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

Abbildungsverzeichnis **II**

1 Introduction **1**

2 Definition **2**
2.1 Data Warehouse 2
2.1.1 Fact tables 2
2.2 OLAP 4

3 Usecases **5**
3.1 Business Analysis 5

4 Software products **6**

5 Summary **7**

Literatur **8**

Abbildungsverzeichnis

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. Data warehouse 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 only 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 define those 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 ou-

ter dimensions. Therefore, each fact contains 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.

2.1.1.1 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.

2.1.1.2 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.

2.1.1.3 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.

2.1.1.4 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

3 Usecases

3.1 Business Analysis

4 Software products

5 Summary

Literatur

Google (2016). *Google Analytics zu angular, ember und Backbone*. URL: https://www.google.com/trends/explore?hl=en-US#q=ember.js%2C%20angularjs%2C%20backbone.js&date=1%2F2011%2061m&cmpt=q&tz=Etc%2FGMT-2.

McKeachie, Craig (2013). *Choosing a JavaScript MVC Framework*. URL: http://www.funnyant.com/choosing-javascript-mvc-framework/.

Node.js & Co (2012). dpunkt.verlag.

Oracle (2016). *Model-View-Controller (MVC 1.0) Specification*. URL: https://java.net/projects/mvc-spec/pages/Home.

Shaked, Uri (2016). *AngularJS vs. Backbone.js vs. Ember.js*. URL: https://www.airpair.com/js/javascript-framework-comparison#6-angularjs.