A Comparative Study on Oracle and IBM-DB2 Databases

Lavanya Pamulaparty P.Vijaya Babu Varma T.Praveen Kumar

Abstract - As storage of data plays a key role in databases, security and maintenance issues becomes major concerns. Relational databases hold a significant portion of data stored in software, therefore today's database purchase decisions revolve around how secure the product is. This paper provides a categorical head to head feature comparison between Oracle Database (Oracle) and IBM DB2 Universal Database (DB2), in terms of performance, scalability, manageability and consistency features provided in the Secure Way with emerging trends in technology. It explores the impact of IBM's and Oracle's transaction security models on users seeking to protect their critical information systems and contrasts IBM's strategy of building security outside of the DB2 database against Oracle's strategy of securing information in the database server[6].

Index terms: Database security, Oracle Security, DB2 Security, Database Tuning, Partitioning, Database Cost.

Lavanya Pamulaparty Head and Associate Professor
Dept. of CSE, MCET
Hyderabad, INDIA
Lavanaya.p@methodist.edu.in

P.Vijaya Babu Varma Assistant Professor
Dept. of CSE, MCET
Hyderabad, INDIA
vijayvarma.p@methodist.edu.in

T.Praveen Kumar Assistant Professor
Dept. of CSE, MCET
Hyderabad, INDIA
praveenkumar.t@methodist.edu.in

1. Introduction

Oracle Database regularly outperforms its competitors on a wide variety of industry-standard and ISV-specific benchmarks, and is widely recognized as the industry leader in database scalability [12]. As IT vendors deliver increasingly sophisticated solutions to meet the high demands of grid computing, the task of systems management has never been more complex. Hiring highly skilled administrative staff to manage such complicated environments is an expensive proposition. This, coupled with frequent shortage of experienced administrative personnel, often results in spiraling management costs. In order to meet these challenges, Oracle has made the manageability of its products one of its primary goals. Oracle Database automates a number of key administrative tasks, reduces the complexity of administration and provides self-tuning capabilities that deliver optimal performance out-of-the box. Security is at the core of the coding practices employed by the development staff that builds the Oracle database, resulting in the delivery of a secure product [8]. IBM addresses security by delivering it outside of the database and relying on the operating system or Tivoli's product line to secure DB2 and other IBM products. The most obvious result is that data stored in DB2 is not inherently protected. This paper focuses on the major techniques commonly used to ensure good performance and scalability in modern enterprise-class, relational database systems: concurrency model, partitioning, security and tuning. Finally, it briefly compares both products in terms of manageability of cost and performance.

2. Methodology Overview

We study the feature of scalability based on methodology developed by Edison groups on oracle and IBM databases for making product management comparisons [12]. The result is a summary of cost comparisons incurred by any corporate IT departments or ISV running either of these two products. In our study, Oracle and IBM DB2 were compared against a set of methodology metrics in order to determine which of the two products is easier to operate for businesses with real-world database management requirements. The

Task areas that we used to perform this study fall into the following categories:
1. Data centric manageability
2. Cost manageability
3. Performance manageability

3. Data Centric manageability

As the data play a vital role in databases, with superior and innovative technology oracle designed the first database for enterprise grid computing which reduces the cost of data management while providing the highest quality of service. It allows IT to rapidly respond to the needs of the business while greatly lowering the risk.

A. Database transaction models

Transaction processing applications are characterized by very large user populations concurrently accessing large volumes of data for short and frequent insert or update transactions. Such environments require support for high throughput, a choice between several indexing strategies, and excellent data concurrency.

Concurrency Model

Oracle Database and DB2 greatly differ in their implementation of concurrency control. Oracle fully supports mixed workloads environments characterized by simultaneous query and update activities. With Oracle Database, writers never block readers and readers never block writers. Non-blocking multi-version read consistency always provides users with consistent query results while never imposing a performance penalty on concurrent update activity. DB2 lacks Oracle's powerful multi-version read consistency and forces users to choose between accuracy and concurrency. This means that DB2 users must either block writers in order to ensure read consistency or accept inaccurate results, i.e., dirty reads. The basic architecture of Oracle is very efficient for managing large numbers of transactions. The technical feature that makes this possible is Oracle's patented non-escalating row-level locking.

