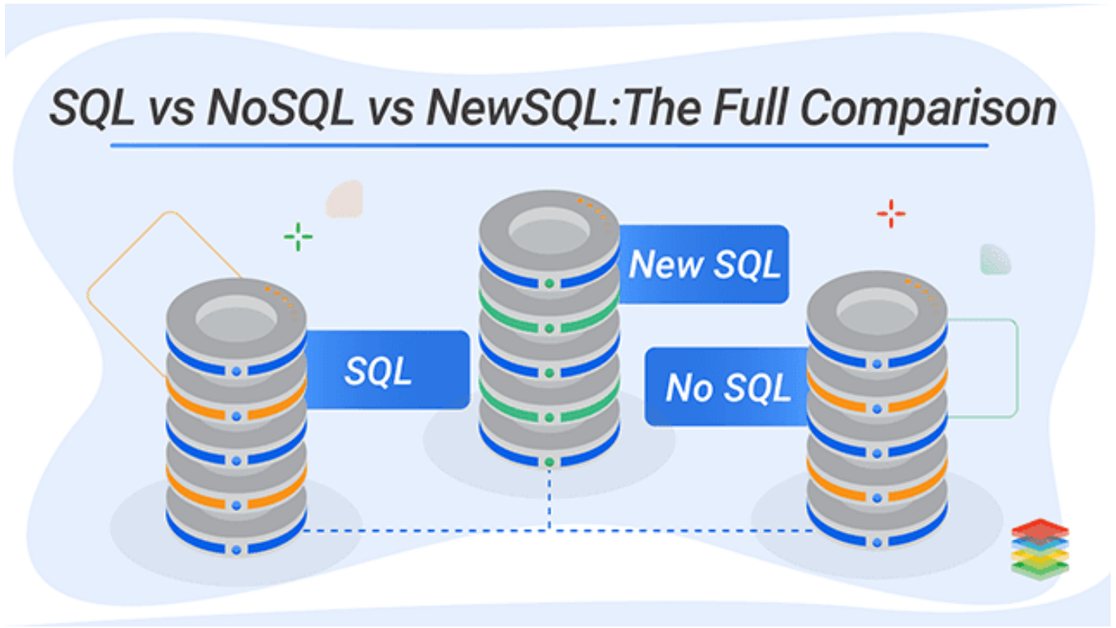
SQL vs NoSQL vs NewSQL: The Full Comparison

Navdeep Singh Gill | SERVERLESS | 8 mins read | Mar 26, 2019

Table of Contents



Overview of SQL vs NoSQL vs NewSQL

In this blog, we will show you the basic difference between SQL vs NoSQL vs NewSQL. These are the types of databases and in the upcoming paragraph, you will understand each of them.

SQL Databases can also be termed as Relational Database management Systems also known as RDBMS which is a classical approach to store and operate on historical data. In such a system,

humongous information processing that tends to be unstructured in nature.

Over time SQL has undergone many iterations to support the vast quantity of data processing and pipelines but still, it is not the efficient answer to Big Data systems expecting a quick response and supreme scalability.

A new approach known NoSQL was introduced was devised to work around the limitations imposed by the former. NoSQL system was designed to provide fast scalability when dealing with unstructured data platform or handling Big Data applications.

NoSQL databases make use of a key-value pair, Documents, graph databases or wide–column stores without a typical schema. It is also horizontally scalable as opposed to only vertical scaling in RDBMS.

NoSQL showed a great promise to be an ideal database system for Big Data applications but like anything else, it fell short due to some major drawbacks discussed below.

This is where NewSQL came alive. NewSQL is the latest development in the world of databases systems. NewSQL is a relational database with the scalable properties of NoSQL.

What is SQL Relational Database

query language and were the pioneers of database design philosophy. Since the mid 80's SQL has been a standard for managing and querying relational datasets however, the early beginnings of the relational model can be dated back to the '60s and '70s where the urgent need to distinguish between the application data and application code emerged, allowing the developers to focus on other aspects of the program development such as access to and manipulation of data at hand. IBM's IMS was the first fully functional relational database which was albeit designed for a different purpose, to organize data for the Apollo space exploration programme.

The relational database is a collection of time-varying, normalized relations of assorted degrees. The following intuitive correspondence can be made

- A relation is a file
- Each file contains only one record type
- The records have no particular order
- Every field is single-valued
- The records have a unique identifying field or composite field, called the primary key field.

Concepts of SQL Relational Database

ACID

Atomicity, Consistency, Isolation, Durability to maintain the reliability of transactions.
Atomicity – completion of the transaction as a whole or none at all
Consistency – assures the stable state of the database with or without changes
Isolation – multiple transactions do not interfere with each other
Durability – permanent effect on the database by the changes

Normalization

A process of designing efficient databases

1NF – Split the table by separating repeating and nonrepeating attributes. All domains are simple and all elements are atomic
2NF – Remove partial dependency between attributes. No attribute should be functionally

Scalability

The capability of the database to handle growing amounts of data. Vertical scaling helps to enhance the existing capacity of the database server. Most SQL databases support vertical scaling. They can, however, scale up not scale out.

