V skills
CERTIFIED
www.vskills.com

Apache Cassandra Sample Material
VS-1046

Vskills Certifications

Vskills Reading Material

Skills for a secure future

1. INTRODUCTION TO NOSQL

NoSQL databases try to offer certain functionality that more traditional relational database management systems do not. Whether it is for holding simple key-value pairs for shorter lengths of time for caching purposes, or keeping unstructured collections (e.g. collections) of data that could not be easily dealt with using relational databases and the structured query language (SQL) – they are here to help.

1.1. NoSQL Basics

A NoSQL (originally referring to "non SQL", "non relational" or "not only SQL") database provides a mechanism for storage and retrieval of data which is modeled in means other than the tabular relations used in relational databases. Such databases have existed since the late 1960s, but did not obtain the "NoSQL" moniker until a surge of popularity in the early twenty-first century, triggered by the needs of Web 2.0 companies such as Facebook, Google, and Amazon.com. NoSQL databases are increasingly used in big data and real-time web applications. NoSQL systems are also sometimes called "Not only SQL" to emphasize that they may support SQL-like query languages.

Motivations for this approach include: simplicity of design, simpler "horizontal" scaling to clusters of machines (which is a problem for relational databases), and finer control over availability. The data structures used by NoSQL databases (e.g. key-value, wide column, graph, or document) are different from those used by default in relational databases, making some operations faster in NoSQL. The particular suitability of a given NoSQL database depends on the problem it must solve. Sometimes the data structures used by NoSQL databases are also viewed as "more flexible" than relational database tables.

Many NoSQL stores compromise consistency (in the sense of the CAP theorem) in favor of availability, partition tolerance, and speed. Barriers to the greater adoption of NoSQL stores include the use of low-level query languages (instead of SQL, for instance the lack of ability to perform ad-hoc joins across tables), lack of standardized interfaces, and huge previous investments in existing relational databases.] Most NoSQL stores lack true ACID transactions, although a few databases, such as MarkLogic, Aerospike, FairCom c-treeACE, Google Spanner (though technically a NewSQL database), Symas LMDB, and OrientDB have made them central to their designs.

Instead, most NoSQL databases offer a concept of "eventual consistency" in which database changes are propagated to all nodes "eventually" (typically within milliseconds) so queries for data might not return updated data immediately or might result in reading data that is not accurate, a problem known as stale reads. Additionally, some NoSQL systems may exhibit lost writes and other forms of data loss. Fortunately, some NoSQL systems provide concepts such as write-ahead logging to avoid data loss. For distributed transaction processing across multiple databases, data consistency is an even bigger challenge that is difficult for both NoSQL and relational databases. Even current relational databases "do not allow referential integrity constraints to span databases." There are few systems that maintain both ACID transactions and X/Open XA standards for distributed transaction processing.

Types and examples of NoSQL databases

There have been various approaches to classify NoSQL databases, each with different categories and subcategories, some of which overlap. What follows is a basic classification by data model, with examples:

- ✓ Column: Accumulo, Cassandra, Druid, HBase, Vertica, SAP HANA
- ✓ Document: Apache CouchDB, ArangoDB, Clusterpoint, Couchbase, DocumentDB, HyperDex, IBM Domino, MarkLogic, MongoDB, OrientDB, Qizx, RethinkDB
- ✓ Key-value: Aerospike, ArangoDB, Couchbase, Dynamo, FairCom c-treeACE, FoundationDB, HyperDex, MemcacheDB, MUMPS, Oracle NoSQL Database, OrientDB, Redis, Riak, Berkeley DB
- ✓ Graph: AllegroGraph, ArangoDB, InfiniteGraph, Apache Giraph, MarkLogic, Neo4J, OrientDB, Virtuoso, Stardog
- ✓ Multi-model: Alchemy Database, ArangoDB, CortexDB, Couchbase, FoundationDB, MarkLogic, OrientDB

By design, NoSQL databases and management systems are relation-less (or schema-less). They are not based on a single model (e.g. relational model of RDBMSs) and each database, depending on their target-functionality, adopt a different one.

There are almost a handful of different operational models and functioning systems for NoSQL databases.:

- ✓ Key / Value: e.g. Redis, MemcacheDB, etc.
- ✓ Column: e.g. Cassandra, HBase, etc.
- ✓ Document: e.g. MongoDB, Couchbase, etc
- ✓ Graph: e.g. OrientDB, Neo4J, etc.

In order to better understand the roles and underlying technology of each database management system, let's quickly go over these four operational models.

Key / Value Based

We will begin our NoSQL modeling journey with key / value based database management simply because they can be considered the most basic and backbone implementation of NoSQL.

