ISSN NO. 2320-5407

Journal homepage: http://www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

A guideline to select indexing technology for Relational Database Management Systems

Adham Mohsen Saeed

Computer Science Dept, AL-Rafidain University College.

Manuscript Info

Manuscript History:

Received: 14 January 2016
Final Accepted: 26 February 2016
Published Online: March 2016

Key words:
RDBMS, database indexing, indexing techniques, indexing methods.

Corresponding Author

Adham Mohsen Saeed.

Abstract

Indexing techniques are utilized for the process of retrieving information form databases. This research presents a comparison among the indexing techniques used in different Database Management System to provide a guide for database developers and DBAs for selecting a suitable indexing technique which is required for the data under consideration .Furthermore, we have compared the results between different types of indexing techniques in Oracle and MS-SQL Server. Even it is very difficult to compare between these two different types of platforms. Finally, after run our experimental tests on these environments, we noted that some indexing techniques in Oracle is faster than MS-SQL server and vice versa, depending on the table size, column selected in the index and SQL statement.

.

Introduction:-

Relational database systems or RDBMSs are used for collecting and maintaining large chunks of data, as to facilitate the storage and timely access to such data. The access to information stored in such databases is conducted with the help of indexing techniques [1,2].

“Indexing processes are particularly useful in the retrieval of complex multimedia information from large databases” [3]. Database indexes can be considered as data structures that have been designed specifically for the purpose making the process of information searching (based on certain criterion) faster [4].

According to the nature of the searching keys used, indexing could be of two types: implicit and explicit [5, 6]. When the location of the target information is obtained as the result of the searching, an implicit method is said to have been put into use [7]. On the other hand, the searching technique results in the confirmation of the presence or absence of any information in the data base, along with its retrieval, is known as explicit indexing technique [8,9].The database indexing techniques that are currently being used are structured as B trees [10]. In such as structure, each page or node in an index has an unique name and the index node contains the reference to all these pages [11]. The utilization of the B tree structure facilitates the process of information retrieval form the database as the SQL servers are able to retrieve data by traversing the branch nodes of the tree and selecting the data according to the search criterion.

B- Tree indexes are two types: i. non clustered indexes and ii clustered indexes. While clustered indexes store the data sets directly in the leaf nodes, the non clustered indexes can be considered as secondary structures capable of pointing information that can be used to point to the exact location of an information in a database [12].

The existing indexing methods and techniques have been tested with the aims of conducting a comparative study of the same: for this purpose, the Oracle platform and the MS-SQL Servers have been used [13]. The Oracle platform

supports the following indexing methods: i. B-Tree Indexing ii. Bitmap Indexing iii. Reverse Key Indexing and iv. Index-Organized Tables [14].

On the other hand, the MS-SQL Servers platform allows the utilization of indexing methods like: i. Non-clustered indexing, ii. Clustered indexing iii. B-tree indexing and iv. Unique indexing [15].

Statement of the Problems:-

With the fast improvement in the field of computer science and engineering, a steady improvement has been noticed in the field of storage devices [16]. Thus, extensive research works are being conducted towards the development of indexing techniques which would result in the fast retrieval of data. This research works aims at providing a proper guidance to the database developers which would facilitate the selection of those indexing techniques which are best suited for the databases designed by them.

Objectives of the Study:-

The primary objectives of this research work are:

i. To analyze the rules (like that of Range SELECT operation, Single row SELECT operation and Full Scan SELECT operation)
and the various criterion, that are used for determining the appropriateness of an indexing technique.

ii. To use a systematic approach of comparing the efficiency of the various readily available indexing methods and techniques. The efficiency of the indexing methods would be tested on Oracle platform whereas the efficiency of the indexing techniques would be conducted on the MS SQL Server platform

iii. To construct a set of guidelines for the database base developers and administrators that would facilitate the selection of the indexing technique appropriate for the particular database structure.

This research work has been conducted with the aim of generating structured guidelines that would facilitate the selection of indexing techniques suitable for a particular database. The guideline would not only be helpful to the database designers and administrators, but would also guarantee the fast and efficient retrieval of information from large databases. This research study is oriented towards the development of structured guidelines, based on experimental results, which would facilitate the selection of appropriate database indexing techniques.

