	Time dimension	Time terms	Time interpretation
1	2009	Year	Year 2009
2	2009-Q1	Quarter	Year 2009, Quarter 1
3	2009-Q1-01	Month	January 2009
4	2009-Q1-01-01	Day	January 1, 2009
5	2009-Q1-01-02	Day	January 2, 2009
…	…	…	…
33	2009-Q1-01-31	Day	January 31, 2009
34	2009-Q1-02	Month	
35	2009-Q1-02-01	Day	February 1, 2008
…	…	…	…

Source: own elaboration.

With this structure the objective of integrating the advantages of time dimension in the form of "non-normalized" and "normalized" is achieved. This consists of:
– assured granularity at all levels; unique SQL structure,
– easy implementation and interpretation,
– easy operations on the table structure and hierarchy,
– star schema avoiding of costly join operations.

The data warehouse model using a hierarchical dimension of time continues to be a star-type, as shown in Figure 3.

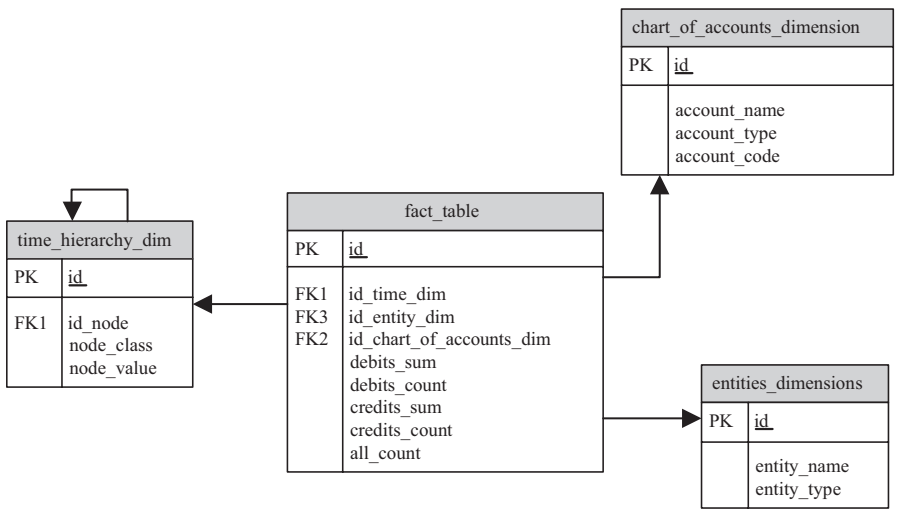


Figure 3. Example of hierarchical model

Source: own elaboration.

This figure presents the logical data warehouse model based on a star schema with three dimension tables, and the time dimension as hierarchical structure.

By applying the hierarchical-representation of time dimension the data warehouse demonstrates many advantages, notably:
– the elimination of costly join operations on the database,
– the possibility of reuse of previously computed aggregates.

In the next section, an example of usage a hierarchical modelling of time dimension will be discussed in Anti-Money Laundering system.

4. Uniform and non uniform dimensions in Analytical SQL

4.1. Data warehouse structures

The previous considerations related to modelling of the time dimension concern the so-called class of homogeneous dimensions which are widely used in current

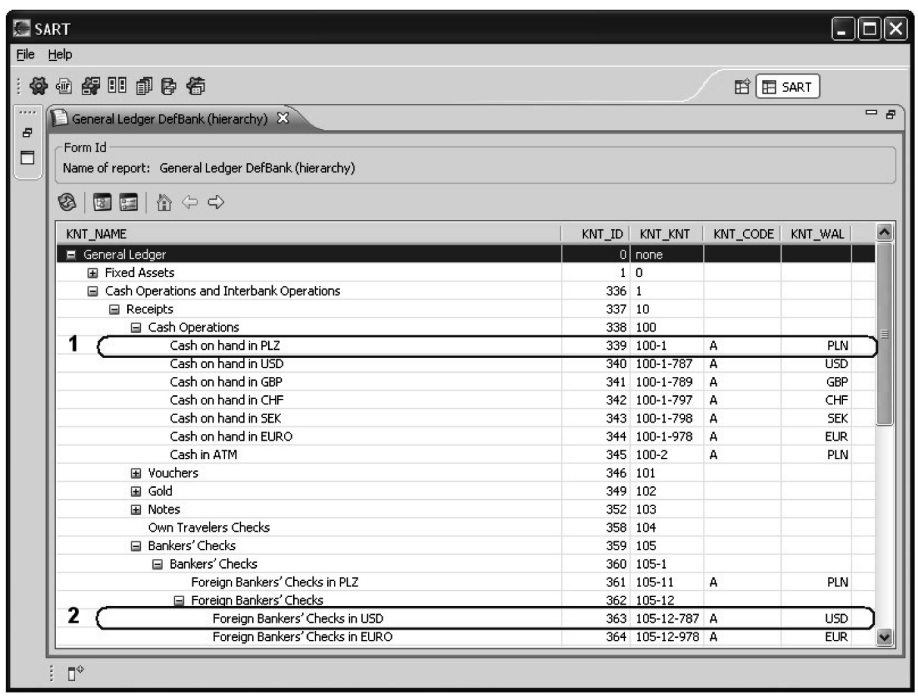
data warehouses (known as the classical data warehouse). It should be pointed out that the current data warehouse modelling allows only homogeneous dimensions. However, in business applications very often we meet with another category, the dimensions named in ASQL heterogeneous dimensions.

