Enterprise Database Options

A comparison of IBM DB2 for z/OS and Microsoft SQL Server

Julian Stuhler
Triton Consulting
July 2012

Contents

Abstract

This White Paper will present a high-level comparison of IBM DB2 for z/OS and Microsoft SQL Server in a large enterprise OLTP environment, looking at key requirement areas for enterprise data servers.

About The Author

Julian Stuhler is a Principal Consultant with Triton Consulting, a UK-based company specialising in the provision of DB2 consultancy, education, software and managed services to clients throughout Europe. He has over 24 years relational database experience, working for a number of clients within the insurance, telecommunications, banking, financial services and manufacturing sectors.

Julian has lectured widely on DB2 subjects, both in the UK and Europe, and won the "Best Overall Speaker" award at the 2000 International DB2 User Group European Conference. He is a co-author of an IBM Redbook on Java Stored Procedures, and a frequent contributor to industry publications such as IDUG Solutions Journal and Database Journal.

Julian is an IBM DB2 Gold Consultant, Past-President of the Board of Directors for the International DB2 User Group and an IBM Information Champion. He can be contacted at **julian.stuhler@triton.co.uk**.

Executive Summary

Some form of data storage sits behind nearly every application, but the unique demands of today's high volume, mission critical enterprise applications place some very specific demands on the back-end database system.

This paper compares the relative merits of IBM DB2 10 for z/OS and Microsoft SQL Server 2012, evaluating their capabilities against some of today's most important enterprise computing requirements.

High Availability and Scalability

IBM's System z platform is generally accepted as being the most highly available solution available today. The new AlwaysOn features in SQL Server 2012 have closed the gap somewhat, but the inherent resilience of the System z platform and the additional protection offered by DB2 data sharing combine to offer an unmatched level of availability with minimal performance impact.

SQL Server allows a reasonable degree of scalability for handling large data and/or transaction volumes within a single instance, even exceeding a single DB2 for z/OS subsystem in a few specific areas. However its real-world scalability is severely limited by the underlying capabilities of the x86 platform when compared to a System z server, which is likely to be able to accommodate a significantly greater workload.

This gap is exacerbated massively when clustering or other multi-server solutions are considered, as SQL Server currently has very limited options for scaling beyond a single server for typical read/write workloads, and all of these involve significant compromises and restrictions within the application and database design. In contrast DB2 data sharing allows up to 32 System z servers to service a single logical database image with no application change required.

Performance

This paper addresses just a few of the hundreds of factors that combine to determine the overall performance capabilities of a given RDBMS environment. However, even with this limited analysis it is clear that DB2 provides a more mature, flexible and capable environment for delivering high performance applications than SQL Server.

DB2's advantages include the power and scalability of the underlying System z hardware platform, z/OS' superior workload management capabilities, a more mature and capable SQL optimiser, static SQL support and superior facilities for caching data in buffer pools.

Data Governance

The System z platform is generally regarded as the IT world's most secure server. Security violations are rare, and malware incidents are virtually unknown. The combination of DB2 and z/OS can boast both a higher overall Common Certification security rating and a

dramatically lower level of reported vulnerabilities than SQL Server and Windows Server.

SQL Server has added some useful features in the last few releases that have improved its manageability and audit capabilities. However, it is still some way behind DB2’s capabilities with regard to roles, network trusted contexts, fine-grained access control and data masking.

Remote Access and Federation

The remote access capabilities of SQL Server and DB2 are broadly similar, but highly dependent upon the additional tooling and available driver support. With the right environment in place, either product is capable of supporting typical enterprise database federation requirements.

Both RDBMS products support a wide set of programming languages for in-house development, but DB2 support for COBOL and Java is both more mature and more extensive than that offered by SQL Server.

Application Support

For packaged applications the most suitable database platform is highly dependent upon the chosen package vendor, and the degree of resilience/scalability required.

For a mission-critical SAP solution, DB2 is a natural choice due to the close IBM/SAP partnership and superior resilience offered by the System z platform. With the exception of PeopleSoft and Siebel, Oracle solutions cannot be hosted by DB2 for z/OS and while SQL Server is supported for the older products, Oracle’s own RDBMS is the obvious strategic choice. Similarly, any enterprise adopting a Microsoft CRM or ERP solution has no choice but to adopt SQL Server as the back-end database.

Conclusion

Microsoft SQL Server has evolved significantly from its roots as a departmental shared database server and its most recent 2012 release contains some significant enhancements, especially with regard to product’s high availability characteristics. However, DB2 for z/OS has had considerably longer to mature, and was designed from the ground up as a scalable, efficient, highly available database platform for enterprise applications. It is also able to fully exploit the underlying strengths of the System z hardware platform, providing synergies that cannot be matched by Microsoft with its reliance on third-party hardware providers.

For these reasons, DB2 for z/OS remains the most attractive database platform for today’s mission critical enterprise applications. Organisations that are considering porting such applications from DB2 for z/OS to SQL Server should carefully evaluate the compromises inherent in such a move.

Introduction

Some form of data storage sits behind nearly every application, but the unique demands of today's high volume, mission critical enterprise applications place some very specific demands on the back-end database system.

Performance is of course critical, but so is the ability to rapidly scale in the event of a sudden increase in data or transaction volumes. The system must be resilient, allowing processing to continue in the event of hardware or software failure. Access to sensitive data must be limited to authorised personnel only, with robust security and audit capabilities that are able to meet strict legislative requirements. Typical enterprise landscapes rarely consist of a single database system, so support for "data federation" in a heterogeneous environment is important, as is proper support for high volume distributed applications in a two or three layer client/server architecture. Many large organisations have chosen to adopt packaged applications from vendors such as SAP and PeopleSoft, so good support for these applications is often critical.

Traditionally, large System z users have turned to IBM's own database systems to address these requirements: first IMS and then later DB2 for z/OS. However, ongoing pressures to reduce IT infrastructure costs have encouraged some organisations to reconsider their strategy and look at alternative off-mainframe database solutions such as Microsoft SQL Server.

In order to better promote its products to enterprise customers Microsoft has formed the Platform Modernization Alliance (PMA), a group that aims to "work together to help customers migrate and modernize their non-Microsoft business critical and mission critical workloads to the Microsoft Application Platform". For its part, IBM is also investing to ensure that System z remains an attractive target for new applications, with modern developer productivity tools and support for the latest application architectures in addition to the platform's traditional strengths in performance, scalability and resilience.

This paper compares the relative merits of IBM DB2 10 for z/OS and Microsoft SQL Server 2012, evaluating their capabilities against the typical enterprise computing requirements briefly outlined above.

Unless otherwise noted, all references to SQL Server and DB2 within this document refer to SQL Server 2012 and DB2 10 for z/OS.

Today's Enterprise Data Server Requirements

Most modern relational database management systems share a common set of core capabilities, with relatively minor differences in implementation. However, to successfully operate in an enterprise environment additional features and capabilities are required. In this section we will examine the most important of these, and provide some justification as to why they are so critical for typical enterprise customers. The remainder of this paper will then evaluate the capabilities of DB2 for z/OS and SQL Server within each of these categories.

Most modern relational database management systems share a common set of core capabilities, with relatively minor differences in implementation. However, to successfully operate in an enterprise environment additional features and capabilities are required.

Performance

IT application performance has been an important factor since the advent of the very first System z systems over half a century ago. Poor application performance can impact brand reputation, decrease customer satisfaction, drive down revenue and reduce employee productivity.

Poor application performance can impact brand reputation, decrease customer satisfaction, drive down revenue and reduce employee productivity.

For online transactions, poor application response time causes customer service representatives in call centres to react more slowly to incoming requests, potentially harming client satisfaction and decreasing the number of calls that can be handled per hour. This issue has become even more important with the advent of internet-connected applications such as online banking, where the customer has a direct perception of the responsiveness of the system that not even a skilled customer services representative can mask. Research by the Aberdeen Group found that a 1-second increase in transaction elapsed time resulted in a 16% drop in customer satisfaction and a 7% decrease in new business conversion.

Research by the Aberdeen Group found that a 1-second increase in transaction elapsed time resulted in a 16% drop in customer satisfaction and a 7% decrease in new business conversion.

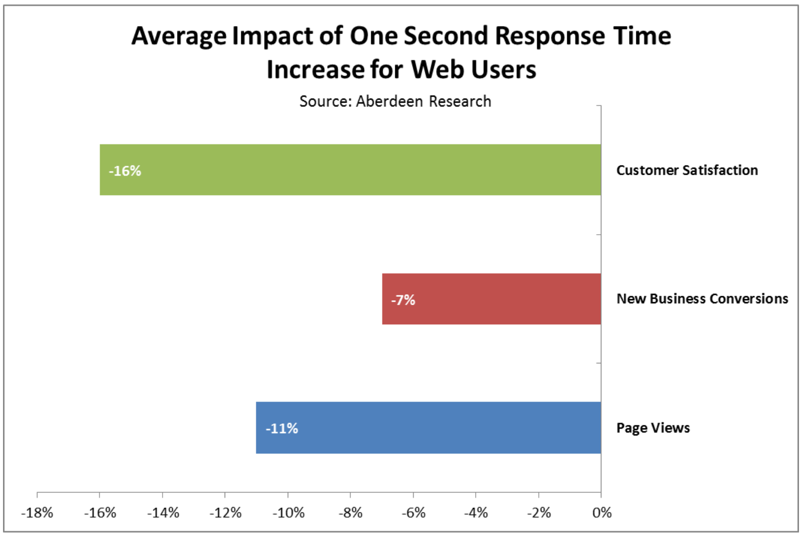


Figure 1 – Business Impact of Increased Web Response Times

Performance can be just as important for batch processes which may be unable to co-exist with online transactions, thereby making the application unavailable to the customer while they are running. In other situations, the output of a batch process has to be produced before a strict deadline in order to satisfy the requirements of an external business partner or to meet regulatory obligations. For example, the US Securities and Exchange Commission (SEC) requires some financial institutions to submit daily transaction reports to strict deadlines, and missing this by even a minute can result in fines of $100,000 or more.

The US Securities and Exchange Commission (SEC) requires some financial institutions to submit daily transaction reports to strict deadlines, and missing this by even a minute can result in fines of $100,000 or more.

Typically database access will be a very significant proportion of the overall elapsed time for a given batch or online process, so it is vital that the DBMS has good performance characteristics. For relational databases such as DB2 and SQL Server, the effectiveness of the optimiser[1] is critical to performance, as are other aspects such as support for static SQL execution, I/O avoidance and the ability to pre-fetch data before it is required by the application. Prioritisation and workload balancing can also be critical in order to ensure that the most important and/or time-critical requests are given the necessary resources to complete within their service level objectives.

Finally, it should be noted that good performance is typically related to lower CPU consumption, which can have a significant impact on the overall cost of running a given system. This aspect is covered in the section on Total Cost of Ownership below.

Scalability and High Availability

Organisations of all sizes have increasingly come to depend on their IT systems, but in today's typical enterprise the stakes can be very high indeed. What began many years ago as a business process automation exercise to improve productivity and reduce costs has evolved into an operational necessity and a key competitive differentiator.

A 2011 research report written by Ponemon Institute (1) makes for sobering reading. The study was based on data gathered from 41 larger data centres (in excess of 2,500 square feet) within the United States, covering all of the major industry sectors. The report concluded that a data centre outage costs an organisation an average of $505,502 per event, with more than half of that total being attributable to lost revenue and "business disruption" (which includes reputation damage and customer churn). The same study found a direct relationship between the duration of the outage and the eventual cost, with a mean figure of $5,617 per minute and a maximum of $11,086 per minute.

A data centre outage costs an organisation an average of $505,502 per event, with more than half of that total being attributable to lost revenue and "business disruption".

Most critical IT applications rely on the back-end database to be able to function, so it is critical that this component is as resilient as any other part of the IT application infrastructure to avoid the kind of costs discussed above. Typically this means adopting a fault-tolerant approach to eliminate any single point of failure, with the database functionality spread over multiple physical servers in either an "active/ passive" or preferably an "active/ active" configuration[2]. Consideration also needs to be given to the base resilience of the database hardware/software, as even the most sophisticated

It is critical that the database is as resilient as any other part of the IT application infrastructure. Typically this means spreading the database functionality over multiple physical servers in either an "active/active" or an "active/passive" configuration.

[1] In relational databases, the optimiser is the component responsible for analysing the SQL statement and selecting the access path to be used when it is executed.

[2] In an "active/active" high-availability configuration, incoming database work is split across multiple nodes/instances under normal conditions, with all nodes accessing a single set of shared disks containing the data (this is also known as a "shared disk" or "shared everything" cluster. In the event of hardware/software failure on one of the nodes, the remaining nodes continue to process incoming work with minimal disruption. An "active/passive" configuration also uses multiple nodes, but only one is active under normal conditions. In the event of a failure processing switches to the passive node after a short interruption while the "failover" takes place. Note that other definitions of these terms also exist, but the ones above will be used throughout this paper.

active/active architecture will still entail some temporary interruption to normal processing in the event of a component failure.

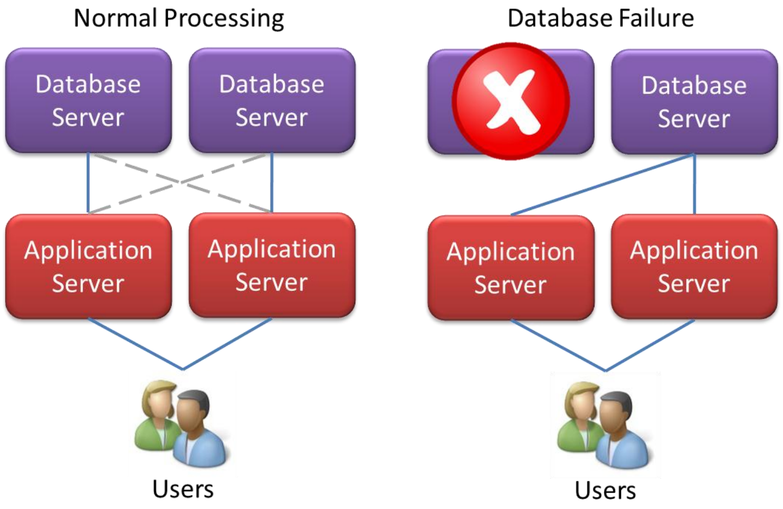


Figure 2 – High Availability Active/Active Database Architecture

As was the case with the performance factors previously discussed, the resilience of the database server can have a direct impact on the total cost of ownership if mission-critical applications are dependent upon it, and this aspect is discussed further in the TCO section below.

A further benefit of adopting an active/active database server architecture is the horizontal scalability that such an approach inherently offers. As enterprises enter new geographies or bring new products to market, data and transaction volumes for a given application can increase quickly. Company mergers and acquisitions can result in even more rapid data growth as applications are integrated.

Enterprise applications typically have to cope with huge data and transaction volumes, and some of these are subject to large fluctuations. Payment service provider GlobalCollect reported a 97% increase in US daily transaction volumes during the 2011 Thanksgiving holiday.

However such growth is not always linear in nature, and some applications have to cope with large workload fluctuations. For example, global payment service provider GlobalCollect reported a 97% increase in daily US transaction volumes during the 2011 Thanksgiving holiday (2). Even outside of these seasonal fluctuations, enterprises need to be able to rapidly and efficiently increase the capabilities of the IT infrastructure in order to support an increase in ongoing business.

In an active/active database architecture, the overall capacity of the system can be rapidly scaled up and down by adding or subtracting nodes from the cluster.

In an active/active database architecture, the overall capacity of the system can be rapidly scaled up and down by adding or subtracting nodes from the cluster. As all of the nodes access a single copy of the data on shared disk, there is no data “rebalancing” required so capacity can be easily added or subtracted according to the demands of the business.

Data Governance

Enterprise application databases typically contain many items of sensitive information, from customer name/address details to credit card numbers and confidential financial data. Keeping this data secure and limiting access to authorised personnel only is critical from both a legal and a corporate reputation perspective.

A Ponemon Institute report issued in 2011 found that the average cost of a data breach in 2010 was $4 million across all five countries in the benchmark, with the US average figure being an even more alarming $7.2 million.

