IBM INTEGRATED ANALYTICS SYSTEM (IIAS)

Next-Generation Netezza (aka Sailfish) Strategies and Opportunities

Prepared by
David Birmingham, Principal Solutions Architect
Gary Shaw, Senior Solutions Architect
Frank Pantaleo, Senior Solutions Specialist
Sirius Computer Solutions, Inc.

November, 2017

SIRIUS

Contents

Executive Summary

IBM has integrated Netezza technology with the IBM® Db2® engine to provide a common, consistent SQL interface for both Netezza and all Db2-centric database offerings, called IBM Integrated Analytics System (IIAS).

The IIAS offering solves a number of problems inherent to both the Mustang- and IBM BladeCenter® server-based Netezza machines (TwinFin, Striper and Mako), while also innovating forward to make the system easier to administer.

While the Common SQL Engine applies to the whole Db2 family, this paper is in context of data warehousing and how the Netezza Technology has been integrated to the Db2 family for analytics and warehousing. As dashDB® has been rolled-up by IBM as Db2 Warehouse, both local and cloud.

The Netezza platform derives its power from massively parallel hardware architecture. This paper will show how this power is harnessed in IIAS to dovetail with Netezza's prior instances.

Netezza: A Short History

Netezza entered the marketplace as an "appliance," a self-contained analytics platform with unique capabilities to rapidly load, process and query very large scales of information. Netezza eclipsed all other platforms in data load and query turnaround speed. Users commonly moved to Netezza to load, host and query billions of rows, and it was not unusual to encounter tables with hundreds of billions of rows. Nothing on the market could match its power and simplicity of operation.

Based on the notion of "where not to look," Netezza inverted the common method of data search and retrieval. In other engines, indexes assist data search. In Netezza, no indexes exist, but when searching tens of billions of rows, at some point the query spends more time searching the index than retrieving data. By using a "where not to look" model, the query could be directed to smaller parts of a table without scanning the entire contents. This is the only truly scalable means of searching such large data sets.

Netezza also shifted the paradigm of query tuning. In other database engines, query tuning is the primary means of boosting a query's performance. In Netezza, the data is laid to disk in the highest-performance configuration to support the queries, and then the queries are configured to leverage this layout. This synergy means common query tuning is no longer effective; if the data is laid out improperly, no query tuning will help, and if the data is laid out properly, no query tuning is necessary (different from query "debugging" where the data may be laid out properly but the query isn't using it). Query tuning in Netezza has been likened to using a steering wheel to make a car go faster. Queries are logical, but ultimately the power is physical.

Until mid-2009, Netezza Corporation used custom-manufactured hardware. Each disk and CPU was mounted on the same hot-swappable hardware assembly (called an SPU, or Snippet Processing Unit). This also carried a field-programmable gate array (FPGA) that provided extremely high-speed data filtration in hardware rather than software. Users could also expand the number of data slices (CPU/disk/FPGA combinations) by simply adding a hardware frame and rebalancing data (horizontally elastic).

On original inception, the open-source Postgres engine was "gutted" and its SQL parsing mechanism was preserved and normalized to ANSI-99 standard SQL. However, this still had some minor leftover aspects of Postgres syntax. The Netezza machine (through Mako) had no official "engine." Netezza used a host operating system (the NPS Host) which managed the massively parallel processing (MPP) hardware. A user would interoperate with the host via SQL or configuration files, but it carried no capacity associated with an engine, such as data exceptions, triggers, referential enforcement, etc. This was beneficial and simplified the user experience. There was no need for indexes, and in data warehousing, referential integrity checks are always turned off when loading data.

In 2009, the IBM BladeCenter blade servers were introduced, with the FPGA as a bolt-on card to the blade, and was dubbed TwinFin. It hosted the CPUs and disk drives separately, linking them at startup time. The CPUs still enjoyed a dedicated drive, but were not hardware-bound. The BladeCenter servers, however, did not support horizontal elasticity. Upgrading from one rack to two racks, for example, meant dropping the two-rack alongside the one-rack, and copying data. But the BladeCenter servers solved other problems, such as decoupling the disk drives from the CPU hardware, making them less expensive to replace.

A key feature of Netezza, from the Mustangs through the BladeCenter servers, was the use of zone maps to shorten query times. Zone maps allow us to bundle data into more easily discovered locations on physical disk. Optimizing zone maps was an inefficient process, usually requiring a table rebuild to fully recover query power over time. IBM later introduced “Organize On” to assist in optimizing data for zone maps, effectively simplifying the process to a few keystrokes.

