



CRITERIA FOR ANALYTIC PLATFORM SELECTION





Criteria for Analytic Platform Selection

August 24, 2011.

www.paraccel.com

TABLE OF CONTENTS

Table of Contents	ii
Executive Summary	1
Driving to the Right Solution	2
Technical Criteria for Business Requirements	3
1. <i>Load and Go</i>	3
2. <i>Performance Without Limits</i>	4
3. <i>Enterprise Strength</i>	8
4. <i>Non-SQL Analytics</i>	9
5. <i>Rapid, Elastic Scaling</i>	10
Comparing Analytic Databases and Analytic Platforms	11
6. <i>Rapid Data and Analytic Integration</i>	11
7. <i>Advanced Function Assimilation</i>	11
Satisfying Your Need to Know	12

Executive Summary

You need to know. Educated guesses and gut instincts are not enough. And with all the volatility in business today, what you need to know is constantly changing.

The need to know hits suddenly; the time to react is short. Your customer churn goes through the roof, bam. A new style of fraud hits the bottom line, hard. You have to combine the product lines of two merged companies very quickly in an optimum mix, or the merged organization fails. The range of issues you face is wide but all share this: the urgency and intensity of your need to know. And perhaps the most urgent question is this: can your ad hoc analysis scale up quickly enough to keep you competitive?

Using an ordinary database, you may be able to muddle through merchandizing optimization or other recurring analytic issues if your IT team does excellent performance tuning. But you'll not be able to handle the unexpected, which requires exploring vast amounts of data deeply, and freely using complex analytics, the best SQL-based business intelligence tools, or highly iterative ad hoc queries; especially when an existing database simply never returns the answers you seek, or comes back so slowly that you cannot do enough iterative refinement of a query to be sure.

In the era of "Big Analytics", you need a platform that supports advanced statistical and data mining operations that can require multiple passes through many – even hundreds -- of terabytes of data, and they must do it swiftly. That's what it takes to detect patterns of collusion that indicate financial fraud before losses grow large. Otherwise risk will remain undetected, just out of reach of your analysis. Models that need to be tested with more than just a few weeks of data will go un-deployed, simply because your database can't go back far enough in time to distinguish which ones fit real-world behavior.

The old saying "Half your marketing budget is wasted, if only you knew which half," is no longer funny. Today you and all your competitors have the vast volumes of data required to measure and optimize your total spend. The first to get to the answer buried at the bottom of the data wins.

Your analysts know they can get to the bottom of almost anything. But managers and executives must empower them to explore the raw, detailed data for anomalies and statistical patterns that will indicate the cause and suggest a fix. They need conclusive data that will guide the right up-sell offers and boost revenue, that will illuminate a new business opportunity. Given the right database, your analysts can help you strategically to know what to do, next. Given an analytic platform, your analysts will be able to expand deep analysis to organizations across your enterprise.

Working against a legacy database, the answers they seek will have been discarded in summarization performed for reporting and processing efficiency. The insights may remain hidden. Data models that can yield daily metrics to guide business as usual cannot anticipate the line of questioning needed to deal with each rapid business change.

The business agility you keep hearing about starts with, and is delivered by, an analytic database. Pick the right one.

Driving to the Right Solution

IT is charged with finding an analytic database that can step up this challenge. Yet they may be hamstrung by one of the organization's most successful ventures, the enterprise data warehouse (EDW). Its infrastructure and process are critical to daily functioning. The folks running the EDW want to help you find answers to unexpected problems and pick up new opportunities, but the queue of similar requests is long and getting answers is hard. Data warehouses built on traditional database technology often depend on materialized views, indexes and specialized data models to provide adequate performance for periodic reporting and predetermined slicing and dicing. Data is carefully controlled, cleansed and summarized. Its meaning is well understood and documented in order to give the company reliable and consistent business intelligence.

But these traditional approaches are highly specific to the data model in use and the queries anticipated for the data. The tuning process required for adequate performance necessarily delays the urgent business need to dive into the data. Moreover, no tuning or design can optimize processing for truly ad hoc usage. The quirky transaction-level details where your analysts might find answers are lost in the process of fitting standard definitions and pre-calculating summary levels. The tuning required to speed up one line of questioning impedes another.

