An Improved Framework for Bitmap Indexes and their Use in Data Warehouse Optimization

Dr. J. Preetha, Dr. S. Lavanya, Dr. T. Kowsalya, Dr. C. Selvi, M. Ganthimathi

[1,2]*Department of Computer Science and Engineering*
[3,4]*Department of Electronics and Communication Engineering*
[5]*Department of Computer Science and Engineering*
Muthayammal Engineering College(Autonomous),Rasipuram,Tamilnadu

Abstract

Compression technique is basically used to compress the size of table or reduce the storage area. Oracle already gives this feature for the table compression as well as for the index compression. when index is created on particular column of a table then it contain some space, which require some storage or disk space by this technique we can save our disk space because in industry the company have to purchase the disk space according to the size of the their data and pay according to their disk space. To utilize this disk space for useful records data rather than wasting it. In this paper used the data pump utility for the compression of Bitmap index and table. Data pump utility performed for the logical backups in database.in this paper implemented data pump for compression, to release the space and change the index pointing location. It will not release the space even after deletion of records. This is of special interest for the case to compress the bitmap index and table space along with the'S (Data Manipulation Language).

Keywords: *Bitmap Index, Compression, Data Pump, DML, DML, Query Performance.*

1 Introduction

Index is the most suitable technique for fast query processing. Here bitmap index is most efficient index in oracle database. This topic basically includes the compression of index. When index is created in oracle database then it occupies the disk space as like other objects in oracle database. Here, the purpose behind compression technique to use data pump and make the index size more efficient. Compression technique is basically used to reduce the size of objects [1]. This technique is used in various fields to save the disk space, compression doesn't reduce the efficiency of object it only reduce the size of objects.

Data pump technique is eligible to reduce all the database objects like table and index. Normally data pump is used in oracle database to take the logical backup of various objects like tables, index, schema, view [2]. But here data pump is using to compress the size of index for this process user will include the import export facility of object. Firstly create the table in the database, insert millions of records and then create index on a particular object [3]. Now, calculate the size of index and size of table in MB, this is the size before the compression. Now, delete some millions of records from the table then commit the statement for permanent changes in database. Now, check the size of index and table. We configure that Index and table is still occupying same space as before, because index is still pointing to same location and those segment which had been occupied by the data is free but pointing of index location is same. So release the space which is occupied by the free segment. Perform logical backup or export the object thorough data pump and now space would be release by free segments.

Now, import the table then find the space occupied by the index or table that would be compressed up to twenty percent according to size of index or table. Now, index would be pointing to new location and the free segment would release the space[4]. Bitmap index is created for the low cardinality that means where the number of distinct values is minimum. So this technique of compression will also apply in the bitmap index.

1.1 Compression

In the present situation the whole world is dependent on the internet, large amount of data are required to run the various organizations successfully. Bulk amount of information are maintaining over a period of time and there is rich content loaded on the internet. Day by day the data is increasing and the growth of the data is almost two times to four times in every years [5]. This sudden increment of data is very challenging task for the administrator. Client has to purchase the space for their data. For manage this problem oracle introduces compression technique. Technologies help the administrator to manage the large amount of data and it provide the resources to the administrator.

This compression technique is useful for almost all objects; basically it is used for OLTP table space where the maximum number of queries and the size of the query is small [6]. In OLTP (online transaction processing) the minimum size of buffers are allocated on the ram suppose if size of data buffer cache is 4GB and db_block_size=2k then 4*1024*1024/2=2097152 buffer would be allocate in RAM. Using compression technique we can compress the RMAN backups, relational data (tables), unstructured data (file).

