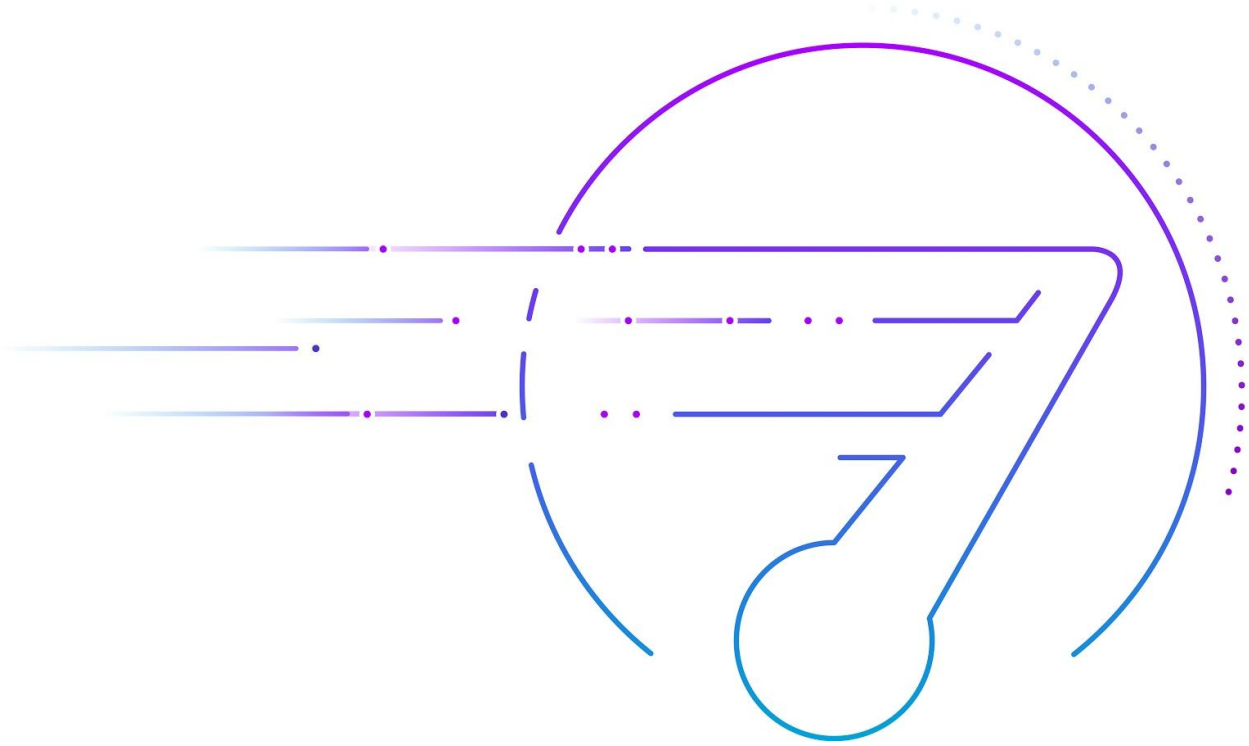
TECHNICAL WHITEPAPER

Real-Time Decision Making at Scale



SingleStore

2020

“

One of the most complex aspects of managing data for analytical workloads is moving it from a transactional database into the data warehouse. What if you didn’t have to do that at all?

– Nikita Shamgunov
Co-CEO, SingleStore

Table of Contents

Executive Summary

Traditional database architectures are inherently limited in their capabilities. Legacy relational databases can't be scaled easily and flexibly across pools of servers, instead being restricted to a single machine at the core of transaction processing or query operations.

