ZESZYT NAUKOWY
Wyższa Szkoła Zarządzania i Bankowości w Krakowie

Tomasz Libera, Piotr Ziuziański
The School of Banking and Management in Krakow
libera@wszib.edu.pl; piotrziuzianski@gmail.com

STRUCTURE CHARACTERISTICS OF DATA WAREHOUSE AND IMPLEMENTATION POSSIBILITIES FOR VARIOUS DIMENSION TYPES

Introduction

Organizations need tools to facilitate prompt and adequate decision-making processes that involve as little risk as possible. Effective decision-making can be achieved thanks to analytical tools with a reliable data collection. The function of such a data source is played in organizations by data warehouses which provide end-users with the indispensable data within the Business Intelligence system[1,2] . The development of a data warehouse is the first step in the creation of OLAP bases. The bases consist of the so called analytics cubes that provide users with a convenient and prompt access to business data. The understanding of the concept of Business Intelligence environment architecture and of the principles of data warehouse designing makes it possible to develop effective analytics platforms to support decision-making processes in organizations.

The objective of the article is to present theoretical determinants of data warehouse designing with the consideration of the dimensions classification and to present practical implementation of various types of dimensions with the application of the Microsoft software. In 2017- for the tenth time in a row - Microsoft was considered in the report of Gartner Research and Analysis Company (*Magic Quadrant for Business Intelligence and Analytics Platforms*[3,4,5]) to be the leader on the market of Business Intelligence and analytics platforms.

[1] Furmankiewicz M., Ziuziański P., Wdrażanie kokpitu menedżerskiego w ramach systemu BI w organizacji, „Przegląd Teleinformatyczny” 2014, No. 1-2 (37), p. 6.

[2] Ziuziański P., *Kokpit menedżerski jako efektywne narzędzie do wizualizacji danych w organizacji*, [in:] Zieliński Z.E. (ed.), *Rola informatyki w naukach ekonomicznych i społecznych. Innowacje i implikacje interdyscyplinarne*, Zeszyt 1/2014, Wydawnictwo Wyższej Szkoły Handlowej, Kielce 2014, p. 62.

[3] http://www.msbifun.pl/micrisoft/bi-magic-quadrant-gartnera/

[4] https://blogs.microsoft.com/blog/2017/02/16/microsoft-breaks-gartner-magic-quadrant-business-intelligence-analytics-platforms/

[5] Furmankiewicz J., Furmankiewicz M., Ziuziański P., *Implementation of business intelligence performance dashboard for the knowledge management in organization* [in:] Wolniak R. (ed.), *Zeszyty Naukowe Politechniki Śląskiej*. Seria: *Organizacja i Zarządzanie*, No. 82, Wydawnictwo Politechniki Śląskiej, Gliwice, 2015.

Furthermore, the aim of the article is to present a practical creation of junk dimension and the management of a rapidly changing monster dimension by minidimensions.

1. Concept of data warehouse structure

The term **data warehouse** can be interpreted either in a wider or narrower sense. In a wider sense, a data warehouse is an overall system whose task is not only to download data from company information resources but also to place the data in the new base and to share it with decision-makers. In a narrower sense, a data warehouse is a data base whose basic task is to store data in order to ensure its efficient availability with the aim to facilitate decision-making processes.[6] The article takes into consideration the latter definition.

Thus, a data warehouse is a coherent, subject-oriented collection of integrated, non-volatile historical data whose purpose is to support business decision-making processes in a company[7]. Data warehouses are most frequently loaded increasingly and abundantly[8]. A data warehouse is a relational data base that has a unique structure which differentiates two types of tables: fact tables and dimension tables. The process of data warehouse loading is referred to as the ETL process (extract, transform, load) and consists in extracting the data from sources, transforming it adequately and uploading to the data warehouse.

A fact table holds actual events in reality that are subject to analysis (e.g. the sales volume). A dimension table includes reference data by which analyses are conducted (e.g. time, product). It is customary to use prefix Dim (for dimension) and Fact (for fact tables) [9] in a data warehouse. This helps to organize the structure of the data warehouse.[10]

A data warehouse can adopt different relational structures depending on the complexity of the business reality being modelled. Three basic data warehouse schema can be distinguished: star, snowflake and fact constellation. The characteristics of the data warehouse logical schema are given in Fig.1.

[6] Januszewski A., *Funkcjonalność informatycznych systemów zarządzania. T. 2, Systemy Business Intelligence*, Wydawnictwo Naukowe PWN, Warszawa 2008, p. 21.

[7] Olszak C.M., *Tworzenie i wykorzystanie systemów Business Intelligence na potrzeby współczesnej organizacji*, Wydawnictwo Akademii Ekonomicznej, Katowice 2007, p. 76.

[8] Nycz M., Smok B., *Jak hurtownia danych może wspomagać menedżera w pracy* [in:] Pańkowska M., Porębska-Miąc T., Sroka H. (ed.), *Systemy Wspomagania Organizacji SWO 2008*, Wydawnictwo Akademii Ekonomicznej, Katowice 2008, p. 432.

[9] Januszewski A., *Funkcjonalność informatycznych systemów* … op. cit., p. 23.

[10] Nycz M., Smok B., *Zastosowanie narzędzi etl w hurtowni danych* [in:] Sroka H., Porębska-Miąc T. (ed.), *Systemy Wspomagania Organizacji SWO 2004*, Wydawnictwo Akademii Ekonomicznej, Katowice 2004, p. 333.

Figure 1. Logical schema of data warehouse

Star schema

Consists of a fact table and the directly related dimensions table. Although the simplicity results in numerous benefits concerning easy development of effective aggregation queries, it also causes a significant abundance of data due to a high degree of the structure denormalization.

Snow *flake schema*

Is developed as the result of a partial normalization of the star schema dimension tables. As a rule, the snowflake schema involves a lower effectiveness of aggregation queries (there are more tables to aggregate when reading the data) , while the processes that load the warehouse (EFL) are relatively less complex.

Fact constellation schema

Is considered a third schema (apart from the star and snowflake schemas) in which several fact tables share the same dimension.

Source: Authors’ research based on: Chodkowska-Gyurics A., *Hurtownie danych. Teoria i praktyka*, Wydawnictwo Naukowe PWN SA, Warszawa 2014, pp. 179-181

2. Basic classification of dimensions

Following a review of the literature on the subject, online sources and their own experience, the authors present a classification of dimensions that is divided into two sections: the basic section and the supplementary one.

One of the basic classification criteria dimension is the type of the data stored, i.e. the content of a particular dimension in a data warehouse. Thus, one can distinguish such dimensions as conformed, junk, degenerate and role playing, which is presented in Table 1. One dimension can be classified into several categories. Dimension *Data* is a universal dimension but it may also play numerous roles in a multi-dimension model. Degenerate dimension *Age* that is given as an integer in a fact table may be transformed together with other customer’s attributes to a junk dimension.

