A Performance Evaluation of Vertical and Horizontal Data Models in Data Warehousing

Victor Gonzalez-Castro[1], Lachlan M. MacKinnon[2], David H. Marwick[1]

[1]Heriot-Watt University, School of Mathematical and Computer Sciences, Edinburgh, U.K
{victor,dhm}@macs.hw.ac.uk
[2] University of Abertay Dundee, School of Computing and Creative Technologies, Dundee, UK. l.mackinnon@abertay.ac.uk

Abstract. In Data Warehouse (DWH) environments administrators commonly face the following problems: exponential growth of the DWH; massive storage requirements; excessive long query response times; excessive Extract, Transform and Load data (ETL) times; big batch processing windows to backup and restore the environment; and increasing complexity of DBA tasks. We propose the use of alternative data models utilising a vertical approach to data storage (Binary-Relational, Triple Store-Associative) as opposed to the traditional horizontal storage approach used by the relational model, as a better approach for DWH environments. We present an impartial evaluation of these models using an extended TPC-H benchmark, the extensions taking into account ETL times, Storage requirements, and Backup / Restore times plus Queries response times. These extensions represent common issues in a production DWH environment, and to the best of our knowledge, are not considered in any existing benchmark.

1. Introduction

From the early days of data processing systems through the development of relational databases up to the present day, data has been stored and processed following a horizontal approach, where data is stored in records or relations with *n* number of fields or attributes. This approach has been called the N-ary storage model (NSM) by Copeland [6] and Direct Image Systems (DIS) by Date in [7].

Other researchers have focused on a vertical approach to store and manage the data and abandon the traditional record structure. The idea of vertical storage models is not new, but its application on Data Warehousing environments is novel.

In 1985 G.Copeland published a paper "A Decomposition Storage Model" (DSM) [6] which follows a *Binary-relational* approach. This approach formed the basis on which Boncz et.al. [4][22] developed MonetDB [11] Stonebraker et.al are building C-Store [21] and it is also the base model of SybaseIQ [22]. In the fundamental paper presented in 1988 by G. Sharman [18] they defined the *Triple Store Model* that was further developed and enriched by P.King in the Tristarp project [24]. S. Williams used this work as the basis to create his *Associative Model of Data* and thence build the SentencesDB [25].

Data Mining and Information Systems.
Research in Computer Science 22, 2006, pp. 67-78

Another approach that abandons the record structure and follows a vertical approach is presented in [7] by Date, where the *Transrelational* ™ Model is described. The authors have implemented the essential algorithms and reported its behaviour in [11].

2. Identified Problems

The Relational Model is the predominant model used in commercial DBMS and of the vast majority of companies use Relational products. RDBMS have been demonstrated to be very successful in transactional environments. However RDBMS have since been used to create Data Warehouses without questioning if this is the best approach to manage this type of system. The following problems have been identified in Relational Data Warehouses:

- Data Warehouse grows at an exponential rate [8]
- The Data Base explosion phenomenon [14] is hard to control or eliminate
- Poor management of data sparsity [10]
- Low Data Density [10]
- Huge amounts of disk storage are required [26]
- The cost of storage and its maintenance are not negligible [8] and the storage itself could be up to 80% of the entire system cost [26]
- Long query response times is one of the main user complains, an average 17% of the sites using OLAP tools, with the worst case reaching 42% of the sites that use Oracle Discoverer [17]
- Long periods of time to Extract, Transform and Load (ETL) data [10]
- Big batch processing windows to Backup and Restore the environment [10]
- Increasing complexity of the Data Base Administration tasks

Different approaches (approximate queries [1], materialized views [2], Iceberg cubes [3], dwarf cubes [19], bit map indexes [20]) have been researched to tackle these problems but the fundamental reason has yet to be addressed properly: The horizontal approach used by the Relational model to store data is not the best approach for data warehouses. We propose the use of alternative models that abandon the traditional record structure and follow a vertical approach to store and manage data to be used in Data Warehouse environments. Boncz et al [4] have been working with this approach using a Binary-Relational approach, but it is necessary to benchmark other data models which use vertical approaches, and also all the daily tasks that are involved in a production data warehousing environment. To the best of our knowledge, any of the existing benchmarks consider the whole data warehousing cycle.