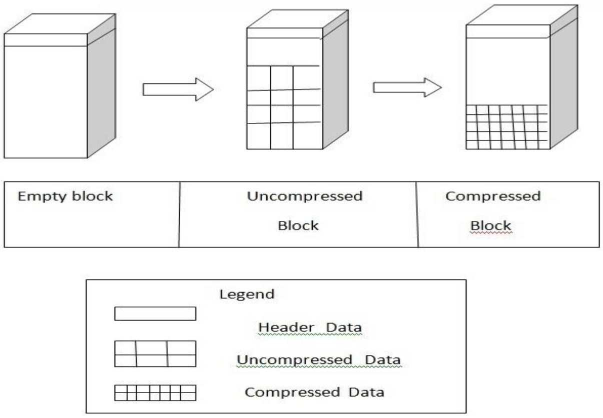


Fig. 1. Compression Technique of Data

2 Literature Review

The idea of object compression was proposed in 9i database. Firstly, in 9i database oracle introduced the table compression [7] feature for high amount of data. Due to the bulk amount of data, there was a requirement of more disk space. Then compression method was successfully implemented to save the disk space. But in 9i database, the compression was possible on the loaded data in database. We cannot implement this technique or compress data along with DML'S applied in database. This features only enabled the compression technique only on the stored data.

Oracle basically develops this technique for the following purpose:

- Improve the query performance on bulk amount of data.
- To propose minimum input /output operation
- For compressed data, buffer cache would be more efficient.

- Efficient compression algorithm for relational database management.
- Compression is performed on the table or partition.
- Data compression is possible on block level.

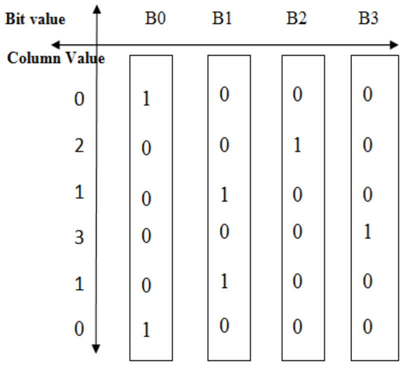
2.1 Compression Behavior

For selection of some records in oracle database, the simplest way evaluating a query is to scan all data records to specific condition. Full table scan sometimes affect the query performance [8,9]. A typical query condition to calculate the number of distinct values in particular column which require full table scan. Indexes accelerate the searching procedure such as variation with Bitmap, Kd-Trees or B-Trees and Bitmap Index on Data Warehouse environment. In database as number of column increases, the number of index possible combination also increases. Bitmap index is widely used in the data warehouse environment [10]. Data pump allows efficient compression [11] of bitmap index after deletion. In this chapter we discuss bitmap index on large table which contains at least millions of records. When user applies bulk deletion from the database and even after fire commit .the index is still pointing to before location. Data pump is utility which basically import export the database object, in industrial application which supports huge amount of data and allow the DML’s very frequently [12]. Large table in database contains the space in Giga bytes. Even the bulk deletion from table, Index pointing location doesn’t change due to the by default behavior of oracle and table and index object still occupying the same space as before deletion. For industry its drawback is that object is still occupying space after deletion. Our objective behind this study is to release the disk space after deletion of records from table.

Bitmap index is the most widespread methodology for fast query handling and is used primarily in data warehouses. We also checked current techniques of data compression and then use the bitmap index technique by data pump. The expert supposition that bitmap index is more effective than standard index. Any bitmap index is created on high level of cardinality or low level of cardinality. We suggest utility for freeing the storage space on database after deletion of documents. Bitmap index also points to the location of records even after deletion of records from chart.

2.2 Bitmap Index Structure

Bitmap index [14] has most efficient structure for single dimensional queries as well as multi-dimensional queries. Logical operations are evaluated with queries, where Individual bitmap value is assigned to each data value. Computer hardware also supports bitmap logical operations. Normally, these operations performed on the bit values which refer data. Here bitmap index with three different values represents the three unique column value.