These type of databases work by matching keys with values, similar to a dictionary. There is no structure nor relation. After connecting to the database server (e.g. Redis), an application can state a key (e.g. the_answer_to_life) and provide a matching value (e.g. 42) which can later be retrieved the same way by supplying the key.

Key / value DBMSs are usually used for quickly storing basic information, and sometimes not-so-basic ones after performing, for example, a CPU and memory intensive computation. They are extremely performant, efficient and usually easily scalable.

When it comes to computers, a dictionary usually refers to a special sort of data object. They constitutes of arrays of collections with individual keys matching values.

Column Based

Column based NoSQL database management systems work by advancing the simple nature of key / value based ones.

Despite their complicated-to-understand image on the internet, these databases work very simply by creating collections of one or more key / value pairs that match a record.

Unlike the traditional defines schemas of relational databases, column-based NoSQL solutions do not require a pre-structured table to work with the data. Each record comes with one or more columns containing the information and each column of each record can be different.

Basically, column-based NoSQL databases are two dimensional arrays whereby each key (i.e. row / record) has one or more key / value pairs attached to it and these management systems allow very large and un-structured data to be kept and used (e.g. a record with tons of information).

These databases are commonly used when simple key / value pairs are not enough, and storing very large numbers of records with very large numbers of information is a must. DBMS implementing column-based, schema-less models can scale extremely well.

Document Based

Document based NoSQL database management systems can be considered the latest craze that managed to take a lot of people by storm. These DBMS work in a similar fashion to column-based ones; however, they allow much deeper nesting and complex structures to be achieved (e.g. a document, within a document, within a document).

Documents overcome the constraints of one or two level of key / value nesting of columnar databases. Basically, any complex and arbitrary structure can form a document, which can be stored using these management systems.

Despite their powerful nature, and the ability to query records by individual keys, document based management systems have their own issues and downfalls compared to others. For example, retrieving a value of a record means getting the whole lot of it and same goes for updates, all of which affect the performance.

Graph Based

Finally, the very interesting flavour of NoSQL database management systems is the graph based ones.

The graph based DBMS models represent the data in a completely different way than the previous three models. They use tree-like structures (i.e. graphs) with nodes and edges connecting each other through relations.

Similarly to mathematics, certain operations are much simpler to perform using these type of models thanks to their nature of linking and grouping related pieces of information (e.g. connected people).

These databases are commonly used by applications whereby clear boundaries for connections are necessary to establish. For example, when you register to a social network of any sort, your friends' connection to you and their friends' friends' relation to you are much easier to work with using graph-based database management systems.

There are following properties of NoSQL databases.

- ✓ Design Simplicity
- ✓ Horizontal Scaling
- ✓ High Availability

Data structures used in Cassandra are more specified than data structures used in relational databases. Cassandra data structures are faster than relational database structures.
NoSQL databases are increasingly used in Big Data and real-time web applications. NoSQL databases are sometimes called Not Only SQL i.e. they may support SQL-like query language.

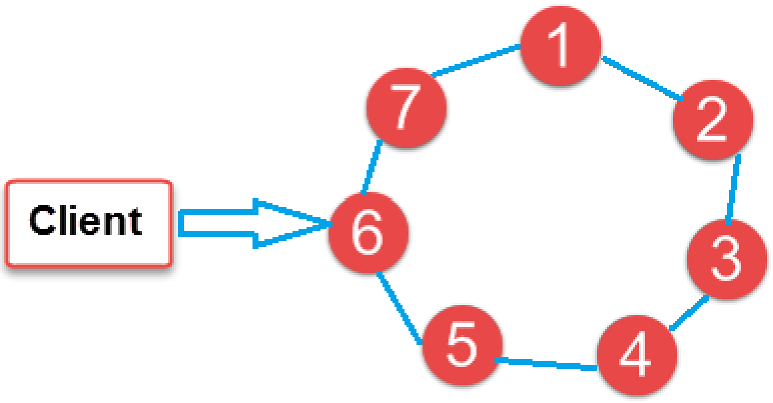
Nosql Vs RDBMS

Here are the differences between relation databases and NoSQL databases in a tabular format.

Relational Database	NoSQL Database
Handles data coming in low velocity	Handles data coming in high velocity
Data arrive from one or few locations	Data arrive from many locations
Manages structured data	Manages structured unstructured and semi-structured data.
Supports complex transactions (with joins)	Supports simple transactions
single point of failure with failover	No single point of failure
Handles data in the moderate volume.	Handles data in very high volume
Centralized deployments	Decentralized deployments
Transactions written in one location	Transaction written in many locations
Gives read scalability	Gives both read and write scalability
Deployed in vertical fashion	Deployed in Horizontal fashion

1.2. Cassandra Basics and Terminology

Apache Cassandra is highly scalable, distributed and high-performance NoSQL database. Cassandra is designed to handle a huge amount of data.



