

Solutions for Highly Scalable Database Applications

An analysis of architectures and technologies

Bryan Thomas

Executive Overview

Background

Oracle Database Server 10g and Microsoft SQL Server 2005 are the two leading Relational Database Management Systems (RDBMS) products on the market. Both are feature-rich data management platforms that have proven their ability to handle large-scale, mission-critical applications. However, both companies are offering different visions for how large scale, mission critical applications of the future will be developed, designed and deployed. Oracle is single-mindedly offering its “Real Application Clusters” (RAC), a “scale-out” technology as the silver-bullet for all types of database applications, essentially writing off the traditional scale-up approach. Oracle claims that RAC offers unprecedented scalability and availability at a low cost due to its ability to utilize low-cost commodity hardware. Microsoft on the other hand is taking a more nuanced approach offering both “scale-up” and “scale-out” technologies, asserting that no one technology fits all.

Objective

The goal of this paper is to provide guidance for anyone interested in choosing a database for deploying highly available enterprise applications. The paper explores the different options available, examining the pros and cons of each. It goes behind the marketing claims to examine how customers are really using the technologies today. It also examines trends in software, hardware, storage and processor technology so that the reader can make longer-term strategic decisions. Most importantly the paper also looks at the economic aspects of each solution -- taking the license and maintenance costs of each option into consideration -- so that readers can make a holistic decision.

Findings

After an in-depth analysis of Oracle RAC and SQL Serve 2005, the author’s concludes that.

- € Oracle RAC is an interesting technology with great potential. However, its high-cost and excessive administrative complexity offsets any potential hardware cost savings obtained by using commodity hardware.
- € SQL Server 2005 on SMP servers with Database Mirroring for high availability is a more cost-effective and easier to manage solution than Oracle 10g RAC. SQL Server 2005 can meet the scalability requirements of 99% of customers’ real-world applications, while providing the desired levels of availability.
- € For situations where scale-out architectures are the only choice, both Oracle 10g RAC and SQL Server 2005 should be considered as equally viable options.

Introduction

This paper will focus on analyzing the architectures and technologies available to information technology (IT) professionals interested in building highly scalable database applications. The paper is restricted to Oracle Database Server 10g and Microsoft SQL Server 2005.

About the Author

As the author, knowing a little about my background is important to understand potential bias and points of view. I have been a database professional for 15 years and have experience with all the major database systems including IMS, DB2, Tandem Non-stop SQL, Oracle, Sybase, and SQL Server. I currently work as a consultant, troubleshooting database installations for a wide range of small to large businesses in North America. I have worked as both a production Oracle DBA and a production SQL Server DBA and designed high volume OLTP databases and multi-terabyte data warehouses. I am an expert in Oracle RAC installations and troubleshooting.

Structure of the paper

Understanding a database system's scalability and availability requirements starts with the analysis of the business requirements. Fast software, robust hardware, and tuning are only a part of the issue. A technical design based on the business requirements is the foundation for your scalable solution. This paper first presents a common set of business requirements that most IT professionals deal with on a regular basis, to help you understand the basis for choosing a scalable database solution. Then we will discuss two vendor solutions -- SQL Server 2005 and Oracle Database Server 10g. We will first evaluate Oracle's RAC technology, since Oracle promotes RAC as its preferred technology for all situations. This will be followed by an evaluation of SQL Server 2005 – first looking at “scale-up” solutions and then “scale-out” solutions.

Business Requirements

There are a number of factors that need to be considered when choosing a data management platform, including: security, availability of skilled database administrators, availability of ISV applications, ease of application development, choice of hardware etc. For the purposes of this paper, we will focus on three of the more important business requirements for mission-critical, high end applications: **Scalability & Performance, High Availability and Cost Effectiveness**¹. It must be noted that the three business requirements are highly inter-related, so we take all three into account simultaneously when evaluating different architectures.

Let us first take an in-depth look at Oracle RAC.

Overview of Oracle RAC

Oracle Real Application Clusters (RAC) has been presented by Oracle as the one solution that addresses all scalability, performance, high availability requirements, at a lower cost than any other solution. Before we examine how RAC solves each of the business requirements, let's go over the details of how RAC works.

¹ The paper uses the commonly understood and conventional definition of these terms.