Comparison Criteria &Test Schema:-

Comparison Criteria:-

The following criteria are used to compare the indexing techniques in order to achieve the final results:

i. Overall speed
ii. Space requirements
iii. CPU time required
iv. Memory requirements
v. Measurements of disk traffic such as numbers of seeks and volumes of data transferred
vi. And Ease of index construction

Test Schema:-

Our tests schemas are performed on two platforms: Oracle 10g, and MS-SQL Server and summarized in the following two Figures 1 and 2.

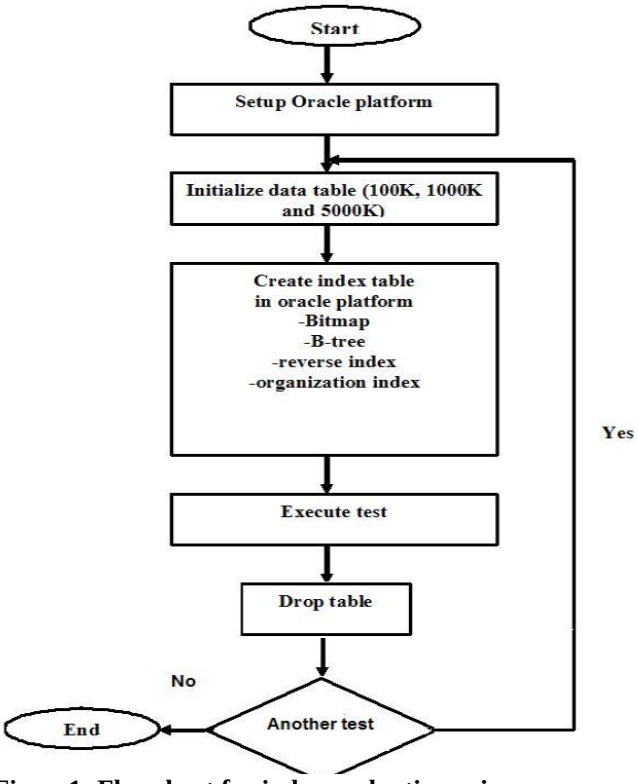

Figure1: Flowchart for index evaluation using The Oracle platform

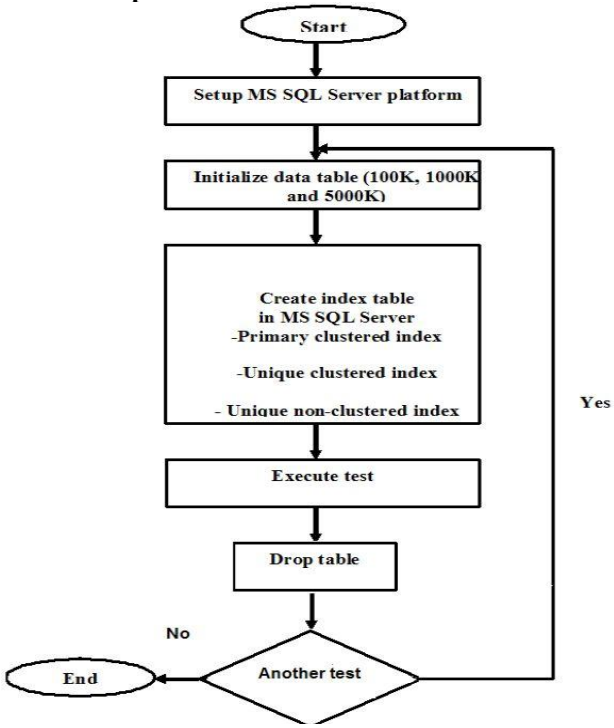

Figure2: Flowchart for index evaluation using the MS-SQL Server platform

Experiments and Test Results:-

The experimental tests was done through measuring the performance(response time) of SELECT operation over Oracle 10g and MS SQL Server on different data table sizes (100K, 1000K, and 5000K).The SELECT test measured the performance on different types of SELECT statements: Single-row, Range, Full scan, and Non-row SELECT performance.