In 2012, IBM purchased Netezza Corporation and re-introduced the machines as PureData® System for Analytics, powered by Netezza technology (IPDA or PDA).

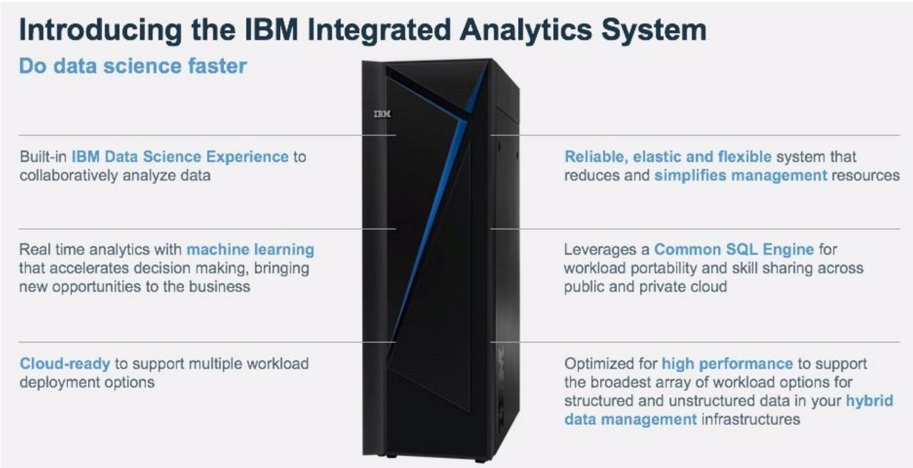
IBM preserved the commitment to the “Enzee Community,” a loyal ecosystem of users Netezza had nurtured over its existence. It also maintained the connectivity between users through the IBM Netezza/Enzee Community, and continued to host the annual Enzee Universe at its own annual conference.

The blade server form had three major releases: TwinFin, Striper and Mako. Each was faster than its predecessor, with better hardware, faster drives, etc. Along with these offerings, software libraries (add-ons) were available, such as the SQL Toolkit, Netezza Analytics Toolkit, Fluid Query, and third-party libraries.

In September 2017, IBM announced the IBM Integrated Analytics System (IIAS). Many major improvements were made over the prior versions, including a departure from BladeCenter to a platform based on IBM Power Systems™ servers. The most significant functional change however, is the seamless interfacing with Db2’s engine, which now opens doors to share the integration of IBM’s many other innovations only available for the Db2 family. The most significant performance change is the solid-state drives.

While much of the Netezza MPP experience remains intact and robust, the more important improvements include:

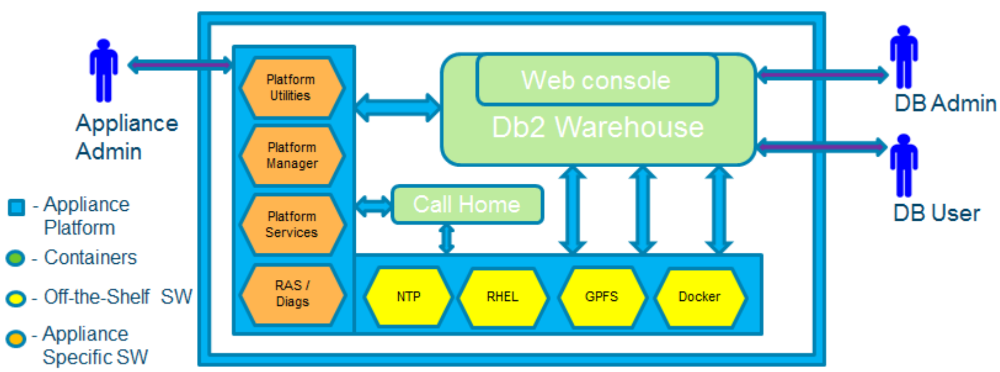
- Common SQL Engine (CSE)
- IBM Power Systems servers for horizontal scaling
- Solid-state drives
- Columnar compression
- Integrated analytics
- Administrative consistency