Column Value	B0	B1	B2	B3
0	1	0	0	0
2	0	0	1	0
1	0	1	0	0
3	0	0	0	1
1	0	1	0	0
0	1	0	0	0

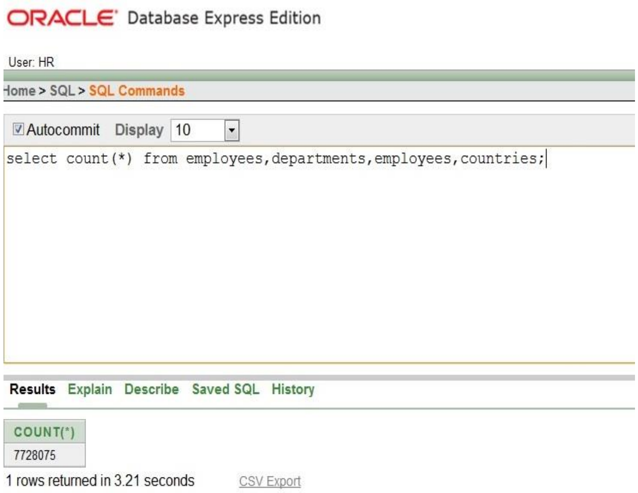
Fig. 2. Representation of Bitmaps of Unique Attribute Value

3 Methodology

3.1 Bitmap Index

A data structure which is used to access large database efficiently is known as bitmap index. The main concept of index is to allot pointers for every row of table [15]. In bitmap index the particular row-id is allotted for every row, in bitmap index the row-id sequentially started from 1.if value is set to 1 then row with correspond row-id contains the key value. Otherwise it is set to zero. Bitmap index are designed for the efficient query processing, queries are processed using bitwise operations like AND, Union, OR operation. For each operation required two same size key value and operator is applied on between them.

This process is used to fast the query like-“sql>select count(*) from the hybrid;”



ORACLE Database Express Edition
User: HR
Home > SQL > SQL Commands
Results Explain Describe Saved SQL History
CSV Export

☑ Autocommit Display 10

```
select count(*) from employees,departments,employees,countries;
```

```
COUNT(*)
7728075
1 rows returned in 3.21 seconds
```

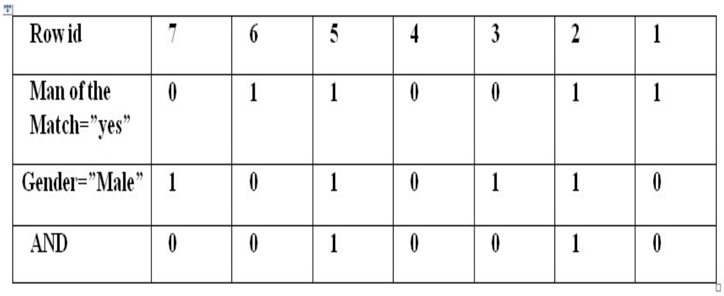
Fig. 3. The Fast Query Execution for the Select Count (*) Query

In this select…count (*) query the query processing speed is fast due to two same size of bitmap value. This process fast execute only those query in which data is not return from the database only counting of database, how many records in the table and how many records with particular identity. Where….or, and….clause.

3.2 Select Query Execution

Sql>select count (*) from the hybrid
Where salary=12000;
For example, to find how many male cricketers, who got the man of the match.

Table 1. For Bitmap (Man of the Match1)=”yes” AND Bitmap (Gender=”Male”)



Rowid	7	6	5	4	3	2	1
Man of the Match="yes"	0	1	1	0	0	1	1
Gender="Male"	1	0	1	0	1	1	0
AND	0	0	1	0	0	1	0

