ISSN-2230-9659

Study of Self Manage Autonomic Computing With Universal Database

B. Cheena Keshavulu
Research Scholar, Manav Bharti University,
H.P., India

ABSTRACT:-

As the cost of both hardware and software falls due to technological advancements and economies of scale, the cost of ownership for database applications is increasingly dominated by the cost of people to manage them. Databases are growing rapidly in scale and complexity, while skilled database administrators (DBAs) are becoming rarer and more expensive. This paper describes the self-managing or autonomic technology in IBM's DB2 Universal Database® for UNIX and Windows to illustrate how self-managing technology can reduce complexity, helping to reduce the total cost of ownership (TCO) of DBMSs and improve system performance.

1. INTRODUCTION

Database vendors are becoming aware that the human cost of operating large database systems is growing dramatically. As the scope of relational database functions has expanded in recent years, the complexity of database systems has also grown. The added complexity and the increase in data size (now frequently into tens of terabytes) have increased the burden on database administrators.

ISSN-2230-9659

The combination of increased data volumes, larger systems, and increased function, has motivated the need for autonomic capability within database management systems in order to reduce cost of ownership and to enable databases to operate in environments with limited access to skilled administration personnel.

Database designers grapple with complex design issues like the choice of hardware platform; the decision to use a sharednothing, shared-everything, or SMP-cluster hardware topology; the schema design; constraints and referential integrity design; the choice of primary key and indexes; the design of materialized views; the clustering model; and the allocation of tables to disks. Once a database has a physical and logical design, substantial human attention is required to operate it. The numerous tasks include table reorganization, data statistics collection, backup control, security modeling and administration, disaster recovery planning, performance tuning, problem analysis, and others.

In recent years, several research and industry attempts have begun to tackle the enormous task of providing intelligent software tools that reduce the burden on database administrators by providing expert design systems, performance tuning, configuration technology, ease-of-use administration interfaces, and automation tools. A number of early research projects focused on the selection and design of table indexes and clustering, with some initial work on optimizing memory (specifically buffer pool) allocation. More recent projects have examined summary table design, statistics and reorganization prioritization, and constraint modeling. Many of the references in the section 7 contain information on database design technology [1, 2, 3, 4, 5, 7, 9, 10, 12, 13, 14].

However, these initial areas of interest focused on a small subset of the larger problem. There is a dearth of research on what has become one of the most compelling areas of industrial application in RDBMSs – algorithms to enable self-designing, self-administering, and self-tuning RDBMSs. In fact, as databases have become more complex and feature-rich, the human cost of ownership has risen due to the need for DBAs to have more skills and the growing salary demands of skilled DBAs in North America. A 1998 study by the Aberdeen Group supports this observation, showing

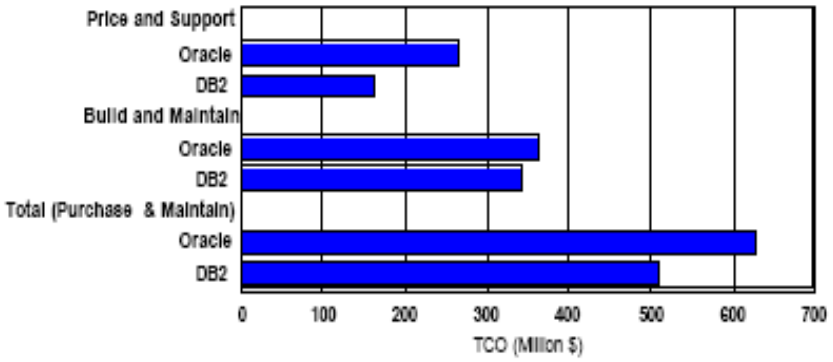
ISSN-2230-9659

that a 5-year 25-user implementation of a leading industrial RDBMS incurred 81% of the TCO in human skills for training, maintenance, and implementation [15].

Another TCO report by D.H. Brown compares the TCO for two industrial database products [6]. This study classified database applications, separating database warehouses from online transaction processing applications. While the human administration costs varied by product and application class, they clearly represented a large component of the TCO in all cases for all users.

D.H. Brown Comparative Study on TCO

OLTP



Recognition of the importance of ease of administration and design tools has spurred renewed interest in research and development of software that reduces the administrative burden. The abundance of papers on index and materialized view (summary table) selection and the development of industrial applications by leading RDBMS vendors such as Microsoft®, IBM®, and Oracle®, as well as tools vendors such as Quest®[18], BMC®[19], DGI®[20], Computer Associates®[21], and others, attest to the growing corporate recognition of this important area of investigation.