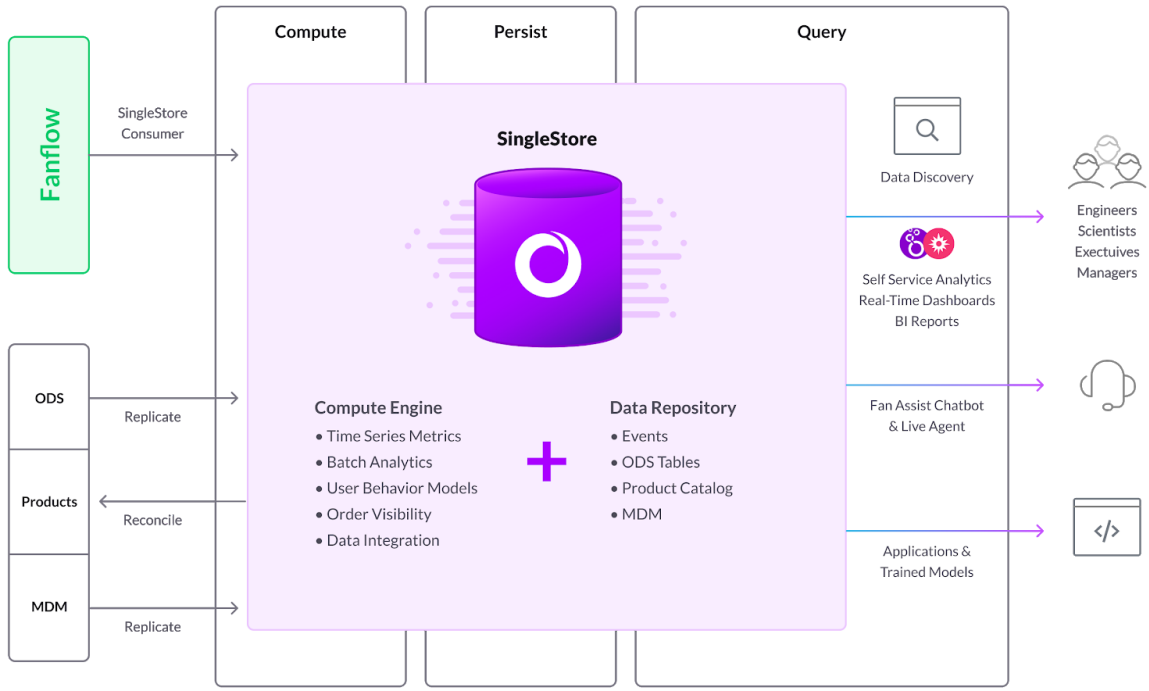
These limitations are so severe that transaction processing is run as a batch process, with the database unavailable or degraded for queries during processing. A complex extract, transform, and load process must then be run to ready data for analytics; and analytics, even against this stale data, are often also not available in real-time.

NoSQL databases overcome the scalability barrier, but are designed for unstructured or semi-structured data; they can't natively maintain consistency and proper join patterns for a structured data store, as required for efficient analytics. This makes NoSQL less than ideal to fully address the problems of legacy operational data for real-time or near-real-time decision making at scale.

One example of real-time decision making in action with SingleStore is Fanatics. The company is the leading vendor of team-branded sports apparel worldwide, with their own standalone site and a presence within hundreds of e-commerce portals and brick and mortar stores.

Fanatics combines a fast ingest system (powered by Kafka) and SingleStore at the core of its analytics solution to handle personalization, recommended purchases, loyalty programs, and more.

The new architecture deployed by Fanatics demonstrates how the combination of fast data ingest, fast transaction processing, and fast analytics response, with high concurrency, can provide real-time decision making and power data-enriched customer experiences, today and into the future.



Fanatics uses SingleStore to power real-time decision making

Introduction

Real-time decision making means using all of your data - the existing information in your databases and new information as it comes in - to make the best possible decisions, immediately. This is critical to all organizations today. If you are not making the best possible decisions, in real time, you can be assured that your competitors are.

Real-time decision making is relatively easy in smaller organizations, where a small group of people know all the key information. But in larger organizations, distributed across many office locations and time zones, it's a lot harder.

