OULU UNIVERSITY OF
APPLIED SCIENCES

Mikko Tuominen

ArangoDB database server for gathering data from social medias

ArangoDB database server for gathering data from social medias

Mikko Tuominen
Thesis
Autumn 2018
Business Information Systems
Oulu University of Applied Sciences

ABSTRACT

Oulu University of Applied Sciences
“Business Information Systems”, “Computer Systems Expertise”

Author(s): Mikko Tuominen
Title of Bachelor´s thesis: ArangoDB database server for gathering data from social medias
Supervisor(s): Jukka Kaisto
Term and year of completion: Autumn 2018 Number of pages: 33 + 2

St. Pölten University of Applied Sciences requested me, the author, to make database server for test usage for one of their projects. This thesis covers how the server was installed, gives insight to the chosen database and reviews its capabilities in the future usage.

Constructive research method was chosen, with some theory and practice. Foundation for theory is acquired from several books, documentations and related webpages.

Main objective of the thesis was to create stable, easy to manage and well secured server for the project. This was achieved, and server was taken to use in the project.

Keywords: ArangoDB, database scalability, guide

CONTENTS

1 INTRODUCTION

The goal of the thesis project is to create stable, well secured and remotely manageable multi-model database test server with automatic backups for the St. Pölten University of Applied Science project called Smart dAta for Music Business Administration (SAMBA). Also, review database performance and scalability. In the project SAMBA, unstructured data is collected from the social media sites, for example YouTube comments. Collecting of the data is made by Python scripts running on the server, which are provided by initiator of the thesis project MSc. Alexis Ringot (later Mr. Ringot), and then gathered data is analyzed with queries. Data amounts may not be that huge on test server that it would be called Big Data, but in production environment that might be the case.

"The aim of SAMBA is to make unstructured data (especially comments) from social media and sharing platforms usable for strategic management. The project specializes on the music industry since there is a multitude of unstructured data available and not being used in this sector so far. Researchers will develop tools to help music companies monitoring songs, artists and trends, implementing a suitable market segmentation and evaluating the life time cycles of songs, artists, and genres."
(St. Pölten University of Applied Sciences 2018. Cited 17.9.2018).

Thesis initiator's requirements

Initial plan for the server was to Install Ubuntu server as OS (Operating System), which wouldn't have graphical user interface, and would have been only managed remotely by SSH (Secure Shell) connection. Then in my opinion it would have been most stable and secure. Mr. Ringot's requirements and infrastructure of the network changed the plans.

Mr. Ringot's requirements included GUI (Graphical User Interface) for the server and remote desktop access. Reason for having GUI on the server is that some scripting development is done using server and in Mr. Ringot's opinion the management of the server is easier for them, after my part of the project is done. Also, Mr. Ringot with his colleagues chose to use multi-model database ArangoDB as it's one of the most popular native multi-model databases and should have good performance and scalability. Goal was to try out both Windows and Linux environments and to use

TeamViewer for remote access, as server will be inside university LAN, without public IP address, which is maintained by independent IT-department.

2 MAIN CONCEPTS

In this chapter there are brief definitions and explanations of main concepts of the thesis. Everything is not explained in detail, but only the essential information is gathered here, with some examples.

2.1 Big data

Even when in the test server the data volume can't be called big data, when planning future production environment, where the data amounts could be that massive and unstructured, that term big data should be familiar.

One definition for big data is following "Data that is massive in volume, with respect to the processing system, with a variety of structured and unstructured data containing different data patterns to be analyzed."(Syed Muhammad Fahad Akhtar 2018, What is big data?) Main characteristics, or dimensions, of big data are often presented with V's of big data, which should differ big data from other data processing. Originally Doug Laney presented three V's volume, velocity, and variety in 2001. After that the three V's have become six with veracity, variability and value added to the dimensions, as seen on figure 1. Data doesn't need to have all these characteristics to be considered big data.

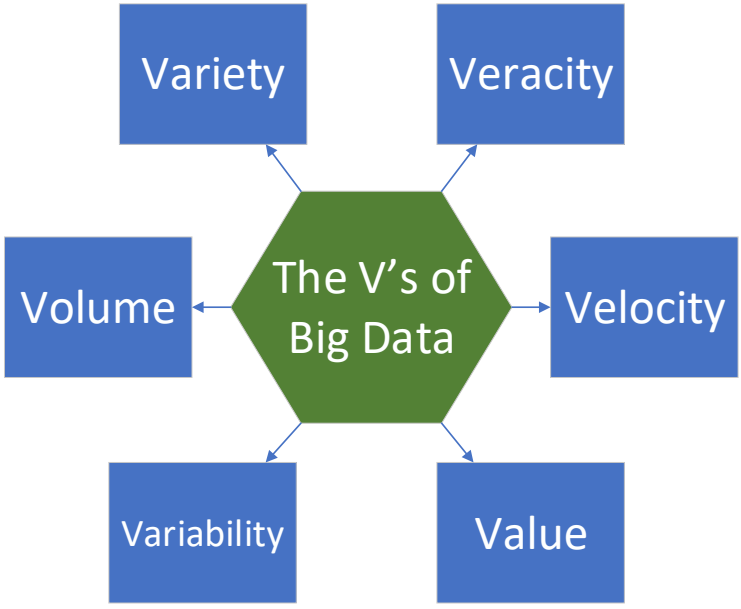


FIGURE 1. V's of Big Data

Volume

Data amounts are so massive, that traditional ways of gathering, storing and processing the data are insufficient. It can be already collected or continuously streamed. Just as an example of big data amounts, with 2 billion users Facebook ingests 600TB into its database daily, according to statistics provided by themselves.

(Syed Muhammad Fahad Akhtar 2018, Characteristics of big data)

Velocity

The rate of data that it's being generated, or how fast it's collected. When talking about velocity, one dimension is also to consider how fast the data gets pointless, even misleading, or is it valuable indefinitely.

(Syed Muhammad Fahad Akhtar 2018, Characteristics of big data)

Variety

Refers to form of the data. There are many different data types, for example it can be application data, streaming data, on-wire data format (JSON, MPEG, Avro), configuration files etc.

(V. Naresh Kumar; Prashant Shindgikar 2018, Data architecture principles)

Veracity

This dimension is about uncertainty of the data. How much value the big data have if you can't trust the output of the data analyze, for example. Main problem in high-velocity data or streaming data is that you might not have time to clean it, and you must take uncertainty to account while processing it.

(Syed Muhammad Fahad Akhtar 2018, Characteristics of big data)

Variability

"This vector of big data derives from the lack of consistency or fixed patterns in data. It's a different from Variety". Variability means that it matters if the meaning of the data keeps changing.

(Syed Muhammad Fahad Akhtar 2018, Characteristics of big data)

Value

Often considered the most important dimension. When investing a lot to the infrastructure to collect huge amounts of data there has to be some value in the data to the business.

(Syed Muhammad Fahad Akhtar 2018, Characteristics of big data)

2.2 Database Structures

There are different types of databases. In this sub-chapter there are introductions to them and examples of the structure.

