



# NUODB

White Paper

2.12.2012

# NuoDB: Webscale SQL for the Cloud



## A Brand New Type of SQL Database For Managing Transactions Elastically At Scale

---

“Elastically scalable transactions are the biggest breakthrough in database technology in 25 years.”

- Jim Starkey, Founder, NuoDB, Inc

---

### Executive Summary

NuoDB is a brand new kind of transactional database that dramatically changes what's possible with an OLTP. It is SQL compliant, guarantees ACID transactions, *yet scales seamlessly and elastically on heterogeneous and decentralized cloud-based resources.*

The NuoDB database is fundamentally different from today's SQL and non-SQL based database management systems in that it features *a ground breaking new distributed database architecture that allows it to operate asynchronously and scale elastically on the cloud.*

From a user's perspective, NuoDB fundamentally behaves like any other traditional SQL database thereby providing all the benefits users have come to expect from standards-based SQL solutions. The NuoDB database simplifies application development, guarantees the integrity of transactions and enables the separation of data from the application, *while seamlessly scaling in and out on cloud-based resources to deliver high performance at scale with highly efficient resource utilization.*

### Business Problem

In today's world, many Web applications have operational requirements that can't be met by traditional database technologies. Big data, highly variable peak loads, geographically distributed loads, and 100% uptime requirements for high volume Web applications, for example, are pushing traditional operational databases to their limits.

While elastic, cloud-based compute infrastructures are designed to scale in and out in response to load and provide organizations with a more cost-effective and simpler way to manage applications at scale, all major SQL-compliant, ACID, relational database management systems to date have their roots in a 1970s architecture of central control that is, quite simply, incompatible with the notion of elastic scalability in the cloud.

What many of today's applications require is elastic *scalability* at the *database level*, without compromising any of the powerful features of traditional databases, including ACID transactions, independence of the data from the application, and SQL compliance.

## Why Aren't Traditional SQL Databases Compatible With The Elastic Cloud

Without doubt, traditional SQL databases are ideal for many types of business applications.

But they are all built on a 30+ year old architecture that is based on a monolithic, synchronous, and centralized approach to managing table-structured files on disk, originally developed to leverage client/server architectures. Traditional SQL databases were simply not designed to leverage newer, cloud-based compute and storage resources that can be easily added and removed to dynamically handle larger loads or to increase performance on demand.

The traditional SQL vendors have recognized the need to support elastic scalable cloud based computing and have retrofitted their products to meet some of these needs. However, what's needed is a new generation of database technology, built on a radically different type of architecture - an asynchronous, peer-to-peer, shared nothing architecture that scales in and out on demand - to enable organizations to fully leverage the capabilities of the elastic cloud.

## NuoDB: A New Type of Operational Database Built to Handle Webscale Transactions in the Cloud

NuoDB is the first and only SQL based database built on the principle of *emergence*.

Emergent behavior is characterized by relatively simple actions and interactions by components that produce complex, coordinated behavior in a system. Think of a flock of birds in the sky, or a school of minnows in the sea.



NuoDB has been designed from ground up, guided by the principle of emergence - object oriented, highly decentralized, and asynchronous - to create a new kind of operational database that is optimized for the cloud, while providing 100% ACID guarantees and 100% SQL compliance.

# Operational Database Landscape

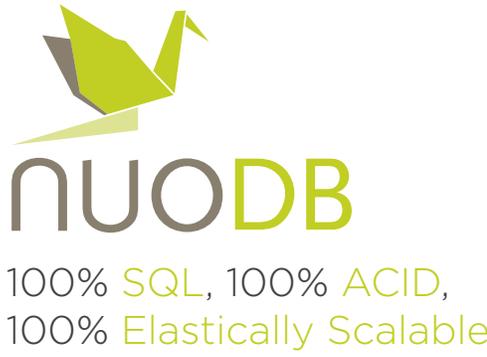
	Not Webscale	Webscale
SQL	<p><b>Examples:</b></p> <ul style="list-style-type: none"> <li>• ORACLE, DB/2, MS SQL Server, MySql, ...</li> </ul> <p><b>Comments:</b></p> <ul style="list-style-type: none"> <li>• Cluster is not cloud, and scale-up is not scale-out</li> <li>• Pushing the hardware is not “commodity machines”</li> <li>• All of the cost arbitrage is cloud elasticity, multi-tenancy, DBA dependencies</li> <li>• Customers cannot provision for peak web load</li> </ul>	 <p>100% SQL, 100% ACID, 100% Elastically Scalable</p>
Not SQL	<p><b>Examples:</b></p> <ul style="list-style-type: none"> <li>• db40, Versant, Objectivity, Gemstone, Progress, Perst, OrientDB,...</li> </ul> <p><b>Comments:</b></p> <ul style="list-style-type: none"> <li>• Specialized application-specific databases</li> <li>• No independent data asset</li> <li>• Mostly NO ACID, NO SQL, NO Webscale</li> <li>• Will continue to reside in specialist riches</li> </ul>	<p><b>Examples:</b></p> <ul style="list-style-type: none"> <li>• BigTable, SimpleDB, DYNAMO, Cassandra, ...</li> </ul> <p><b>Comments:</b></p> <ul style="list-style-type: none"> <li>• Most of these are not only NO ACID, but NO SQL</li> <li>• Can't handle valuable data</li> <li>• Degenerate query modules</li> <li>• Vastly increased application complexity</li> <li>• 100% of corporations depend on SQL/ACID - these needs will not change</li> </ul>