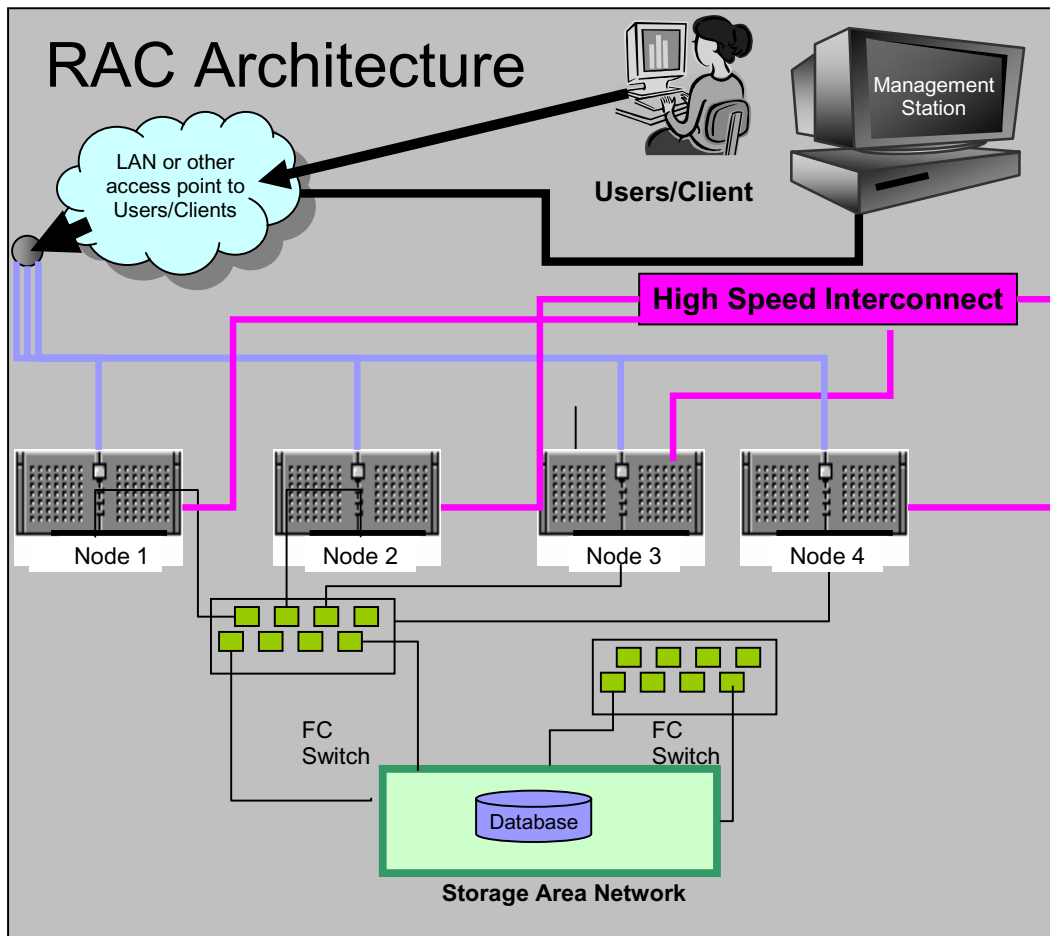


Figure 1: A typical 4-node RAC configuration

Figure 1 shows a typical 4-node RAC configuration. The Oracle RAC Database Servers run on all the nodes in the cluster. The data resides in the shared storage. All the nodes in the cluster have equal access to all the tables in the database. There is no notion of data being “owned” by any particular node. As a result data does not have to be partitioned, though very often it is partitioned to increase performance. Applications just connect to the RAC cluster, not to a specific node in the cluster; RAC distributes the load evenly across all the nodes of the cluster.

Oracle 10g RAC is made up of several different components including the Global Cache Services (GCS), Cluster Ready Services, Automatic Workload Management, Virtual Internet Protocol, the Oracle Cluster Registry and the Oracle 10g database. There is a logical “global lock manager” that coordinates among all the nodes in the cluster so that they don’t step over each other.

Oracle 10g RAC Benefits

Oracle claims that RAC running on a cluster provides the highest level of capability in terms of availability, scalability, and low-cost computing. Here is how RAC is supposed to provide these benefits:

Availability: If a node in the cluster fails for any reason, Oracle RAC continues running on the remaining nodes. All the applications (users) connected to the failed node are transparently

reconnected and distributed among the surviving nodes in the clusters. The failover is expected to be completed in 20 seconds or less.

Scalability: When more processing power is needed, new nodes can be easily added to the cluster, without having to modify the application or the database in any way. The load is redistributed so that it is balanced across all the nodes of the cluster. Oracle 10g R2 RAC supports up to 100 nodes in a cluster.

Cost Savings: RAC reduces hardware cost by running applications just as efficiently on clusters of small (< 4 CPUs), standardized, low-cost commodity hardware as on the more expensive SMP systems. For example a 16 node cluster of 4 CPUs each costs significantly less than an equivalent 64 CPU SMP machine. There are a number of reasons for the price differential, including the fact that smaller boxes benefit from the economies of scale. See the appendix for a comparison of cost-per-processor of low-end commodity boxes versus high-end SMP boxes.

Analysis of Oracle 10g RAC

There have been numerous heated *technical* debates about whether RAC really works as advertised. An in-depth, technical analysis of RAC is beyond the scope of this paper. The author believes that most such debates are inconclusive and rarely help a decision maker.

Of the three claims that Oracle makes about the benefits of RAC, the claim about cost savings is the weakest. But before we get to that, let us quickly analyze the other two claims, namely Availability and Scalability.

Availability: It is true that RAC offers a good solution for server failures. However, RAC by itself offers *no protection against disasters or storage failures*. RAC is based on shared-data architecture; therefore the storage is a single point of failure. If the storage fails for any reason, the whole cluster fails. Oracle offers DataGuard as the solution to this problem. DataGuard involves having one or more duplicate databases called “Standby” database. “Standby” databases are kept in sync with the “primary” using log-shipping technology. It must be noted that Oracle will charge full price for each “Standby”, so the total cost of an Oracle solution has to be multiplied by the number of standby databases.

Scalability: Oracle claims that RAC offers “transparent” scalability i.e. applications designed for a single servers can scale on RAC clusters without application or schema changes. In author’s opinion this is not strictly true, especially for high-end OLTP applications. Consider the fact that in their TPC-C benchmark², Oracle partitioned the RAC database for better performance. The number of partitions is equal to the number of nodes in the cluster. If you have to partition a database to get better performance, then that throws the claim of “transparent” scalability out of the window, because every time a node has to be added or removed from the cluster, the number of partitions has to be modified accordingly³.

While Oracle claims that it can scale to a 100 nodes, the author has seen little evidence of this. The largest cluster that Oracle has publicly talked about is Amazon, running a data warehouse on 16 node cluster. The vast majority of the RAC clusters are just 2-4 nodes. It is clear that most customers are using RAC for high availability rather than for scalability.

² Check out the full-disclosure report for this 16-node Oracle RAC TPC-C clustered benchmark:
http://www.tpc.org/tpcc/results/tpcc_result_detail.asp?id=103120803

³ Modifying partitions is a non-trivial, time-consuming operation. Furthermore, the database will be unavailable during this period.

Lower Costs: This is an area where Oracle’s claim is the weakest. While it is true that RAC can lower hardware costs by using a cluster of cheaper, smaller, commodity servers, any hardware cost savings is more than offset by extra cost of:

- € RAC software
- € Additional storage and networking costs and
- € Extra cost of administration.

RAC is a very expensive piece of software. The list price for RAC is US \$20k per processor. This is more than the difference between low-end commodity servers and high-end SMP systems. See Figure 2 below.