In order to do this we propose to extend the TPC-H benchmark and to consider the whole Data Warehousing cycle. The extended benchmark is explained section 3 and the results for each model are presented in section 5.

The Authors have published performance metrics of the behaviour of the alternative data models in [10], [11],[13].

3. Extension to the TPC-H Benchmark

In order to carry out the Performance Evaluation of the selected models in a Data Warehouse Environment, it was necessary to select a benchmark that: can be considered useful to measure different models; well defined; impartial; complete; and general accepted in the research and commercial communities. The following benchmarks have been analysed.

- The 007 benchmark [5] is designed for Object Oriented Data bases, none of the vertical models rely on OODBMS.
- The APB-1 (Analytical Processing Benchmark) created by the OLAP council favours products based on cubes and does not consider relational vendors, and as mentioned before the bigger enterprise Data Warehouses in production are based on Relational Products.APB-1 lacks wide acceptance and has been declining in the last few years [16]
- The Drill down Benchmark [4] designed by Boncz et. al. at the Institute for Mathematics and Computer Science Research at the Netherlands (CWI) been used to benchmark CWI's MonetDB vs. other RDBMS, but lacks wide acceptance.
- The Transaction Processing Council (TPC) has a suite of benchmarks that are widely accepted in industry and have been widely used by the research community. Each of the Benchmarks is targeted to different computing environments but focused on Relational DBMS.

As no existing benchmark satisfies all the criteria, we propose to extend the TPC-H benchmark [23] to consider the complete cycle of an Enterprise Data Warehousing Environment.

Two of the TPC benchmarks (H and R) are targeted to Decision Support Systems which are typical applications on a Data Warehousing Environment. The TPC-H benchmark was chosen because it offered the best constructs to evaluate pristine Data Models and not technology. The only type of auxiliary structures allowed in TPC-H are indexes on primary and foreign keys (it should be remembered that indexes are not part of the Relational model) which makes it more restrictive. In contrast, TPC-R allows the use of extended Relational technology, like indexes over any column, join indexes materialized views, pre-aggregates computation, and practically any technology that the DBMS can have to improve performance.

TPC-H was design to run on Relational based products; its schema consists of 8 tables and a workload of 22 queries which are typical queries in a DW environment. The queries are evaluated with different DW sizes, called the Scale Factor (SF) [23].

TPC-H considers times to load data and execute queries, but it does not consider other tasks that are important in DW environments. The proposed extensions follow the philosophy of the TPC-H Power test, where all queries are executed sequentially. In Table 1 a comparison between the metrics considered in the extended TPC-H and the original TPC-H benchmarks is presented. We refer to a database created without using any auxiliary performance structure as pristine mode.

Table 1. TPC-H and Extended TPC-H pristine models metrics

Metric	Extended TPC-H	TPC-H
Extraction times from transactional systems	Yes	No
Transformation times to conform to the target data model	Yes	No
Input files sizes measurement	Yes	No
Load data times in to the Data Warehouse	Yes	Yes
Database tables sizes after load (Data Base size)	Yes	Yes
Data Density Measurement	Yes	No
Queries execution times (Pristine mode)	Yes	No
Data Warehouse Backup time	Yes	No
Backup size	Yes	No
Data Warehouse restore time	Yes	No

There is another set of metrics that are useless while evaluating data models because they are technology improvements over the relational technology (indexing, statistics computation and query optimizer effects). However, these were measured to provide pragmatic performance metrics for real DW environments (Table 2).

