УДК 004.45

DATA WAREHOUSE DESIGN PROBLEMS IN STATISTICAL ANALYSIS TASKS

Rassul Yunussov
Suleman Dimerel University

Түйін

Мақала деректердің статистикалық анализ жүйесі құрылымы тақырыбына және ақпараттың сақтау моделін таңдау мәселесіне арна

Кілт сөздер: деректердің қоймасы, статистикалық анализ, деректердің өңдеу.

Резюме

Статья посвящена теме построения систем статистического анализа данных и проблеме выбора модели хранения информации.

Ключевые слова: хранилище данных, статистический анализ, обработка данных.

Abstract. *This article is dedicated to the problems of data warehouse analysis systems design.*

INTRODUCTION

Business Intelligence solves the problems of representing the analytical information to stakeholders based on the results of some process data aggregation by the established rules.

For instance – the state business process of accepting the requests from citizens in service centers. The relational database management system of such processes automation certainly will have some entities like state authority, the request type, the state database of citizens on so on. And if there is very intensive data exchange in this process, there is certainly exists problem of analyzing all this data and presenting the summary reports of business process health and for decision making, including solving the economic, social, politic and administrative tasks. For state authorities there can be:

- The first quarter requests number;
- Last year requests total number from all state authorities;
- Requests trend for the particular type of requests along the year;
- And so on.

When it comes to the state authorities business processes, that can concern different type of questions, it is important to understand the huge data amount of such processes and necessary hardware to handle all these data. And with data growth usually grows the demand in different type of analyses of this data including different type of reports presentations.

Thus, there is problem that usually faces engineers – is to provide effective and usable instruments with maximum abilities for data analyses. There are a lot of existent software that can help in this problem – represented by Oracle, SAP, Microsoft, Microstrategy, Pentaho and many others.

Sometimes solutions of such companies do not ideally comply with existent requirements, and sometimes do not comply at all. And in most cases the established budget can not afford the proposition of existent software.

Today more and more evidence we see of SaaS and PaaS actuality. Such huge companies as SAP, Oracle make their first steps into implementation of such approaches and providing such solutions in cloud. One of the evident example can be SAP HANA, that is specially designed in memory column oriented database. Such a solution allows make analysis of data without involving the ETL processes in to the pipe.

Different approaches, that are exist in the solution of data analysis tasks shows that researches in this area continue, and it is important to make researches in this direction.

MODEL DESIGN

The main feature of aggregates data warehouse – is intensive data read and stream update. And based on this principle many solutions was built. And one of them is Oracle OLAP Data Warehouse. This particular software allows different variations of implementation, depending on the particular task – relational model, relational model with materialized views, and multidimensional olap.

Each of these models has its advantages and disadvantages.

And all theses models has one disadvantage, that is obvious for developers, who are trying to create universal analytic software, that can be easily migrated from one business process to another. And this problem necessitate to recreate again and again algorithms of extraction data from these OLAP warehouses, which lead to increase of budged of distributing such solutions.

This topic provide the solution for unifying the aggregate data on the basis of transforming the entire entities to the strict finished form with RAM usage to store all necessary pre-calculated aggregates. And this task consists of:

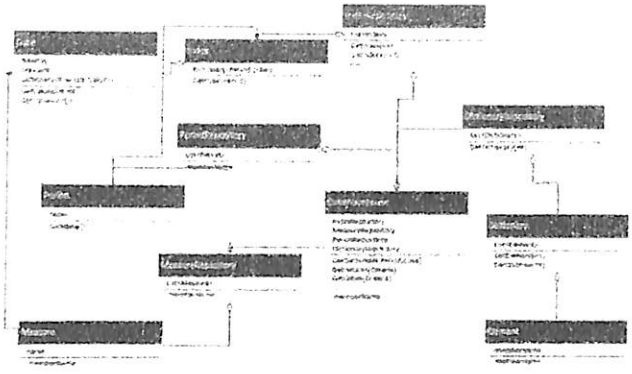
- Effective memory usage;
- Building effective model for data extraction;
- Persistence of data.

