Hybrid real-time data processing

The market

Various analyst firms, and others, have coined a multitude of acronyms for the subject under discussion. These include, but are not necessarily limited to, HTAP (hybrid transactional analytic processing), Translytics, HOAP (hybrid operational analytic processing) and OLTAP (on-line transactional analytic processing). Bloor Research does not intend to add to this list of acronyms. The truth is: we do not care what you call it. We will discuss this further in due course.

There are multiple ways in which databases that support real-time operational/transactional and analytic processing may be implemented. These include:

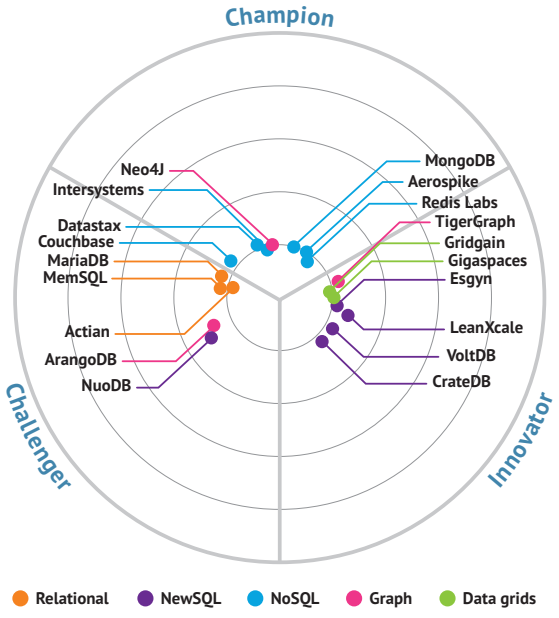
- **Relational databases.** All of the leading database vendors (Oracle, IBM, SAP, Microsoft and others) offer hybrid systems. For historical reasons, row-based data is stored on disk and columnar data is loaded into memory. Conversely, there are newer vendors on the market that have re-thought this and store data in columnar fashion on disk and use row storage in memory.
- **NewSQL databases.** These are products that look and feel like relational databases, using SQL but, under the covers, do not use relational tables to store the data. The contention is that the storage mechanisms used, often combined with distributed in-memory processing, are suitable to meet all the requirements of a hybrid system. SQL on Hadoop products would also fit into this category.
- **NoSQL databases.** There are a number of products in this space though some may only be appropriate for operational but not transactional environments because of the lack of ACID guarantees. Note that some *"NoSQL"* databases actually pre-date the concept of NoSQL.
- **Graph and RDF databases.** These will almost always be appropriate for use in hybrid environments though, again, with the proviso that some products may not be ACID compliant.
- **In memory data grids.** These typically work in conjunction with either relational (third party) or NoSQL (third party or provided) databases. In effect, they can turn a relational database or key value store (say) that does not natively support hybrid capability into one that does.

There are a great many potential solutions and we have had to make decisions about what and what not to include. We have opted not to include any of the 800lb gorillas, firstly because they are very well known, secondly because they are expensive and, thirdly, because we cannot imagine anyone adopting one of these solutions unless they were already a user of said database. Other than that, we have included a representative sample of suppliers from each of the other categories of hybrid suppliers. Vendors included in this evaluation are:

- **Relational:** Actian, MariaDB, MemSQL
- **NewSQL:** CrateDB, LeanXcale, NuoDB and VoltDB; Esgyn (SQL on Hadoop)
- **NoSQL:** Aerospike, Couchbase, DataStax, MongoDB, Redis Labs and InterSystems IRIS (pre-NoSQL)
- **Graph:** ArangoDB, Neo4j, TigerGraph
- **Data grids:** GigaSpaces, GridGain

In order to ensure apples to apples comparisons the various products in each of these groupings are colour coded in the Bullseye diagram in Figure 1.

Figure 1: The highest scoring companies are nearest the centre. The analyst then defines a benchmark score for a domain leading company from their overall ratings and all those above that are in the champions segment. Those that remain are placed in the Innovator or Challenger segments, depending on their innovation score. The exact position in each segment is calculated based on their combined innovation and overall score. It is important to note that colour coded products have been scored relative to other products with the same colour coding.