A Ponemon Institute report issued in 2011 (3) examined the business impact of actual data breaches across 154 global organisations in 17 business sectors. The report found that the average cost of a data breach in 2010 was $4 million across all five countries in the benchmark, with the US average figure being an even more alarming $7.2 million. This corresponds to a global average of $156 per compromised record ($214 per record in the US). Of the various types of breach reported, malicious or criminal attacks were both the most common and the most expensive.

Of course, the database is only a single component in the overall IT security architecture, but for maximum resilience against any unauthorised attempt to obtain sensitive data it is critical that each of those components implements robust security measures. From a database perspective, this includes the provision of a comprehensive security layer that is able to integrate with existing corporate authentication mechanisms, support encryption of especially sensitive data[3] and implement fine grained access control to ensure that only the minimum required access is given.

A robust database security layer is only part of the story, however. Some form of audit capability is also required in order to allow regular compliance monitoring to take place, and to provide forensic data that can be retrospectively analysed in the event that any unauthorised access or data loss does occur.

Remote Access and Federation

In an ideal world, each enterprise would have a single DBMS hosting all of its operational and analytical data, and aggregating data from multiple application systems would be a simple as joining the relevant tables together in a single SQL statement. In reality, most enterprises have to contend with a vast array of application data stores that are spread over multiple database systems and physical server platforms. The ability to combine, or federate[4], these disparate data sources to allow them to be queried as if they reside in a single logical (and local) database is therefore a huge advantage for both operational and data warehouse applications.

However, physically being able to access multiple data sources is of limited value unless performance remains within acceptable limits. Therefore, any federation capability should also include the necessary distributed optimisation enhancements to be able to intelligently perform remote access in the most efficient way. For example, when joining a DB2 for z/OS table containing 1 million qualifying rows with a SQL Server table with 1,000 qualifying rows, the optimiser should move the SQL Server data to DB2 to perform the join and not the other way around.

The potential business benefits of efficient federated access to heterogeneous data stores are well documented. Staff and infrastructure costs can be reduced through simpler data access, business agility and responsiveness can be increased and regulatory compliance requirements can be met more quickly.

The potential business benefits of efficient federated access to heterogeneous data stores are well documented[5]. Staff and infrastructure costs can be reduced through simpler data access, business agility and responsiveness can be increased and regulatory compliance requirements can be met more quickly.

[3] As required by PCI DSS and other industry standards

[4] Data federation is also known as information integration, or data virtualisation.

[5] For an example, please see TDWI Checklist Report on data federation (12)

Application Support

A traditional enterprise System z application environment would consist of hand-written COBOL programs executing under the control of a batch scheduler or an OLTP monitor such as CICS or IMS/DC. The programs would typically reside in the same z/OS LPAR as the supporting database, and present a character-based “green screen” interface to the user.

Today, that “green screen” application interface has largely been replaced by a much richer GUI-based presentation layer, with messaging components such as MQ or CICS Transaction Gateway responsible for communicating with the back-end applications.

More modern distributed application architectures tend to make use of a three tier approach, with a rich GUI-based client (typically browser based) interacting with one or more application servers which are responsible for executing the business logic. Finally, the application servers access one or more database servers on the third tier.

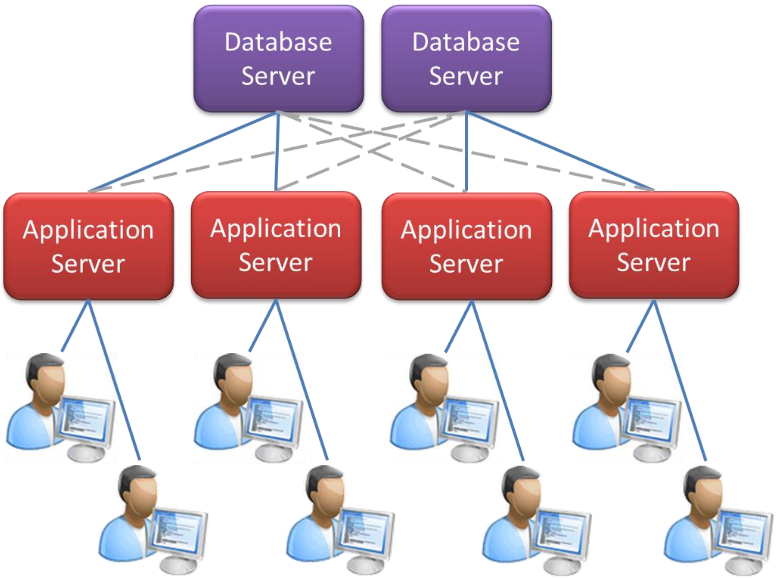


Figure 3 - Typical Three-Layer Architecture with Active/Active Database Servers

The widespread adoption of three-tier architectures has placed additional demands on the back-end RDBMS. The DBMS must be capable of accepting and efficiently handling incoming work from network-connected application servers. This could result in many hundreds or even thousands of incoming connections

The widespread adoption of this architecture has placed additional demands on the back-end RDBMS in an enterprise context. Firstly, the DBMS must be capable of accepting and efficiently handling incoming work from a network-connected remote client or application server. In the case of large enterprise systems with many application servers, this could result in many hundreds or even thousands of incoming connections.

The DBMS also has to support the most commonly-used communications protocols and application APIs, and ideally these will be based on open standards in order to avoid unnecessary vendor tie-in. In practice this typically means support for TCP/IP as the underlying network protocol, with API support for JDBC, ADO.NET and ODBC. As many of these APIs typically imply the use of dynamic SQL[6], efficient SQL statement caching and/or some form of dynamic to static SQL conversion capability can be critical in order to maintain acceptable performance.

The past decade has seen a significant increase in the popularity of packaged applications from vendors such as SAP, PeopleSoft and several others. Such applications are usually highly capable and configurable, and promise to allow enterprises to more rapidly deploy well-tested IT solutions for their most fundamental business processes (most of which tend to be very similar for organisations in the same business sector). This approach typically allows the enterprise to be more responsive and deliver new products to market more quickly, while freeing developers to concentrate on more advanced functionality that can differentiate the company and provide competitive advantage.

The past decade has seen a significant increase in the popularity of packaged applications from vendors such as SAP, PeopleSoft and several others. Adopting a packaged application imposes some restrictions on the supporting database infrastructure

Attractive though many of these benefits are, adopting a packaged application does impose some restrictions on the supporting database infrastructure. At the most basic level, the package must formally support the DBMS that will be used to host its data. But beyond this basic requirement is a host of additional considerations. What is the package vendor’s preferred database? How much cooperation is there between the package vendor and the database vendor to ensure both components work well together and are making use of the latest database features and enhancements? What database platform provides the best performance and cost profiles? All of these questions and more must be considered when selecting the RDBMS for a given package (or when selecting packages that are suitable for a chosen RDBMS).

[6] With dynamic SQL, each statement is prepared for execution at run time. This involves parsing the SQL, checking the syntax and authorisation, and performing optimisation in order to pick the best access path to the data. In a static SQL model, these activities are only performed when a program is first deployed, with an “access plan” being stored and used each time the SQL is executed. Static SQL is typically more efficient and easier to tune than dynamic SQL, although DBMS features such as dynamic SQL statement caching can close the performance gap considerably.

High Availability and Scalability

High availability and scalability have traditionally been seen as core strengths of the System z platform, and with very good reason. However, the capabilities of SQL Server have expanded beyond its roots as a departmental shared database server, and Microsoft are attempting to position it as a real alternative to System z. This section of the paper examines each product's capabilities more closely in this critical area.

High Availability

When evaluating the resilience of a mission-critical application, several aspects must be considered. Leaving aside the robustness of the application code itself[7] the server hardware and operating system must also be considered alongside the actual database software. In all cases, the cost and operational impact of both planned and unplanned outages must be taken into account.

Overview

This section will provide a brief overview of the approaches taken by IBM and Microsoft in architecting high availability for their respective database products.

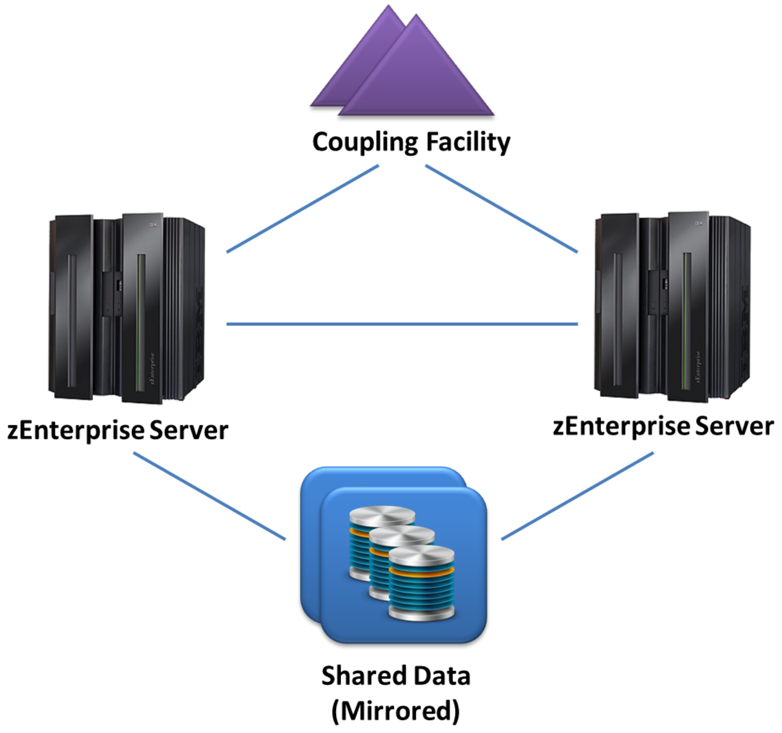


Figure 4 – Basic zEnterprise Parallel Sysplex Architecture

IBM's System z platform is generally accepted as being the most resilient technology available today.

IBM's System z platform is generally accepted as being the most resilient technology available today. It makes extensive use of redundant hardware within each server, seamlessly utilising spare components in the event of a failure without impacting the workload (and subsequently allowing failed components to be transparently

[7] The robustness of the application code is not considered in this paper as it is assumed equally robust code can be delivered for each RDBMS and it is therefore irrelevant from the perspective of a database platform evaluation.

replaced). These capabilities are further enhanced in a parallel sysplex environment, where multiple System z servers are able to share work in an active/active clustering approach (see Figure 2 – High Availability Active/Active Database Architecture on page 8). A parallel sysplex hardware configuration provides a solid and capable foundation that both z/OS and middleware services such as DB2 are able to exploit.

A parallel sysplex hardware configuration provides a solid and capable foundation that both z/OS and middleware services such as DB2 are able to exploit.

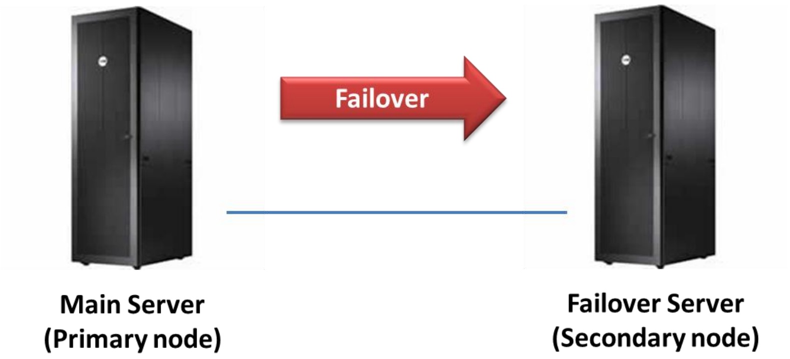


Figure 5 - Basic Windows Server Failover Cluster (WSFC) Architecture

Microsoft's recommended solution for SQL Server high availability is based on an active/passive architecture using the Windows Server Failover Clustering (WSFC) capabilities built into Windows Server 2008. Unlike parallel sysplex, WSFC uses a peer-to-peer architecture, with each node sharing metadata and notifications directly with other nodes in the cluster using standard network interconnects. Although Microsoft does have some minimum requirements for WSFC servers[8], no specialised components are used to support the clustering so resilience at the hardware level is down to each database server vendor.

SQL Server 2012 includes two new features called AlwaysOn Failover Cluster Instances (AlwaysOn FCI) and AlwaysOn Availability Groups. Both options depend upon WSFC functionality and therefore use a failover approach, but take slightly different routes to improving availability (see section on Database Software on page 16 for further details).

Unlike the active/active approach used by IBM's parallel sysplex technology, an entire application running in a WSFC environment may be unavailable for many minutes while the failover process takes place

Unlike the active/active approach used by IBM's parallel sysplex technology, an entire application running in a WSFC environment may be unavailable for many minutes while the failover process takes place. It should also be noted that the WSFC architecture is designed as a high availability configuration only, and its use precludes the use of other clustering approaches for performance and scalability requirements[9].

Hardware and Operating System

The following discussion will compare the latest generation of server hardware for DB2 for z/OS (the IBM zEnterprise z114 and z196) and SQL Server (an x86-based server configured for mission critical applications, such as the NEC Express5800/R320 Series).

[8] From Windows Server 2008 onwards, Microsoft no longer maintains a Hardware Compatibility List (HCL). A "cluster validation wizard" is supplied instead, which will test a given configuration to see if it meets the minimum requirements.

[9] Other Microsoft clustering solutions include Network Load Balancing and the Compute Cluster Server.

> ***The System z environment was originally conceived and designed as a highly resilient business platform. A single IBM zEnterprise server has a quoted MTBF of 30 years, and the overall availability is increased still further when the zEnterprise server is integrated in a parallel sysplex.***

The System z environment was originally conceived and designed as a highly resilient business platform. Extensive use of redundancy and advanced fault-toleration techniques result in an industry-leading Mean Time Between Failure (MTBF[10]): a single IBM zEnterprise server running z/OS V1.12 has a quoted MTBF of 30 years (4). The overall availability is increased still further when the zEnterprise server is integrated in a parallel sysplex, where other servers can quickly and efficiently take on the workload if a server does happen to fail.

The same hardware redundancy that protects against unplanned outage also allows planned outages to proceed with minimal impact in a zEnterprise server. Many individual hardware components[11] can be replaced or upgraded without impacting the availability of the critical applications, as the redundant spare will take on the load. In a parallel sysplex environment, entire zEnterprise servers can even be swapped out while the workload is redirected to the other members in the sysplex.

As far as data storage is concerned, IBM's DS8000-series make extensive use of RAID technology to provide fault-tolerance for individual disks. In the event of a failure in the entire storage subsystem, synchronous and aynchronous mirroring capabilities provide additional protection, and work withthe HyperSwap capability to seamlessly allow the secondary copy of the data to be used.

> ***IBM's unique position as the vendor of both the hardware and the operating system allows it to deeply integrate the two and ensure that enhancements in one are rapidly and effectively exploited in the other***

IBM's unique position as the vendor of both the hardware and the operating system allows it to deeply integrate the two and ensure that enhancements in one are rapidly and effectively exploited in the other. This deep integration also pays dividends in the event of a system or component failure, where faciltiies such as the Sysplex Failure Manager (SFM) and Automatic Restart Manager (ARM) can rapidly and autonomously take the relevent recovery actions to maintain availability.

> ***The MTBF for a single x86 server is widely regarded as being measured in months rather than decades.***

The Intel x86 architecture originated in the late 1970's and was designed for use in personal computers. Since that time it has also been pressed into much more demanding server duties and has therefore been incrementally enhanced to improve both its processing power and reliability. Most "normal" x86 servers offer limited hardware redundancy[12] that stops a long way short of that implemented with a System z server. As a result the MTBF for a single x86 server is widely regarded as being measured in months rather than decades (x86 server suppliers do not generally publish MTBF figures).

In recognition of these limitations, some vendors have gone further to try and improve x86 hardware fault tolerance. For example the NEC Express5800/R320 is still based on an Intel Xeon CPU[13], but the manufacturer has effectively implemented two complete motherboards in a single 4U server (see Figure 6 on page 15). The

[10] The Mean Time Between Failure (MBTF) is the predicted elapsed time between inherent / unplanned failures of a system during operation.

[11] This includes power supplies, processor books, memory, cryptographic cards, and coupling links.