As a result, most large organizations make decisions slowly, and make them based on old data. Their data processing pipelines are set up in such a way that this is guaranteed.

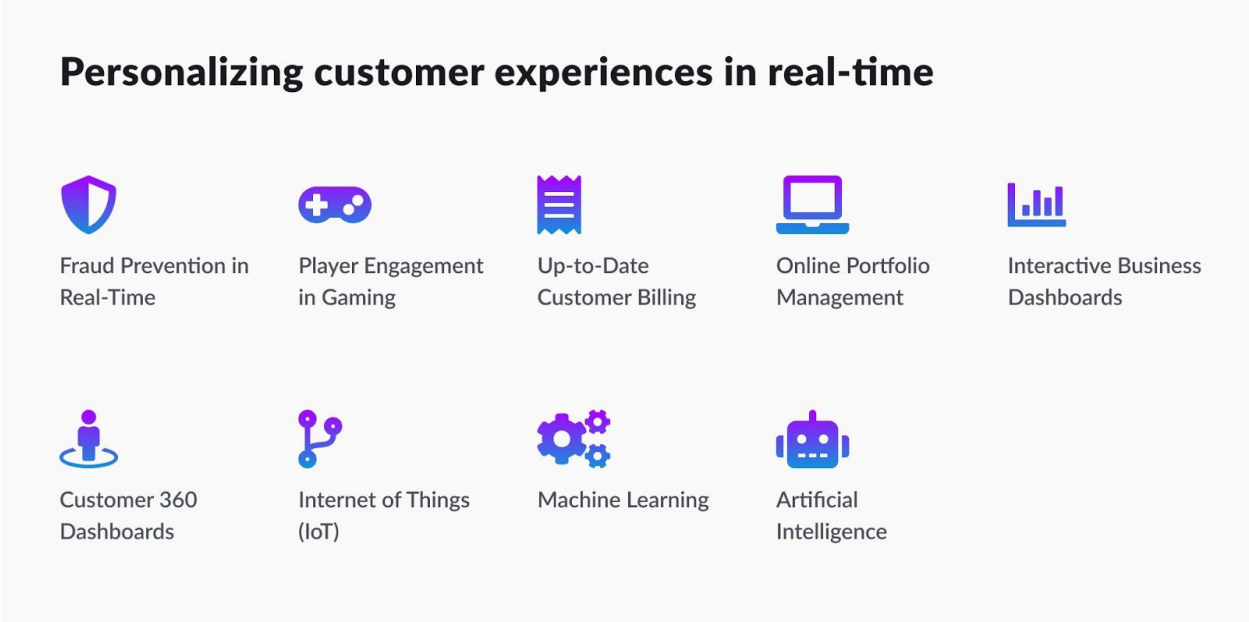
When new data is generated, it's batched - that is, it's delayed from the start - then used, perhaps daily, to update a transactional database. Once the update is complete, data is

reformatted, recombined, and stored in an analytical database, in a process widely referred to as extract, transform, and load (ETL).

Queries are run against this stale data. A NoSQL database may ingest data faster - but it's hard and slow to run queries against a NoSQL database, so the ETL process is still needed to make it available for queries.

Real-time decision making at scale is, at the same time, novel, hard to achieve, and the new standard for how organizations make decisions.

Here are a few specific examples of use cases that are difficult or impossible to implement any other way.