What's needed is a database that can return any query against any data model quickly, however deep into the data your analysts ask it to dive, no matter how broadly and spontaneously they must look, and however far back into the data they need to go. New generation analytic platforms are emerging to handle growing data diversity and leverage the analytics and data scattered throughout the organization.

So how can IT plan for the unplanned and unforeseeable? How can they get the answers you need quickly when today's databases require design and tuning to return answers at all? What does it take to provide query response times fast enough for analysts to stay focused and iterate quickly through many increasingly detailed questions as they explore for an answer?

The answer is an analytic database with the performance, usability, query handling and adaptability to ensure rapid return of the answers you seek. An even better solution is an analytic platform with the extensibility and integration capabilities to quickly bring in new data, integrate external analytics, and assimilate advanced functions. Selection criteria includes the following characteristics:

1. Load and Go
2. Performance without Limits
3. Enterprise Strength
4. Non-SQL Analytics
5. Rapid, Elastic Scaling
6. Rapid Data and Analytic Integration
7. Advanced Function Assimilation

In this whitepaper, we explain the business benefit of each of these essential criteria and the technical underpinnings required to achieve them at a level of detail appropriate for your technical leadership.

Technical Criteria for Business Requirements

Business leaders look to the data to know what to do. In a time of accelerating change, the reporting that guides business-as-usual operations cannot provide all the answers. Thus the need for an analytic database that will let you explore data in an ad hoc fashion to understand a problem or uncover new opportunities. What does it take to tease these patterns and relationships from data?

One problem to overcome is that the volumes of data holding the answers are exploding. Web teams want to understand what works best in terms of improving user experience, increasing sales and providing advertisers with more tightly targeted advertising, and the click-stream data that can tell them is measured in Terabytes. Retailers analyze detailed market basket data and shopping cart paths through stores to drive better merchandizing and marketing. Financial companies mine transactions to detect fraud and to segment their customers into most and least profitable. The leading lights in every industry are trying to analyze once unimaginable amounts of data for competitive advantage.

Traditional databases were not designed to spontaneously analyze large amounts of data. Most database management systems were designed to handle online transactions that involve finding and updating one or two rows of data at a time. They were not designed to run complex analytical queries and algorithms that often require making multiple passes through millions or billions of records. Business analysts cannot keep up if analyzing a day's worth of data takes a week, if complex queries run out of temporary table space or otherwise fail to complete, if iterating to the next answer takes so long that significant changes have taken place in the meantime.

Analytic databases or platforms that can successfully provide the answers business leaders need do all of the following:

1. Load and Go

What's meant by "load and go"? Expand the phrase to "load and go without tuning" and the answer is clear: issue a simple command to load, and be ready to run queries without a need for specialized physical schemas, aggregation tables, indexes and materialized views.

Not needing this labor- and compute-intensive tuning reduces the time to load very large data volumes, whether it's facts stored in the data warehouse or normalized production data where the answer lies in the lowest level of detail. Quick loading would be pointless, however, if the database was not then able to "go". And by "go" we mean to find whatever answers are buried in the data with queries that return quickly enough to allow iterative refinement whether operating against normalized, de-normalized, or dimensional schemas.

Databases designed for analytics must have this ability to return ad hoc queries against ad hoc data collections without labor-intensive modeling and performance tuning. The cost is greater than lost time; it is the cost of lost answers.

Something like a dimensional or star schema implemented for acceptable query performance on traditional databases imposes assumptions and inhibits free-form discovery. Performance aids like indexes and projections speed access only to where you've predetermined the answers lie. Neither technique can optimize truly ad hoc queries.

A specialized analytic platform can provide great performance on this load and go basis avoids delays when an urgent problem hits. More important, it avoids the kind of paralysis in perpetuity that reigns when DBAs must continually alter designs in an attempt to boost performance to make your latest line of inquiry feasible. The personnel who can perform this optimization are a scarce resource to begin with and have their hands full with existing EDW work. Today's escalating

need for analysis makes it difficult for them to allocate enough time. Thus the answer to your problem escapes. The opportunity is not seized. The query never returns unless the systems is capable of a simple load and go.