Relational databases

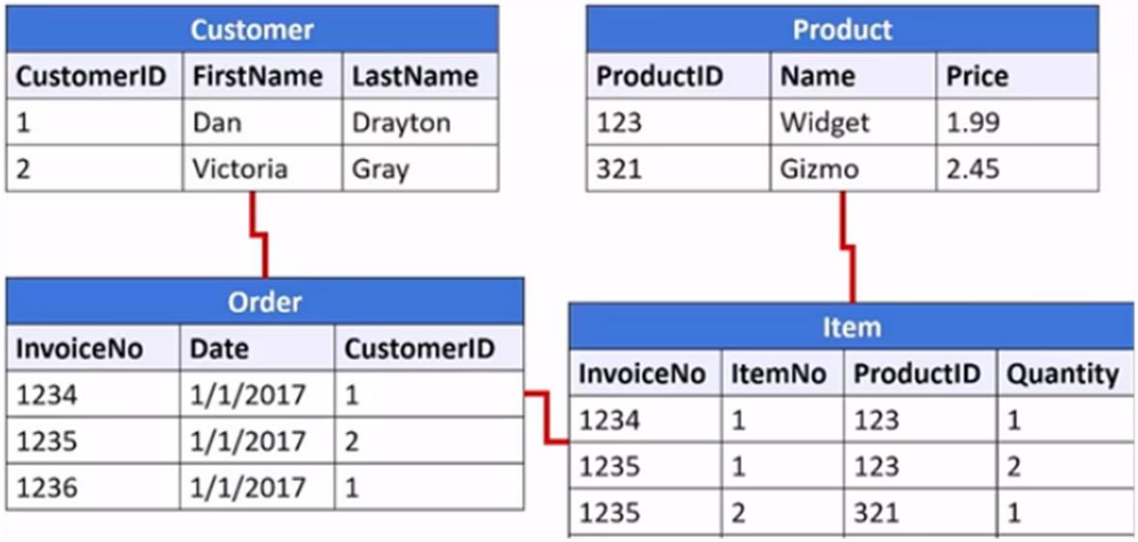
These databases use some variant of the Structured Query Language (SQL) and the data is then in structured in fixed fields in tables, which are connected through keys. They are handled mostly with a DBMS (DataBase Management System), which is the software that enables users and applications to interact with databases, not by API (Application Programming Interface). SQL-based databases have still own purposes, where they prevail over newer NoSQL (Not Only SQL) databases, but they are not meant to handle big data.
(Zacharias Voulgaris PhD 2017, Database Platforms)

Figure 2 gives an example of relational database schema with four tables. “Customer” table is connected to the “Order” table with the “CustomerID” key, “Order” and “Item” tables are connected through “InvoiceNo”. Lastly “Item” and “Product” tables are connected with “ProductID”.

Simple query would be for example

“SELECT ProductID, Name from Product WHERE Price > 2.00;”

Which would output “Gizmo”, because from “Product” table it’s only one which cost more than 2.00.



Customer

CustomerID	FirstName	LastName
1	Dan	Drayton
2	Victoria	Gray

Product

ProductID	Name	Price
123	Widget	1.99
321	Gizmo	2.45

Order

InvoiceNo	Date	CustomerID
1234	1/1/2017	1
1235	1/1/2017	2
1236	1/1/2017	1

Item

InvoiceNo	ItemNo	ProductID	Quantity
1234	1	123	1
1235	1	123	2
1235	2	321	1

FIGURE 2. Database example (Graeme Malcolm Relational Databases. Cited 17.9.2018)

NoSQL databases

Database mainly for unstructured data. Still, NoSQL doesn't mean that data is always in unstructured form, database just allows the data to be unstructured, but it can be also in structured form. Big data is always, at least should be, stored in these kinds of databases, because of their high speed, scalability and flexibility. Many times, queries are made in SQL like language, but handling is usually done through API, not DBMS.

(Zacharias Voulgaris PhD 2017, Database Platforms)

In figure 3 there is example of key-value data model (there are also other data models in NoSQL), which is very flexible way to store data. "key value pair can consist of any kind of unique key and any kind of value. The value could be a complex type like a JSON document, or even a set of columns that contain various properties of the data item."

(Graeme Malcolm Key-Value Data Stores. Cited 17.9.2018)

Key	Value
1	20,123.00
CustSvc	555-123-4567
G7371	{a: 1, b: 2}
dan@contoso.com	FirstName: Dan; LastName: Drayton; Phone: 555-321-7654

FIGURE 3. Key Value pair example (Graeme Malcolm Key-Value Data Stores. Cited 17.9.2018)

Multi-model databases

NoSQL database with multiple data models. Defining it perfectly is challenging, because many vendors call their databases as "multi-model". There is also difference between native (also called universal) multi-model database and when database has added layer, graph layer on top of a document database for example.

(ArangoDB Inc 2016, 3)

One short definition for a native multi-model database can be following: "A native multi-model database has one core, one query language, but multiple data models."

(ArangoDB Inc 2016, 3)

2.3 JSON

The JavaScript Object Notation (JSON) is human-readable, open-standard data interchange format, consisting any serializable value, often attribute-value pairs and array data types. It is a subset of JavaScript language and doesn't have any features that JavaScript doesn't possess. There are other programming languages that can parse and generate JSON-format data.
(Ben Smith 2015, Introducing JSON)

Small example of what JSON content could be:

```
{"menu": {
  "id": "file",
  "value": "File",
  "popup": {
    "menuitem": [
      {"value": "New", "onclick": "CreateNewDoc()"},
      {"value": "Open", "onclick": "OpenDoc()"},
      {"value": "Close", "onclick": "CloseDoc()"}
    ]
  }
}}
```

(json.org 2018. Cited 17.9.2018)

JSON is the default storage format of ArangoDB native multi-model database.
(ArangoDB Inc 2018, 66).

2.4 Cloud Services

Cloud Services are used over the internet, which can be for example different applications, servers or storage space. Services enable to user to have a lot of computing power or storage space, if needed, without having vast own local infrastructure. Good side of using cloud services is that you only pay from the resources you need to use.
(Sami Tavasti 2018, 7)

Service types are categorized to three different levels. These types are IaaS (Infrastructure as a Service), PaaS (Platform as a Service) and SaaS (Software as a Service) as seen in figure 4. (Éric Dusablon 2017. Cited 27.9.2018).

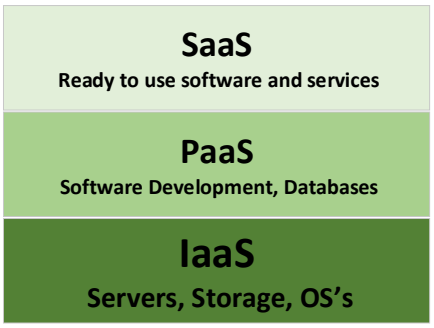


FIGURE 4. Service type levels (Éric Dusablon 2017. Cited 27.9.2018).

2.5 Server Clusters

Computer cluster consist multiple servers (nodes), often near each other and with same operating system running (many times identical hardware), connected with dedicated network to form centralized data processing unit. Cluster can distribute the workload across connected servers. Idea is to improve availability, scalability, performance, and fault tolerance.
(Vijay Shankar Upreti 2016, OS Clustering Concepts).

3 ARANGODB

ArangoDB is open source (Apache License 2.0) multi-model (NoSQL) database written in C++ and JavaScript. Development started at 2011, when it was called AvocadoDB, later changed to ArangoDB. Company behind ArangoDB is called ArangoDB Inc. Database supports three data models: key value pairs, documents and graphs. It has one database core and one unified query language; thus the native multi-model database term is used by its developers, often term universal database is used by others. Default storage format is JSON, internally it uses binary format "VelocyPack" for serialization and storage. It can store nested JSON object as a data entry inside a collection natively.
(ArangoDB Inc 2018, 44, 47, 66).