Table 1. Basic classification of dimensions – dimensions by content

Dimensions by content (kind of data stored)		
Type	**Characteristics**	**Example**
conformed	Dimension that maintains the same significance although remains in relationship with numerous facts	Dimension *Data*; a particular date is explicit for all facts; it provides the information on the date of the occurrence of a particular fact.
junk	An artificially created dimension that is a combination of various flags and indices with an insignificant number of states.	Dimension *Customer* that constitutes a combination of several customer's features (e.g. gender, age group, type: individual/business).
degenerate	A particular case of data that constitutes a dimension but is stored in fact tables. This dimension does not possess attributes.	Dimension *Age*, which is presented in a sales facts table as an integer informing about customer's age. It may also be the *Order Number* in a fact table.
role-playing	Dimension that can hold various significance depending on the context. In a physical model, the dimension is represented as a single object, while in a logical model, the object is given by several independent dimensions.	Dimension *Data* can be used several times in a sales facts table which can include references to the dimension *Data* in the context of the sales date, the order shipment date and the order delivery date.

Source: Authors' research based on Chodkowska-Gyurics A., *Hurtownie danych*... op. cit., pp. 33-41; Kimball R., Ross M., *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*, John Wiley & Sons, Indianapolis 2013, pp. 47, 51.

The other crucial classification criterion of dimensions is the management type of changes that occur in dimensions. This is directly related to the way the data is loaded to the warehouse. One can distinguish fixed, slowly and rapidly changing dimensions. The characteristics of dimensions together with the examples is given in table 2.

Table 2. Basic classification of dimensions – dimensions by the management of changes (ways of loading)

Dimensions by the management of changes (the way of loading)		
Type	**Characteristics**	**Example**
fixed	Dimensions are constant throughout the whole period of the warehouse existence as regards the number of lines and the content.	Dimension *Data*, which may be loaded at the stage of the data warehouse implementation.
slowly changing	Dimensions where the values of particular attributes may be subject to slow/rapid changes over time.	The elements of the dimension *Product* may change over time, e.g. the name or colour.
rapidly changing	The determination of the pace of change depends on the designer.	

Source: Authors' research based on Chodkowska-Gyurics A., *Hurtownie danych*... op. cit., pp. 33-41; Kimball R., Ross M., *The Data Warehouse* ...op.cit.p147 .

In the case of changing dimensions, versioning of selected attributes can be applied. This is a fairly broad issue that is precisely characterized by Kimball and Ross.[11]

[11] Kimball R., Ross M., *The Data Warehouse*… op.cit., p. 53-56

3. Additional classification of dimensions

The literature on the subject also includes classifications of dimensions with regards to various criteria.

Table 3. Additional classification of dimensions

Additional classification of dimensions		
Criterion	**Type**	**Characteristics**
Growth pace	*slowly growing*	Standard dimension where the number of elements increases slowly.
	rapidly growing	Dimension where the number of elements increases relatively rapidly.
Structure	Standard, for the *star* structure	Dimension related to a star schema of a data warehouse. Joined with a relation with a fact table only.
	hierarchical: - *parent-child* - *snowflake*	Dimension joined with a relation with another dimension table (snowflake warehouse structure) or dimension with the relations with itself (a parent-child type relation).
Necessity of implementation	*required*	Dimension that is indispensable for the data warehouse to operate.
	optional	Warehouse dimension that expands its data scope that does not affect its basic functionality.
Volume of data stored	average	Relatively average dimension as regards its width (number of columns) and – first of all – its length (number of lines).
	monster	A very big dimension as regards the number of columns and lines.

	minidimension	Used to manage rapidly changing monster dimensions. It constitutes a selected part of the monster dimension that is frequently subject to change.
Application in fact constellation	*not shared*	Used by one fact table only.
	shared	Used by more than one fact table.
Dependence with other dimensions	*dependent*	Dimension that is dependent on another dimension (regards a snowflake schema).
	independent	Dimension is joined only to the fact table; independent of other dimensions.
Data quality guarantee	*dirty*	Dimensions with data without quality guarantee, with a high error margin.
	clean	Dimensions with cleaned data and narrow error margin.
Write ability	*read only*	Assumption of the constancy of dimension data without the opportunity to add new records or edit the existing ones.
	write enabled	Assumption of the data change with the possibility to add new lines and modify the existing ones.
Distance to fact tables	*primary*	Dimensions that are directly joined to a fact table are referred to as *primary*. Dimensions that are joined to a fact table through an additional dimension table are referred to as *secondary*. In the cases when one more table is used to join a fact, the dimension is referred to as *tertiary*.
	secondary	
	tertiary	
Function	*time*	Dimension indispensable in every data warehouse – provides time.
	data mining	Special dimension with a parent-child type hierarchy which is based on relations that are revealed in the course of data mining as opposed to a typical dimension where hierarchies are defined by the user.

	demographic	Dimension including demographic data. It can be developed e.g. as the result of the division of a rapidly growing monster dimension *Customer*.
	data quality	Some authors suggest adding such special dimension to describe every record in the fact table with regard to data quality. Exemplary values of such a dimension are: *normal value, out-of-bounds value, unlikely value, verified value, unverified value, uncertain value.*
	multi value	Dimension is a bridge between tables that shows many-to-many relationship in the model.

Source: Authors' research based on : Kimball R., Ross M., *The Data Warehouse*… op.cit., www.oracledbasupport.co.uk/dwh-terms/, , stackoverflow.com/questions/8184214/what-are-the-types-of-dimension-tables-in-star-schema-design, ramprapanna.wordpress.com/2014/11/08/dimension-types/, www.geekinterview.com/question_details/17677, http://de.dwhwiki.info/analyse/data-mining-dimension

It should be pointed out that frequently the dimension types are closely correlated to the logical model that is used in a data warehouse. This particularly refers to such criteria as structures, distances from fact tables and the occurrence of relationships with other dimensions.

Moreover, the following dimension types can be specified

- Shrunk dimension
  - In the cases of dimensions with hierarchies the concept of shrunk dimension may appear that constitutes a subset of attributes applied to a higher level of summary.[12]
- Inferred dimension
  - While loading fact tables, a dimension may not be ready. One solution to the problem is to generate a surrogate key with a null value for all other attributes. This should be technically referred to as an inferred member but is frequently called an inferred dimension[13]
- Heterogeneous dimension

[12] http://www.kimballgroup.com/2011/08/design-tip-137-creating-and-managing-shrunken-dimensions/

[13] https://social.technet.microsoft.com/Forums/en-US/2cd9bb62-8b94-46f3-b2fc-3cd7147f57b9/why-some-many -different-types-of-dimensions?forum=sqldatawarehousing

    - Heterogeneous dimensions are a variation of one dimension (e.g. dimension Product has attributes common both to food and household products.) In order to avoid numerous empty fields, variations of a particular dimension are created[14] .
- Reference dimension
    - A reference dimension relationship between a cube dimension and a measure dimension exists when the main dimension key is joined indirectly to a fact table through a key in another dimension in a snowflake schema design.[15]
- Outriggers
    - A dimension can contain a reference to another different dimension table, e.g. BankAccount dimension can reference to a separate dimension that represents the account opening date. The latter dimension is referred to as the outrigger dimension. According to Kimball, they are permissible but should be used sparingly. In most cases, the correlations between dimensions should be demoted to a fact table, where both dimensions are represented as separate foreign keys. [16]
- Flat dimension
    - A dimension without hierarchies and levels. [17]

4. Time dimension

The definition of a data warehouse states clearly that time dimension is an indispensable element of every data warehouse. Thus, with reference to the classification given in the previous chapter, this dimension is required. Time dimension can be generated manually for a data warehouse by means of T-SQL, which frequently applies the so called *Common Table Expression*, (CTE). A sample code together with the result is given in Fig.1.

