kognitio

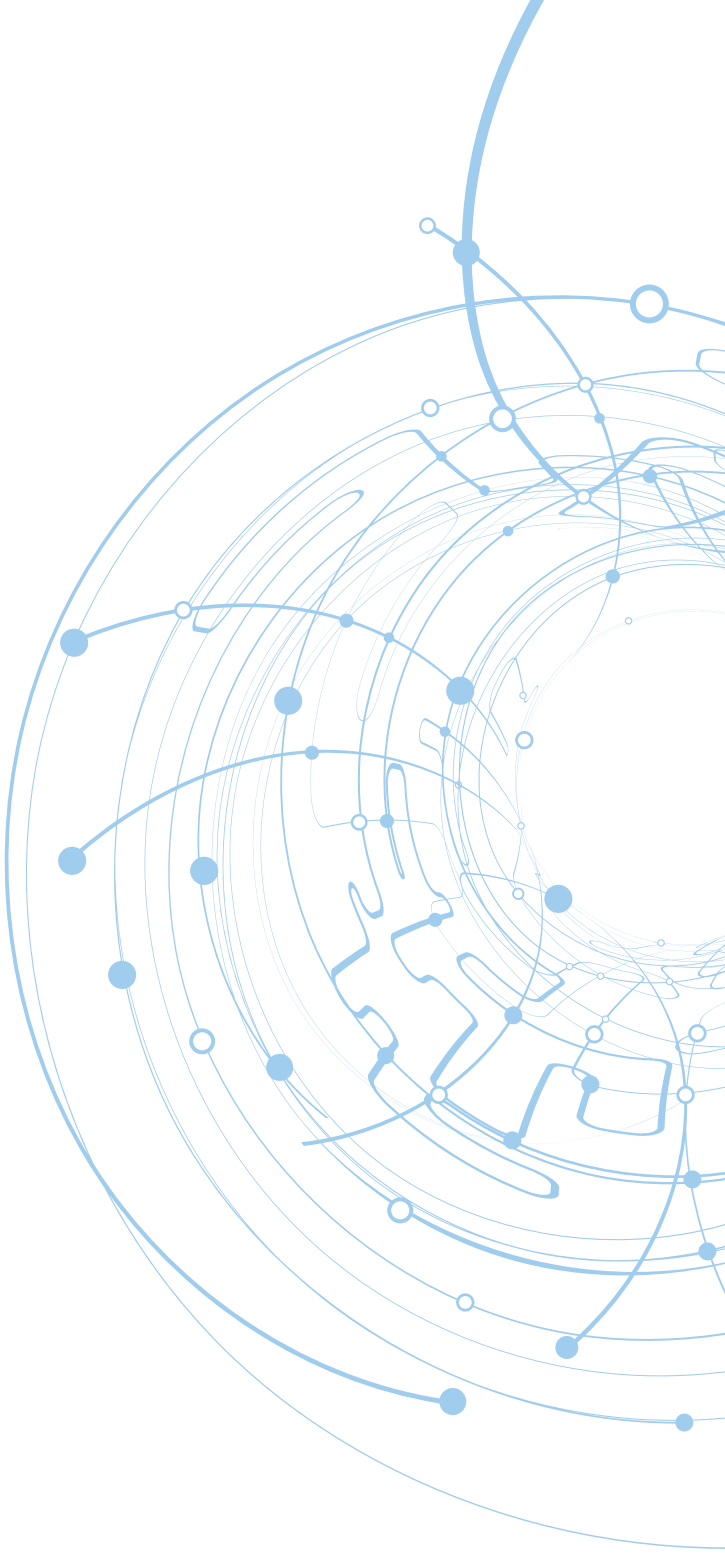
Kognitio Analytical Platform

Technical Profile Overview

Kognitio is a pioneer in high-performance, scalable Big Data analytics for Data Science & Business Intelligence

Updated to include Version 8.1

Contents



Purpose

The *Kognitio Analytical Platform, Technical Profile*, Overview is designed to provide a full technical introduction to the Kognitio Analytical Platform: its purpose, strengths and principal architecture.

Introduction

The **Kognitio In-memory Analytical Platform** is the core technology underpinning everything Kognitio does. The Kognitio Analytical Platform is a software product that can also be supplied as a pre-integrated appliance or as a cloud based service.

The technology was previously known as Kognitio WX2. The new name now encompasses the previously separately branded Pablo MDX Interface, Picasso Cube Designer and WxConsole products.

For the purposes of the rest of this document the Kognitio Analytical Platform will be referred to as simply "Kognitio".

Kognitio allows users to easily pull very large amounts of data from existing data storage (persistence) systems into high-speed computer memory, allowing complex analytical questions to be answered interactively, regardless of how big the data is. Its very high performance and scalability make it ideally suited for this purpose.

The data storage or persistence systems can be existing traditional disk-based data warehouse products, operational systems, Kognitio's internal disk subsystem or, increasingly, distributed parallel file systems such as Hadoop.