Figure 1: Where NuoDB fits in the OLTP landscape

## NuoDB Benefits

NuoDB offers key benefits to users that have never before been available in a single product offering:

- Elastic Scalability
- SQL Compliance
- ACID Transactions
- Multi Tenancy
- Extreme Availability
- Geographically Distributed, Active-Active Operations
- Simple Administration

	Not Webscale	Webscale	
	Traditional SQL	NoSQL	
SQL	✓	X	✓
ACID Transactions	✓	X	✓
Data/App Independence	✓	X	✓
Elasticity	X	✓	✓
Optimized for Multi Tenancy	X	✓	✓
Extreme Availability	X	✓	✓
Geo-distribution	X	✓	✓
Simple DBA	X	✓	✓

Figure 2: Traditional versus Webscale Database Feature comparison

**Elastic Scalability.** NuoDB’s shared-nothing architecture and asynchronous operation are designed to allow the database tier to work in concert with the elastic cloud. The database can dynamically scale out and in based on load. It can support multiple platforms dynamically, so that an organization can use on-premises infrastructure for normal traffic loads, and dynamically spill over to a private or public cloud to handle peak loads. Organizations can use different public cloud providers so that an outage with one provider will not interrupt operations.

**SQL Compliance.** NuoDB is a SQL-compliant database. A developer simply writes applications using standard SQL connecting over the network using one of many standards-based language or environment-specific drivers such as JDBC or ODBC. To a developer, NuoDB is almost indistinguishable from a traditional client-server RDBMS.

NuoDB implements the majority of the SQL99 standard. Existing standards-compliant databases will port and run with no, or minimal, modification. NuoDB is not simply a new storage engine for MySQL or other existing databases; it is a new SQL engine written to take full advantage of distributed, cooperative nodes. The result is that database designers don’t need to compromise on schema design by de-normalizing tables, removing joins, or to get the performance their application needs.

**ACID Transactions.** NuoDB guarantees ACID (Atomicity, Consistency, Isolation, Durability) in shared-nothing, decentralized environments, enabling application developers to rely on the database tier to process transactions reliably and consistently, simplifying application development for the cloud.

## NoSQL Databases

While traditional relational databases struggle to handle webscale applications, recent approaches to working with large datasets on distributed resources utilize non-relational, distributed key-value stores. These “NoSQL databases” do not provide ACID properties, and are not SQL compliant. They are generally used for OLAP applications for performing analytic queries on data, and with unstructured data.

**Multi Tenancy.** NuoDB’s unique architecture supports running a far greater number of database instances on less infrastructure, allowing organizations to more easily and affordably grow, shrink, allocate, reassign, and provision database service resources according to need or policy. Just as virtualized systems allow more efficient utilization, provisioning, monitoring, and repurposing of hardware resources, NuoDB enables efficient multi-tenancy, coupled with automatic policy-based provisioning to maximize resource utilization.

**Extreme Availability.** NuoDB’s shared-nothing architecture has no master instance, and no single point of failure. NuoDB is resilient to infrastructure and partition failures. It allows “hot” upgrades to hardware and software infrastructure, dynamic changes to the database structure, and “hot” datacenter moves. NuoDB adjusts as resources are removed without the need for downtime. Nodes performing transactions maintain no persistent state, and data is distributed across transaction and storage nodes and is always available.

When network partitions do occur, NuoDB continues offering services via the most available and healthy segment while the other segments provide a consistent read-only snapshot while awaiting reconnection - at which point they synchronize with their counterparts. Clients can simply reconnect to the active partition of the cluster to continue processing updates.