ArangoDB can be managed with Web Interface in a browser, as seen in figure 5, or with "arangosh" in a shell. One ArangoDB instance on a server can contain multiple databases (figure 5) which can have multiple collections.

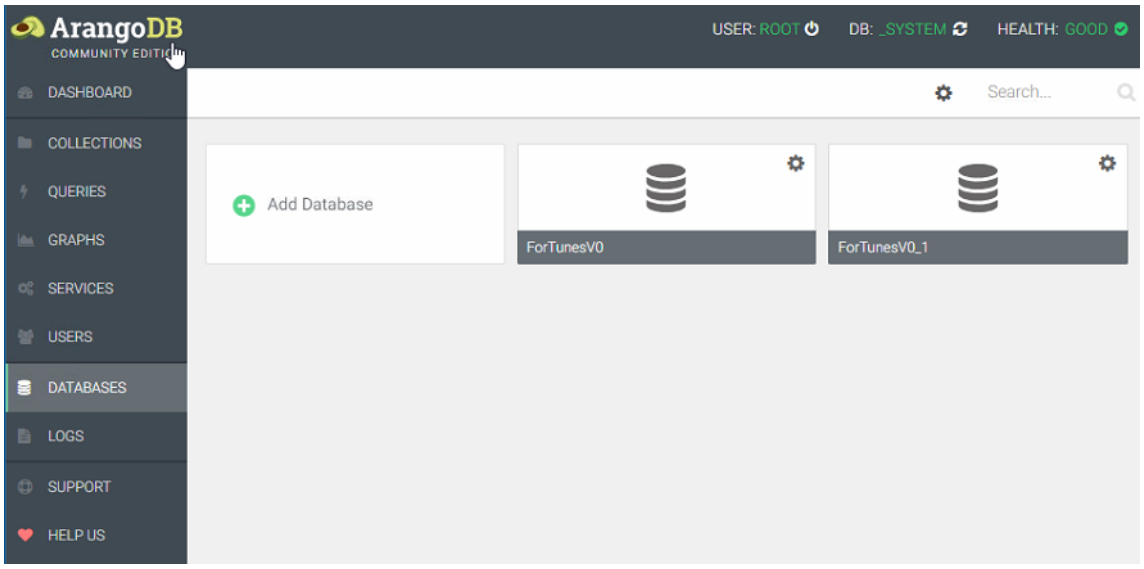


FIGURE 5. ArangoDB WebUI

3.1 Query Language

ArangoDB Query Language (AQL) is similar to SQL. AQL is mainly declarative, human-readable and pure data manipulation language (DML). It is not data control language (DCL) or data definition language (DDL), therefore reading and modifying data is supported, but it is not possible to create or drop databases, collection data and indexes. Design goals of AQL were to support complex query patterns, different data models and that it would be client independent.
(ArangoDB Inc 2018. AQL Docs, 3).

Query example, which outputs all the actors in the movie “The Matrix” from the tutorial database.

```
db._query("FOR x IN ANY 'movies/TheMatrix' actsIn OPTIONS {bfs: true, uniqueVertices: 'global'}
RETURN x._id").toArray();
```

Output of the query is following.

```
[
  [
    "actors/Keanu",
    "actors/Hugo",
    "actors/Emil",
    "actors/Carrie",
    "actors/Laurence"
  ]
]
```

This is an example from the vendor’s provided tutorial database in the documentation of ArangoDB. (Michael Hackstein 2018. Cited 27.9.2018).

3.2 Storage Engines of ArangoDB

There are two storage engines to choose in ArangoDB called RocksDB and MMFiles, which have different usage purposes. It's not possible to mix these storage engines on installation, thus one must be chosen for a whole server or cluster.
(ArangoDB Inc 2018, 528).

RocksDB

Newer engine optimized bigger than main memory datasets. Keeps data in main memory but loads more data from disk if not cached already. Engine writes all index values also to disk, which means faster startup times. Performance can increase during usage, when engine fills its caches while running operations.
(ArangoDB Inc 2018, 528).

MMFiles

When datasets fit into main memory MMFiles is good choice as most index operations run in-memory speed. Longer startup times, because indexes are only built in main memory and needed to rebuild from disk when restarted or collection is accessed first time.
(ArangoDB Inc 2018, 528).

3.3 Scalability

ArangoDB is scalable both horizontally and vertically. Vertically scaling doesn't have any built-in limitations, for example, more threads are automatically used when there are more CPUs or cores on the server. According to developers of ArangoDB this is not optimal solution, because the costs grow faster, and beneficial resilience and dynamical capabilities are not present compared to horizontal scaling.
(ArangoDB Inc 2018, 40).

Horizontally scaling can be done with commodity hardware (or using cloud service), using many servers as a cluster. Performance, capacity and resilience are increased, and automatic fail-over can be achieved, with having multiple servers running ArangoDB instance. It is possible to build systems that scale capacity automatically depending of demand.
(ArangoDB Inc 2018, 40).

3.4 ArangoDB cluster

Collection data is organized in shards. This allows to use cluster of servers that constitute a single database, enabling to store more data and usually gaining better data throughput. Distribution of the data is done automatically.
(ArangoDB Inc 2018, 497).

ArangoDB must be run in cluster management system to have all cluster features: monitoring, scaling, automatic failover and replacement of failed nodes. At the moment, ArangoDB is dependent of Apache Mesos as cluster management system. Mesosphere DC/OS (the DataCenter Operating System), is distributed operating system based on the Apache Mesos distributed systems kernel, and it's highly recommended to be used when installing the cluster locally or into the cloud.
(ArangoDB Inc 2018, 309).

3.5 Performance

Several NoSQL-based databases were benchmarked in website article written by Claudius Weinberger on "DZone" and ArangoDB was tested with both storage engines. Overall results can be seen in figure 6 and ArangoDB performed in overall well compared to other tested databases, only exception is memory usage where it's not on top three.

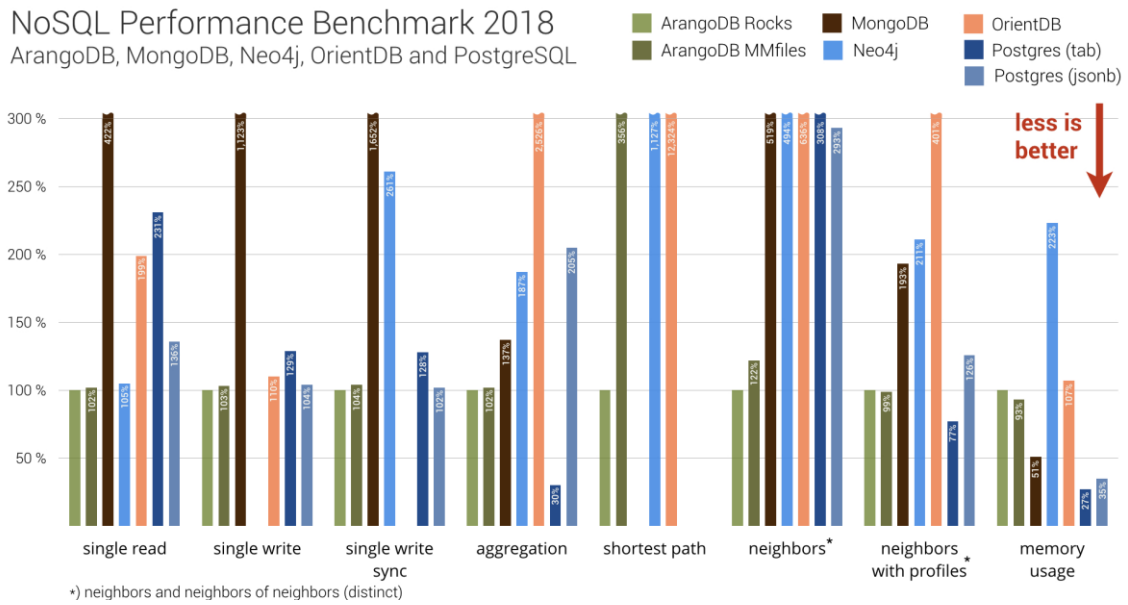


FIGURE 6. Performance benchmark (Claudius Weinberger 2018. Cited 17.9.2018).

