Business Intelligence Part 1

OLAP and Data Warehousing

Hausarbeit

Studiengang Angewandte Informatik

Duale Hochschule Baden-Württemberg Stuttgart

Abgabedatum:	3. Mai 2016
Matrikelnummer, Kurs:	2006895, 5707482, TINF14A
Gutachter der Dualen Hochschule:	Marcel Thoms

Inhaltsverzeichnis

1 Introduction

Database-Management-Systems are fundamental to any kind of business. The possibility to store data in a structured way and to access it at any later time is the most obvious usecase for databases. It is essential to be able to access stored data at any time in the future. But since the sole, dump access of data is going to overwhelm the user with increasing storage-complexity, the need for an intelligent access of data is a must for larger database-applications. Imagine a Database which holds all countries of the world with their capitals and regions. This is an reasonable data-set. Most Database will have an extremely larger set of data-entries. But even in this case, the direct access of all data without filtering or intelligent accessing wouldn't scale in any affordable effort. If a user wants to see if a specific region belongs to a country, he wouldn't like to search through the whole data-set to find his region. For this reason, one implemented intelligent data-access from the begin of Database-Management. In a simple case as the previous, it seems to be sufficient to use the simple SQL-Language and it's predefined methods and syntax to retrieve the perfect result. This would scale perfectly for such a small databases. But, as mentioned before, the usual data which is gathered from larger companies is far more complex and thus needs far more tables, entries and accesses than our country-data. In times of globalization, many companies and therefore many databases will be distributed all over the world as well. And even if a company is only located at one location, the data should be distributed at different locations to reduce the threat of data-loss in case of local disasters. To bring such data to an affordable scale for usage, the world of Database-Management defined contexts like Data Warehouses or OLAP. This document will give an introduction to those terms to display the possibilities of modern DBMS for high performance data management. We will furthermore define some usecases to give the reader an more intense overview into the subject. The third part will compare the usual software products which are available on the market to implement those terms into a real database system.

2 Definition

2.1 Data Warehouse

Nowadays companies, government agencies and other organizations all produce huge amount of data. All this data is distributed over different applications and their database management systems. In order to actual use, compare and process the data it needs to be accessible at one single location. For this reason, Data warehouse was created, which is a central storage, where data from many different applications is stored. Each application has it's own database but in order to analyse the data, it has to be accessible through one common access-point, the data warehouse. Since another large database which contains all data entries from the sub-databases wouldn't scale for larger applications, the data warehouse tries to store transactional data to make it possible to analyse large amount of data without accessing every single tuple in the sub-databases. This is commonly used to retrieve for example daily logs of a specific database. The user is able to get the analysis through its periodic snapshots instead of manually analysing each tuple of interest in the actual SQL-Database. Especially for long-term analysis, this kind of snapshot is extremely performant since it is only necessary to view on e.g. the snapshots of 100 days instead of taking the average from every transaction within those 100 days. From a conceptional point of view, a data warehouse is build in a star-like formation. The 'inner star' consists of the gathered meta-data for example the previous mentioned periodic snapshots. This is also the position, where the user will post his requests. The 'outer star' in this structure consists of the storage of the raw data so that it is always accessible for the user as well. This enables the access to every single dataset if the meta-data isn't sufficient for the user-request. Data warehouses defines this meta-data in the so-called fact tables. The next section will explain this concept in greater detail.

2.1.1 Fact tables

A fact table is the kind of storage-structure where data warehouses store their meta-data. It is located in the innermost position of the star-wise data warehouse structure. A fact can contain periodical data over specific dataset from the outer

dimensions. Therefore, each fact consists out of basically 2 different informations being the foreign key as a reference of the source of the data and the 'fact-column' which describes the collected data for example the average of the source-dataset over a specific time-interval. There are 3 different kind of measurement-types for facts in a data-warehouse.

Additive Measurement

Additive measurement is used, when the data which is stored in the fact table can be used over all dimensions of the data warehouse. For example if it is possible to add the amount of a sold product to a total amount of products when changing for example the time-dimension.

Non-additive Measurement

The term non-additive is used to describe meta-data which is extinct to a specific dimension in the warehouse. In other words, it describes measurement methods which cannot be made over all dimensions since it is a more specific information. One example for such data would be the average temperature of a heat-sensor in a room. Since the measured heat of at one specific time shouldn't be simply added up with the one before or after. For heat-values, an average has to be measured instead of the summed up values of all measurements.