In the image above, circles are Cassandra nodes and lines between the circles shows distributed architecture, while the client is sending data to the node. Cassandra handles the huge amount of data with its distributed architecture. Data is placed on different machines with more than one replication factor that provides high availability and no single point of failure.

Cassandra History

- ✓ Cassandra was first developed at Facebook for inbox search.
- ✓ Facebook open sourced it in July 2008.
- ✓ Apache incubator accepted Cassandra in March 2009.
- ✓ Cassandra is a top level project of Apache since February 2010.
- ✓ The latest version of Apache Cassandra is 3.2.1.

The 3.0 release was made available in November 2015. It includes features are

- ✓ The underlying storage engine has been rewritten to more closely match CQL constructs
- ✓ Support for materialized views (sometimes also called global indexes)
- ✓ Java 8 is now the supported version
- ✓ The Thrift-based Command Line Interface (CLI) is removed

Apache Cassandra Features

There are main features of Cassandra are

- ✓ Massively Scalable Architecture: Cassandra has a masterless design where all nodes are at the same level which provides operational simplicity and easy scale out.
- ✓ Masterless Architecture: Data can be written and read on any node.
- ✓ Linear Scale Performance: As more nodes are added, the performance of Cassandra increases.
- ✓ No Single point of failure: Cassandra replicates data on different nodes that ensures no single point of failure.
- ✓ Fault Detection and Recovery: Failed nodes can easily be restored and recovered.
- ✓ Flexible and Dynamic Data Model: Supports datatypes with Fast writes and reads.

- ✓ Data Protection: Data is protected with commit log design and build in security like backup and restore mechanisms.
- ✓ Tunable Data Consistency: Support for strong data consistency across distributed architecture.
- ✓ Multi Data Center Replication: Cassandra provides feature to replicate data across multiple data center.
- ✓ Data Compression: Cassandra can compress up to 80% data without any overhead.
- ✓ Cassandra Query language: Cassandra provides query language that is similar like SQL language. It makes very easy for relational database developers moving from relational database to Cassandra.

Application of Cassandra

Cassandra is a non-relational database that can be used for different types of applications. Here are some use cases where Cassandra should be preferred.

- ✓ Messaging - Cassandra is a great database for the companies that provides mobile phones and messaging services. These companies have a huge amount of data, so Cassandra is best for them.
- ✓ Internet of things Application - Cassandra is a great database for the applications where data is coming at very high speed from different devices or sensors.
- ✓ Product Catalogs and retail apps - Cassandra is used by many retailers for durable shopping cart protection and fast product catalog input and output.
- ✓ Social Media Analytics and recommendation engine - Cassandra is a great database for many online companies and social media providers for analysis and recommendation to their customers.

Distributed Database

Cassandra is distributed, which means that it is capable of running on multiple machines while appearing to users as a unified whole. In fact, there is little point in running a single Cassandra node. Although you can do it, and that's acceptable for getting up to speed on how it works, you quickly realize that you'll need multiple machines to really realize any benefit from running Cassandra. Much of its design and code base is specifically engineered toward not only making it work across many different machines, but also for optimizing performance across multiple data center racks, and even for a single Cassandra cluster running across geographically dispersed data centers. You can confidently write data to anywhere in the cluster and Cassandra will get it.

Once you start to scale many other data stores (MySQL, Bigtable), some nodes need to be set up as masters in order to organize other nodes, which are set up as slaves. Cassandra, however, is decentralized, meaning that every node is identical; no Cassandra node performs certain organizing operations distinct from any other node. Instead, Cassandra features a peer-to-peer protocol and uses gossip to maintain and keep in sync a list of nodes that are alive or dead.

The fact that Cassandra is decentralized means that there is no single point of failure. All of the nodes in a Cassandra cluster function exactly the same. This is sometimes referred to as "server symmetry." Because they are all doing the same thing, by definition there can't be a special host that is coordinating activities, as with the master/ slave setup that you see in MySQL, Bigtable, and so many others.

Decentralization, therefore, has two key advantages: it's simpler to use than master/slave, and it helps you avoid outages. It can be easier to operate and maintain a decentralized store than a master/slave store because all nodes are the same. That means that you don't need any special knowledge to scale; setting up 50 nodes isn't much different from setting up one. There's next to no configuration required to support it.