**Geographically Distributed, Active-Active Operations.** *NuoDB is designed to operate in an Active-Active configuration in geographically distributed datacenters, which is ideal for applications that need to service worldwide loads with low latency. In addition, running in an Active-Active configuration in multiple data centers and geographic locations (and even using different providers) is ideal for ensuring business continuity for mission critical applications that require 100% availability, even if an entire datacenter fails or is taken offline.*

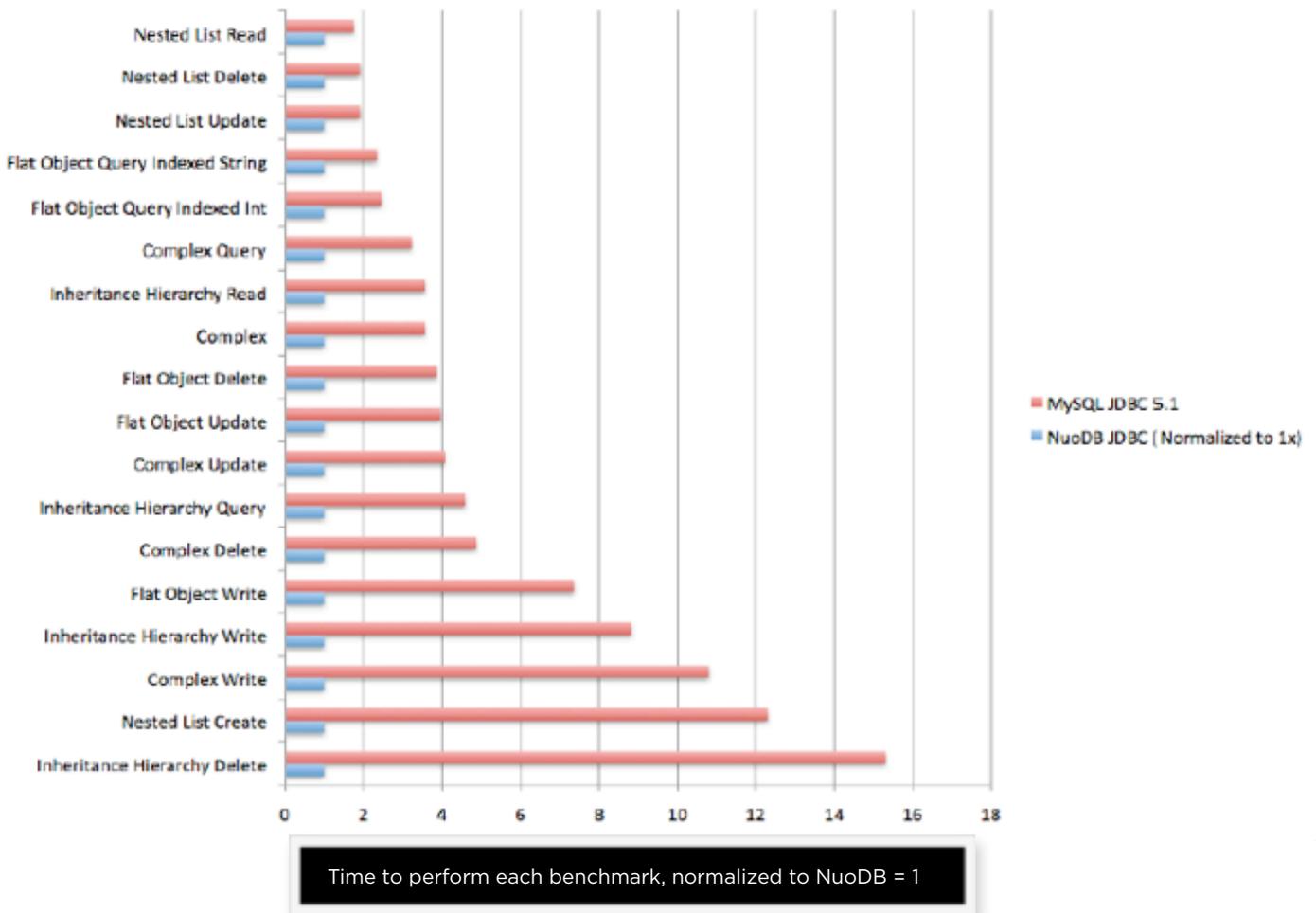
**Simple DBA.** The complexities of performance tuning and database administration are well known to traditional DBAs.

In contrast, NuoDB requires minimal configuration and almost no oversight in production. Administrators configure policy files and allocate systems resources in the cloud or from local resources. NuoDB manages the resources automatically, adjusting as required. Monitoring consoles provide a detailed operational view of the NuoDB database as a whole showing how resources are allocated, operational characteristics, and any system alerts.