The key reasons behind Kognitio's very high performance are:

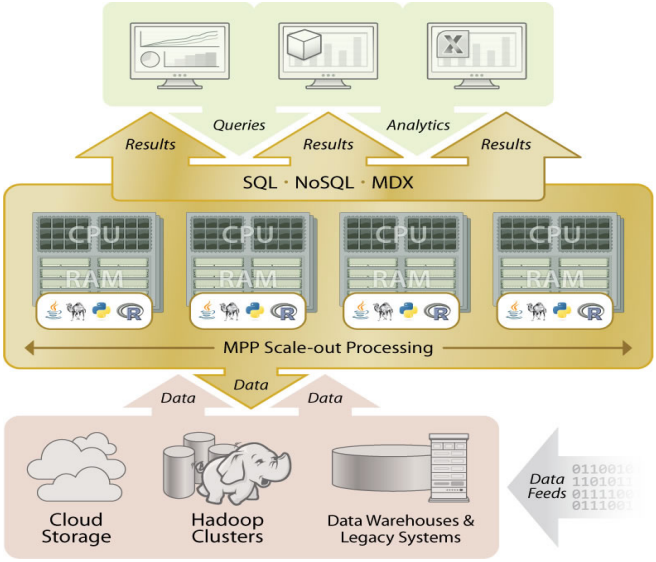
- Data is held in very fast high-speed computer memory or RAM
- Data is held in structures that are optimized for in-memory analysis. It is not a copy of disk-based data i.e. a traditional cache
- Massively Parallel Processing (MPP) allows platforms to be scaled out across large arrays of low-cost industry standard servers from 1 to 1000+ servers
- True query parallelization allows every processor core, on every processor (CPU), on every server to be equally involved in every query
- Machine code generation and advanced query plan optimization techniques ensure every processor cycle is effectively used to its maximum capacity; this "shared nothing" approach has been a Kognitio hallmark from the software's earliest days

Unlike many competing products, Kognitio's in-memory Analytical Platform can handle even the very largest data sets associated with the emergent "big data" market. It does this by scaling out across arrays of low-cost industry-standard servers in the same way that Hadoop solves the "big data" storage and processing problem. For this reason Kognitio can be seen as a natural fit with Hadoop infrastructures. Kognitio, unlike Hadoop, has speed and industry standard application connectivity and so can deliver a high-speed analytical layer for business applications wanting to interoperate with Hadoop.

Kognitio supports many open standards for connectivity to front-end applications and visualization tools, thus allowing users to continue using their tool of choice, whether that be an SQL-generating relational tool or an MDX-generating OLAP tool.

Kognitio does not manipulate the data in any unnatural way to gain its industry-leading performance levels. The data held in the Kognitio platform is in its natural row-based state, which means that it is fast to load (no complex data reformatting) and it can support any workload without time consuming changes to the data structure. Other products typically use columnar storage techniques to increase performance, an approach that was designed to minimize disk I/O (Input/Output speed) for simple queries at the cost of slow complex load operations. The benefit of columnar storage also diminishes rapidly as the query complexity increases, as the emphasis moves away from how the data is stored to complex processing of the data of interest. With the increasing move towards advanced analytics rather than simple Business Intelligence (BI) reporting, the need for low-latency complex processing will become the key requirement of any analytical platform and how the data is stored will become irrelevant. Instead,

a platform's ability to "do high-volume work" (e.g. calculations, computations, text processing etc.) in near real-time will become the key metric. Kognitio is designed from the ground up to "do lots of work" quickly no matter how big the data is or how tough the calculations.



Analytical Platform

To the outside world, the Kognitio Analytical Platform looks like a Relational Database Management System (RDBMS) in the same way that Oracle™, IBM DB2™ and Microsoft SQL Server™ are databases. However, unlike these databases, Kognitio has been architected specifically for an analytical query workload as opposed to the more traditional on-line transaction processing (OLTP) workload. The optimal architecture for an effective transactional system is very different to that required for successful analytical performance.

True MPP

While Kognitio has been delivering in-memory analytics for more than 20 years, it has recently been joined by a growing new breed of databases designed specifically for analytics. All of these databases claim to use the principle of shared nothing, massively parallel processing (MPP) to solve the problem of performing complex analytics across large volumes of data. The degree to which these different analytical databases parallelize their operations varies and they cannot all be classed as "true" MPP. Some, like Kognitio, parallelize all aspects of their operation, whilst some only parallelize their data scanning. Despite the different degrees of parallelism, all of these databases have a key feature in common; they split the data and queries across many individual computer elements or compute nodes. Each individual node has a portion of the total data, individual queries are sent to all nodes and each node works on its own portion of the data.