[12] For example, a top of the range Dell PowerEdge R910 offers redundant power supplies, cooling fans and networking. Note that some online hardware configuration changes (such as adding memory or CPU) are only supported by SQL Server Enterprise Edition.

[13] Xeon is Intel's most popular x86 server CPU. It also produces the Itanium line of processors which are specifically designed for high-availability fault tolerant servers, but in April 2010 Microsoft announced that it would no longer be supporting Itanium-based systems beyond Windows Server 2008 R2 (14). Therefore these systems have not been considered in the SQL Server hardware discussion above.

unit appears externally as a single logical server, and specialised internal hardware keeps both motherboards in strict “lockstep” during normal processing. In the event of a critical failure of any component on the primary motherboard, processing switches immediately to the secondary unit.

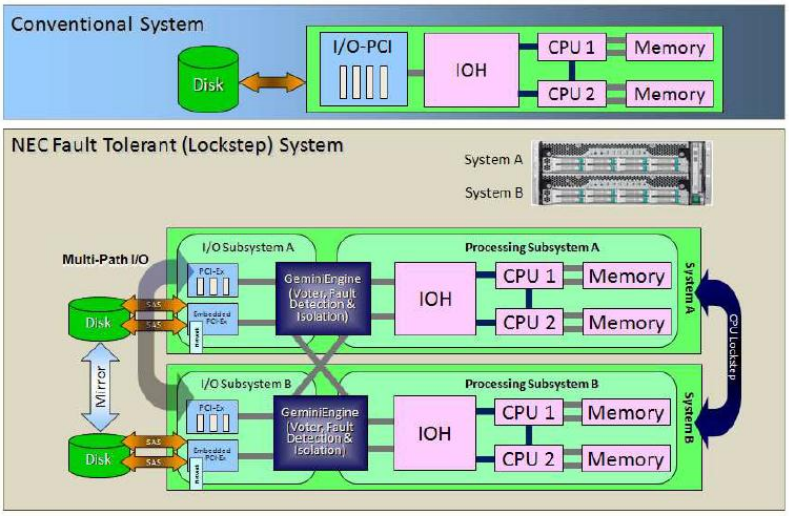


Figure 6 - NEC Express5800 Fault Tolerant Server Architecture

At face value this appears to at least approach the redundancy offered by IBM’s System z architecture, with each of the NEC’s CPU modules being somewhat analogous to a “book” in a System z server. However, there are some crucial differences:

- Although the NEC server offers 100% redundancy at the CPU unit level, the individual components (processor, RAM, etc.) are relatively standard items which are not themselves specifically engineered for high resilience. Therefore the chance of a given component (and therefore a whole CPU unit) failing remain higher than with a System z server.

- The NEC system only supports a maximum of two CPU units, so once processing has switched to the alternate CPU unit the system is no longer fault tolerant until the failed unit is replaced. System z servers support up to four books per server[14], and also support fault tolerance at the sub-book level so that the failure of an individual component (such as an I/O card) will not cause the entire book to become inoperative.

In addition to x86 server hardware being inherently less resilient than its System z counterpart, the use of Microsoft clustering services also falls far short of the benefits provided by IBM’s parallel sysplex. Microsoft’s recommended approach is to use Windows Server Failover Clustering (WSFC) which is implemented at the operating system level (see below). Parallel sysplex utilises specialised hardware (in the form of the Coupling Facility and special high-speed interconnects) to optimise communication between each zEnterprise server and make an active/active architecture possible from a performance perspective. In contrast, WSFC has to rely on commodity hardware and is implemented

In addition to x86 server hardware being inherently less resilient than its System z counterpart, the use of Microsoft clustering services also falls far short of the benefits provided by IBM’s parallel sysplex

[14] With each book hosting up to 80 processing units.

entirely within the OS, and therefore uses an active/passive setup in order to maintain acceptable performance.

Database Software

DB2 for z/OS was designed from the ground up to handle large-scale, mission critical enterprise workloads, and many customers have been using it as such for 25 years. However, even the most robust and well-tested software is prone to human error so a single DB2 subsystem still represented a single point of failure for critical applications.

DB2 users are able to use IBM's parallel sysplex technology to eliminate a DB2 subsystem, z/OS logical partition (LPAR) or System z server as single points of failure.

When DB2 V4 was made available in November 1995, DB2 users were able to use IBM's parallel sysplex technology to eliminate a DB2 subsystem, z/OS logical partition (LPAR) or System z server as single points of failure. This feature, known as data sharing, allows multiple DB2 subsystems (or "members") to access a single copy of the data, with data integrity being enforced by an additional component known as the Coupling Facility. A single "data sharing group" can contain up to 32 DB2 subsystems.

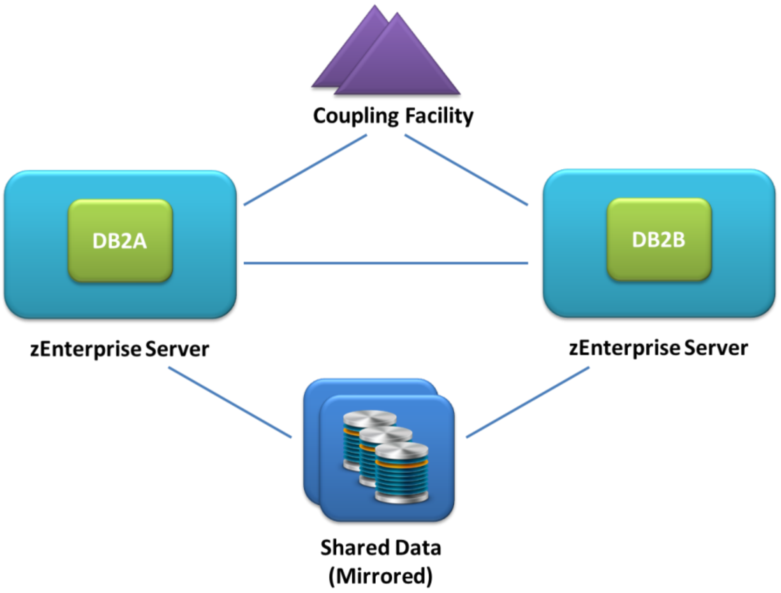


Figure 7 - DB2 Data Sharing Using Parallel Sysplex

As each DB2 member can reside on a separate System z server, loss of an individual server and/or a DB2 member still allows processing to continue on the surviving DB2 systems[15]. Subsequent releases of DB2 have systematically enhanced Data Sharing, supporting duplexed Coupling Facilities, improving performance, reducing restart times and adding additional autonomic repair capabilities.

To truly achieve high availability planned outages must also be minimised. Most common changes to a DB2 database schema and housekeeping operations can be accomplished online. A data sharing group can even be upgraded to a new release of DB2 while maintaining application access.

Of course, avoiding unplanned outages is only part of the challenge and to truly achieve high availability planned outages must also be minimised. Most common changes to a DB2 database schema can be accomplished online, as can routine housekeeping operations such as statistics collection and reorganisation. A data sharing group can even be upgraded to a new release of DB2 while

[15] Note that any data being updated on the failing system will remain locked until that system can be restarted, for integrity reasons. This process is usually automated, so the locked data is typically available again seconds (or in some cases minutes) after a failure. In the meantime, processing against all of the other application data can continue on the surviving members as usual.

maintaining application access, with each member being shut down, upgraded and restarted in turn[16]. With these capabilities, it is not uncommon for DB2 applications to run for many months or even years without any significant loss of availability.

Microsoft SQL Server was originally developed from the Sybase codebase, aimed at departmental and low-end enterprise data serving requirements. Since that time it has undergone significant change in an attempt to better position it for more demanding applications. SQL Server 2012 was released in March 2012 and contains a number of interesting new features aimed specifically at improving availability. In particular, AlwaysOn Failover Cluster Instances (FCI) and AlwaysOn Availability Groups aim to provide protection against hardware/OS/database failures using different forms of active/passive clustering[17]. Both techniques depend upon the WSFC (Windows Server Failover Clustering) services provided by Windows Server 2008.

AlwaysOn FCI uses multiple WSFC nodes which access shared disk storage (such as a SAN) as depicted in Figure 8 - SQL Server AlwaysOn Failover Cluster Instance, below. An FCI consists of a single active node and one or more passive nodes, which all present themselves to the outside world as a single SQL Server instance.

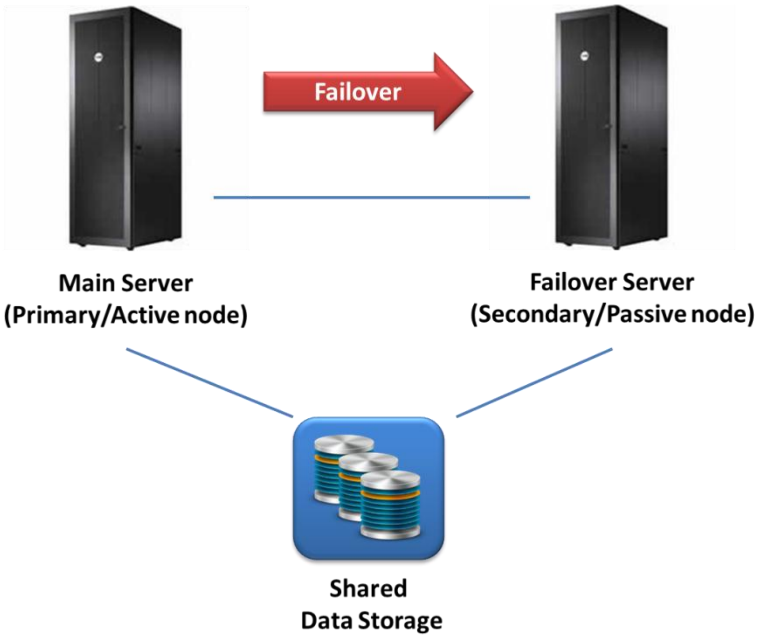


Figure 8 - SQL Server AlwaysOn Failover Cluster Instance

In the event of a failure on the FCI primary node, failover is automatically initiated and the SQL Server instance will re-start on the secondary node and perform standard recovery tasks. This means that all of the application data is unavailable from the moment the active note fails until the restart/recovery is complete on the passive node.

In the event of a failure on the FCI primary (active) node, failover is automatically initiated and the SQL Server instance will re-start on the secondary node and perform standard recovery tasks. Note that this means that all of the application data is unavailable from the moment the active note fails until the restart/recovery is complete on the passive node.

[16] This capability is known as a “rolling upgrade”, and exactly the same approach can be taken for applying routine preventative maintenance to DB2 while maintaining availability.

[17] Note that some SQL Server documentation states that “active/active” clustering is possible. This refers to two **separate** SQL server instances hosted on different servers, with each one being able to fail over to a passive node on the other server. This is not true active/active clustering according to the definition in this paper, as the instances are separate and application workload must be specifically directed at one or the other.

Microsoft's alternative solution, known as an AlwaysOn Availability Group, removes the requirement for a shared disk and depends on synchronous replication[18] between locally-attached disks for each WSFC node. Once again a single active node (known as the Primary Replica) handles all incoming work, but each time a transaction is committed on the primary replica the associated log records are synchronously written to the secondary replica which then applies the updates to its copy of the data. In the event of a failure in the primary replica, automatic failover will route workload to the secondary replica, which has an up-to-date copy of all committed data.

Microsoft's alternative solution, known as an AlwaysOn Availability Group, removes the requirement for a shared disk and depends on synchronous replication between locally-attached disks for each WSFC node.

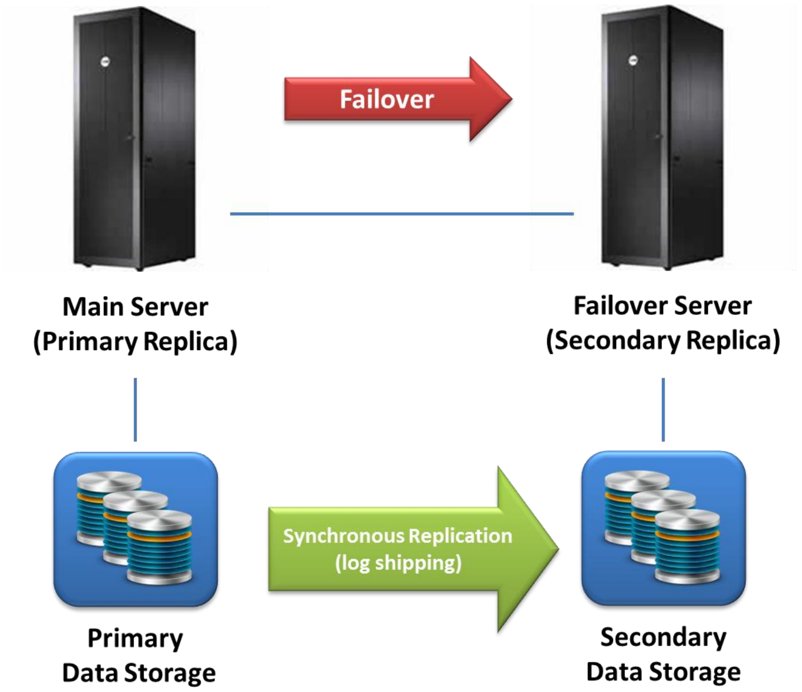


Figure 9 - SQL Server AlwaysOn Availability Group

As no recovery has to take place following failover, this solution provides quicker access to data in the event of a failover. However, all transactions running at the time of the failure will still be terminated and clients will be forced to reconnect to the secondary replica. This solution also entails additional overhead under normal processing, due to the need to synchronously transmit committed log records to the secondary replica. Finally, any new objects created on the primary replica are not available for recovery until they have been fully synchronised on the secondary replica, leading to a possible recovery exposure if a failover occurs in the meantime.

This solution also entails additional overhead under normal processing, due to the need to synchronously transmit committed log records to the secondary replica

Although differences do exist, SQL Server has broadly the same online schema and housekeeping capabilities as DB2. Manual failover can also be used with either of the clustering solutions to allow for hardware/OS maintenance and upgrades. However, none of the SQL Server solutions currently available match DB2's capabilities for rolling version upgrades.

SQL Server has broadly the same online schema and housekeeping capabilities as DB2, but none of the SQL Server solutions currently available match DB2's capabilities for rolling version upgrades.

[18] Note that it is possible to configure AlwaysOn Availability Groups to use asynchronous replication to reduce the performance impact, but automatic failover is not supported and Microsoft only recommends its use for disaster recovery scenarios.

Disaster Recovery Considerations

The previous discussion has focused primarily on high availability within a single-site context, but enterprises also need to be able to safeguard their critical applications in the event of a more major event that may result in the total loss of an entire data centre. This capability is commonly referred to as "disaster recovery", and will typically involve the use of additional data centres that are geographically distant from the primary location. In the event of a disaster at the primary data centre, processing can be transferred to the remote location, with varying degrees of impact to the application.

Two commonly-used criteria for evaluating the effectiveness of a disaster recovery solution are:

- **Recovery Time Objective (RTO**). This represents the maximum elapsed time between the disaster occurring and the application being available at the remote site.
- **Recovery Point Objective (RPO)**. This represents the maximum amount of processing time lost in the event of a disaster (i.e. the amount of time between the recovery point and the disaster). The lower this figure is, the more current the data will be at the remote site.

DB2 for z/OS offers a wide number of solutions to address disaster recovery requirements. For enterprises that have more demanding RTO/RPO requirements, the high availability capabilities inherent in the parallel sysplex architecture can be extended to encompass multiple physical locations using facilities such as GDPS (Geographically Dispersed Parallel Sysplex) and HyperSwap (which allows a remote secondary disk subsystem to seamlessly take over in the event of the primary subsystem being lost). These capabilities make it possible to implement a disaster recovery solution with RTO/RPO of a few seconds or less.

DB2 for z/OS offers a wide number of solutions to address disaster recovery requirements. These capabilities make it possible to implement a disaster recovery solution with RTO/RPO of a few seconds or less.

SQL Server 2012 also offers disaster recovery options that are built upon the local high availability options previously discussed. FCI failover to a remote site is possible (provided that a suitable storage replication facility is in place) and similarly the failover server in an AlwaysOn Availability Group can be situated remotely. However, both of these suffer from the same restrictions and drawbacks as the single-site solutions, and involve significantly greater RPO and/or RTO times than DB2.