Domains

A domain is a named set of scalar values, all of the same type. They can be used to impose semantic constraints.
Rely on traditional features and utilizes a defined data schema.
Support JOIN functionality, engineered for data integrity

Drawbacks of SQL Relational Database

Although RDBMS provides exclusive features, they suffer greatly from some major drawbacks.

The Rigidity of Data Modeling

One of the biggest limitations of the relational database is the rigidity that comes from organizing the data into the particular structure in tables and relations. Since all the data cannot be fitted into tables conveniently, hence this approach can't be applied to all natural data and be store as trees and graphs
But, RDBMS work around this limitation by modeling this data in a normalized manner with parent-child relations, which is still not enough
Diversity

The complexity of the data also creates a limitation in the relational database. These databases are made to organize the data by common characteristics. Complex numbers, images, and multimedia data are hard to store, access and process.

Inefficient Usage of Space

When we define the schema of the relation we define the size of all the attributes. Not all the records have data that use the full space. Some have a short length. Every record needn't necessarily fit into the given data type again resulting in the wastage of the space.

Heavy Weight Changes

the schema of a database which is already existing.

Inefficient for Big Data

SQL is not suitable for volume, velocity and variety data, rendering it highly inefficient for a cloud-based application.

What is NoSQL

These problems turned out to be the origin of the impetus for NoSQL movement in mid to late 2000's. The key working strategy being that they forgo strong transactional guarantees and relational model of DBMS in favor of eventual consistency and alternative data models like key-value pair, graphs. This was done following the belief that these aspects of existing DBMS inhibit their ability to scale out and achieve high availability needed to support web apps on the go. The two most well-known systems that followed the creed was Google's BigTable and Amazon's Dynamo and were restricted to use inside their own organization's only, leading to organizations creating their own open-source clones like Facebook's Cassandra and powerset's HBase. By the end of the 2000s, there was a diverse set of scalable and affordable DBMS.

Support for structured, semi-structured and unstructured data. No specific schema needs to be defined before entering data into NoSQL databases. New fields can be added and also supports nested data implementation and retrieval. Developers can use data type and query options requisite for the specific application resulting in faster development. Faster Development time is considerably reduced due to no complex SQL queries or join statements.

Auto Balancing

Division of data among multiple servers automatically, with no assistance required from applications.

Integrated Caching

NoSQL database cache data in system memory, so that it can increase data throughput and increase the performance in advance.

High scalability, reliability with a simple data model and simple query language.

The BASE Principle For Transaction

The base is to NoSql as what ACID is to SQL. Base ensures that NoSQL databases ensure their reliability in spite of the loss of consistency. Base stands for Basically Available Soft-state Eventually consistent.
Eventually consistent – System can become eventually consistent, information is updated wherever necessary.

Drawbacks of NoSQL

Lack of Consistency

Since these systems prefer availability over consistency, they fail miserably in cases where consistency is the most important thing as in financial transactions where there is a risk of system failure due to the non-synchronization of data nodes.

Lack of Analytics

For analytics, you require a relational model to process the data, as a result of which the whole database needs to converted using some sort of relational model, this leads to increased cost overhead.

Lack of Standardization

Doesn't provide security at the elemental level of data.

Transactional Nature

It is important in areas to facilitate fraud detection before completing the transaction, check for balance whilst on a call. NoSQL fails in cases where the database needs to compete against a high volume transactions per day as they require a highly scalable, consistent database.

What is NewSQL

The early counter-measures to the above approaches were a powerful single node machine that is able to handle all the transaction, a custom built middleware system to distribute queries over traditional DBMS nodes. But both of them are prohibitively expensive to be carried out.

As a result, there was a need of an intermediate database system which combines the distributed architectures of NoSQL systems with multiple node concurrency and a whole new storage mechanism. Thus Newsql can be defined as a class of modern relational DBMSs that seek to provide the same scalable performance of NoSQL for OLTP workloads and simultaneously guaranteeing ACID compliance for transactions as in RDBMS. In other words, these systems want to achieve the scalability of NoSQL without having to discard the relational model with SQL and transaction support of the legacy DBMS.

- Scaling out by splitting a database into disjoint subsets called either partitions or shards, leading to the execution of a query into multiple partitions and then combine the result into a single result.
- NewSQL systems preserve the ACID properties of databases.
- Enhanced concurrency control system benefits upon traditional ones.
- Presence of secondary index allowing NewSQL to support faster query processing times.
- High availability and Strong data durability is made possible with the use of replication mechanisms.
- NewSQL systems can be configured to provide synchronous updates of data over the WAN.
- Minimizes Downtime, provides fault tolerance with its crash recovery mechanism.