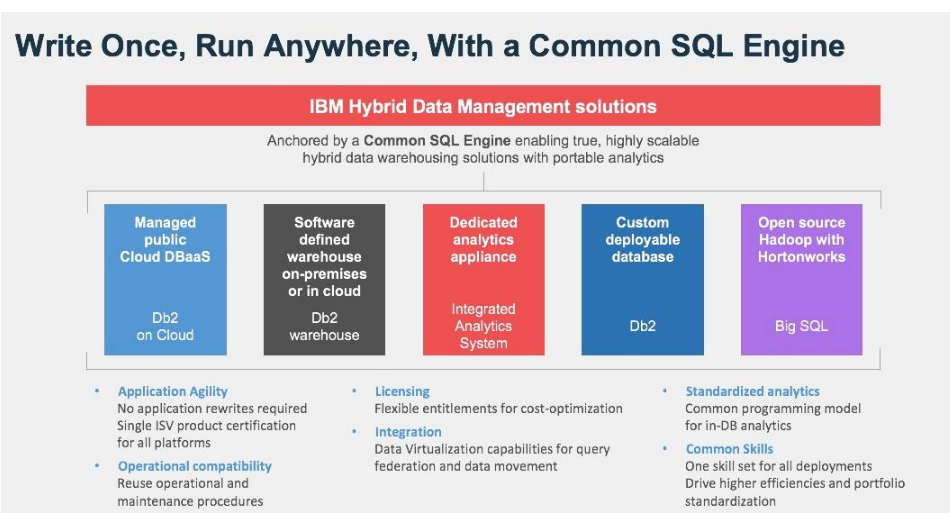
Storage	CPU Cores	Nodes
15 TB	72	3
30 TB	120	5
45 TB	168	7

The system can be incremented in one-third racks.

IBM's September 2017 announcement



IBM's layout of the internal software systems



IBM's announcement: One SQL engine for the Db2 family of database products

Common SQL Engine (CSE)

The most dramatic and visible change (and capability) is the merging of Netezza Technology with the Db2 SQL engine. The original Postgres-like syntax is gone, and the data is accessed by Db2 syntax with some Netezza-centric extensions.

This exposes the Netezza technology to all the capabilities of the Db2 engine, and will be included in the innovations and upgrades forthcoming for all the Db2 family. This means it will share a common interface with Db2 and Db2 Warehouse (formerly dashDB), including the on-premise and cloud forms of these implementations.

SQL is therefore *functionally* portable across all these platforms in transparently interchangeable form.

Db2 Warehouse normally sits on commodity-configured hardware, and behaves as a normally encountered transactional engine with extended analytic capability. The lack of MPP-configured hardware, however, limits its capacity as a highly scalable analytics platform. Its use of indexes to search for records places a hard limit on its query turnaround efficiency.

The IBM Integrated Analytics System (IIAS) uses MPP-configured hardware plus the extended analytic capability, and between these two provides an unprecedented integration of software function and hardware power. Add to this IBM's General Parallel File System (GPFS™), and best-of-all-tech is under one umbrella; users will experience no inherent functional or performance limitations with this configuration.

A caveat to this capability (discussed later) is that, while the SQL for Db2 Warehouse is functionally portable to IIAS, it will not necessarily provide efficient performance on the MPP substrate. As is the case with common Netezza-centric migrations, tables and queries that work fine under Db2 Warehouse on commodity hardware may not scale under IIAS, and in fact may stress the platform with inefficient operation.

IBM Power Systems Servers for Horizontal Scaling

The IBM BladeCenter blade server technology (TwinFin, Striper and Mako) had the drawback of being inelastic for scaling. An upgrade from a one-rack to a two-rack system required the two-rack to be placed alongside the one-rack, and data copied from one to the other. IBM attempted to leverage a "growth on demand" model, which didn't behave as consistently as required.

In the IIAS model, Power Systems servers are the underpinning hardware, so can be *horizontally elastic*; place another Power Systems server alongside the existing one(s), perform simple connection and reconfiguration, and the data will rebalance itself with no further effort. This is how the original Netezza Mustangs would horizontally scale, and is significantly more elastic than offerings from competing vendors.

Moreover, this hardware configuration is already cloud-enabled, as is the Common SQL Engine, a huge asset for those wanting to explore cloud-based analytics.

Columnar Compression (via BLU Acceleration®)

Through the Mako release, all Netezza tables are row-based. Netezza reads pages rather than rows, and once a page arrives in memory, it is decompressed and stripped of the unnecessary rows and columns, such that only the required columns and rows survive and enter the CPU.

As the depiction at right shows, Db2 BLU Acceleration is available for IIAS.

Columnar tables are now available, which is highly advantageous for those using analytic methods to query data. This allows us to search or query specific columns rather than the whole rows. A given page is read into memory, but only the required column is stored on it, and not the entire row.