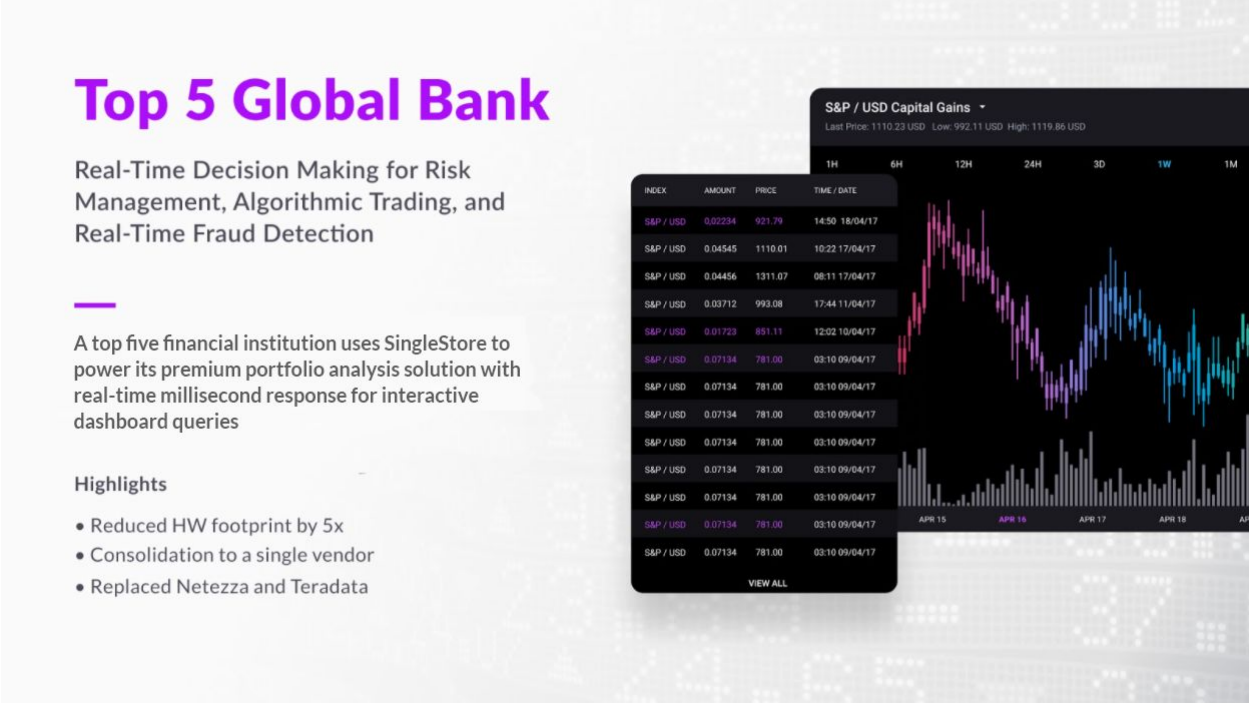
Any analytics process is worse if it's running on incomplete and out-of-date data. Any missing data is a major hindrance to the usefulness of analytics, and missing the "latest and greatest" can render the output of analytics more or less useless. You need to treat real-time decision making as an absolute requirement for your business to survive and thrive in today's environment.

The only way to deliver true real-time decision making at organizational scale is to combine ingest, transactions, and analysis in a single, continually updated, fast database. For years, this wasn't possible, but there's new technology that makes real-time decision making far more practical and affordable.

New Solutions to an Old Problem

There's a new class of databases that promises to make real-time decision making at scale possible. They make up a category so new that it has several overlapping names: hybrid operational and analytical databases (HOAP databases), hybrid transactional and analytical databases (HTAP databases), translytical databases, converged databases, or NewSQL databases.

SingleStore is part of this new group of solutions, using a breakthrough architecture that distributes a single database across multiple machines to support scalability. SingleStore has been able to run many operations ten times faster than a comparable Oracle solution and at one-third the cost, all against a fully relational database that supports structured query language (SQL) for analytics.