Transaction Processing

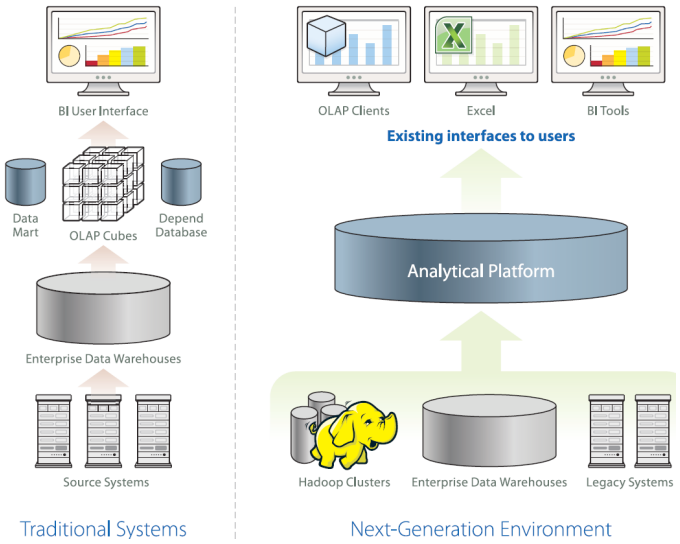
High performance OLTP database architecture, on the other hand, requires each node to be able to see all of the data. Even when an OLTP database is run on a cluster of nodes, each node needs access to a complete copy of the data. More usually the data is distributed, but each node needs to see the data held on physically different nodes. This creates huge amounts of inter-node network traffic, limiting OLTP database clusters to a small number of physical nodes. In fact, eight nodes is a large OLTP cluster and it is well known that Oracle Real Application Clusters (RAC), for instance, is best limited to two nodes. Even when several nodes are used an individual query is general satisfied by only one node in the cluster. The different architectural requirements of an OLTP system versus an analytical system means that OLTP databases perform poorly when asked to do analytics on large data volumes. Conversely, analytical systems have relatively poor transaction processing performance. Some analytical databases are actually unable to do any transaction processing. Kognitio supports full transaction processing and is ACID compliant, but its performance is moderate when compared to a high-performance OLTP database.

An Analytical Database

Although Kognitio can be classed as an Analytical Database, it operates very differently to the majority of other analytical databases on the market. To help clarify this differentiation, we use the term "platform" as opposed to "database." Although Kognitio has its own optional internal disk subsystem, it is primarily used as an analytical layer on top of existing storage/data processing systems, e.g. Hadoop clusters and/or existing Enterprise Data Warehouses, cloud storage etc.

Changing the Analytical Model

The very high-performance levels achieved by a Kognitio Analytical Platform are about far more than simply making things go faster. In-memory analytical platforms fundamentally change the way organizations will go about building future analytical infrastructures. The traditional analytical infrastructure with its onerous data latency, lack of flexibility, poor scalability and high maintenance will move towards a much more dynamic model, based on the power of low-cost commodity hardware rather than relying on expensive system administration skills. The next figure contrasts the two approaches.



In-memory Platforms

Most database systems, whether they are OLTP or analytical, store data on mechanical disk. Mechanical disks are relatively slow devices and the speed with which data can be read from disk is very limited. Mechanical disks generally have a maximum read speed of around 100MB per second. Disk I/O speed is the primary performance bottleneck for disk based databases. Writing data to disk is even slower than reading, so analytical queries that generate intermediate temporary result sets are further impacted by the need to perform disk write operations.

Kognitio, on the other hand, is an in-memory system. The data of interest is held directly in fast Dynamic Random Access Memory (DRAM), typically just called RAM. Every industry-standard computer or server has some RAM (they cannot function without it), but modern low-cost industry-standard servers allow very large amounts of RAM to be fitted at very low cost.

Memory is much faster than disk. If you have ever run Microsoft Windows on a computer that has run low on memory, you will have experienced a dramatic and painful slow down when running applications, as Windows tries to swap data to and from disk.

A typical industry-standard server will have memory with access speeds of at least 6400MB per second. This is 64× faster than a disk drive read and more than 100× faster than a disk drive write. It is also important to note that DRAM as its name implies, is a random access device where data can be read or written in very small chunks, from anywhere in the memory, with virtually no overhead. Disk drives, on the other hand, are a sequential block access device, which means that data is read in sets of sequential blocks. During a read operation, these blocks must be found and read from the drive and copied into RAM before the data of interest can be extracted. This three stage operation slows access to the data of interest even further.

Moving between blocks on the disk usually involves "seek time." This is the physical repositioning of the read head over the new track and is a very slow operation. Analytical databases are generally relatively "seek time immune," as data is normally scanned sequentially. However, when an analytical query involves the generation of an intermediate result set, the seek time becomes hugely significant, since the disk must now read data from one area whilst also having to write the intermediate result sets back to a completely different area of the disk.

Kognitio does not write intermediate result sets back to disk. In fact, when all the data of interest is held in memory, Kognitio does not need to perform any disk access even when executing the most complex of queries. Instead, intermediate result sets are created in memory, and leverages Kognitio's sophisticated query streaming mechanism that allow queries to run, even if the available free memory is too small to hold the intermediate result set.

In-memory vs. Cache

At first glance, in-memory simply sounds like a large cache, but it is in fact very different.

A cache is a buffer of the most frequently used disk blocks, held in-memory, for opportunistic re-use. Only the caching layer knows which data is actually held in-memory. So when a query is physically executed, the processors must run code that asks the question, "is the data I need cached or not cached?" for every row. This code is not trivial

and significantly increases the number of instructions the processor has to execute as it runs the query. Caches themselves are highly dynamic depending on what operations are accessing data; the contents of a cache can widely vary over time – processing cycles are also wasted merely calculating what data blocks are best retained in cache.

