The 2nd International Conference on Applied Information and Communications Technology

Proceedings

28 - 29 April 2014

كلية الشرق الأوسط
Middle East College

International Conference on Applied Information and Communications Technology (ICACIT-14)

ELSEVIER
A division of
Reed Elsevier India Private Limited

ISBN: 9789351072850

Published by Elsevier, a division of Reed Elsevier India Private Limited

Registered Office: 305, Rohit House, 3 Tolstoy Marg, New Delhi – 110 001
Corporate Office: 14th Floor, Tower 10B, DLF Cyber City, Phase-II, Gurgaon – 122 002, Haryana

Printed and bound in India

Int. Conf. on Applied Information and Communications Technology
(ICAICT-14) (684–689)

Modeling of fact tables in Data Warehouse

Zlatinka Kovacheva* Ina Naydenova**

*Middle East College, Knowledge Oasis Muscat, P.B. No. 79, Al Rusayl, PC: 124, Sultanate of Oman
**University of Sofia St. Kliment Ohridski, 5, James Bourchier boul.,Sofia, Bulgaria

Abstracts—Nowadays the warehouses store data that is not always suitable for presentation in a dimensional manner. This requires the use of advanced approaches to modeling fact tables especially when using a bottom-up data warehouse architecture. In this paper we provide are view of both basic and more complex techniques for fact tables modeling. We also discuss how some fact modeling techniques are suitable for a given query, but are not suitable for others. The given examples address the need of a methodology which combines the presentation flexibility of the top-down Inmon's approach and the quick results given by Kimball's bottom-up modelling approach.

Keywords: fact tables; query analysis; data warehouse; modeling.

1. Introduction

When it comes to designing a data warehouse, the two most commonly discussed methods are the approaches introduced by Bill Inmon and Ralph Kimball. Debates on which one is better and more effective have been on for years. But a clear cut answer has never been arrived upon, as both philosophies have their own advantages and differentiating factors, and enterprises continue to use either of these [1].

Bill Inmon recommends building a data warehouse that follows a top down approach. In Inmon's philosophy, it starts with building a big centralized enterprise data warehouse where all available data from transaction systems are consolidated into a subject-oriented, integrated, time-variant and non-volatile collection of data that supports decision making [2]. This centralized enterprise model is designed and normalized in a 3-rd normal form. Then the dimensional data marts, which contain data required for specific business processes or specific departments are created from the data warehouse [1].

In a contrast to Bill Inmon's approach (known as Corporate Information Factory Data Warehouse), Ralph Kimball recommends building a data warehouse that follows bottom up approach. In Kimball's philosophy, it is a first start with mission critical data marts that serve analytic needs of departments [2]. Keeping in mind the most important business aspects or departments, data marts are created. These data marts are eventually integrated together to create a data warehouse using a bus architecture, which consists of conformed dimensions between all the data marts. So the data warehouse ends up being segmented into a number of logically self-contained and consistent data marts, rather than a big and complex centralized model [3].

The approach to designing a data warehouse depends on the business objectives of an organization, nature of business, time and cost involved, and the level of dependence between various functions [1]. According to some sources [3]. Kimball's is the most frequently used methodology if you are using the Microsoft BI stack. It is popular because business users can see some results quickly, with the risk you may create duplicate data or may have to redo part of a design because there was no master plan. Oracle on the other hand prefers Inmon's approach in its Data Warehouse Reference Architecture. The Reference Architecture addresses the need for integration of data across the organization creating the "single version of the truth" and the different layers of abstraction. The so called Foundation layer records data at the lowest level of granularity and it is modeled in a normalized fashion. On the base of that layer a specific representation of data is built, according to business intelligence or data mining tools requirements [4]. With that

*Corresponding author. Tel.: +968-92-315-576; fax: +968-24-446-028
E-mail address:zlatinka@mec.edu.om

approach there is a master plan and usually you will not have to redo anything, but it could be a while before you see any benefits, and the up-front cost is significant. Another risk is that the time you start generating results, the business source data has changed or there are changed priorities and you may have to redo some work anyway [3].