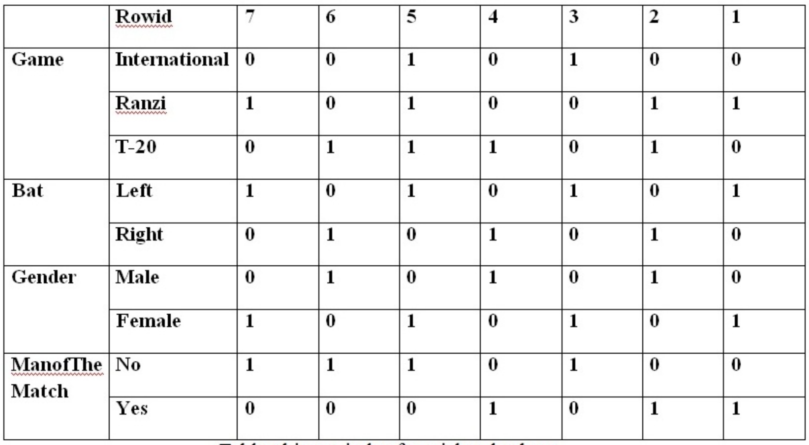
To achieve the decision based tree on the basis of the above example for how many cricketers who are "male" and who won the "man of the match". In this example the bitmap is used to improve the select…count (*) query by applying the AND operation between the two bitmap bits. Basically bitwise AND operation is used by the sql engine. In the above example obtain those records that are in category of "male" and got "man of the match". For the decision node ID3 algorithm would be applied. Here firstly obtain the current node and successive node and apply the AND operation between them.

For obtaining the successful result at the end we count the number of "1" in the final resulting bitmap. Here current node of bitmap represent the value either '0' or '1'.and successive node of bitmap also represents the value either '0' or '1'. Then successively bitwise operation applied. Current bitmap node '0' the person didn't won the man of the match and '1' means the person won the man of the match. Successive node of bitmap '0' means person is female or successive node of bitmap '1' means the person is male.

For the desired output the result after applying bitwise AND operation should be '1', which represents the cricketer who are male and won the man of the match. To elaborate this feature of bitmap index with suitable block diagram where different kind of variety of cricket and different bat style are there to execute how the option are selected on the basis of bitmap index design equation to show decision based tree [16].

To show our approach, let's take the example of cricket database. here the description of the cricketers with the different attributes Game={International,Ranzi,T-20}, Bat={Left handed, Right handed},Gender={Male, Female},Man of the Match={'yes', 'no'}.here on the basis of following attribute we generate the decision based tree.

Table: 2. Bitmap Index for Cricket Database



	Rowid	7	6	5	4	3	2	1
Game	International	0	0	1	0	1	0	0
	Ranzi	1	0	1	0	0	1	1
	T-20	0	1	1	1	0	1	0
Bat	Left	1	0	1	0	1	0	1
	Right	0	1	0	1	0	1	0
Gender	Male	0	1	0	1	0	1	0
	Female	1	0	1	0	1	0	1
ManofThe Match	No	1	1	1	0	1	0	0
	Yes	0	0	0	1	0	1	1

Here we can see the different types of game format as International, Ranzi, T-20, IPL. Bat like left handed or right handed which defines that which types of batsman basically play the types of game

[17]. Here there are two types of player male cricket player or female cricket player at last through the binary number interpretation we will decide whose player have won the man of the tournament in which type of cricket format here ‘1’ indicate the condition is satisfying and ‘0’ indicate the condition is collapsing. Row-id from 1 to 7 denotes the seven player whose id from 1 to 7, whose are player of different kind of cricket format.

3.3 Data Pump

The oracle 10g database basically consist of the three components:

- Expdp and impdp command line tools.
- PL/SQL package DBMS_DATAPUMP.
- PL/SQL package DBMS_METADATA.

These three components by default are called when we use the data pump utility like expdp and impdp command line tool and it is basically introduced in the 10g database. DBMS_DATAPUMP sometimes it is called API (application programming interface) data pump. These are the packages basically consisting set of rules and functions and is automatically called when we use DBMS_DATAPUMP and DBMS_METADATA Sometimes it refers API meta data. This package also consist some set of rules and functions. Here expdp/impdp basically used to export and import the data of particular objects.

This data pump export and import facility represents like original export import utility when we export the data then data pump export generate the dump file and oracle logically write on the directory which is created by the worker. Similarly when we import data then data pump import facility actually generate the dump file. Each utility generates its own file, here the file generated by the export utility would not be useful for import utility.