Real-time decision making is more required than optional in financial services (Source: SingleStore webinar on HOAP databases)

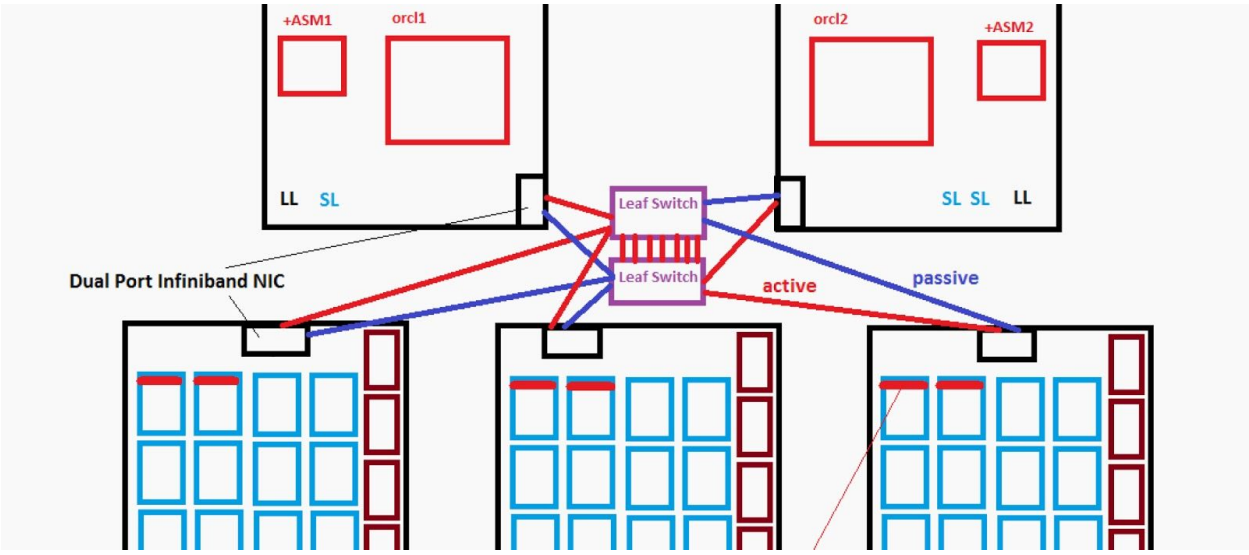
SingleStore is used by a Top 5 global bank for real-time decision making in support of risk management, algorithmic trading, and real-time fraud detection. Their usage of SingleStore includes a portfolio analysis solution for well-off customers that delivers responses to queries from an interactive dashboard within milliseconds.

Legacy Databases and Scale

"You can't solve a new problem with old thinking," to paraphrase Albert Einstein. Legacy databases, called online transactional processing (OLTP) databases, were designed to run on a single server. They have a kernel architecture that requires core operations to run on a single machine. Workarounds to this design limitation are complex, expensive, difficult to maintain, and limited in scope.

A well-known, and somewhat extreme, example of a legacy database trying to work at enterprise scale is the Exadata database machine from Oracle. It combines integrated hardware and software, including in-memory processing and storage, flash memory processing and storage, terabytes of operational memory, hundreds of terabytes of disk storage, and specialized, integrated hardware and software. These are all attempts to deliver scale on core software that was never designed with modern, distributed computing systems in mind.

A typical Exadata database machine might easily cost a million dollars and requires a small team to run. Updating it to a new software and hardware version can take six months to a year. Running an Exadata database in high-availability mode requires doubling most of the related costs, and you can't even be sure that your failover strategy will work; the architecture is so complex that just testing your failover strategy is prohibitively difficult and expensive.



Reducing operational complexity is a core benefit of SingleStore

And all the expense, work, and organizational commitment, plus a little luck, just gets your transactional data updated faster. You still have analytics to run and decisions to make. The problems of legacy databases for data ingest are so severe that a rather massive workaround was created, in the form of NoSQL databases. These databases take in data very fast because they eliminate costly data locking mechanisms and can scale horizontally - you can just keep adding servers as your ingest needs grow. But they scale horizontally by skipping the tasks of structuring data and running transactions. They simply bring in semi-structured or unstructured data and get it onto disk, fast.