Table 2. TPC-H and Extended TPC-H technology specific metrics

Metric	Extended TPC-H	TPC-H
Index creation times	Yes	Yes
Index sizes	Yes	Yes
Query times (with indexes)	Yes	No
Statistics computation times	Yes	Yes
Query times (with indexes & statistics)	Yes	Yes
Query times (with statistics without indexes)	Yes	No

4. Experimental Design

This experiment covers Relational, Binary-relational and Associative-TripleStore models. The results for Transrelational had been reported by the authors in [11]. All DBMSs are used with the default parameters, further tuning can be done to all DBMSs, but for the objectives of our research we wished to consider raw performance. The TPC-H data set was used with two different scale factors SF=0.1 (100MB), SF=1 (1GB). An ETL tool was developed to read the tables from a relational data source and generate Flat Files according to the structure and characteristics required for the Binary-relational and Associative models. The loading features of the tool are based on the bulk loaders of each target DBMS. The machine used for the experiment has 1 Pentium IV@1.60 GHz, 512 MB RAM, Cache size 256 KB, Bus Speed 100 MHz and O.S. Fedora Core 2 system V 2.4.9-12.

5. Results and Analysis

Extraction, Transformation times and flat files sizes can be considered constants for the tested models, due the fact that differences are small no matter the scale factor used, the results are in Table 3 where a linear behaviour is observed. Relational

measures are included because when building Data warehouses it is common to extract data from Transactional systems built on relational DBMS and loaded into the Data Warehouse (in this case the TPC-H data base).

Table 3. Extraction,Transformation times and input files sizes

	Scale Factor	Relational	Binary-Relational	Associative
Extract. & Transf. time (min)	100MB	2.5	2.6	2.7
Extract. & Transf. time (min)	1 GB	27.6	29.5	27.5
Generated Flat File (MB)	100MB	102.8	98.3	100.0
Generated Flat File (MB)	1 GB	1049.6	1004.9	1021.7

After extraction the corresponding input files were loaded into each target DBMS. **Loading times** are in Fig. 1. For Associative with SF=1GB times for Orders and Lineltem are estimated after loading 300,000 records (Orders) and 10,000 records (Lineltem) because their actual processing times were too long. The best loading times are achieved for the Binary-relational model while the worst times were for Associative, being several orders of magnitude greater (note the use of logarithmic scales on the y-axis).

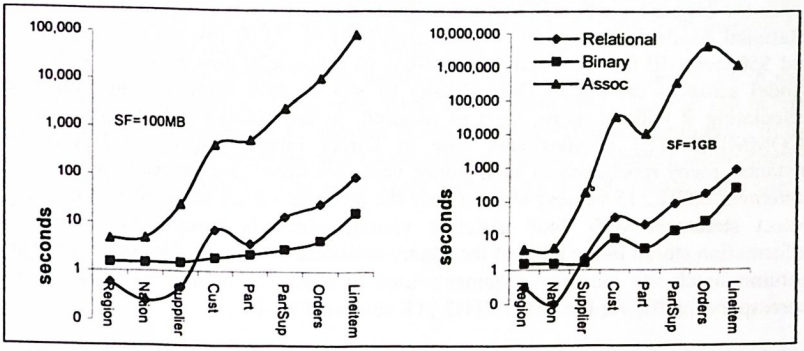

Fig. 1. Loading times

The best savings in time are achieved with the bigger tables. The Binary-relational Model instantiation requires an average of 76% less time to load the data set than the Relational Model instantiation.

When data is loaded into the Binary-relational Model instantiation, size reductions are achieved. The bigger reductions are in bigger tables, whereas in contrast the Associative model produces bigger table sizes (Fig. 2). The total data base size in the Binary-relational model instantiation has savings of 32% compared with the relational model instantiation, no matter the scale factor used, demonstrating linear scalability.