Each of the a fore mentioned methodologies has its advantages and disadvantages but if you choose Kimball's approach,you should keep in mind that nowadays the warehouses store data that is not always suitable for presentation in a dimensional manner. Also some of the reports provided by data warehouse stend to be operational rather than analytical, but because they contain integrated data from multiple sources,the business users and corporate managers expect to receive them from the data warehouse. This requires non-standard approaches to modeling fact tables. In this paper we present both popular and not so common techniques for fact tables modeling.

2. Basic Techniques for Fact Table Modeling

In literature, there are three main types of fact tables according to the way of history storage: Transactional, Periodic snapshots and Accumulating snapshots.

2.1 Transaction Fact Tables

A transaction table is the most basic and fundamental. It represents an event that occurred at an instantaneous point in time. Transaction data fits easily into a dimensional framework. Atomic transaction data is the most naturally dimensional data, enabling you to analyze behavior in extreme detail. After a transaction has been posted in the fact table, you typically don't revisit it [5].

Example 1

If a customer buys 3 different products from a point of sale then the fact table will have 3 records for each transaction indicating 3 different types of product sale. Basically, if 3 transactions are displayed on the customer receipt, then we have to store 3 records in the fact table as the granularity of the fact table is at transaction level[6].

Table 1. An example of a transaction fact table.

Customer	Product	Amount	Data
Customer1	Product 1	10000	01.02.2014
Customer1	Product 2	5000	01.02.2014
Customer1	Product 3	1000	01.02.2014

2.2. Periodic Snapshots

Periodic snapshots are needed to see the cumulative performance of the business at regular, predictable time intervals. Unlike the transaction fact table where a row is loaded for each event occurrence, with the periodic snapshot, you take a picture (hence the snapshot terminology) of the activity at the end of a day, week, or month,then another picture at the end of the next period, and so on. The periodic snapshots are stacked consecutively into the fact table. The periodic snapshot fact table often is the only place to easily retrieve a regular, predictable view of long itudinal performance trends [5].

Example 2

The daily state of a bank account's balance should be modeled by taking a snapshot of that state on a daily basis. The balance of each account is recorded in the fact table at the end of the accounting day. There is a measurement for each account on each day (the data is dense in contrast to transaction fact tables).

Table 2. An example of a periodic snapshot fact table.

Account No	Currency	DB Balance	CR Balance	Accounting Date
7313323242	EUR	2500	10000	01.02.2014
7313323242	EUR	7000	5000	02.02.2014
7313323242	EUR	12000	1000	03.02.2014

2.2 Accumulating Snapshots

A transaction fact table records one row for each significant event in a business process. When the focus of analysis is the elapsed time between events, this form of organization is not optimal. Queries will be complex and perform poorly. When it is easy to identify the individual things being processed, an accumulating snapshot can streamline this kind of analysis [7]. A row in an accumulating snapshot fact table summarizes the measurement events occurring at predictable steps between the beginning and the end of a process. Pipeline or work flow processes, such as order fulfill ment or claim processing, that have a defined start point, standard intermediate steps, and defined end point can be modeled with this type of fact table. There is a date column in the fact table for each critical milestone in the process. An individual row in an accumulating snapshot fact table, corresponding for instance to a line on an order, is initially inserted when the order line is created. As pipeline progress occurs, the accumulating fact table row is revisited and updated [8].

Example 3

Let us take an example of order processing. Assume that the modeling process in a normalized form uses the following table structure:

Table 3. An example of order processing in conventional normalized table format.

Order No	Order Status	Date
12345	Customer Ordered Product	01-01-2012
12345	Order Product Dispatched from Warehouse	02-01-2012
12345	Handed Over to Courier Company	03-01-2012

If you look at the above events, you could see that each date has its own name, e.g., Customer Order Date, Warehouse Dispatch Date etc. In accumulating the fact table at each stage, dates will be updated with relevant facts [6]. That is, we have a table with columns Order number, Date of Order, Date of Order Dispatched from the Warehouse and so on. Here it is very important to determine the milestones to be documented.