Test Oracle:-

The SELECT test measured the performance on different types of SELECT statements: Single-row, Range, Full scan, and Non-row SELECT performance.

Test 100k:-

Depending on our test on Oracle table with size of 100K and as shown in Table 1, the Bitmap index has the optimal retrieving time and performance compared with other indexing techniques.

Table 1: 100k on Oracle

Indexing Technique	Scan Operation	Data Attribute	CPU Consumption (%)[iii]	Cost[iv]	Response Time (ms)	# of rows retrieved
Bitmap	Full Scan	STU_ID	8	316	4	99996
Bitmap	Index Scan	STU_ID	0 {Big Fraction}	2	2	1
Bitmap	Index Scan	STU_ID	0	2	2	0
Bitmap	Full Scan	STU_GRADE	6	340	6	99996
Bitmap	Index Scan	STU_GRADE	0	2	2	19972
Bitmap	Index Scan	STU_GRADE	0	2	2	0
Unique (B-tree)	Full Scan	STU_ID	6	340	6	99996
Unique (B-tree)	Index Scan	STU_ID	0	5	3	1
Unique (B-tree)	Index Scan	STU_ID	0	5	3	0
Reverse	Full Scan	STU_ID	6	340	6	99996
Reverse	Index Scan	STU_ID	0	5	3	1
Reverse	Index Scan	STU_ID	0	2	2	0
Reverse	Full Scan	STU_Grade	6	340	6	99996
Reverse	Index Range Scan(i)	STU_Grade	0	5	3	19972
Reverse	Index Range Scan	STU_Grade	0	5	3	0
Organization	Index Fast Full Scan(ii)	STU_ID	4	552	8	99996
Organization	Index Unique Scan	STU_ID	0	4	2	0
Organization	Index Unique Scan	STU_ID	0	4	2	1

(i) INDEX RANGE SCAN: Retrieval of one or more row ids from an index. Indexed values are scanned in ascending order.
(ii) INDEX FULL SCAN: Retrieval of all row ids from an index when there is no start or stop key. Indexed values are Scanned in ascending order.
(iii) CPU_COST (NUMERIC) CPU cost of the operation as estimated by the query optimizer's approach. The value of this column is proportional to the number of system cycles required for the operation. For statements that use the rule based approach, this column is null.
(iv) COST (NUMERIC): Cost of the operation as estimated by the optimizer's query approach. Cost is not determined for table access operations. The value of this column does not have any particular unit of measurement; it is merely a weighted value used to compare costs of execution plans. The value of this column is a function of the CPU_COST and IO_COST columns.

Test 1000k:-
Again, our test on Oracle table with size of 1000K and as shown in Table 2, the Bitmap index relatively has the best retrieving time and performance compared with the other indexing techniques.

Table 2: 1000k on Oracle

Indexing Technique	Scan Operation	Data Attribute	CPU Consumption (%)	Cost	Response Time (ms)	# of rows retrieved
Bitmap	Full Scan	STU_ID	8	3018	38	999731
Bitmap	Index Scan	STU_ID	0	6	2	1
Bitmap	Index Scan	STU_ID	0	6	2	0
Bitmap	Full Scan	STU_GRADE	8	3020	38	999731
Bitmap	Index Scan	STU_GRADE	8	3026	38	200720
Bitmap	Index Scan	STU_GRADE	8	3026	38	0
Unique (B-tree)	Full Scan	STU_ID	8	3008	38	999731
Unique (B-tree)	Index Scan	STU_ID	0	8	3	0
Unique (B-tree)	Index Scan	STU_ID	0	8	3	1
Reverse	Full Scan	STU_ID	8	3020	38	999731
Reverse	Index Scan	STU_ID	0	13	3	0
Reverse	Index Scan	STU_ID	0	13	3	1
Organization	Index Fast Full Scan	STU_ID	4	5272	64	999731
Organization	Index Unique Scan	STU_ID	0	4	2	0
Organization	Index Unique Scan	STU_ID	0	4	2	1

3. Test 5000k

Finally, our test on Oracle table with size of 5000K and as shown in Table 3, the performance of Bitmap index, B-tree and Reverse index are much better than Organization index especially at full scan select performance.