However, there are other ways to skin this particular cat. You could, for example, compare products by the extent to which they support SQL: ANSI standard SQL, quasi-SQL languages or something completely different. Or you could differentiate on the extent to which multi-model support is provided. A third alternative would be to distinguish between products based on their approach to persistence, ranging from none (data grids) to once (graphs and others) to NoETL architectures where you might have separate clusters for operational/transactional processing versus analytic processing, as well as, possibly, hybrid clusters. More interesting than any of these are the use cases that different products are targeted at. These include:

- Replacements for existing databases (usually relational, possibly in conjunction with a second, typically NoSQL) database, usually because the existing environment is transactional but does not support the required analytics.
- Extensions to existing databases, providing the hybrid capabilities that the existing system does not have but without requiring any *"rip and replace"*.
- Greenfield deployments.

Data grids typically fulfill the second of these roles while other product types address the replacement and greenfield opportunities.

Finally there is the issue of whether particular products are most suitable for operational versus truly transactional requirements, whether they support concurrent and /or in-process analytics (a subject we discuss later). The latter is particularly significant as there are some products that have a paucity of support for embedding machine learning and other advanced analytics capabilities.

The concept

The first of the acronyms on this subject to be promulgated, by the Gartner Group in 2013, was HTAP and the company has since refined this concept into what it calls *"Point-of-Decision HTAP"* and *"In-Process HTAP"* where the former is *"an architecture using in-memory computing techniques and technologies to enable concurrent analytical and transaction processing on the same in-memory data store"* and the latter is *"an application architecture whereby, in the context of a given application, analytical and transaction processing techniques are weaved together as needed to accomplish the business task."* In practice the former commonly refers to environments where you are generating real-time dashboards while in the latter you might have a machine learning algorithm embedded within a real-time application.

Where Bloor Research takes issue with the Gartner Group's definition is on the insistence on in-memory capabilities. Just as we don't care what you call hybrid application environments, we also don't care how you get the performance and scale needed to support those deployments. If you could get these without making use of memory that would be fine with us. But, of course, databases have always leveraged memory in one way or another, so to make it a requirement is – in our view – unnecessary.

The characteristics of hybrid systems

First and foremost we are talking about databases and data processing. This is assumed, but not mentioned, in any of the existing acronyms mentioned (though, to be fair to Gartner, its definitions refer to architectures rather than databases). Secondly, hybrid systems are hybrid in the sense that they support both operational/transactional and analytic use cases. However, these terms need to be clarified.

In the case of operations we regard these as a superset of transactions. OLTP (online transaction processing) was coined as a term when the main concern of database processing was the processing of transactions between parties, typically in the form of orders: sales orders, orders placed on suppliers, works orders and so on. In order to ensure accuracy when updating database tables with these orders, transactions require support for ACID (atomicity, consistency, isolation and durability) properties, along with two-phase commit, to ensure the integrity and validity of transactions in the event of an error or failure.

While transactions of this type remain the lifeblood of many organisations there are also a class of operations, for example sensor readings

in an Internet of Things (IoT) environment, or call detail records in telecommunications, or stock ticks in capital markets, where information needs to be processed in a similar timescale to transaction processing (in other words, in as close to real-time as you can get) but which do not necessarily require the same stringency with respect to ACID and two-phase commit. Thus, for example, a NoSQL database that supports eventual but not immediate consistency might – probably would – be more suitable for some operational use cases but not those that are explicitly concerned with transactions. We will discuss the issue of consistency in more detail in the next section.

With respect to analytics, we agree with the Gartner Group that there is a valid distinction between concurrent versus in-process capabilities, but we must also make a distinction between real-time analytics and batch-based analytics. Some of the solutions evaluated in this report do only the former, while some do both. We do not regard the latter as a requirement for inclusion in this report, though it may be a nice-to-have. Note that this is not the same as having a persistent data store. This is often necessary where real-time data needs to be combined with historic data for analysis purposes.

Finally, we should add that a number of vendors have pointed out to us that they see search as an intrinsic part of hybrid environments, in addition to operational/transactional processing and analytics. We are inclined to agree with this perspective, which would invalidate all the acronyms mentioned previously.