Table 4. An example of an accumulating snapshot fact table for order processing.

Order No	Order Date	Dispatch Date	Send Date
12345	01-01-2012	02-01-2012	03-01-2012

3. Less Common Techniques for Fact Table Modeling

3.1 Factless Fact tables

A factless fact table captures the many-to-many relationships between dimensions, but contains no numeric or textual facts. They are often used to record events or coverage information. Common examples of factless fact tables include tracking student attendance or registration events, identifying product promotion events (to determine promoted products that didn't sell), tracking insurance-related accident events and others [10].

The first type of factless fact table is a table that records an event. Many event-tracking tables in dimensional data warehouses turn out to be factless. Sometimes there seem to be no facts associated with an important business process. Events or activities occur that you wish to track, but you find no measurements. In situations like this, build a standard transaction-grained fact table that contains no facts [11].

One good example is shown in Figure 1.

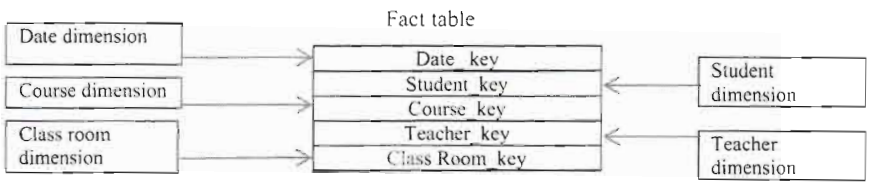

Figure 1. Example of Factless Fact table

The example presents how to track student attendance at a college. The grain of the fact table in Figure 1 is the individual student attendance event. In this case the dimensions are all well-defined and the fact table record, consisting of just the five keys, is a good representation of the student attendance event. Also there is no obvious fact to record each time a student attends a lecture. Tangible facts such as the grade for the course don't belong in this fact table. This fact table represents the student attendance process, not the semester grading process or even the midterm exam process [16].

Although there is no measurements in fact table a lot of interesting questions can be asked of this dimensional schema, including: "Which classes were the most heavily attended? Which classes were the most consistently attended? Which teachers taught the most students? Which teachers taught classes in facilities belonging to other departments? Which facilities were the most lightly used? What was the average total walking distance of a student in a given day?"

3.2 Slowly Changing Fact Tables (Time span tracking in fact table)

Traditionally, SCD 2 (Slowly Changing Dimension Type 2) is a trade mark for historical data storage in dimensional tables. When an object of the table has been changed, a new row is added (version of the object) with a date of validity "from –to".In some cases,this technique can be applied to store data in a fact table. Let us assume the organization has a huge number of customers and the task is to track very detailed customer profiles that include customer preferences, market segmentation, customer geography etc. The customer profiles are changing very often. The table would be very wide, huge and growing very fast. We can split the dimension table in a number of smaller tables (mini-dimensions). The question is how to join them together to be able to get complete customer profile as at any point in time. The first design could be to add a snapshot fact less fact table that joins these customer dimensions. This is not the best solution as the current snapshot of the customers will be added to every loading of the fact table. The table will grow extremely fast. Most often the rows inserted into the snapshot fact table will be the same as previous just with a new date. Therefore we can apply SCD 2 dimension technique for the fact table to avoid duplication of rows when the attributes are not changing. If any attribute of a customer profile changes, a new row will be inserted in the fact table. The number of rows in this fact table will be the same as the number of rows in the original dimension table Customer Profiles; however the data will take less of the disk space. It is often reasonable to apply Slowly Changing Fact technique to prevent the snapshot fact table from growing too fast [9].

4. Fact Table Modeling Query Analysis

The data warehouse input data (source data) can be represented in different formats using the techniques for modeling of fact tables discussed above. The question "Which approach to choose?" arises for the data warehouse designers. Usually, in this decision making, they are guided by the nature of the data, the information the system must provide and, not seldom, they use intuition and associations, based on their previous experience. Let us consider some examples:

Example 1

Let us have a transaction table with sales by products and customers. If a report must be presented to show the income from each customer at the end of the calendar month then the transaction fact representation will be ineffective and inconvenient – a customer may not have purchased a single product throughout the month but he must appear in the result. Also, the amount of data that must be aggregated may be considerable. A representation of the data as periodical snapshot on a monthly basis will be much more convenient for such kind of queries.