SQL Server 2012 also offers disaster recovery options that are built upon the AlwaysOn capabilities. However, both of these suffer from the same restrictions and drawbacks as the single-site solutions, and involve significantly greater RPO and/or RTO times than DB2.

Conclusions

The table below attempts to summarise the key differences in the high availability capabilities of DB2 data sharing, SQL Server AlwaysOn Failover Cluster Instances and SQL Server AlwaysOn Availability Groups.

	DB2 Data Sharing	SQL Server FCI	SQL Server High Availability Group
Hardware / OS Integration	High	Low	Low

	DB2 Data Sharing	SQL Server FCI	SQL Server High Availability Group
OS / Database Integration	High	High	High
Hardware resilience	High	Low/Medium (dependent on database server configuration)	
Clustering approach	Active/ Active using Parallel Sysplex	Active/Passive using WSFC	
Maximum number of active read/write nodes in the cluster	32	One[19]	
Can transactions continue running when server fails?	Yes, for transactions running on surviving members	Only a single read/write server can be active at any one time, so all transactions are aborted when the active server fails.	
Proportion of data available immediately following server loss	All, except data actually being updated at time of failure	None	None
Time until all data available following server loss	Seconds to minutes, while failed subsystem is restarted	Minutes, while failover to standby server and instance recovery takes place	Seconds to minutes. while failover to standby server takes place
Solution Maturity	High – DB2 data sharing in use since mid-1990s, thousands of customers worldwide	Medium – based on function available in previous releases	Low – new facility available in SQL Server 2012 which has just been released.
Performance Impact	Low – 0-5% for a well-designed application	Low	Medium (synchronous log shipping overhead)
Support for rolling version upgrade	Yes	No	No
Online Schema Change	Yes	Yes	Yes

[19] Note that it is possible to configure a secondary replica of an AlwaysOn Availability Group to allow read-only access, but read/write is not supported.

	DB2 Data Sharing	SQL Server FCI	SQL Server High Availability Group
Online Housekeeping	Yes	Yes	Yes
Disaster Recovery Capabilities	High	Medium	Medium

Table 1 - Summary of High Availability Characteristics

The new capabilities in SQL Server 2012 have closed the gap somewhat, but the inherent resilience of the System z platform and the additional protection offered by DB2 data sharing combine to offer an unmatched level of availability with minimal performance impact.

In conclusion, the new capabilities in SQL Server 2012 have closed the gap somewhat, but the inherent resilience of the System z platform and the additional protection offered by DB2 data sharing combine to offer an unmatched level of availability with minimal performance impact.

Scalability

In this context, scalability refers to the ability of the RDBMS to cope with large data and/or transaction volumes. As growth can happen quite suddenly, the RDBMS should ideally be able to handle large increases without having to make significant changes to the application/database design or suffer significant application outage.

Data Growth

When dealing with ever-increasing data volumes one of the main tools that the DBA's disposal is table partitioning. This allows large tables to be split into more manageable subsets, with potential advantages for manageability, performance and recovery.

DB2 10 for z/OS allows up to 4,096 partitions to be defined on a single table. This is enough to provide good flexibility and give the DBA plenty of options when deciding on the partitioning strategy for a given table. For example, if the DBA decides to explicitly partition a table according to date, 4,096 partitions allows one partition per day for over 11 years – more than enough to cope with the historic data retention requirements for most applications. DB2 also provides facilities to "rotate" partitions, allowing partitions containing old data to be emptied and re-used for new data without having to perform any data actual movement.

If no suitable partitioning key is available, the DBA can instead use the Partition By Growth (PBG) option introduced in DB2 9 for z/OS. This still allows a table to be partitioned for manageability, but rows are simply added to the current partition until it fills up, whereupon DB2 allocates the next partition and starts using that one.

Partitioning can be implemented using DB2's dynamic schema change capabilities, allowing a non-partitioned table to be partitioned (or changes made to an already partitioned table) with minimal application impact[20]. Partitioning is purely a physical measure, and partitioned tables still appear as a single logical table

[20] Typically, such changes are implemented by executing a series of SQL ALTER statements, followed by a REORG to allow the new structure to be materialised. If an online REORG is used, application access may continue throughout the entire process, except for a few seconds at the end when access must be switched to the shadow REORG copy.

from an application perspective. Therefore, no application changes are typically required when partitioning is implemented.

The maximum size of a single partitioned table in DB2 is 128TB which is enough for the vast majority of applications, especially if the table is compressed[21]. If necessary this limit can be bypassed by creating multiple tables and connecting them with UNION statements in SQL, but this complicates applications and can have performance drawbacks.

Partitioning is fully compatible with DB2 data sharing (see section on Transaction/Workload Growth on page 24) so the query workload for large tables can be split across multiple DB2 systems on multiple System z servers. There is no requirement to route workload for a given partition to a specific DB2 member (although there may be some performance advantages for doing so in some circumstances).

Many customers implement large databases on DB2. For example, HM Land Registry in the UK runs the world's largest known OLTP database (71 TB) on DB2 for z/OS.

Many customers implement large databases on DB2. For example, HM Land Registry in the UK runs the world's largest known OLTP database (71 TB) on DB2 for z/OS.

SQL Server also supports the concept of table partitioning, and offers many of the same advantages and capabilities as DB2's implementation. However, there are some important differences:

- SQL Server supports up to 15,000 partitions for a single table, giving somewhat more flexibility than the 4,096 supported by DB2.

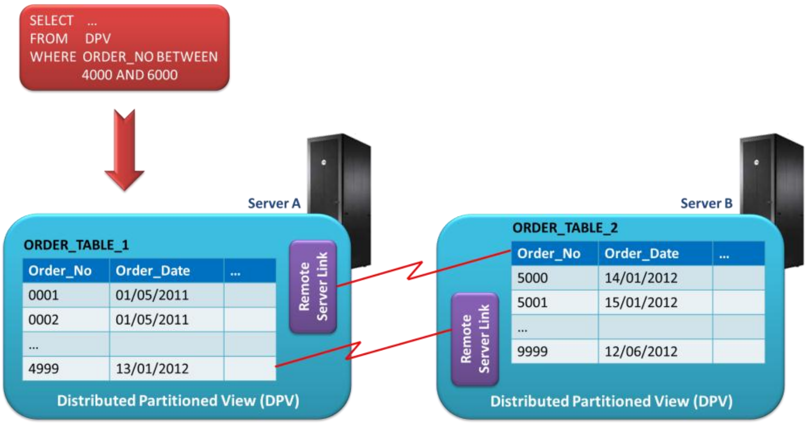
- There is no provision in SQL Server for partitioning an existing table using online schema change. Therefore, if partitioning is not implemented as part of the initial database design it will be necessary to unload the existing data, drop and recreate the table and then reload the data. Depending on the amount of data in the table, this could result in a significant application outage.

- SQL Server only supports native partitioning within a single database, which must reside on a single SQL server instance/node. SQL Server's various clustering options (described in the section on Transaction/Workload Growth on page 24) do not allow for multiple active read/write nodes in a cluster.

- In order to allow more than one server to access a given table, it must be split into multiple sub-tables which are then placed on separate SQL Server instances/servers. This can either be accomplished using a Distributed Partitioned View, (DPV, see Figure 10 below) or Data-Dependent Routing (DDR, see Figure 11). Either solution requires significant application changes and outages to implement, and imposes additional complexity and DBA overhead due to having to separately maintain multiple copies of the various

[21] Enabling compression for a table tells DB2 to build a "dictionary" of commonly repeated sequences within the data and replace each instance with a much smaller dictionary token. Depending on the data, it is not uncommon to see compression ratios of 30-40% for a typical OLTP table. Compression is often considered to be low cost or "free" in terms of CPU overhead, as all System z servers have dedicated co-processors for handing compression and decompression tasks.

table definitions. DPVs can also cause performance issues due to network overheads[22].



ORDER_TABLE_1

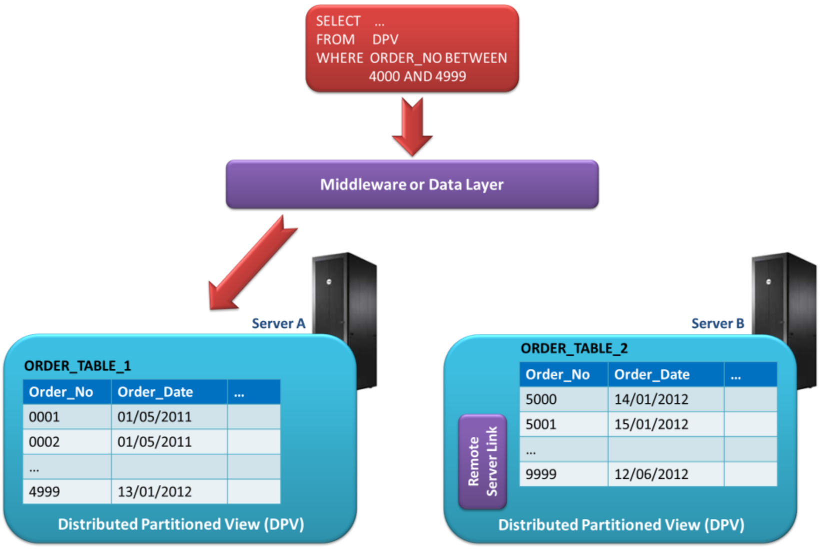
Order_No	Order_Date	...
0001	01/05/2011	
0002	01/05/2011	
...		
4999	13/01/2012	

ORDER_TABLE_2

Order_No	Order_Date	...
5000	14/01/2012	
5001	15/01/2012	
...		
9999	12/06/2012	

Figure 10 - SQL Server Distributed Partitioned View

In Figure 10 above, a client connected to Server A issues an SQL statement against the DPV. The data is retrieved from the local table (ORDER_TABLE_1) and joined with data fetched across the network from the remote table (ORDER_TABLE_2) on Server B. SQL statements issued by clients connected to Server B would do the same thing in reverse.



ORDER_TABLE_1

Order_No	Order_Date	...
0001	01/05/2011	
0002	01/05/2011	
...		
4999	13/01/2012	

ORDER_TABLE_2

Order_No	Order_Date	...
5000	14/01/2012	
5001	15/01/2012	
...		
9999	12/06/2012	

Figure 11 - SQL Server Data-Dependent Routing

Data Dependent Routing removes the network performance overheads of DPVs by introducing an additional middleware or data layer which decides which sub-table contains the necessary data and routes the query to the correct server. This has better performance characteristics, but precludes any queries that need data from more than one sub-table/server and therefore places major restrictions on the database and application design. It should also be noted that this technique does not rely on any capabilities within SQL Server, and could just as easily be implemented on DB2 for z/OS.

[22] DPVs use SQL Server's federation capabilities to access data stored in another instance. All qualifying rows from the DPV that reside on a remote server must be fetched to the server running the client's query, which can cause performance issues due to network delays.

Transaction/Workload Growth

When fully configured IBM's zEnterprise servers are capable of handling the world's most demanding workloads, but even the lower-end models boast impressive computing capabilities. The table below summarises some key specifications for the lowest (z114 2818-M05) and the highest (z196 2817-M80) models in the zEnterprise range, to demonstrate the breadth of the platform's capabilities.

When fully configured IBM's zEnterprise servers are capable of handling the world's most demanding workloads, but even the lower-end models boast impressive computing capabilities

Processor Units (Cores)	1-80[23]
Clock Speed	3.8 – 5.2 GHz
zMIPS[24]	26 – 50,000
Main Memory (GB)	8 – 3,056
Internal Bus Speed	8 GB/sec
I/O Interconnects	6 GB/sec

Table 2 - z114 / z196 Specification Range

As previously discussed in the section on high availability, the majority of these upgrades can be performed online with little or no disruption to active workloads. In a parallel sysplex environment, even "disruptive" upgrades can take place with no workload interruption as individual System z servers can be taken offline while work continues to be processed on the others.

In a parallel sysplex environment, even "disruptive" upgrades can take place with no workload interruption as individual System z servers can be taken offline while work continues to be processed on the others.

Of course, the ability of the hardware to effectively scale is only part of the story. The RDBMS and application infrastructure must also be able to cope with the additional demands that significant increases in transaction volume will inevitably generate. Once again, the pedigree of DB2 as a purpose-built enterprise data serving platform stands it in good stead.

A single stand-alone DB2 for z/OS subsystem is capable of handling a significant workload. In previous releases the maximum number of concurrent connections to a single subsystem was usually limited by virtual storage constraints, with practical limits of 400-500 concurrent connections being common (depending on the type of workload). DB2 10 for z/OS removed most of these constraints, and is typically able to support 5,000 concurrent connections or more within a single DB2 system[25].

An increase in the number of concurrent active transactions typically places more stress on other aspects of the database system, with recovery log writes and application locking being the two most common examples. DB2 10 delivered further enhancements for log write performance, with over 180MB per second measured in IBM tests[26].

[23] Note that a fully-configured z196 2817-M80 actually has 96 cores for customer use, but only 80 of these are configurable by the customer. The remaining 16 are reserved for System Assist Processors (to run I/O processing) and as spares in the event of a core failure.

[24] It should be noted that the zMIPS figure relates to IBM's internal measurement of the number of System z Million Instructions per Second achievable on the various models of zEnterprise server. No comparison can, or should, be drawn between this figure and the MIPS capabilities occasionally published for other CPUs such as Intel's, as the two measurements are not related.

[25] Note that the theoretical limit for concurrent connections to a single DB2 10 for z/OS subsystem is actually 20,000, but other factors such as available real storage make the practical limit somewhat lower.

[26] Figures taken from IBM DB2 10 for z/OS Performance Topics Redbook (16) and are based on asynchronous log writes to a DS8800 using zHPF (High Performance FICON).

Application locking can be more problematic and is largely down to good application and physical database design, but DB2 offers a number of features designed to assist with this issue including: row level locking, uncommitted read, optimistic lock semantics and most recently the ability to read through locks to access the most recently committed (consistent) version of a row that is currently being updated.

DB2 Data Sharing also provides near-linear workload scalability. Additional DB2 subsystems can easily be added to an existing data sharing group in order to increase the overall capacity

In addition to the considerable availability benefits previously discussed (see Database Software on page 16), DB2 Data Sharing also provides excellent (and near-linear[27]) workload scalability. As shown in Figure 12 below, additional DB2 subsystems can easily be added to an existing data sharing group in order to increase the overall capacity (up to a maximum of 32 subsystems). As all subsystems access a single set of shared disks, such increases do not require any data rebalancing or other application changes.

Additional subsystems can either be added on an existing zEnterprise server if spare capacity exists (as shown in Figure 12 in the example on the right), or on a completely separate System z server (shown on the left). In this way, the single logical system image of a parallel sysplex can greatly exceed the capacity of any single System z server[28].

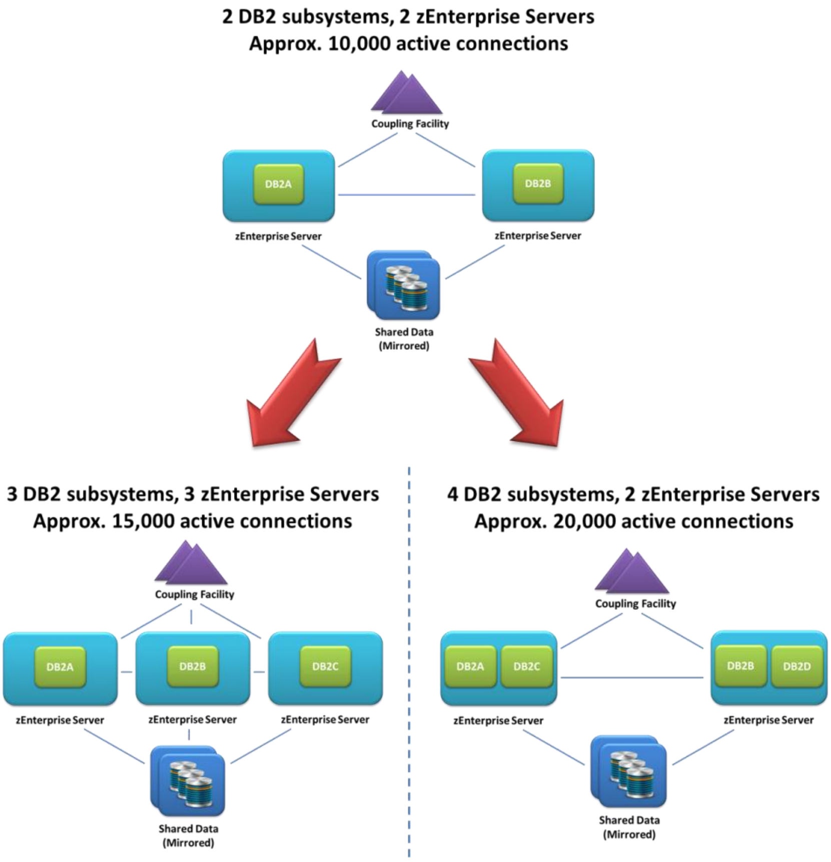


Figure 12 - Horizontal Scaling using DB2 Data Sharing

This approach also provides great flexibility for dealing with shifting workload patterns. Customers that have to deal with cyclical peaks

[27] Overheads for data sharing vary according to the workload and application design, but are generally accepted to be around 0.5% per additional data sharing member.