## Is NuoDB Right For Your Application?

- Is your application “hitting the wall” with a traditional database like MySQL
- Does it require a high performance, “webscale” database application infrastructure
- Must it deal with high database loads with read and write operations
- Must it deal with high peak transient loads
- Does it require 24/7 operation
- Must it be highly reliable and fault tolerant
- Must it utilize cloud-based resources
- Must it support a heterogeneous infrastructure (e.g. running on-premises and bursting to a public cloud)
- Must it handle geographically distributed traffic loads with low latency
- Does it require very high multi-tenancy
- Is its data structure likely going to change over time

In tests, NuoDB running on a single node was 2x to 20x faster than MySQL 5.1 running the industry standard Poleposition Benchmarks...



Simply adding a second NuoDB node nearly tripled NuoDB's transaction throughput...



Figure 3: NuoDB - Blazing Performance

## NuoDB Architecture

NuoDB is based on a radically new and elegant approach to the database. It was patterned from the most successful distributed algorithms, systems, and techniques - those that have had the most success in real world systems providing scalability of service without proportional costs.

NuoDB is the first and only relational database built on the principle of emergence. Emergent architectures are object oriented, highly decentralized, and asynchronous - ideal for operating in the cloud.

*[E]mergence is the way complex systems and patterns arise out of a multiplicity of relatively simple interactions. Emergence is central to the theories of integrative levels and of complex systems. - Wikipedia*

Although various types of software applications are successfully using "emergent architectures" to exploit cloud-based resources, *until now there has not been an operational database built from the ground up with this type of architecture.*

To understand NuoDB's architecture, start by setting aside everything that you know about traditional synchronous and centralized database architecture.

At the heart of NuoDB is the atom. Atoms are self-describing data and metadata that together comprise the database including the schema, the indexes, and the underlying data. Everything in the database is an atom. For example, each table is an atom that describes the metadata for the table and references other atoms that describe ranges of records in the table and their versions. *The atoms communicate with each other in an asynchronous, peer-to-peer manner that is consistent with the shared-nothing requirements of an elastic cloud infrastructure.* Copies of atoms may be present on several computers at any time. When an atom changes, the node sends messages to the other nodes that also have copies, automatically replicating the changes using highly reliable, asynchronous message queues.

The **NuoDB Transaction Engine** is comprised of one or more processes (transaction nodes) running on (diskless) compute nodes. These processes execute the SQL layer by operating on database atoms, listen for changes, and communicate changes with peers.

The **NuoDB Storage Manager** is simply a special kind of Transaction Engine that knows how to store and retrieve data on disk.

A **NuoDB Broker** is a process running on at least one node that provides applications SQL/JDBC access to the database.

A **NuoDB database** can be as simple as a single transaction node, a single storage node, and a broker.

From the point of view of a client application, NuoDB looks very much like a traditional relational database. It has a Java/JDBC interface that allows clients to read and write data, create and drop domains, tables, schemas, indexes, etc. Client applications are not aware that the database is not local.

SQL is parsed and executed on the transaction nodes, where the client requests are serviced. The transaction nodes update the system tables (represented as atoms) with the appropriate SQL metadata statements to create, alter, and drop metadata objects.

Storage nodes serialize atoms to disk. If a transaction node needs an atom, it first requests the atom from memory on the most responsive transaction node, where it resides in memory. If the atom does not exist on any transaction node, it is retrieved from the most responsive storage node. The function of the storage nodes is to write changed atoms to disk, to be the source of last resort when transaction nodes need atoms that do not exist elsewhere on other transaction nodes, and to undo changes made by active transactions on nodes that disappear from the cloud. Adding a new storage node automatically creates a copy of the entire database which increases both performance and resiliency. The NuoDB Storage Manager can leverage virtually any type of data storage that can manage a key-value store, for example a local file system, Amazon S3, or a SAN.

Scaling out is achieved by simply adding nodes. Scaling in is achieved by removing nodes. As long as a single transaction node, a storage node, and a broker exist, the database continues to operate.