Table 3:5000k on Oracle

Indexing Technique	Scan Operation	Data Attribute	CPU Consumption (%)	Cost	Response Time (ms)	# of Rows Retrieved
Bitmap	Full Scan	STU_ID	8	15610	1880	4994346
Bitmap	Index Scan	STU_ID	0	10	3	1
Bitmap	Index Scan	STU_ID	0	10	3	0
Bitmap	Full Scan	STU_GRADE	8	15280	1840	4994346
Bitmap	Index Scan	STU_GRADE	8	15310	1840	998391
Bitmap	Index Scan	STU_GRADE	0	2	2	0
Unique (B-tree)	Full Scan	STU_ID	8	15280	1840	4994346
Unique (B-tree)	Index Scan	STU_ID	0	8	3	1
Unique (B-tree)	Index Scan	STU_ID	0	8	3	0
Reverse	Full Scan	STU_ID	8	15280	1840	4994346
Reverse	Index Range Scan	STU_ID	0	11	3	1
Reverse	Index Range Scan	STU_ID	0	2	3	0
Reverse	Full Scan	STU_Grade	8	15280	1840	4994346
Reverse	Index Range Scan	STU_Grade	0	11	3	0
Reverse	Index Range Scan	STU_Grade	8	15310	1840	998391
Organization	Index Fast Full Scan	STU_ID	2	26694	3220	5001165
Organization	Index Unique Scan	STU_ID	0	2	2	1
Organization	Index Unique Scan	STU_ID	0	2	3	0

Test MS-SQL Server:-
The SELECT test measured the performance on different types of SELECT statements: Primary Key Clustered and B-tree (Unique non-clustered).

Test 100k:-
Depending on our test on MS-SQL Server table with size of 100K and as shown in Table 4, the performance of Primary Key Clustered is relatively better than B-tree (Unique Non-clustered) especially at full scan SELECT performance.
Table 4: 100k on MS-SQL Server

Indexing Technique	Scan Operation	Data Attribute	CPU cost(%)	Operator Cost	Response Time (ms)	# of Rows Retrieved
Primary Key Clustered	Full Scan	STU_ID	11	69	15	99996
Primary Key Clustered	Index Scan	STU_ID	1	1	0.1	1
Primary Key Clustered	Index Scan	STU_ID	1	3	0.4	0
Unique (B-tree) Non-Clustered	Full Scan	STU_ID	11	69	16	99996
Unique (B-tree) Non-Clustered	Index Scan	STU_ID	1	1	0.1	1
Unique (B-tree) Non-Clustered	Index Scan	STU_ID	1	1	0.4	0

Test 1000k

Again, our test on MS-SQL Server table with size of 1000K and as shown in Table 5, the performance of Primary Key Clustered is relatively better than B-tree (Unique Non-clustered) especially at full scan SELECT performance.

Table 5: 1000k on MS-SQL Server

Indexing Technique	Scan Operation	Data Attribute	CPU cost(%)	Operator Cost	Response Time (ms)	# of Row Retrieved
Primary Key Clustered	Full Scan	STU_ID	110	688	11	999999
Primary Key Clustered	Index Scan	STU_ID	1	1	0.4	1
Primary Key Clustered	Index Scan	STU_ID	1	1	0.5	0
Unique (B-tree) Non-Clustered	Full Scan	STU_ID	110	688	14	999999
Unique (B-tree) Non-Clustered	Index Scan	STU_ID	1	1	0.4	1
Unique (B-tree) Non-Clustered	Index Scan	STU_ID	1	1	0.5	0

3. Test 5000k

Finally, our test on MS-SQL Server table with size of 5000K and as shown in Table 6, the Performance of Primary Key Clustered is relatively better than B-tree (Unique Non-clustered) and Unique Clustered especially at full scan SELECT performance.