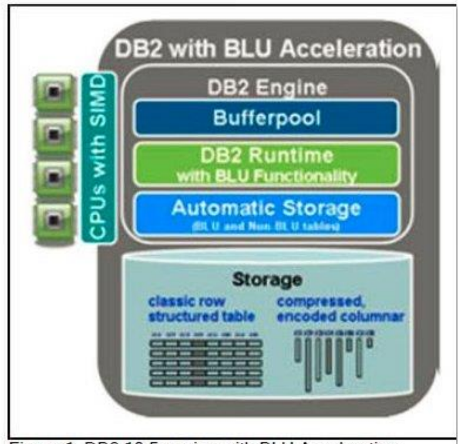


Figure 1. DB2 10.5 engine with BLU Acceleration

Columnar databases have been available from other vendors for eight years or more, and have proved their performance capability to compete with Netezza for many common applications. IBM enabled Db2 for analytics through BLU, which provided columnar tables to compete with these vendors, but only on "par" without the ability to accelerate beyond.

Now the BLU columnar capability is available for Netezza tables (row-based is still an option, but not the default). This coupling will allow Netezza to scale storage and performance far beyond its columnar-only competitors. In short, it provides the best of both worlds: row-based tables for common storage, and columnar-based for analytics power.

This is just one example of how a mature capability in another IBM technology is made available through this integration.

Solid-State Drives

Netezza's current offerings leverage electro-mechanical drives, which have two drawbacks: the first is occasional failure that requires replacement, and the second is the physical read-head, which must be dynamically dedicated to the needs of a given query, effectively serializing the queries at the disk level.

For sheer performance, Netezza paid a lot of attention to drive configuration. For example, a Netezza hard drive is divided into three sections: the outer ring holds user data), the middle ring holds user data from a sister disk, and the inner ring holding system data. This way, on any given query, the user will access the fastest spinning disk. It also means each query will ultimately have to queue behind another for serialized access to the drive's data. The read-head must be positioned for the given query, read the data, and then be positioned for the next query and so on.

Solid-state drives are memory-based, and have orders-of-magnitude faster read-speed than electro-mechanical drives. They neither experience regular failure, nor do they have read-heads. Query access is no longer serialized, improving both performance and concurrency. And all the drive's memory is accessible asynchronously.

An interesting shift has occurred here. In Netezza Mako and prior, the single most significant performance drag was reading the disk, which now may be of far less consequence than other potential query bottlenecks.

Integrated Analytics

IBM has integrated the libraries for analytics and others into the engine without further installation, upgrade or maintenance.

Moreover, IBM has included Data Science Experience (DSX) and Apache Spark, two of the most productive environments for data science. This is an important move; as DSX and Spark improve over time, data scientists will be able to use them on different platforms, anywhere the Common SQL Engine is available.

The systems still support stored procedures and UDXs, for more surgical control at the MPP level.

Administrative Consistency

A common benefit with Netezza administration is that it's "part time," which is to say less than one-fourth of a full-time database administrator (DBA) is required to maintain and oversee the machine. This can become an issue in a heterogeneous environment with multiple different database engines, where the DBA is immersed in the other engines and, only upon occasional request, addresses the Netezza technology. The administration of Netezza is different enough that it requires specialized knowledge on the technology. This is the only case where the simplicity and self-contained nature of Netezza could lead to a potential risk of lost knowledge or a slow response to a critical problem.

The consolidation of Netezza under the Db2 engine family makes its administration consistent across the platforms. Configuration, administrative tools and behaviors, etc. are all understood, invoked and monitored in the same way. This alleviates the pressure on DBAs to acquire (and maintain) specialized knowledge of one, infrequently-administered technology.

Considerations

A common propensity of new Netezza users is to move existing Third-Normal-Form (3NF) structures from a transactional source into the Netezza machine, and begin reporting from these structures "as is."

A Common SQL Engine as supported by the IIAS platform might make this propensity easier to implement.

A common outcome of this approach however, is a perniciously increasing performance drag until the queries "spike" in the duration, if they come back at all. This "tipping-point effect" is well documented and part of the Netezza experience across the world. This, of course, is something the Netezza users should avoid, and rather than report from 3NF-style tables, should reformulate the data into a dimensional-style model to consolidate, denormalize and simplify access for the users. Netezza is a data warehouse appliance, and presumes its users will apply data warehouse practices, a mainstay of which is to transform source data into a consumption-ready form.

Unfortunately, the tipping-point effects are not normally visible at the inception of an implementation. The initial “free” 10x to 100x boost in performance is so exciting, people “run with it” and don’t realize they might be sacrificing a 1,000x or even 10,000x boost in capacity, which is readily achievable with a few tweaks to data structures and queries.