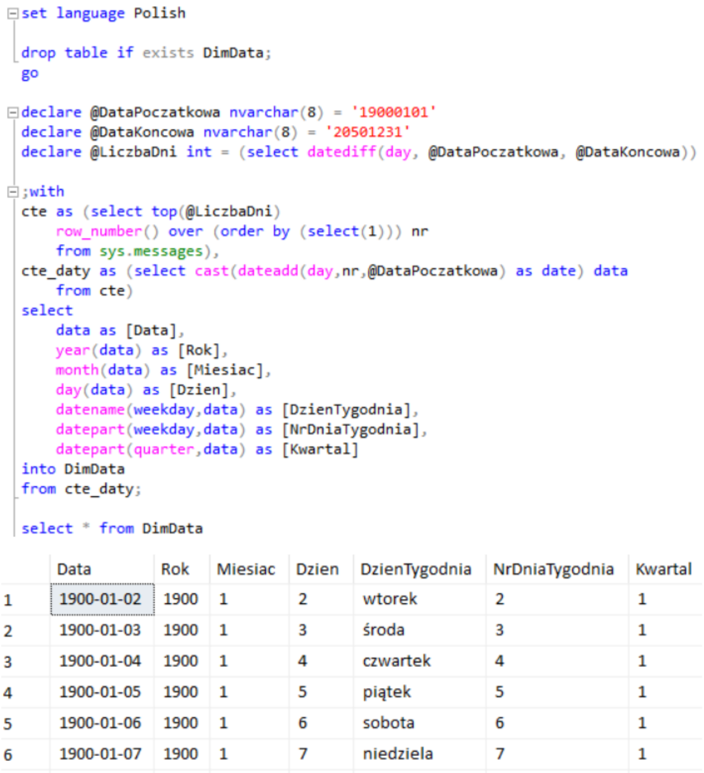
[14] https://techbusinessintelligence.blogspot.com/2014/09/junk-heterogeneous-and-degenerate.html

[15] https://msdn.microsoft.com/en-us/library/ms175669.aspx

[16] http://www.kimballgroup.com/wp-content/uploads/2013/08/2013.09-Kimball-Dimensional-Modeling-Techniques11.pdf

[17] Januszewski A.: *Funkcjonalność informatycznych systemów* … op. cit., p. 117.

Figure 1. Sample code to generate time dimension with the application of CTE



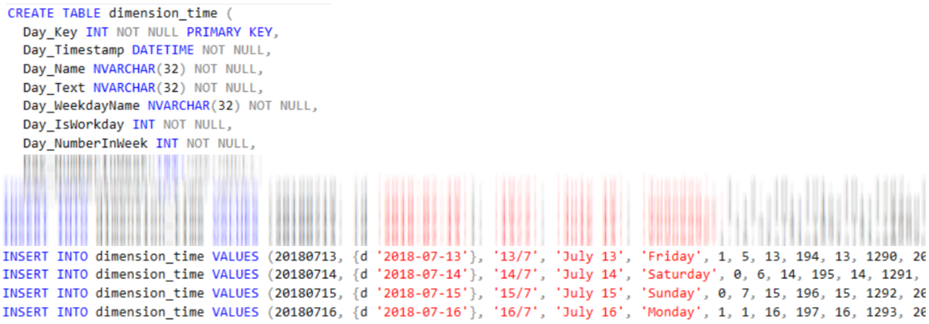
```
set language Polish

drop table if exists DimData;
go

declare @DataPoczatkowa nvarchar(8) = '19000101'
declare @DataKoncowa nvarchar(8) = '20501231'
declare @LiczbaDni int = (select datediff(day, @DataPoczatkowa, @DataKoncowa))

;with
cte as (select top(@LiczbaDni)
    row_number() over (order by (select(1))) nr
    from sys.messages),
cte_daty as (select cast(dateadd(day,nr,@DataPoczatkowa) as date) data
    from cte)
select
    data as [Data],
    year(data) as [Rok],
    month(data) as [Miesiac],
    day(data) as [Dzien],
    datename(weekday,data) as [DzienTygodnia],
    datepart(weekday,data) as [NrDniaTygodnia],
    datepart(quarter,data) as [Kwartal]
into DimData
from cte_daty;

select * from DimData
```

	Data	Rok	Miesiac	Dzien	DzienTygodnia	NrDniaTygodnia	Kwartal
1	1900-01-02	1900	1	2	wtorek	2	1
2	1900-01-03	1900	1	3	środa	3	1
3	1900-01-04	1900	1	4	czwartek	4	1
4	1900-01-05	1900	1	5	piątek	5	1
5	1900-01-06	1900	1	6	sobota	6	1
6	1900-01-07	1900	1	7	niedziela	7	1

Source: Authors' research

Ready-made generators that create scripts and are available on the Internet can also be used here. A good example is the *National Danish Center for Advanced Calculations* (NDCAC) which facilitates a precise determination of the output data format in the form of a script that creates a table and loads it with data.[18] Figure 2 presents an example of a script created with the application of this generator.