Consistency

Consistency is a fraught issue when it comes to discussing distributed databases. There are two things to consider: ACID (atomicity, consistency, isolation and durability) and the CAP (consistency, availability and partition tolerance) theorem. The former provides guarantees about the processing of transactions within a database and the latter, also known as Brewer's theorem, states that in a distributed data store you cannot guarantee all elements of CAP and, in particular, you can have availability guarantees or consistency guarantees but not both. Note that this isn't an issue when the cluster is running normally. Also note that this theorem has been proved (which is why it is called a theorem).

One difficulty that arises is that the *"C"* for consistency in ACID is not the same as the *"consistency"* in the CAP theorem, which actually stands for *"atomic consistency"*. As the authors of the CAP theorem state: *"discussing atomic consistency is somewhat different than talking about an ACID database, as database consistency refers to transactions, while atomic consistency refers only to a property of a single request/response operation sequence. And it has a different meaning than the Atomic in ACID, as it subsumes the database notions of both Atomic and Consistent."*

On top of this confusion of terms, there are multiple types of consistency (à la CAP) supported by different vendors. There is strong consistency, weak consistency, immediate consistency, eventual consistency, strong eventual consistency, causal consistency and a whole bunch of other consistency models. In addition, you will hear vendors referring to linearizability, which provides guarantees about single operations performed on single objects and is related to the CAP Theorem; and serializability, which refers to guarantees about transactions and is ACID related. There is also a concept known as strict serializability. Note that some vendors offer tuneable consistency, by which they mean that you can choose different consistency models depending on requirements, including having different models in place within the same cluster.

As far this report is concerned, we do not intend to discuss this topic in any detail. We do, however, want to point out that this confusion in terminology can make it easy for vendors to pull the wool over potential user's eyes: consistency, as should be clear, can mean different things to different people, and it is important that you and your provider are using the same terminology with the same meaning. In this context, a useful resource is ***https://jepsen.io/analyses*** (see below).

One final remark worth making is that some commentators on this market seem to think that support for eventual consistency should be a requirement for the sort of hybrid environments we are discussing. We do not agree with this view. There are some use cases that require immediate consistency and there are some that don't. There are some applications that require full-time availability and there are some that can stand a few seconds of down-time. It depends. There are certainly applications that do not require immediate consistency, or even ACID guarantees for that matter, so we do not take a position on which is preferable: it will depend on the use case, but we do not believe that either approach should be mandated. As a general principle we would argue that support for eventual consistency is a good idea when it comes to operational requirements, because you can guarantee high availability, but that immediate consistency is required for transactional environments.

Market trends

We have already mentioned, or alluded to, many of these. However, perhaps the most obvious trend is that more and more database (and data grid) providers are targeting at least a part of this market. Companies that previously only offered operational capabilities, as an example, are now offering analytics as well. Nor do we expect this trend to diminish. In particular, as 5G becomes prevalent, the demands for real-time processing will, if anything, accelerate.

Other trends that relate to this market are increased multi-model capabilities, especially with support for both JSON documents and graph capabilities; more tuneable consistency; increasing implementation of resource management to support mixed workloads; and more support for both synchronous and asynchronous replication. For data grids, as opposed to databases, there is a notable trend towards adding persistence capabilities, though this is part of a wider trend that also applies to streaming products such as Kafka and Flink.

Most, but not all products, support Kafka for streaming data and a number of vendors already do or plan to support Flink. With respect to support for machine learning and AI, Spark is commonplace as is support for Python and R though TensorFlow, surprisingly, is supported by very few vendors. PMML (predictive modelling mark-up language) has growing support but remains a minority interest.

Proof points

Some vendors are still wedded to performance and scalability statistics and benchmarks. In our view, these have very little value. More or less everybody in this market can reasonably claim to be able to process millions of transactions/operations with latency measured in milliseconds. So this is not a distinguishing feature. What is a differentiator for some vendors is how much resource you need to be able to achieve comparable performance? Here there are some quite startling differences, with some suppliers requiring only a fraction of the cluster size that other providers might require. Of course, any such claims should be tested via proofs of concept. As all the vendors in this report have cloud-based offerings, many of them via managed services, this should be easy to test.