Oracle	DB2
Multi-version-read	Not available
No read locks	Requires read locks to avoid
No dirty reads	Dirty reads if not using read
Non-escalating-row-level	Locks escalate
Readers don't block writers	Readers block writers
Writers don't block readers	Writers block readers
No deadlocks under load	Deadlocks can be a serious

Table 1: Concurrency Models

Multi-version read consistency

Database implementations differ in their ability to prevent well-known phenomena encountered in multi-user environments:

• Dirty, or uncommitted reads happen when a transaction can read changes made to the database that have not yet been committed.
• Non-repeatable reads occur when a transaction re-reads data it has previously read and finds that another committed transaction has modified or deleted the data.
• phantom reads happen when a transaction executes twice a query returning a set of rows that satisfy a search condition, and finds that the second query can retrieve additional rows which were not returned by the first query, because other applications were able to insert rows that satisfy the condition.

Oracle's implementation of multi-version read consistency always provides consistent and accurate results. When an update occurs in a transaction, the original data values are recorded in the databases undo records. Rather than locking information to prevent it from changing while being read, or to prevent queries from reading changed but uncommitted information, Oracle uses the current information in the undo records to construct a read-consistent view of a table's data, and to ensure that a consistent version of the information can always be returned to any user.

DB2 does not provide multi-version read consistency. Instead DB2 requires applications either to use read locks, with various levels of isolation, or to accept dirty reads. Read locks prevent data that is read from being changed by concurrent transactions. Clearly, this implementation restricts the ability of the system to properly service concurrent requests in environments involving a mix of reads and writes. The only alternative users have is to build separate workload environments. The result is that DB2 users always have to find some compromise in their application design in order to get acceptable data concurrency and accuracy. For an example of how this affects application development, consider SAP. In order to avoid the disastrous effects read locks could have on concurrency, SAP has to compensate for DB2 dirty reads. This is done through additional code implemented in the database-dependent layer of the SAP interface. In the Oracle interface for SAP, nothing extra has to be done to ensure read consistency since the database server takes care of it.

Non-escalating row-level locking

Row-level locks offer the finest granularity of lock management, and thus, the highest degree of data concurrency. Row-level locking ensures that any user or operation updating a row in a table will only lock that row, leaving all other rows available for concurrent operations.

Oracle uses row-level locking as the default concurrency model and stores locking information within the actual rows themselves. By doing so, Oracle can have as many row level locks as there are rows or index entries in the database, providing unlimited data concurrency.

DB2 also supports row-level locking as the default concurrency model. However, because it was not the initial default level of lock granularity in earlier versions of the database, the late addition of row-level locking was made

possible only through the use of additional, separate memory structures called lock lists. As for any memory structures, these lock lists have limited size and thus impose a limitation on the maximum number of locks that can be supported by the database.

B. Database partitioning

Partitioning allows large database structures (tables, indexes, etc.) to be decomposed into smaller and more manageable pieces [12]. Partitioning can help improve performance with the technique known as partition pruning. Partition pruning enables operations to be performed only on those partitions containing the data that is needed. Partitions that do not contain any data required by the operation are eliminated from the search. This technique dramatically reduces the amount of data retrieved from disk and shortens the use of processing time, improving query performance and resource utilization.

Oracle's partitioning options

Oracle Database offers several partitioning methods designed to be more appropriate for various particular situations:

• Range partitioning uses ranges of column values to map rows to partitions. Partitioning by range is particularly well suited for historical databases. Range partitioning is also the ideal partitioning method to support 'rolling window' operations in a data warehouse.

• Hash partitioning uses a hash function on the partitioning columns to stripe data into partitions. Hash partitioning is an effective means of evenly distributing data.

• List partitioning allows users to have explicit control over how rows map to partitions. This is done by specifying a list of discrete values for the partitioning column in the description for each partition.

Feature	Oracle	DB2
Range partitioning	Yes	-
List partitioning	Yes	-
Hash partitioning	Yes	Yes
Composite partitioning	Yes	-
Local index	Yes	Yes
Global partitioned index	Yes	-
Global non-partitioned index	Yes	-

Table 2: Partitioning options

DB2 only supports the hash partitioning method, which has considerable limitations and weaknesses when compared to Oracle's partitioning capabilities.