2. Performance Without Limits

Optimal Massively Parallel Processing (MPP). Massively parallel processing (MPP) provides a good foundation for analytic databases. They are built from inexpensive commodity servers connected by industry-standard Ethernet connections and scale linearly with each additional server. Massively parallel analytic databases divide data into subsets for even distribution across multiple compute nodes connected by one or more Ethernet interconnects, allowing queries to execute with many units of parallelism to speed execution. If the number of nodes is increased, the number of data subsets distributed is increased.

The best analytic databases leverage multiple distribution methods. Often the optimal approach is a hashing algorithm applied to a primary key or other popular join column that provides even distribution as well as the efficiency of collocation of data that needs to be joined at each compute node. If an inherently balanced distribution key isn't available, round robin distribution can be used to balance the data.

A built-in analytic optimizer should minimize transfer of intermediate result sets across the MPP interconnect because interconnects can quickly become a bottleneck. Beyond that logical efficiency, look for optimization of the protocol used on the interconnect for shipping subsets in a many-to-many scenario. Avoid fabrics that use ordinary TCP/IP, which can bottleneck with packet loss in the 95 percent range. Currently the price-performance leader for interconnects is Gigabit Ethernet with an appropriate custom protocol that handles simultaneous many-to-many transfers while minimizing collisions. The database should be able to upgrade to higher bandwidths and leverage multiple interconnects for both high availability and scalability.

I/O to disk can also be a bottleneck on each compute node, so look for fast direct-attached storage (DAS) onboard each server, which greatly improves performance as opposed to the shared access of external storage such as SANs. Beyond this contention issue, algorithms at the lowest level of SANs are optimized for the random reads and writes of OLTP databases rather than for the high performance sequential scans required by analytical databases. In some virtualized and cloud environments, servers do not have DAS and must depend solely on SAN data, and you should consider the resulting impact on performance. SAN systems however do offer a flexible and mature approach for data management and replication. For instance for implementing a Disaster Recovery capability using known tools, while also supporting additional test and development through SAN-based cloning of data.

Columnar Storage. Look for analytic databases that store and read data in column fashion to minimize the amount of data read in typical analytic queries. This is especially true when you have very wide datasets in one or more tables. For instance a fact table with a billion rows and 100 attributes (columns) of data in each record.

Traditional databases are optimized for reading and writing individual full records of data (for example, a customer account record) and therefore store data row-wise. For most analysis, answers involve only the data in a small number of columns or fields. Functions like scans, joins, and aggregations that dominate analytic processing are inefficient for standard row-wise databases, which must read an entire row to use any of its fields. When a business analyst is determining the number of customers with a certain characteristic in common, row-wise storage requires a "full-width table scan" (all columns and all rows must be read). Reading a 100-field row to get one field value is only 1 percent efficient. Obviously, the 100 percent efficiency of reading only the one needed field column makes more sense.

Indexes provide a similar performance boost for particular queries, but add overhead and increase database size while helping only that particular class of query.

Performance acceleration from a columnar approach varies, but it closely parallels the ratio of the number of columns in the schema to the average number of columns that participate in the queries.

Compression. Look for analytic databases that maximize compression algorithms. Obviously, compression reduces the amount of data to be stored, reducing storage cost and requirements. More importantly compression reduces the amount of data to be read, reducing cost even further by using surplus CPU resource to compress and decompress data. In general, columns of data often have a limited number of discrete entries repeated many times, making them much more compressible than rows, which provides another advantage for analytic databases that use a columnar approach. Adaptive compression that takes into account the type of data in each column can also deliver improved compression compared to strictly computational methods. Compression is also provides a significant boost to processing data in-memory as it can increase the scale of feasible data processed by 5-10X (in terms of raw data).

In-memory and Flash-based Processing. Retrieving data from memory rather than off mechanical disk drives as often as possible maximizes the advantage of memory's faster access times. Furthermore, memory-centric processing means that the disks are more available for the scan needs of other concurrent queries.

The lower data volumes from using both a columnar approach and compression clearly increases the percentage of data coming from memory (either RAM cache or Flash drives) vs. disk, and that altered cost of I/O must factor into determining the optimal query execution plan. Look for a database that decides at runtime whether to use a memory-based or disk-based algorithm, based on the hardware configuration and the behavior of the individual queries.