We should also mention ***https://jepsen.io/analyses*** to which we have previously alluded. To quote the company *"Jepsen analyses the safety properties of distributed systems – most notably, identifying violations of consistency models."* The web site not only provides a useful guide to consistency models in general, but also presents the results of every database it has ever tested, including a number that are included in this report. Amongst other things it has found *"replica divergence, data loss, stale reads, read skew, lock conflicts, and much more"*. Unfortunately, participation is voluntary, and a number of the tests are out-of-date. Nevertheless it may prove a useful resource to readers.

Metrics

We have scored products across eight metrics, bearing in mind that data grids, for example, have different requirements from, say, graph databases. The eight metrics are as follows:

- **Concurrent analytics** – the extent to which the offering supports the creation of real-time dashboards and analytics, its support for real-time alerts, and the extent to which full-text search is provided. Tools for developing dashboards will be advantageous as will be support for federated query into other database environments. The ability to combine real-time and historic data for analytic purposes may be necessary. Workload management is a relevant factor in ensuring appropriate performance.
- **In-process analytics** – includes many of the same requirements as for concurrent analytics but specific issues would include whether models can be trained within the platform or only outside of it, the extent of support for deploying models built in Java, Python, Scala, R and so forth, as well as import via PMML. Pre-built algorithms (common for graph databases) will be an advantage.
- **Performance** – this covers not just run-time performance for both operations/transactions and analytics but also ingestion rates. Support for third party streaming environments such as Spark, Kafka and Flink will be important. The capabilities of the database optimiser, as well as workload management, will be relevant. While in-memory capabilities are universal, specific support for SSDs and persistent memory (Intel Optane), alongside spill-to-disk functions, will be useful.
- **Scalability** – how easy is it to scale the solution? Is the database capable of handling tens of millions of events per second? Millions? Or hundreds of thousands? Features such as support for read replicas will help to scale the environment. Also relevant is the footprint of the solution: is it suitable for deploying in edge devices or gateways?

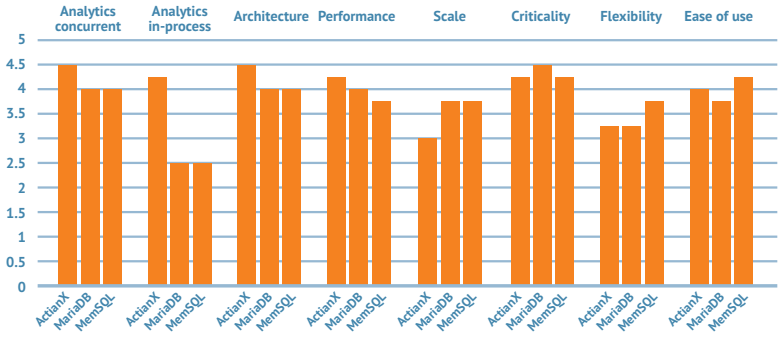
- **Architecture** – performance and scalability are all very well but perhaps more important is the efficiency of the architecture: how demanding is the solution in its use of hardware resources? If one solution requires a cluster that is five times the size of another solution that would not be very efficient (or cost-effective). Architecture also subsumes consistency considerations: the extent to which the product supports operational and/or transactional capabilities, where the former may benefit from eventual consistency, while the latter will require ACID compliance and immediate consistency. Where ACID compliance is provided: when is it provided? For example, there is a distinction between maintaining ACID guarantees within a JSON document and across JSON documents. It will be advantageous to offer tuneable consistency.
- **Criticality** – this criterion overlaps with architecture because the flip-side of consistency is availability and maintaining high availability is essential for mission critical applications. Support for both Active/Active and Active/Passive environments will be important. Other critical requirements include security, support for authentication and so forth. Some environments also have built in capabilities for data masking.
- **Flexibility** – some use cases relate strictly to structured data, but others may incorporate documents and other semi-structured data. To what extent does the product support multiple datatypes, either directly by offering a multi-model approach or indirectly by supporting a variety of datatypes? Flexibility also relates to language support for development purposes, including IDEs such as Eclipse.
- **Ease of Use** – should be self-explanatory: includes administrative tools, cluster monitoring, graphical visualisation capabilities and so forth. The query language supported is also relevant here: is it ANSI standard SQL, a quasi-SQL language or something completely different?