Table 6:5000k on MS-SQL Server

Indexing Technique	Scan Operation	Data Attribute	CPU cost(%)	Operator Cost	Response Time (ms)	# of Rows Retrieved
Primary Key Clustered	Full Scan	STU_ID	560	351	16	5099800
Primary Key Clustered	Index Scan	STU_ID	1	1	15	1
Primary Key Clustered	Index Scan	STU_ID	1	1	18	0
Unique (B-tree) Non-Clustered	Full Scan	STU_ID	560	378	17	5099800
Unique (B-tree) Non-Clustered	Index Scan	STU_ID	1	1	15	1
Unique (B-tree) Non-Clustered	Index Scan	STU_ID	1	1	18	0
Unique (B-tree) Clustered	Full Scan	STU_ID	560	369	17	5099800
Unique (B-tree) Clustered	Index Scan	STU_ID	1	1	16	1
Unique (B-tree) Clustered	Index Scan	STU_ID	1	1	18	0

Test Analysis:-

This section will show results of the tests that conducted to select the indexing technique. It is very difficult to compare between two different platforms: Oracle and MS-SQL Server because we do not know the background process for each platform. However, we have attempted to run our experiment on the identical environment to

achieve the possible encouraging results. It is clear from the empirical results that the performance of indexing techniques in MS-SQL
Server is much faster than Oracle 10g as shown in Figure 3, 4 and 5using our sample database table (100K, 1000K and 5000K). Therefore, using the index in the SELECT statement over MS-SQL Server is much less costly in terms of I/O operations, CPU consumptions, and response time than Oracle. However, Figure 5 shows that
The B-tree index in the SELELCT statement over Oracle is faster in terms of performance than
MS-SQL Server in the following cases:

- Index scans SELECT performance when retrieved single row.
- Index scan SELECT performance when retrieved non-row.

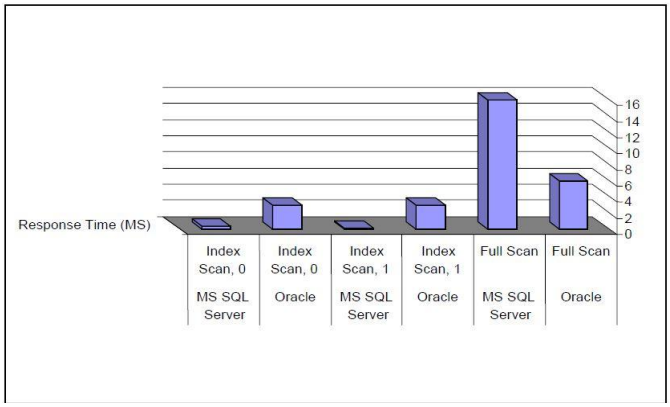

Figure 3: Comparison between the response time (ms) of B-tree over Oracle and SQL Server on 100K

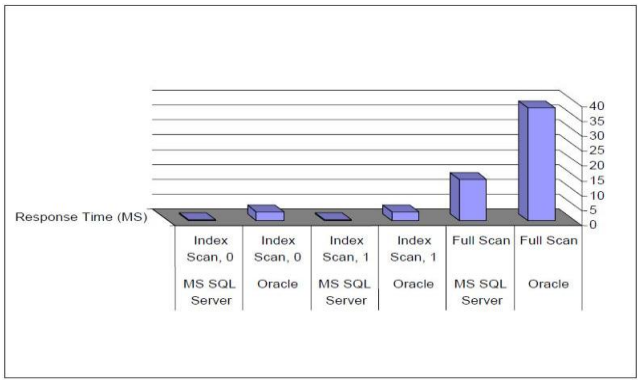

Figure 4: Comparison between the response time (ms) of B-tree over Oracle and SQL Server on 1000K

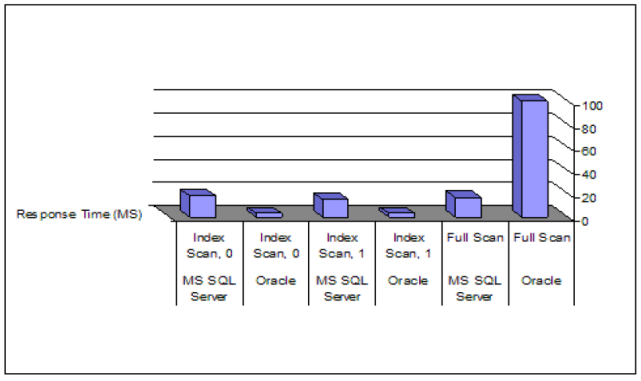

Figure 5: Comparison between the response time (ms) of B-tree over Oracle and SQL Server on 5000K