As data sizes continue to grow, increasing the demand for large complex systems with more CPUs, more disks, and disk arrays, the need for simplified administration will grow as well. The Asilomar Report on Database Research [8] projects that relational data and unstructured data stored in relational data servers will continue to grow for the next several years.

ISSN-2230-9659

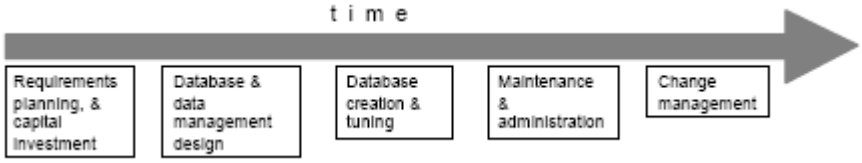
To address these problems we propose a reduction in system complexity and therefore cost of ownership, by introducing autonomic capabilities into the database management system. In his recent research manifesto on autonomic computing [17] Paul Horn notes that autonomic systems are ones which are:

"...capable of running themselves, adjusting to varying circumstances, and preparing their resources to handle most efficiently the workloads we put on them. These autonomic systems must anticipate needs and allow users to concentrate on what they want to accomplish rather than figuring how to rig the computing systems to get them there".

While no commercial DBMS can yet be said to be fully autonomic, we present below components that currently exist within DB2 Universal Database for UNIX and Windows that support autonomic behavior by adding elements of automation and expert advice.

2. SCOPE OF RESPONSIBILITIES

In this section we briefly describe the scope of tasks for a database administrator, providing a very cursory view that serves to illustrate the large scope of tasks and responsibilities incurred today with typical RDBMS products. The scope of tasks is easiest to view when imagined along a timeline, as shown in the following diagram:



During the initial requirements planning and capital investment, the DBA must determine a rough understanding of the performance and storage requirements for the system, and must purchase products that will support these requirements. This includes the selection of the DBMS as well as server and storage devices, a selection process often referred to as "capacity planning."

During the second stage of system development, database designers concentrate on the logical and physical design of the database (table layout, normalization, referential integrity, indexes and

ISSN-2230-9659

materialized views, triggers, and so on), as well as overall process strategy for high availability and disaster recovery, data distribution, security, and user management.

During the third stage of development the database is created, populated, and tuned. Generally there is a substantial period of testing to validate the operation of the new system with applications, and to ensure integration with other systems and operational processes.

During the fourth stage, the system goes into production. During operation, extensive involvement of human operators is needed to monitor system operational health, perform query tuning, maintain data statistics, decluster and fragment data, maintain storage systems, attend to system repairs and outages, modify system design and configuration to account for new operational requirements, or to respond to increasing storage needs.

Recovery needs may require management of periodic backup and archival data. In large distributed systems, data replication (cloning) across systems, and data integrity checking are common tasks. Moreover, almost all of the design and setup operations performed in stages two and three may need to be revisited during operation to account for new requirements, data growth, or poor performance. Many systems often require complex extra-database operations for data extract, transformation and load, and data replication, with these also requiring special tuning and management.

During the fifth stage, database logical or physical design may need to be adjusted for changing application and usage needs. This can require schema changes and changes to system design and implementation as defined in stages two and three. In large, modern DBMSs, these can be daunting responsibilities as database sizes grow to sizes in terabytes, on systems containing hundred of CPUs, thousands of storage spindles, and tens of thousands of database storage objects (relational tables and their associated access structures, including indexes, materialized views, and system catalogs). In the next section we describe existing features within DB2 UDB that are

ISSN-2230-9659

supportive of autonomic computing by providing either automation of tasks, or expert system advice.

3. FUTURE WORK

The current set of features in DB2 UDB only scratches the surface of the larger goal of complete autonomic computing for relational databases. Future releases of DB2 UDB will feature key infrastructure capabilities that will allow DB2 UDB to evolve to the next level of system automation and self-control by allowing a large set of operations, configuration changes, and maintenance utilities to run concurrently with online systems, and provide enhanced monitoring and reporting of system activity and resource utilization. These monitoring and online capabilities have set the stage for research in adaptive resource control, dynamic tuning, automatic maintenance, and adaptive query access plan refinement. Development continues as well in enhanced physical database design technology and self protecting and self-healing technology [26].

