White Paper
October 2009

Transparent Application Scaling with IBM DB2 pureScale

Information Management

DB2 software

Contents

Introduction

In the middle of a recovering economy, instant access to core business data has become a critical component to success, and in some cases survival. As more and more dollars make their way into the economy, businesses need to be nimble, with highly available and adaptable infrastructures, so that they can grab opportunities for renewed growth.

The marketing veneer of most distributed software companies is to associate availability levels with terms such as *"mainframe-like"* or *"5 – 9s"* availability. These key phrases try to convey the continuous availability set by the industry deemed "gold" standard for high availability: DB2® for z/OS®.

Availability	Downtime per Year
99.999%	5 minutes
99.99%	50 minutes
99.9%	8 hours, 20 minutes
99%	3 days, 11 hours, 18 minutes
95%	18 days, 6 hours
90%	34 days, 17 hours, 17 minutes
85%	54 days, 18 hours

Nowadays, availability means more than just surviving component failures and resuming normal transaction processing. If your service level agreement (SLA) dictates that expected query response times should be in seconds and the server is returning queries in a minute; it is an availability issue. To be available, your systems not only need to service transactions, they need to service them within the time period defined in your SLA.

For example, if seasonal fluctuations in the business cycle cause availability issues from a scaling perspective, a truly available architecture needs to *transparently* add resource *without application changes* to meet changing performance requirements. The word *transparent* is key: when adding capacity, applications should not need to be cluster aware (the application is aware of what data is on what node to avoid contention between nodes). Businesses can not afford to invest in building these complex applications for decent scaling. Why? First, the obvious: cluster-aware applications have to change as your volume of data and distribution changes. Cluster-aware applications do not just require code changes as the cluster grows: they need to be tested, go through the Quality Assurance (Q/A) process, be deployed, certified, and so on. This can cause weeks of coordination efforts across the enterprise and inevitably drains the infrastructure from resources that can be better used elsewhere.

Other offerings for transactional scale-out databases on distributed platforms (non-mainframe) are characterized by outdated architectures that add unwanted impediments (such as increased overhead) to scaling which can lead to SLA violations.

The *IBM DB2 pureScale* technology (herein referred to as DB2 pureScale) satisfies your current and future business needs for continuous availability with a system that combines **high availability** with true **transparent application scaling**. The DB2 pureScale architecture is further bolstered with an integrated exploitation of the IBM® Power™ Systems servers and IBM storage solutions to deliver on this high-value solution.

Up until now, *mainframe-like* was a marketing catch phrase. DB2 pureScale marks the first time that a true transparent scaling architecture is available for distributed platforms. This paper introduces to you the DB2 pureScale technology, what it looks like, where it comes from, and how it provides unique advantages from the high availability and transparent application scaling perspectives.

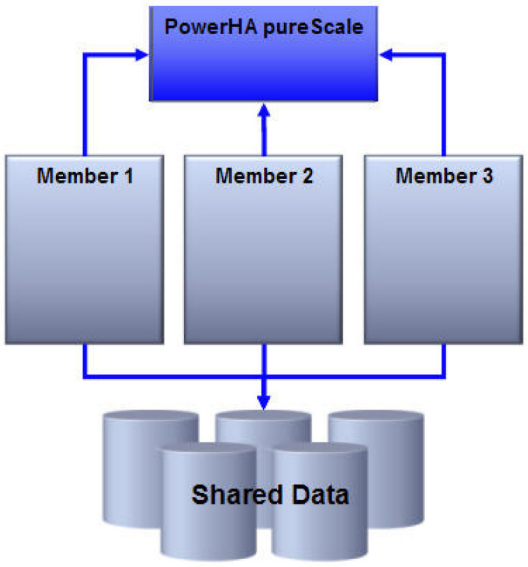
What does DB2 pureScale look like?

DB2 pureScale is a new optional DB2 feature that allows you to scale out your database on a set of servers in an "active-active" configuration delivering high levels of both availability and scalability. In this configuration the copy of DB2 running on each host (or server) can simultaneously access the same data for both read and write.