Example 2

Imagine that we have a table with relatively few rows – for instance, 5,000 contracts and each contract on the average changes once a year. With a daily loading of the data warehouse for a year we shall have 365*5,000 = 1,825,000 records in the fact table versus 10,000 if we apply the SCD 2 approach.

If we want to create a query for counting the contracts for a particular date, this can be easily achieved by using a filter of the kind "the report date to be between the validity dates of the version of the contract". Meanwhile, if we want to follow the history of the changes of a given contract throughout a year. this is also easily achieved. If we have used the snapshot scheme, however, in order to create the second query, it is necessary to dig into all snapshots of the contract for one year, to keep only the distinct records and then to show the date of each first change of the contract during that year. It is quite difficult to achieve this result by the end users of the BI instrument. In the presence of such queries the designers would prefer the SCD 2 model.

What will happen, however, if another part of the customers need report which follows the trend of the number of contracts on an annual basis. In such cases, the snapshot representation will be the more convenient choice, particularly if the company has not 5,000 but 1,000,000 contracts of the respective kind. In order to provide both kinds of reports, the designers will model both types of fact tables.

Discussion and Conclusion

Though the BI solutions are famous for their ad-hoc functionality, there are no universal models of the data. For some problems one representation is good, and for others – another one. Moreover, some instruments (such as data mining tools) require more specific representations. For Kimball's approach, there is no central atomary model which should be the source of all representations needed for the goals of the reports. To overcome this shortcoming, advanced techniques of fact tables modeling appear. Also, the necessity arises to keep the same data in different structures, i.e. a denormalization and duplication of the data in the facts appear. Dependences between the metrics are created, which the multidimensional model does not reflect.

Business nowadays is much more dynamic. In order to be competitive, innovation and changes in the models of work of business analysts is often needed. The necessity of new architecture allowing flexibility in the reflection of these changes is felt. For instance, in a bank, a reorganization of the profit and losses dimension and its hierarchies is often needed. New services and accounting models appear which modify the algorithms of calculation of many business metrics. The normalized, centralized model of Inmon is more flexible for such changes and allows to more easily following the dependences between basic and derived metrics. Thus the changes in the algorithms can be consistently carried out on the entire system. On the other hand, this architecture requires more time and resources. With the work dynamics and shrunk market nowadays, most companies are not disposed to spend so much money, having in mind the risk of permanent changes in the requirements. Moreover, the system maintenance has to be done by very good specialists due to the complexity of the architecture and the model.

In table 5 we present the architecture approach of several successful data warehouse projects as well as the assessment of a data warehouse supporting team about the risk of data inconsistency during the life time of the system. Some indicators were assessed in 5 degree scale (0 – none, 1 – very low, 2 – low, 3 – medium, 4 – high, 5 – very high). The provided information confirms the conclusions.

Table 5. Real life data warehouse project indicators.

Indicators	Project 1	Project 2	Project 3
Business Area	Enterprise Banking Data Warehouse	Data Warehouse of a holding dealing with insurance, banking and retirement services (focused on customer holding profile and services consumption)	Electricity Company Data Warehouse focused on meters energy consumption
Architecture	Kimball oriented	Inmon oriented	Inmonoriented
Project time and resources	1 year and 3 months (8 persons)	2 years (3persons)	6 months (2 persons)
Business area scope	Big	Medium	Small
Development and supporting team skills	Medium (3)	Very High (5)	High (4)
Advanced dimensional modeling techniques usage	Medium (3)	Very low (1)	Very low (1)
De-normalization and duplication of data in a multidimensional model	High (4)	Very low (1)	Low(2)
Risk of data inconsistency during the evaluation of the system	High (4)	Very low (1)	Low(2)

To address the problem of a changing business environment a modeling approach called Data Vault is used. To achieve this, the input data is stored in a format that separates structural information from descriptive attributes [12]. However, Data Vault makes no distinction between good and bad data unlike the practice in other data warehouse methods where data that does not conform to the definitions is removed or cleansed [13].