Performance in cluster

ArangoDB developers claim in their published cluster performance white paper that with 640 vCPUs they sustained a write load of 1.1 million JSON documents per second (approximately 1GB of data per second) and that they deployed for the testing 80 node ArangoDB cluster using DC/OS in few minutes and with one short command.
(ArangoDB Inc 2015, 1).

Same paper consists performance test graphs (appendix 1). There are made 100% write test and 100% read test. Also 50% read and 50 write test has been made. In all test graphs the performance increases linearly when adding nodes to the cluster, without significant increase of latency. So, it can be said that ArangoDB cluster scales well.
(ArangoDB Inc 2015, 7-9).

4 INSTALLATION PREPERATIONS

There were few things to consider before starting installing the system. University provided the hardware and first thing to do was just to check it was working, what kind of specifications it had and those were sufficient to make the test server.

4.1 System Requirements

Test server runs with following hardware:
CPU (Central Processing Unit): Intel Core i7 3.20Ghz
RAM (Random Access Memory): 18GB
Storage: 512GB SSD (Solid State Drive) and 3TB HDD (Hard Disk Drive)

There are no official system requirements for ArangoDB. It has been run in Ubuntu server with only 1GB of ram and 2,2Ghz CPU.
(Tutorials Point 2018. Cited 27.9.2018).

It supports Linux, Windows and MacOS X. ArangoDB will only work on 64bit Windows.
(ArangoDB Inc 2018, 9).

4.2 Naming, storage and OS

IP address and DNS name

The test server is inside of the University LAN and was assigned a static IP address by university's IT department, which maintains the Network completely. There are no VPN connection possibilities to the University's LAN. Server's name was decided to be "jukebox", therefore after ArangoDB installation the Web Management Interface can be accessed with address "jukebox.fhstp.local" inside the LAN.

Storage

Before installation the SSD was partitioned to two equal sized partitions (approximately 256GB each), for one to be tested with Linux installation and other one with Windows. The 3TB HDD was reserved for backups.

Operating Systems

Mr. Ringot and I agreed to install Ubuntu Server 16.04 first to the test server and see how that will work, and afterwards Windows 10 Education N version was installed for testing. For licensing reasons Windows Server version was not used, as Education version was what IT-department provided for the project.

5 LINUX ENVIROMENT

Only brief documentation of installations, because of issues in remote access to the server, the environment was quickly chosen to be Windows one.

5.1 OS installation

Ubuntu Server 16.04 was installed to the first partition of the server by using USB-stick and with generally well-known step-by-step GUI installation, with following settings:

OS Language: English
Location: Austria
Keyboard: de_DE
Network: automatically from DHCP
hostname: jukebox
Home Directory Encryption: On
Timezone: CET – Vienna
Drive partitioning: Whole system to 256GB half of the 512GB SSD
Automatic Updates: On
Tasksel choices: Basic Ubuntu Server, Ubuntu Desktop

5.2 ArangoDB installation

Installation was done via Package Manager, when it will be installed as service automatically to start on boot.

repository key added first with commands.

```
curl -OL https://download.arangodb.com/arangodb33/xUbuntu_16.04/Release.key

sudo apt-key add - < Release.key
```

apt-get used then to do the installation.

```
echo 'deb https://download.arangodb.com/arangodb33/xUbuntu_16.04/ /'
| sudo tee /etc/apt/sources.list.d/arangodb.list

sudo apt-get install apt-transport-https

sudo apt-get update

sudo apt-get install arangodb3=3.3.14
```

5.3 Setting up remote access

TeamViewer was chosen to be used as a remote access software, because it works outside from University LAN, without public IP address and host machine can be used with normal graphical user interface.

installation can be made with package manager, with following command.

```
sudo apt install teamviewer_13.x.yyyy_amd64.deb
```

(x and y replaced with newest version numbers)

Conclusion of testing remote access

After testing the remote access several times, TeamViewer remote access from Windows client machine to Linux host machine worked randomly, sometimes the connection didn't work at all, sometimes with minor issues. For that reason, Mr. Ringot and I, decided quickly to set up Windows environment to be tested.

6 WINDOWS ENVIRONMENT

ArangoDB was installed on Windows 10 Education N version as a service, with remote access, system backups and remote database web management interface was configured to use SSL (Secure Sockets Layer) encryption. Automatic Windows updates and restarts were disabled.

6.1 OS installation

Operation system was installed from image, provided by university's IT-department, using USB (Universal Serial Bus) - stick to the second partition of the drive. Installation USB-stick was prepared with program called "Rufus"(http://rufus.akeo.ie/), which makes possible to create relatively quickly Windows installer USB-stick from image file.

Original language was German, but it is possible to change language to English in Education version of Windows, after installation is done. Some parts of the system stay in German language, like user group names ("Administrators" is "Administratoren" etc.).

6.2 Users

Server can only have local users as it is not possible to join to the university IT-department's maintained Windows-domain, when test server administration is not done by them. First account was named as "admin" and it has administrator user rights from start.

More users are created when needed, ideally every user has own account and no shared accounts are used. User rights are decided with Mr. Ringot when creating new users.

6.3 Setting up remote access

TeamViewer was installed to Windows environment using latest installer downloaded from vendors website. Installing TeamViewer is a simple task with the installation wizard. Almost all configurations were right on default. Only to make server accessible even after possible reboots, unattended remote access to the server was set up with a strong password. Random password was disabled. These configurations can be seen in the figure 7 below in the security tab of the TeamViewer options.