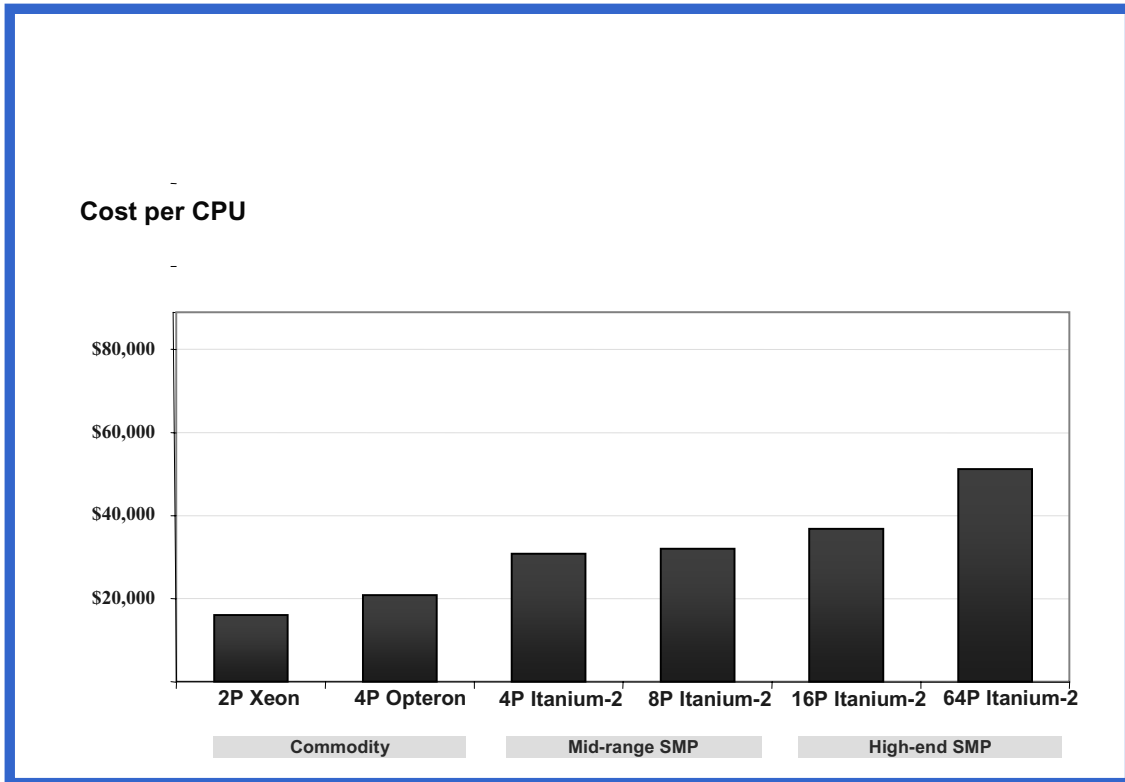


Figure2: Shows how the cost-per-processor increases from commodity to high-end servers

A typical 4 processor server costs between US \$20k and US \$30k per processor. At the high end the cost is around US \$50k per processor. So the average difference between small commodity servers and a high-end server is \$20k to \$30k per processor. However the cost of RAC itself is \$20k, which pretty much wipes out any hardware cost savings. In addition, the cost of the SAN switches must also be factored into the equation. As the number of nodes in the cluster increases, SAN switches with more number of ports are required. These high-end switches can be very expensive⁴. So when all these costs taken into account, RAC ends up being more expensive than a non-clustered Oracle solution and—as we will see later—a lot more than expensive than a SQL Server based solution.

⁴ It is interesting that RAC uses commodity servers, but requires non-commodity high-end storage switches. It is also important to note that contrary to Oracle’s claims RAC does not run on any off-the-shelf commodity hardware—RAC requires “certified” hardware that limits your choice and consequently increases the cost.

RAC is Very Complex to Manage

Putting aside the lack of economic justification for RAC, probably the single biggest reason to avoid RAC might be its complexity⁵. The complexity associated with RAC is documented succinctly in “Real-World Challenges for Oracle RAC Implementation”⁶. The author believes that the very fact that there are several books dedicated to RAC, each over 800 pages, should be a warning sign⁷.

Here is a brief overview of what makes Oracle RAC so complex:

RAC requires Application and Schema Design Changes

Contrary to Oracle’s claims, the authors experience has been that applications (and the associated database schemas) have to be specifically designed (or modified in the case of existing applications) in order to get them to perform on RAC. The nature and extent of the changes depends on a number of factors including the size of the clusters, the nature of the application (OLTP or DW), the speed of the cluster inter-connect and transaction volumes. As mentioned earlier, very often data has to be partitioned, especially tables that are hotspots.

As a result of the application and schema change required for RAC, only a small percentage of the ISV applications that are certified for the non-clustered version of Oracle are certified on Oracle RAC. For example at the time of writing, SAP, one of the earliest ISVs to support RAC, had not certified its applications on Oracle 10g R2 RAC.

RAC requires special storage solutions

Oracle RAC cannot run on a regular file system i.e. the file systems that usually ship with operating systems. Oracle RAC either works directly on **raw devices** or requires a clustered file system (such as the **Oracle Cluster File System (OCFS)**) or the **Oracle Automatic Storage Manager (ASM)**. Due to the inherent complexity of using RAW storage devices, Oracle recommends OCFS or ASM, with more emphasis on ASM lately. While ASM has many interesting capabilities, it is very complex to manage and administer—it is a full-blown instance of Oracle requiring significant DBA time and skills to manage. Furthermore ASM is a proprietary file system; most of the backup & restore, diagnostics, monitoring and performance tuning tools that you are currently using will not work with ASM.

Patching RAC is difficult

Patches for RAC come in two flavors—those that can be applied one node at a time and those that need to be applied to the entire cluster at once. In the former case only the node to which the patch is being applied has to be down; the rest of cluster is functioning. In the latter case the entire RAC cluster has to be shutdown, thereby making the entire database unavailable. An analysis of Oracle’s patches shows the majority of the patches belong to the latter category. In this case Oracle recommends a very complicated technique for “rolling upgrades” that involve two standby RAC clusters connected with Oracle Data Guard.

⁵ Complexity is relative. So RAC is complex compared to what? In this case I am comparing the complexity of RAC with non-clustered database, be it Oracle or SQL Server.

⁶ The paper is available here: <http://www.linxcel.co.uk/Whitepapers/Real-World%20Challenges%20for%20Oracle%20RAC%20Implementation-Issue%201.pdf>

⁷ Oracle 10g Grid and Real Application Clusters: Mike Ault and Madhu Tamma. 844 pages.