This rapid but incomplete process simply kicks the can – the need to structure most data to make it useful – down the road. Often, the data just sits there, submerged in what is rightly called a data lake; available, in theory, but rarely used. To get to the data, people or programs that want to use it need to have some kind of pre-analysis stage to organize it, which slows analysis. Or a modified ETL process puts the data into some kind of order to make it available for analytics using SQL, the industry standard. (This process tends to run slowly, as it's having to do a lot of processing to add structure.)

Today, both OLTP and NoSQL databases are legacy databases, each limited in their own way; OLTP databases lack horizontal scalability, while NoSQL databases lack structure and, well, SQL.

Legacy Databases, Analytics, and Decision-Making

Data structured for transaction processing commonly requires additional transformation operations, such as joining with other data to support various queries. Also, the OLTP database system is optimized to keep transactions up to date, with little additional capacity or table structures for fast query support. NoSQL databases, on the other hand, have no structure at all, and lack a robust query optimizer, resulting in a system that's poorly adapted to supporting queries.

These problems led to the creation of specialized online analytical processing (OLAP) databases. OLAP databases were designed to supplement and complement OLTP and, later, NoSQL databases. To get an OLAP database "updated" (almost always with data that's far from current), an organization runs an extract, transform, and load process (ETL) against the OLTP database or, in some cases, against NoSQL data.

During an ETL process, data is taken out of the OLTP or NoSQL database (extracted), reorganized and recombined (transformed), and loaded (loaded) into an OLAP database, also referred to as a data warehouse. For a NoSQL database, an additional step of structuring the data is needed.

After ETL, the data is, hopefully, in a format where it can be used to quickly answer queries. Note that, during the loading process, the OLAP database may have limited or no availability, just like OLTP databases running a batch transaction update.

Like OLTP databases, OLAP databases tend to run a single core process. That's because it's very difficult to distribute data storage and transaction processing across multiple machines, and deal with the inevitable hardware, software, and network failures, without ending up with duplicate records, missing records, corrupted records, or simply uncertainty as to whether any of these problems have occurred.

Using legacy databases imposes delays and complexity

For both OLTP and OLAP databases, there are then just two kinds of scalability: scaling up and caching. Scaling out is not possible. Scaling up just means upgrading to faster disk drives or disk farms, using the widest possible data bus, using a fast processor with multiple cores, optimizing system and application software, and tuning all of the pieces to run together as fast as possible.

Caching means loading large chunks of data into memory in a smart way so the loaded data is what needs to be processed now, and persisting cached data to disk and loading new data are both optimized and reliable. When an interruption occurs, or the cache otherwise becomes suspect, it's flushed and re-initiated. Maintaining a fast and accurate

cache requires detailed understanding of the data's structure, its content, and what queries are most likely to be run against it.