Figure 2. Fragment of a sample script generating the structure and content of time dimension



```
CREATE TABLE dimension_time (
  Day_Key INT NOT NULL PRIMARY KEY,
  Day_Timestamp DATETIME NOT NULL,
  Day_Name NVARCHAR(32) NOT NULL,
  Day_Text NVARCHAR(32) NOT NULL,
  Day_WeekdayName NVARCHAR(32) NOT NULL,
  Day_IsWorkday INT NOT NULL,
  Day_NumberInWeek INT NOT NULL,

INSERT INTO dimension_time VALUES (20180713, {d '2018-07-13'}, '13/7', 'July 13', 'Friday', 1, 5, 13, 194, 13, 1290, 2
INSERT INTO dimension_time VALUES (20180714, {d '2018-07-14'}, '14/7', 'July 14', 'Saturday', 0, 6, 14, 195, 14, 1291,
INSERT INTO dimension_time VALUES (20180715, {d '2018-07-15'}, '15/7', 'July 15', 'Sunday', 0, 7, 15, 196, 15, 1292, 2
INSERT INTO dimension_time VALUES (20180716, {d '2018-07-16'}, '16/7', 'July 16', 'Monday', 1, 1, 16, 197, 16, 1293, 2
```

Source: Authors' research based on http://www.regnecentralen.dk/time_dimension_generator.html

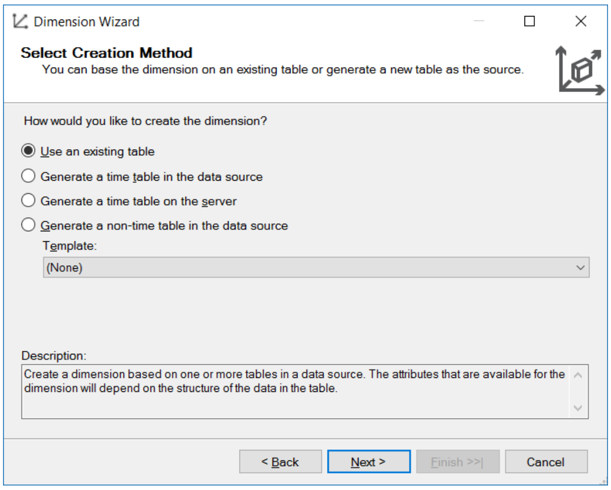
[18] http://www.regnecentralen.dk/time_dimension_generator.html

One should bear in mind that a day, referred to as DimDate is the most common level of detail in time dimension. Moreover, the time of a particular event is of significance. In such cases a separate hour dimension (*DimHour*) is created, which may be useful for instance in a sales data warehouse as it facilitates the analysis with regard to time of day which constitutes a dimension hierarchy. The possession of such business knowledge can change the way the sales is conducted.

1. Implementation possibilities of various dimension types in Microsoft SQL Server

The Microsoft SQL Server offers directly or indirectly the support to dimensions of different types. When creating a degenerate dimension in the Microsoft SQL Server environment, a dimension creator from Visual Studio can be used (*Analysis Services and Data Mining Project* design type). The creator is given in Figure 3.

Figure 3. Creation of a degenerate dimension for OLAP Microsoft SQL Server OLAP cube



Dimension Wizard

Select Creation Method

You can base the dimension on an existing table or generate a new table as the source.

How would you like to create the dimension?

● Use an existing table

○ Generate a time table in the data source

○ Generate a time table on the server

○ Generate a non-time table in the data source

Template:

(None)

Description:

Create a dimension based on one or more tables in a data source. The attributes that are available for the dimension will depend on the structure of the data in the table.

Source” Authors’ research

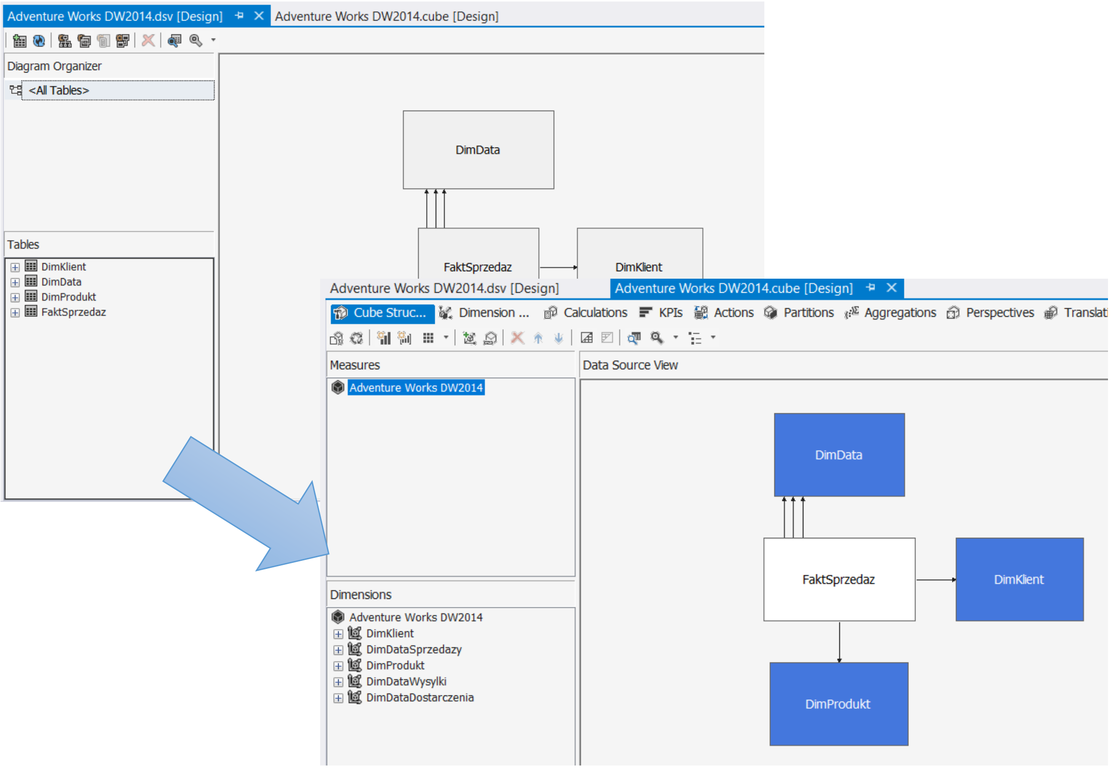
The creation of a degenerate dimension consists in selecting a fact table and columns that will constitute the dimension. The degenerate dimension derived from a fact table does not possess any unique attributes that would be visible in the dimension metadata .[19]

It should be added that the creation of a **multi-use dimension** consists in a single addition of a particular data warehouse dimension table to the data source view for a cube and a

[19] Cameron S., *Microsoft SQL Server 2008 Analysis Services Krok po kroku*, Microsoft Press, Warszawa 2009, p. 203.

subsequent use of the dimension as many times in the cube structure editor as the number of roles played by the dimension. Thus, for instance, dimension *Data* for a sales cube can play the role of the sales, shipping and delivery dates. The implementation of such a solution is given in Figure 4.