Semi-additive Measurement

The last measurement-type describes a mixture of the previous defined types. A measurement can be called semi-additive if there is not only one dimension which holds the information and the type isn't additive. So the data is not unique and not in every dimension. An example would be the stock level of any company since the measured values of one product can be added to the ones from another product to get the count of all products but they could also be viewed at as one-dimensional data-set which changes when products are sold or bought.

Types of fact tables

These measurement-types can now be used for different types of fact-tables. The types can be used as a 'rule-of-thumb' which measurement types could be found in the fact tables. The Two most frequently used types are cumulative and snapshot. Cumulative fact tables describe what happened over a given period of time. It mostly consists of additive data for example the amount of sold onions during each week. The second fact table type, snapshot, is used to see how the data was

gathered at a specific point in time. Whilst cumulative fact tables consists mostly of additive data, the snapshots usually contain non-additive and semi-additive datasets. An example would be the snapshot of a stock at the end of a month. Where each entry describes the trend of selling a specific product.

2.2 OLAP

For readability and comprehensibility purposes, the following chapter will use the term abstraction instead of the more common termination of aggregation. OLAP, which stands for Online Analytical Processing, is a high performance analytic query tool for very large datasets. It is designed to work with multidimensional databases and works perfect with a data warehouse structure as described in chapter 2.1. Online Analytical Processing was initially built to be able to drive large analytical research and generate information out of it to support business decisions. It interactive Design implementation tries to make the data as presentable and usable as possible even for user without intense IT-Background knowledge. Since OLAP needs multiple dimensions, it is possible to abstract the data from the data warehouse to an more intuitive level. To illustrate this behaviour, check again the example of a database with all planets, their countries, regions and cities in our universe. This is a slightly more complex situation than in the introduction where we only looked at our earth's regions. It would be difficult for anyone, to immediately find the most specific information over regions or cities if he would only be confronted with the pure data. OLAP now defines the dimensions from it's underlying data warehouse which could be the previous mentioned parameter planets, countries and so on. The standardized structure how OLAP is organized is called Hypercube. A Hypercube is the representation of the usually 2-dimensional relational data into a 3-dimensional cube-wise structure. The different dimensions and abstractions can be modelled as separate levels of the cube. To be able to handle the levels of the Hypercube, they have to be related to each other. In a more complex system, several Hypercubes will be present to represent disjoint datasets in their own abstraction-levels.

2.2.1 Codd Requirements

The official requirements to any OLAP-system where defined by Edgar F. Codd back in 1993 A.D. The following 12 Rules are the principles after which every OLAP-based Analytic should be formed. (vgl.[1]).

[1] *Providing OLAP (On-line Analytical Processing) to User-Analysts: An IT Mandate* 1993.

Multi-dimensional conceptual view

The data set of the underlying data warehouse has to be presentable in a multi-dimensional pattern. This ensures the possibility for OLAP to structure the data into it's Hypercube(s). The multi-dimensional structure enables the intuitive representation and manipulation of the data.

Transparency

The processes from OLAP should be all time transparent to the user. This ensures the backtracing of failed analytics as well as the security for the user, that the exact analysis was made. The other part of this rule states, that OLAP should be implementable in every type of system which holds the preconditions for OLAP regardless of which type of analysis was made before / after. This ensures compatibility to many kind of Data Warehouses.

Accessibility

The OLAP service has to be accessible for the user at all time and independently from where the data source lays. This includes, that the user shouldn't care where the physical storage lies.

Consistent reporting performance

The performance of OLAP shouldn't suffer significantly when additional dimensions are added to the data set.

Client/server architecture

The system should be implemented as a classical client/server architecture. This implies, that there should be one OLAP server which is able to handle several user input simultaneously. Those users should be able to interact with the service independently

Generic Dimensionality

Each data dimension in all Hypercubes of the OLAP should have a generic dimension type which defines similar capabilities and structure for every dimension. This ensure the reusability and compatibility over more than one dimension and therefore ensures the analytical procedures to work with generic definitions regardless of the dimension input / internal structure.

Dynamic sparse matrix handling

Since many business matrices are appearing to have a sparse dataset nowadays, the service should be able to ignore larger gaps in all datasets. This means, it should be dynamically work around sparse matrices.

Multi-user support

Every OLAP should be able to process concurrent user inputs and analysis. With this rule, the security of all data and it's integrity has to be ensured, similar to the basic requirements for standardized Database Management Systems (DBMS).