Consider the example of a hierarchical structure of a Chart of Accounts within a bank's General Ledger in which there are two types of accounts: analytic and synthetic. Each synthetic account may consist of any number of synthetic and analytical accounts, and analytical account does not contain any associated accounts.

In Figure 4 the heterogeneity can be seen in two encircled dimensions which differ in terms of the structure of the hierarchy. We can see that both hierarchical structures:

– [PK, KS, KS, KS, KA] for a KA "Cash in hand in PLZ (ID: 339 in column KNT_ID),
– [PK, KS, KS, KS, KS, KS, KA] for a KA "Foreign bankers' cheques in U.S. dollars" (ID: 363 in column KNT_ID),

are different from each other in terms of the structure of the hierarchy.



KNT_NAME	KNT_ID	KNT_KNT	KNT_CODE	KNT_WAL
General Ledger	0	none		
Fixed Assets	1	0		
Cash Operations and Interbank Operations	336	1		
Receipts	337	10		
Cash Operations	338	100		
Cash on hand in PLZ	339	100-1	A	PLN
Cash on hand in USD	340	100-1-787	A	USD
Cash on hand in GBP	341	100-1-789	A	GBP
Cash on hand in CHF	342	100-1-797	A	CHF
Cash on hand in SEK	343	100-1-798	A	SEK
Cash on hand in EURO	344	100-1-978	A	EUR
Cash in ATM	345	100-2	A	PLN
Vouchers	346	101		
Gold	349	102		
Notes	352	103		
Own Travelers Checks	358	104		
Bankers' Checks	359	105		
Bankers' Checks	360	105-1		
Foreign Bankers' Checks in PLZ	361	105-11	A	PLN
Foreign Bankers' Checks	362	105-12		
Foreign Bankers' Checks in USD	363	105-12-787	A	USD
Foreign Bankers' Checks in EURO	364	105-12-978	A	EUR

Figure 4. An example of a Chart of Account of General Ledger

It should be noted that this type of structure cannot be described as a classic dimension in the data warehouse. Hence, to model heterogeneous structures in Ana-

lytical SQL Server, a new type of structures called hierarchical SQL tables has been introduced.

4.2. Hierarchical tables in Analytical SQL

Analytical SQL data structures such as hierarchical tables are a combination of two common structures used in SQL databases: the hierarchy and the table. From the viewpoint of dimensional modelling of the hierarchical data warehouse, the table seems to be the ideal data structure as we can process it both as a tree, as well as a typical table. In addition, operations on hierarchical structures are carried out iteratively rather than recursively. The examples show that the dimension modelling of data warehouses using hierarchical tables is profitable not only as hierarchical structures but also as tabular representation In addition, the hierarchical tables also provide the possibility of hierarchical modelling of heterogeneous dimensions, which, in turn, solves one of the biggest problems existing in traditional data warehouses. In Analytical SQL Server all dimensions are modelled as hierarchical tables.

4.3. Operations on dimension structures in Analytical SQL

The hierarchical data model has to be completed by the definition of the basic operations on data structures. In this section the operations implemented in Analytical SQL based on hierarchical dimensions in homogeneous and heterogeneous hierarchies will be described.