Moreover, in a master/slave setup, the master can become a single point of failure (SPOF). To avoid this, you often need to add some complexity to the environment in the form of multiple masters. Because all of the replicas in Cassandra are identical, failures of a node won't disrupt service.

Elastic Scalability

Scalability is an architectural feature of a system that can continue serving a greater number of requests with little degradation in performance. Vertical scaling—simply adding more hardware capacity and memory to your existing machine—is the easiest way to achieve this. Horizontal scaling means adding more machines that have all or some of the data on them so that no one machine has to bear the entire burden of serving requests. But then the software itself must have an internal mechanism for keeping its data in sync with the other nodes in the cluster.

Elastic scalability refers to a special property of horizontal scalability. It means that your cluster can seamlessly scale up and scale back down. To do this, the cluster must be able to accept new nodes that can begin participating by getting a copy of some or all of the data and start serving new user requests without major disruption or reconfiguration of the entire cluster. You don't have to restart your process. You don't have to change your application queries. You don't have to manually rebalance the data yourself. Just add another machine—Cassandra will find it and start sending it work.

Consistency

Consistency essentially means that a read always returns the most recently written value. Consider two customers are attempting to put the same item into their shopping carts on an ecommerce site. If I place the last item in stock into my cart an instant after you do, you should get the item added to your cart, and I should be informed that the item is no longer available for purchase. This is guaranteed to hap pen when the state of a write is consistent among all nodes that have that data.

But as we'll see later, scaling data stores means making certain trade-offs between data consistency, node availability, and partition tolerance. Cassandra is frequently called "eventually consistent," which is a bit misleading. Out of the box, Cassandra trades some consistency in order to achieve total availability. But Cassandra is more accurately termed "tuneably consistent," which means it allows you to easily decide the level of consistency you require, in balance with the level of availability.

Certifications

Accounting, Banking and Finance
– Certified AML-KYC Compliance Officer
– Certified Business Accountant
– Certified Commercial Banker
– Certified Foreign Exchange Professional
– Certified GAAP Accounting Standards Professional
– Certified Financial Risk Management Professional
– Certified Merger and Acquisition Analyst
– Certified Tally 9.0 Professional
– Certified Treasury Market Professional
– Certified Wealth Manager

Big Data
– Certified Hadoop and Mapreduce Professional

Cloud Computing
– Certified Cloud Computing Professional

Design
– Certified Interior Designer

Digital Media
– Certified Social Media Marketing Professional
– Certified Inbound Marketing Professional
– Certified Digital Marketing Master

Foreign Trade
– Certified Export Import (Foreign Trade) Professional

Health, Nutrition and Well Being
– Certified Fitness Instructor

Hospitality
– Certified Restaurant Team Member (Hospitality)

Human Resources
– Certified HR Compensation Manager
– Certified HR Stafffing Manager
– Certified Human Resources Manager
– Certified Performance Appraisal Manager

Office Skills
– Certified Data Entry Operator
– Certified Office Administrator

Project Management
– Certified Project Management Professional

Real Estate
– Certified Real Estate Consultant

Marketing
– Certified Marketing Manager

Quality
– Certified Six Sigma Green Belt Professional
– Certified Six Sigma Black Belt Professional
– Certified TQM Professional

Logistics & Supply Chain Management
– Certified International Logistics Professional
– Certified Logistics & SCM Professional
– Certified Purchase Manager
– Certified Supply Chain Management Professional

Legal
– Certified IPR & Legal Manager
– Certified Labour Law Analyst
– Certified Business Law Analyst
– Certified Corporate Law Analyst

Information Technology
– Certified ASP.NET Programmer
– Certified Basic Network Support Professional
– Certified Business Intelligence Professional
– Certified Core Java Developer
– Certified E-commerce Professional
– Certified IT Support Professional
– Certified PHP Professional
– Certified Selenium Professional
– Certified SEO Professional
– Certified Software Quality Assurance Professional

Mobile Application Development
– Certified Android Apps Developer
– Certified iPhone Apps Developer

Security
– Certified Ethical Hacking and Security Professional
– Certified Network Security Professional

Management
– Certified Corporate Goverance Professional
– Certified Corporate Social Responsibility Professional

Life Skills
– Certified Business Communication Specialist
– Certified Public Relations Officer

Media
– Certified Advertising Manager
– Certified Advertising Sales Professional

Sales, BPO
– Certified Sales Manager
– Certified Telesales Executive

& many more job related certifications

Contact us at :

Vskills

011-473 44 723 or **info@vskills.in**

www.vskills.com