Difference Between SQL, NoSQL, and NewSQL

Feature	SQL	NoSQL	NewSQL
Relational Property	Yes, follows relational modeling to a large extent.	No, it doesn't follow a relational model, it was designed to be entirely different from that.	Yes, since the relational model is equally essential for real-time analytics.
ACID	Yes, ACID properties are fundamental to their application	No, rather provides for CAP support	Yes, Acid properties are taken care of.
SQL	Support for SQL	No support for old SQL	Yes, proper support and even enhanced functionalities for Old SQL
OLTP	Inefficient for OLTP databases.	Supports such databases but it is not the best suited.	Fully functionally supports OLTP databases and is highly efficient
Scaling	Vertical scaling	Vertical scaling	Vertical + Horizontal

Query Handling	fails when they get complex in nature	processing complex queries	queries and smaller queries.
Distributed Databases	No	Yes	Yes

SQL, NoSQL or NewSQL – Which is Best Solution to Big Data?

SQL complies with ACID properties and does well with vertical scalability, while NoSQL offers its own horizontal scaling and provides for BASE properties. NoSQL, however, does not play by ACID rules which are necessary to maintain a reliable and consistent database. The fast-paced enterprises and organizations generate terabytes of transactional data daily while working in an OLTP system, NewSQL is the ideal choice. NewSQL improves upon SQL by providing horizontal scalability whilst maintaining ACID properties. This facilitates working with Big Data by implementing concurrency, it also does well with ACID compliance. NewSQL thus seems to have found the sweet spot between speed, scalability, consistency, and availability Even though it is still in its nascent stage, NewSQL ticks all the right boxes to be an ideal database for Big Data and OLTP applications, as of now. You can also explore the difference between Virlet and Kubevirt in this blog.

How Can XenonStack Help You?

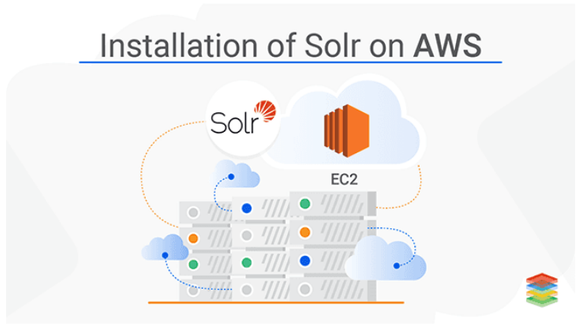
With XenonStack Database Managed Services, you get 24×7 access to our DBAs, architects, and engineers to handle day to day activities. XenonStack Database Managed Services offers Database Monitoring, Incident Management, DB Performance Tuning, Backup/Recovery Management, Database Security Management, Database Security Audits, and more. Get in Touch with us.

Tags

Big Data Management , Databases

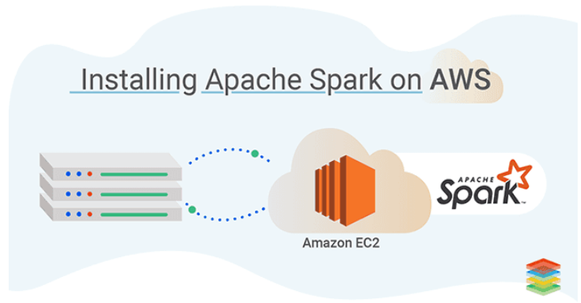
Share

Related Posts



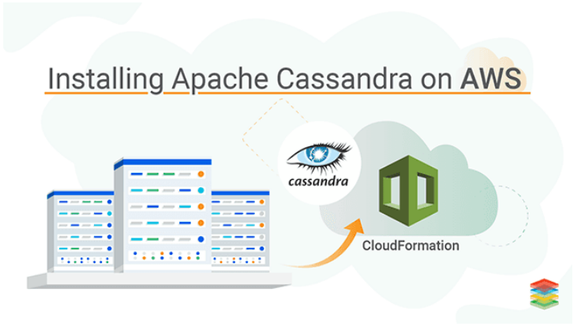
BIG DATA ENGINEERING | May 21, 2019

Apache Solr Architecture, Installation of Solr on AWS



BIG DATA ENGINEERING | May 18, 2019

Overview of Apache Spark, Installation of Spark on AWS



BIG DATA ENGINEERING | May 17, 2019

What is Apache Cassandra, Installation of Cassandra on AWS

Leave a Comment

Name *

Name

Email Address *

Email address

Comment *

Your Comment

Submit

Browse by Tags

Advanced Analytics
Artificial Neural Networks
Automation Tools
Behavior Driven Development
Big Data Applications
Big Data Development
Big Data Integration
Big Data Management
Big Data Services
Big Data Solutions
Big Data Storage
Big Data Strategy
Big Data Visualization
Bitcoin
BlockChain
Business Analytics
Business Intelligence
Careers
CircleCI
Cloud Computing
Continuous Delivery
Continuous Deployment
Continuous Integration
Continuous Testing
Cryptocurrency
Data Catalog
Data Streaming Tools
Data Visualization Techniques
Data Visualization Tools
Data Wrangling
Databases
Deep Learning
Deep Learning Visualization Tools
Devops Cloud
DevOps Security
DevOps Tools

Microservices
Predictive Modeling
Reactive Programming
Real Time Analytics
Semantic Analysis
Storage
Streaming Data Analytics
Test Driven Development
Testing
Time Series Prediction
Visual Analytics
Visualizing ML Models

XENONSTACK
A Stack Innovator