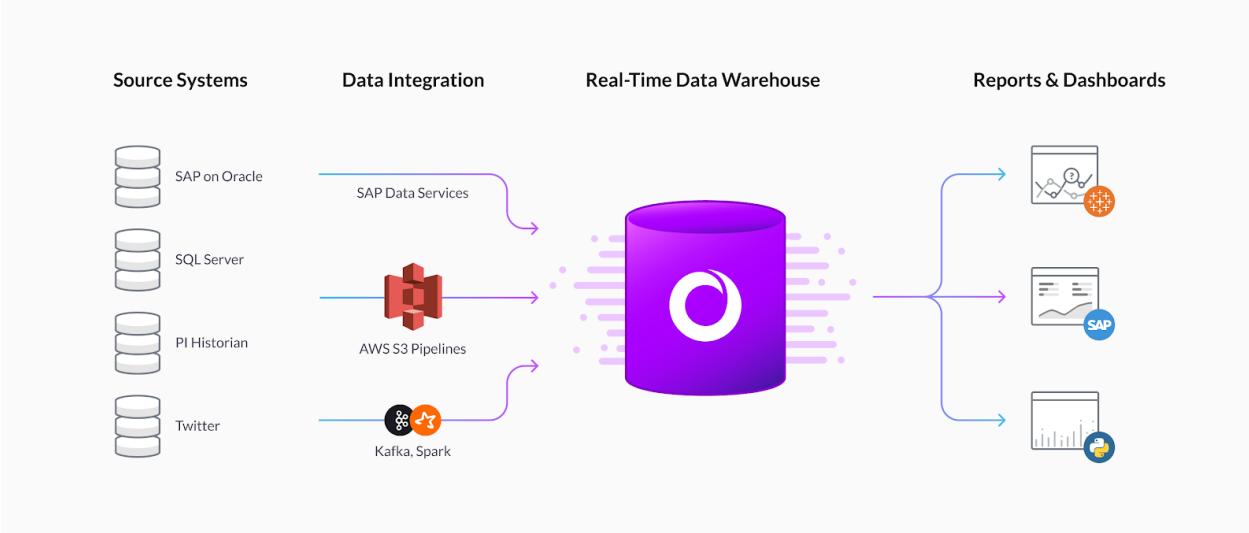
The goal here is to enable real-time analytics, but "real-time" should be put in quotes. It means that the query runs fast enough that the user can wait only a little bit for the data, and can then use the response to frame their next query. (Or, alternatively, that a business intelligence program can run through the data fast enough to produce intra-day reports.) However, when using legacy databases and traditional processes such as ETL, real-time analytics are run against stale data.

Enabling Real-Time Decision Making

We've seen that real-time decision making is very difficult to accomplish when using legacy database technology, especially when delivered at scale. Legacy database technology can't stream data in, run transactions against existing data, and support real-time analytics against current, up-to-date data. How, then, can we enable true real-time decision making across an enterprise or business unit?

The answer is to use technology to give a large organization the same operational speed and flexibility as a small one. To do this, we bring data acquisition, transaction processing, data transformation, and real-time analytics into a single, fully scalable database. And we run ingest, updates, transformations, and querying simultaneously. So queries are made against the latest, updated data, with sub-second response.

This is extremely hard to do with legacy architectures that require separation of the needed operations into separate silos, with batch operations and limited availability, and unavailable databases at different steps along the way. But it's possible with a NewSQL database.



Kellogg's uses SingleStore to deliver real-time decision making

It took decades for the legacy architectures to evolve. Why is it suddenly possible to collapse data ingest, database updating, and querying into a single, responsive operation?

The secret is full, not limited, scalability. Traditional databases, which are designed around a single core process, can only scale up: throw a single bigger, more powerful machine at the problem. Unfortunately, there are limits on the capacity of any single machine. (And cost and complexity start to rise exponentially as performance increases only linearly, as shown with Oracle's Exadata integrated solutions.) Additionally, OLTP and OLAP databases are kept separate, resulting in data replication, additional data copies, and maintenance, all used to compensate for the limitations of legacy single-threaded databases by spreading the work of transaction updating and query response across two distinct operations.

The new generation of databases is fully distributed, able to run as a single database across a pool of machines while maintaining consistency and stability.

This allows you to scale out: to throw many machines, as ordinary or as powerful as you would like, at the problem. And you can use this new generation of databases to speed up transaction processing, avoiding batching on ingest; to speed up analytics processing, taking real-time analytics to new heights; or to combine both processes and enable true, end-to-end real-time decision making.

How True Scalability Works

How does the new generation of databases deliver true scalability that was, until recently, considered impossible?

Solving this problem took a combination of hard work and new technology. The new technology, while crucial, is well understood. CPUs, RAM, flash memory, disk technology, and the interconnections among them have all become exponentially faster and cheaper. The rise of open source software for highly distributed architectures across storage and compute has increased knowledge sharing and improved the resources available to both existing and new technology companies.

The demands on the core technology, however, increased even faster than it improved, as an increasingly interconnected world seems to require that we process all the data, all the time, with all the availability that one could imagine. These ever-increasing demands made it ever more obvious that a new, fully scalable database is the only real solution to delivering real-time decision making at scale.