In addition to the data reduction of columns and compression, an analytic database that minimizes or eliminates the need for indices, materialized projections and aggregate summaries in a load and go approach may make it feasible to move totally to more expensive but much faster Flash (solid state disk) storage instead of mechanical disks, depending on your total data volume. Forrester's Boris Evelson estimates traditional database approaches increase system storage requirements by as much as 8-to-1 or more via the indices and other data redundancy workarounds they use to improve query performance.

Viewed from a price/performance perspective, the additional cost of Flash for the most I/O efficient analytic databases is often offset by increased performance. Fewer compute nodes are required for equivalent query return times. Remember too that the higher cost of purchasing Flash is mitigated by lower ongoing cost of ownership, since Flash access consumes less power than the constant seeking and spinning of mechanical drives. As with any solution that relies on flash-based disks, it's also important to understand how the data will be protected in the case of unexpected interruptions in power.

Compilation. Another feature that can provide tremendous performance benefits on many analytic queries is query compilation. Since most analytic processing is I/O bound (the bulk of a query's processing time is spent on data retrieval), CPUs are not kept fully busy. However, systems with all the architectural attributes already discussed can sometimes reduce I/O cost to the extent that the CPU becomes the next potential performance limitation. Thus compilation can dramatically increase performance of the system.

Without compilation, each node's CPU spends more time interpreting and reinterpreting sections of the access plan it executes repeatedly than actually doing anything, particularly when it comes to performing aggregations, which are so important in analytics.

Remember that traditional databases were designed to optimize OLTP activities where each operation involves a very small number of records. Consequently, there was no motivation to compile queries. Analytical databases, by their nature, process a massive number of rows for each operation. It makes more sense to figure out what to do once rather than a massive number of times.

Compilation thus adds a small up-front cost, but as a result, most queries run many times faster. Also look for analytic databases that reuse compilation segments to avoid overhead for follow-on queries that have similar characteristics. This becomes particularly impactful in the kind of iterative ad hoc exploration required to find the source of a business issue.

Loading. There is a myth that columnar storage slows database loads because data comes in a record at a time and must be reorganized into columns for physical storage. But a well-designed analytic database massively parallelizes the loading or unloading process. Each node performing loading uses a suitable key to determine where the incoming data belongs in the column order and transfers it to the appropriate node.

The many nodes performing loading also split other preparatory work including parsing raw character-based data, reformatting data to write to disk, converting numeric character strings to binary, and allocating space. Finally, analytic databases that eschew the summaries, indexes and materialized views of manually tuned environments already have the advantage of less to load.

Consequently, look for an analytic database that can satisfy the desired loading performance simply by assigning the appropriate number of nodes to the loading process.

Optimizing Joins. The simplified schemas, summarizations and other performance optimizations used by traditional databases are designed mostly to avoid the need for joins. Without them, joins multiply.

One of the reasons traditional databases and indeed many analytic databases fall down in actual business scenarios is their inability to be efficient about complex join processing.

An analytic database needs to include a high-performance optimizer, which creates the plan for how to perform joins, needs to make good algorithmic choices. A good discussion of such optimizations is available in this SIGMOD paper.¹ One of the reasons data warehouses avoid the normalized model usual for production databases is that they are composed of many tables, which means many expensive joins. Even most analytic databases tend to be join adverse. Join Ordering. Join order is critical in producing good plans in general, as the difference in efficiency of a single example illustrates. Performing a join of the line item table with orders before customer instead of joining the customer and order table first results in a million-fold increase in the first set of intermediate results. While this example is extreme, smaller, much less obvious mistakes can be magnified many time in a chain of joins.

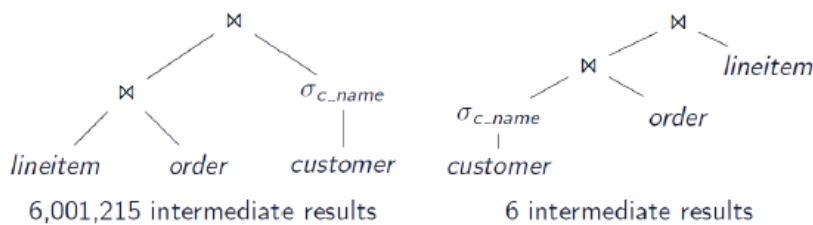


Figure 1. Optimal join ordering in a many-way join is one important characteristic of an analytic database that can handle real-world analytic applications that can require 100 table joins

¹ "Partial Join Order Optimization in the ParAccel Analytic Database," Yijou Chen, Richard L. Cole, William J. McKenna, Sergei Perfilov, Aman Sinha, Eugene Szedenits Jr., Proceedings of the 35th SIGMOD International Conference on Management of Data, 2009.