Figure 4. Implementation possibilities of a multi-use dimension for Microsoft SQL Server OLAP cube

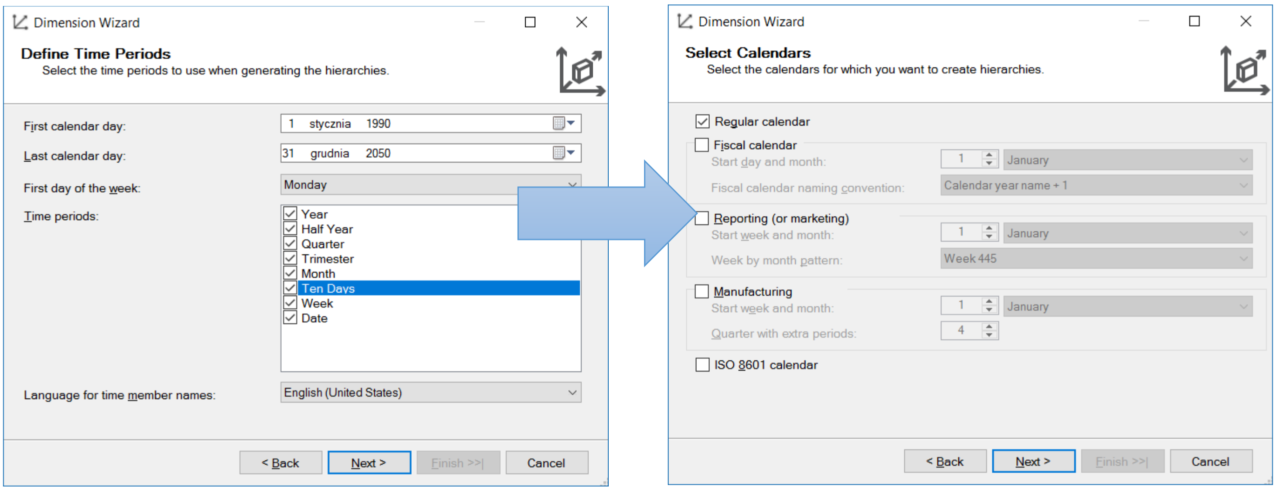


Source: Authors' research

It should be added that Analysis Services also manages a **many-to-many relationship**. The implementation method is the same as in the case of a reference dimension. The management is conducted through the application of two dimensions and two measure groups.

Apart from the above mentioned possibility to implement a degenerate dimension, the Microsoft software provides the opportunity to apply the creator in the development of a time dimension, which is given in Fig.5.

Figure 5. Possibilities to generate a time dimension for Microsoft SQL Server OLAP cube



Dimension Wizard

Define Time Periods
Select the time periods to use when generating the hierarchies.

First calendar day: 1 stycznia 1990
Last calendar day: 31 grudnia 2050
First day of the week: Monday
Time periods:
☑ Year
☑ Half Year
☑ Quarter
☑ Trimester
☑ Month
☑ Ten Days
☑ Week
☑ Date

Language for time member names: English (United States)

< Back | Next > | Finish >>| | Cancel

Dimension Wizard

Select Calendars
Select the calendars for which you want to create hierarchies.

☑ Regular calendar
☐ Fiscal calendar
Start day and month: 1 January
Fiscal calendar naming convention: Calendar year name + 1
☐ Reporting (or marketing)
Start week and month: 1 January
Week by month pattern: Week 445
☐ Manufacturing
Start week and month: 1 January
Quarter with extra periods: 4
☐ ISO 8601 calendar

< Back | Next > | Finish >>| | Cancel

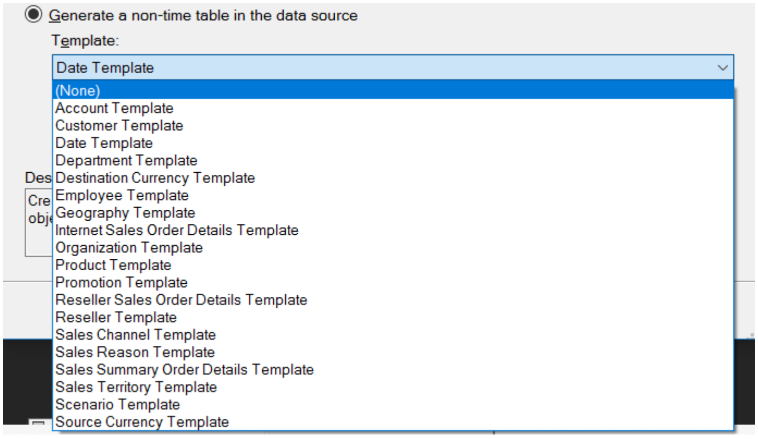
Source” Authors’ research

Visual Studio 2015 creator makes it possible to:

- determine the first and the last date of the dimension;
- select the required time period (year, halfyear, quarter, trimester, month, decade, week, day);
- select the language of the names of dimension elements (English, French, Japanese and German are available);
- take into consideration several hierarchies, including regular, financial, report production and ISO 8601 calendars.

Moreover, there is a possibility to create a dimension on the basis of predefined templates. A list of templates that are available is given in Fig.6.

Figure 6. List of dimension templates for Visual Studio 2015



● Generate a non-time table in the data source
Template:
Date Template
(None)
Account Template
Customer Template
Date Template
Department Template
Destination Currency Template
Employee Template
Geography Template
Internet Sales Order Details Template
Organization Template
Product Template
Promotion Template
Reseller Sales Order Details Template
Reseller Template
Sales Channel Template
Sales Reason Template
Sales Summary Order Details Template
Sales Territory Template
Scenario Template
Source Currency Template

Source: Authors’ research

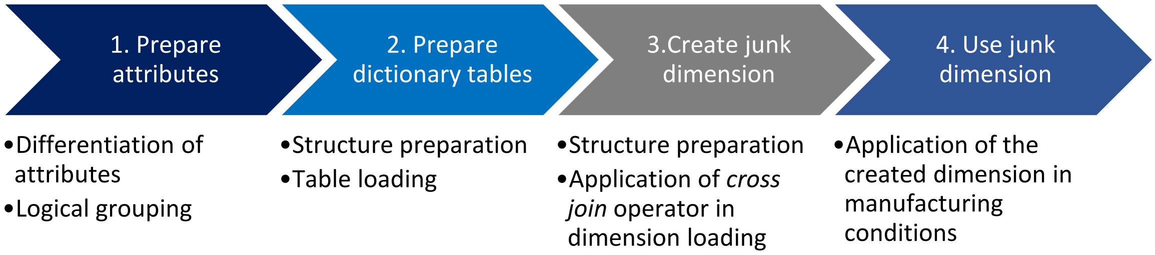
The Microsoft SQL Server Analysis Services provides the possibility to implement reference dimension i.e. a dimension that refers to a different dimension, in OLAP cube. The first dimension, which is combined directly with the fact table, includes a foreign key to another dimension. The application of a reference dimension can be useful in two cases:

- sharing a dimension in a snowflake schema;
- sharing dimension attributes that are stored in various data sources.

2. Creating a junk dimension – a case study

The implementation of a junk dimension in a Microsoft SQL Server environment is possible; however, it requires an adequate preparation of data. The development process of a junk dimension can be divided into several steps, which is given in Fig.7.