[28] The capacity of a data sharing group can be expanded all the way up to the theoretical maximum of 32 subsystems hosted by 32 separate z196 servers, although in practice it is rare to see more than 8-10 members in a data sharing group.

in their workload (such as retail organisations) can define additional DB2 members which are usually dormant, but get activated at peak times in order to accommodate the additional workload.

The logistics firm UPS regularly process in excess of 1.1 billion SQL statements per hour at peak periods. HM Land Registry in the UK runs the world's largest known OLTP database (71 TB) on DB2 for z/OS.

Enterprise customers have proven the scalability of the System z platform time and time again in the real world. Figure 13 below shows two benchmarks[29] performed on System z using TCS BaNCS banking platform. Both environments used parallel sysplex and DB2 data sharing to provide near-linear scalability as the transaction rate increased. The logistics firm UPS regularly process in excess of 1.1 billion SQL statements per hour at peak periods. HM Land Registry in the UK runs the world's largest known OLTP database (71 TB) on DB2 for z/OS.

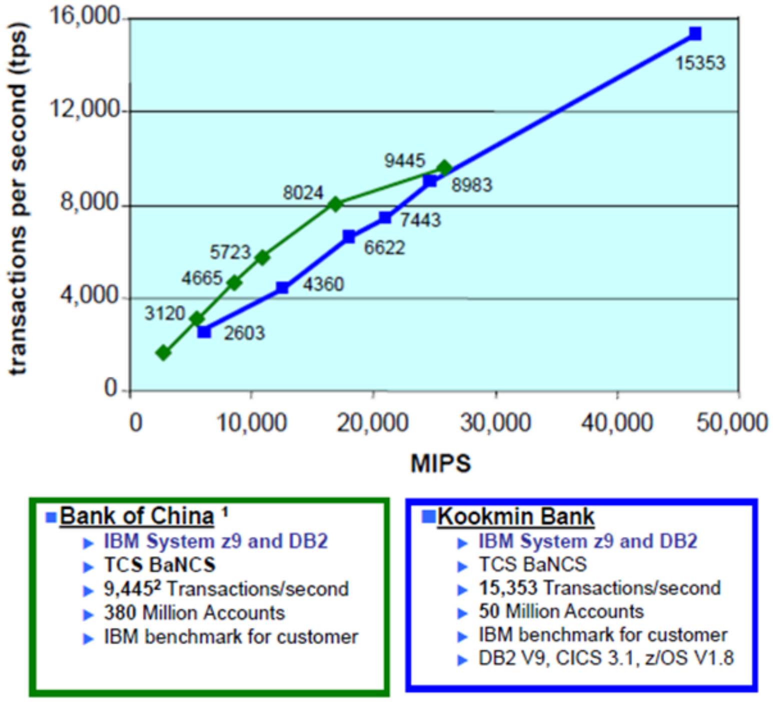


Figure 13 - System z and BaNCS Online Banking Benchmarks

Most Xeon based servers are inherently less powerful than even a low-end zEnterprise z114 in terms of processor and I/O capacity. Some powerful options do exist, but an SQL Server customer must make a choice between high availability or good scalability in terms of the server hardware.

Most Xeon based servers are inherently less powerful than even a low-end zEnterprise z114 in terms of processor and I/O capacity, but some powerful options do exist. Top-end servers such as the HP ProLiant DL980 G7 support up to eight Xeon E7-4870 processors, for a total of 80 processing cores and can use up to 2TB of RAM. However, it should be noted that the above server does not include the advanced fault-tolerance and redundancy included in the NEC Express5800/R320 covered earlier, and the NEC has a much lower maximum specification. The SQL Server customer must therefore make a choice between high availability or good scalability in terms of the server hardware.

Processor Cores	24-80
Clock Speed	1.86 – 2.4 GHz
Main Memory (GB)	8 – 2,056

Table 3 – HP ProLiant DL980 G7 Xeon Server Specification Range

[29] Taken from IBM presentation entitled "zEnterprise – The Ideal Platform for Smarter Computing" (17)

Unlike IBM's parallel sysplex, the WSFC clustering architecture does not provide any assistance in scaling up workloads that have outgrown the capacity of a single server. For most OLTP applications the maximum scalability of an SQL Server instance is restricted by the processing capacity of a single Xeon server unless the application has been specifically written to use Distributed Partition Views or Data-Dependent Routing.

Unlike IBM's parallel sysplex, the WSFC clustering architecture does not provide any assistance in scaling up workloads that have outgrown the capacity of a single server, as both flavours of AlwaysOn clustering only allow a single read/write WSFC node to be active at once[30]. Therefore, for most OLTP applications[31] the maximum scalability of an SQL Server instance is restricted by the processing capacity of a single Xeon server unless the application has been specifically written to use Distributed Partition Views or Data-Dependent Routing.

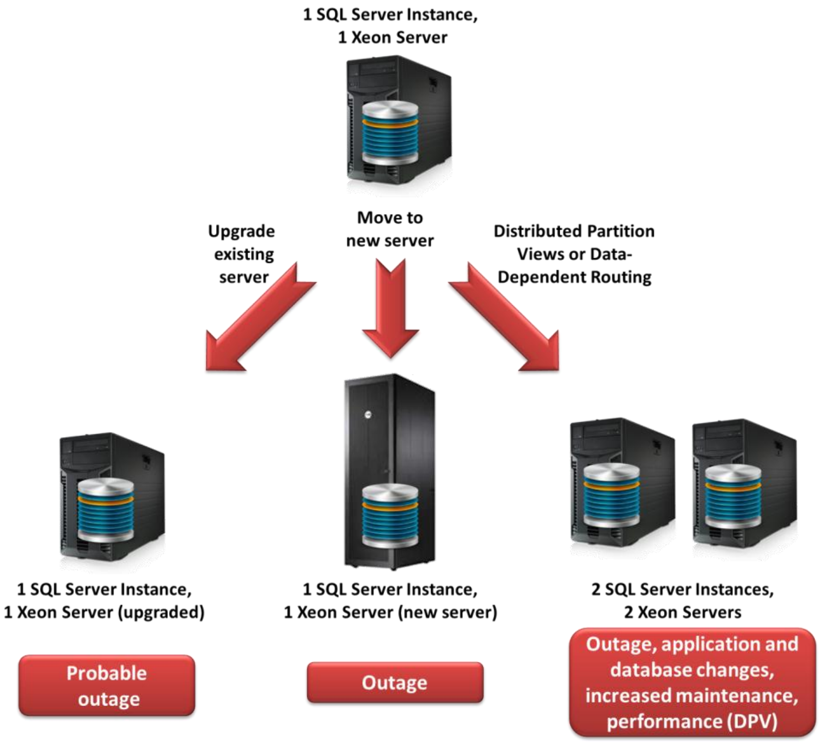


Figure 14 - SQL Server Scalability Options

If a particular workload outgrows the maximum capacity of the database server hosting it, the SQL Server DBA has limited options, which will require major changes to the application and database requiring significant time, effort, risk and application outage to implement.

If a particular workload outgrows the maximum capacity of the database server hosting it, the SQL Server DBA has limited options, as summarised in Figure 14 above. He could consider moving the entire database to a larger capacity server (assuming one is available, and the organisation is able to cope with the associated application outage). Alternatively, the workload could be split across multiple database servers by horizontally partitioning the tables across several instances by using Distributed Partitioned Views or Data-Dependent Routing (as described earlier in the section on Data Growth on page 21). However both of these require major changes to the application and database which will entail significant time, effort, risk and application outage to implement. They also require duplicate table definitions to be created and maintained on each server, adding DBA overhead and complexity.

[30] Note that it is possible to configure a secondary replica of an AlwaysOn Availability Group to allow read-only access, but read/write is not supported.

[31] For some read-only or very low update applications, other SQL Server scale-out solutions are possible. These include Scalable Shared Databases (read only) and Peer-to-Peer Replication (no conflict resolution, can't handle high update frequency due to replication performance impact) . These are not considered further as they do not lend themselves to typical enterprise OLTP requirements.

Conclusions

The table below attempts to summarise some of the key differences in the scalability features of DB2 for z/OS and SQL Server.

It should be noted that the SQL Server Data Dependent Routing has not been included in this comparison, as the approach does not rely on any specific features of SQL server and could also be implemented in DB2 for z/OS.

	DB2 with Data Sharing	SQL Server	
		Native Database Partitions	DPV
Maximum Database size	Only limited by available storage	524,272 TB	524,272 TB for each sub-table
Maximum number of partitions	4096	15,000	256 (max tables in a UNIONed view)
Application transparent partitioning?	Yes	Yes	No
Single copy of table definitions to maintain?	Yes	Yes	No
Potential Performance Degradation	Minimal	None	Yes
Partitioning Choices	Range or partition by growth	Range only	
Data Compression support?	Yes	Yes	
Hardware acceleration for compression ?	Yes	No	
Database Server Hardware scalability	High	Low-Medium (dependent on database server vendor)	
Maximum processor cores per server	80	80	

	DB2 with Data Sharing	SQL Server	
		Native Database Partitions	DPV
Maximum processor cores for all read/write nodes in a cluster	2,560	80	
Maximum number of active read/write nodes in the cluster	32	1	
Maximum concurrent connections per instance	20,000	32,767	
Theoretical maximum concurrent active connections per cluster	640,000	32,767 (Same figure as above, as clustering allows only one active read/write node at a time)	

Table 4 - Summary of Scalability Characteristics

SQL Server itself allows a reasonable degree of scalability for handling large data and/or transaction volumes within a single instance. However its actual scalability is severely limited by the underlying capabilities of the x86 platform when compared to a System z server. This gap is exacerbated massively when clustering or other multi-server solutions are considered.

In summary, SQL Server itself allows a reasonable degree of scalability for handling large data and/or transaction volumes within a single instance, even exceeding a single DB2 for z/OS subsystem in a few specific areas. However its actual scalability is severely limited by the underlying capabilities of the x86 platform when compared to a System z server, which is likely to be able to accommodate a significantly greater workload.

This gap is exacerbated massively when clustering or other multi-server solutions are considered, as SQL Server currently has very limited options for scaling beyond a single server for typical read/write workloads, and all of these involve significant compromises and restrictions within the application and database design. In contrast DB2 data sharing allows up to 32 System z servers to service a single logical database image with no application change required.

Performance

The usual approach for comparing performance between database vendors is to use standard industry benchmarks such as those defined by the Transaction Performance Council (TPC). For example, TPC-C and the newer TPC-E benchmarks model standard OLTP workloads, while TPC-H simulates an ad-hoc decision support workload. SAP provides a similar set of standard benchmarks for a number of their packaged application solutions. While these benchmarks seem to provide a useful comparison point, many question their applicability to more varied real-world situations and raise concerns about vendors specifically tailoring their products to optimise the benchmark results.

Whatever the reason, IBM does not publish TPC or SAP benchmarks for DB2 for z/OS, so no direct comparisons can be drawn on that basis. This section will therefore concentrate on some of the major factors impacting the performance of a database workload, and attempt to compare the relative strengths and weaknesses of each platform.

Hardware and Operating System

A key aspect of maintaining acceptable performance is the ability for a system to scale if transaction loads increase for an otherwise well-tuned application workload. This aspect has been covered in a previous section (please see Transaction/Workload Growth on page 24) and the combination of System z Parallel Sysplex and DB2 Data Sharing provides a clear advantage in this area, as none of the various SQL Server clustering options allow more than one server to share a typical OLTP load.

The System z platform has many performance benefits beyond the pure scalability aspect. The existence of speciality coprocessors allow many common tasks to be offloaded from the main central processors, freeing them for other workload and allowing the offloaded tasks to be more efficiently handled. The Intel Xeon servers used to support SQL Server do not have anything comparable

The System z platform has other performance benefits beyond the pure scalability aspect. The existence of speciality coprocessors[32] allow many common tasks to be offloaded from the main central processors, freeing them for other workload and allowing the offloaded tasks to be more efficiently handled. The Intel Xeon servers used to support SQL Server do not have anything comparable, and have to rely on the main CPUs and the I/O subsystem to perform all of the database processing.

System z platforms routinely support hundreds of diverse application workloads while maintaining high levels of performance. WLM is able to balance the resource requirements of each transaction within a workload in order to best meet the business performance objectives

Workload management is another long-standing System z strength, reflecting decades of ongoing investment by IBM. System z platforms routinely support hundreds of diverse application workloads while maintaining high levels of performance. z/OS Workload Manager (WLM) is able to balance the resource requirements of each transaction within a workload (in an individual System z server or across an entire sysplex) in order to best meet the business performance objectives, and DB2 fully supports and exploits this capability for its own workloads. This highly integrated approach is illustrated in Figure 15, below.

[32] These include standard processors installed on every System Z server such as the SAP (System Assist Processor, a dedicated I/O engine) and optional components such as the zIIP (System z Integrated Information Processor, used to offload certain workloads such as incoming distributed calls, network encryption and XML validation), the zAAP (System z Application Assist Processor, used for Java and XML workloads) ,and the IFL (Integrated Facility for Linux, a dedicated Linux processor).

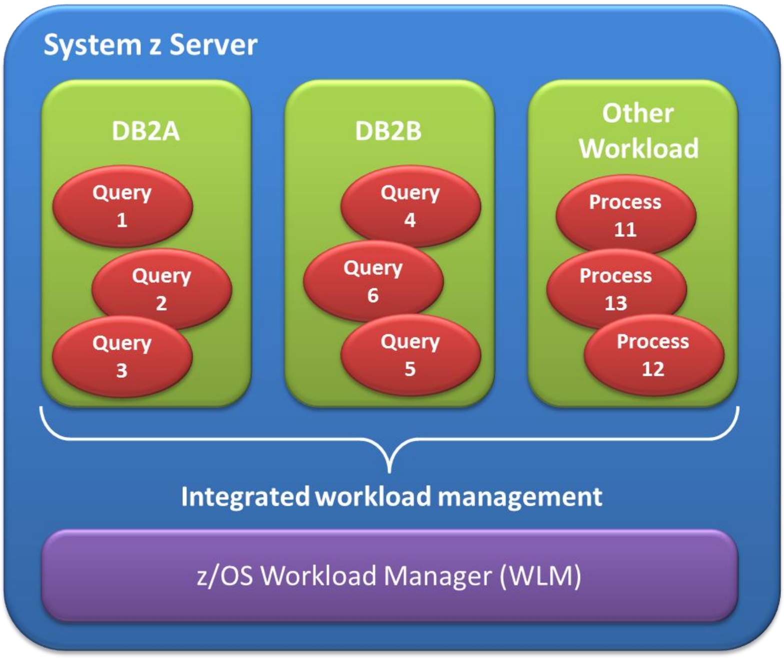


Figure 15 – z/OS Workload Management

WSRM allows basic workload policies to be defined, but does not interact directly with individual SQL Server processes. SQL Server has a "Resource Governor" available, but this allows prioritisation only within a single SQL server instance and does not interact with WSRM. It is therefore of limited value where multiple instances are deployed on a single server

Unfortunately, Windows Server does not have anything directly analogous to z/OS WLM. Windows System Resource Manager (WSRM) allows basic workload policies to be defined, but does not interact directly with individual SQL Server processes. SQL Server has had a "Resource Governor" available since the 2008 release, but this allows prioritisation only within a single SQL server instance and does not interact with WSRM. It is therefore of limited value where multiple instances are deployed on a single server as shown in Figure 16 below.