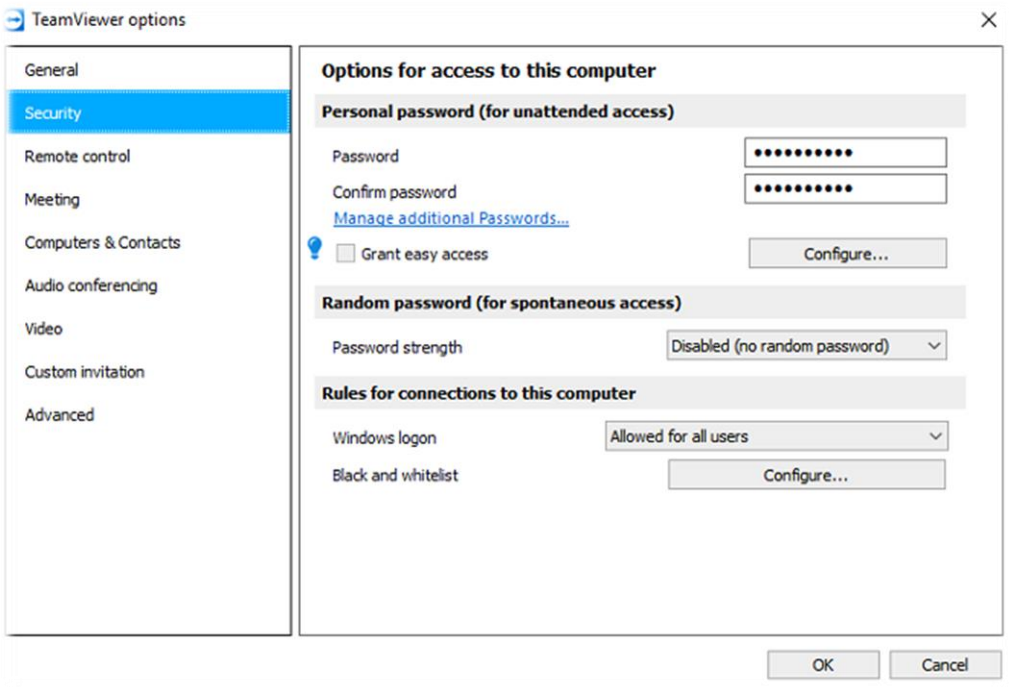


FIGURE 7. Security settings

6.4 ArangoDB installation

Latest installation packet was downloaded from the vendors website (https://www.arangodb.com/download-major/windows/). Running the executable file starts an installer wizard.

During the installation things which needed attention were making sure the ArangoDB is installed as a service and choosing wanted options as seen in figure 8. Later there is need of typing root user password and choosing storage engine. Storage engine was chosen to be “rocksdb” in this

project with Mr. Ringot, because it's the newer engine and should be able to handle very large datasets.

Otherwise pressing "Next >" button is enough and in the last screen "finish" button. Then the ArangoDB service starts automatically, on reboots also. Database default directory is "C:\ProgramData\ArangoDB\" and installation directory is "C:\ProgramFiles\ArangoDB-3.x.x\" (x.x part of the folder name is the installed version number).

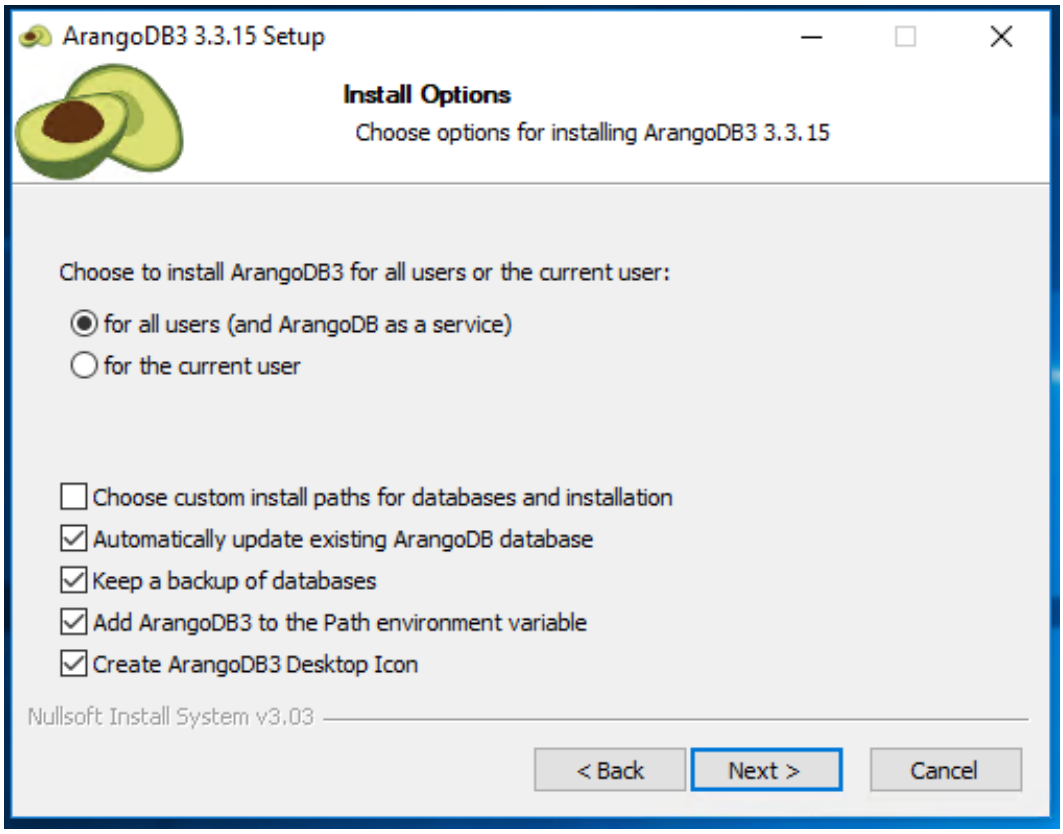


FIGURE 8. Installation configurations

6.5 Securing server and database

As the server is inside of the university LAN and physically accessible only to the workers of the university, no special server securing was done. The ArangoDB Web interface connection is in default not secured and for that reason university IT-department provided an ".pfx (Personal Information Exchange)" file to create certificates for SSL connections.

ArangoDB needs X.509 certificate to be in ".pem" (Privacy-enhanced Electronic Mail) file extension, which is Base64 encoded DER (Distinguished Encoding Rules) certificate. The .pem file needs to contain the private and public key together. After creating proper certificate from .pfx file by using "openssl" software, the configuration file needs to be updated with few lines. These lines can be seen in figure 9. First choose the protocol level (values 1 to 5) where 5 is most secure and 1 least. After that location of the certificate file. This file is located in default "C:\Program Files\ArangoDB3 3.x.x\etc\arangodb3\arangod.conf".

```
[ssl]
protocol = 5
keyfile = C:\fhcert\fhcert.pem

endpoint = ssl://0.0.0.0:443
```

FIGURE 9 Configuration file

Same configuration file includes the endpoint configuration. In test usage value "ssl://0.0.0.0:443" (as seen on figure 9) on endpoint makes accessing the ArangoDB Web Interface easy, only typing "jukebox.fhstp.local" in the browser address bar inside university LAN opens the login page.

6.6 Configuring update and backup procedures

Update

Using Windows 10 Education N version, instead of the any server version, means that the OS updates are downloaded and installed automatically on default and machine will reboot automatically after installation. This was not wanted behavior. One local group policy object configuration fixes the problem and disables the automatic reboots by waiting the user input to start installing updates.

The group policy object "Configure Automatic Updates" is found from Local Group Policy Editor under the location "Computer Configuration / Administrative Templates / Windows Components / Windows Update". As seen in figure 10, correct configuration is "3 – Auto download and notify to install" from under the "Configure automatic updating:".