Unlike range or list partitioning, hash partitioning does not allow typical queries to take advantage of partition pruning. -By supporting more partitioning options for tables as well as indexes Oracle is able to prune partitions in more queries. -By only supporting hash partitioning, DB2 does not allow for 'rolling window' support. With this process, a data warehouse is periodically kept up to date by loading new data and purging old data in order to always keep the most recent data online.

C. Database security

IBM researchers developed the Data Encryption Standard (DES). The security model they choose to secure the database, however, has flaws that impact their customers. The DB2 security model favored by IBM hurts customers in three ways:

• A less secure database, more vulnerable to users or hackers subverting the security due to the security model that adds security after the fact. It is difficult to add layers of security after a product has been designed, coded and shipped [8].

• Higher up-front costs because of the additional products necessary to secure DB2. Customers

• Higher long-term cost of ownership because customers must pay for the database product

IBM has delivered an introductory database encryption capability in the most recent release, DB2 UDB 7.2, available since June 2001. DB2 has functions that enable an application to encrypt and decrypt data using an RC2 block cipher with a 128-bit key and using an MD2 message digest. It provides column-level encryption, enabling all values in a column to be encrypted with the same key— an encryption password. First delivered in Oracle in 1999, Oracle provides an encrypt/decrypt interface to encrypt especially sensitive data in the database server. Oracle has been enhancing the database encryption solution over the years, adding in Triple-DES encryption and MD5 cryptographic checksums in a subsequent Oracle8i release [2]. The first Oracle9i release enhanced the Random Number Generator (RNG) to use a FIPS 140 Level 2-certified RNG, another example of security with assurance. In the current release, Oracle provides DES (56-bit), 2-key and 3-key Triple-DES (112- and 168-bits, respectively) in an encryption toolkit package that enables applications to encrypt data within the database. The IBM solution is password-based; the user supplies a password as the encryption key to encrypt and decrypt data [2].

Network Encryption

DB2 database itself does not provide network encryption to secure communications between any client and the database, but IBM does support DES and RC2 in the network. Oracle offers Oracle Advanced Security to protect all communications with the Oracle Database. Wherever the database is available, Oracle9i Advanced Security is available and ships on the same media as the database software [2]. To encrypt network traffic, it provides Secure Sockets Layer (SSL). [11] The Internet standard offers:

• RC4 in 256-bit, 128-bit, 56-bit, and 40-bit key lengths,

• DES in 56-bit and 40-bit key lengths,

• 2-key or 3-key Triple-DES (3DES) with 112-bit and 168-bit keys, respectively, which is especially high-strength encryption. These cryptographic modules have undergone the laborious certification process to claim Federal Information Processing Standard (FIPS 140-1) Level 2

compliance, providing assurance of the implementation down to the randomness of key generation [2].

D. Database tuning

To start the Tuning exercise, we need to understand how the existing system is dealing with its content. This includes:

· Document types and counts

Documents should be grouped and categorized by type (for example, invoices or claims). These groupings will be based on the way the documents are to be treated for security, and indexing data and storage requirements. The groups will probably translate to Content Manager item types.

· Document arrival rate

When, where, and how many documents are entering the system? Although annual document volumes are often used as a benchmark, it is dangerous to apply these statistics without knowing document arrival peaks and valleys. It is critical to identify the peak arrival rate and size the system accordingly. It is also useful to understand the locations and formats of the documents that can be captured. Are all application forms sent to a central mail room and processed there, or are they sent to individuals in branch offices?

· Document usage

When, where, and how is each kind of document used? For example, one type of form might go through a business process, following a specific workflow. The workflow process comprises seven different activities by different people. At each workflow step, this document is viewed by the person performing the activity. This example means that each form is retrieved at least seven times. That does not seem like much, but if there are 10,000 of these forms arriving each day, then there are 70,000 retrievals happening each day.

Storage tuning guidelines:

File cache management, file system architecture, and volume management translate application calls into individual storage access requests. These requests traverse the storage driver stack and generate streams of commands that are presented to the disk storage subsystem. The sequence and quantity of calls, and the subsequent translation, can improve or degrade performance. In order to improve disk I/O usage; Distribute I/O to multiple disks. Use external storage devices with large numbers of physical drives if possible. External storage devices deliver higher performance and reliability for demanding applications in data-intensive computing environments. · Internal built-in disks should be used only for operating systems and program object files.