Unrestricted cross-dimensional operations

The dimension of each OLAP should be arranged and created in a manner, which makes it possible to interact with independently many dimensions without having structural errors.

Intuitive data manipulation

The Userinterface of the software should be arranged to make changing dimensions or other processes as intuitively as possible

Flexible reporting

The reporting and logging of status codes or updates from queries should be customizable for the user.

Unlimited Dimensions and abstraction levels

The number of Dimensions shouldn't depend on the software implementation. This means that it should be theoretically possible to have an unlimited number of dimensions and abstractions.

2.2.2 Multi-dimensional data-handling

Since OLAP is defined for multi-dimensional data it provides the functionality to handle it's dimensions fluidly. The most often used functionalities to handle the dimensions and levels of the Hypercube with OLAP are called 'Drill-Down', 'Roll-Up', 'Slice' and 'Drill-Through'.

Drill-Down and Roll-Up

Drill-Down and Roll-Up are the possibilities to 'go' levels of abstraction 'down' or 'up' in the multi-dimensional dataset of a OLAP-Datawarehouse-System. It is used to stay at a specific dimension-view but reduce or enlarge the 'height' of view. To make the advantage of Drill-Down or Roll-up clear, remember the country database from the introduction of this chapter. Imagine, the database user wants to retrieve all regions which are currently reported on the mars. He therefore starts to define his dimension-parameter to see all countries. Since our database holds only areas and subset's of them, this shouldn't be a difficult task. When he now starts to see all planets, the highest level of abstraction, he needs to drill down his view on seeing only countries on planets. This process is called

Drill-Down. He is now able to further substantiate his view to finally get all regions. If he later on decides to see the countries again, he can Roll-Up again one or two levels to the more abstract view of countries. This enables the user to exactly define his level of abstraction for his view. The problem which he might have for the moment is, that he is now presented by hundreds or thousands of regions where he will definitely find the regions of interest, but with an unaffordable price of inefficiency since he needs to search through each region to find his desired output. This is the situation where the 'Slice' functionality comes into play which will be discussed in the next section.

Slice

In section 2.2.2, the concept of going abstraction levels down or up was explained. The problem which occurred was the overwhelming mass of outputs when reducing the view's abstraction. The solution in OLAP to substantiate this output is called Slice. The Slice-functionality is not varying the abstraction of the given view but is 'cutting' the substantial parts away from the unnecessary information. The results are then called dices, since they are smaller Hypercubes on their own. To stay with our given Example, if the user now really wants to find all regions or cities on the Mars, he can define his Slicing-parameters to be Mars. The previous output is now reduced to the desired cities / regions on mars.

Drill-through

The last important function for OLAP-Analytics is called Drill-Through. If the accessed view is at the bottom of abstraction, displaying the most concrete dataset, it is possible to 'join' the view with other 'on-the-bottom' views which share the same dimensions. This feature is used, if it comes to a more complex database where several Hypercubes are defined which originally don't share all dimension, but potentially some of them. The user is now able to drill down on both cubes to retrieve 'dices' with equal dimensions. To get aggregated information over both cubes, he can now join, or Drill-Through, both dices.

2.3 Underlying storage modes

The data which is stored by the datawarehouse has to be in pre-specified storage modes to enable the usage of OLAP-Software. The OLAP-System differentiates between the heterogeneous modes ROLAP and MOLAP and the hybrid HOLAP storage. These differences can be seen in the way, the Data warehouse stores the most exact data. While ROLAP, which stands for Relational OLAP holds it's data

in classical relational schemas, namely SQL-Schemas, MOLAP is designed to use Multidimensional (that's where the M in MOLAP comes from). The advantage of ROLAP is it's stability when it comes to larger datasets. It is stable as long as the storage volume is sufficient no matter how many entries each dataset has. Against this huge advantage, MOLAP is capable of handling large datasets in a more flexible and faster way than ROLAP. So it is less stable for larger data, but more flexible and has a higher performance when it comes to the actual requests. Since both advantages are not easily to separate for a usecase, the HOLAP storage mode was designed. HOLAP is the possibility to use ROLAP and MOLAP whichever fits better to the related dataset. The disadvantage is of cause the higher complexity which leads to less performance than MOLAP and less stability than ROLAP in the total outcome.

3 Usecases

3.1 Business Analysis

4 Software products

5 Summary

Literatur

Providing OLAP (On-line Analytical Processing) to User-Analysts: An IT Mandate (1993). Codd & Date Inc.