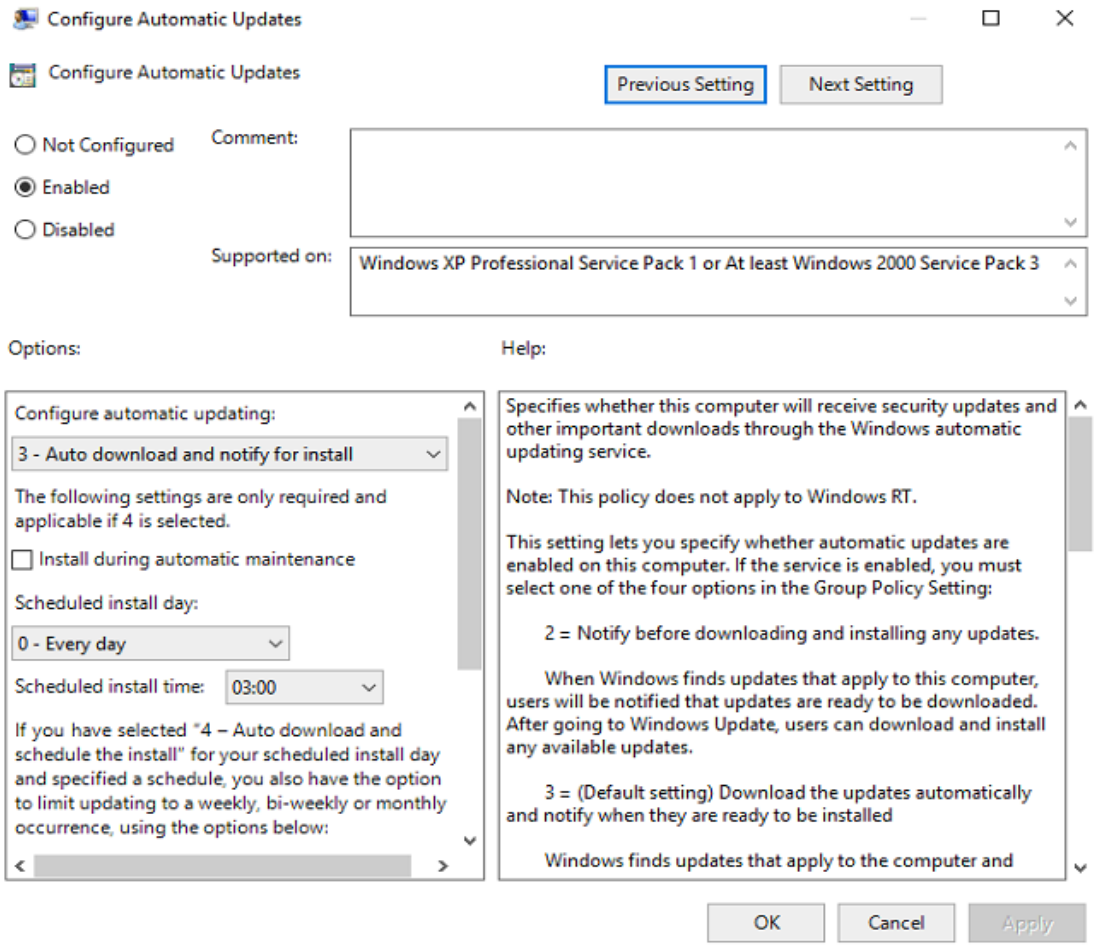


FIGURE 10. Local group policy object

Backup

Server backups and system image were configured from “control panel” option “Backup and Restore (Windows 7)”. Even when the name includes the “Windows 7” it works well in Windows 10. Backups were configured to be done every day during early mornings (5AM) to the 3TB HDD of the server as seen in the figure 11.

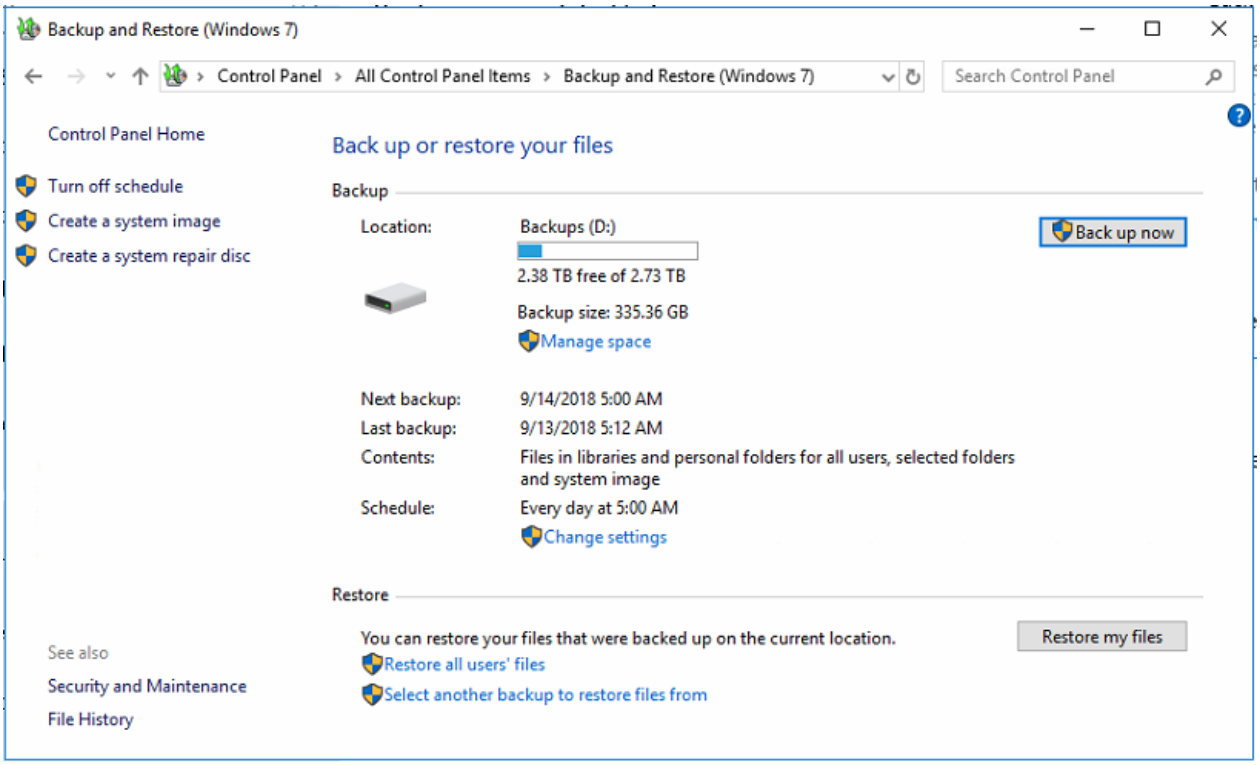


FIGURE 11. Backup settings

6.7 Data gathering

Data gathering to the database is done with Python-scripts running on the server. Mr. Ringot is responsible of creating, running and maintaining the scripts. Database tests have been done with gathering YouTube comments and making queries on that data. ArangoDB database has worked as intended and tests have been successful on Windows environment.

Also, in the need of extra backups outside of the scheduled one, Mr. Ringot handled backing up the database data with arangosh shell commands “arangodump” and if needed recovering data with the command “arangorestore”.

7 FUTURE DATABASE SCALING

When the data amounts would grow into big data scale, there are few options to scale the server to handle those volumes.

7.1 Single server set up

Depending of the size of the database increasing the CPU power or RAM size is more effective. As long as RAM is sufficient size for database indexes adding more doesn't increase performance but adding more CPUs to the system would make an impact. Having the database fully stored in SSD or in fast RAID setup naturally is better option than plain HDD as a storage medium.