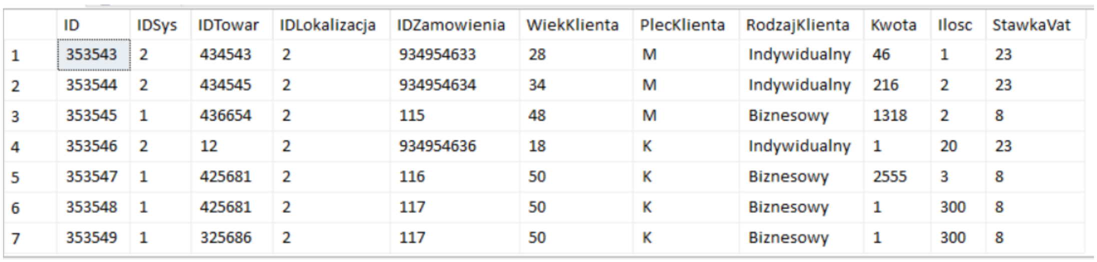
Figure 7. Steps in the development of a junk dimension


1. Prepare attributes
2. Prepare dictionary tables
3.Create junk dimension
4. Use junk dimension
•Differentiation of attributes
•Logical grouping
•Structure preparation
•Table loading
•Structure preparation
•Application of *cross join* operator in dimension loading
•Application of the created dimension in manufacturing conditions

Source: Authors' research

The first step in the development of a junk dimension is to analyze the attributes from the fact table that could form the dimension. These are the degenerate dimensions which were characterized above. If the attributes can be grouped in topical sets, the work on the development of vocabulary tables can be started. The tables consist of only two fields: the identifier and the attribute value. The next step is to load the dictionary tables. This will enable the creation of a junk dimension which will constitute a combination of all possible values in the dictionary tables (a cross join operator in T-SQL). In order to apply the junk dimension, an identifier has to be used that consists of the subsequent dictionary table identifiers.

Figure 8. Sample fact table with degenerate dimensions



	ID	IDSys	IDTowar	IDLokalizacja	IDZamowienia	WiekKlienta	PlecKlienta	RodzajKlienta	Kwota	Ilosc	StawkaVat
1	353543	2	434543	2	934954633	28	M	Indywidualny	46	1	23
2	353544	2	434545	2	934954634	34	M	Indywidualny	216	2	23
3	353545	1	436654	2	115	48	M	Biznesowy	1318	2	8
4	353546	2	12	2	934954636	18	K	Indywidualny	1	20	23
5	353547	1	425681	2	116	50	K	Biznesowy	2555	3	8
6	353548	1	425681	2	117	50	K	Biznesowy	1	300	8
7	353549	1	325686	2	117	50	K	Biznesowy	1	300	8

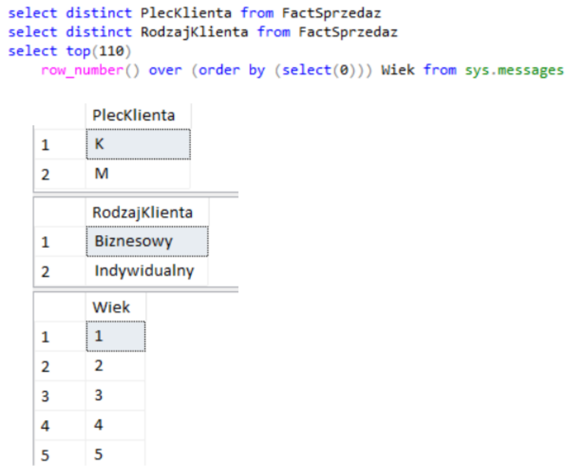
Source: Authors' research

Assuming that the fact table contains fields given in Fig.8, one can see such topical degenerate dimensions as *WiekKlienta (CustomerAge)*, *PlecKlienta (CustomerGender)*, *RodzajKlienta (CustomerType)*.

The preparation of dictionary tables that will be the basis for a junk dimension can be conducted through the development of a list of unique attributes (e.g. for *RodzajKlienta/ CustomerType*) or through the use of the general knowledge (e.g. for *WiekuKlienta / CustomerAge* the age range will be 0-110 as the age cannot have a negative value and the number of people 110+ is marginal). The selection of the method used to generate dictionary tables depends on the designer and his/her knowledge regarding the data. If it is certain that the facts include all the states of a given attribute, it is sufficient to prepare a query that will return a list of unique occurrences. If there is a risk of omitting a state, an appropriate dictionary should be developed to the best of the designer's general/business knowledge.

Figure 9 presents a method of developing dictionaries based on unique values and the possessed knowledge (the last query *select* makes it possible to generate a sequence of integers from 1 to 110).

Figure 9. Dictionaries developed on the basis of a fact table and the possessed knowledge



```
select distinct PlecKlienta from FactSprzedaz
select distinct RodzajKlienta from FactSprzedaz
select top(110)
    row_number() over (order by (select(0))) Wiek from sys.messages
```

	PlecKlienta
1	K
2	M

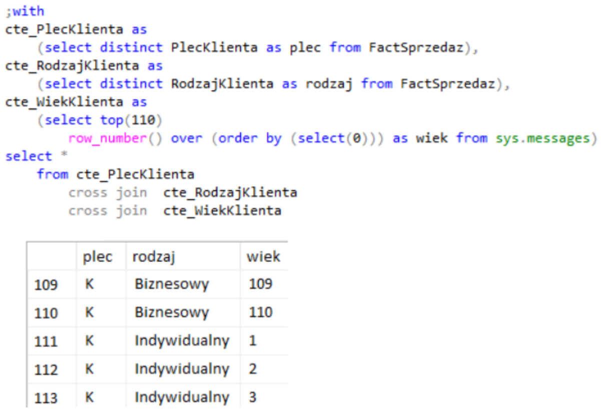
	RodzajKlienta
1	Biznesowy
2	Indywidualny

	Wiek
1	1
2	2
3	3
4	4
5	5

Source: Authors' research

Another step is to create all possible combinations of gender, type and age of customers. The necessary code is given in Fig.10.

Figure 10. The application of *cross join* operator to develop all possible combinations in degenerate dimensions



```
;with
cte_PlecKlienta as
    (select distinct PlecKlienta as plec from FactSprzedaz),
cte_RodzajKlienta as
    (select distinct RodzajKlienta as rodzaj from FactSprzedaz),
cte_WiekKlienta as
    (select top(110)
        row_number() over (order by (select(0))) as wiek from sys.messages)
select *
    from cte_PlecKlienta
        cross join  cte_RodzajKlienta
        cross join  cte_WiekKlienta
```

	plec	rodzaj	wiek
109	K	Biznesowy	109
110	K	Biznesowy	110
111	K	Indywidualny	1
112	K	Indywidualny	2
113	K	Indywidualny	3

Source" Authors' research

SQL *cross join* operator is a Cartesian product of three sets (tables). In the presented case, 440 elements will be returned: 2 (*CustomerGender* states) x 2 (*CustomerType* states) x 110 (*CustomerAge* states). Such a dimension can be loaded once in the course of the data warehouse implementation and it remains static. When loading sales facts, an adequate identifier should be taken of the junk dimension in line with the attribute values.