The fundamental problem in modelling the data warehouse based on the dimensions of homogeneous and heterogeneous type hierarchies is to identify the elements of dimension. Just as it has already been signalled, a major challenge in full database granularity at all levels. As an example, the time dimension will be considered. In Analytical SQL in order to define a date, we have introduced an additional data type. It is a vector of integers corresponding to successive date elements. Individual entries of the vector have the following meanings:

Position in vector	Position meaning
0	Time dimension
1	Year
2	Quarter
3	Month
4	Day
5	Hour
6	Minute
7	Second
8	1/100 second

For example, the date represented by the vector [0,2009,1,1,25] is the 25 January 2009.

While [0,2009,1] means the first quarter of 2009. As has already been described in our previous articles [Korczak et al. 2008], the vector type is a specific extension of the ASQL Server. This introduction of a new data type for time is due to the need to use the time points in full granularity. With this type of data we can easily build a time dimension model and identify its individual elements at each level of the hierarchy.

4.4. Identification of element in dimension

The first and basic operation of the structure dimension of the data warehouse is to identify the element dimensions. In this case SQL query using the parameter "*time*" has to return a data record describing the time element. Let us consider a typical example of the identification data record "*day*" of time dimension:

```
SELECT * FROM time_dimension WHERE sql_date = '2009-01-25'
```

This query returns the record which in the field SQL date has the value **'2009-01-25'**, i.e. 25th January 2009. It should be noted that the type DATE in SQL allows only the manipulation of the complete date. Therefore, it is extremely difficult in pure SQL to build the warehouse dimension of time to identify for example months – we cannot do that in a simple and intuitive manner, for example, setting the date for '**2009-01**', because the date type in SQL does not provide such notation. Therefore, as mentioned earlier, ASQL introduces the vector type allowing definition of the "partial" date. The above example of time element identification of the vector notation can be defined as follows:

```
SELECT * FROM time_dimension WHERE vdate = [0, 2009, 1, 25]
```

In this case the identification of the month of January in the year 2009, the query will look as follows:

```
SELECT * FROM time_dimension WHERE vdate = [0, 2009, 1]
```

So we can see here two very important operational ASQL features of data warehouse processing: easy identification of data records, e.g. – the vectors [**0,2009,1,1,25**] and [**0, 2009,1,1**] identify two completely different types of data records: day and month, respectively, plus the facility to parametrize queries to retrieve particular data records.

4.5. Identification of Cartesian products in data warehouse

The next step in using the data warehouse is to identify the Cartesian products OLAP based on dimensions. Cartesian products OLAP are built as intersections with

different levels of dimensions defined by the measurements of the fact table. Because the dimensions in the Analytical SQL Server have full granularity, the OLAP cubes are also aggregated at each level of their dimensions in relation with the dimensions of the data warehouse and measurements of fact table.

Let us consider then a simple example of an event in the General Ledger based on the time dimension, the Chart of Accounts, the operation code and the customer's account, consisting of the decree acquisition based on the mentioned dimensions. The event registration in the data warehouse consists of writing a data record in which is kept information about the identifiers of dimensions and measurement.

On this basis, Cartesian products are built in such a way that starting with the identifiers of fact dimensions, all Cartesian products are created respecting the granularity of all dimensions composing the given fact.

For example, consider the dimensions of time and account:

– ID indicates the time of day: 2009-01-25 which gives the vector notation [**0,2009,1,1,25**];
– account ID: 21456 which gives a vector notation [**0,21456**] (0 encodes all accounts).

For this example will be built among others the following Cartesian products ([*time dimension*] × [*account dimension*]):

– [**0,2009,1,1,25**]× [**0,21456**] describes the product of account #21456 in 2009-01-25,
– [**0,2009,1,1**] × [**0,21456**] describes the product of account #21456 in the month of 2009-01,
– [**0,2009,1**] × [**0,21456**] describes the product of account #21456 in the first quarter of 2009,
– …
– [**0,2009,1**] × [**0**] – describes the product of all accounts in the first quarter of 2009, etc.