Tuning RAC is Complex

The complexity of RAC and the numerous moving parts involved in a RAC setup makes it hard to debug and tune. In addition to all the things that a DBA needs to know to tune an Oracle database, with RAC a DBA has to take numerous other factors into account including inter-connect traffic, inter-connect latency, pinging of data blocks between nodes, disk I/O for each of the nodes, table hotspots etc. Here is just a partial list of some of the workarounds that Oracle has suggested to get around the performance issues with RAC:

- € Assign transactions with similar data access characteristics to specific nodes, by partitioning users and applications.
- € Create data objects with parameters that enable more efficient access when globally shared.
- € Avoiding sequences as hotspots by creating node-specific staggered sequence ranges.
- € Reduce the number of rows-per-block (RPB) in order to reduce page contention.
- € Use as few indexes as possible to reduce intra-node pinging of index blocks.
- € Pre-allocate space by turning on dynamic space management.
- € Use reverse-key indexes to reduce index-page hotspots. This has the undesirable side-effect of eliminating the ability to use index-scans.
- € Design indexes such that the clustering factor is as close to the number of used blocks as is possible.
- € ...

The list is based on content from the book “Oracle 10g and Real Application Clusters”. These are very complex tuning recommendations, requiring a deep understanding of the inner-working of the Oracle database management system. The book summarizes the situation quite aptly when it says, **“This may seem perplexing, since some of the suggestions are contradictory”!**

Summary: RAC is an extremely complex piece of technology. Unless the complexity is reduced by an order of magnitude, RAC falls far short of its promise as a viable technology for the vast majority of database applications.

Overview of SQL Server 2005

SQL Server 2005 is the latest version of SQL Server and a major upgrade from the previous release. SQL Server 2005 has numerous enhancements in the areas of **Business Intelligence**, **Developer Productivity** and **Enterprise Capabilities**. An in-depth analysis of all the new capabilities is beyond the scope of this paper⁸. The focus of this paper is on those SQL Server features that are required to develop and deploy large-scale, highly available mission-critical applications. One particular feature of interest is Database Mirroring.

SQL Server 2005 Database Mirroring

Database mirroring is a new SQL Server 2005 technology for increasing database availability. Database mirroring ships transaction log records directly from the primary server to a standby server, ensuring that at all times the standby is a mirror image of the primary database. Database Mirroring also quickly fails over to the standby server in the event of the primary server going down for any reason. You can code client applications to automatically redirect their connections so that in the event of a failover, they automatically connect to the standby database. While the precise time to failover depends on a number of factors, it is possible to failover in around 10 seconds. SQL Server 2005 Database Mirroring does not require proprietary hardware and is easy to set up and manage. More details about Database Mirroring can be found at:
<http://www.microsoft.com/technet/prodtechnol/sql/themes/high-availability.mspix>

⁸ More details about SQL Server 2005 can be found at: <http://www.microsoft.com/sql>

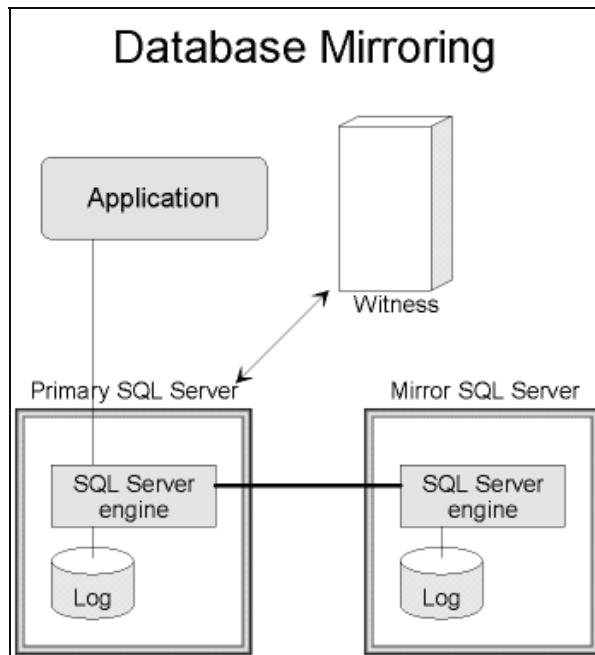


Figure 3: A typical SQL Server 2005 Database Mirroring setup



Comparison of SQL Server 2005 and Oracle RAC

In this section we will compare “SQL Server 2005 with Database Mirroring” as an alternative to “Oracle 10g RAC” and see how it stacks up. Specifically we will evaluate whether SQL Server 2005 with Database Mirroring can match Oracle RAC in the areas of **Scalability, Availability** and **Cost**.

Oracle 10g RAC and SQL Server 2005 – Scalability comparison

It would appear that Oracle RAC has the edge here. Oracle 10g RAC R2 can support up to 100 nodes. Oracle does not specify whether there are any limitations on the number of CPUs per node. Therefore, in theory, Oracle RAC can scale beyond 64 CPUs. However, it must be noted that Oracle has not demonstrated publicly that Oracle RAC can scale beyond 64 CPUs. The largest TPC-C benchmark with Oracle RAC has 64 CPUs—a 16 by 4 cluster. The largest RAC installation known to the author is a 64-CPU data warehouse at Amazon, also a 16 by 4 cluster.

How does SQL Server 2005 compare? SQL Server 2005 has proven that it can scale to 64 CPUs on a single SMP server. Interestingly, **SQL Server 2005 has a better performance and price-performance than Oracle 10g RAC for 64 CPUs** as can be seen from the table below.

Company	System	tpmC	Price/tpmC	System Availability	Database	Date Submitted	Cluster
	hp Integrity Superdome	1,231,433	4.82 US \$	06/05/06	Microsoft SQL Server 2005 Enterprise	11/28/05	N
	HP Integrity rx5670 Cluster 64P	1,184,893	5.52 US \$	04/30/04	Oracle Database 10g Enterprise Edition	12/08/03	Y

These two TPC-C benchmarks clearly demonstrate that that on a system with 64 CPUs, SQL Server 2005 outperforms Oracle 10g RAC *and* is more cost effective.

Scalability beyond a single SMP Server

But what if someone wanted to scale beyond a single SMP server? While this is an interesting topic for academic discussions, the question is less interesting for practical purposes. Why? Because, today the largest SMP server has 64 processors and can run over 99% of the world's real-world applications! In fact the author is not aware of any application, OLTP or Data Warehousing, which cannot be run on a single 64-CPU SMP server. According to Winter Corp's Annual Survey⁹ of the largest databases in production today, the largest OLTP and Data Warehousing database run on SMP servers, not clusters. See table below.