Choosing a good join order is particularly crucial for processing the large number of joins typically found in business analytics queries. Without an altered schema to reduce the number of tables, initial ordering mistakes are multiplied many times. Even with a star schema that reduces joins to a minimum for most business reporting, analytic questions that cross subject areas (for example, looking at the impact marketing has on sales) definitely involve more tables. Market basket analysis often has to perform many self joins to find affinities in the purchase of various products. And while the tables contributing to a view can be ignored when ordering joins, better plans are possible if the optimizer adds in the view's member tables.

An exhaustive search of all possible join orders is extremely resource intensive, causing otherwise solid optimizers to burn minutes of CPU time when attempting to order as few as 10 to 15 tables. Yet organizations performing business analytics on detail-level data routinely sees queries that exceed 100 tables. Join order heuristics like those patented by ParAccel² are required to cope with the problem.

The analytic database's communication interconnect protocol also comes into play. A properly architected analytic solution can rely on the interconnect to quickly distribute or broadcast intermediate join data at query time. Without a robust communication interconnect, even a parallelized shared nothing platform must rely instead on extensive data replication to avoid a painfully slow broadcast or redistribution between nodes. Such data replication should be a red flag.

Handling Correlated Subqueries. A correlated subquery's embedded query depends on the outer query for its values. In effect, that means you suffer the inefficiency of an inner loop. The subquery must be executed once for each value selected by the outer query. Correlated Sub-Queries (CSQs) are a good way to approach many business and analytic problems and thus are a common occurrence in complex analytic environments, but they are especially challenging because data movement must occur after each inner query executes. Look for analytic database optimizers that replace CSQs with joins and aggregations of intermediate result sets. Such an optimizer will have no problem with the many business analytics application packages that tend to emit correlated subqueries to the analytic database. It is not acceptable for a database to place the burden on application developers to manually de-correlate, which can be time consuming and error prone.

Handling Complex SQL. In terms of practical business benefits that may not show up in a narrow benchmark but will in a real-world analytics environment, the most important criteria is to be sure the optimizer of the analytic database you choose handles complex SQL well. Test, for example, whether the analytic database you're testing can aggregate multiple scalar correlated subqueries. Make sure it runs well without materialized views. Test whether it can perform a fact-to-fact join. Test outer join support. Test challenging queries that involve many-way joins of a realistically large number of tables pulled in from a production database, and not just what the smaller number of joins it will face in a performance optimized data warehouse schema. Test ad hoc queries without allowing the vendor to optimize for them beforehand.

² IBID

Performance Bottom Line. "Performance and price/performance measures for analytic databases in TPC-H benchmarks and in quoted customer examples over the past few years, show gains of at least 2X to 10X, and in many cases considerably more, over more general-purpose databases." says database expert Barry Devlin of 9sight Consulting. After studying the impact of the kind of architectural features just discussed, he concluded: "While any decision to invest in an analytic DBMS must, of course, be based on wider criteria and include comparative proofs of concept, these figures are impressive."³ See Figure 2 for his take on the cumulative performance impact based on actual customer use of ParAccel's PADB analytic database.

Technique	Query Type	Acceleration
Columnar Storage	All	Ratio of columns in query to columns in row
Compression	All	4x to 12x depending on data
Compilation	Aggregation Dimensional Join Complex arithmetic, case statements, restrictions	20x 10x 5x to 20x
Adding Nodes in an MPP configuration	All	Linear (or better if system was under-specified and writing intermediate results out to disk)

Figure 2. Scope of performance acceleration achieved by ParAccel's PADB analytic database from architectural techniques in actual customer scenarios.

3. Enterprise Strength

Many analytic databases can grow to become mission critical. For example, a number of early adopters feed a variety of data to inline "recommendation engines" that help optimize each customer's experience on a website or when talking to call center agents by producing "next best offers." The recommendations variously reflect an analysis of click-stream behavior, similar customers' transaction histories, demographic profiles, and recent overall purchase trends.