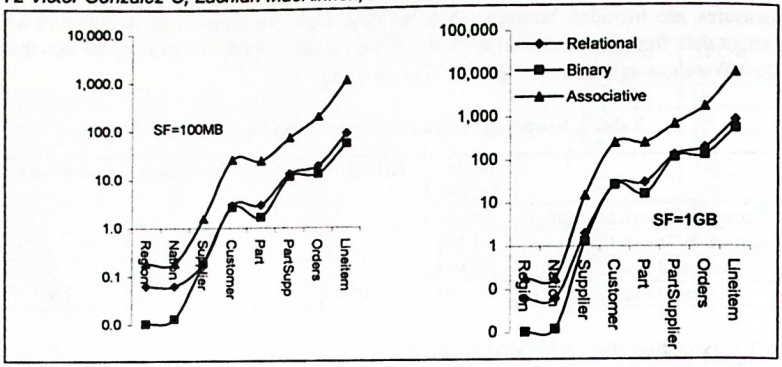

Fig. 2. Table sizes (MB)

Data Density is defined as the number of rows stored per Megabyte on disk. In Fig. 3 relative data density for each data model is presented. Observe the linear behaviour in this metric for each table. The Binary-relational model has the highest Data Density while Associative the lowest no matter the size of the table (Fig. 3).

For the bigger table (Lineitem), the highest Data Density is achieved by the Binary-relational Model instantiation (11,000 rows/MB) vs. 7,000 rows/MB for Relational and 550rows/MB for Associative. Fig. 4-b is an example of how the Binary-relational Model achieves the higher Data Density by storing each value just once and then associating it with as many rows as required. In the example a Date type column (COMMITDATE) demonstrates how in Direct Image systems, Relational for instance, many repetitions of each atomic value are stored (see the result of the select statement 6,001,215 values) while if only the different values are displayed in the 2nd select statement with 2466 different values, which is exactly the amount of information stored in the table of the Binary-relational Model (no duplicate values at a column level; see the unix statements used to count the number of values in the corresponding file for the COMMITDATE column, Fig. 4-b).

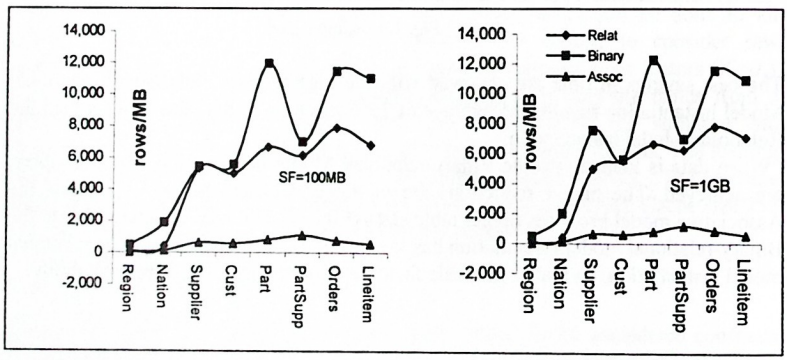

Fig. 3. Relative Data Density (rows/MB)

SQL> select count (L_COMMITDATE)
from LINEITEM;
---------- 	
6001215
SQL> select count(distinct L_COMMITDATE)
from LINEITEM;
---------- 	
2466
	$ strings 34.theap >
	L_COMMITDATE.lis
	$ wc -l L_COMMITDATE.lis
	2466 L_COMMITDATE.lis
	$

(a)	(b)

Fig. 4. Data stored by relational (a) and binary-relational (b) models instantiations

Query Execution Times in Pristine Mode are shown in Fig. 5. As defined earlier pristine mode refers to loading data into the instantiations of the Relational and Binary-relational models without running indexing or statistics computation that have great influence over the RDBMS Query Optimizer, but that are not part of the relational model.

We did not evaluate more metrics for the Associative model, because it has the worst results in loading times and also in the size of the resulting Data warehouse, which are some of the key problems that we are trying to solve by using alternative data models.

The Query Language used was SQL, even though MonetDB offers a native query language called MIL that could thus avoid the translation time from SQL language to MIL, but even with it the query response times were superior.