Scores

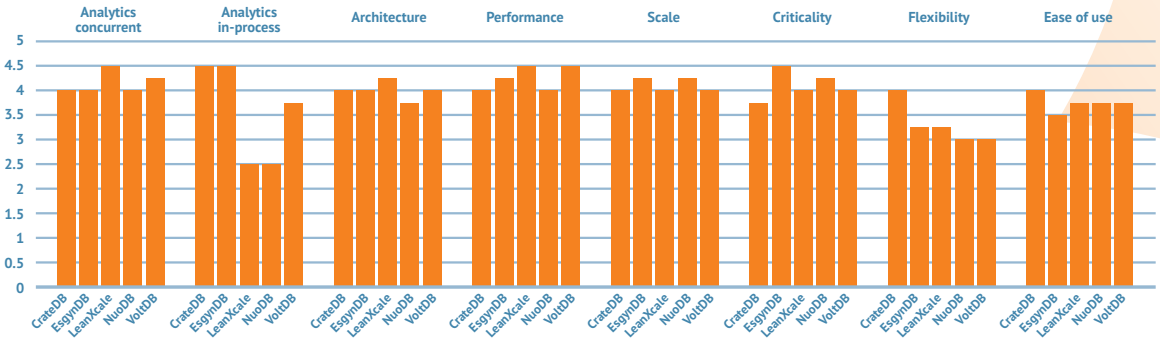
Some metrics may be more relevant to some readers than others, which is why we have separated them, and made care to retain the principle of comparing apples with apples. Note that these scores only reflect our opinion of the technical aspects of these products; the score rating in the Bullseye diagram includes other factors such as company stability, user base, worldwide geographical support and so on.

This section illustrates our star ratings for each of the metrics discussed previously, with companies/ products with equal ratings listed in alphabetic order.