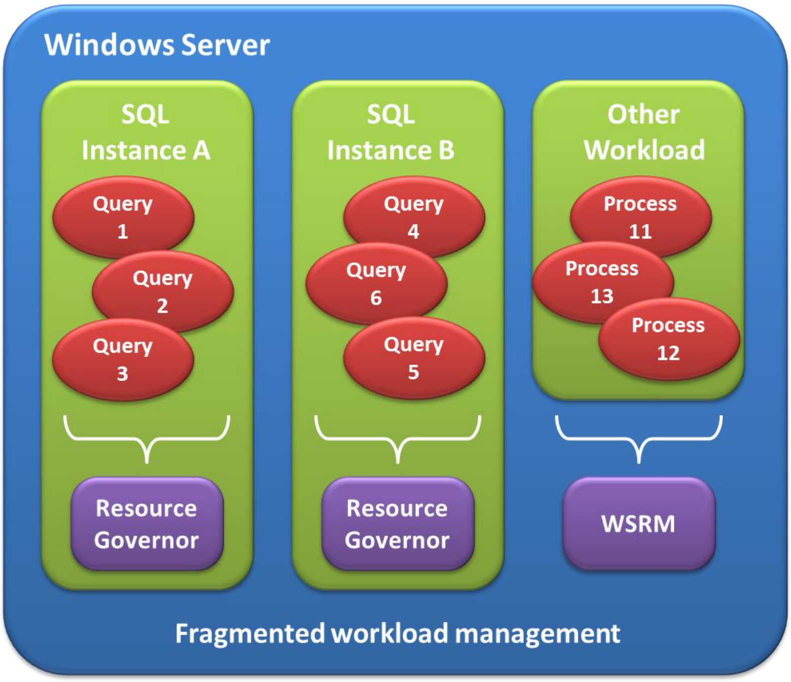


Figure 16 – Windows Server Workload Management

Furthermore, the SQL Server Resource Governor only allows workload to be limited based on the percentage of CPU and I/O resources it uses – there is no way of specifying the underlying business goals of each workload and allowing the system to manage the specific prioritisation as z/OS WLM does.

These differences in workload management capability are one of the major reasons for the huge differential between average utilisation rates in a System z environment (typically 90-95%) and a Windows Server environment (typically 15-20%). This can have a significant impact on the TCO for the respective platforms, in addition to ensuring that finite computing resources are directed to the processes that are most important to the business.

Differences in workload management capability are one of the major reasons for the huge differential between average utilisation rates in a System z environment and a Windows Server environment. This can have a significant impact on the TCO for the respective platforms, in addition to ensuring that finite computing resources are directed to the processes that are most important to the business

Database Software

Leaving aside the impact of the server hardware and operating system, the RDBMS itself can have a huge impact on the resources required for a given workload, and therefore the performance of that workload.

The optimiser is one of the most critical components of any RDBMS, and is responsible for selecting the most efficient access path for a given query. This may involve evaluating a large number of possible access strategies, especially in a BI/warehouse environment where there may be many indexes on a given table. The difference this decision makes can be very significant: it is not uncommon for a “good” access path to be several orders of magnitude quicker than a “bad” one for a given query.

IBM pioneered this cost based approach to access path optimisation in the mid 1970’s (5) and the DB2 product has since had the benefit of over 35 years of constant research and development to further develop its capabilities. In contrast, SQL Server’s current optimiser has its roots in the Cascades Framework and was introduced relatively recently in 1999. A detailed comparison of the relative capabilities of each product is beyond the scope of this paper, but the DB2 optimiser is generally accepted as being the most mature and sophisticated in the industry.

IBM pioneered this cost based approach to access path optimisation in the mid 1970’s and the DB2 product has since had the benefit of over 35 years of constant research and development to further develop its capabilities. In contrast, SQL Server’s current optimiser has its roots in the Cascades Framework and was introduced relatively recently in 1999.

Another related area is the way in which SQL statements are executed at run time. DB2 supports both “static” SQL (where the statement goes through a one-off “bind” process to create an access plan used for each execution) and “dynamic” SQL (where the optimisation process is conducted at run time each time the statement is executed). Each approach has its benefits for specific situations, but from a performance and security perspective static SQL is generally accepted as being the best overall approach for OLTP requirements. However, the extensive use of dynamic SQL within Java and packaged applications such as SAP has resulted in dynamic statement caching within DB2, which caches the access plan for a given SQL statement after it has first been prepared and therefore brings some of the performance benefits associated with static SQL.

SQL server does not support static SQL in the DB2 sense, but it does allow the access plan for dynamic SQL statements to be cached in much the same way as DB2’s dynamic statement cache.

Once an efficient access strategy has been selected, the speed with which the database engine can complete the necessary I/O operations is usually a major factor for all but the most simple SQL queries. Both DB2 and SQL Server use “pre-fetch” strategies to allow data to be fetched into memory before it is actually required, thereby reducing the query elapsed time.

Once data has been read from disk, significant performance benefits can be realised by caching the data in memory, thereby avoiding future I/O operations. Both SQL Server and DB2 use the term “buffer pool” for this area of memory, but once again DB2’s implementation is considerably better equipped for demanding enterprise workloads.

SQL Server has a single buffer pool, used to cache all data accessed by that instance. There is no way of segregating tables according to their usage characteristics. In contrast, DB2 supports up to 80 separate buffer pools, which can be sized and tuned separately from the others.

SQL Server has a single buffer pool, which is used to cache all data accessed by that instance. This means that there is no way of segregating tables according to their usage characteristics and it is therefore possible for commonly-used data pages to be persistently overwritten by others that are rarely re-referenced. In contrast, DB2 supports up to 80 separate buffer pools, which can be sized and tuned separately from the others. It is therefore possible to segregate objects to ensure that frequently referenced data remains in the pool and expensive I/O operations are avoided.

DB2 also supports multiple buffer pool page sizes, giving the database designer a choice of 4K, 8K, 16K or 32K pages[33]. This allows an optimal page size to be selected according to the size of each table row and the desired performance characteristics. SQL Server supports only 8K pages, so no such optimisation is possible.

Conclusions

The table below summarises the key differentiators covered within this section of the paper.

	DB2 for z/OS	SQL Server
Hardware / OS Scalability	High	Low
Specialised hardware co-processors for offloading system tasks	Yes	No
Workload management / prioritisation capabilities	High	Medium / Low
Optimiser maturity / sophistication	High	Medium
Support for true static SQL	Yes	No
Statement caching for dynamic SQL	Yes	Yes
Prefetch to minimise I/O delays	Yes	Yes
Multiple Buffer Pool Support	Yes (up to 80)	No
Multiple Buffer Pool Page Sizes	Yes (4K, 8K, 16K, 32K)	No (8K only)

Table 5 - Summary of Performance Characteristics

[33] In this context, a “page” is a logical subdivision of the data in a given table, and is the unit of I/O when data is read from disk to memory.

Even with a limited analysis it is clear that DB2 provides a more mature, flexible and capable environment for delivering high performance applications than SQL Server.

This section has reviewed just a few of the hundreds of factors that combine to determine the overall performance capabilities of a given RDBMS environment. However, even with this limited analysis it is clear that DB2 provides a more mature, flexible and capable environment for delivering high performance applications than SQL Server.

Data Governance

Security Certification

As with many other areas of this evaluation, direct comparison of SQL Server and DB2 for z/OS is difficult without descending into distracting detail on a feature-by-feature basis. One way of establishing a meaningful baseline at a higher level is to consider the third-party security certifications awarded to each database platform (which comprises of the hypervisor, operating system and RDBMS itself).

ISO 15408 defines the Common Criteria for Information Technology Security Evaluation (usually referred to as Common Criteria or CC) and is the generally-accepted international standard. Certification is awarded based on one of seven Evaluation Assurance Levels (EAL), with EAL7 being the most stringent. Current assurance levels are documented on the Common Criteria portal (6), and this is the source for all EAL references in this section.

IBM's PR/SM is the only hypervisor to achieve EAL5+ certification, with the equivalent Windows technology (Windows Server Hyper-V) awarded an EAL4+ rating

IBM's PR/SM is the only hypervisor to achieve EAL5+ certification, with the equivalent Windows technology (Windows Server Hyper-V) awarded an EAL4+ rating. Both operating systems (z/OS V1R12 and Windows Server 2008) have also been awarded EAL4+.

Although SQL Server 2012 has not yet been awarded any formal CC certification, SQL Server 2008 R2 achieved EAL4+ and the new version can be expected to at least equal this. Similarly, DB2 9 for z/OS is the most recently certified release, and that has also achieved EAL4+.

Both database platforms can boast a similar level of security certification, with the DB2 solution somewhat ahead due to the superior EAL5+ rating awarded to z/OS PR/SM.

Based on the certifications above, both database platforms can boast a similar level of security certification, with the DB2 solution somewhat ahead due to the superior EAL5+ rating awarded to z/OS PR/SM.

Reported Breaches

Useful though they may be for baseline comparison purposes, it is obvious that security certifications only tell part of the story. It is also important to understand the number of actual vulnerabilities and security breaches experienced in the real world.

The US National Institute of Standards and Technology (NIST) maintains a National Vulnerability Database (7), which records reported instances of software vulnerabilities that could lead (or have led) to a security breach, and tags each one with the associated software product(s).

The following analysis is based on information extracted from the National Vulnerability Database (NVD) for SQL server, DB2 and their associated operating systems. Please refer to Appendix B – NIST NVD Statistics on page 51 for a more detailed breakdown of the NVD statistics used.

Figure 17 shows the total reported software vulnerabilities by year for DB2 and SQL Server. Since collection of statistics began in 1992 SQL Server has had a total of 77 vulnerabilities reported, while DB2 for z/OS has had none.

Since collection of statistics began in 1992 SQL Server has had a total of 77 vulnerabilities reported, while DB2 for z/OS has had none

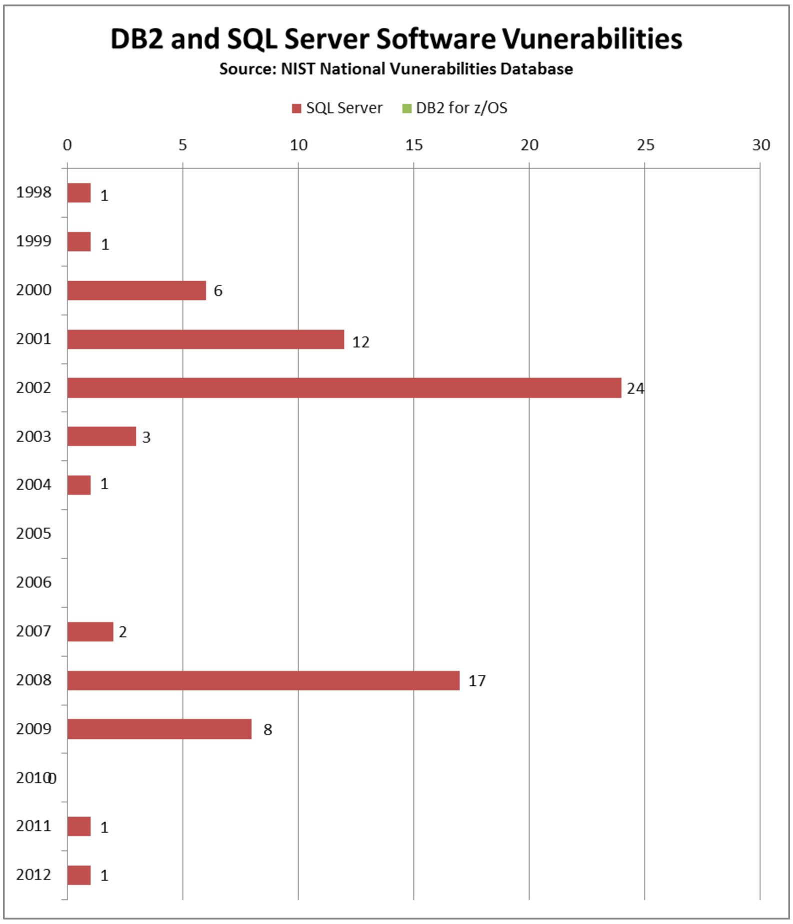


Figure 17 – Reported DB2 and SQL Server Software Vulnerabilities

SQL Server had a total of 831 reported vulnerabilities compared to just 14 for z/OS in the same period

The difference between the two database platforms becomes even more marked when considering the underlying Operating System – a critical attack vector as both database products commonly rely on OS authentication to determine database access privileges. As shown in Figure 18 below, SQL Server had a total of 831 reported vulnerabilities compared to just 14 for z/OS in the same period.

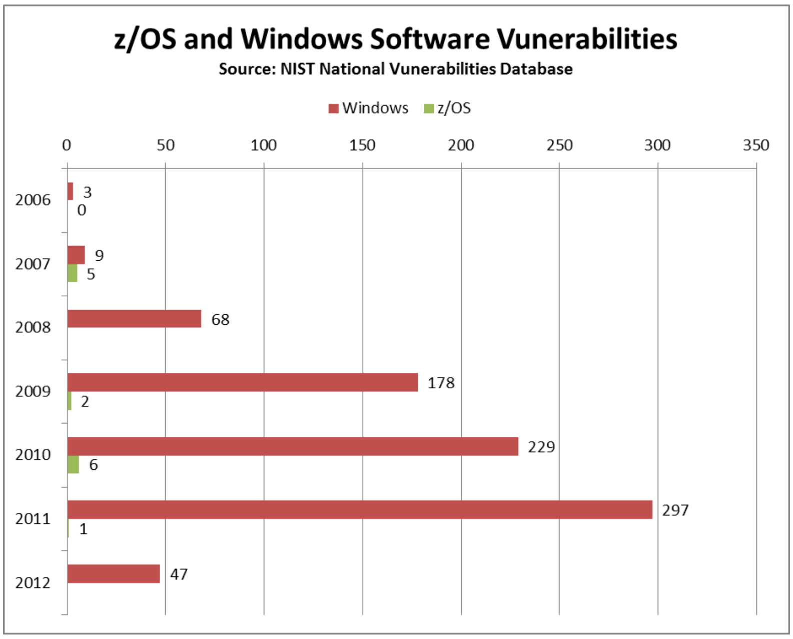


Figure 18 – Reported z/OS and Windows Server Software Vulnerabilities

It appears that from an historical perspective the combination of DB2 and z/OS is far less likely to suffer from a potentially harmful vulnerability than SQL Server on a Windows Server platform.

It must be acknowledged that NVD statistics cannot tell the whole story: some vulnerabilities will be undiscovered or unreported, or may be identified incorrectly in the database. However, even taking such considerations into account it appears that from an historical perspective the combination of DB2 and z/OS is far less likely to suffer from a potentially harmful vulnerability than SQL Server on a Windows Server platform.

Additional Governance Features

Most relational database systems implement a basic set of security mechanisms to secure data stored within them. Beyond these, each product offers additional features that intended to improve flexibility, reduce risk and increase productivity. This section highlights the most important of these features for each RDBMS being evaluated.

Role-Based Access

Enterprise data governance policies typically support the principles of *least privilege* and *separation of duties*.

Least privilege entails giving a user or process the minimum authority needed to perform their function, and from a database perspective it requires authorities to be sufficiently granular to allow each job role (developer, DBA, system administrator, etc.) to access only the minimum necessary database resources. Multiple low-level authorities typically need to be combined for a given job role, and this can increase administration overhead and lead to inconsistent authorities unless strictly controlled. Database roles are designed to address these problems, by bundling together a customised set of lower-level authorities for a specific requirement.

Roles also help to support separation of duties, which is a principle aimed at reducing fraud and/or errors through the implementation of appropriate checks and balances. A typical example in a database context would be to separate the security administration role from that of the production DBA. Removing the DBA’s ability to grant himself access to sensitive production data makes fraud or accidental data corruption less likely.