On Fig. 5 with SF=100MB for the binary-relational model all queries ran in sub-seconds with the exception of 3 queries, while in relational only 3 queries ran in sub-seconds and Query 4 need 31.31 minutes to ran. With SF=1GB all the queries in the binary-relational model ran in seconds (none of them reached 1 minute), while queries in Relational ran in minutes. The worst cases were Query 4 with 14 days and Query 21 with 30 days.

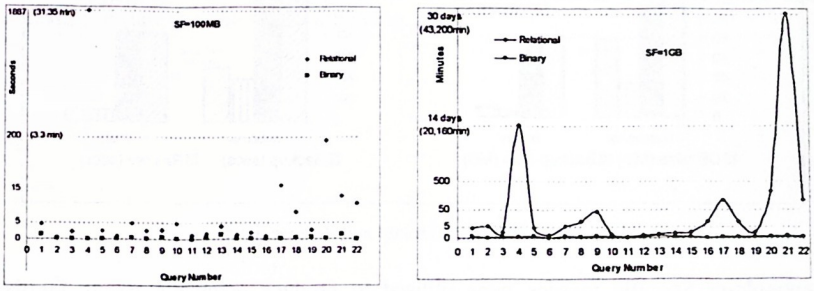

Fig. 5. Query Execution Times in Pristine Mode

The Total Query workload is the total processing time for the 22 queries, as can be seen in Fig. 6, the differences are considerable. With SF=100 MB, binary-relational required 9.5 seconds to process the 22 queries while relational required 36.1 minutes. With SF=1GB binary-relational took 3.8 minutes to process the 22 queries while relational took 43.8 days. These are the times using only relations (tables) as defined

by the relational model and it is the way to compare model achievements, but of course in order to have pragmatic results we measure the queries using extended relational technology (indexes and statistics only) that are used extensively in relational products. The results are presented after the pristine metrics for backup and restore.

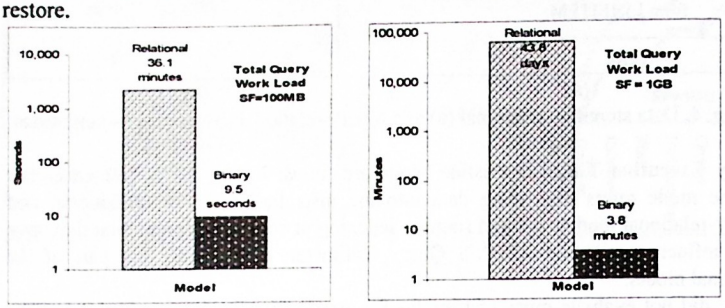

Fig. 6. Total Query Work Load in Pristine Mode

Another important fact that arose while running the queries is that the temporary space required to process the queries by the relational model instantiation grew to 1,487 MB, this size is more than the DB size itself (1,227 MB).

Backup and Restore times are other tasks in a DW environment that are frequently out of the main research focus in the Data Warehouse area. The results are presented in Fig. 7 for different Scale Factors, considering relational and binary-relational models; Binary-relational has better Backup/Restore times than Relational with around 80% less time, no matter the scale factor.

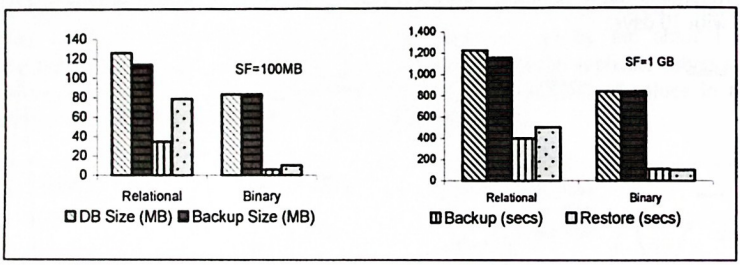

Fig. 7. Backup and Restore times

Technology Specific metrics were utilised to produce pragmatic results for the relational model. One way to improve performance on relational products is by indexing the tables and computing statistics but these need extra processing time and disk space. Fig. 8 shows the total processing time in Relational, which includes Data loading, Index creation and statistics computation. The optimization time (Index + Statistics) is not trivial compared with the required loading time. This optimizing time is not required by the Binary-Relational time. Apart from that, indexes required extra space to be store, Fig. 9 shows the disk space required by indexes.