When data is loaded (pinned) into memory by Kognitio, it is explicitly formatted and placed in structures that guarantee immediate ongoing random access; every aspect of the system knows which data is held in memory and which is not. When the Kognitio optimizer/complier produces a query plan, it can therefore take into account the different cost of memory access versus disk-based data fetch and produce an appropriate plan, depending on whether or not all the data resides in memory. More importantly, the code being executed does not need to keep asking the "data cached, not cached?" question. This reduces the code path length by a factor of 10.

In a Kognitio system, the data loaded into memory is not just a copy of a disk block. Instead, the data is held in structures designed to take advantage of the low-latency random access nature of RAM. When combined with Dynamic Machine Code Generation, this also significantly reduces the code path length, thereby increasing query performance.

Because Kognitio software has been engineered from its earliest version to work against data held in RAM, all of its algorithms for processing data (joins, sorts, grouping, etc.) have been specifically optimized to take advantage of the random access nature of memory as well as modern CPU instruction optimizations. This is not true of other databases that are fundamentally designed to work against disk-based data and which have introduced, at a later date, extended caches or faster I/O sub-systems that have been inappropriately labelled as in-memory data storage.

Solid State Disk Drives

Solid State Disk drives (SSDs) are now available that use silicon memory to replace the mechanical disk drive. SSDs cannot be considered as being equivalent to computer memory or RAM for several reasons. Although they do make disk based systems faster, there are several reasons why they certainly do not produce anything like the same level of performance as in-memory platforms:

- SSDs do not use DRAM, instead they use a much slower memory technology called FLASH
- SSDs mimic conventional disk drives; as such, they are block access devices
- Server class SSDs are still very expensive
- SSDs have lower capacities than traditional hard disk

On paper, SSDs seem to have a significant performance benefit over hard disks. In fact, the bulk of this performance benefit comes from the elimination of seek times. As previously discussed, analytical databases are relatively seek time immune, so the performance gains are not as dramatic. An application that involves fairly random disk access (OLTP database, file server, Windows, etc.) may see a 10–20× performance increase from SSD, whilst Kognitio's testing of SSDs in an analytical platform showed a more modest 2–3× increase in performance.

Whilst this is still significant, it produces nowhere near the performance level of DRAM. The high cost of server class SSD drives also means that, terabyte for terabyte, fast DRAM is not much more expensive than SSD drives. So, why do people use SSDs? Because the vast majority of applications were designed to work with disk drives and are unable to work instead with large amounts of memory. Simply replacing the mechanical device with a solid state device means that they can get some performance gains without any re-engineering of the product. The complication is that the removal of the performance bottlenecks from the disk I/O exposes the code's inability to parallelize across all the available CPU cores and its inherently inefficient use of the CPU cycles. This means that a significant amount of the potential performance gain available from in-memory is not realized.

"True" Parallelism

Having all the data of interest held in computer memory does not, in itself, make an analytical platform fast. Memory is simply another place to "park" the data. What makes an analytical database fast is its ability to bring lots of processing (CPU) power to bear against a given data set.

Analytical queries, when compared with transactional queries, are generally looking for patterns of behavior across large portions of data, as opposed to transactional queries, which usually hit a very small number of rows and involve relatively simple operations. Analytics usually involve complex processing operations, across large portions

of the data and, assuming they are not disk I/O bound, they become CPU bound.

Because of the compute-intensive nature of analytical queries, the ratio of data-to-CPU cores becomes very important and is a key measure of a platforms ability to "do work" with data, as opposed to its ability to simply store data. In a disk-based system, this ratio is very high since the data can only be accessed quickly enough to keep a small number of CPU cores busy. Holding data in very fast computer memory allows huge amounts of CPU cores to be kept efficiently busy for the duration of a query. Kognitio platforms usually have a data-to-core ratio of around 4–8 GB per core.

To support the sorts of data volumes that organizations are trying to analyze today, an in-memory analytical platform must be able to parallelize every individual query operation across any number of CPU cores, from a just a few to tens of thousands.

Kognitio succeeded in that mission: its in-memory analytical platform can scale from a single core on a single CPU, in a single server, to thousands of cores, in hundreds of individual servers. Each and every core participates equally in every single query; advanced techniques such as dynamic machine code generation are used to ensure that every CPU cycle is efficiently used. This is "true" parallelism.

> As data volumes grow, true parallelism allows a platform to be expanded a few servers at a time, maintaining the data/core ratio and keeping query performance constant. This is linear scalability. True parallelism also allows a platform's performance to be increased by simply lowering the data/core ratio.

Kognitio is able to support true parallelism because it was designed from inception to do so. It is not a conventional sequential disk-based database that has been modified to parallelize parts of its operation. From its inception in 1988, Kognitio was designed to be a fully parallel, in-memory analytical platform and has been under continual development since then. For many years, this was regarded as niche technology. The advent of modern low-cost industry-standard servers with huge amounts of memory, however, combined with the explosion of "big data" and the need for more accurate and complex analytics has made in-memory analytics the hottest topic in today's data analysis market.

