Datenbank Implementationen

Das E-Book.

at the Cooperative State University Baden-Württemberg Stuttgart

by

IT14C

April 2017

Contents

1 Riak

Riak is available in two versions. There is Riak TS for Time Series Data and Riak KV. The chapter about Riak concentrates on Riak KV (further "Riak").

1.1 Introducion into Riak

The following chapter will give you an introduction into Riak and its main features.

1.1.1 General Information

Riak is a distributed key-value NoSQL database which is designed for high availability. As long as the client can reach one Riak server the data is available since data is saved across multiple servers. How the clustering works will be shown in the next chapter.
Riak is available for different operating systems, e.g. Ubuntu, CentOS or Mac OS X but not for Windows. The installation is straight-forward because you just have to download a package from the official website and install the package.
As data is saved across multiple servers even hardware or network failures can be handled by Riak. If you need more space for your data new servers can be added easily. By adding new servers the scalability is nearly linear which is very impressive.
Data is saved in buckets. A bucket in Riak can be compared with a table in a SQL-database.
Now if you have a look on the CAP-Theorem one can say that Riak definitely concentrates on "A" and "P" - Availability and Partition Tolerance. If your system needs a high availability and you can't accept downtime Riak is probably the best solution. Another feature of Riak is its latency: since the CRUD-operations don't involve complex joins the requests are serviced promptly.
On the other hand if your system needs a high consistency of the data Riak is not the right choice.

1.1.2 Riak Clustering

The high availability of Riak can only be achieved by the Riak clustering. The official website of Riak recommends that there should be at least 5 nodes in one cluster. A node is a server in production environment - during the development of the software there can be more than one node on a server.
All nodes have the same responsibility, so there is no kind of master-node which has special tasks.

The clustering is visible in the logo of Riak, too. There is one node and three lines to three other nodes which symbolizes the replication of data:

Figure 1.1: Riak Logo

Automatic re-distribution of data

When new servers are added or when machines are removed Riak automatically re-distributes the data with no downtime. Data is spread in the cluster until each node owns the same amount of data. So developers don't need to care about where the data is saved. Riak uses consistent hashing to distribute data evenly across the nodes in a cluster. Consistent hashing limits the reshuffling of keys when a hash-table structure is rebalanced.

Intelligent Replication

Even if nodes go down the user should be able to read and write data. The replication scheme ensures this by setting a replication variable, that specifies the number of nodes on which a value will be replicated. The default number is 3 which means that each object is replicated 3 times. If Riak can access one node there the object is replicated it's available for the client.
The following picture describes the replication:

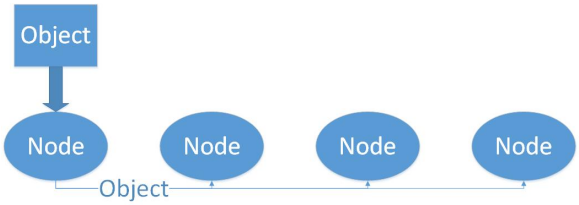

Figure 1.2: Riak Clustering

1.2 Open Source vs. Enterprise

Basho provides five different versions of the Riak KV database, so that the customers always find the perfect configuration for their purpose. The variants are named *Open Source*, *Developer*, *Pro*, *Enterprise* and *Enterprise Plus*. Every configuration has a different composition of features. Figure 1.3 shows a short overview of the versions and the related features.

	RIAK KV OPEN SOURCE	RIAK KV DEVELOPER	RIAK KV PRO	RIAK KV ENTERPRISE	RIAK KV ENTERPRISE PLUS
KEY VALUE DATA MODULE	✓	✓	✓	✓	✓
MASTERLESS WITH BUILT-IN REPLICATION	✓	✓	✓	✓	✓
HTTP API AND PROTOCOL BUFFERS	✓	✓	✓	✓	✓
SEARCH: FULL-TEXT, INTEGRATED, SOLR, SECONDARY INDEXES	✓	✓	✓	✓	✓
RIAK DATA TYPES (DISTRIBUTED COUNTERS, SETS, MAPS)	✓	✓	✓	✓	✓
MULTI-CLUSTER REPLICATION		✓		✓	✓
SNMP/JMX SUPPORT		✓		✓	✓
BASHO BASELINE AND SYSTEM ASSESSMENT					✓
BASHO ENGINEERING SUPPORT		Business Hours	Business Hours	24x7x365	24x7x365
ONSITE REVIEW AND SYSTEM ASSESSMENT					2x/year
ONLINE TICKET TRACKING		✓	✓	✓	✓
EMERGENCY PATCHES				✓	✓
SLA		24 hour	4 hour	1 hour	30 minutes
LICENSE TYPE	Apache 2	Commercial	Apache 2	Commercial	Commercial

Figure 1.3: Overview of different versions of Riak KV (Basho, 01.04.2017)

In the following, every feature is individually viewed and explained in detail.