A collection of one or more DB2 servers that share DB2 data is called a *data sharing group*. A DB2 server that belongs to a data sharing group is a member of that group. All members of a data sharing group share the same database. Currently, the maximum number of members in a data sharing group is 128.

In addition to the DB2 members, there is also the PowerHA pureScale™ component which provides centralized lock management as well as a centralized global cache for data pages (known as the group buffer pool).

Each member in the data sharing group can interact directly with the PowerHA pureScale component, as shown below, through a very efficient InfiniBand™ network which means each member has point-to-point connectivity to the centralized locking and caching facility.

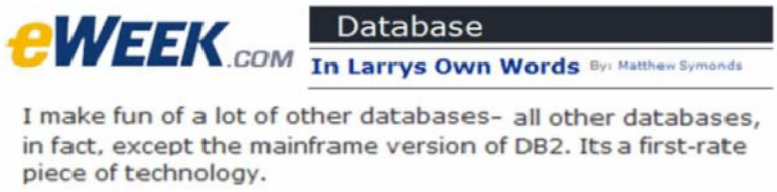


Where Does DB2 pureScale Come From?

When you hear or read about references to *mainframe-class availability*, they are referring to the gold standard of availability set by DB2 for z/OS. In fact, there is no database solution in the world that can match the availability characteristics of a System z® server running DB2 for z/OS.

The technology behind the DB2 for z/OS data sharing implementation allows the servers using it to continuously deliver on SLAs due to the Coupling Facility with centralized locking and global cache coherency which enable quick recover from failure. In fact, DB2 for z/OS offers true 5-9s availability and is well known for its ability to seamlessly scale workloads linearly.

When you read or hear about DB2 for z/OS, you probably think about massive scalability and ultra high availability. This reputation is not marketing fluff – it is derived from those systems consistently leading the industry for having database workloads available for a very long time. Perhaps the best tribute to the power of DB2 for z/OS is from Oracle's co-founder and CEO Larry Ellison who notes[1]:


eWEEK.COM Database
In Larrys Own Words By: Matthew Symonds
I make fun of a lot of other databases- all other databases, in fact, except the mainframe version of DB2. Its a first-rate piece of technology.

What is so special about DB2 for z/OS that lead Mr. Ellison to make this kind of statement? Users of DB2 for z/OS are very familiar with its data sharing 'secret sauce': the *Coupling Facility*. The Coupling Facility allows DB2 for z/OS to scale-out linearly, provides a centralized facility to manage locks, acts as a global shared buffer pool for dirty pages (which can assist with both scalability and recoverability operations), and more.

DB2 pureScale technology has direct lineage from the DB2 for z/OS Coupling Facility and therefore accrues many of the benefits which have lead DB2 for z/OS to have the 'gold' standard label for availability and scalability. How so? DB2 pureScale comes with the IBM powerHA pureScale component which delivers the same centralized locking and true global shared buffer pool architecture.

Other vendors have implemented shared-disk architected databases, most notably Oracle Real Application Clusters (Oracle RAC). However, at the time Oracle RAC was developed, distributed platform technology did not allow for efficient access to a *centralized* shared cache. As a result, the Oracle RAC design is an attempt to emulate the technology found in DB2 for z/OS; this resulted in Oracle RAC's *distributed* lock management technology and *distributed* caching architecture. The Oracle RAC architecture missed the succinct value of the DB2 for z/OS solution when it introduced its scale-out shared-disk architecture. On the other hand, both DB2 for z/OS and DB2 pureScale deliver the same *centralized* resource management which solves these scalability and availability complexities which we explain later in this paper.

The bottom line is that there is only one architecture on the market that delivers true transparent application scalability and ultra high availability. With modern hardware interconnects available on distributed platforms and the ability to deliver deep exploitation of interrupt free Remote Direct Memory Access (RDMA) over InfiniBand, it is now finally possible to leverage the same centralized locking and buffer caching algorithms found in DB2 for z/OS. DB2 pureScale is an evolution in the IBM family that extends the reach of this industry-proven technology to distributed platforms.