Data Persistence

The Kognitio in-memory analytical platform stores the data being analysed in RAM. By its nature, this memory is volatile; when the computer is switched off or restarted the data in memory is lost. So where in a Kognitio platform does the data "persist" i.e. what happens when the lights go out?

Kognitio currently supports two methods of persisting the data. It can store the data on its own internal disk subsystem or can access the data directly from "external tables".

Optional Internal Disk Subsystem

The internal disk subsystem capacity is the sum of all the disk storage attached to each individual server. This storage can be formed from the locally attached disk on each individual server or from SAN or network attached storage. Kognitio software manages the distributed storage and provides RAID across groups of servers to protect against disk and server failures.

The storage is fully scalable and parallelized. Each logical disk is accessed in parallel and the overall bandwidth is directly proportional to the number of physical drives e.g. if an individual disk can provide 100MB per second read rate then Kognitio has n × 100MB per second aggregate bandwidth, where n is the number of disks.

When data is loaded into the Kognitio platform, the user can choose to load only to memory, to internal disk and memory, or just to disk.

Users of Kognitio see data held on disk as conventional schemas comprised of a collection of tables and can either run queries directly against that data, in which case Kognitio will automatically pull into memory just the data it needs to complete the query, or tell Kognitio in advance to pull all or specific portions of the data into memory. In the latter case, the data will stay memory resident ("pinned") once it is pulled into memory until it is explicitly dropped from memory (a big difference from a cache). This is described in more detail in subsequent sections.

Although Kognitio has a capacity-based licensing model, the license only pertains to the available memory.

Specifically, there is no charge for data that is held on the optional Kognitio internal disk subsystem. Put simply, if a user has a system with 10TB of memory but chooses to store 100TB of data on Kognitio's internal disk, the user only requires a license for the 10TB of memory.

External Tables

Kognitio allows persistent data to reside on external systems. External Tables is a feature that allows the system administrator, or privileged user, to simply setup access to data that resides in another environment, typically a disk store such as a Hadoop cluster or a data warehouse.

Once access is defined, Kognitio users see this external data alongside or instead of the Kognitio disk resident internal data and can choose to pin it into memory within Kognitio, either manually or automatically, in the same way as internal data held on the Kognitio local disk.

Kognitio uses external connectors to implement connectivity for External Tables. This framework allows Kognitio to rapidly support a range of external systems, including Hadoop, Cloud Storage, Data Warehouses, File Systems and Operational Systems.

Data Loading

As mentioned above, Kognitio allows data to be optionally loaded to memory-only, to internal disk and memory, or just to internal disk. Data can be loaded via multiple parallel streams into the same table or different tables and tables can be queried while they are being loaded.

The loader is designed to be very fast, supporting demonstrable load rates into memory of 14TB hour on a moderately sized system of 24 servers with 10GbE networking. Subject to the limits of the delivery network's bandwidth, the load rate scales as the platform size increases. The load rate to local disk is a function of the disk's write speed multiplied by the number of physical disks present, and scales linearly, again subject to the limits of the delivery network's bandwidth.

The Kognitio client side bulk loader sends data to the Kognitio Platform in the data's original format. Conversion can be performed on a massively parallel basis within the Kognitio Platform, dramatically reducing the impact on the source system sending the data. Kognitio also supports trickle loading for continuous delivery and real-time environments, with commit data by row count and/or time window e.g. every five seconds or 1000 rows, whatever comes first.

Options for Pinning Data into Memory

If a query is run against data that is held on the Kognitio internal disk subsystem or an external disk subsystem, Kognitio will automatically import the data it needs into memory and execute the query. On completion of the query, the data is automatically dropped. However, for the highest possible performance, Kognitio allows all or portions of the data to be permanently pinned in memory. These are called "images." Queries that access imaged data do not have to access data contained on disk at all.

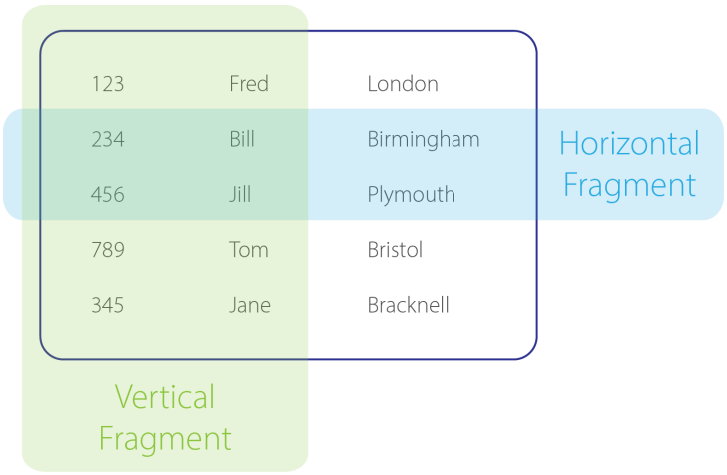
Simple one-line extensions to standard ANSI SQL are used to create memory images. Issuing these commands causes data to be read from disk (internal or external) and loaded into RAM. Memory images can be defined in a number of different ways; this allows for the optimum use of the available memory for a full range of analytical operations.