	Workload	Company	Server Type	Non-clustered SMP Servers in Top Ten
Largest Data Warehouse	By Size (Unix)	Yahoo	Non-clustered SMP	6 of 10
	By Size (Windows)	UPSS	Non-clustered SMP	10 of 10
	By # of Rows (Unix)	AT&T	Federated MPP	9 of 10
	By # of Rows (Windows)	ComScore Networks	Non-clustered SMP	10 of 10
Largest OLTP Database	By Size (Unix)	US Patent Office	Non-clustered SMP	8 of 10
	By Size (Windows)	AIM Healthcare	Non-clustered SMP	10 of 10
	By # of Rows (Unix)	Anonymous	Non-clustered SMP	9 of 10
	By # of Rows (Windows)	Verizon	Non-clustered SMP	10 of 10

We can expect this situation to continue in the future as processors speed will continue to outpace increase in workloads. Advancements in processor technology such a multi-core processing will only serve to further solidify this situation.

Summary: SQL Server 2005 on single SMP server can easily scale to run even the most demanding real-world applications.

Oracle 10g RAC and SQL Server 2005 – Availability comparison

Protection from Server Failure

Both Oracle 10g RAC and SQL Server 2005 Database Mirroring provide protection from server failures. In the event of the failure of the database software, operating system or the hardware, both solutions can failover the applications transparently, within 10-20 seconds, in order to minimize the disruption to the end user. From this perspective both solutions offer equivalent capabilities.

Protection from Storage Failure

Oracle 10g RAC does not provide any protection against storage failures. If the disk subsystem fails or becomes unavailable for any reason, all the data is lost and the entire cluster goes down.

⁹ http://www.wintercorp.com/VLDB/2005_TopTen_Survey/TopTenWinners_2005.asp

SQL Server 2005 with Database Mirroring offers protection from *both* server and storage failures. Oracle 10g RAC requires Oracle Data Guard to protect from storage failure. Oracle Data Guard like SQL Server Database Mirroring is based on shipping log files from the principal to the standby/secondary database server.

But there is one important difference between Oracle Data Guard and SQL Server Database Mirroring: Microsoft does not charge for SQL Server software running on the mirror, whereas Oracle charges full price for the Oracle instance mirror/standby.

Summary: While technically both “Oracle RAC with Data Guard” and “SQL Server 2005 with Database Mirroring” have similar technical capabilities, SQL Server 2005 is a more cost effective solution.

Oracle 10g RAC and SQL Server 2005 – Cost comparison

Oracle claims that RAC provides substantial hardware cost savings by enabling customers to use cheap commodity servers. The claim is only partially true. Oracle 10g RAC is *so expensive* that any hardware cost savings is more than offset by the extra cost of the Oracle software. This is especially true when costs are compared with SQL Server 2005. The cost comparison is shown in the figure 4.

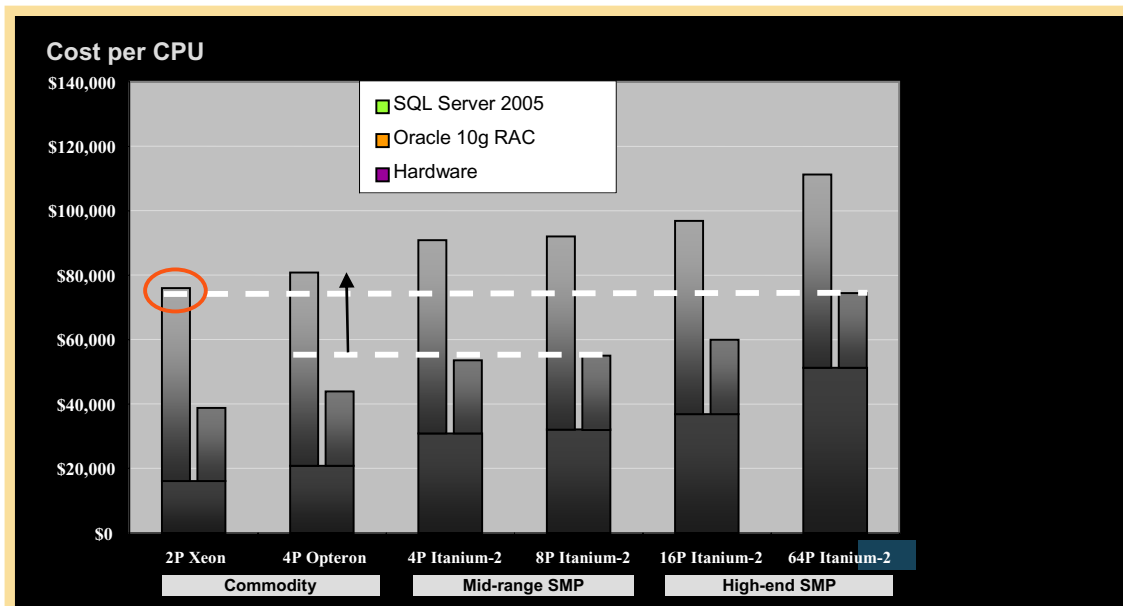


Figure 4: Cost comparison, Oracle RAC vs. SQL Server 2005 on different hardware

As can be seen from the figure¹⁰:

For the cheapest commodity servers, the cost per processor = US \$18k.
 For the most expensive high-end servers, the cost per processor = US \$50k.
 Oracle 10g Enterprise Edition with the RAC option = \$40k + \$20k = US \$60k
 SQL Server 2005 Enterprise Edition = US \$25k

¹⁰ We are using the publicly available list prices for both Oracle 10g and SQL Server 2005. It is quite likely that nobody pays full price for either product. However that does not change the relative pricing differential between Oracle and SQL Server 2005.

SQL Server 2005 EE on the most expensive server = \$50k + \$25k = \$75k per processor.
Oracle 10g EE + RAC on the cheapest server = \$60k + \$18k = \$78k per processor.

The simple fact is that: *Oracle 10g RAC on the cheapest commodity servers is still more expensive than the SQL Server 2005 on the most expensive, high-end, SMP Servers¹¹.*

Oracle 10g RAC and SQL Server 2005 – Comparison Summary

As shown in the section above, SQL Server 2005 matches and exceeds all three value propositions of Oracle 10g RAC, namely scalability, availability and cost effectiveness.

- ≠ SQL Server 2005 on a single SMP server can scale-up to meet the most demanding real-world applications. While theoretically Oracle 10g RAC can scale beyond the limitations of single SMP, it is important to note that:
 - Oracle has not offered any public evidence to backup this claim.
 - There are no workloads that cannot be handled by a SMP server.
- ≠ “SQL Server with Database Mirroring” can match the high availability capabilities of Oracle 10g RAC—at a substantially lower cost.
- ≠ SQL Server 2005 on single SMP server is always cheaper than Oracle 10g RAC on an equivalent cluster of commodity servers. The hardware cost savings go to Oracle, not to their customers.