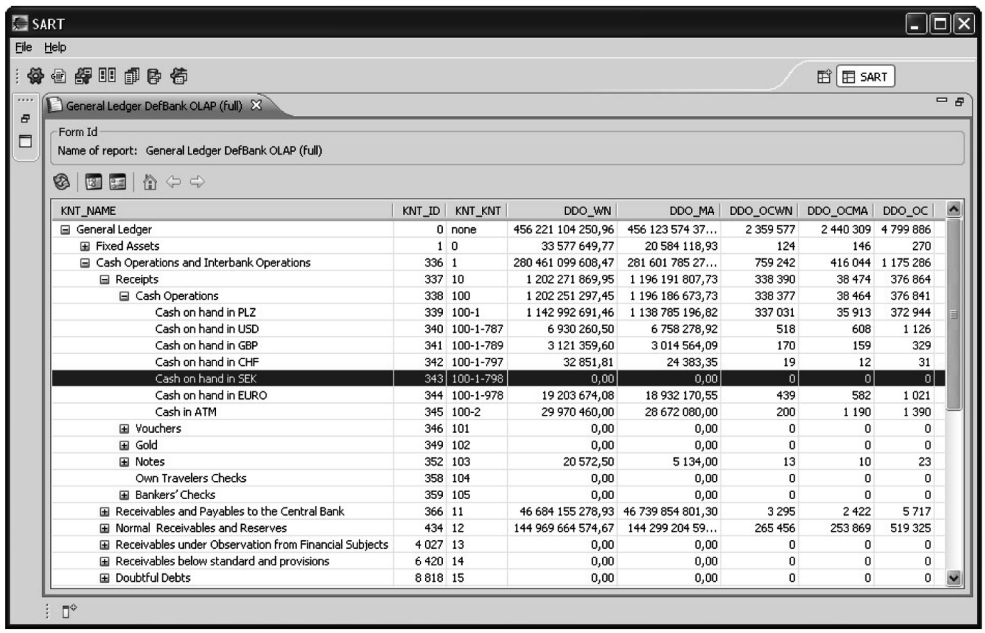
The above example shows the use of vector type for building the Cartesian products of OLAP cubes of the data warehouse – used here in a position in the vector to determine the level of dimension.

From the practical point of view, this mechanism was used to build the OLAP cubes of data warehouse of General Ledger in the SART system[1].

In the following figures, two reports are shown relating to the turnover on the account of the same customer, with that in Figure 5 showing the turnover view the Chart of Accounts, while in Figure 6 the report is based on the view-time dimension. In both cases, we have a complete aggregation of data relating to all levels of dimensions.

---

[1] SART – System for Analysis and Registration of Transactions (in Polish: System Analiz i Rejestracji Transakcji – SART), AML software based on Analytical SQL Server developed by TETA SA Company.



KNT_NAME	KNT_ID	KNT_KNT	DDO_WN	DDO_MA	DDO_OCWN	DDO_OCMA	DDO_OC
General Ledger	0	none	456 221 104 250,96	456 123 574 37...	2 359 577	2 440 309	4 799 886
Fixed Assets	1	0	33 577 649,77	20 584 118,93	124	146	270
Cash Operations and Interbank Operations	336	1	280 461 099 608,47	281 601 785 27...	759 242	416 044	1 175 286
Receipts	337	10	1 202 271 869,95	1 196 191 807,73	338 390	38 474	376 864
Cash Operations	338	100	1 202 251 297,45	1 196 186 673,73	338 377	38 464	376 841
Cash on hand in PLZ	339	100-1	1 142 992 691,46	1 138 785 196,82	337 031	35 913	372 944
Cash on hand in USD	340	100-1-787	6 930 260,50	6 758 278,92	518	608	1 126
Cash on hand in GBP	341	100-1-789	3 121 359,60	3 014 564,09	170	159	329
Cash on hand in CHF	342	100-1-797	32 851,81	24 383,35	19	12	31
Cash on hand in SEK	343	100-1-798	0,00	0,00	0	0	0
Cash on hand in EURO	344	100-1-978	19 203 674,08	18 932 170,55	439	582	1 021
Cash in ATM	345	100-2	29 970 460,00	28 672 080,00	200	1 190	1 390
Vouchers	346	101	0,00	0,00	0	0	0
Gold	349	102	0,00	0,00	0	0	0
Notes	352	103	20 572,50	5 134,00	13	10	23
Own Travelers Checks	358	104	0,00	0,00	0	0	0
Bankers' Checks	359	105	0,00	0,00	0	0	0
Receivables and Payables to the Central Bank	366	11	46 684 155 278,93	46 739 854 801,30	3 295	2 422	5 717
Normal Receivables and Reserves	434	12	144 969 664 574,67	144 299 204 59...	265 456	253 869	519 325
Receivables under Observation from Financial Subjects	4 027	13	0,00	0,00	0	0	0
Receivables below standard and provisions	6 420	14	0,00	0,00	0	0	0
Doubtful Debts	8 818	15	0,00	0,00	0	0	0

