Applications of Data Compression Approach In Data Warehouse Design

M.K. Sharma*, Jyotsana Sah**
*Amrapali Institute, Haldwani (Uttarakhand), **Kumoun University, Nainital

Abstract - As we enter the era of petabyte scale data warehouses, advanced technologies such as data compression and high-density bulk disk drives are required to design, compress and store a very large data warehouses and to make it cost effective to keep enormous data volumes in the data warehouse. The paper will introduce some applications of data compression on data warehouses to retain history data for longer periods of time. This paper is paced by the affordability of massive warehouses. By recognizing that the performance requirements for data managed by the warehouse are varied, we suggested some technologies that will reduce the effective price in massive data warehouses generation. This paper focuses on data compression, a technology that both reduces the effective price of logical data storage capacity, and improves query performance. Data warehouse design using data compression can experienced up to 50% capacity savings and scan performance can increases up to 35%.

II. TECHNOLOGIES TO REDUCE THE EFFECTIVE PRICE OF DATA WAREHOUSE STORAGE

The technologies we can use to reduce the effective price of data warehouse storage when the data warehouse contains different types of large amount data are:

- High capacity disk drives
- High capacity configurations
- Data compression
- Alternative storage for data

III. THE BENEFITS OF COMPRESSION

Data compression reduces storage cost by storing more logical data per unit of physical capacity. Performance is improved because there is less physical data to retrieve during scan-oriented queries. Performance is further enhanced since data remains compressed in memory and the data warehouse cache can hold more logical rows. The compression algorithm used by us is extremely efficient and the savings in compute resources from reducing disk accesses more than compensates for the CPU used for compression. Another way to improve performance is to use the space saved by compression for advanced indexes [4]. All of these benefits will be explored in the remaining sections of this paper.

A. Lossless compression

The paper will introduce some applications of data compression on data warehouses to retain history data for longer periods of time. This paper is paced by the affordability of massive warehouses. By recognizing that the performance requirements for data managed by the warehouse are varied, we suggested some technologies that will reduce the effective price in massive data warehouses generation. This paper focuses on data compression, a technology that both reduces the effective price of logical data storage capacity, and improves query performance. Data warehouse design using data compression can experienced up to 50% capacity savings and scan performance can increases up to 35% [3].
column. The compressed value is stored in the table header. A
bit field in each row header indicates whether the field is
compressed and whether or not the value is NULL.

Fixed width fields that are not part of the primary index
are candidates for data compression approach. We try to find
out some data types which can compressible. Following is the
list and the native number of bytes used for each data type is
indicated in parenthesis.

- Integer Date (4)
- CHAR (N, where N < 256)
- BYTEINT (1)
- SMALLINT (2)
- INTEGER (4)
- FLOAT/REAL (8)
- DOUBLE (8)
- DECIMAL (1, 2, 4 or 8)
- BYTE (N, where N < 256)

When a column has a frequently occurring value it can be
highly compressed. Some examples include the following:

- NULL
- Zeros
- Default values
- Flags
- Spaces
- Binary indicators (like T/F)

Our data compression approach is completely
transparent to applications, ETL, queries and views. Compression is easily specified when tables are created or
columns are added to an existing table.

For example, here is the syntax for compressing a city
having high population:

```
CREATE TABLE city
(
    Address VARCHAR (40),
    City CHAR (20)
    COMPRESS ('Dehradun'),
    StateCode CHAR (2)
);
```

IV. OPTIMIZATION OF COMPRESSION

We tried to use a free ware cost based optimizer. The
optimizer evaluates the relative cost of many potential plans
and picks a low cost plan. One of the costs considered is the
number of estimated I/O operations needed to execute a plan.
Tables using data compression have fewer physical data
blocks and will therefore need less physical I/O operations to
accomplish certain tasks. Hence, the plan chosen by the
optimizer for a query operating on compressed tables will be
naturally optimized to take advantage of compression.