Another example is prediction of required car production levels based on searches and click patterns on sites consumers visit to research car purchases, which is used for accurate just-in-time part delivery to car manufacturing plants. These uses of analytics have become operational. The business processes depend on them being available.

As such, analytic databases also require high availability and rapid disaster recovery approaching the level of traditional (OLTP) enterprise databases. The following are thus important technical considerations as you choose the right analytic database for your business.

Data Availability. One level of availability deals with single disk failures. When time is truly important in finding an answer to a business problem or when production depends on the outcome of an analysis, you don't want the failure of a single disk on one of the nodes in your MPP configuration to force the restart of a long-running query. One solution is for each node of an analytic database to mirror its disk image to the direct attached storage of another compute node. Should a disk fail, the sibling node provides the mirror data across the MPP interconnect with only minor impact on overall performance until a spare disk can be replaced and repopulated.

RAID technology can provide an additional level of higher availability, but more disks are required to achieve the same level of performance as the mirror/failover approach.

Handling Node Failures. An important criterion in enterprise-class MPP systems is the ability to meet Service Level Agreements for performance even after an entire node fails. Look for analytic databases that offer one or more hot standby nodes (HSN). Should a compute node fail, the HSN can take over using the mirrored data image on the failed

³ 9Sight Consulting "Analytic Databases in the World of the Data Warehouse," Barry Devlin, April 2009

node's sibling. The hit to overall query performance is scarcely detectable. When the failed node is replaced, it becomes an HSN.

SAN's for Data Protection and Incremental Speed. SANs are the usual solution for mission critical data management requirements like backup and offsite replication for disaster recovery. But in an MPP-based analytic database environment, SANs are less than optimal due to the limits of their channel throughput. Each MPP compute node should use direct-attached storage (DAS) because it outperforms the shared bandwidth available from the SAN. Unfortunately, the many islands of storage that result present an untenable backup and disaster recovery challenge.

The solution is to look for an analytic database that performs parallel writes to the compute node's DAS and to a SAN, with a commit only when the write to the SAN is complete. As a result, the data of record on the SAN matches the data being analyzed on each compute node's DAS. The normal SAN backup and replication capabilities can provide a quick restore at a remote site in the event of a local site outage.

Further, while you should expect any analytic database to depend on local DAS (mechanical or Flash) for the bulk of its high-speed I/O, consider approaches such as ParAccel's blended scan. Blended scan reads from idle SAN spindles as well as from DAS to optimize performance without impacting the fault tolerance provided by the SAN.

Mixed Workloads. Analytic databases need to handle continuous updates of the data used in operational analytics, which are often referred to as trickle updates. In some scenarios, the analytic database also balances concurrent users against the appropriate level of resources dedicated to both reporting and ad hoc modes.

Clearly, parallel analytic systems must balance resource utilization while appropriately favoring the primary mission of query throughput. The best implementations eliminate operating system overhead and maximize bandwidth usage in all forms of reading, caching, and moving data while at the same time writing data to disk with maximum efficiency. Look for solutions that reduce potential bottlenecks in all of these areas by interleaving movement of data and processing of analytic operations.

4. Non-SQL Analytics

The perceived and real limits of declarative SQL have resulted in offloading of much quantitative analysis to an application server separate from the database using programming languages such as Java, C#, Python, C++, as well as other open-source statistics environments. Given the volume of data many businesses need to explore, this architecture is untenable. Transferring large data volumes between a database and the application-tier for analytics does not leverage MPP and may not scale for your application as they rely on traditional SMP-based environments.

Your organization may also have some stored procedures that get around the declarative limits of SQL. You might like to be able to run them unchanged on a new high-performance analytic database. Unfortunately, some stored procedures cannot leverage the parallel power of analytic databases if they replace SQL rather than generating SQL. Verify that the vendor directly supports stored procedures.

Another extension mechanism for SQL is the User Defined Function, or UDF. Though useful, some UDF implementations are difficult to use because they make you rewrite the UDF for different input and output schemas. If you depend on UDFs or are thinking they are important to your future use of an analytic database, be sure to understand each vendor's current capabilities and forthcoming plans. A simple example of where this would be needed is a simple transpose function. Ideally you would want to write/support only one function, regardless of whether the inputs/outputs are in a 4x4, 10x10, or 20x20 format.