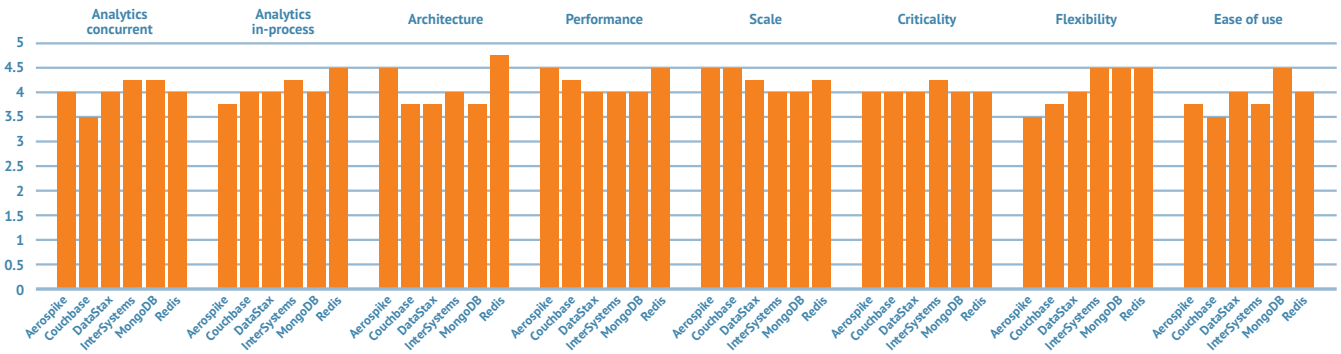
Relational databases



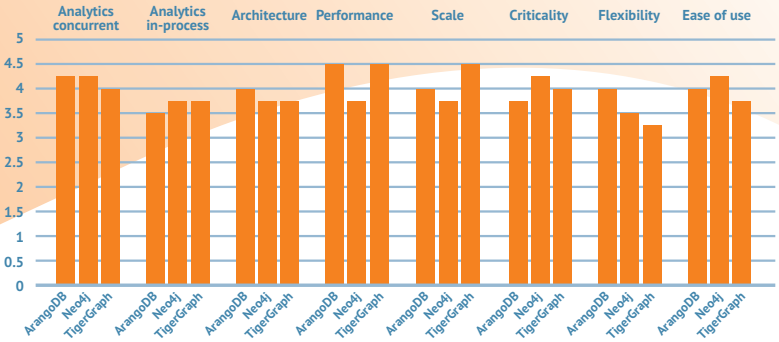
NewSQL databases



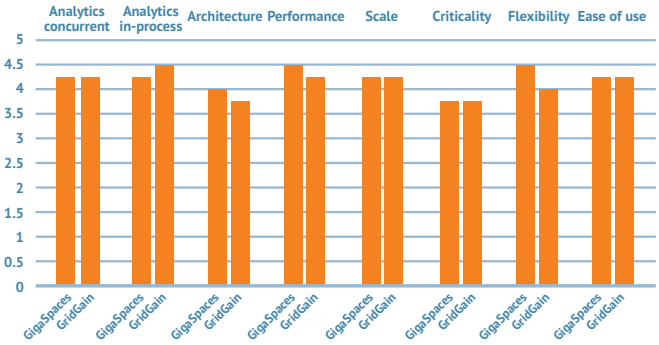
NoSQL databases



Graph databases



Data grids



Conclusion

There are multiple ways to combine analytics, concurrent and/or in-process, with the processing of operations and/or transactions within a single platform. Not only that, but there are literally hundreds of potential products that you might use for that purpose. We have here presented a representative sample of potential options. With only a couple of exceptions they all represent mature, if still evolving solutions; and even the exceptions have been years in development. It should therefore not come as a surprise that all the products we have reviewed offer genuine solutions, depending on the use case. We trust that this Market Update will help to inform the buying decisions of companies looking to adopt hybrid, real-time data processing environments.

About the author

PHILIP HOWARD
Research Director / Information Management

Philip started in the computer industry way back in 1973 and has variously worked as a systems analyst, programmer and salesperson, as well as in marketing and product management, for a variety of companies including GEC Marconi, GPT, Philips Data Systems, Raytheon and NCR.

After a quarter of a century of not being his own boss Philip set up his own company in 1992 and his first client was Bloor Research (then ButlerBloor), with Philip working for the company as an associate analyst. His relationship with Bloor Research has continued since that time and he is now Research Director, focused on Information Management.

Information management includes anything that refers to the management, movement, governance and storage of data, as well as access to and analysis of that data. It involves diverse technologies that include (but are not limited to) databases and data warehousing, data integration, data quality, master data management, data governance, data migration, metadata management, and data preparation and analytics.

In addition to the numerous reports Philip has written on behalf of Bloor Research, Philip also contributes regularly to ***IT-Director.com*** and ***IT-Analysis.com*** and was previously editor of both ***Application Development News*** and ***Operating System News*** on behalf of Cambridge Market Intelligence (CMI). He has also contributed to various magazines and written a number of reports published by companies such as CMI and The Financial Times. Philip speaks regularly at conferences and other events throughout Europe and North America.

Away from work, Philip's primary leisure activities are canal boats, skiing, playing Bridge (at which he is a Life Master), and dining out.

Bloor overview

Technology is enabling rapid business evolution. The opportunities are immense but if you do not adapt then you will not survive. So in the age of Mutable business Evolution is Essential to your success.

We'll show you the future and help you deliver it.

Bloor brings fresh technological thinking to help you navigate complex business situations, converting challenges into new opportunities for real growth, profitability and impact.

We provide actionable strategic insight through our innovative independent technology research, advisory and consulting services. We assist companies throughout their transformation journeys to stay relevant, bringing fresh thinking to complex business situations and turning challenges into new opportunities for real growth and profitability.

For over 25 years, Bloor has assisted companies to intelligently evolve: by embracing technology to adjust their strategies and achieve the best possible outcomes. At Bloor, we will help you challenge assumptions to consistently improve and succeed.

Copyright and disclaimer

Bloor Research International Ltd
20–22 Wenlock Road
LONDON N1 7GU
United Kingdom

Tel: **+44 (0)20 7043 9750**
Web: **www.Bloor.eu**
Email: **info@Bloor.eu**