Key Value Data Module

Of course every configuration uses the key value data module developed by Basho.

Masterless with Built-In Replication

All configurations are masterless with an integrated replication. This means that data is replicated automatically on multiple nodes so that the application remains available for both read and write operations. There is no single master and no single point of failure. This is the way the database achieves high availability. (Basho, 01.04.2017)

HTTP API and Protocol Buffers

Every version works with a simple HTTP API and Protocol Buffers. Protocol Buffers is a method of serializing structured data and is especially useful for storing data. It is developed by Google. (Google Developers, 06.04.2017)

Search

All implementations of the Riak KV support an integrated fulltext search and Apache Solr. Apache Solr is a popular open source enterprise search platform built on Apache Lucene. This means with Apache Solr the user can search the whole database at once. (The Apache Software Foundation, 06.04.2017)

Riak Data Types

Certainly all versions use the Riak data types which are *Flags*, *Registers*, *Counters*, *Sets* and *Maps*. Flags are similar the same as Boolean, except the values are called *enable* and *disable*. Registers are essentially named binaries like Strings. Flags and Registers are no bucket-level Riak data types. They cannot be used on their own and have to be embedded in Maps. Counters keep track of increments or decrements. Sets are collections of unique binary values. Maps enable the creation of complex, custom data types because all other data types could be embedded. (Basho, 06.04.2017)

Multi-Cluster Replication

This feature is just available for the commercial versions of Riak. The data clusters are replicated automatically in several datacenters of the customer. If one cluster fails another one provides the necessary data. (Basho, 01.04.2017)

SNMP / JMX Support

SNMP means *Simple Network Management Protocol* and is a protocol for collecting and organizing information about managed devices on networks. (L8 ManeValidus, 06.04.2017) JMX stands for *Java Management Extensions* and is a Java technology that provides tools for managing und monitoring applications, devices and system objects. (Bryan ssm, 06.04.2017) Customers that use a commercial configuration of Riak are able to implement these extensions and monitor their database.

Basho Baseline and System Assessment

This feature is obtainable just for customers of the Enterprise Plus package. An engineer of the Basho team reviews the whole configuration of the database via remote access before Riak is deployed. (Basho, 01.04.2017)

Basho Engineering Support

The Basho Engineering Support is a simple support hotline. Since the Basho team only provides payed support, this offer is not available for the open source variant. The user has to have an account to contact the support team. (Basho, 01.04.2017)

Onsite Review and System Assessment

That system assessment is nearly the same as the previous one. The only difference is that the assessment is not done via remote access, but a team of engineers come to the customer's location and review the system onsite. (Basho, 01.04.2017)

Online Ticket Tracking

The Online Ticket Tracking is a functionality of the account that is needed if you do not use the open source version. In combination with the ability to contact the support team, the user can see the current state of his support ticket. (Basho, 01.04.2017)

Emergency Patches

If the customers of the Enterprise versions face a problem with their database, the Basho team provides a patch as soon as possible. (Basho, 01.04.2017)

Service Level Agreements

The Service Level Agreements are between 24 hours and 30 minutes response time after a problem report was sent by the customer. Afterwards the engineering team provides a solution for the problem as soon as possible. (Basho, 01.04.2017)

License Type

The last point of this comparsion is the license type. Riak KV Open Source and Riak KV Pro are available under the open source license Apache 2. The other three configurations use a commercial license. (Basho, 01.04.2017)

1.3 Advantages

Since Riak is a distributed database there are different advantages making Riak special. Riak focuses on high availability, easy scalability and data safety. This is achieved through the distribution of the database across several nodes. The main advantages of the distributed approach are:

- Installation
- Scalability
- Availability
- Interaction
- Error management

In the following subsections this concepts will be described in more detail.