3.4 Bitmap Index for Data Warehouse Environment

Bitmap index are useful for the Data Warehouse environments, the environment which contains maximum storage or maximum amount of data and large and complex query but minimum number of DML’S (data manipulation language) are applied. For this type of database environment, Bitmap index:

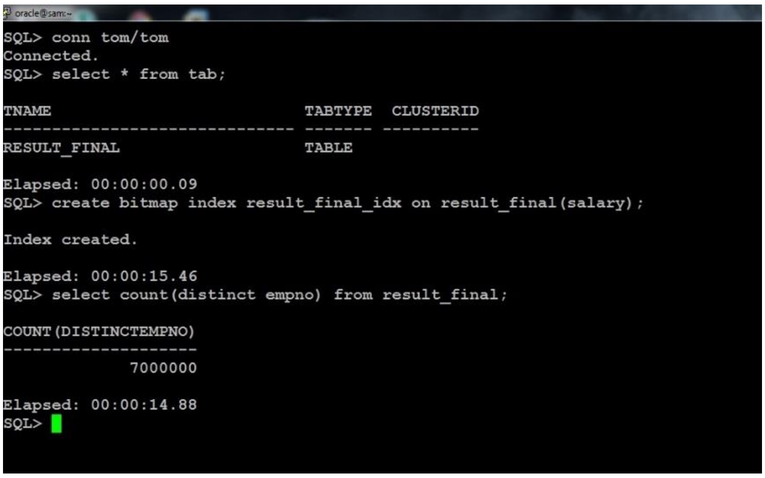
- Reduces the time of response for complex and large query.
- Minimum storage as compared to other indexing technique.
- Performance would be considerable with relatively minimum number of Processors and the CPUs.
- Database maintenance would be efficient during loads.

Index on the large table basically contains the maximum disk space. Sometimes it is also possible in B-Tree index that size of index is more than the table. But it is the excellent feature of bitmap index that it contains far minimum size then the table. Index basically derives the pointer that is generating the key value. Oracle basically provides the OFBR (object file block row) id here index contain the row-id of particular row. Here pointer or key value point to the particular row_id. Bitmap index is most efficient in the data warehouse, the reason is, it contains the large query which is suitable for the bitmap index and normally large query contains the WHERE clause, and it makes bitmap index more efficient for query processing [18]. When we insert millions of records in a particular table then we find there are millions of logs that would be generated because internally only one record is executed at a time. So, for the bulk insertion we should keep database on NOLOGGING mode. Normally in the industries, database are on the archive mode and for the bulk insertion there are lots of redo would be generated so performance of the database would be slow due to the bitmap index. For the data warehouse environment bitmap index is more efficient because it supports minimum updates.

4 Experiments and Results

4.1 Query Performance with the Bitmap Index

Sometime a problem exists in the database that is physical I/O or logical I/O. This is not due to the creation of bitmap index on the database objects or not due to maximum number of distinct values. This problem arises if data is fetched from the disk means from the data file then this case condition would occur of hard parse or physical IO. If data would be available on the RAM then in this case there would be logical IO and the speed of the query would be fast.



```
SQL> conn tom/tom
Connected.
SQL> select * from tab;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
RESULT_FINAL                   TABLE

Elapsed: 00:00:00.09
SQL> create bitmap index result_final_idx on result_final(salary);

Index created.

Elapsed: 00:00:15.46
SQL> select count(distinct empno) from result_final;

COUNT(DISTINCTEMPNO)
--------------------
             7000000

Elapsed: 00:00:14.88
SQL>
```

Fig. 4. Query Performance after Creation of Bitmap Index

When bitmap index created on the table then table's column degree of cardinality should be least, this process enhances efficient bitmap indexing. User has created a bitmap index on the table which contains seventy millions records and chose appropriate column for issuing degree of cardinality. Here we have created bitmap index on the salary column because in the result_final table salary column have minimum number of distinct value range (1000,7000) and now we will fire the select query to find out the number of distinct value in empno column. The basic syntax is used to create the bitmap index on the table.

Syntax- create bitmap index (index name) on (table name)(column name);.