This is not to be taken lightly, because if a query is running 10x faster when it could be running 100x faster, we have simply agreed it is okay for the query to take 10x more time and 10x more machine resources to complete the work. This inefficiency will eventually catch up with us.

However, because the disk drives in IIAS are so much faster, it might not catch up to us as quickly. This is the primary caveat of high-powered systems—that their power can mask inefficiency. Or rather, Netezza can make a terrible query look awesome, and a horrible data model look like a champ. As the data grows, the weaknesses will slowly emerge. A query with an original duration of five seconds today might be 10 seconds six months from now, or triple the duration within a year. Unfortunately, it doesn’t just degrade until it “takes too long.” It will degrade until it hits a hockey-stick-style tipping point, and may never finish at all.

A common symptom of this effect is the necessity to write highly complex queries and nested views, such that a great deal of work happens “on demand.” A typical example of this query style follows:
Select <columns – complex “case” statements, nvl/coalesce scrubs>
From <table or view>
Left Outer Join <table or view>
Left Outer Join <table or view>
Where (columns – with complex “case” statements or SQL functions in the “where” logic)
…

The presence of “left outer join,” complex nested views, on-demand scrubbing, where-clause functions or transforms, etc. is indicative of a model that is not “consumption ready.” The key problem with this model is the on-demand necessity of re-integration, re-calculation and re-formulation *of the same data* for each query instance. For example, calling “upper()” in the where-clause for a table’s column compare will invoke the “upper()” for each row, and for billions-of-rows databases, this is invoked billions of times.

We must keep in mind the Netezza platform does not use indexes. Also, the Netezza platform is a data warehouse appliance, and users are expected to embrace the basic principles and practices of data warehousing.

One of the primary maxims of data warehousing is that *the table structures in a transactional source are poorly configured for high-speed analytics*. This does not change by moving them to a massively parallel platform. In fact, it makes the situation worse, because even though the queries may run faster than the original platform, they are still orders of magnitude away from their optimum duration. Moreover, they are so inefficient that they saturate and stress the machine, and will degrade in performance as the data grows.

The spirit of data warehousing is to transform, integrate, consolidate, denormalize and optimize the multifold tables of a transactional model into fewer, larger tables, usually in the form of a dimensional model. To get there, the data is pre-integrated, pre-calculated and pre-formulated to minimize the on-demand query drag. Queries simply consume, they do not re-integrate and re-calculate.

Some leading consultants have even advised their Netezza clients to simply move the data from the transactional source and use them "as is" for reporting. This has a highly predictable outcome:

- Complex queries
- Query durations measured in minutes
- Performance degradation as the data grows

This is because the original (source) data structure implementation is:

- Optimized for transactions, not analytics
- Configured to use primary keys and indexes, not distribution keys and co-location
- Improperly leveraging distribution and zone maps
- Largely seen as logical rather than physical, where queries are tuned first

As IIAS continues to circumscribe the best practices of data warehousing, including moving the third-normal-form/transactional model into a dimensional/hybrid model, plus deliberately pre-integrating, pre-calculating and pre-formulating data points, some highly desirable outcomes follow:

- Simplified queries, including the elimination of scrubbing functions
- Query durations in near-second and sub-second
- Performance is stable as the data grows

Because the new data structures are:

- Optimized for analytics
- Configured to use massively parallel joins
- Properly leveraging distribution and zone maps
- Largely regarded as a synergy between the physical and logical

Conclusion

In conclusion, the primary SQL considerations when moving from the standard Db2 Warehouse configuration to an IIAS hardware configuration are that:

- Queries are logical, but performance is physical.
- The SQL may be functionally portable, but performance is not in the SQL.
- Structures must be reviewed for proper distribution and organization, so they perform according to the queries using them.
- There is no such thing as a general-purpose schema in Netezza MPPs, and this does not change with IIAS. Schemas should be configured and optimized separately for each user group.
- Index-dependent queries must be reviewed and modified to properly leverage distribution and zone maps (now synopsis tables).
- Many such changes in a mature implementation may be trivial, but not if the model remains transactional in nature and was never transformed to a dimensional model.
- The step to the dimensional model is the next order of business, which will necessarily require changes to the SQL.
- The new model and SQL will however, be optimized and simplified in this approach.

IBM's bold step to integrate the most powerful analytic concepts into a common engine arrives with major benefits to the Enzee Community, as key problems are solved and key innovations are merged.

Migration to IIAS will provide an unprecedented performance boost for analytics, eclipsing all other vendors while maintaining flexible adaptability for on-prem or the cloud.