Fragmentation

Any table or portion of a table can be instructed to become a memory image. The table can be loaded into memory in its entirety or vertical and/or horizontal fragments of the data can be loaded.

Vertical fragmentation is used when there are fields in the table that are not often used for analysis e.g. address text fields or comment fields. Horizontal fragmentation is often used when the majority of analysis is performed on ranges of data such as the most recent data and the rest is accessed less frequently. For example, the last year of transactions would be held in memory and the rest left on disk.

For data that is stored on Kognitio local disk, Horizontal fragments are created by simply adding a "where" clause to the image creation statement or by creating a conventional view of the data and then instantiating the view into memory. When a table is horizontally fragmented, the system will automatically execute the query from memory



if all the data it needs is memory-resident. If it is not, it will automatically perform the necessary disk scans to fetch any additional required data in order to fulfil the query.

Standard relational views (a virtual table defined via an SQL query) can be used to perform complex data manipulations, the results of which can then be pinned into memory (a "view image"). View images can be created for any view, including complex table joins and aggregates, their content is a read-only snapshot of the underlying data. Data from external tables can be pinned into memory using "*insert select into a ram-only table*" or via appropriately defined views.

Updatable / Read only

A benefit of view images is that they carry a very small row overhead (can be zero overhead) when stored in memory. Full table images ("table images") or vertical table fragments ("fragmented table images") from Kognitio local disk are updatable (read only vertical fragmented images can be created using views if required). When updates are applied to a local table that has a table image or fragmented table image, the update is applied to both the memory image and the disk-based data. The advantage of view images is that they use less memory than table images for the equivalent data, as they contain no links back to the original disk-based data. External tables must be treated as read-only.

Memory Distributions

Kognitio supports a number of memory distribution algorithms to support a wide variety of workloads. Available distributions include random, replicated, partitioned, sorted, hashed and partial hashed.

External Connectivity

SQL

The Kognitio in-memory analytical platform has been designed to be as open as possible so that it can work with a whole range of front-end (query and visualization) and back-end (ETL, ELT, DI) tools. To this end, Kognitio supports ANSI Standard SQL via ODBC (Open Database Connectivity) standard or JDBC (Java Database Connectivity) standard. Virtually all tool and application vendors support these well-defined standards.

Kognitio SQL support is very rich. Kognitio fully supports the core features of ANSI SQL:2008 (with certain exceptions) and many of the optional features of ANSI SQL:2008. The ANSI SQL:2008 standard encompasses SQL92, SQL99, SQL:2003 and SQL:2006. In addition, Kognitio supports a number of the new optional features present in the most recent ANSI SQL:2011 standard. A full list of the supported features can be found in the document, Kognitio Technote ANSI SQL:2008 Compliance Summary.

Kognitio has been verified against and has partnerships with many of the key vendors of these tools (see our website for an up-to-date list).

Additionally, Kognitio offers support for the most common Oracle non-standard SQL syntax variations and all of the Oracle non-standard functions. This support simplifies the process of making applications that were written or customized for Oracle run against Kognitio.

MDX and Virtual Cubes

Alongside SQL, Kognitio also supports the MDX (Multi-Dimensional eXpressions) language via ODBO or XLMA. MDX is the language used by applications that expect to talk to a pre-aggregated online analytical processing (OLAP) cube.

Building OLAP cubes is a time-consuming and administrative-heavy exercise, with large cubes taking many hours to build. They also deliver a "fixed-in-time" snapshot, so there is always the business pressure to update them more

frequently.

Kognitio has OLAP-class performance without the need for pre-aggregation. Instead, results can be calculated on the fly from the underlying detailed data, or with the Kognitio MDX connectivity and virtual cube technology, making this data appear as a cube to the application.

Virtual cubes are simply metadata that describe how the underlying relational data maps into a cube model. They are created using the Kognitio Cube Builder Tool. Once a cube has been logically designed, it only takes several seconds to publish the metadata and make the virtual cube immediately available for querying.

This has the advantage that new data can be made immediately available without requiring expensive cube rebuilds; changes to the cube structure can be made available in minutes rather than days.

Because of this inherent ability to perform well for complex queries, Kognitio is an ideal platform for the rapidly emerging "Advanced Analytics" market. Advanced Analytics involves applying advanced algorithms to data to try and gain more meaningful insight, including predictions about what might happen in the future, as opposed to simply using the data to report on what has happened in the past. These algorithms involve lots of heavy-duty data crunching and are CPU-intensive operations.

In-Database Analytics

Because of its ability to deploy large amounts of raw processing power against any given data set, Kognitio has always been an ideal technology for the complex analytics end of the Business Intelligence market. To explain what we mean by "complex" queries, it helps to think of most database queries as having two distinct steps:

– Step 1: the "filter" step. This is the step that finds the subset of the data in which the query is interested.

– Step 2: the "crunching" step. This is where the filtered data is processed to calculate a meaningful result.

In general, simple queries are dominated by the filter step with the crunching step being a trivial operation. For complex queries, the crunching step is predominant.