In some cases, you may think your business needs analytic capabilities that extend beyond what can be expressed in SQL alone. Just to be sure, check with the vendors of the analytic offerings you're considering to see whether a SQL business intelligence tool or hand-crafted SQL approach can handle the problem efficiently.

Some analytic database vendors have implemented the popular emerging platform for massively parallel computing, MapReduce, in an SQL wrapper, influenced perhaps by its successful use at Google and availability on the Amazon cloud via Hadoop. The fact that a growing number of programmers know and use the open source parallelization environment may seem to make it attractive, but consider the cost benefits of the declarative power of SQL alongside procedural languages such as Python and Java. The planning, debugging and maintenance of a procedural language project may be higher, and the functionality more fixed-function than familiar SQL-based approaches.

Also consider a recent SIGMOD paper that described a MapReduce versus SQL benchmark⁴. The work was performed on an open source version of MR over a cluster of 100 nodes, then by analytic SQL on MPP databases in the same environment. The results showed the DBMS performance to be strikingly better across a number of tasks usually cited as the domain of MapReduce, including a great deal of click-stream aggregation and a text processing task.

Additionally, be aware that MapReduce environments such as Hadoop are often roll-your-own-solutions. The programmer becomes responsible for load balancing, data distribution, join algorithms, data type conversions, and so on. This is functionality handled automatically and extremely adeptly by a sophisticated MPP analytical database. Why make programmers essentially reinvent MPP database technology? Why limit your ability to ask ad hoc and iterative questions? Look for a database that builds upon the power of SQL and allows for integration with data and analytics on MapReduce where appropriate and needed.

5. Rapid, Elastic Scaling

Users are feeling the need for analytic databases that offer rapid, seamless and elastic scaling from a few nodes to thousands. Some vendors are touting how simple it is to instantiate new instances of the database using virtualization or cloud hosted facilities. Not only can you scale up to handle rapid understanding of a business problem, they claim, you can adjust scale down as the problem recedes.

Examine those claims with care and ask questions to separate hype from reality. If analytic databases require use of direct attached storage (DAS) for true high performance, how will that work (or what does it even mean) in a fully virtualized environment? Does the vendor have the required relationship with the virtualization vendor to work out the underlying intricacies? Look for benchmarks that show the analytic database you are considering makes best use of the virtualization environment.

Scaling from 5 to 25 nodes requires a redistribution of data to give each compute node the appropriate slice of the data to operate on. Does that require a pause to, in essence, reload the data? Whatever you do, carefully consider the implications of scaling before assuming it is seamless. Also ensure that your scaling "horizon" extends across two or more years, and takes into account the path for supporting datasets that are between 2-5X larger than your current environment.

⁴ "A Comparison of Approaches to Large-Scale Data Analysis," Andrew Pavlo, Erik Paulson, Alexander Rasin, Michael Stonebraker et al., Proceedings of the 2009 ACM SIGMOD International Conference (June 2009)

Comparing Analytic Databases and Analytic Platforms

Up until this point, the discussion has focused on criteria for selecting an analytic database. With the maturing of the market for analytic databases, your selection criteria should also include items that create a platform for new analytics, well beyond what you would expect in a database. This move from database to platform includes new capabilities such as, the ability to easily bring in data from many different sources and quickly assimilate new functionality critical to handling different kinds of analytic workloads. This kind of extensibility is critical to the process of rapidly delivering new analytic applications for the enterprise and spreading the use of analytics across entire organizations.

6. Rapid Data and Analytic Integration

There is a constant influx of new data and external analytics available for the analytic process. The ability to integrate new and different data is critical for maintaining a competitive advantage. Leveraging analytic work done by external applications will also speed the spread of analytics.

The process for integrating data into a traditional data warehouse environment can be long, and complex, with major steps being taken around extracting, cleansing, transforming, and loading the data in the database. The complexity of these processes do not lend to the speed needed by organizations today in order to respond to changing market needs. In addition, because many of the new analytic databases appeared only in the last 5-7 years, they have not invested in enterprise strength integration capabilities. Look for a database that can quickly integrate new sources of data into their analytic framework. It's this kind of extensibility that sets certain vendors apart as platforms, rather than just databases.