V. COMPRESSION ANALYSIS RESULTS

The amount of space savings for a particular column is
determined by the percentage of values that can be

As discussed earlier, compression is implemented by
including a bit field in every row header. There is a tradeoff
between the capacity for adding the bit field and the savings
from compression. The following table gives the break-even
point for different field widths. For example, if a column has
an integer data type with a field width of 4 bytes, then if the
most common value (and NULL also) occurs more than
3.13% of the time, then compression will reduce the storage
needed for that column.

<table>
<thead>
<tr>
<th>Field Width (Bytes)</th>
<th>Break Even Frequency of Occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.50%</td>
</tr>
<tr>
<td>2</td>
<td>6.25%</td>
</tr>
<tr>
<td>3</td>
<td>4.17%</td>
</tr>
<tr>
<td>4</td>
<td>3.13%</td>
</tr>
<tr>
<td>5</td>
<td>2.50%</td>
</tr>
<tr>
<td>6</td>
<td>2.08%</td>
</tr>
<tr>
<td>7</td>
<td>1.79%</td>
</tr>
<tr>
<td>8</td>
<td>1.56%</td>
</tr>
<tr>
<td>&gt; 12</td>
<td>&lt; 1%</td>
</tr>
</tbody>
</table>

The graph below shows the break-even trends. As you
can see, for any field larger than two bytes, it takes less than a
5% frequency of occurrence to make compression a better
choice.

The next table shows example
compressibility for fields of 1 and 4 bytes. "Percent
compression" is the savings.

<table>
<thead>
<tr>
<th>Field Size (Bytes)</th>
<th>Compressible Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16%</td>
</tr>
<tr>
<td>2</td>
<td>32%</td>
</tr>
<tr>
<td>3</td>
<td>48%</td>
</tr>
<tr>
<td>4</td>
<td>64%</td>
</tr>
<tr>
<td>&gt; 12</td>
<td>&lt; 1%</td>
</tr>
</tbody>
</table>

Table 2: Compressibility for fields of 1 and 4 bytes
VI. APPLICATIONS OF COMPRESSION APPROACH ON DATA WAREHOUSE IN DIFFERENT AREAS

There are a number of data warehouse applications areas where data compression can help as a good cost saving and performance enhancement experiences.

Retail Industry
- Capacity savings on tables, with over 1 billion rows. Biggest capacity savings is compression of decimal zero.
- Capacity savings on table scan and archive times that reduced proportionately. Large capacity savings on single byte flag fields.

Financial Industry
- Capacity savings and response time can improve.
- Capacity savings and multiload elapsed time [7] can decrease.

Telecommunications Industry
- Capacity savings using compression on spaces and zeros on a financial application
- Capacity savings (over a terabyte) on a table that has terabytes data
- Using compression on millions of rows table with more than 100 columns, I/O can s reduced on a query mix and reduction in CPU time can achieved.

CONCLUSION

The new approach of using data compression for data warehouse delivers a significant storage savings, while simultaneously improving data warehouse performance, especially for scan-oriented queries. This data compression approach is tested by a cost based optimizer. The space saved by compression could be used to reduce storage cost, or to keep more online history, or to further enhance performance with advanced indexing, or to improve availability with increased use of fallback. Data compression is one of new approach that will enable designers of data warehouses as well users of data warehouse to put everything in the data warehouse and keep it forever.

REFERENCES:

BIOGRAPHIES

M.K.Sharma did his M.Tech and now pursuing his Ph.D. Presently is working as Senior Lecturer, in Department of Computer Science, Amrapali Institute Haldwani (Uttarakhand). He has 10 years experience of academics and industry. He has authored 8 books and published international and national research papers. He wrote study material for Chaudhary Devi Lal University, Sirsa and Uttarakhand Open University Uttarakhand and IASE University. He is active member of Computer Society of India and Special Interest Group for E-Governance.

Jyotsana Sah is a research scholar of Kumaun University.