Network tuning guidelines:

· Set the MTU size to a setting appropriate for your network. All servers and switch ports in the environment must be set the same. Optimizing the MTU setting can improve network performance.

· Disable auto-negotiation for 10/100 Ethernet cards and set them to a fixed media speed and duplex mode. It is also imperative that the settings match at the switch port. ·

4. Database cost manageability

The core premise of any Comparative Management Cost Study is that the true cost of owning and operating complex systems like Oracle and IBM DB2 only start to accrue after the product has been purchased. In most real-world business environments, the management costs will far outweigh the licensing and support costs throughout the life of the product. With this in mind, we estimated the annual costs that businesses can expect to save due to the DBA-related time savings that result from one product being easier to administer and operate than the other [12].

In order to compute cost savings, we used DBA salary figures published by Enterprise Systems. Further information on these salary figures can be obtained from: http://esj.com/it_info_center/article.aspx?EditorialsID=27.

From Enterprise Systems survey: The median total compensation including benefits for a typical Database Administrator in the United States is **$83,300. 4** This basic market pricing report was prepared using analysis of survey data collected from thousands of HR departments at employers of all sizes, industries, and geographic regions.

If we insert the median DBA compensation salary found in the Enterprise Systems survey into the formula below, we arrive at the following quantitative management cost (MC) saving calculation.

Median DBA Salary * (DBA time savings) = $83,300 *38% = $31,654

This result can be interpolated to match to your company's DBA salary expenses by applying the above formula. When multiplied across all of the DBAs in an organization, these management cost savings quickly grow into a figure that dwarfs the one-time licensing fee required to acquire a product of this nature. The time difference between the products is 44% in favor of Oracle Database because once again the IBM DB2 management interface was more complex. Edison Group believes that both vendors are paying significant attention to lowering the complexity and time spent by administrators for these tasks. Oracle Database exhibited a 31% time advantage in creating and indexing a partitioned table. Not only was there a greater time differences for these tasks, but also the tasks themselves were more complicated to perform under DB2 [12].

5. Database performance manageability

Oracle and DB2 differ greatly in terms of diagnostics and self-tuning capabilities. With Oracle Database 10g, users can benefit from many internal tools and features that simplify performance monitoring and automate the detection and resolution of performance problems. Oracle also provides many self-tuning capabilities that dynamically adjust the database parameters to take advantage of variations in the consumption of system resources. Finally, Oracle Database also offers a number of intelligent

advisories for performance tuning that allow administrators to simulate a variety of "what-if" scenarios: index advisory, summary advisory, memory advisory, MTTR advisory, table/index usage advisory. While DB2 offers some self-tuning capabilities and advisories, administrators are required to know a lot about the database. For example, to perform real time monitoring, DB2's Control Center provides administrators with a lot of metrics but without any precision about which ones are important indicators of the overall performance or health of the system. When confronted with a vague problem like "system is slow" the DB2 administrator has to know where to look and poke around to find the cause of the problem. Oracle on the other hand guides the administrator via advice, help and drill-downs through a process of analyzing the cause of the problem. The following table summarizes the unique features provided by Oracle that enhance the information that can be used to tune databases, and help automate the tuning process. The absence of such features in DB2 requires administrators to use empirical approaches and manual interventions to tune the performance of the database.

	Oracle	**DB2**
Manageability of performance	**AWR, ADDM, AST**	No equivalent or limited features

Table 3: manageability of performance and self-tuning

Automatic Workload Repository (AWR)

The Automatic Workload Repository (AWR) is a persistent repository, within the Oracle Database, which contains performance data and statistics about the operations of the database. At regular intervals, Oracle Database makes a snapshot of all its vital statistics and workload information and stores them in the AWR. The statistics collected and processed provide the data for the diagnostic facilities of Oracle Database that support both pro-active and reactive monitoring. DB2 does not provide an equivalent infrastructure.