Trying to improve vertically the performance is not the best option as written in third chapter before. It's gets easily costlier and doesn't have all the benefits of horizontal scaling with a cluster.

7.2 Server cluster

For production usage ArangoDB developer recommends using Apache Mesos as the cluster supervisor. DC/OS is the recommended way to install a cluster as it eases the process to install a Mesos cluster. Mesos has the concept of persistent volumes which makes it suitable for a database.

Preparing a DC/OS cluster is not discussed here as it could be an own thesis topic. After the DC/OS cluster is ready, the installation of ArangoDB through GUI is simple, because ArangoDB is an official partner of DC/OS and can be installed from package management. From DC/OS admin interface under "Universe" tab there is arangodb to choose from and by clicking "Install Package" ArangoDB will be deployed to the cluster.

From ArangoDB UI and under "Nodes" it's possible to scale cluster up and down by adding or removing nodes on the cluster. After changing the settings framework will handle scaling automatically. Scaling cluster up is simple as Mesos will just launch another task. Scaling down is different as the data must be moved first, so that will take a bit longer.
(ArangoDB Inc 2018, 309).

7.3 Cloud service options

Running the cluster in a cloud service can be a reasonable choice when there is no ready infrastructure for the local server cluster. Setting up ArangoDB on different cloud providers is very simple to do. Here are three examples.

Microsoft Azure

To deploy cluster on Azure needed is to install the azure-cli, download a bash script and execute it. Permission needed in Azure account are adding ssh-keypairs, creating instances and managing virtual networks.
(ArangoDB Inc 2018. ArangoDB in the Azure Cloud. Cited 17.9.2018).

Google Compute Engine

When deploying cluster on Google Compute Engine needed is to install the gcloud tool, download a bash script and execute it. Prerequisite is that the script needs to be available and configured in Google Project.

(Heiko Kernbach 2015. Create an ArangoDB cluster on Google Compute Engine with a single command. Cited 17.9.2018).

AWS (Amazon Web Services)

"To easy-deploy an ArangoDB cluster on AWS you just need to install the official awscli, download a single bash script and watch the tool take care of the rest for you. Your aws account needs permission for creating instances, adding ssh-keypairs and managing security groups."

(Heiko Kernbach 2015. Create an ArangoDB cluster on Amazon Web Services (AWS). Cited 17.9.2018).

REFERENCES

ArangoDB Inc 2015. Scaling ArangoDB to Gigabyte/s
bandwidth on Mesosphere, an ArangoDB White Paper. Cited 17.9.2018
https://www.arangodb.com/arangodb-white-papers/whitepaper-arangodb-cluster-benchmark/

ArangoDB Inc 2016. What is a multi-model database and why use it? An ArangoDB White Paper. Cited 17.9.2018.
https://www.arangodb.com/arangodb-white-papers/white-paper-multi-model-database/.

ArangoDB Inc 2018. General ArangoDB Manual. Cited 17.9.2018.
https://download.arangodb.com/arangodb33/doc/ArangoDB_Manual_3.3.16.pdf.

ArangoDB Inc 2018. ArangoDB in the Azure Cloud. Cited 17.9.2018.
https://www.arangodb.com/azure/.

ArangoDB Inc 2018. AQL Docs. Cited 27.9.2018.
https://download.arangodb.com/arangodb33/doc/ArangoDB_AQL_3.3.16.pdf.

Ben Smith 2015. Beginning JSON. Apress.

Claudius Weinberger 2018. NoSQL Performance Benchmark 2018: MongoDB, PostgreSQL, OrientDB, Neo4j and ArangoDB. Cited 17.9.2018.
https://dzone.com/articles/nosql-performance-benchmark-2018-mongodb-postgresq.

Éric Dusablon 2017. Cloud - NDT – Presentation. Cited 27.9.2018
https://www.slideshare.net/ricDusablon/cloud-ndt-presentation.

Graeme Malcolm Key-Value Data Stores. Edx.org course: Introduction to Big Data. Cited 17.9.2018.
https://edx-video.net/MSXBDOXX2017-V004600_DTH.mp4

Graeme Malcolm Relational Databases. Edx.org course: Introduction to Big Data. Cited 17.9.2018. https://edx-video.net/MSXBDOXX2017-V001900_DTH.mp4

Heiko Kernbach 2015. Create an ArangoDB cluster on Google Compute Engine with a single command. Cited 17.9.2018.
https://www.arangodb.com/2015/04/gce-cluster/

Heiko Kernbach 2015. Create an ArangoDB cluster on Amazon Web Services (AWS). Cited 17.9.2018
https://www.arangodb.com/2015/05/aws-cluster/

json.org 2018. JSON Example. Cited 17.9.2018.
https://json.org/example.html

Michael Hackstein, AQL Example Queries on an Actors and Movies Database 2018. Cited 27.9.2018.
https://docs.arangodb.com/3.2/Cookbook/Graph/ExampleActorsAndMovies.html.

Sami Tavasti 2018. Openstack-pilvipalvelualustan konfigurointi. Cited 17.9.2018.
http://urn.fi/URN:NBN:fi:amk-201805229381.

St. Pölten University of Applied Sciences 2018. SAMBA Smart Data for Music Business Administration. Cited 17.9.2018.
https://samba.fhstp.ac.at/.

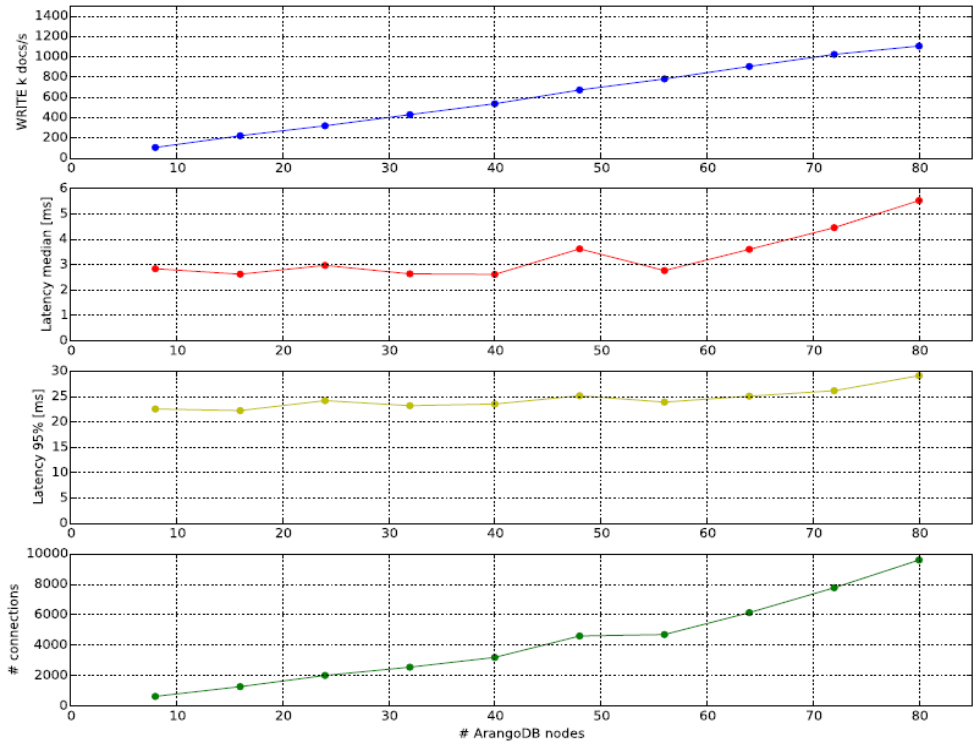
Syed Muhammad Fahad Akhtar 2018. Big Data Architect's Handbook. Packt Publishing.