Both SQL Server and DB2 for z/OS support these concepts. DB2 10 for z/OS introduced a new set of more granular system authorities specifically aimed at least privilege and separation of duties, and these can be further customised using roles if required. Similarly, SQL Server introduced user-defined database role support in the 2008 version, and this has been expanded to support system roles in SQL Server 2012.

However, DB2’s implementation of roles goes beyond these concepts and introduces the possibility of further increasing “defence in depth” by interacting with network trusted contexts[34] and providing a location dimension to authentication.

As shown in Figure 19 below, a user would normally identify himself to DB2 via a userid/password combination and these would be checked in RACF (or other security product) to ensure they were valid. Once authenticated, the DB2 Catalog (or RACF if external DB2 security is being used) would be used to determine what DB2

[34] A network trusted context is a relationship between a trusted entity on the network (typically an application server inside an organisation’s firewall) and DB2.

privileges the user should be allowed. A similar overall process is also used by SQL server.

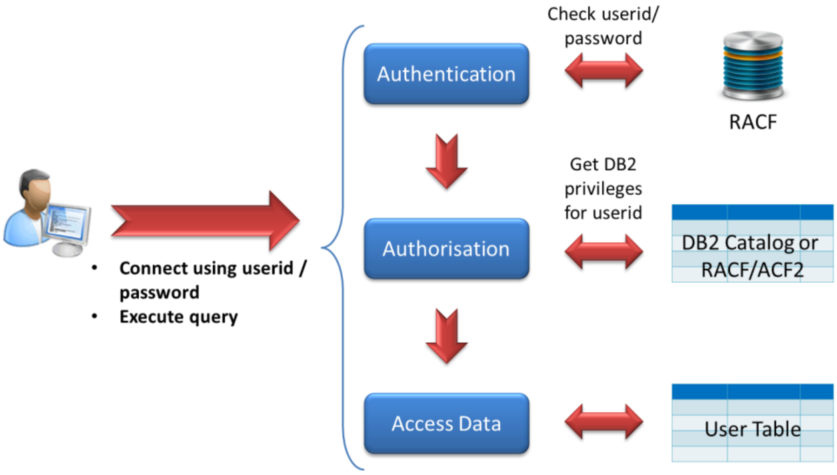


Figure 19 – Normal DB2 Authentication/Authorisation Process

A combination of network trusted contexts and roles provides additional capabilities for limiting DB2 access based on more than just a userid and password

A combination of network trusted contexts and roles provides additional capabilities for limiting access based on more than just a userid and password. As shown in Figure 20 below, if the attributes for an incoming connection (e.g. the TCP/IP address) match those of a pre-defined trusted context on the DB2 server, a default role can be assigned to the connection and used to determine the DB2 privilege set.

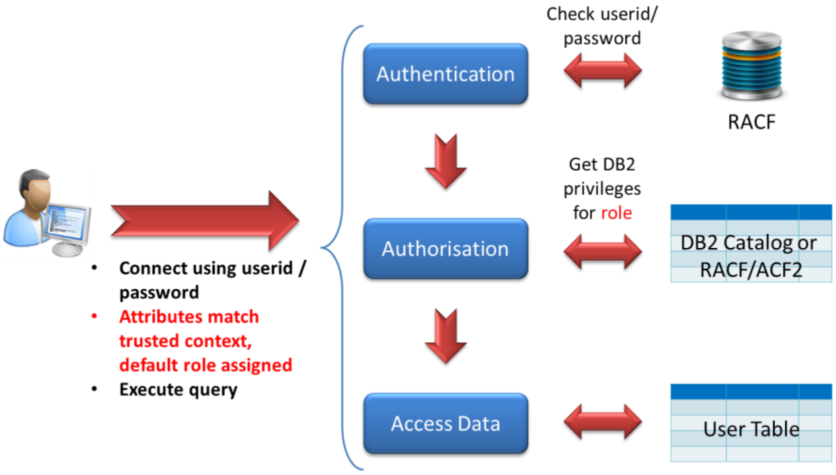

Connect using userid / password
Attributes match trusted context, default role assigned
Execute query
Authentication
Authorisation
Access Data
Check userid/ password
Get DB2 privileges for role
RACF
DB2 Catalog or RACF/ACF2
User Table

Figure 20 –DB2 Authentication/Authorisation Process with Trusted Context

By granting authorities only to the role and not the specific user, DB2 privileges become location-specific. This capability provides an additional dimension to the traditional authentication/authorisation process, increasing the depth of DB2's defence against malicious intent.

By granting authorities only to the role and not the specific user, DB2 privileges become location-specific. In the example above, the user connecting from a TCP/IP address associated with the trusted context would have all of the privileges associated with the role, whereas the same user connecting from elsewhere would have no access. This capability provides an additional dimension to the traditional authentication/authorisation process, increasing the depth of DB2's defence against malicious intent.

Fine-Grained Access Control

Most standard RDBMS security mechanisms allow for data access privileges to be set at the overall table level (for example, a user may be given authority to SELECT and INSERT into a specific table). However for compliance purposes it is often necessary to use a more granular approach and be able to grant access at the individual row or column level. For example, a manager may require access to view sensitive salary information for all employees in his department without being able to access other employee’s details.

Traditionally such requirements were met by using table views, database triggers or only providing access via stored procedures or application code, but all of these approaches had significant drawbacks. More recently DB2 began to support Label Based Access Control (LBAC), based on capabilities within z/OS and RACF. This addressed the requirement for fine-grained access control implemented at the database layer, but proved to be complex to implement and administer and has not been widely adopted by clients.

SQL Server does not natively support LBAC. Microsoft does offer some advice on how to implement a similar approach (8) but this mainly relies on the use of table views and therefore suffers from the traditional drawbacks while also introducing all of the complexity issues mentioned above.

DB2 10 for z/OS introduced a new way of defining access at the row level, known as fine grained access control (FGAC). This allows the DBA to define and enable access policies using familiar SQL syntax, and provide most of the flexibility offered by LBAC while being much easier to implement and administer. SQL Server currently offers nothing comparable.

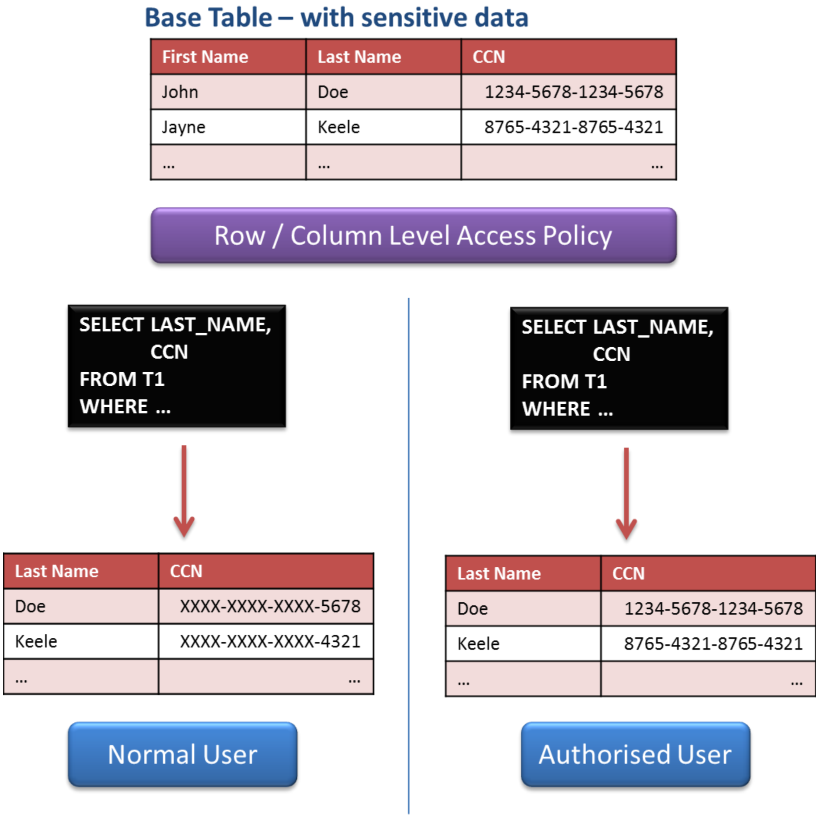
DB2 10 for z/OS introduced a new way of defining access at the row level, known as fine grained access control (FGAC). This allows the DBA to define and enable access policies using familiar SQL syntax, and provide most of the flexibility offered by LBAC while being much easier to implement and administer. SQL Server currently offers nothing comparable.

Data Masking

Another common compliance requirement is to selectively obscure or “mask” portions of a sensitive table column depending on the authority of the user requesting the data. A common example is a Credit Card Number (CCN) used within banking and online commerce sites. A full CCN typically consists of 16 digits, but for security reasons it is common for only the last four digits to be used for confirmation purposes.

DB2 10 for z/OS introduced the ability to create a mask on a given table as part of the FGAC capability mentioned above. As with the row-level table access policies, column masks are deeply integrated into the database layer and therefore cannot be potentially bypassed in the same way as views and application-based techniques can. In the example shown in Figure 21 below, a mask has been created on a sensitive table containing credit card details. All digits of a credit card number except the last four are masked with an “X” character for normal users, but authorised users are automatically allowed the see the full number.

No comparable facility exists within SQL Server.


Base Table – with sensitive data
Row / Column Level Access Policy
Normal User
Authorised User

First Name	Last Name	CCN
John	Doe	1234-5678-1234-5678
Jayne	Keele	8765-4321-8765-4321
…	…	…

```
SELECT LAST_NAME,
CCN
FROM T1
WHERE …
```

Last Name	CCN
Doe	XXXX-XXXX-XXXX-5678
Keele	XXXX-XXXX-XXXX-4321
…	…

```
SELECT LAST_NAME,
CCN
FROM T1
WHERE …
```

Last Name	CCN
Doe	1234-5678-1234-5678
Keele	8765-4321-8765-4321
…	…

Figure 21 –DB2 Data Masking

Audit

Strict regulatory compliance requirements are in place for the vast majority of organisations, especially within the financial services sector. From an IT perspective, these requirements will inevitably involve the production of strict audit trails to document access to sensitive data within the RDBMS, and activities of individuals with high access levels.

SQL Server 2012 extends the audit facilities previously introduced in the 2008 release and makes them available within all of the various packaging options (they were previously available only within the premium SKUs). Events can be audited at both the instance and database level, with audit data being written to the Windows Application Log, the Security Log or a dedicated file. A Microsoft paper (9) quotes an elapsed time overhead in the range of 6-35% for five typical OLTP workloads, compared to no audit being enabled.

DB2 10 for z/OS introduces a comprehensive new policy-based audit capability which is functionally similar to that within SQL Server. Policies can be defined for specific activities against specific objects, with the audit data being written to any standard DB2 trace destination (typically SMF). The DB2 10 Performance Topics Redbook (10) references a test based on the standard IRWW OLTP workload with and without audit enabled. The test showed a 4% elapsed time overhead when a full DML[35] audit policy was enabled.

[35] In DB2 terms this is an EXECUTE audit category, and records all SELECT, INSERT, UPDATE and DELETE activity against the relevant tables.

Conclusions

The table below summarises the key differentiators covered within this section of the paper.

	DB2 for z/OS	SQL Server
CC Certification Level – Hypervisor	EAL5+ (PR/SM on z196)	EAL4+ (Windows Server 2008 Hyper-V Role with HotFix KB950050)
CC Certification Level – Operating System	EAL4+ (z/OS V1R12)	EAL4+ (Windows Server 2008 R2)
CC Certification Level – Database	EAL4+ (DB2 for z/OS V9[36])	EAL4+ (SQL Server 2008 R2[37])
Total Reported NVD Vulnerabilities – Database	0	77
Total Reported NVD Vulnerabilities – Operating System	14	831
Role Based Access Control	Yes	Yes
Location-Sensitive Authorities	Yes[38]	No
Integrated Support for Label Based Access Control (LBAC)	Yes	No[39]
Integrated Support for Data Masking	Yes	No
Audit Performance Overhead	Low	Medium

Table 6 - Summary of Data Governance Characteristics

The combination of DB2 and z/OS can boast both a higher overall Common Certification security rating and a dramatically lower level of reported vulnerabilities than SQL Server and Windows Server

The System z platform is generally regarded as the IT world's most secure server. Security violations are rare, and malware incidents are virtually unknown. The combination of DB2 and z/OS can boast both a higher overall Common Certification security rating and a dramatically lower level of reported vulnerabilities than SQL Server and Windows Server.

SQL Server has added some useful features in the last few releases that have improved its manageability and audit capabilities. However, it is still some way behind DB2's capabilities with regard to roles, network trusted contexts, fine-grained access control and data masking.

[36] Note that this certification is for DB2 9 for z/OS. No certification for DB2 10 for z/OS had been published at the time of writing this paper.

[37] Note that this certification is for SQL Server 2008 R2 SP1. No certification for SQL Server 2012 had been published at the time of writing this paper.

[38] Via the Network Trusted Contexts feature.

[39] Recommended Microsoft solution requires the use of views and therefore suffers from the traditional drawbacks.

Remote Access and Federation

In this section, we will briefly examine the abilities of each RDBMS to access data within remote database elsewhere in the enterprise, and in turn respond to any incoming data access requests. The remote database can either be homogenous (e.g. DB2 for z/OS accessing another DB2 for z/OS system) or heterogeneous (e.g. DB2 for z/OS accessing an Oracle system).

Remote Access Capabilities

DB2 uses an open protocol called Distributed Relational Database Architecture (DRDA) as the basis for incoming and outgoing communication with external databases (both homogeneous and heterogeneous)[40]. In DRDA terms, the component requesting remote access is known as the Application Requester, whereas the remote target environment is called the Application Server.

DB2 for z/OS is able to act as an Application Requester or an Application Server, connecting to any other DRDA-compliant Application Server/Requester on the network with no additional tooling. If the remote RDBMS does not have DRDA capability, an additional federation product such as InfoSphere Federation Server may be required.

SQL Server uses Microsoft’s proprietary OLEDB interface to allow remote access to and from SQL Server Instances, via the Linked Server feature. OLEDB was designed as a replacement for the older ODBC API. OLEDB calls the requesting system the Consumer, and the remote target the Provider. The Distributed Partitioned View feature shown in Figure 10 on page 23 uses these remote links to allow communication between SQL Server instances.

An SQL Server link can be defined to any remote database that has a supported OLEDB provider installed on the SQL Server. This includes DB2 for z/OS, DB2 for LUW, Oracle and most other commonly used RDBMS products.

Access to remote databases can be classified as shown in the table below. Note that there are many different terms to describe these capabilities, but the four capability levels used in the table are the terms used by DRDA.

	SQL Statements per Unit of Work	DBMSs per Unit of Work	DBMSs per SQL Statement
Remote Request	1	1	1
Remote Unit of Work	>1	1	1
Distributed Unit of Work	>1	>1	1

[40] Note that DB2 for z/OS previously implemented a proprietary protocol (known as “private protocol”) for communication between DB2 for z/OS systems. This has been deprecated by IBM, and is not considered further within this discussion.

	SQL Statements per Unit of Work	DBMSs per Unit of Work	DBMSs per SQL Statement
Distributed Request	>1	>1	>1

Table 7 – The Four Levels of Remote Database Access

DB2 for z/OS natively supports these capabilities up to and including Distributed Unit of Work. Distributed Request is also possible, but requires an additional product called InfoSphere Classic Federation Server for z/OS. SQL server's capabilities are dependent upon the features implemented within the OLEDB Provider driver for each RDBMS, but a fully-featured driver is capable of everything up to and including Distributed Request.

Location Transparency

Both SQL Server and DB2 require a remote table to be identified by one or more additional qualifiers to denote its location[41]. These additional qualifiers can be hard-coded into SQL statements, but this can cause issues if the data moves to a new location at some point in the future.

Location transparency refers to the ability to refer to a remote table within an SQL statement without having to explicitly specify its location. This can be accomplished in DB2 by creating an alias at the local DB2 which points to the remote table. All SQL executed against the DB2 alias is then redirected to the remote table, and if the remote table moves to a new location in the future all that is required is to drop and recreate the alias. SQL Server provides a similar construct called a synonym, which works in much the same way.