4.2 Features of Bitmap Index Compression

There are two bits that is RID bits and FAST bits plays important role in the index of oracle database. This basically maintains the data in table. Each table is vertically divided into various parts where the rows and columns are stored into different files [19]. Sometimes the large table is horizontally divided into various parts. Different parts of table contain millions of records. This part is maintained as directory. In bitmap index particular column value is shown by the bit vector.

Table: 3. Value of Bitmap Index Vector

Column Value	B0 B1 B2 B3 B4 B5 B6 B7 B8 B9

9	0 0 0 0 0 0 0 0 0
3	1
4	0 0 0 1 0 0 0 0 0
2	0
1	0 0 0 0 1 0 0 0 0
	0
	0 0 1 0 0 0 0 0 0
	0
	0 1 0 0 0 0 0 0 0
	0

Here, length of bit vector is equal to the number of records in the table. If column value is 4 then bit value B4 would be respond to 1 and the rest value would be 0. Due to this feature bitmap index is more efficient for the compression technique. In this technique, a particular bit value is generated for a group of same value. A bit value represents the group of same value in bitmap index. So it decomposes four values and release some space. Actually, in uncompressed model, oracle index points to each and every location, either value is same or is different.

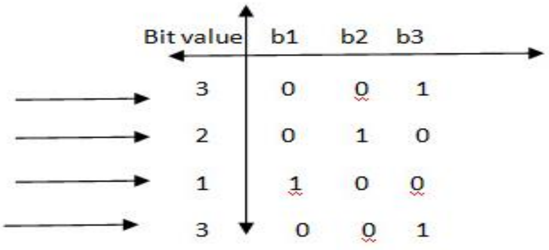


Fig. 5. Pointing of Index on Same or Distinct Values

4.3 Compression Technique in Oracle Database

Oracle database is most efficient database in the world. It contains more advance features in comparison to other database like SQL Server, DB2 and MS-ACCESS. Compression is also advance feature of oracle database. Initially oracle developed the compression feature for table because in the database almost 90 percent objects are stored in table, remaining 10 percent are contained in the Meta data of oracle in the data dictionary. Bulk insertion of data is also performed in the table so its primary task to reduce the table size with the available records. Compression is efficient feature for oracle database in following terms:

- Compression technique basically minimizes the required resources and the cost required to maintain the database.
- It also reduces the storage system from which this space can be used for other operations.
- To release or minimize available network bandwidth, as if size would be more then it will consume more network bandwidth.
- It also minimizes the memory requirements. Because for high amount of data as there is maximum requirement of RAM.

Initially the compression on DML's (data manipulation language) operation was not allowed due the bulk load in the database. After that oracle improves this feature and now data can be compress along with the DML's also. This feature improves the query performance of large data, because there would be fewer inputs /output operation on the database and data buffer cache would be more efficient [20]. Compression process are performed on the block level. New compression process basically includes the features for the OLTP (online transaction processing) database. It supports the compression technique along with the DML's means data would be inserted in compressed form in the database.