Furthermore, as discussed earlier, Oracle 10g RAC is very complex to design, tune, debug and administer. SQL Server 2005 on the other hand has a well established reputation as one of the easiest databases to manage.

“Scaling-Out” with SQL Server 2005

The focus of the discussion so far has been on comparing the “scale-out” strategy with Oracle 10g RAC with the “scale-up” strategy with SQL Server 2005. I conclude that scaling-up with SQL Server 2005 is the preferred solution—when the primary criteria are scalability, high availability and cost. However you might still prefer scale-out architectures for the following reasons.

1. Some companies have standardized on commodity servers, refusing to deploy medium or high-end SMP servers. Typically these companies have standardized on vendors such as Dell who do not offers servers with more than 4 processors.

¹¹ This calculation does not be includes the cost of other Oracle “options” that are typically required in a RAC environment such as partitioning which would add another \$10k per processor.

2. Some customers are uncomfortable with the 64 processor limit of SMP servers, even if the servers meet their current requirements. They want to know that their database can scale beyond 64 CPUs if the need arises in some distant future.
3. Some customers have a highly distributed architecture that does not lend itself very well to large centralized SMP servers.

So, is RAC the only option for customers who prefer to scale-out rather than scale-up? The short answer is no. SQL Server 2005 offers a choice of scale-out technologies. Unlike Oracle, which is single-mindedly pushing RAC as the only viable scale-out architecture, Microsoft has taken a more nuanced approach, offering multiple scale-out technologies. Each of these technologies has its own advantages and disadvantages. You should pick the technology that best meets your business requirements.

In the following section we will discuss four scale-out strategies with SQL Server 2005.

- € Service Oriented Database Architecture (SODA)
- € Shared Scalable Database (SSD)
- € Peer-to-Peer (P2P) Replication
- € Data Dependent Routing

Service Oriented Database Architecture (SODA)

The last three to five years have seen the emergence of large-scale, loosely-coupled, distributed system architectures, particularly as Internet e-commerce sites have grown into major commercial operations. Service Oriented Architecture (SOA) has emerged as the dominant loosely-coupled, service-centric architecture. Applications based on SOA are more resistant to failure and are more easily scaled up by adding resources using a variety of methods as necessary to meet changing demands, and allow integration of legacy systems into B2B and other systems.

SOA service providers, consumers, and other components handle data as a natural feature of their roles in an SOA application. An SOA application typically still uses central databases to store and protect data, but is likely to have many such large databases that hold classes of data, such as separate storage of sales, manufacturing, and operations data, and specialized subsets of each. Each service provider and consumer may have a localized need for cached data or its own specialized data store. The messages that travel between the distant parts of the application are themselves often data worth archiving for various uses.

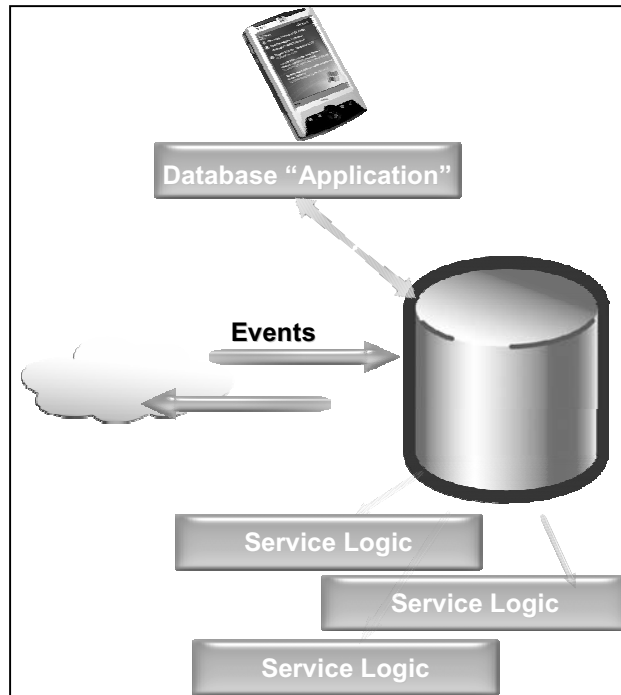


Figure 4: Overview of a sample SODA-based application

In a SODA, data can be partitioned based on its characteristics in the system in four general ways:

- € **Reference data** is used to create service requests, such as a product catalog. It must be in a format usable by all parties, and is identified in a way that doesn't change over time, such as a catalog date.
- € **Activity data** is ephemeral data used to perform a specific activity, such as a pick list used to retrieve purchased items from inventory. Since it is private to the service, the format doesn't need to be understood by other parties.
- € **Resource data** is long-lived data that is used internally by a service, such as SKUs, customer data, and account data.
- € **Service Interaction data** is used to communicate between services. It must be in a format that is understood by all parties, and must remain constant over time. For example, an order form is communicated between services. If the order is lost, it must be able to be regenerated in the same form as the original and transmitted again.

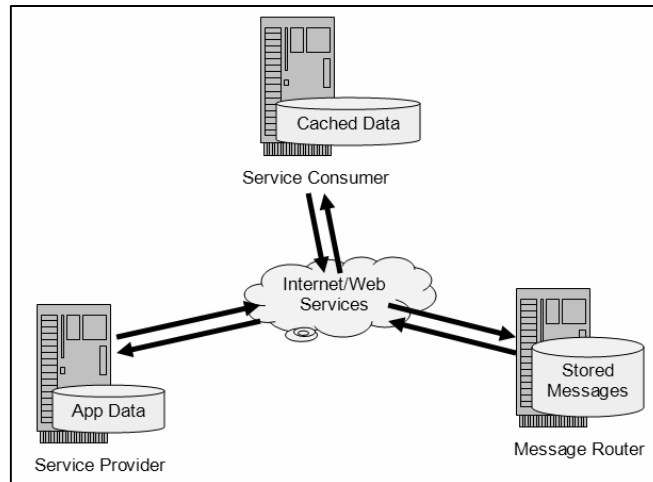


Figure 5: Small portion of a Service Oriented Architected application

SQL Server 2005 SODA features

The SQL Server 2005 includes a number of features to address the needs of Service Oriented Database Architectures. These include:

- ⊄ **Native Web Service Access:** SQL Server is directly integrated with the Windows 2003 database server to enable any SOAP compliant client to directly invoke stored procedures.
- ⊄ **Service Broker,** a new class of transactional middleware that is service- rather than message-centric to support scalable services.
- ⊄ **CacheSync** that allows data-dependent caches to receive a notification that its data requires refreshing because the underlying database has changed based on complex queries.
- ⊄ **SQLCLR** that deeply integrates sophisticated logic processing into the database to reduce latencies due to remote data access

Summary: The Service-Oriented Database Architecture provides for the construction of highly-scalable, highly-available, databases. By moving the transparency boundary to the service level, SODA avoids the scaling limitations of SQL statement-level scale-out solutions. With its highly reliable inter-service communications mechanism, SODA supports the design of databases that can achieve near-continuous availability.