Figure 5. Example of OLAP report with a view of General Ledger as heterogeneous dimension with enabled filling of partial OLAP cube (shadowed row)

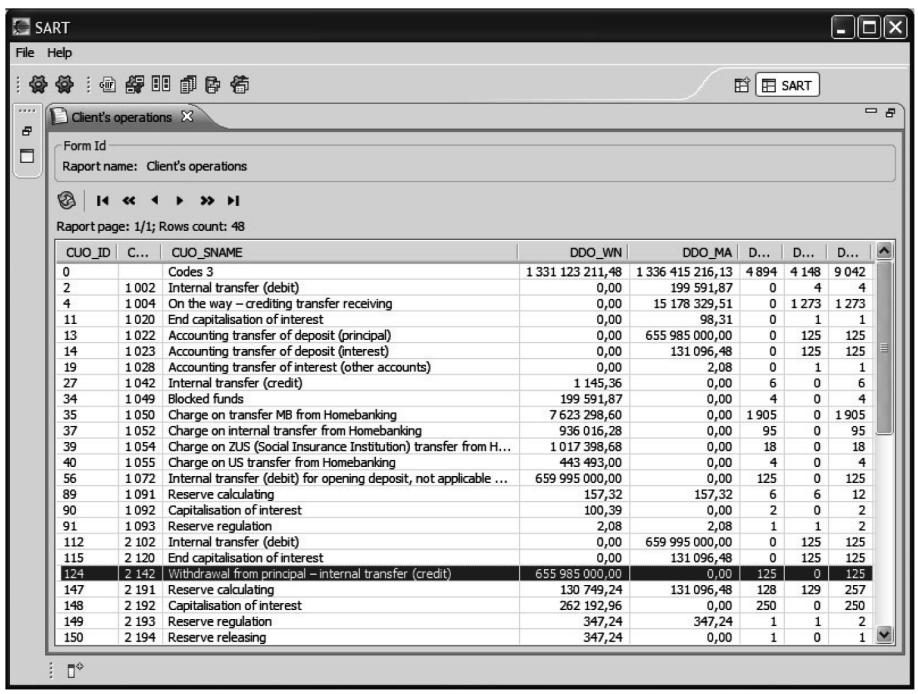
5. Advantages in AML systems – investigation of economic relations

So far the types of dimensions used in modelling the data warehouse were discussed as well as new types used in Analytical SQL. In this section the attention will be focused on the purpose of their introduction in Analytical SQL. One of the key requirements of the Regulation Act [Act, 2000; Directive 2005] is the application of security measures, and in particular the financial survey of economic relations between the bank and its client. This requirement says that the bank should know the exact profile of its clients, the nature of their activities and most importantly, to examine whether this information corresponds to the business reality.

In the SART system the investigation of economic relations is based on a data warehouse using a General Ledger, dimension of Chart of Accounts and dimension of operations codes. This approach only assures to carry out the advanced analysis of economic relations based on data already gathered in the bank information system. One of the core problems is the fact that the Chart of Accounts of General Ledger had to be modelled as a heterogeneous Analytical SQL dimension.

Figure 6 shows a report on the economic profile of one of the clients describing the types of his operations performed within a given period of time. The report was