Over the last decade Inmon's and Kimball's modeling approaches have evolved. Kimball describes 4 stages of evaluation of his architecture (Stage 1 – simple dimensional model phase; Stage 2 – conformed dimension/master conformed dimension phase; Stage 3 – MDM phase; Stage 4 - hub and spoke architecture). The last Stage 4 architecture is very similar to Inmon's corporate information factory [15]. Inmon's next generation architecture (known as DW 2.0) contains many different architectural components that have been added to the basic corporate information factory. Some of the new aspects of the DW 2.0 architecture include unstructured data near line (or alternate) storage, taxonomies, changed data capture, an archival component and others.

Despite all efforts of designer research community to develop the best data warehouse architecture, for the moment there is no technique described to effectively combine the flexibility of Inmon's approach with the low cost and quick results given by Kimball's approach. Our experience in the development and maintenance of Data Warehouse systems shows that an important place in such new architecture must be taken by a system following and maintaining the dependences between the data and its duplication.

References

[1] Sansu, G.(2012).Inmon vs. Kimball: Which approach is suitable for your data warehouse?. Available at: http://searchbusinessintelligence.techtarget. in/tip/Inmon - vs- Kimball- Which- approach- is- suitable- for- your- data- warehouse

[2] Zent T. (2014).Kimball vs. Inmon Data Warehouse Architectures. Available at: http://www. zentut. com/data-warehouse/kimball-and-inmon-data-warehouse-architectures/

[3] Serra, J. (2012). Data Warehouse Architecture – Kimball and Inmon methodologies. Available at: http://www.jamesserra.com/archive/2012/03/data- warehouse- architecture- kimball- and- inmon- methodologies/

[4] Oracle Corporation (2010). Enabling Pervasive BI through a Practical Data Warehouse Reference Architecture, *Technical Whitepaper*

[5] Kimball, R., &Ross, M.(2013).*The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling* (3rd ed.). ISBN: 978-1-118-53080-1. John Wiley & Sons Inc.

[6] BidwbooksOrg. (2014).Data Warehousing Concepts: Fact Tables. Available at: http://www.bidwbooks.com/data-warehousing-concepts-fact-tables/

[7] Adamson C. (2010). Star Scheme: The Complete Reference™.ISBN-10: 0-07-174432-0.McGraw-Hill

[8] Kimball Group Corporation(2013).Accumulating Snapshot Fact Tables. Available at: http://www.kimballgroup.com/data-warehouse-business-intelligence-resources/kimball-techniques/dimensional-modeling-techniques/accumulating-snapshot-fact-table/

[9] Bukhantsov Org.(2012). Slowly Changing Fact Tables. Available at: http://bukhantsov.org/2012/04/slowly-changing-fact-tables/

[10] Kimball Group Corporation (2003).Design Tip #50: Factless Fact Tables?. Available at: http://www.kimballgroup.com/2003/10/16/design-tip-50-factless-fact-tables-sounds-like-jumbo-shrimp/

[11] Informatica Reference (2012).What is a factless fact table? Where we use Factless Fact?. Available at: http://informaticareference.wordpress.com/author/informaticareference/

[12] Hultgren, H.(2012). Data Vault Modeling Guide. Gnesee Academy. Available at:http://hanshultgren.files.wordpress.com/2012/09/data-vault-modeling-guide.pdf

[13] Linstedt,D. (2010).*Super Charge your Data Warehouse*. ISBN 978-0-9866757- 1-3.

[14] Bukhantsov Org.(2012).What is Data Vault?. Available at:http://bukhantsov.org/2012/04/what-is-data-vault/

[15] Inmon, B.(2010).A tale of two architectures.*Technical white paper*.Available at: http://lltcorp.com/sites/default/files/DW-2.0-Architecture.pdf

[16] Kimball Group Corporation (1996).Factless Fact Tables. Available at: http://www.kimballgroup.com/1996/09/02/factless-fact-tables/