Automatic Database Diagnostic Monitor (ADDM)

ADDM is a self-diagnosis engine in the database that proactively analyzes data captured in AWR to understand the state of the system. The goal of ADDM is to identify those parts of the system that are consuming the most 'DB time', and to reduce this time whenever possible, either by recommending solutions, or by referring to other 10g advisory components, such as the new SQL Access Advisor. ADDM drills down to identify the root cause of problems rather than focusing just on the symptoms and reports the overall impact of the problem [11].

Automatic SQL Tuning (AST)

Automatic SQL Tuning is based on the Automatic Tuning Optimizer. In automatic tuning mode, the Oracle Query Optimizer is given more time to perform the investigation and verification steps required for the tuning process. This additional time allows the optimizer to use techniques, such as dynamic sampling or partial execution, which could not be used under the time constraints of the regular operating mode. These techniques help the optimizer validate its own estimates of cost, selectivity and cardinality [11]. As a result, using the automatic tuning mode augments the probability of generating well-tuned plans for SQL statements. The functionality of the Automatic Tuning Optimizer is exposed via the new SQL Tuning Advisor. The Automatic Tuning Optimizer performs multiple analyses: statistics analysis, SQL profiling, access path analysis, and SQL structure analysis.

4. Conclusion

Oracle designed the first enterprise grid computing which provide the industry a leading performance, scalability, resource utilization, manageability and consistency with emerging trends in technology. At first glance, Oracle and IBM appear to offer similar security solutions, but with closer inspection, it is plain to see that the two companies approach security differently and ship solutions at vastly different levels of maturity. Independent evaluations and feature-for-feature comparisons prove that the Oracle Database is more effective than IBM's DB2 Universal Database. The Oracle database builds-in security and stands on its own; the database itself has achieved nine independent evaluations performed by industry experts. IBM has not completed any evaluations of DB2. It will take DB2 at least several more releases before it can approach the current self management capabilities of Oracle Database. With its self-managing capabilities, Oracle Database eliminates time-consuming, error-prone administrative tasks, so database administrators can focus on strategic business objectives instead of performance and cost. Oracle continues to improve its self-management capabilities, which contribute to lowering Oracle's TCO, while IBM struggles to close the functionality gap that exists between the two products.

5. References

[1]. Rashid, Z.; Basit, A.; Anwar, Z.; "TRDBAC: Temporal reflective database access control" 6th International Conference on Emerging Technologies (ICET), 2010

[2].Lavanya Pamulaparty, T. Praveen Kumar, P. Vijaya Babu Varma, "A Survey : Security perspectives of Oracle and IBM DB2 databases", International Journal of Scientific and Research Publications, Volume 3, Issue 1, January 2013 ISSN 2250-3153.

[3]. Zheng-Fei Wang; Ai-Guo Tang; "Implementation of encrypted data for outsourced database" Computational Intelligence and Natural Computing Proceedings (CINC), Volume 2, 2010

[4]. Yan Zhao; Yongcheng Luo; Jian Wang; Jiajin Le; "A novel privacy preserving approach for database security" ICTM Volume 1, Pages 408-411, 2009.
[5]. Anbalagan, P.; Vouk, M.; "Towards a Unifying approach in Understanding Security Problems", ISSRE, pages 136- 145, 2009.
[6]. Michel Abdalla, Emmanuel Bresson, Olivier Chevassut, Bodo Muller and David Point cheval, "Strong Password-Based Authentication in TLS using the Three-Party Group Diffie-Hellman Protocol", International Journal of Security and Networks, vol. 2, numbers 3/4, pp. 284-296, Inderscience, 2007.
[7].OracleTechinicalDocumentation – http://www.oracle.com
[8]. IBM-DB2 Technical Documentation - http://publib.boulder.ibm.com/infocenter/
[9]. Shehab, M.; Bertino, E.; Ghafoor, A.;" Watermarking Relational Databases Using Optimization-Based Techniques" IEEE transactions on Data Engineering, Volume 20, Issuer 1, 2006.
[10]. Bertino, E.; Sandhu, R.;" Database security- concepts, approaches, and challenges"IEEE transactions on Secure Computing, Volume 2, Issue 1, pages 2-19, 2005.
[11].D.Taylor; "SSL for TLS Authentication"IETF draft-ietf-tls-srp-01.txt (work in progress) June 29, 2001
[12]. Steve Mintz,Edison Groups – A Comparative cost study of oracle and IBM DB2 enterprises, Jan 2009.