That's where the hard work comes in. Applying new techniques in software design to the problem of real-time data processing at scale resulted in a novel set of architectures, such as the distributed architecture used by SingleStore.

Within a SingleStore database, a table is broken down across two kinds of nodes. Aggregator nodes hold the data schema and "own" part of the table's contents; they dispatch queries to be run against "their" data and, as the name implies, aggregate the results. Multiple leaf nodes hold the data controlled by each aggregator node. Leaf nodes run queries against the data stored in them and return results up the chain for aggregation.

Explained that way, it sounds simple, but of course it isn't. Before making a decision to adopt SingleStore as a solution, you'll want to review a much more detailed description of SingleStore's architecture. And you may also want to learn more about SingleStore's evolution, at both the company and the technology level.

Today, SingleStore's software solves the core problems found in traditional architectures and delivers real-time decision making. However, traditional architectures have been evolving for generations. SingleStore is still being evolved to deliver all the features of

existing solutions and all the points of integration needed to work optimally within existing solutions.

Tomorrow's Solution, Today

Fifty years ago, relational databases were a breakthrough solution to data processing problems. And fifteen years ago, Hadoop and other NoSQL solutions were a solid first attempt at bringing full scalability to databases, without tackling the challenging problems of maintaining structured data and running transactions against it. As a result, most organizations have big, and growing investments in legacy relational database technology, and in NoSQL solutions as a partial alternative.

NewSQL databases such as SingleStore, however, represent the best of both worlds - the reliability and consistency found in relational databases and the full scalability that was formerly only available with NoSQL. As such, NewSQL databases are better for transactions and also better for real-time analytics.

An especially valuable benefit that arises in many implementations of SingleStore is the architectural simplification caused by putting ingest and transactions (OLTP) and analytics (OLAP) on a single database. There's no need for two silos anymore, nor for ETL as a chasm separating them. A single database handles ingest, processing, and analytics, fully enabling real-time decision making.

Implementation decisions do not have to be "either-or". SingleStore can also be used to accelerate and smooth existing processes. This helps organizations meet challenging SLAs in the face of mushrooming demands for the volume of data ingested, for concurrency in querying, and for more speed through the whole process.

Data ingest can go to both Hadoop, for long-term storage, and to SingleStore, for processing and analytics. Transactions can run through Oracle database software and then be sent to SingleStore for an abbreviated ETL-like process and analytics.

Leading organizations in all market segments are currently using SingleStore to overcome the limitations of traditional databases for transaction processing, real-time analytics, and converged operations such as real-time decision making, as described here. We recommend that you book a demo with us to help you evaluate your organization's current and upcoming challenges.

	Legacy Relational Database	NoSQL Database	NewSQL Database
Ingest Performance	**POOR** Can't scale out for faster ingest	**EXCELLENT** Easy "scale out" for faster ingest	**EXCELLENT** Easy "scale out" for faster ingest
Data Accessibility	**EXCELLENT** Structured data SQL queries	**POOR** Unstructured/semi-structured data and no SQL queries	**EXCELLENT** Structured data SQL queries
Analytics Performance	**FAIR** Limited concurrency (support for multiple simultaneous users)	**POOR** Queries run slowly; requires custom queries	**EXCELLENT** Fast queries and high concurrency
Scalability	**POOR** No "scale out"	**EXCELLENT** Full "scale out"	**EXCELLENT** Full "scale out"
Price-Performance	**POOR** "Scale up" is expensive and limited	**EXCELLENT** "Scale out" is affordable and almost unlimited	**EXCELLENT** "Scale out" is affordable and almost unlimited

Legacy relational databases vs. NoSQL databases for real-time decision making at scale.

Get Started

Talk to a Database Specialist and see how SingleStore can transform your business into real-time today. Please reach out to us via singlestore.com/db to talk to our representative.