Although space constraints and intellectual property considerations restrict a full description of current research projects, the following subset of projects provides some interesting insight into DB2 UDB's future capabilities:

3.1 Learning in query optimization

The cost model used by DB2 UDB's Query Optimizer depends directly on estimates of the number of rows to be processed at each step of the plan. This so-called "cardinality model," in turn, depends upon the statistics on the database, which are used to estimate the selectivity of each predicate in the query.

Updating the statistics after each update would cause a locking "hot spot," so they are instead updated periodically with the RUNSTATS utility. Hence the statistics may be temporarily out of date or incomplete. Furthermore, all query optimizers multiply the selectivities for each predicate

ISSN-2230-9659

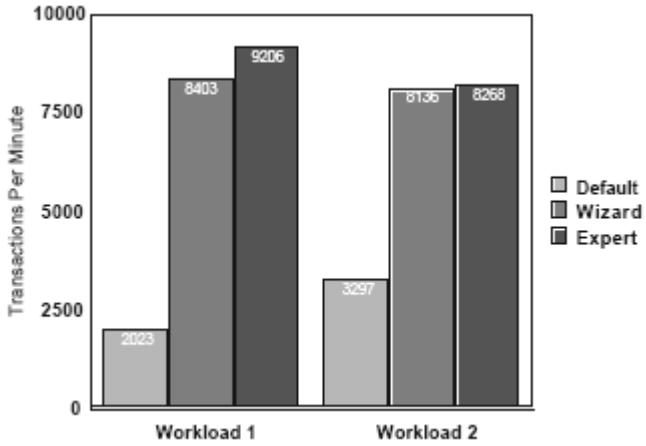
together, essentially assuming that all predicates are independent. This and other assumptions underlying the Query Optimizer model can, when violated, occasionally generate errors significant enough to adversely affect the choice of plan.

As part of the SMART project, we are developing LEO, DB2 UDB's LEarning Optimizer, which automatically self-validates the Query Optimizer's cardinality model. LEO instruments the execution module to collect the actual cardinalities at each step of the plan. After the query completes, LEO compares these actuals to the Query Optimizer estimates, to produce adjustment factors to be exploited by future optimization of queries that have similar predicates. In this way, LEO actually learns from its mistakes over time by accumulating metadata on the database that augments the statistics indicating where data is queried the most. Note that LEO's approach is very general and can correct the result of any sequence of operations, not just the access of a single table or even just the application of predicates [28].

3.2 Advances for the Configuration Advisor

Research and development continues on the Configuration Advisor, enhancing its modeling for a future release of DB2 UDB. Recent experiments with the remodeled algorithms have shown dramatic results, particularly with OLTP and batch database systems. The figure below shows the results of two experiments using an industry standard OLTP benchmark. The experiments were performed on two distinct servers. The diagram illustrates for each experiment how the system throughput was improved over the default settings after running the Configuration Advisor. The throughput was then compared to the performance achieved by a human expert, who was given an extended period of several days to adjust the database configuration for improved performance. In the first experiment, the Configuration Advisor achieved 91.3% of the throughput performance of the system tuned by an expert. During the second experiment, the Configuration Advisor achieved 98.4% of the throughput of the expert-tuned system. In both cases, the system performance after configuration by the Configuration Advisor outperformed the performance of the system with default settings by several factors.

ISSN-2230-9659



These early results suggest that autonomic performance configuration is an achievable goal in the near future for an important class of database workloads.

3.3 Autonomic health assessment & Health Center

Problem detection, determination, and resolution are key tasks for administrators. To address this, an autonomic health assessment engine works with a companion set of problem determination and resolution tools. The autonomic health assessment capability provides a continual monitoring process of system health that is evaluated by collecting a suite of system statistics and comparing the observed system metrics against a predefined policy that defines the health of a system. When some aspect of the system is found to be not completely healthy, a warning or error is raised, and depending on the policy actions an administrator can be notified via e-mail or page of the system concern. Paired with this capability is a Health Center that provides a set of command and graphical interfaces to drilldown into detailed statistics, evaluate the system concerns, and recommend actions.