Peer-to-Peer Transactional Replication

Another method for scale-out with SQL Server 2005 is using peer-to-peer transactional replication (P2P). (This replication method is similar to bi-directional transactional replication in SQL Server 2000, but has been greatly enhanced with SQL Server 2005 and called P2P.) P2P was designed to allow applications to read or modify data on any of the nodes participating in the replication. Load-balancing of reads across nodes can be achieved, as well as partitioned updates across nodes. This type of replication is suitable for environments that require high availability and read scalability, such as for OLTP and reporting.

The basic concept of replication is that data is published from a source server, called the Publisher, and replicated to a destination server, called the Subscriber. With standard transactional replication, the data replicated to a Subscriber is read-only and cannot be modified. With P2P replication, each node involved in the replication acts as both a Publishers and a

Subscriber to the other nodes, and each node can modify data by allowing '2-way' transactional replication. Each server node maintains its own copy of the database, having the identical schema and data on all nodes.

With a custom application, data modifications (inserts, updates, deletes) can be partitioned across nodes such that only certain rows of data are updated per node, so that no two nodes would update the same row at the same time. For example, updates to customer data with a last name that begins with 'A-M' would be directed by the application to one node, and updates to 'N-Z' to another node. (If updates must be allowed on all data for all nodes, then consider using merge replication.)

In Figure 7 below, the system on the left shows P2P replication between two nodes, with updates going to both nodes and getting replicated to the other node. Reads are balanced across the two nodes by application server assignment. The right side shows a two node system with updates going only to one node. This one could also be accomplished using standard transactional (one-way) replication.

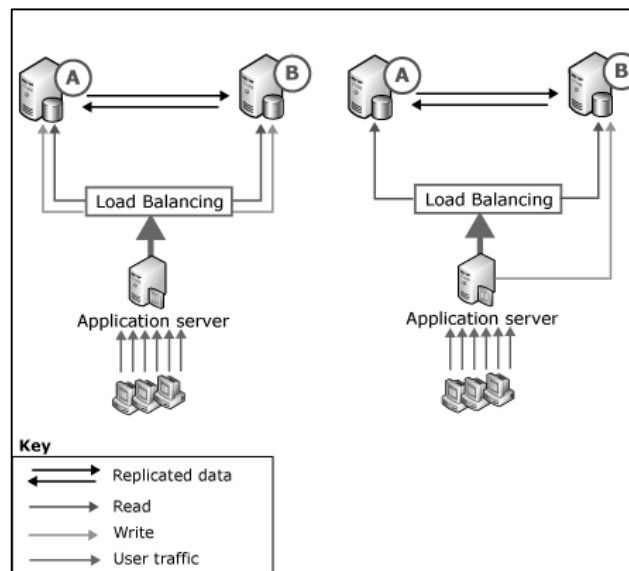


Figure 6: A typical P2P replication configuration

For high availability with P2P replication, a custom application can also be coded such that if one node is down, the database request can be redirected to a remaining node involved in the replication, which holds an identical copy of the database.

As with transactional replication, the P2P replication time delay depends on various factors such as the speed of the network, replication configuration settings, disk configuration, etc. But generally, P2P replication can provide replicated data within a matter of seconds, not minutes.

Scalable Shared Database (SSD)

Another scale-out solution for high read environments such as reporting and data warehouse/data mart scenarios is called Scalable Shared Database (SSD). SSD is supported by Windows Storage, which requires Windows 2003 SP1 (which introduced Read Only Volumes), and SQL Server 2005 Enterprise Edition. The premise is to allow concurrent access to a single read-only database by multiple SQL Server instances on multiple servers, referred to as reporting

servers. SSD guarantees an identical view of data to all reporting servers. The shared database is referred to as the report database.

This solution is applicable for read-only transactions that allow time-delayed, but consistent data. This could be thought of as “read-only RAC” because multiple servers are allowed access to the same database (on the same storage LUN) at the same time. Microsoft fully supports up to 8 server instances accessing the read-only database, as that is the number fully tested, although there is no hard limit to the number of instances.

There are two methods of implementing SSD. Both methods require the use of SAN storage and SQL Server 2005 data-copy techniques - such as Integration Services, backup and restore, file copy, and SMO Transfer (SMO copies the database while it remains online) – which are used to initially build and periodically refresh the report database.

The first SSD method involves creating a single copy of the database, called the report database, and allowing multiple report servers to concurrently access that database in read-only mode. This is a good solution for reporting systems and data mart scenarios in which the data gets periodically updated. With this method of SSD, the report database will get updated periodically - data is periodically refreshed to bring it up to date with the source database. While the report database is being refreshed, it cannot be accessed by the report servers. See Figure 6.

The second method for SSD is similar to the first, but minimizes the time that data is unavailable for reporting by creating two copies of the report database, such that one copy is available for read-only access while the other copy is being refreshed.