DB2 pureScale For Transparent Application Scalability

The key to real cost savings in a scale-out database environment is the delivery of true transparent application scaling. Transparent scaling means that the database engine can deliver increased throughput and efficient response times for OLTP applications without requiring *locality* of data.

Locality of data means that the data that an application needs is on the server that the application is connected to and that there is little contention between nodes for the same page of data. Locality of data is essential for scale-out architectures that have a heavy network-based messaging infrastructure for sharing data in a cluster.

Scale-out architectures that rely on locality for effective scaling require developers building sophisticated transactional applications make their applications *cluster aware*. Cluster-aware applications are more complex and costly to develop and deploy, and also require application rework when the cluster changes. Some vendors might claim that their architectures can run any application without modification; however, they don't always ***scale*** any application without being designed with some form of cluster awareness.

Transparent application scaling means that applications do not have to be cluster aware in order to take advantage of the scale-out architecture. DB2 pureScale is unique on distributed platforms and derives its efficiency from the exploitation of modern day network and hardware architectures, and the pureScale centralized locking and caching.

In order to reduce communication between nodes in the cluster for lock management and global caching services, DB2 pureScale uses the powerHA pureScale cluster acceleration facility (here-in referred to simply as the *CF*) along with RDMA technology to deliver transparent application scalability.

RDMA allows each member in the cluster to directly access memory in the CF and for the CF to directly access the memory of each member. For example, assume that a member in a cluster (Member 1) wants to read a data page that is not in its local buffer pool. DB2 assigns an agent (or thread) to perform this transaction; the agent then uses RDMA to directly write into the memory of the CF to indicate that it has interest in a given page. If the page that Member 1 wants to read is already in the CF's global centralized buffer pool, the CF will push that page directly into the memory of Member 1 instead of having the agent on that member perform the I/O operation to read it from disk. The use of RDMA allows Member 1's agent to simply make a memcopy (memory copy) call to a remote server without the need for costly process-to-process communication calls, processor interrupts, IP stack calls, and so on. Quite simply, pureScale allows a member's agent to perform what appears to be a local memory copy operation when in fact the target is the memory address of a remote machine.

These lightweight remote memory calls, along with a centralized buffer pool and lock management facilities, means that an application does not have to connect to the member where the data already resides. It is just as efficient for any member in the cluster to receive a data page from the global buffer pool regardless of the size of the cluster. Most RDMA calls are so fast that the DB2 process making the call does not need to yield the CPU while waiting for the response from the CF and does not have to be rescheduled to complete the task. For example, to notify the CF that a row is about to be updated (and therefore an X lock is required) a member's agent performs a Set Lock State (SLS) request by writing the lock information directly into memory on the CF. The CF confirms that there are no other members in the cluster that already have this row X locked and will directly write into the requesting member's memory to grant the lock. The entire round trip for this SLS can take as little as 15 microseconds and therefore the agent does not need to yield the CPU. The agent can continue to be productive rather than waiting on an IP interrupt (avoiding unnecessary context switches) as is the case with other scale-out architectures. If for a specific operation, such as long running batch transactions, it would make more sense for the DB2 agent to yield the CPU, DB2 will make an autonomic decision to dynamically yield the CPU.

Another important DB2 scalability feature that goes hand-in-hand with transparent application scaling includes the DB2 pureScale built-in load balancing across the members of a cluster. Applications do not need to be cluster aware to take advantage of the load balancing. The same client-side drivers that DB2 for z/OS data sharing customers use today work with DB2 pureScale for cluster load balancing.

DB2 pureScale For Availability

A scale-out architecture is not solely reserved for capacity increases. This type of architecture delivers improved availability by creating systems that can continue to process transactions in the event of a component failure.

DB2 pureScale takes availability to a new level when compared to other offerings available on distributed platforms. DB2 pureScale provides full access to every page of data that does not need recovery and is aware at all times of which specific pages need recovery without having to perform a single I/O operation. This is yet another important innovation made possible through the unique capabilities of a centralized CF.