It is also time consuming and resource intensive to move massive amounts of data from other platforms into an analytic environment. Where possible, platforms should be able to leverage external analytic processing being done by platforms like Hadoop and other vendors.

Analytic databases should be built in an open environment that allows for simple and diverse integration. Integration modules should be able to quickly access and acquire structured data from the data warehouse and other outside sources. In addition, look for databases that have worked with outside vendors like Hadoop for access to semi-structured and unstructured data. There should be some kind of deeper integration work done with leading vendors to leverage their technology in the data acquisition process, as well.

Beyond these capabilities, it should be easy to create new modules for rapid integration as the need for new data continues to change. Begin to think of the kinds of data you will need to access in the coming future, such as email archives, social media content, ticker data, transaction data, sensors, monitors, and other sources. The more mature vendors will have a roadmap and be able to articulate the effort involved in acquiring these different kinds of data.

Also, because the analytic process often includes an iterative approach, you don't want to always bring the data into the analytic database ahead of time. The ad hoc discovery process used by analyst requires the ability to bring data into the analytic platform during a query, not just before.

7. Advanced Function Assimilation

The analytic process also requires the use of advanced functions and highly specialized or customized algorithms. The need to fully leverage time consuming work already done and algorithms already in the marketplace is important to

expanding the use of analytics. Look for a library approach that allows for the cataloguing of existing and new advanced functions. In addition, the analytic functions should run within the structure of the database for maximum performance. Simple advanced function assimilation is the second kind of extensibility that separated platform vendors from analytic database vendors.

Analytic databases should be built in an open environment that allows for simple and diverse assimilation. New advanced functions should be easily stored in a library and simply available for reuse. In addition, look for databases that have already imported algorithms from vendors like Fuzzy Logic. Leading companies will want to build their own intellectual property over time by developing their own set of algorithms, uniquely designed to drive their business.

Satisfying Your Need to Know

Analytic databases tend to address specific analytic applications, usually with large volumes of (reasonably) consistent source data. One of the beauties of this approach is that it doesn't force the strict EDW data cleansing standards on projects that are using disposable data. Such effort could be wasted or even harmful because irregularities are often the source of useful information. Bottom line, analytic databases need the appropriate level of governance. They will deliver a high return on investment (ROI) by significantly speeding up query response times, by enabling previously-impossible analyses, and by allowing a new way of analytic thinking. In addition, the more mature analytic platforms will speed the use of analytics across organizations for greater business impact and accelerate the speed of analytics.

Historically, best practices often suggested analytic platforms should migrate to an interdependent position in the EDW architecture—that is, much of their raw data will be fed from the EDW. One effective use of this approach is to offload high-performance analytic workloads off of EDWs, and migrate them to analytic platforms that can handle the escalating analytic requirements emerging today. This enables not only better overall performance across all workloads, but enables companies to extend the working life of their existing EDW.

Another use-case is for the analytic database to take on the data conditioning and transformation performed in the EDW. In effect, the analytic database can provide master data management. Much of the ETL or conditioning process involves the type of look-ups, joins and re-mapping for which analytical databases have been optimized. For example, Merkle, a leading database marketing agency, uses ParAccel to integrate and consolidate many terabytes of highly redundant consumer name and address data in a process considerably more efficient than their previous use of ETL tools.

But supporting a complete data warehouse environment is challenging. The reward of reducing latency for operational business intelligence needs and driving down development and maintenance effort for IT could be a huge win. Before moving forward, you need to carefully analyze whether analytic platforms (including the underlying core databases) have sufficiently optimized database writes, ACID considerations, and workload management.

Also consider the business use case. An analytic platform goes beyond the capabilities of a database to fuel the expansion of analytics across an entire company. Formerly constrained by technology, a platform allows you to think about what business opportunities would benefit most from new analytics. With a lower cost of entry and quicker time to production, a platform opens new discussion about where to go next with your analytics.

Whatever you do, don't let the larger goal of rethinking the venerable institution of the EDW get in the way of handling your business's urgent and intense need to know. The right choice of analytic platform can scale up your ad hoc analysis capabilities quickly and help you meet and beat the competition.