The impact of indexes on data warehouse performance

El Amin Aoulad Abdelouarit[1], Mohamed El Merouani[2] and Abdellatif Medouri[3]

[1] Laboratory modeling and information theory
Abdelmalek Essaadi University, Tétouan, Morocco

[2] Laboratory modeling and information theory
Abdelmalek Essaadi University, Tétouan, Morocco

[3] Laboratory modeling and information theory
Abdelmalek Essaadi University, Tétouan, Morocco

Abstract

A data warehouse designer should consider the effectiveness of data query, this depends on the selection of relevant indexes and their combination with the materialized views, note that the index selection is a NP-complete problem, because the number of indexes is exponential in the total number of attributes in the database, So, it is necessary to provide, while the data warehouse design, the suitable type of index for this data warehouse.

This paper presents, in some steps, a comparative study between the index B-tree and Bitmap type, their advantages and disadvantages, with a real experiment based on two factors: size of index and clustering factor, this shows that the Bitmap index is more advantageous than the B-tree one.

Keywords: *Data Warehouse DBMS, indexes, business intelligence.*

1. Introduction

The data warehouse administrator takes several decisions regarding the administration tasks, such as databases logical and physical designs, management of storage space and performance tuning (performance tuning).

The most important task is the physical design of databases, including data organization and improving access to these data. To improve the access time, the administrator uses general index to quickly find the necessary information without a request to review all the data [1], [3], [5], [7].

Index selection is difficult because their number is exponential in the total number of attributes in the database. So the index plays an important role in the performance of databases, for that we focus on this aspect of the data warehouse, which it considers the focus of the designer when editing and query optimization selection.

The objective is to minimize the query execution time. And as queries in a data warehouse are based on the index, we will work on the problem of choosing the type of index when designing our warehouse data.

There are several types of indexes supported by databases such as Bitmap [4] B-tree [3], [6], [7], [8], Bitmap join [9], range-based bitmap index [10] etc... In this sense we have chosen two types of index relevant to this study, the index type: B-tree index and type Bitmap.

2. Bitmap Index

2.1 Definition

A bitmap index is a data structure defined in a DBMS used to optimize access to data in databases. It is a type of indexing is particularly interesting and effective in the context of selection queries. The index bitmap attribute is encoded in bits, where its low cost in terms of space occupied. [7] All possible attribute values are considered, the value is present or not in the table. Each of these values is an array of bits, called bitmap, which contains as many bits as n-tuples present in the table. Thus, this type of index is very effective when the attributes have a low number of distinct values. Each bit represents the value of an attribute for a given tuple. For each bit, there is an encoding presence / absence (1/0), which indicates that a tuple or not the present value characterized in bitmap.

To illustrate how a bitmap index works, we take an example EE-PP-O'Neil and O'Neil [2]. Table 1 illustrates a basic bitmap index into a table containing 9 records, where the index is created in the C column with integers ranging from 0 to 3, we say that the cardinality of the column C is 4, by what there are 4 distinct values [0, 1, 2, 3], where the index bitmap C Contains 4 bitmaps shown as B0, B1, B2 and B3 corresponding value represents. In this example, the first line where RowID = 0, column C is worth 2, consequently, B2 column bit value "1", while the other

bitmaps are set to "0". Same for the next line, where C = 1 corresponds to the bitmap B1 is set to 1 and the rest to "0". This process is repeated for the remaining lines. [12].

Table 1: Basic Bitmap adopted by [9]

ROWID	*C*	*B0*	*B1*	*B2*	*B3*
0	2	0	0	1	0
1	1	0	1	0	0
2	3	0	0	0	1
3	0	1	0	0	0
4	3	0	0	0	1
5	1	0	1	0	0
6	0	1	0	0	0
7	0	1	0	0	0
8	2	0	0	1	0

2.1 Properties

Bitmap indexes have a very interesting property of responding to certain types of requests without returning the data themselves, thus optimizing the response time, disk storage. This is possible by counting operations (COUNT) and logical operators (AND, OR, etc.) that act "bit by bit" on bitmaps.