Every time that a member reads a page into its buffer pool, the CF becomes aware of it and keeps track of that fact. Anytime a member wants to update a row on a page, the CF is aware of that as well. Whenever an application commits a transaction, dirty pages are written directly into the CF's memory by that member. This process allows any other member in the cluster that want to read this changed page to get the updates directly from the CF's. More importantly, from a recovery perspective, if any member fails the CF has a list of pages that the failed member was in the process of updating as well as the pages that were updated and committed by the failed member but were not yet written to disk.

The recovery process for *any* relational database management system (RDBMS) involves first redoing any transactions that were committed to ensure the pages on disk for those transactions are up-to-date on disk (this process is known as *redo recovery*). In addition, any database server must also undo any in-flight transactions which made changes to data that were flushed to disk but had not yet been committed prior to the failure (this process is known as *undo recovery*).

In a shared disk cluster, it is critical that no other node in the cluster reads or updates any pages from disk that might not have been recovered as yet (the recovery of those pages must take place before any new transactions can be performed on those rows). Here is where the CF really shines: because the CF knows which pages were in the process of being updated by the failed node, and the CF already has the dirty committed pages from that node in its centralized buffer pool, DB2 pureScale does not need to block other members from continuing to process transactions while it determines what pages need recovery. Other architectures require what may be significant processing time to determine what must be recovered due to their distribution of locking information (more on that subject later).

This process of recovery in DB2 pureScale environments is easy to explain at a high-level. Each member has processes that are sitting idle, but ready in the event of a failure. Should a member fail, one of these recovery processes is activated; since these processes already exist, there is no need for the operating system to waste valuable system time to create a process, allocate memory to it, and so on . This recovery process instantly begins to prefetch dirty pages from the CF into its own local buffer pool. The vast majority of recovery will require no I/O operations because the pages that need recovery are already in the CF's centralized buffer pool. Furthermore, this page prefetching is using lightweight RDMA for very fast and efficient transfer between the CF and the recovery member. During this time, all other applications on all other members continue to process requests. If they need any data from any page that does not need recovery, they can continue to

perform their transactions. As well, they can continue to read pages from disk because the CF already knows exactly which pages on disk are clean and which need recovery. The recovery process then reads the failed member's log file in order to replay the necessary transactions to redo and undo the updates made by the failed member.

For typical transactional workloads, the time from the member failure until the time the pages that were being updated in-flight on the failed node are available to another transaction is typically 20 seconds or less. Note that this also includes the failure detection times which some vendors may exclude when referring to recovery times. All other pages in the database are fully available a***t all times even after a member fails***.

In addition, components in the system like the PowerHA pureScale cluster accelerator are redundant. DB2 pureScale allows for duplexing of the CF capability such that locking and shared cache information is stored in two separate locations in the event the primary CF fails.

Summary

By leveraging modern hardware architectures, DB2 pureScale is able to bring to the distributed platform the centralized locking and caching capabilities previously only available on DB2 for z/OS. This hardware and network exploitation allows for greater levels of concurrency and significantly reduced overhead which in turn delivers higher levels of scalability. In addition, the centralized locking and page caching allows DB2 pureScale to continuously be in a state of awareness as to what pages would need recovery should any member fail. Thus, in the event of a failure, all data that does not need recovery is continuously available to other applications while the page that were in process of being updated by the failed node are already known to the system and are recovered more quickly.

For applications that need high levels of availability and for which horizontal growth delivers a cost benefit, DB2 pureScale brings to the table a solution that is tailored to meet these needs and has the lineage which has already proven itself in the marketplace.

For more information

To learn more about how IBM DB2 is lowering the cost of managing data, contact your IBM representative or visit ibm.com/db2

IBM Canada Ltd.
8200 Warden Avenue
Markham, ON, Canada L6G 1C7

Produced in Canada
October 2009

Endnotes

1 http://www.eweek.com/c/a/Database/In-Larrys-Own-Words/2/

IMW14253-USEN-01