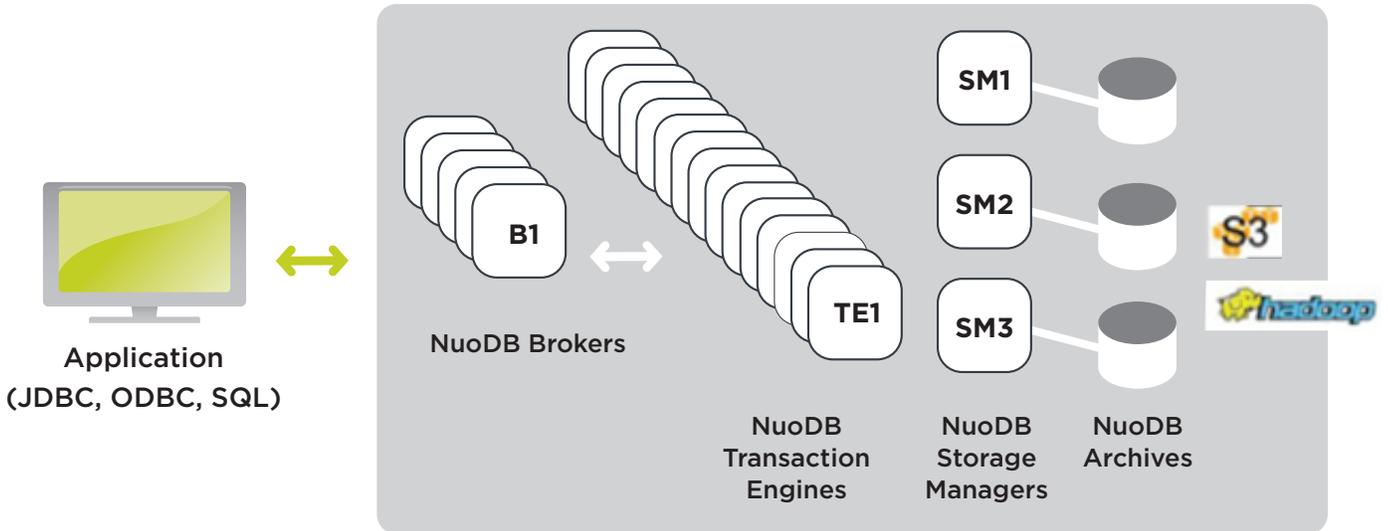


Figure 4: NuoDB Architecture

A true peer-to-peer database architecture requires a new type of concurrency control. NuoDB uses a new and innovative type of multi-version concurrency control (MVCC) to guarantee that each transaction has a consistent view of data, and to prevent concurrent transactions from overwriting each other's changes.

NuoDB's implementation of MVCC is based on a distributed version management system that eliminates the need to communicate locking messages between nodes, and does not rely on "eventual consistency" to provide ACID transactions. Changes are sent *asynchronously* to other nodes using highly reliable message queues. An UPDATE does not replace the stored record, and a DELETE does not remove it. Instead, NuoDB manages multiple versions of the record by carefully tracking which versions belong to each transactional context. Each transaction reads the most recent version of the record that was committed when the transaction started. Transactions see a stable view of data, except for the changes they make themselves.

NuoDB's MVCC implementation eliminates read/write conflicts by allowing the writer to create a new version without changing the reader's view of data and by allowing a reader access to a stable view of data even if it has been changed by a concurrent transaction.

NuoDB's MVCC implementation eliminates write/write conflicts by restricting transactions so they can modify or delete records only if the transaction sees the most recent version of the records. MVCC transactions are consistent, and require less communication than transactions that use traditional locking or two-phase commit for concurrency control.

## ACID

ACID is a set of properties (atomicity, consistency, isolation, durability) that guarantees that every database transaction is processed reliably. ACID databases simplify application development by leveraging the database tier to guarantee the reliability of the data, so that the developer does not need to build the logic into each application.

NuoDB guarantees ACID transactions.

NuoDB transactions are atomic. All the changes made by the transaction are either made permanent when the transaction commits, or are completely removed from the database.

NuoDB transactions are consistent. They transform the database from one consistent state to another as defined by the explicit and implicit constraints in the schema. Each transaction sees a consistent view of the database at all times.

NuoDB transactions are isolated. No transaction can see changes made by transactions that were not committed when it started – except, of course, its own changes. MVCC also prevents transactions from overwriting changes from concurrent transactions.

NuoDB transactions are durable – with or without disks. NuoDB handles durability differently than most databases. By default, changes are durable when they are present in memory on at least two nodes (including a storage node), even before the node has serialized the data to disk.

### Summary

NuoDB is a brand new generation of operational database that is built from the ground up using the principles of *emergence* to fully leverage the cloud in an asynchronous and highly decentralized manner - while providing both SQL compliance and ACID transactions.

For the first time, a SQL, ACID, OLTP database can scale elastically in the cloud, dramatically changing what's possible with an operational database.

NuoDB is currently in private beta.

If you have a webscale application that can benefit from NuoDB, we'd love for you to participate.

You can sign up at [www.nuodb.com/download](http://www.nuodb.com/download)

NuoDB, Inc.  
18 Hurley St.  
Cambridge, MA 02141  
p +1 (617) 500-0001  
[www.nuodb.com](http://www.nuodb.com)