3. Bitmap Index

3.1 Definition

The index B-tree stores the index values and pointers to other index nodes using a recursive tree structure. [3], [6], [7], [8] The data are easily identified by traces pointers. The highest level of the index is called the root while the lowest level is called the leaf node or "leaf node". [7] All other levels between them are called branches or internal nodes. All roots and branches contain entries that point to the next level of the index. Leaf nodes consist of the index key and pointers pointing to the physical location of records. We present details of the index B-tree structure [7].
The B-tree structure is used by the database server to configure the index (Figure 1)

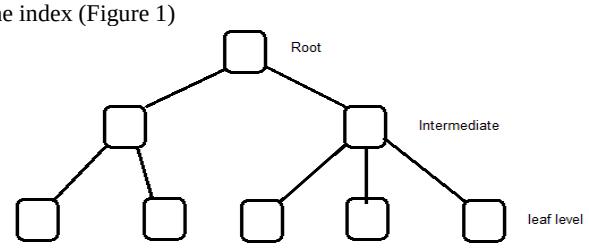

Fig 1: B-tree structure

Root or root is the highest level of the index points to the following levels of nodes branches.

Intermediate nodes or branches contain pointers to the following branches or to the leaf nodes level.

Node leaves or leaf nodes: the lowest level of the index points to other node leaves.

4. Hypothesis

The conventional wisdom is that bitmap indexes are most appropriate for columns having low distinct values - such as gender, marital status, and relationship. This assumption is not entirely accurate, however. In reality, a bitmap index is always advisable for systems in which the data is not updated frequently by many competing systems. In fact, as I will demonstrate here, a bitmap index on a column with unique values to 100% (candidate of the primary key column) is as effective as a B-tree index.

5. Analysis and results

5.1 Analysis

As known, the bitmap index is more efficient than the b-tree index by its low cardinality columns, we present in this experimentation the performance given by Bitmap index comparing with the B-tree.
We use two factors to of cost-based optimizer:

- The index size
- Clustering factor

Step 1:

In our Data warehouse schema, we created a table named “Employees” with 100000 records and with a column named employee_id with 100000 distinct values, and then we added a GRADE column with 4 distinct values only.

Step 2:

We create now a standard B-Tree index on the GRADE column using this SQL:

SQL> create index employees_grade_i on employees(grade);
Then we check the index size using this query:

SQL> select index_name, index_type, distinct_keys, blevel, leaf_blocks from dba_indexes where index_name=’EMPLOYEES_GRADE_I’;

And we got this result:

Table 1: Basic Bitmap adopted by [9]

INDEX_ NAME	*INDEX _TYPE*	*DISTINCT _KEYS*	*BLEVEL*	*LEAF_ BLOCKS*
employee s_grade_i	Normal	4	1	176

Step 3: Creating a bitmap index on the same column to compare the size (dropping the b-tree index created in first step)
SQL> create bitmap index employees_grade_bitmap_ii on employees(grade);

Step 4: In the bitmap index size checking, we use the same query:

SQL> select index_name, index_type, distinct_keys, blevel, leaf_blocks from dba_indexes where index_name='EMPLOYEES_GRADE_II';

Table 3: Bitmap index size checking result in GRADE column

INDEX_ NAME	*INDEX _TYPE*	*DISTINCT _KEYS*	*BLEVEL*	*LEAF_ BLOCKS*
employee s_grade_i i	Bitmap	4	1	10

Note that the index size is reduced from 176 to 10 (while going from B-tree to bitmap index)

Step 5: the bitmap index creation on employee_id column that contains 100000 distinct values:

SQL> create bitmap index employees_empid_bitmap_i on employees(employee_id).

By checking the index size using the same query, we have this table as result:

Table 4: Bitmap index size checking result in EMPLOYEE_ID column

INDEX_ NAME	*INDEX _TYPE*	*DISTINCT _KEYS*	*BLEVEL*	*LEAF_ BLOCKS*
employee s_empid_ bitmap_i	Bitmap	100000	1	348

And when trying with B-tree index we have this result:

Table 5: B-Tree index size checking result in EMPLOYEE_ID column

INDEX_ NAME	*INDEX _TYPE*	*DISTINCT _KEYS*	*BLEVEL*	*LEAF_ BLOCKS*
employee s_empidb tree_i	B-tree	100000	1	222

5.2 Results

For large distinct values B-tree index occupies less size, and for minimal distinct values, the bitmap index occupies less size.