4.4 Results & Analysis

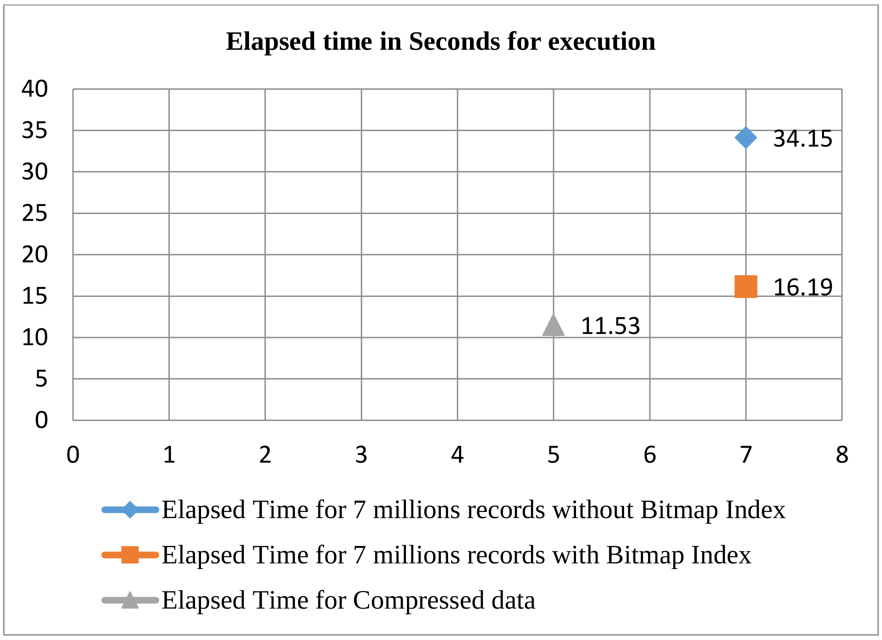


Fig. 6. Graphical Representation of Query Execution Time

From the above results we have:

1 18 percent occupied disk space of Bitmap index RESULT_FINAL_IDX is released by data pump after deletion of one million records.
2 16 percent occupied disk space of table RESULT_FINAL is released by data pump after deletion of one million records.
3 Execution time is calculated 11.53 seconds for five million records compressed data.
4 Execution time is calculated 34.15 seconds for seven million records before creation of index
5 Execution time is calculated 16.19 seconds for seven million after creation of Bitmap Index.

6 Conclusion and Future Work

Bitmap index is most commonly used technique for efficient query processing and mostly in the Data warehouse environment. Data pump is utility which basically import export the database object, in industrial application which supports huge amount of data and allow the DML's very frequently. Large table in database contains the huge amount and occupy the space in Giga bytes. Even the bulk deletion from table Index pointing location doesn't change due to the by default behavior of oracle and table and index object still occupying the same space as before deletion. For industry it's drawback that object is still occupying space after deletion. Our objective behind this study is to release the disk space after deletion of records from table. This article identifies the compression technology and presents the bit-map index compression by data pump. According to traditional wisdom bit-map index is more effective than the more conventional range-based index. To do this it does not require some kind of checkered index heights or degrees; it does not require either. It does not require a bitset. . In this article, we suggest that we need data pump because we do not want to waste the disc space after deleting the records. Bitmap index points to the old position in the database even though records are removed from the table, this utility clears no disc space. In this paper we tested the bitmap index compression algorithm used in SQL Server and the release occupied disc space of database items including tables and indexes contained in the table before and after deleting records. The cloud-based server is now enabling the bulk data addition and deletion. An auto-generated database script will remove millions of records from the table and will not release up the free storage space. Our future studies will result in the allocation of the disc space that was previously used by the removed documents. Oracle database comprises the behavior to occupy the disk space due segment related issue because once segment grow for available records.