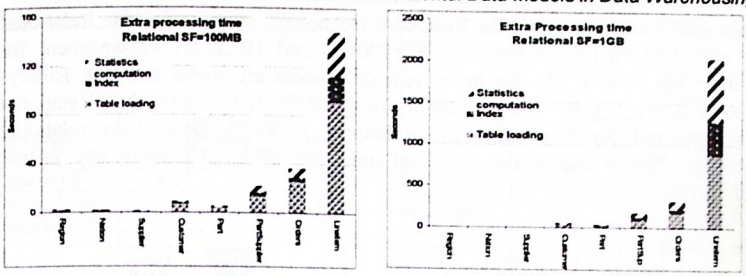

Fig. 8. Extra processing time to improve performance in Relational

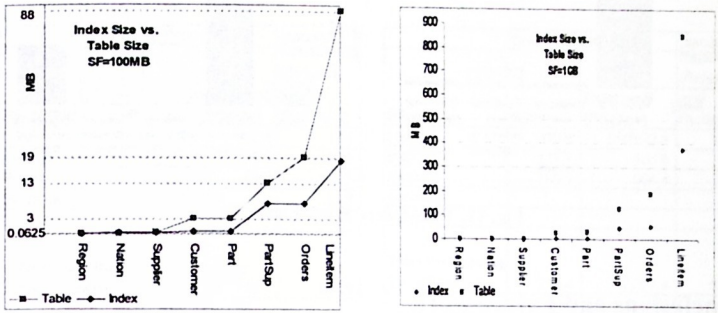

Fig. 9. Index space requirements

After creating indexes and computing statistics in order to help the Query optimizer to produce better execution plans, 3 scenarios were run: Queries executed only with indexes; Queries executed only with statistics; and Queries executed with both indexes and statistics, the results are in Fig. 10.

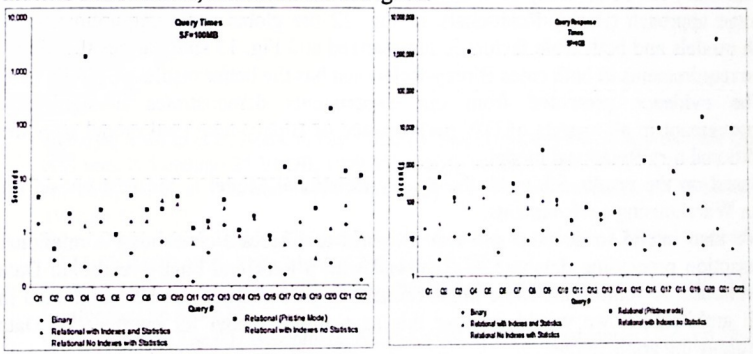

Fig. 10. Query response time with different scenarios to optimize relational

Fig. 11 shows the Total query work load considering the pristine mode of Relational plus the three scenarios described previously. In both scale factors huge

improvements were achieved for Relational technology over the pristine Relational case with 5,887% improvement for SF=100MB and 195,222% improvement for SF=1GB; but in both SF, the better relational times are worse than the Binary-relational times. For SF=100 MB Binary-relational is 388% faster than Relational technology and for SF=1GB Binary-relational is 848% faster than relational technology. Notice that Binary-relational times are achieved without any further optimization.