Clustering factor: considered as the sum of rows orders in a table based on the index values.

- If this amount is near the number of blocks, then the table order is well done, and the index entries in a single leaf block are pointing to rows stored in the same data blocks.
- If the value is near the number of rows, then the table is randomly ordered, so is improbable that the index entries in a single leaf block are pointing to rows stored in the same data blocks.

Table 6: Clustering factor and blocks used for B-Tree index on GRADE column

INDEX_NAME	*CLUSTERING _FACTOR*	*BLOCKS*
employees_grade_i	1148	256

Table 7: Clustering factor and blocks used for Bitmap index on GRADE column

INDEX_NAME	*CLUSTERING _FACTOR*	*BLOCKS*
employees_grade_ii	20	16

6. Conclusion and future work

By using the B-tree index, the optimizer opted for a full table scan; this operation makes a higher clustering factor, whereas in the case of bitmap index that makes a low Clustering factor, he used to answer the query. You can deduct the performance by the number of I / O required fetching the result.
The message here is pretty clear. Both indices have a similar goal: to return results as fast as possible. But the choice of which one to use should depend only on the type of application, and not on the level of cardinal.
As future work, another study will be done on data warehouse schema comparison, especially it impact on data warehouse performance.

References

[1] S. Chaudhuri, U. Dayal,An Overview of Data Warehousing and OLAP Technology., ACM SIGMOD RECORD. 1997
[2] E. E-O'Neil and P. P-O'Neil, Bitmap index design choices and their performance impli-cations, Database Engineering and Applications Symposium. IDEAS 2007. 11th International, pp. 72-84.
[3] R. Kimball, L. Reeves, M. Ross, The Data Warehouse Toolkit. John Wiley Sons, NEW YORK, 2nd edition, 2002
[4] W. Inmon, Building the Data Warehouse., John Wiley Sons, fourth edition, 2005
[5] C. DELLAQUILA and E. LEFONS and F. TANGORRA, Design and Implementation of a National Data Warehouse. Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 pp. 342-347
[6] D. Comer,Ubiquitous b-tree, ACM Comput. Surv. 11, 2, 1979, pp. 121-13
[7] R. Strohm, Oracle Database Concepts 1g,Oracle, Redwood City,CA 94065, 2007
[8] C. Dell aquila and E. Lefons and F. Tangorra, Analytic Use of Bitmap Indices. Proceedings of the 6th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases, Corfu Island, Greece, February 16-19, 2007 pp. 159
[9] P. O'Neil and G. Graefe, Multi-table joins through bitmapped join indices, ACM SIGMOD Record 24 number 3, Sep 1995 , pp. 8-11.
[10] K. Wu and P. Yu, Range-based bitmap Indexing for high cardinality attributes with skew, In COMPSAC 98: Proceedings of the 22nd International Computer Software and Applications Conference. IEEE Computer Society, Washington, DC, USA, 1998, pp. 61-67.

El Amin Aoulad Abdelouarit Is a Database administrator in Tanger Med Port, Morocco, PhD Student doing research in Data warehouse and Data mining and it application in Port Management.

Mohamed El Merouani is professor of mathematics, with interests in Probability, Statistics, Stochastic operational research and Data mining, professor of Statistics and Computer Sciences, Poly disciplinary Faculty of Tétouan, Abdelmalek Essaâdi University, Morocco.
University of Granada, Spain, Ph.D. in Mathematics, 1995.

Abdellatif Medouri is Full professor of physics, with interests in Telecommunications, Information theory and Databases; professor of Statistics and Computer Sciences, Poly disciplinary Faculty of Tétouan, Abdelmalek Essaâdi university, Morocco.
University of Granada, Spain, Ph.D. in physics, 1993.