References

1. Samy Chambi, Daniel Lemire, Owen Kaser, and Robert Godin. 2016. Better bitmap performance with roaring bitmaps. Software: practice and experience 46, 5 (2016), 709–719.
2. Daniel Lemire, Gregory Ssi-Yan-Kai, and Owen Kaser. 2016. Consistently faster and smaller compressed bitmaps with roaring. Software: Practice and Experience 46, 11 (2016), 1547–1569.
3. Peng Lu, Sai Wu, Lidan Shou, and Kian-Lee Tan. 2013. An efficient and compact indexing scheme for large-scale data store. In Data Engineering (ICDE), 2013 IEEE 29th International Conference on. IEEE, 326–337.
4. Yogeshwari B, Dr.V.Sharmila, Dr.M.Somu, Dr.V.Vennila, Dr.J.Preetha "A Survey on Prediction of Building cost using k means++ Algorithm in Python Framework", International Journal of Innovative Research in Computer and Communication Engineering, Volume 8, Issue 12, December 2020
5. Apache ORC. 2018. Apache ORC, High-Performance Columnar Storage for Hadoop. (2018). Retrieved 2018-10-18 from https://orc.apache.org/
6. Lefteris Sidirourgos and Martin Kersten. 2013. Column imprints: a secondary index structure. In Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data. ACM, 893–904.
7. Idreos S, Groffen F, Nes N, Manegold S, Mullender S, Kersten M. MonetDB: two decades of research in column-oriented database architectures. IEEE Data Engineering Bulletin. 2012;35(1):40-45.
8. Deliège F, Pedersen TB. Position list word aligned hybrid: optimizing space and performance for compressed bitmaps. In: Proceedings of the 13th International Conference on Extending Database Technology; 2010; Lausanne, Switzerland.
9. Van Schaik SJ, de Moor O. A memory efficient reachability data structure through bit vector compression. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data; 2011; Athens, Greece.
10. Byna S, Chou J, Rübel O, et al. Parallel I/O, analysis, and visualization of a trillion particle simulation. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis; 2012; Salt Lake City, UT.
11. Aktas MS, Plale B, Leake D, Mukhi NK. Unmanaged workflows: their provenance and use. In: Data Provenance and Data Management in eScience. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2013:59-81.
12. Lin K-W, Byna S, Chou J, Wu K. Optimizing fastquery performance on lustre file system. In: Proceedings of the 25th International Conference on Scientific and Statistical Database Management; 2013; Baltimore, MD.
13. Ankit Kumar, Dr. Dinesh Goyal, Pankaj Dadheech, (2018), "A Novel Framework for Performance Optimization of Routing Protocol in VANET Network", Journal of Advanced Research in Dynamical & Control Systems, Vol. 10, 02-Special Issue, 2018, pp-2110-2121, ISSN: 1943-023X.

14. Pankaj Dadheech, Dr. Dinesh Goyal, Dr. Sumit Srivastava, Ankit Kumar, (2018), "A Scalable Data Processing Using Hadoop & MapReduce for Big Data", Journal of Advanced Research in Dynamical & Control Systems, Vol. 10, 02-Special Issue, 2018, pp-2099-2109, ISSN: 1943-023X.

15. Preetha J , Raju S , Abhishek Kumarc, Sayyad Samee, and Vengatesan R"Data Mining Technique Based Critical Disease Prediction in Medical Field",Intelligent Systems and Computer Technology, This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).doi:10.3233/APC200126,2020.

16. A. Kumar and M. Sinha (2014), "Overview on vehicular ad hoc network and its security issues," International Conference on Computing for Sustainable Global Development (INDIACom), pp. 792-797. doi: 10.1109/IndiaCom.2014.6828071.

17. Pankaj Dadheech, Ankit Kumar, Chothmal Choudhary, Mahender Kumar Beniwal, Sanwta Ram Dogiwal & Basant Agarwal (2019), "An Enhanced 4-Way Technique Using Cookies for Robust Authentication Process in Wireless Network", Journal of Statistics and Management Systems, 22:4, 773-782, DOI: 10.1080/09720510.2019.1609557.

18. Ankit Kumar, Pankaj Dadheech, Vijander Singh, Linesh Raja & Ramesh C. Poonia (2019), "An Enhanced Quantum Key Distribution Protocol for Security Authentication", Journal of Discrete Mathematical Sciences and Cryptography, 22:4, 499-507, DOI: 10.1080/09720529.2019.1637154.

19. Ankit Kumar, Pankaj Dadheech, Vijander Singh, Ramesh C. Poonia & Linesh Raja (2019), "An Improved Quantum Key Distribution Protocol for Verification", Journal of Discrete Mathematical Sciences and Cryptography, 22:4, 491-498, DOI: 10.1080/09720529.2019.1637153.

20. Ankit Kumar and Madhavi Sinha (2019), "Design and analysis of an improved AODV protocol for black hole and flooding attack in vehicular ad-hoc network (VANET)", Journal of Discrete Mathematical Sciences and Cryptography, 22:4, 453-463, DOI: 10.1080/09720529.2019.1637151.