The problem with Advanced Analytics is that most of the algorithms used are difficult or impossible to express in SQL. These operations, therefore, have generally been performed externally to the data source or database. Extracting the data from the database to run it through an external algorithm is a painful process at best, but with large data volumes, it becomes pretty much impossible. The biggest challenge of all, however, is that to process large data volumes in a timely manner often requires multiple copies of the algorithm to be run in parallel, a very complex and difficult exercise for any organization to undertake.

The solution adopted by some is to throw away SQL altogether and invent another way of querying data; i.e., MapReduce. This is very much an engineering led approach to the problem and ignores the fact that SQL has many important strengths, as well as being the de facto standard for most BI tools and applications; most business users are familiar with SQL. Not having SQL access to data severely restricts the people within an organization who can freely interact with this valuable resource.

Other vendors have chosen to embed various analytical algorithms directly into the database. Some have done this in a way that allows parallel execution of the algorithm. Others have simply added a way of calling out to an external process, but only support algorithms and functions that they, the vendor, have specifically embedded within the product. Alternately, they restrict the languages that can be used to create these algorithms.

Kognitio has taken a different approach by allowing any script or binary that can be run in a Linux environment to be embedded into the database – e.g. R, Python, Perl, Java, SAS, custom scripts etc. This feature is called Massively Parallel In-Memory Code Execution. As long as the code can accept data in and send output data, Kognitio can execute it in-place within the database via code in-line with an SQL query. The code is automatically executed in a massively parallel context with one distinct copy of the code running on each and every CPU core by default. With some simple additions to the SQL syntax, Kognitio allows users to easily control the number of parallel code executions, data partitioning, data sequencing and break points in execution. Output from all scripts is gathered into a single virtual table that continues into the next stage of the query execution plan.

By taking this approach, Kognitio does not limit the analytics that can be run in-database to those specifically supported by Kognitio, opening up an amazing freedom of choice and capability. By using SQL as the management wrapper, business users can easily control the process and visualize the results using standard BI applications and tools, through traditional interfaces such as ODBC/JDBC.

Platform Requirements

As has already been mentioned, Kognitio combines the full power of arrays of unmodified industry standard, low-cost server hardware into a single, high-performance analytical platform. No proprietary hardware is required.

As long as the servers are x64- or x86-based and suitably networked, Kognitio can run on them. For a high-performance high-capacity system, Kognitio recommends the use of servers with a high number of physical cores (8–32) and large amounts of memory (128–512GB). Kognitio also recommends that each server has a minimum of dual 10Gigabit Ethernet network interfaces.

The requirements of the Kognitio Operating System (OS) are very modest – no clustering or other specialist software is needed – no specific OS configuration or tuning is required. A standard base level installation of 64-bit Linux (typically RedHat or SuSe Enterprise Linux but other distros work just fine) on each server is all that is needed for Kognitio to run.

In-memory Analytics for Hadoop

Hadoop is experiencing widespread adoption as the preferred flexible centralized data storage and processing platform for organizations trying to gain insight from ever-increasing volumes of varied data. There are a number of reasons for this; massive scalability, resilience and low-cost being the most important. The open source nature of Hadoop and the consequent lack of license fees are attractive to organizations that have rapidly growing data volumes and increasingly demanding BI requirements, but so is the ability to build very large capacity, resilient clusters using cheap non-enterprise standard hardware.

While Hadoop offers many advantages, its barriers to entry for most organizations are significant. Hadoop implementations require complex engineering skills, and a large amount of design and programming effort are needed to build a complete solution. Increasingly, many organizations appear ready to invest in this effort in order to reduce the amount of on-going license and maintenance fees that they pay traditional data warehouse vendors. A growing number of industry analysts agree with the opinion that within the next ten years, a model that is far more cost-effective and nimble will supplant the traditional data warehouse model.

Hadoop is not an Analytic Platform

With significant engineering effort, it is hard to deny that Hadoop can be used to build a cost-effective and hugely scalable data storage and processing platform, but it is by no means suitable as a platform for readily accessible business analytics. There are two key reasons for this.

First, while query and visualization tools are available for Hadoop, they are still primitive in quality. Complex analytics still requires a significant degree of hand coding by a skilled engineer, which takes time. For batch organized reporting, this may be acceptable, but it completely precludes ad hoc or interactive train-of-thought analytics. These tools will improve over time, but no matter how advanced they become, their use as interactive analysis tools will always be limited by the second key issue with Hadoop as an analytic platform:

Hadoop is very slow

Let us clarify that statement. It is true that Hadoop can bring a large number of servers together, allowing it to process huge volumes of data very quickly. However, when answering individual analytically-complex queries, its inherent inefficiencies and disk-based architecture means that it cannot provide the instantaneous response required for train-of-thought analytics. Interactive, ad-hoc analytics also requires low-latency, high frequency, highly-variable interactions with the data. Hadoop was not designed for this.

Power of In-memory for Analytics

The continuing commoditization of RAM provides us with an answer; in-memory technologies provide the

necessary capability in terms of speed and low-latency since they do not require pre-aggregation to achieve the levels of performance associated with analytics. However, they do not scale cost-effectively to multiple petabytes of information as Hadoop can. Consequently, in-memory technologies are to Hadoop what OLAP is to relational databases.