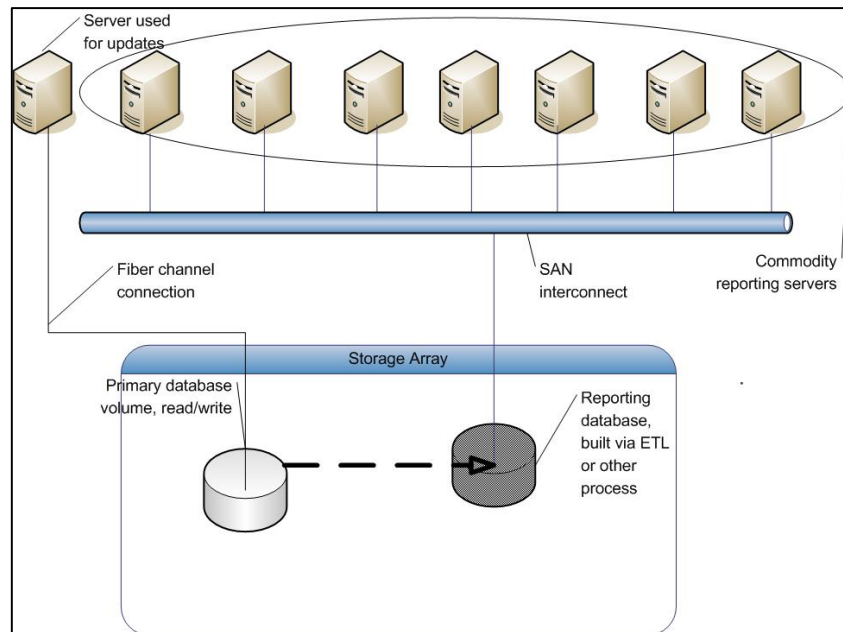


Figure 7: A typical SSD configuration

For more details on how to implement SSD with SQL Server 2005, see the Microsoft KB article #910378.

Data Dependent Routing (DDR)

Data Dependent routing is a scale-out architecture that involves the following:

- Data is partitioned into two or more individual, autonomous, federated databases. Each database owns its data. Each of the databases may or may not be aware of the existence of the other databases. There may be no distributed views that span databases. Typically the data is partitioned based on the value of a column such as customer id.
- Intelligence is provided in the application itself so that it can “route” the database transactions to the appropriate database. How the application makes the routing decision is up to the application itself. A partition lookup table that maps what data resides in which database, is a simple solution that is often used.
- If a single transaction involves multiple databases, the application will have to break up to transaction into discrete sub-transaction that are specific to a single database and route each of the sub-transactions separately. The application is also responsible for rolling up the results of the sub-transactions.
- An external transaction coordinator such as Microsoft DTC is required if multiple databases have to be updated in a single transaction.

The figure below shows an example of how Microsoft has implemented a scale-out solution using data-dependent routing techniques of SQL Server to implement a large-scale communication service platform for Microsoft’s MSN and Hotmail services.

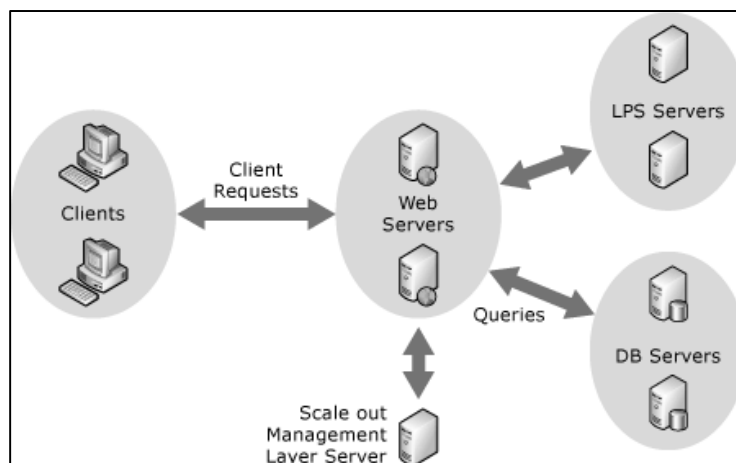


Figure 8: An example of data-dependant routing architecture

The MSN communication services platform consists of four tiers:

- € Web servers running Microsoft Internet Information Services (IIS)
- € Lookup partition database servers (LPS) running SQL Server 2000
- € Database back-end servers running SQL Server 2000
- € MSN scale-out management layer.

Records are ordered and partitioned by Passport User ID (PUID) across the back-end database servers. The scale-out management layer stores the mapping of data partitions to the physical back-end database servers in its own SQL Server database, which is independent from LPS and back-end databases. The LPS database stores the mapping of PUID to data partitions, and is partitioned across multiple LPS servers to accommodate growth. Consumers of the

communication services post requests to the Web server, which queries the LPS repository with the PUID to obtain the data partition where the records are located. Then the Web server queries the scale-out management layer to determine which back-end database server contains the information for that user. Information is returned to the client in a matter of seconds.

Scaling-out with SQL Server 2005: Summary

The key thing to take away from this discussion of SQL Server scale-out is that there are different kinds of data in any application, and an effective scale-out solution may include different approaches for different data. Reference data may be replicated and cached in many different places; historical data may be exposed through distributed queries on low-cost, high-capacity storage; activity data may be partitioned across a number of servers; and resource data may be split by application.

The decision to use a scale-out solution is influenced by several factors. Table 1 summarizes the importance of these factors for each solution.

	Update Frequency	Ability to Change Application	Data Partitionability	Data Coupling
Scalable Shared Databases	Read Only.	Little or no change required.	No requirement.	No requirement.
Peer-to-Peer Replication	Read mostly, no conflicts.	Little or no change required.	No requirement.	No requirement.
Linked Servers	Minimize cross-database updates.	Minor changes.	Not generally required.	Very important to have low coupling.
Distributed Partitioned Views	Frequent updates OK.	Some changes may be required.	Very important.	Little impact.
Data-Dependent Routing	Frequent updates OK.	Significant changes possible.	Very important.	Low coupling may help some applications.
Service-Oriented Data Architecture	Frequent updates OK.	Extensive changes required.	Not generally required, unless combined with DDR.	Low coupling between services required.

Figure 9: Factors influencing the selection of scale-out solutions.

Remember that some scale-out architectures may incorporate multiple solutions. DDR and SOA can be combined effectively, and replication and linked servers are generally part of any scale-out architecture.

With a good understanding of the data, requirements, and constraints for an application, an effective SQL Server solution can be designed to meet almost any level of scale-out.

Conclusion

Both SQL Server 2005 and Oracle 10g RAC are a highly scalable and reliable database platforms that can run the most demanding, mission-critical enterprise applications. However, when all things are taken into account, you will find that SQL Server 2005 is substantially easier to manage and is more cost effective than Oracle 10g RAC. Furthermore, while Oracle emphasizes a one size fits all “shared-everything”, “clustering-based” scale-out architecture SQL Server 2005 offers a greater choice of scale-out technologies.