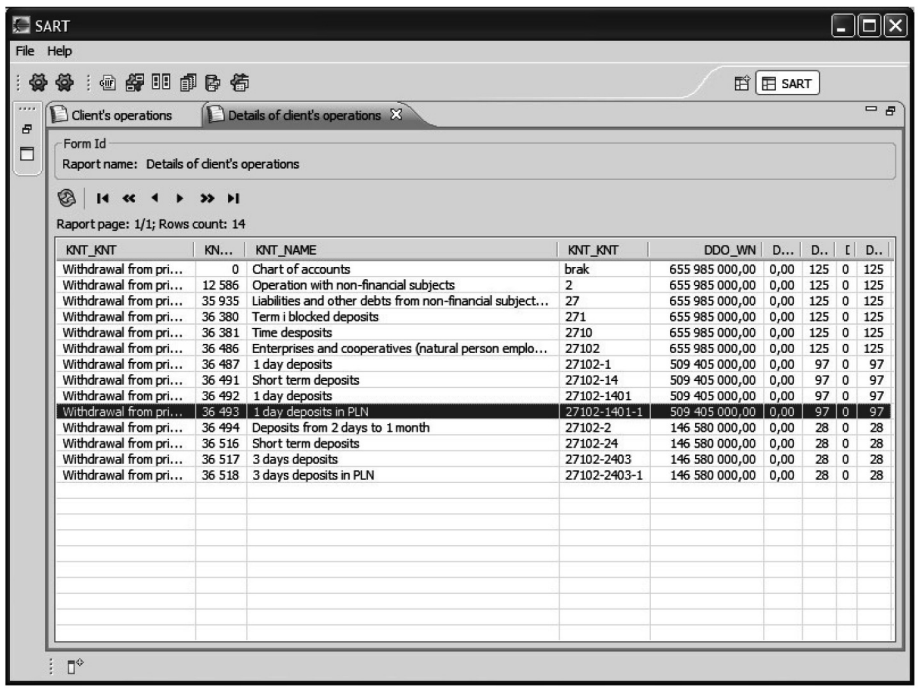
made by setting the view taking into consideration the operation codes and the parameters of the report by stating: Customer ID, time period (six months) and the position of the Chart of Accounts (all accounts the Chart of Accounts). In the report OLAP the aggregates have been used with respect to various types of operations and parameters defined in the query. It should be noted that the aggregation of all banking operations carried out by the customer in the long period gives us information about his typical behaviour. Hence, the case of operations performed by the client that are not listed in the above report may reveal unusual behaviour, and imply the increased risk of money laundering.



CUO_ID	C...	CUO_SNAME	DDO_WN	DDO_MA	D...	D...	D...
0		Codes 3	1 331 123 211,48	1 336 415 216,13	4 894	4 148	9 042
2	1 002	Internal transfer (debit)	0,00	199 591,87	0	4	4
4	1 004	On the way – crediting transfer receiving	0,00	15 178 329,51	0	1 273	1 273
11	1 020	End capitalisation of interest	0,00	98,31	0	1	1
13	1 022	Accounting transfer of deposit (principal)	0,00	655 985 000,00	0	125	125
14	1 023	Accounting transfer of deposit (interest)	0,00	131 096,48	0	125	125
19	1 028	Accounting transfer of interest (other accounts)	0,00	2,08	0	1	1
27	1 042	Internal transfer (credit)	1 145,36	0,00	6	0	6
34	1 049	Blocked funds	199 591,87	0,00	4	0	4
35	1 050	Charge on transfer MB from Homebanking	7 623 298,60	0,00	1 905	0	1 905
37	1 052	Charge on internal transfer from Homebanking	936 016,28	0,00	95	0	95
39	1 054	Charge on ZUS (Social Insurance Institution) transfer from H...	1 017 398,68	0,00	18	0	18
40	1 055	Charge on US transfer from Homebanking	443 493,00	0,00	4	0	4
56	1 072	Internal transfer (debit) for opening deposit, not applicable ...	659 995 000,00	0,00	125	0	125
89	1 091	Reserve calculating	157,32	157,32	6	6	12
90	1 092	Capitalisation of interest	100,39	0,00	2	0	2
91	1 093	Reserve regulation	2,08	2,08	1	1	2
112	2 102	Internal transfer (debit)	0,00	659 995 000,00	0	125	125
115	2 120	End capitalisation of interest	0,00	131 096,48	0	125	125
124	2 142	Withdrawal from principal – internal transfer (credit)	655 985 000,00	0,00	125	0	125
147	2 191	Reserve calculating	130 749,24	131 096,48	128	129	257
148	2 192	Capitalisation of interest	262 192,96	0,00	250	0	250
149	2 193	Reserve regulation	347,24	347,24	1	1	2
150	2 194	Reserve releasing	347,24	0,00	1	0	1

Figure 6. Report of client operations

Figure 7 presents the highlighted data from the report of the Figure 6 data record relating to the type of operation, "the payment" – an internal transfer DB (concerning the operation of transfer of funds from deposit account to another account of a customer via internal transfer within the bank). From this report we learn that such operations were carried out on two accounts: Deposits in PLN 1-day and 3-days. One can note from the report that such operations were carried out on two accounts: Deposits 1-day and 3-days in PLN. By applying specific OLAP extensions in Analytical SQL Server we have obtained a very rich and easy to use functionality allowing