These paired features combine to form a methodology named “management by exception” where multiple large systems can be managed by requiring intervention only in the event that system health has fallen into a warning or error range. The user is provided with APIs and command level interfaces to adjust the default policies for system health. In this model, problem determination and resolution follows a path of i) problem recognition, ii) determination, and iii) resolution.

ISSN-2230-9659

The autonomic health monitor and Health Center respectively monitor and provide determination capability for items such as system availability and accessibility, storage usage, memory consumption for caching and sorting, logging behavior, and application concurrency.

3.4 Extensions to the Design Advisor

As noted earlier, the current Design Advisor only recommends indexes to create (or drop). But the selection of indexes is only one of the many design decisions that DBAs must make. Materialized views (called Automatic Summary Tables, or ASTs, in DB2) can provide significant performance gains to queries, but consume disk space and require updates when the tables from which they are derived are updated. Judiciously choosing the ASTs to define provides yet another challenge to even knowledgeable DBAs, and of course ASTs as stored tables themselves require indexes to perform well [7]. Furthermore, in a partitioned environment, these ASTs, as well as all base tables, must be optimally partitioned among the nodes to minimize the re-partitioning needed by queries performing joins, aggregates, and so on. Of course, these design decisions interact in ways that are hard for even seasoned experts to predict.

We are currently developing components of the Design Advisor to augment the recommendation of indexes with recommendations similar to those of ASTs and partitionings for each stored table. Both follow the architecture of the current Design Advisor, exploiting the DB2 UDB Query Optimizer detailed cost model to recommend the best candidates for each query, and to evaluate global solutions efficiently. The AST component exploits multi-query optimization [28] to find ASTs that can benefit many queries, rather than simply pre-computing a few key queries. It also optionally uses sampling of the candidate ASTs contents to better estimate its ultimate size. In its RECOMMEND PARTITIONINGS mode, the Partition Advisor exploits the "interesting partitions" already computed by the Query Optimizer for each query, generating alternative plans for each such partition and then letting the Query Optimizer choose the preferred plan as usual. The EVALUATE PARTITIONINGS mode then evaluates all queries in the workload using just one of these candidate partitions for each table, to find the best global solution [29].

ISSN-2230-9659

4. CONCLUSIONS

DB2 UDB is on a clear path towards building a truly autonomic database management system. Several autonomic features exist in the product available today, particularly in support of system integrity assurance, physical database design, and database tuning. A number of additional features are in development that will expand this technology for physical database design, problem determination, and system tuning.

5. REFERENCES

[1] S. Chaudhuri, E. Christensen, G. Graefe, V. Narasayya, and M. Zwilling. "Self-Tuning Technology in Microsoft SQL Server," *IEEE Data Engineering Bulletin*, 22(2) June 1999, pp. 20-26.

[2] B. Schiefer and G. Valentin. "DB2 Universal Database Performance Tuning," *IEEE Data Engineering Bulletin* 22(2) June 1999, pp. 12-19.

[3] K. Brown, M. Mehta, M. Carey, and M. Livny. "Towards Automated Performance Tuning for Complex Workloads," *Proceedings, 20th International Conference on Very Large Databases*, Santiago, Chile, 1994.

[4] G. Weikum, C. Hasse, A. Moenkeberg, and P. Zabback. "The COMFORT Automatic Tuning Project," *Information Systems*, 19(5), 1994.

[5] G. Lohman, G. Valentin, D. Zilio, M. Zuliani, A Skelly, "DB2 Advisor: An optimizer smart enough to recommend its own indexes," *Proceedings, 16th IEEE Conference on Data Engineering*, San Diego, CA, 2000.

[6] D.H. Brown Associates, "DB2 UDB vs. Oracle8i: Total Cost of Ownership," D.H. Brown Associates, Inc., Port Chester, NY., December 2000.

[7] Sanjay Agrawal, Surajit Chaudhuri, and Vivek R. Narasayya. "Automated Selection of Materialized Views and Indexes for SQL Databases," *Proceedings, 26th International Conference on Very Large Databases*, 2000, pp. 496-505.

ISSN-2230-9659

[8] P. Bernstein, M. Brodie, S. Ceri, D. DeWitt, M. Franklin, H. Garcia-Molina, J. Gray, J. Held, J. Hellerstein, H.V. Jagadish, M. Lesk, D. Maier, J. Naughton, H. Pirahesh, M. Stonebraker, and J. Ullman, "The Asilomar Report on Database Research" September, 1998. http://www.acm.org/sigs/sigmod/record/issues/9812/asilomar.html

[9] M. R. Frank, E. R. Omiecinski, S. B. Navathe, "Adaptive and Automated Index Selection in RDBMS," *International Conf. on Extending Database Technology*, Vienna, Austria, 1992, pp. 277292.