It is worth remembering one thing when considering the usage of in-memory databases as the analytical arm of Hadoop clusters: they must be scaled, even though the data sets may be a summarized or sampled view of the underlying raw mass of data. Summaries or projections of very large data sets remain large and cannot be addressed by single server instances easily or cost-effectively, if at all. This is especially true if the analytical entities run into multiple terabytes and/or have very low latency requirements.

Hadoop for Storage and Processing, Kognitio for Analytics

Kognitio is an in-memory, very high performance, and scalable analytic platform. Kognitio's massively parallel architecture (MPP) allows arrays of low-cost, commoditized, industry standard servers – the same hardware platform as used under Hadoop – to be combined into a single very powerful analytic machine, featuring many terabytes of fast computer memory and vast amounts of processing power. By keeping all the pertinent data in memory, Kognitio can get the most from every single core of every single processor on every server, enabling it to answer each individual query at near-OLAP speeds. It does this without any pre-aggregation or summarizing of the data.

> Kognitio delivers a "hub and spoke" architecture with Hadoop at the center, and Kognitio acting as a high-performance analytic spoke. Data from operational systems feeds into the Hadoop hub, where any required data preparation steps takes place. Datasets for analysis are made visible to Kognitio as external tables so that users can pull the data they need directly into memory at very high speed.

Although this looks like a traditional dependent data mart architecture with its inherent issues of data duplication, it is in fact very different, as the Kognitio Hadoop connector provides very tight product integration.

Using the external table functionality (discussed earlier), Kognitio users see Hadoop HDFS files as non-memory resident tables which they can then query directly, or alternatively instruct Kognitio to pin selected data sets or portions of data sets into memory for subsequent further analysis. The connectivity and tight integration is provided by two high-speed Hadoop connectors, the MapReduce Connector and the HDFS connector.

The MapReduce connector wraps Kognitio's data filtering and projection code together with the bulk loader tools into a MapReduce job, submitted on-demand in the standard Hadoop manner. The filtering and projection code is the same code that runs on Kognitio's internal disk subsystem. It filters out just the rows and columns required, before sending to Kognitio memory. The MapReduce job executes on all Hadoop nodes in parallel and Kognitio exploits this to send data on a massively parallel basis to all Kognitio nodes, with every Hadoop node sending data in parallel to every Kognitio node.

For example, a Hadoop dataset may contains five years of historic data, but the user is only interested in the last quarter and does not need any of the description fields. As an external table, the user sees the entire data set but can tell Kognitio to load only the data he or she is interested in into memory via a simple SQL "where" statement. Kognitio will ask Hadoop to do the gathering and filtering prior to sending the data. This tight integration means that Hadoop is doing what it is good at, namely, filtering rows and Kognitio does what it does best; providing a platform for low-latency, high-frequency, high complexity queries for data analytics.

While the MapReduce Connector works very well when large data sets have to be filtered out of even larger Hadoop data stores, the limitations of MapReduce means that there is always a fixed overhead, in the order of tens of seconds, to read even the smallest data set from Hadoop. For this reason, Kognitio also has an alternative connector that bypasses MapReduce and reads data directly from the Hadoop HDFS file system. Whole files must be pulled and no filtering is applied until the data is present in Kognitio – although a semblance of granular filtering can be achieved if data is stored split across files, directories or file hierarchies. This connector is primarily designed to allow smaller data sets, such as dimension tables, to be pulled very quickly into memory.

This bypassing of the standard Java-heavy ways of extracting data from Hadoop is done to provide simplicity and achieve scalable high throughput data rates between Hadoop disk-based data and Kognitio memory. Because of the scalability, the actual data rates will depend on the relative size of the two platforms and the core network but throughput rates of tens of terabytes per hour are achievable on moderately-sized solutions.

In Summary

With rich SQL and MDX and virtual cube support, the Kognitio in-memory approach allows users to access data from all over the business with minimal impact and engage with it, using existing application and/or via a huge range of analysis and visualization tools. This recognizes and supports the existing business investment and will help to centralize valuable analytical data models for wide use, avoiding copies and silos.

In the future, Kognitio will offer deployment on the same physical cluster as Hadoop creating a seamless environment. The Kognitio software is available on a free-to-use basis for configurations under 128GB of RAM and can be downloaded from kognitio.com/free-download.

www.facebook.com/kognitio
www.twitter.com/kognitio
www.linkedin.com/company/kognitio
www.youtube.com/kognitio
info@kognitio.com

About Kognitio

For more than a generation, Kognitio has been a pioneer in software for advanced analytics, helping companies gain greater insight from large and complex volumes of data with low latency and limitless scalability for competitive business advantage. Sitting at the nexus of Big Data, in-memory analytics and cloud computing, Kognitio extends existing data and BI investments as an analytical accelerator, providing a foundation for data scientists and analytical information services. The Kognitio Analytical Platform can be used as a data science lab or to power comprehensive digital marketing analytics; it runs on industry-standard servers, as an appliance, or in Kognitio Cloud, a ready-to-use analytical Platform-as-a-Service (PaaS) in a public or private cloud environment. To learn more, visit kognitio.com and follow us on Facebook, LinkedIn and Twitter.