3. Concept of minidimensions for managing rapidly changing monster dimension – a case study

Mini-dimension is a subset of attributes of a monster dimension (one that contains a significant number of lines and/or columns) that are frequently modified. For the purpose of tracking the changes in rapidly slowly growing dimensions, in the case of the change in value of any of the columns, a new line is added as a new version of data. This is the so called type-2 technique of managing the changes in slowly changing dimensions. This implies that in the case of monster dimensions any change results in an over-sized storage of values from the columns that were not subject to change. The above mechanism has an impact on the volume of the data stored and also on the effectiveness of the warehouse loading process.

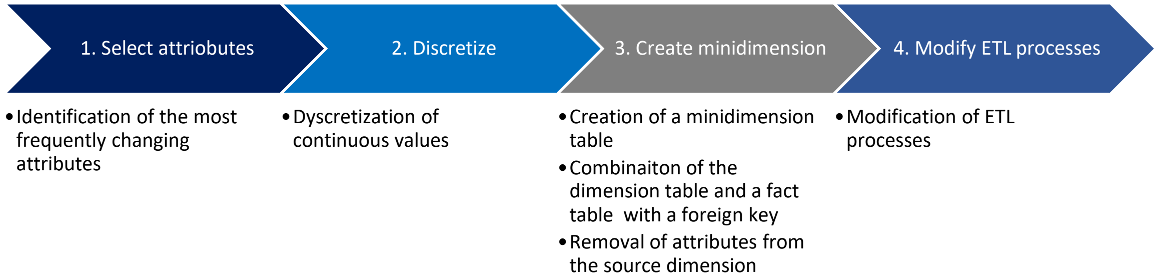
The above problem can be solved by extracting unique combinations of the changing columns (or the ones that change most frequently) to another column and by correlating them directly with the fact table. Consequently, the basic dimension will contain significantly fewer versions of data.

Minidimension implementations that are frequently encountered are the *Klient/Customer* dimension that contains constant columns (name, birthdate) and the *KlientDemografia/ CustomerDemography* that includes information about the age, number of children, maritial status, etc.[20] The creation of a minidimension can be conducted in several steps, which is given in Figure 11.

The first step is to select from the base dimension the attributes that frequently change. In the cases when all changing dimensions are selected, at the end of the process, columns that are specific for type-2 slowly changing dimension (*CzyAktualny, Od, Do*) */ (IsValid. From*, *To)* can be completely removed from the dimension table.

[20] Kimball R., Ross M., *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*, John Wiley & Sons, Indianapolis 2013, p. 260.

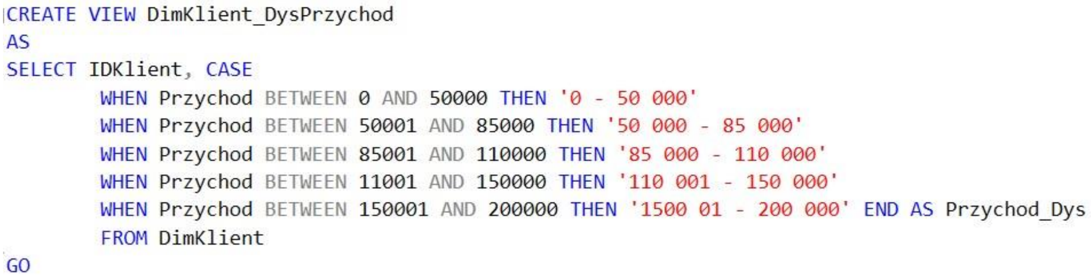
Figure 11. Steps to create a minidimension


1. Select attriobutes
2. Discretize
3. Create minidimension
4. Modify ETL processes
•Identification of the most frequently changing attributes
•Dyscretization of continuous values
•Creation of a minidimension table
•Combinaiton of the dimension table and a fact table with a foreign key
•Removal of attributes from the source dimension
•Modification of ETL processes

Source:Authors' research based on https://dylanwan.wordpress.com/bi-and-olap-glossary/minidimension/

In order to limit the number of lines for continuous attributes (wiek/age, przychód/income) discretization can be applied before the creation of a minidimension. Figure 12 presents an example where continuous values can be changed to discrete values of the sample *Przychód/Income* attribute

Fig.12. Discretization of continuous values



```
CREATE VIEW DimKlient_DysPrzychod
AS
SELECT IDKlient, CASE
        WHEN Przychod BETWEEN 0 AND 50000 THEN '0 - 50 000'
        WHEN Przychod BETWEEN 50001 AND 85000 THEN '50 000 - 85 000'
        WHEN Przychod BETWEEN 85001 AND 110000 THEN '85 000 - 110 000'
        WHEN Przychod BETWEEN 11001 AND 150000 THEN '110 001 - 150 000'
        WHEN Przychod BETWEEN 150001 AND 200000 THEN '1500 01 - 200 000' END AS Przychod_Dys
        FROM DimKlient
GO
```

Source: Authors' research

Another step is to create a minidimension table that stores a Cartesian product (every-to-every) of the selected attributes or their discrete representation. The new table should be completed with values that originate from the base table and its main key is constituted by a new, automatically numbered numeric column.

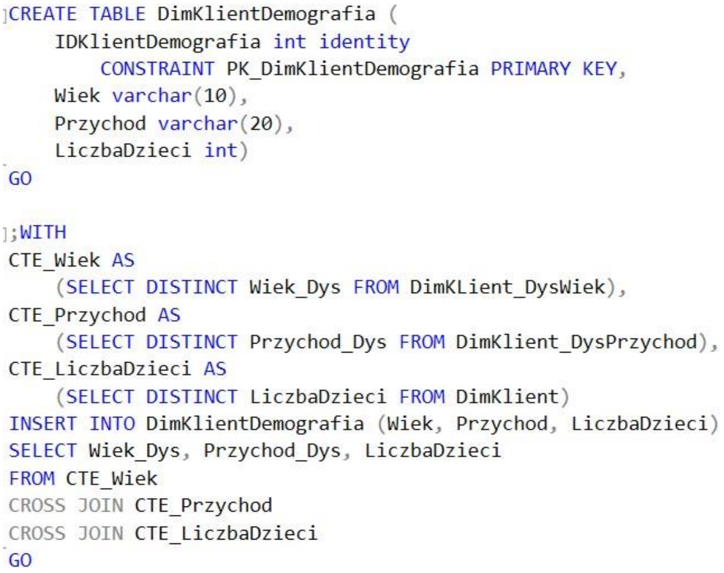
Figure 13 presents a sample structure of a minidimension table that stores age, income, the number of customer's children and the instruction to complete the table with data on the basis of the base dimension. The first two attributes are given in a discrete form and consequently there is a necessity to apply the pattern given in figure 12.