KNT_KNT	KN...	KNT_NAME	KNT_KNT	DDO_WN	D...	D..	[	D..
Withdrawal from pri...	0	Chart of accounts	brak	655 985 000,00	0,00	125	0	125
Withdrawal from pri...	12 586	Operation with non-financial subjects	2	655 985 000,00	0,00	125	0	125
Withdrawal from pri...	35 935	Liabilities and other debts from non-financial subject...	27	655 985 000,00	0,00	125	0	125
Withdrawal from pri...	36 380	Term i blocked deposits	271	655 985 000,00	0,00	125	0	125
Withdrawal from pri...	36 381	Time desposits	2710	655 985 000,00	0,00	125	0	125
Withdrawal from pri...	36 486	Enterprises and cooperatives (natural person emplo...	27102	655 985 000,00	0,00	125	0	125
Withdrawal from pri...	36 487	1 day deposits	27102-1	509 405 000,00	0,00	97	0	97
Withdrawal from pri...	36 491	Short term deposits	27102-14	509 405 000,00	0,00	97	0	97
Withdrawal from pri...	36 492	1 day deposits	27102-1401	509 405 000,00	0,00	97	0	97
Withdrawal from pri...	36 493	1 day deposits in PLN	27102-1401-1	509 405 000,00	0,00	97	0	97
Withdrawal from pri...	36 494	Deposits from 2 days to 1 month	27102-2	146 580 000,00	0,00	28	0	28
Withdrawal from pri...	36 516	Short term deposits	27102-24	146 580 000,00	0,00	28	0	28
Withdrawal from pri...	36 517	3 days deposits	27102-2403	146 580 000,00	0,00	28	0	28
Withdrawal from pri...	36 518	3 days deposits in PLN	27102-2403-1	146 580 000,00	0,00	28	0	28

Figure 7. Report of client operations on accounts of the Chart of Account

customer relation management. It should also be noted that in addition to the obvious benefits of the data warehouse modelling facilities based on the Chart of Accounts, we have obtained in a natural way the possibilities to study relations with customers without any additional modifications of data structures. This last feature is extremely important because it allows definition and interpretation of the data warehouse structure by each group of users of SART (accountants, client, database administrators, the operators of AML).

6. Conclusions

In this article the three main models of data warehouse dimensions (classical and hierarchical, based on hierarchical tables) have been presented along with details of the advantages and disadvantages of each. Then it has shown the specific extension of Analytical SQL Server: hierarchical tables and vector type used to identify elements in hierarchical tables. Another issue raised in the article is the computing of Cartesian products OLAP based on Analytical SQL dimensions.

Theoretical considerations concerning the dimension modelling of data warehouse and computation of Cartesian products have been illustrated by examples extracted from the real-life data warehouse in connection with the requirements for

AML systems. This case study has exposed difficulties in modelling the data warehouses and provided concrete solutions implemented in the Analytical SQL Server

References

Act (2000), Act of 16th November, 2000, on preventing introducing into financial turnover properties derived from illegal or undisclosed sources and on counter-financing of terrorism, Dz.U.03.153.1505.

Adamson C. (2006), *Mastering Data Warehouse Aggregates: Solutions for Star Schema Performance*, Wiley, Indianapolis.

Directive (2005), Directive 2005/60/EC of the European Parliament and the Council of October 26, 2005. Commission Directive 2006/70/EC of August 1, 2006 laying down implementing measures for Directive 2005/60/EC of the European Parliament and the Council.

Inmon W.H. (2005), *Building the Data Warehouse*, Wiley, Indianapolis.

Kimball R., Ross M. (2002), *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modelling*, Wiley, Indianapolis.

Korczak J., Merchelski W., Oleszkiewicz B. (2008), A New Technological Approach to Money Laundering Discovery using Analytical SQL Server, [in:] *Advanced Information Technologies for Management – AITM 2008*, Eds. J. Korczak, H. Dudycz, M. Dyczkowski, Research Papers of the Wrocław University of Economics No. 35, Publishing House of the University of Wrocław, Wrocław.

Szirtes T. (2007), *Applied Dimensional Analysis and Modeling*, Elsevier, Burlington.

The Forty Recommendations of the Financial Action Task Force on Money Laundering (FATF-GAFI), http://www.fatf-gafi.org/