Tutorials Point 2018. ArangoDB Tutorial. Cited 27.9.2018.
https://www.tutorialspoint.com/arangodb/arangodb_system_requirements.htm.

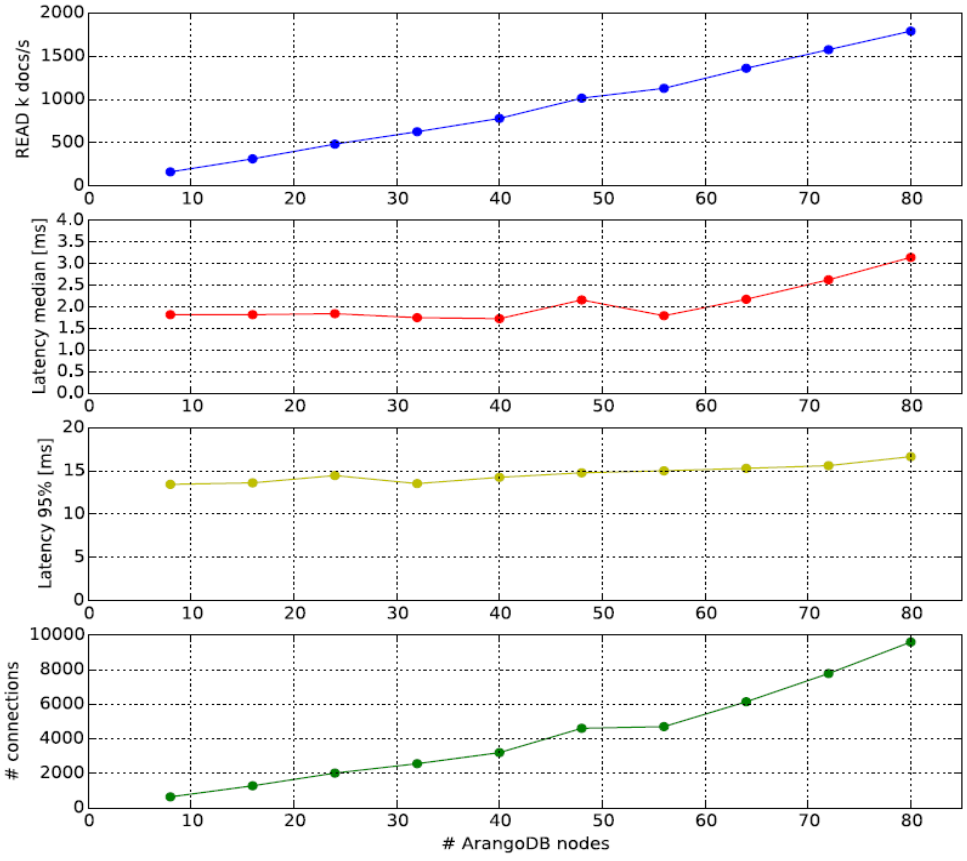
V. Naresh Kumar; Prashant Shindgikar 2018. Modern Big Data Processing with Hadoop. Packt Publishing.

Vijay Shankar Upreti 2016. Oracle Solaris and Veritas Cluster: An Easy-build Guide. Apress.

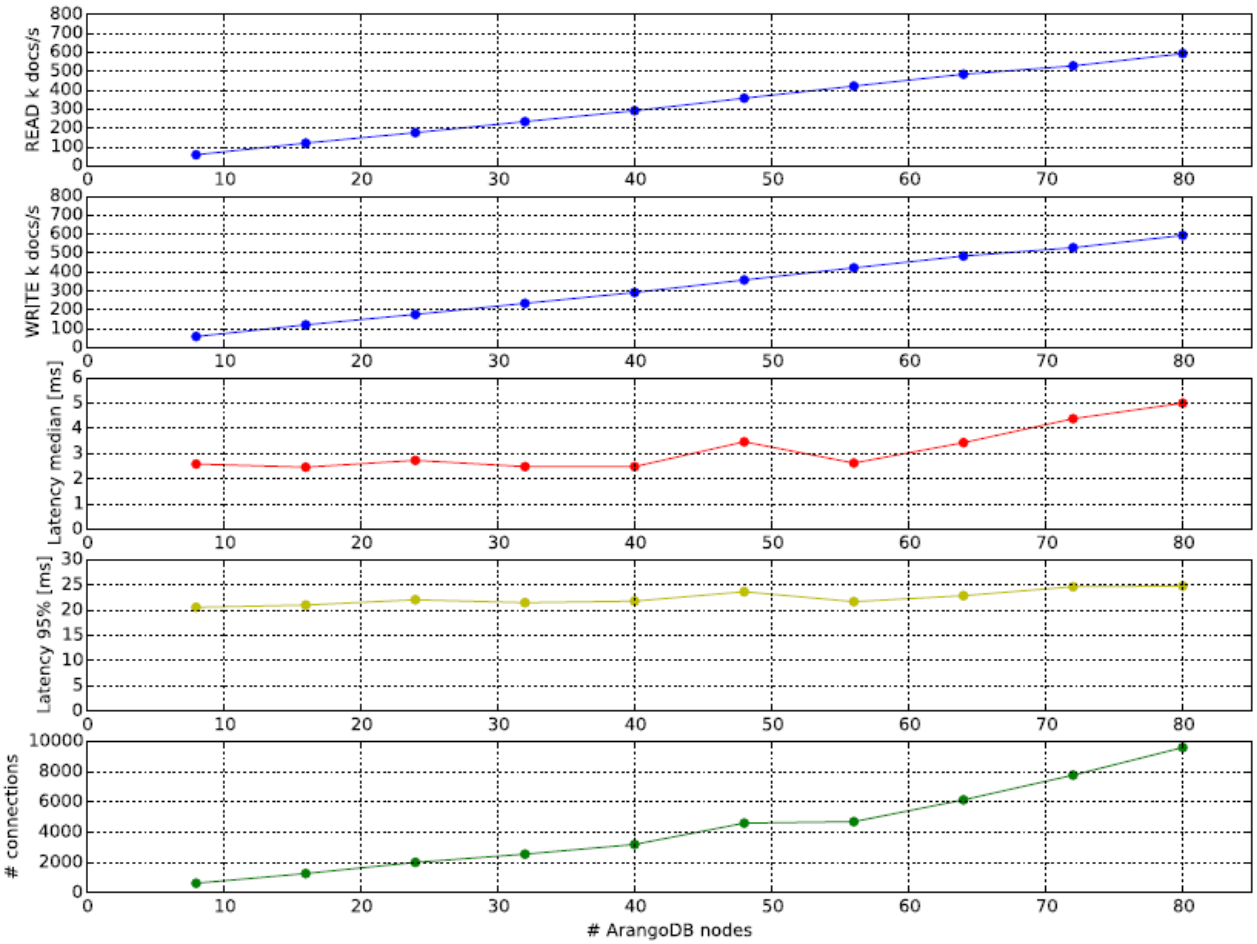
Zacharias Voulgaris PhD 2017. Data Science: Mindset, Methodologies, and Misconceptions. Technics .



100% write graphs



100% read graphs



50% write and 50% read graphs