There are a number of technical reports and studies indicate that the use of the index in the retrieval systems on Oracle platform consumes a larger response time from the MS- SQL server. Thus, support for these studies and from the results of this research,it should be noted that the B-tree is the common between Oracle and MS-SQL Server. As shown in Table7 and 8, the B-tree is the best indexing technique and it is more effective on huge data. The results of Bitmap index in Oracle are interesting and significant and unfortunately, this index is not supported by MS-SQL Server.

Table 7: A summary of the best indexing techniques with different data sizes on Oracle

Data Size	Best Indexing Techniques	Comments
0-100K	Bitmap	Low Cardinality (i) Medium Cardinality(ii)
100-1000K	Bitmap, B-tree	Low Cardinality Medium Cardinality High Cardinality(iii)
1000-5000K	Bitmap, B-tree	Low Cardinality Medium Cardinality High Cardinality

Table 8: A summary of the best indexing techniques with different data sizes on MS SQL Server.

Data Size	Best Indexing Techniques	Comments
0-100K	Non clustered index	Low Cardinality Medium Cardinality
100-1000K	Primary clustered index	Medium Cardinality High Cardinality
1000-5000K	Primary clustered index	Medium Cardinality High Cardinality

(i) Low-cardinality refers to columns with few unique values.
(ii) Medium-cardinality refers to columns with values that are somewhat uncommon. Medium-cardinality column data values such as names, street addresses, or vehicle types.
(iii) High-cardinality refers to columns with values that are very uncommon or unique.

Observations and recommendations:-

After conducting these tests using our sample database table (100K, 1000K and 5000K), we have made the following general observations and recommendations. Thus, it is suggested that
Database DBA's are encouraged to use them as standard recommendations only and validate the applicability of the results to certain target scenario.

1. In Oracle, the Bitmap index on 100K has the efficient retrieving time compared with the other indexing techniques.
2. In Oracle, the Bitmap index on 1000K relatively has the best retrieving time compared with the other indexing techniques.
3. In Oracle with size: 5000K, the performance of Bitmap index, B-tree and Reverse index are Much better than Organization index especially at full scan select performance.
4. In MS-SQL Server with size: 100K, the performance of Primary Key Clustered is relatively faster than B-tree (Unique Non-clustered) especially at full scan select performance.
5. In MS-SQL Server with sizes: 1000K and 5000K, the performance of Primary Key Clustered is relatively better than B-tree (Unique Non-clustered) especially at full scan select performance.
6. Some technical studies indicate that using index in the retrieval systems over Oracle Platform consumes greater response time than MS-SQL Server. Thus, those studies support our results in this thesis.

7. B-tree is the common between MS-SQL Server and Oracle. The results of Bitmap index in Oracle are interesting and significant and unfortunately, this index is not supported by MSSQL Server.

Conclusion:-

Based in the results obtained in the experiments, the following guidelines have been devised for the selection of appropriate indexing techniques:

a. Our methodology is based on a series of experiments to test a set of indexing techniques on two different platforms (Oracle and MS SQL Server) with different data sizes (small – 100K, medium – 1000K, and very large – 5000K) over the same technical environment (Multiple processors, memory, and I/O devices).

b. To run the experiments, we have taken the following indexing techniques in Oracle: B-tree, Bitmap, Reversed, and organization index. In the meanwhile, we have taken the following indexing techniques in MS-SQL Server: B-tree, Clustered index, and unique non-clustered index and Primary Key Clustered index.
c. The empirical results show that the overall performance of indexing techniques (B-tree, reverse, organization, clustered, and bitmap indexes) in MS-SQL Server is much faster than Oracle 10g. Thus using the index in the SELECT statement over MS-SQL Server is less much costly in terms of I/O operations, CPU consumptions, and response time than Oracle. However, the B-tree index in the SELELCT statement over Oracle is faster in terms of performance than MS-SQL Server in the following cases:
i. Index scans select performance when retrieved single row.
ii. Index scan select performance when retrieved non-row.
d. The main outcomes are summarized as follows:
i. Building a referenced guide is to help database developers and DBAs for selecting the indexing method in order to retrieve their data in efficient method. This outcome is very clear because we have shown the best indexing technique on Oracle and MS-SQL Server with different data sizes (100K, 1000K, and 5000K).
ii. Finding the rules and criteria that make the decision of selecting appropriate indexing technique. This outcome is because we have provided a flowchart of methodology procedure and methodology scenarios for conducting the comparisons between the available indexing techniques.