Then, the minidimension should be combined with the fact table. Having completed the fact table with adequate values in order to facilitate the analysis of data in the warehouse on the basis of the minidimension, the columns that were selected from the dimension base table in the first step can be removed.

The final step is to modify processes that load the data warehouse (ELT) with the consideration of the following:

- add a new line to the minidimension in the case when values appear that are beyond the defined ranges;
- check the minidimension key when recording every new transaction (on the basis of the customer-related attributes).

Figure 13. Structure of a minidimension table and the script



```
CREATE TABLE DimKlientDemografia (
    IDKlientDemografia int identity
        CONSTRAINT PK_DimKlientDemografia PRIMARY KEY,
    Wiek varchar(10),
    Przychod varchar(20),
    LiczbaDzieci int)
GO

;WITH
CTE_Wiek AS
    (SELECT DISTINCT Wiek_Dys FROM DimKLient_DysWiek),
CTE_Przychod AS
    (SELECT DISTINCT Przychod_Dys FROM DimKlient_DysPrzychod),
CTE_LiczbaDzieci AS
    (SELECT DISTINCT LiczbaDzieci FROM DimKlient)
INSERT INTO DimKlientDemografia (Wiek, Przychod, LiczbaDzieci)
SELECT Wiek_Dys, Przychod_Dys, LiczbaDzieci
FROM CTE_Wiek
CROSS JOIN CTE_Przychod
CROSS JOIN CTE_LiczbaDzieci
GO
```

Source: Authors' research

(Wiek/Age, Przychód/Income/ KlientDemografia/Customer Demography, Przychód/Income, LiczbaDzieci/Number of Children)

It is worth noting that the above minidimension does not allow for recording changes within the scope of the given attributes prior to a subsequent transaction in the fact table. Thus, in the example given above, any changes regarding customer's data about the age, the number of children and the income cannot be registered in the data warehouse unless the customer places another order.

Conclusions

On the basis of the literature on the subject and their own knowledge and experience, the authors reviewed and classified dimensions of various types of data warehouses which was the main aim of the article. It should be emphasized that the research in this field remains open and should be continued; the presented classification should be extended by additional criteria so that the knowledge on multidimensional modelling should be systematized. The authors presented also selected implementation opportunities of various types of dimensions in the Microsoft SQL Server Analysis Services environment.

The objectives concerning a practical creation of a junk dimension and minidimensions to service a rapidly changing monster dimension were accomplished in two separate chapters of the article in the form of case studies. The presented techniques make it possible to solve problems that are common when building data warehouses.

Literatura

Bibliography

[1] Cameron S., *Microsoft SQL Server 2008 Analysis Services Krok po kroku*, Microsoft Press, Warszawa 2009.

[2] Chodkowska-Gyurics A., *Hurtownie danych. Teoria i praktyka*, Wydawnictwo Naukowe PWN SA, Warszawa 2014.

[3] Furmankiewicz J., Furmankiewicz M., Ziuziański P., *Implementation of business intelligence performance dashboard for the knowledge management in organization* [in:] Wolniak R. (ed.), *Zeszyty Naukowe Politechniki Śląskiej*. Seria: *Organizacja i Zarządzanie*, No. 82, Wydawnictwo Politechniki Śląskiej, Gliwice, 2015.

[4] Furmankiewicz M., Ziuziański P., *Wdrażanie kokpitu menedżerskiego w ramach systemu BI w organizacji*, „Przegląd Teleinformatyczny” 2014, No. 1-2 (37).

[5] Januszewski A., *Funkcjonalność informatycznych systemów zarządzania. Vol. 2, Systemy Business Intelligence*, Wydawnictwo Naukowe PWN, Warszawa 2008.

[6] Kimball R., Ross M., *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*, John Wiley & Sons, Indianapolis 2013.

[7] Nycz M., Smok B., *Jak hurtownia danych może wspomagać menedżera w pracy* [in:] Pańkowska M., Porębska-Miąc T., Sroka H. (ed.), *Systemy Wspomagania Organizacji SWO 2008*, Wydawnictwo Akademii Ekonomicznej, Katowice 2008.

[8] Nycz M., Smok B., *Zastosowanie narzędzi etl w hurtowni danych* [in:] Sroka H., Porębska-Miąc T. (ed.), *Systemy Wspomagania Organizacji SWO 2004*, Wydawnictwo Akademii Ekonomicznej, Katowice 2004

[9] Olszak C.M., *Tworzenie i wykorzystanie systemów Business Intelligence na potrzeby współczesnej organizacji*, Wydawnictwo Akademii Ekonomicznej, Katowice 2007.

[10] Ziuziański P., *Kokpit menedżerski jako efektywne narzędzie do wizualizacji danych w organizacji*, [in:] Zieliński Z.E. (ed.), *Rola informatyki w naukach ekonomicznych i społecznych. Innowacje i implikacje interdyscyplinarne*, Zeszyt 1/2014, Wydawnictwo Wyższej Szkoły Handlowej, Kielce 2014.

[11] http://www.msbifun.pl/micrisoft/bi-magic-quadrant-gartnera/

[12] http://www.oracledbasupport.co.uk/dwh-terms/

[13] https://stackoverflow.com/questions/8184214/what-are-the-types-of-dimension-tables-in-star-schema-design

[14] http://www.geekinterview.com/question_details/17677

[15] http://de.dwhwiki.info/analyse/data-mining-dimension

[16] https://msdn.microsoft.com/en-us/library/ms175669.aspx

[17] http://www.kimballgroup.com/wp-content/uploads/2013/08/2013.09-Kimball-Dimensional-Modeling-Techniques11.pdf

[18] http://www.regnecentralen.dk/time_dimension_generator.html

[19] https://social.technet.microsoft.com/Forums/en-US/2cd9bb62-8b94-46f3-b2fc-3cd7147f57b9/why-some-many-different-types-of-dimensions?forum=sqldatawarehousing

[20] http://www.kimballgroup.com/2011/08/design-tip-137-creating-and-managing-shrunken-dimensions/

[21] https://techbusinessintelligence.blogspot.com/2014/09/junk-heterogeneous-and-degenerate.html

[22] https://ramprapanna.wordpress.com/2014/11/08/dimension-types/

[23] https://blogs.microsoft.com/blog/2017/02/16/microsoft-breaks-gartner-magic-quadrant-business-intelligence-analytics-platforms/

Abstract

The concept of a company data warehouse assumes the retention of data in the form of fact tables that describe reality as dimension tables. The structures of dimensions vary with regard to business requirements and implementation possibilities. The authors present the classification of dimensions that appear in the structure of data warehouses. The article discusses the issue of data warehouse design with the application of the Microsoft SQL Database Server. A particular attention is paid to the so called junk dimension and the management of the rapidly changing monster dimension with the application of minidimensions; a method of their implementation in the project is presented in the form of a case study.

Keywords

dimensions classification, data warehouse model, junk dimension, minidimension, rapidly changing monster dimension