First, we need to define entities of such data warehouse:

- Index – The entity that is represented by quantitative and quality characteristics. For example – income or outcome.
- Measure – Represents different types of indicators measures, for example – meter, liter, dollar or tenge.
- Dictionary and it's elements – named entity, that define the quality characteristics of indicators, for example – the gender dictionary, that can contain two elements – male and female.
- Period – named entity, that defines the period of time in which aggregation have been made, for instance – year, quarter, month, day.

- Value – named entity, that represents the aggregated number as a value

Just as we have defined the entities, we need to make class object model, that represents their relationships.



The image also contains service classes, like ObjectNameRepository, DataWarehouse. These classes are very convenient in practice, when it is necessary to create interface for working with collection to query data from storage by defined rules.

Index class – contains characteristics, such as index name, and other system attributes necessary for particular application. And contains the collection of Cube classes.

Cube class – is the storage for the quality characteristics of index values and values their selves. The quality of characteristics – are the elements from the dictionary collections. Each collection of characteristics is the point in the defined cube space. For example – if we have three dimensional cube with three sides – weight, color and size, then each point in this 3 dimensional space will represent defined criteria for each side of the cube. Any dimensional cube can be presented as two dimensional matrix where in columns will be the dictionaries and in rows – different available combinations of the dictionaries' elements. For example, the row 1 can have values: «Weight — 100 kg», «Color – red», «Size – small». To optimize the memory usage and search engine for such matrix we can use two approaches:

- Store two dimensional matrix in one dimensional with offsets
- Use the bit array. In this case it is supposed to analyze each column of the matrix to discover diversity of elements. For example, for the column of genders there can be used only three values – total, male and female. Then maximum allocation size for this column can be defined as 2 bits. And so on for each column of the matrix.

The Cube class should contain Measure class inside, that determines the measure units for all rows in the matrix. This approach allows each index of the data warehouse to have many measure units according to the cube. As instance we can consider the index Sales, that can be measured in natural units – items and in money – KZT. This article covers only simple cubes with one measure to all values under

the cube. But in real applications there are some cases when even one cube can have different measure units. To solve this situation it is necessary to bring up new abstraction layer for separating such cubes from each other – the simple cubes with one measure, and complex cubes with amount of measures.

The Cube class should also contain the Value collections. Thus each amount of characteristics in the Cube's matrix – should have values, expressed as number. And it is possible to optimize memory usage storing real values by classifying them. For example, working with the relational database, when we use the column type for storing numbers we usually use the maximum available type – for example double, that needs 10 bytes. But when we work in RAM we definitely need to use different approach, because not all indicators require such precision. For example the index – Plan execution rate needs only integer values from 0 to 100, that require only one byte instead of 10, that double provides. Thus we can implement preprocessing analyses for each index to choose the right storage type – from byte to double. We can create the abstract class Value and make derivate classes from it with all necessary types that are required to create the solution. In practice this particular approach gives huge effect, because usually only 5% of all aggregated data require the type double.

CONCLUSION

There are some situations where relational databases with all their scalability and redundancy are not suitable. This topic shows the alternative way of Business Intelligence system design. And as we can see this approach allows to illuminate disk reads for information extraction from data warehouse, which is crucial for any information system. Excluding the disk reads we expand the data extraction speed boarders that were imposed by hard disk drives abilities. Also such approach allows to reduce expenses for designing instruments of business analysis, illuminating the necessity to build new extractors for new data warehouses, because the BI software works with strictly predefined entities in this approach and doesn't depend on the particular warehouse structure.

Refferences

1. M. Rittman, Oracle Business Intelligence 11g Developers Guide, Oracle Publications, 2012

2. R. Kimball, M. Ross, B. Becker, J. Mundy and W. Thornthwate, Practical Tools for Data Warehousing and Business Intelligence, John Wiley & Sons, 2010

3. R. Kimball, M. Ross, The Data Warehouse Toolkit, Third Edition, John Wiley & Sons, 2013

4. S. Few, Information Dashboard Design, Second edition, Analytics Press, 2013