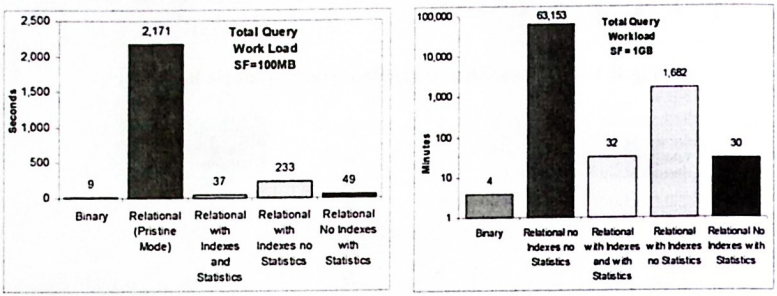

Fig. 11. Total Query Work Load

6. Conclusions and Future work

We are investigating the use of alternative data models which abandon the traditional N-ary approach or Record Structure to store and process data within Data Warehouse environments. According to the results achieved with our extended version of the TPC-H Benchmark, which considers a broader set of tasks that are common in Data warehousing environments, the future of N-ary horizontal storage approach models (Relational) can be certainly be challenged by a better approach which uses a vertical storage approach (Binary-Relational). In Fig. 12 the global space requirements for both models and both scale factors is summarized and Fig. 13 summarizes the global time requirements in both cases Binary-Relational has the better results.

The evidence presented from our experiments demonstrates a significant improvement in all aspects of DW performance of Binary-relational model over the traditional n-ry Relational model.

Based on the results achieved, the Binary-Relational Model is the best option for Data Warehousing environments.

We also intend to develop and test a hybrid architecture combining a relational transaction processing database as front end with a back-end binary-relational Data Warehouse. Assuming the same improvements in performance can be maintained in such architecture, we would propose this as a future model for commercial Data Warehousing applications.

We also are going to test with real life Data Warehouses to verify that the results achieved by the Binary-Relational model are as good as they are with the synthetic data set of TPC-H.

	Space Requirements		(MB)
SF=100MB	Binary	Relational (Pristine)	Relational (Indexes and Statistics)
Input File	98.4	102.8	102.8
Space in DB	83.5	126.3	126.3
Indexes Space	-	-	49.2
Backup Space	83.6	114 0	162.3
Total (MB)	265.5	343.1	440.6

	Space Requirements		(MB)
SF=1GB	Binary	Relational (Pristine)	Relational (Indexes and Statistics)
Input File	1,004.9	1,049.6	1,049.6
Space in DB	838.1	1,227.1	1,227.1
Indexes Space	-	-	489.6
Backup Space	838.1	1,157.0	1,604.0
Aditional Temporal space to run Queries	-	1,497.0	-
Total (MB)	2,681.1	4,930.7	4,370.3

Fig. 12. Total Space Requirements

	Time Requirements		(seconds)
SF=100MB	Binary	Relational (Pristine)	Relational (Indexes and Statistics)
Load	0.5	2.4	2.4
Backup	6.2	34.8	48.4
Restore	10.4	78.9	112.7
Statistics	-	-	51.3
Indexing	-	-	33.1
Total Processing Time(Seconds)	17.1	116.1	247.9
Total Query Response Time (Seconds)	9.5	2,171.2	36.9
Minutes		36.2	

	Time Requirements		(seconds)
SF=1GB	Binary	Relational (Pristine)	Relational (Indexes and Statistics)
Load	303.6	1,192.4	1,192.4
Backup	111.7	400.6	555.0
Restore	103.1	504.4	700.0
Statistics	-	-	937.9
Indexing	-	-	511.5
Total Processing Time(Seconds)	518.4	2,097.4	3,896.8
Minutes	8.6	35.0	64.9
Total Query Response Time (seconds)	228.7	3,789,161.4	1,941.0
Minutes	3.8	63,152.7	32.3
Days		43.9	