Conclusions

The table below summarises the key differentiators covered within this section of the paper.

	DB2 for z/OS	SQL Server
Enabling Protocol	DRDA	OLEDB
Protocol Status	Open (The Open Group)	Proprietary (Microsoft)
Remote Request Support	Yes	Yes
Remote Unit of Work Support	Yes	Yes
Distributed Unit of Work Support	Yes	Yes
Distributed Request Support	Yes, with additional product[42]	Yes, if supported by OLEDB provider[43]

[41] In DB2, a three-part name is used to denote the location, table qualifier/schema and table name. SQL Server requires a four-part name containing the server link name, database, schema and table name.

[42] This capability requires an additional product called InfoSphere Classic Federation Server for z/OS.

	DB2 for z/OS	SQL Server
Location Transparency	Yes, via aliases	Yes, via synonyms

Table 8 - Summary of Federation Characteristics

The remote access capabilities of SQL Server and DB2 are broadly similar, but highly dependent upon the additional tooling and available driver support. With the right environment in place, either product is capable of supporting typical enterprise database federation requirements.

In conclusion, the remote access capabilities of SQL Server and DB2 are broadly similar, but highly dependent upon the additional tooling and available driver support. With the right environment in place, either product is capable of supporting typical enterprise database federation requirements.

[43] This capability is dependent on the features supported by the OLEDB Provider.

Application Support

Databases serve no purpose without applications to use their services. This section examines the support offered by each RDBMS for both bespoke and packaged applications.

Bespoke Applications

Today's enterprises use a variety of different programming languages, depending on the specific needs of the application being developed. Despite its age COBOL remains a major factor for most large organisations, while Java has become the strategic language for many enterprises in recent years. These are probably the most popular enterprise programming languages in use today but a number of others are in common use for specific requirements, including C, C#, C++, PL/1, Ruby and Visual Basic.

DB2 has fully supported COBOL applications since it was first released nearly 30 years ago. Soon after Java became a significant enterprise programming language in the late 1990's, JDBC drivers were made available to allow Java applications to access DB2 data.

DB2 has fully supported COBOL applications since it was first released nearly 30 years ago. Soon after Java became a significant enterprise programming language in the late 1990's, JDBC drivers were made available to allow Java applications to access DB2 data. The current drivers[44] offer support for JDBC 4.1 as well as supporting static SQL via the open SQLJ API, and Java developers have the choice between type 2 (local) or type 4 (remote) connectivity. Support for both COBOL and Java has continued to be expanded in every new release, with additional features being added to the database engine and associated drivers to improve performance and developer productivity.

In addition to COBOL and Java, DB2 directly supports assembler, C, C++, Fortran, PL/I and REXX programs running locally in the System z environment. Distributed applications can also make use of the various APIs supplied by the DB2 Connect product, which include ODBC, OLEDB, ADO, ADO.NET, JDBC and SQLJ. Between them, these API's allow nearly any programming language to access DB2 for z/OS.

SQL Server support for COBOL and Java is less well established. Some third parties offer solutions that allow COBOL applications that have been ported to a Windows environment to access SQL Server, but porting is usually a major exercise. The JDBC Driver only supports Type 4 connections it lacks the maturity and the static SQL options available from the IBM solution.

SQL Server support for COBOL and Java is less well established. Some third parties offer solutions that allow COBOL applications that have been ported to a Windows environment to access SQL Server, but porting is usually a major exercise that involves at least some changes to the application code. Alternatively, federation products such as IBM's Infosphere Federation Server can allow existing System z COBOL applications to access SQL Server data but there may be significant performance issues to be considered.

Microsoft support for Java is a relatively recent phenomenon[45]. The newly released JDBC Driver 4.0 supports the JDBC 4.0 API, but only Type 4 connections are supported and it lacks the maturity and the static SQL options available from the IBM solution[46]. Support for secondary enterprise languages is good, due to the extensive use of OLEDB, ADO and ODBC APIs.

[44] Java drivers are delivered as part of the DB2 Connect product, which is required for all distributed access to DB2 for /OS data.

[45] For several years, Microsoft viewed Java as a major threat to the Windows .Net platform, and adopted its "embrace, extend, extinguish" strategy in an attempt to lessen its impact (18)

[46] SQLJ and pureQuery both offer the ability to execute SQL from Java applications statically, potentially improving performance, response time consistency and security.

Packaged Applications

Enterprises are becoming increasingly reliant on packaged applications to support core business processes, so it is important for any RDBMS to be able to boast solid support from the vendors of these applications. Solutions areas include Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), Supply Chain Management (SCM) and Financial Management (FM).

SAP, Oracle[47] and Microsoft dominate this market, with each offering solutions in one or more of the product categories above. Each of these will be discussed separately in the sections that follow.

SAP

SAP is arguably the market leader in packaged enterprise solutions, with offerings that span all of the critical business process categories. IBM and SAP have a partnership stretching back 40 years, and DB2 is SAP's preferred database platform, (the company even run their own internal ERP system using DB2 as the back-end database). Every recent release of DB2 contains numerous features specifically designed to improve the performance and manageability of SAP applications, and the close collaboration between SAP and IBM means that these new features are aggressively adopted in each new SAP code release.

IBM and SAP have a partnership stretching back 40 years, and DB2 is SAP's preferred database platform.

The system z platform also offers some important TCO benefits for large-scale SAP deployments. Integrated Information Processors (zIIPs) can offload between 40% and 80% of SAP database workloads, and other hardware engines enable 40%-75% compression of SAP data, reducing disk and tape storage capacity requirements and accelerating data movement processes. Both may significantly reduce costs. In addition, SAP Business Warehouse queries can take advantage of the IBM DB2 Analytics Accelerator (IDAA)[48] to significantly improve their price/performance (11).

SAP and Microsoft have a formal alliance, but this is much more recent than that in place with IBM so both products have had less time to evolve around each other.

SAP and Microsoft also have a formal alliance, but this is much more recent than that in place with IBM so both products have had less time to evolve around each other. The mission-critical nature of a typical SAP solution means that the resilience and scalability of the underlying RDBMS is critical. DB2's advantages in this area (see High Availability and Scalability on page 12) make it a much stronger contender for large-scale implementations than SQL Server.

Oracle

Oracle's unique position as vendor of a complete "stack" consisting of packaged business solutions, RDBMS and hardware place it in an interesting position with regard to partnerships with both IBM and Microsoft.

DB2 used to be the preferred RDBMS for PeopleSoft, but Oracle's acquisition of the company in 2005 rapidly put an end to that. However, DB2 for z/OS is still an officially supported platform for both PeopleSoft and Siebel, and many customers continue to run

[47] Oracle solutions include PeopleSoft, Siebel and JD Edwards in addition to the products originally developed in-house.

[48] An optional component based on IBM's Netezza technology that is able to act as an offload engine for certain analytics workloads, greatly reducing their elapsed time and cost.

their applications using DB2 for z/OS as the back-end RDBMS. Many of the enhancements added to DB2 to optimise SAP workloads also benefit PeopleSoft and Siebel, but Oracle's strategy is clearly to encourage such customers to move to their own RDBMS and/or hardware offerings. None of the other Oracle solutions are targeted at DB2 for z/OS.

SQL Server is certified for use with JD Edwards, PeopleSoft and Siebel applications, and is a strong contender for implementations targeted at the Windows platform. However the next generation of Oracle's applications, known as Fusion, are focused firmly on Oracle's own RDBMS as the company works towards its vision of offering its customers a total stack solution.

Microsoft

Microsoft is increasing its presence in the packaged application space. It is adopting the same approach as Oracle and only supporting its own RDBMS product as a back-end, so customers who wish to implement these solutions have no choice but to use SQL Server

Microsoft is increasing its presence in the packaged application space, but to date its solutions have been focused purely on CRM and ERP requirements.

Not surprisingly, Microsoft is adopting the same approach as Oracle and only supporting its own RDBMS product as a back-end so customers who wish to implement these solutions have no choice but to use SQL Server.

Conclusions

The table below summarises the key differentiators covered within this section of the paper.

	DB2 for z/OS	SQL Server
Support for Primary Enterprise Programming Languages (COBOL and Java)	High	Low/Medium
Support for Secondary Enterprise Programming Languages	High	High
SAP Solution Support	High	Medium
Oracle Solutions Support	Medium	Medium
Microsoft Solution Support	N/A	High

Table 9 - Summary of Application Support Characteristics

Both RDBMS products support a wide set of programming languages for in-house development, but DB2 support for COBOL and Java is both more mature and more extensive than that offered by SQL Server.

Both RDBMS products support a wide set of programming languages for in-house development, but DB2 support for COBOL and Java is both more mature and more extensive than that offered by SQL Server.

For packaged applications the situation is highly dependent upon the chosen package vendor, and the degree of resilience/scalability required. For a mission-critical SAP solution, DB2 is a natural choice due to the close IBM/SAP partnership and superior resilience offered by the System z platform. With the exception of PeopleSoft and Siebel, Oracle solutions cannot be hosted by DB2 for z/OS and while SQL Server is supported for the older products, Oracle's own RDBMS is the obvious strategic choice. Similarly, any enterprise adopting a Microsoft CRM or ERP solution has no choice but to adopt SQL Server as the back-end database.

Summary and Conclusions

Some form of data storage sits behind nearly every application, but the unique demands of today's high volume, mission critical enterprise applications place some very specific demands on the back-end database system.

Microsoft SQL Server has evolved significantly from its roots as a departmental shared database server. However, DB2 for z/OS was designed from the ground up as a scalable, efficient, highly available database platform for enterprise applications.

Microsoft SQL Server has evolved significantly from its roots as a departmental shared database server and its most recent 2012 release contains some significant enhancements, especially with regard to product's high availability characteristics. However, DB2 for z/OS has had considerably longer to mature, and was designed from the ground up as a scalable, efficient, highly available database platform for enterprise applications. It is also able to fully exploit the underlying strengths of the System z hardware platform, providing synergies that cannot be matched by Microsoft with its reliance on third-party hardware providers.

DB2 for z/OS remains the most attractive database platform for today's mission critical enterprise applications.

For these reasons, DB2 for z/OS remains the most attractive database platform for today's mission critical enterprise applications. Organisations that are considering porting such applications from DB2 for z/OS to SQL Server should carefully evaluate the compromises inherent in such a move.

Appendix A – Acknowledgements

The author would like to thank the following people for their invaluable contributions to this paper:

Jim Pickel	STSM, DB2 for z/OS Development
Barbara Thorne	DB2 for z/OS Release Team
Terrie Jacobi	Program Director, DB2 for z/OS
Chris Crone	Distinguished Engineer, DB2 for z/OS Development

Appendix B – NIST NVD Statistics

The following tables present the detailed NIST National Vulnerability Database statistics used to compile the charts in Reported Breaches on page 35 of this document. All statistics were current as at 9th May 2012, and obtained from the NIST NVD website (7).

Database Vulnerabilities

For SQL Server, three CPE (Common Platform Enumeration) tags were used as follows:

- cpe:/a:microsoft:sql_server
- cpe:/a:microsoft:sql_server_desktop_engine
- cpe:/a:microsoft:sql_server_express_edition

No explicit CPE tag is listed within the NVD for DB2 for z/OS, so the platform-neutral tag of cpe:/a:ibm:db2 was used with an additional search keyword of "z/OS". No reported vulnerabilities were found. Extensive manual checking was also conducted to ensure that no DB2 for z/OS vulnerabilities had been reported under other CPE tags, and again, none were found.

Year	NVD CPE Tag			Total
	cpe:/a:microsoft:sql_server	cpe:/a:microsoft:sql_server_desktop_engine	cpe:/a:microsoft:sql_server_express_edition	
1998	1			1
1999	1			1
2000	6			6
2001	12			12
2002	24			24
2003	3			3
2004	1			1
2005				0
2006				0
2007	2			2
2008	11	4	2	17
2009	8			8
2010				0
2011	1			1
2012	1			1

Table 10 – Detailed NVD Statistics for SQL Server

Operating System Vulnerabilities

For Windows Server, five CPE tags were used as follows:

- cpe:/o:microsoft:windows_server
- cpe:/o:microsoft:windows_server_2000
- cpe:/o:microsoft:windows_server_2003
- cpe:/o:microsoft:windows_server_2008
- cpe:/o:microsoft:windows_srv

Year	NVD CPE Tag					Total
	cpe:/o:microsoft:windows_server	cpe:/o:microsoft:windows_server_2000	cpe:/o:microsoft:windows_server_2003	cpe:/o:microsoft:windows_server_2008	cpe:/o:microsoft:windows_srv	
2006	1	1	1			3
2007	5		2	2		9
2008			34	34		68
2009	6		78	93	1	178
2010	1		100	128		229
2011			137	160		297
2012			20	27		47

Table 11 – Detailed NVD Statistics for Windows Server

For z/OS, two CPE tags were used as follows:

- cpe:/o:ibm:z%2Fos
- cpe:/o:ibm:zos

Year	NVD CPE Tag		Total
	cpe:/o:ibm:z%2Fos	cpe:/o:ibm:zos	
2006			0
2007		5	5
2008			0
2009	2		2
2010	2	4	6
2011	1		1
2012			0

Table 12 – Detailed NVD Statistics for z/OS

Bibliography

1. **Ponemon Institute.** *Calculating the Cost of Data Center Outages.* s.l. : Ponemon Institute LLC, 2011.

2. **GlobalCollect.** GlobalCollect Sees Increase in US Transaction Volume of 96.7% on Thanksgiving. *GlobalCollect Corporate Website.* [Online] 1 December 2011. http://www.globalcollect.com/online-payments/news/2011/online-transaction-volume-Thanksgiving-holiday/.

3. **Ponemon Institute.** *2010 Annual Study: Global Cost of a Data Breach.* s.l. : Ponemon Institute LLC, 2011.

4. **IBM.** IBM z/OS V1.12 - Smart technologies for a new dimension of computing. *IBM Website.* [Online] 22 July 2010. http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=ca&infotype=an&supplier=897&letternum=ENUS210-235.

5. *Access Path Selection in a Relational Database Management System.* **al, Pat Selinger et.** s.l. : SIGMOD, 1979.

6. **CCRA.** Certified Products. *The Common Criteria Portal.* [Online] http://www.commoncriteriaportal.org.

7. **National Institute of Standards and Technology.** Vunerabilities. *National Vunerability Database.* [Online] http://web.nvd.nist.gov/view/vuln/search.

8. **Rask, Art.** *Implementing Row- and Cell-Level Security in Classified Databases.* s.l. : Microsoft, 2005 (Updated in 2011).

9. **Lee, Il-Sung and Rask, Art .** Auditing in SQL Server 2008. *Microsoft Website.* [Online] February 2009. http://msdn.microsoft.com/en-us/library/dd392015%28v=sql.100%29.aspx.

10. **IBM.** *DB2 10 for z/OS Performance Topics.* 2011. SG24-7942-00.

11. **Joachim Rese, Georg Maye.** *Rapid SAP NetWeaver BW ad-hoc Reporting Supported by IBM DB2 Analytics Accelerator for z/OS.* s.l. : SAP AG, 2011.

12. **Eckerson, Wayne.** *TDWI Checklist Report: Data Federation.* s.l. : The Data Warehousing Institute, 2009.

13. **Clabby Analytics.** *IBM's New z114: A Replacement for SPARC/Itanium Servers?* s.l. : Clabby Analytics, 2011.

14. **Microsoft.** Windows Server Blog. *Microsoft Website.* [Online] 2 April 2010. http://blogs.technet.com/b/windowsserver/archive/2010/04/02/windows-server-2008-r2-to-phase-out-itanium.aspx.

15. **Cox, Kevin.** *What are the largest SQL projects in the world?* [Powerpoint Presentation] 2010.

16. **IBM.** *DB2 10 for z/OS Performance Topics.* s.l. : IBM Redbooks, 2011. SG24-7942-00.

17. —. zEnterprise – The Ideal Platform for Smarter Computing. *IBM Website.* [Online] http://www.ibm.com/ibm/files/G517149O36051A34/02.pdf.

18. **Wikipedia.** Embrace, extend and extinguish. *Wikipedia.* [Online] http://en.wikipedia.org/wiki/Embrace,_extend_and_extinguish.