References:-

1. GaffarA. ,(2001), *' Design Of A Framework For Database Indexes* ', Thesis , degree of master of computer science ,Concordia Univirsity.
2. Zobel, J., Moffat, A. and Ramamohanarao, K. (1996), 'Guidelines for Presentation and Comparison of Indexing Techniques'. *SIGMOD Record*, Vol. 25.,PP 10-15.
3. An Oracle White Paper,(2005), '*Technical Comparison of Oracle Database 10g vs. SQLServer 2005: Focus on Performance'*, an Oracle White Paper October , viewed, <http://www.oracle.com/technetwork/database/features/performance/twp-perf-oracle-2.pdf>.
4. Aouiche K., Darmont J., Boussaïd O. and Bentayeb F., (2005) , ' Automatic Selection of Bitmap Join Indexes in Data Warehouses', *Data Warehousing And Knowledge Discovery* , Volume 3589, pp. 64-73.
5. Ozgur A., Taflan I. Gundem, (2006), ' Efficient indexing technique for XML-based electronic product catalogs' , Electronic Commerce Research and Applications, pp. 56–77.
6. Elbassioni K., Elmasry A., and Kamel I., (2003), 'An efficient indexing scheme for multi-dimensional moving objects', *In ICDT*, pp. 422–436.
7. Yu B. and Guoliang L., (2007), 'Effective Keyword-based Selection of Relational Databases', *SIGMOD'07*, Beijing, China. Copyright 2007 ACM .
8. Burleson C., (2010) , '*Bitmapped Index Usage'* , viewed at 1 December 2010 , < http://www.remote-dba.net/t_grid_rac_bitmapped_index_usage.htm> .
9. Nguyen L., Walid G. Aref and Mohamed F. Mokbel, (2003), 'Spatio-Temporal Access Methods: Part 2 (2003 - 2010)', *Bulletin of the IEEE Computer Society TechnicalCommittee on Data Engineering* .
10. Chaudhuri, S., Datar, M., and Narasayya, V. , (2004) ,' Index Selection for Databases', A Hardness Study and a Principled Heuristic Solution. *IEEE Transactions on Knowledge andData Engineering*, 16(11), pp. 1313–1323
11. Fisher D., (1966), 'Data, Documentation and Decision Tables', *CommACM* , Vol. 9 No. 1, pp. 26–31.
12. Lin D., Jensen C., Ooi B., and Saltenis S., (2005), 'Efficient indexing of the historical', present and future positions of moving objects. In MDM, pp. 59–66.

13. Graefe G., (2010) ,'A survey of B-tree locking techniques' , *published in ACMTransactions on Database Systems (TODS)*, Volume 35, Issue 2.
14. Choi I. ,Bongki M. and Hyoung J.,(2007), 'A clustering method based on path similarities of XML data', *Data & Knowledge Engineering 60* , pp. 361–376.
15. Ponce S., Vila P. and Hersch R.,(2002), 'Indexing and selection of data items in huge data sets by constructing and accessing tag collections', *19th IEEE Symposium on Mass StorageSystems &TenthGoddard Conf. on Mass Storage Systems and Technologies Uni†v. of Maryland*, College Park, Maryland, USA, pp. 181-192.
16. Lo J., Barroso L., Eggers S., Gharachorloo K., Levy H., and Parekh S., (1998), 'An analysis of database workload performance on simultaneous multithreaded processors', in ISCA'98: *Proceedings of the 25th annual international symposium on Computerarchitecture*, pp. 39–50.