Fig. 13. Total Time requirements

References

1. Acharya, Swarup et al. Aqua: A Fast Decision Support System Using Approximate Query Answers. Proceedings 25 VLDB, 1999. Edinburgh, Scotland. pp. 754-757.
2. Agrawal, D. et al. Efficient View Maintenance at Data Warehouses. ACM-SIGMOD 1997.pp 417.
3. Beyer, Kevin, et.al. Bottom-Up computation of Sparse and Iceberg CUBEs. Proceedings ACM-SIGMOD 1999. Philadelphia, USA. pp.359-370
4. Boncz, Peter et al. The Drill Down Benchmark. Proceedings of the 24th VLDB Conference, pp 628-632.
5. Carey, M. et al. The 007 Benchmark. ACM-SIGMOD 1993. Washington USA. pp 10-21.
6. Copeland, George P. Khoshafian, Setrag N. A Decomposition Storage Model. In Proc of the ACM SIGMOD Int. Conf. On Management of Data, pp 268-279, May 1985.
7. Date, C.J. An introduction to Database Systems. Appendix A. The Transrelational Model, Eighth Edition. Addison Wesley. 2004. USA. ISBN: 0-321-18956-6.

8. Datta, Anindya, et al. Curio: A Novel Solution for efficient Storage and Indexing in Data Warehouses. Proceedings 25th VLDB conference, Edinburgh, Scotland 1999. pp 730-733.
9. Gonzalez-Castro, Victor. MacKinnon, Lachlan. A Survey "Off the Record" - Using Alternative Data Models to increase Data Density in Data Warehouse Environments. Proceedings BNCOD 21 Volume 2. pp 128-129. Edinburgh, Scotland 2004. ISBN-1-904410-12-X
10. Gonzalez-Castro, Victor. MacKinnon, Lachlan. Data Density of Alternative Data Models and its Benefits in Data Warehousing Environments. Proceedings BNCOD 22 Volume 2. pp 21-24. Sunderland, England U.K. 2005. ISBN-1-873757-55-7.
11. Gonzalez-Castro, Victor. MacKinnon, Lachlan. Marwick, David. An Experimental Consideration of the use of the Transrelational Model for Data Warehousing. Proceedings BNCOD 23 pp 47-58. Belfast, Northern Ireland U.K. 2006. ISBN-3-540-35969-9.
12. MonetDB web site. http://monetdb/cwi.nl
13. Petratos, Panagogiotis. Michalopoulos, Demitrios (eds). Gonzalez-Castro, Victor. MacKinnon, Lachlan. Using Alternative Data Models in the Context of Data Warehousing. 1st International conference in Computer Science and Information Systems. Athens, Greece. 2005. ISBN-960-88672-3-1.
14. Pendse Nigel. Database explosion. http://www.olapreport.com Updated Aug, 2003.
15. Pendse, Nigel. Multidimensional data Structures. www.olapreport.com . March 19, 2001.
16. Pendse Nigel. OLAP Benchmarks. www.olapreport.com. March 2003.
17. Pense, Nigel. Summary Results from The OLAP survey 4. Microstategy 2005, COLL-0566 0105. pp12.
18. Sharman G.C.H. and Winterbottom N. The Universal Triple Machine: a Reduced Instruction Set Repository Manager. Proceedings of BNCOD 6, pp 189-214, 1988.
19. Sismanis, Yannis, et al. Dwarf: Shrinking the PetaCube. ACM SIGMOD 2002, Wisconsin, USA. pp 646-475.
20. Stockinger, Kurt et al. Strategies for Processing ad hoc Queries on Large Data Warehouses DOLAP 2002. USA. Pp 72-79.
21. Stonebraker, Mike, et.al. C-Store A Column Oriented DBMS. Proceedings of the 31st VLDB conference, Trondheim, Norway, 2005. pp. 553-564.
22. Sybase Inc. Migrating from Sybase Adaptive Server Enterprise to SybaseIQ White paper USA 2005.
23. TPC Benchmark H (Decision Support) Standard Specification Revision 2.1.0. 2002.
24. Tristarp project web site. www.dcs.bbk.ac.uk/~tristarp
25. Williams, Simon. The Associative Model of Data. 2nd Ed, Lazy Software Ltd. ISBN: 1-903453-01-1. 2003. www.lazysoft.com
26. Zukowski, Marcin. Improving I/O Bandwidth for Data-Intensive Applications. Proceedings BNCOD 22 Volume 2. pp 33-39. Sunderland, England U.K. 2005.