[10] H. Gupta, V. Harinarayan, A. Rajaraman, J. D. Ullman, "Index Selection for OLAP," *Proceedings, International Conference on Data Engineering*, Birmingham, U.K., April 1997, pp. 208-219.

[11] A. Silberschatz, S. Zdonik, *et al*, "Strategic Directions in Database Systems -- Breaking out of the Box," *ACM Computing Surveys*, 28(4), December 1996.

[12] S. Chaudhuri, V. Narasayya, "AutoAdmin 'Whatif' Index Analysis Utility," *Proceedings, 1998 ACM SIGMOD Conference*, Seattle 1998, pp. 367378.

[13] S. Chaudhuri, V. Narasayya, "Microsoft Index Tuning Wizard for SQL Server 7.0," *Proceedings, 1998 ACM SIGMOD Conference*, Seattle 1998, pp. 553554.

[14] S. Finkelstein, M. Schkolnick, P. Tiberio, "Physical Database Design for Relational Databases," *ACM Transactions on Database Systems*, 13(1), March 1988, pp. 91128.

[15] "Database Cost of Ownership Study," The Aberdeen Group 1998. http://relay.bvk.co.yu/progress/aberdeen/aberdeen.htm

[16] R. Winter, K. Auerbach, "The Big Time: The 1998 Winter VLDB Survey Program winners are bigger, better, and overwhelmingly relational," *Intelligent Enterprise Programming and Design Online*, http://www.dbpd.com/vault/9808win.html

[17] P. Horn, "Autonomic Computing: IBM's Perspective on the State of Information Technology", http://www.ibm.com/research/autonomic, International Business Machines, Armonk, NY, 2001

[18] "Quest software central for DB2 " http://www.quest.com/quest_central/db2/

[19] BMCSoftware, Products and Services, SmartDBA. http://www.bmc.com/ddm/index.html

ISSN-2230-9659

[20] DGI Achieving Breaktrough Results for DB2, http://www.breakthroughdb2.com/

[21] Computer Associates, Enterprise Management, http://www3.ca.com/Solutions/Solution.asp?ID=335

[22] H. Pirahesh, J. M. Hellerstein, W. Hasan, "Extensible/Rule Based Query Rewrite Optimization in Starburst", Procs. 1992 ACM SIGMOD Conference, 1992, pp. 39-48.

[23] H. Pirahesh, T. Y. C. Leung, W. Hasan, "A Rule Engine for Query Transformation in Starburst and IBM DB2 C/S DBMS", Procs. 1997 IEEE Intl. Conf. On Data Engineering, 1997, pp. 391-400.

[24] P. Gassner, G. M. Lohman, K. B. Schiefer, Y. Wang. "Query Optimization in the IBM DB2 Family", IEEE Data Engineering Bulletin 16(4), 1993, pp. 4-18.

[25] M. Zaharioudakis, R. Cochrane, G. Lapis, Hamid Pirahesh, M. Urata, "Answering Complex SQL Queries Using Automatic Summary Tables", Procs. 2000 ACM SIGMOD Conference, 2000, pp. 105-116.

[26] D. C. Zilio, S. Lightstone, K. A. Lyons, G. M. Lohman, "Self-Managing Technology in IBM DB2 Universal Database", Procs. Of 2001 CIKM, 2000, pp. 541-543.

[27] M. Stillger, G. M. Lohman, V. Markl, M. Kandil, "LEO - DB2's LEarning Optimizer", Procs. 27th Intl. Conf. On Very Large Databases, Rome, Italy, 2001, pp. 19-28.

[28] W. Lehner, R. Cochrane, H. Pirahesh, M. Zaharioudakis, "fAST Refresh using Mass Query Optimization", Procs. 2001 IEEE Intl. Conf. On Data Engineering, 2001, pp. 391-398.

[29] J. Rao, C. Zhang, G. Lohman, N. Megiddo, "Automating Physical Database Design in a Parallel Database System", Proc. 2002 ACM SIGMOD Intl. Conf. On Management of Data, Madison, WI, 2002 (to appear).