1.3.1 Installation

The installation of Riak is easy and straight forward. Riak is available for various Linux distributions and Mac OS X. There is an Dabian package for Ubuntu, delivered through bashos web page (HTTP://docs.Bashocom/riak/kv/2.2.2/downloads/). This package can be easily installed with Ubuntus package manager. After the setup you are ready to go. With the command *"riak start"* a Riak cluster with one node and the standard settings is started.

1.3.2 Scalability

Riak uses a distributed cluster approach built up of several nodes which makes it highly scalable. If there is a need for higher performance or stability there could be easily added several nodes to the cluster. This can be done even when the database is running. Before a node can be added to a cluster it needs to be started with the *"riak start"* command. After the node is running it can be added to a cluster with the *"riak cluster join"* command.

1.3.3 Availability

A special feature of Riak is its high availability Riak uses a masterless system resulting in no single point of failure. If all nodes fail except one this last node will take all of the responsibilities of the other nodes. The cluster gets very slow, but it stays available and responsive.

1.3.4 Interaction

Another unique selling point of Riak is its native HTTP 1.1 API. This API is designed as RESTful Web service. Create, read, update and delete (CRUD) actions can be performed over the corresponding HTTP methods. This makes Riak a very flexible and handy database. Besides that Riak guarantees client support for common programming languages like Java, Ruby, Python, C#, Node.js, PHP, Erlang and Go.

1.3.5 Error management

If multiple clients can write concurrently, potentially to the same key it is very likely that errors could happen. Therefore Riak has to use an error management. There is an logical approach called vector clock which abstracts the states of a data set on an analogous clock to track the history of updates to value. If the data is corrupted Riak through conflict writes, it can be restored through this mechanism.

1.4 Disadvantages

As the previous section shows is the distributed approach of Riak a very beneficial and useful. But besides this advantages there are some downsides as well. Several nodes and the masterless concept lead to various problems like:

- Inconsistency
- Vector-clocks
- Data chaining
- No rollbacks

In the next subsections we dig a little bit deeper in those problems.

1.4.1 Inconsistency

The main problem of highly-available, clustered systems like Riak is the validity of data sets. Due to the fact that there are no ACID transactions data sets can get inconsistent. There is no mechanism guaranteeing the transfer of a consistent state into another. This results in conflicting responses and anomalies which have to be handled.

1.4.2 Vector-clocks

Although the error management concept of Riak is very reasoned the vector-clock system is leading to some difficulties. It is possible that Riak creates different values for an object on various nodes. These values are called siblings. This could lead to inconsistency and conflicts.

1.4.3 Data chaining

1.4.4 No rollbacks

1.5 Conclusion

As a conclusion one can say that the feature "CRUD"-Operations via REST is very useful but in newer versions **C**ross **O**rigin **R**esource **S**haring should be available. The example project could have been easier if CORS would have been available as you could send HTTP requests directly from the front-end to the database and there would be no need of a back-end.
Furthermore Riak is a **distributed**, **scalable** and **fault-tolerant** NoSQL database. Use cases are mostly applications where high availability of data is the most important point. Another use case are applications with fast growing data as you can add new servers/nodes to your cluster and the data is replicated automatically.
As already described in the chapter "Use Cases" Riak is especially useful for session data, documents, chat applications and business continuity as all of the use cases need a high availability of the data.
You should not use Riak if you expect the database to be always consistent since consistency is not possible because of the CAP-Theorem where Riak concentrates on **A**vailability and **P**artition Tolerance.

References

Basho. (01.04.2017). *Riak kv.* Bellevue, Washington. Retrieved 01.04.2017, from http://basho.com/products/riak-kv/

Basho. (06.04.2017). *Riak kv documentation.* Bellevue, Washington. Retrieved 06.04.2017, from http://docs.basho.com/riak/kv/2.2.3/

Bryan ssm. (06.04.2017). *Java management extensions.* San Francisco, California. Retrieved 06.04.2017, from https://en.wikipedia.org/wiki/Java_Management_Extensions

Google Developers. (06.04.2017). *Protocol buffers.* Mountain View, California. Retrieved 06.04.2017, from https://developers.google.com/protocol-buffers/

L8 ManeValidus. (06.04.2017). *Simple network management protocol.* San Francisco, California. Retrieved 06.04.2017, from https://de.wikipedia.org/wiki/Simple_Network_Management_Protocol

The Apache Software Foundation. (06.04.2017). *Apache solr.* Forest Hill, Maryland. Retrieved 06.04.2017, from http://lucene.apache.org/solr/